Engineering Applications of Artificial Intelligence 111 (2022) 104775

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

A surrogate-assisted Jaya algorithm based on optimal directional guidance and historical learning mechanism

Fuqing Zhao [a,*], Hui Zhang [a], Ling Wang [b], Ru Ma [a], Tianpeng Xu [a], Ningning Zhu [a], Jonrinaldi [c]

[a] School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China
[b] Department of Automation, Tsinghua University, Beijing, 10084, China
[c] Department of Industrial Engineering, Universitas Andalas, Padang, 25163, Indonesia

ARTICLE INFO

Keywords:
Jaya algorithm
Surrogate-assisted mechanism
Co-evolutionary mechanism
Optimal directional guidance
Historical learning mechanism

ABSTRACT

An improved Jaya algorithm named surrogate-assisted Jaya algorithm based on optimal directional guidance and historical learning mechanism (SDH-Jaya) is proposed in this study to address the continuous optimization problems. In the SDH-Jaya, a surrogate-assisted model combined with the polynomial model and radial basis model built by the individual with real fitness is introduced to decrease the expensive computational simulations and accelerate the convergence speed. Two co-evolutionary mechanisms, which are named assisted co-evolutionary mechanism and self-learning co-evolutionary mechanism, are proposed to optimize the surrogate model and evolutionary population. Search directions and steps of the SDH-Jaya are adjusted adaptively by the differential vector resulting from the best solution and worst solution in the candidates at each generation. The historical population stored in an archive is selected randomly to provide new search areas for improving the diversity of the population during the evolution process of the SDH-Jaya. The performance of SDH-Jaya is tested on CEC2017 benchmark problems. The experimental results reveal that the effectiveness of the SDH-Jaya algorithm outperforms the classical Jaya algorithm, its variants, and state-of-the-art algorithms in terms of the quality of solution and execution time.

1. Introduction

Real value optimization problems widely exist in engineering design and scientific research (Liang et al., 2020; Zhang et al., 2021a). In real life, real value optimization problems often have the characteristics of nonlinearity, non differentiability, discontinuity, and multipolarity, so the traditional optimization algorithms are not applicable, such as gradient descent algorithm (Zhang et al., 2021b). Unlike traditional optimization algorithms, evolutionary algorithms have the advantages of global search ability, not limiting to any objective function form, black box problem optimization ability, parallelization, and so on. The above advantages of the evolutionary algorithm make it widely used and have become one of the most popular optimization technologies.

The classical evolutionary algorithms are proposed and researched by scholars. The differential evolution (DE) (Bilal Pant et al., 2020) is composed of mutation, crossover, and selection operations to improve the global search capability of the algorithm. The speed and position of each individual are utilized to control the evolutionary step and direction to close to the most preeminent individual in the particle swarm optimization (PSO) (Tang et al., 2015). The trade-off of exploitation and exploration in the biogeography-based optimization (BBO) algorithm (Bhattacharya and Chattopadhyay, 2010; Deng et al., 2019; Simon, 2008) is adjusted by biological migration and mutation operations. The information of the nearest neighbor is learned by the individuals to improve the optimization ability in the whale optimization algorithm (WOA) (Long et al., 2020). The desirable information from the optimal solution is spread in the water wave optimization (WWO) (Zhao et al., 2019) by propagation, refraction, breaking operations during the search process. The k-means method is utilized to cluster the population, and the optimal individual in one class is replaced randomly by the other optimal solution to ensure the entire search space searched in the brainstorm optimization (BSO) algorithm (Zhao et al., 2021a). A hierarchical knowledge guided backtracking search algorithm (HKBSA) (Zhao et al., 2021b) is presented according to multi-population strategy and multi-strategy mutation to discover the optimal solution. However, evolutionary algorithms still face challenges in practical applications, which include challenges from computationally expensive optimization problems.

Computationally expensive optimization problems are often characterized by a high degree of non-linearity and the absence of an explicit objective function. The natural selection based on individual

* Corresponding author.
E-mail addresses: Fzhao2000@hotmail.com (F. Zhao), 1601199793@qq.com (H. Zhang), wangling@tsinghua.edu.cn (L. Wang), 312783672@qq.com (R. Ma), xutp@lut.edu.cn (T. Xu), 307516638@qq.com (N. Zhu), jonrinaldi@eng.unand.ac.id (Jonrinaldi).

https://doi.org/10.1016/j.engappai.2022.104775
Received 11 November 2021; Received in revised form 27 January 2022; Accepted 16 February 2022
Available online xxxx
0952-1976/

fitness value makes the evolutionary algorithm no longer need gradient information from the problem for evolutionary algorithms to efficiently solve optimization problems with characteristics such as black boxes, non-linearity, non-differentiability, and discontinuity. For this reason, evolutionary algorithms are popular in application areas which involve computationally expensive problems. In recent years, surrogate-assisted evolutionary algorithms (SAEAs) are generally utilized to optimize engineering optimization problems that have computationally expensive simulations (Cai et al., 2020; Wang et al., 2020; Zhao et al., 2020b). SAEAs overcome the deficiencies of traditional mathematical methods in settling the engineering optimization problems without a precise mathematical model. As compared to traditional evolutionary algorithms, SAEAs reduce the problems of high computational simulation cost and are long time-consuming in solving optimization problems. The main purpose of SAEAs is to use computationally inexpensive surrogate models to evaluate individuals and replace real fitness assessments. Thus, the better solution with limited computational resources is allowed to be found by evolutionary algorithms.

The Jaya algorithm, which is an algorithm-specific parameter-less evolutionary algorithm, is proposed by Venkata Rao (2016) to address engineering optimization problems. As one of the new swarm intelligence optimization algorithms, the Jaya algorithm has the characteristics of easy implementation and few specific parameters. The operation mechanism of the Jaya algorithm is to guide each individual close to the best individual and guide each individual away from the worst individual. The offspring population is produced after individual renewal. The fitness function values of parent and offspring individuals are compared, and the individuals with better fitness values in parent and offspring are selected to form the next generation population. In the Jaya algorithm, the offspring will be close to the best individual of each generation and away from the worst individual of each generation. As a part of the evolutionary algorithm, the Jaya algorithm requires a lot of evaluation time in solving optimization problems. In the current research literature, there is no research on how to decrease the evaluation time used by the Jaya algorithm in solving optimization problems, and this research direction is vital in practical engineering applications.

A surrogate model combined polynomial model and radial basis model is studied and a new variant of Jaya algorithm (SDH-Jaya) is proposed in this study. A surrogate-assisted model, which is constructed by dispersed points, uncertain points, and the optimal point, is introduced to save computational resources in the SDH-Jaya. Besides, two co-evolutionary mechanisms are proposed to improve the accuracy of the surrogate model and accelerate the evolution of the population. The optimal directional guidance strategy is introduced to adjust adaptively the search direction and search step. The historical learning mechanism is utilized to balance the exploitation and exploration ability of the SDH-Jaya. The main contributions of this study are summarized as follows.

- Two co-evolutionary mechanisms are proposed to enhance the accuracy of the SDH-Jaya. In the assisted co-evolutionary mechanism, the evolution of the population is assisted by the surrogate model and an optimal individual is returned to provide a new search area. The reward and punishment rules are designed based on the calculation speed, fitness evaluations, and the performance of the Jaya algorithm in the self-learning co-evolutionary mechanism.
- A new evolutionary strategy based on optimal directional guidance and historical learning is provided to improve the performance of the classical Jaya algorithm. The difference vectors between the best solution and worst solution are proposed to guide the mutation of candidates. The historical population stored in an archive is utilized randomly to improve the diversity of the population.

The remainder of this work is organized as follows. Literature about the Jaya algorithm and surrogate model is reviewed in Section 2. The proposed SDH-Jaya is introduced in Section 3. The experimental result analysis and discussion are presented in Section 4 and Section 5. Continuous engineering optimization problems and scheduling problems are utilized to test the performance of the SDH-Jaya in Section 6. Finally, conclusions are summarized and future work is suggested in Section 7.

2. Literature review

2.1. Jaya algorithm

The Jaya algorithm is an algorithm-specific parameter-less evolutionary algorithm. Apart from population size and termination conditions, no algorithm-specific parameters are set in advance. The main idea of the Jaya algorithm is that all individuals keep moving closer to the optimal individual and away from the worst individual in each generation. Research on the Jaya algorithm has concentrated on two main areas, namely research on the improvement of the algorithm and research on the application of the Jaya algorithm.

In the improvement of the algorithm, a self-adaptive multi-population mechanism is introduced in the self-adaptive multi-population based Jaya (SAMP-Jaya) algorithm (Venkata Rao and Saroj, 2017), which is adjusted by the number of sub-populations adaptively based on the landscape to balance the exploration and exploitation during the search process. The population size is adjusted adaptively to reduce the effect of population size on optimization accuracy in the self-adaptive Jaya (SA-Jaya) algorithm (Rao and More, 2017). A quasi-oppositional learning strategy is proposed in the quasi-oppositional Jaya (QO-Jaya) algorithm (Warid et al., 2018), which expands the search area by the approximate opposite population. The chaos theory is utilized in the chaotic-Jaya (C-Jaya) (Ravipudi and Neebha, 2018) algorithm to replace the random numbers in the mutational formula of the Jaya algorithm to prevent the algorithm from premature convergence and falling into the local optima. According to the characteristics of search space, the population is adaptively divided into multiple sub-populations, and the elitist strategy is utilized in the elitist-based self-adaptive multi-population Jaya (SAMPE-Jaya) (Rao and Saroj, 2019) algorithm. The current research on Jaya optimization algorithms and evolutionary algorithms simply introduces various evolutionary strategies, learning mechanisms, and other improved algorithms, without combining traditional optimization theory.

The Jaya algorithm has also been very productive in solving various engineering optimization problems. In solving the scheduling problem, the efficient Jaya algorithm (Mishra et al., 2020) solves the optimization of the maximum completion time and the total delay time under the lead time constraint multi-objective permutation flow shop scheduling problem. To improve the global search ability of Jaya, an enhanced Jaya (E-Jaya) algorithm (Zhang et al., 2021b) is proposed for global optimization. An improved Jaya algorithm is proposed (Jqla et al., 2020), while several problem-specific local search operators are proposed. The discrete Jaya (D-Jaya) (Gao et al., 2019) algorithm is used to solve the flexible job shop rescheduling problem. To improve the performance of the D-Jaya algorithm, five goal-oriented local search operators are introduced. A hybrid adaptive differential evolution and Jaya algorithm (aDE-Jaya) (Son et al., 2020) is proposed for the Bouc–Wen hysteresis model of piezoelectric actuators. The Jaya algorithm, which is a part of evolutionary algorithms, requires a large number of evaluations to solve the optimization problem. The current research literature has not studied how to reduce the evaluation times used by Jaya algorithm in solving optimization problems. This is a very important research direction in practical engineering application. For interested readers, we have found some literature and books on the Jaya algorithm, which are published on GitHub. The link is https://github.com/zzzhhh-320/literature.

2.2. Surrogate model

In SAEAs, the optimization problem is modeled by different surrogate models with the solutions evaluated by the fitness function. The common surrogate models, which include polynomial response surface (PRS) (Zhou et al., 2005), radial basis function (RBF) (Amouzgar et al., 2018), support vector regression (SVR) (Chen et al., 2021), and kriging (Huang et al., 2006; Liu et al., 2014) are introduced to assist the search process of evolutionary algorithm. One of the surrogate models is selected and analyzed for directing the evolution of algorithms in numerous studies (Cai et al., 2019; Chugh et al., 2018; Emmerich et al., 2006). A sequential approximate optimization (SAO) procedure based on the radial basis function (RBF) network is advised (Kitayama et al., 2011). A multi-fidelity meta-model based on radial basis function is recommended to approach the expensive black-box problem (Cai et al., 2017b). Besides, collaborative mechanisms between multiple models have been studied by various scholars. A hybrid meta-model, which is combined with the cut high dimensional model representation (Cut-HDMR) method, Co-kriging, and kriging, is proposed to approximate black-box problems (Cai et al., 2017a). A collaborative framework of multi-surrogate models is proposed (Goel et al., 2007) that approximations are weighted based on error statistics of each surrogate model to reduce the misuse of surrogate models. Furthermore, the use of the surrogate model is combined with exact fitness evaluations to reduce the misleading of the surrogate model (Jin, 2011).

There are two strategies to connect fitness evaluations with the surrogate model (Jin et al., 2002, 2001). One strategy is named individual-based evolution control. In the search process, certain individuals are assessed by the fitness function, and approximate fitness values are provided by the surrogate model for others. For individual-based evolution control, the core problem is which individuals are selected to assess by the fitness function. The uncertainty of the individual is calculated according to the distance of each individual. The individuals with high uncertainty are selected and evaluated by the exact fitness function (Johnson et al., 1990). The committee-based active learning for surrogate-assisted particle swarm optimization is proposed (Wang et al., 2017), in which the most uncertain point and the best point of the models are selected and evaluated by the fitness function. The other strategy is called generation-based evolution control. The initial population assessed by fitness function is utilized to construct a surrogate model (Ratle, 1998). The fitness of candidates is provided by the surrogate model until the optimal solution is not altered. Afterwards, the remaining candidates are evaluated by fitness functions and further reconstructing the surrogate model.

However, the cooperative coevolution of evolutionary algorithms and surrogate models has not been considered in most literature. Cooperative coevolution speeds up the convergence speed of the algorithm and reduces the time-consuming and costly problems. The feasible solutions searched by the evolutionary algorithm contribute to improving the accuracy of the surrogate model and enable the surrogate model to characterize the engineering optimization problem more accurately. The cooperative coevolution of evolutionary algorithms and surrogate models is significant to improve the accuracy of the surrogate model and speed up the convergence speed of the algorithm. Moreover, cooperative coevolution saves cost and improves resource utilization rate when solving practical engineering optimization problems.

3. SDH-Jaya algorithm

3.1. Jaya algorithm

Assuming that $f(x)$ is the function to be optimized. The iteration is g (i.e. $g = 1, 2, \ldots, G$), the population size is p (i.e. $p = 1, 2, \ldots, P$), the number of design variables is d (i.e. $j = 1, 2, \ldots, D$). First of all, p initial solutions are initialized randomly in $[lower_d, upper_d]$. $lower_d$ and $upper_d$ are the lower bound and upper bound of the design variable d, respectively. Then the Eq. (1) is utilized to generate offspring individuals according to parent individuals. In Eq. (1), the term "$r_1(x_{g,best,d} - |x_{g,p,d}|)$" indicates that the offspring individual approaches the best individual, the term "$-r_2(x_{g,worst,d} - |x_{g,p,d}|)$" shows that the offspring individual escapes from the worst individual. The offspring individual is accepted to the next iteration when the offspring individual is better than the parent individual. The pseudocode of the Jaya algorithm is shown in Algorithm 1.

$$x_{g+1,p,d} = x_{g,p,d} + r_1(x_{g,best,d} - |x_{g,p,d}|) - r_2(x_{g,worst,d} - |x_{g,p,d}|) \tag{1}$$

where $x_{g,p,d}$ is the individual of dth variable for pth candidate in gth iteration. $f(x)$ is the function to be optimized. $x_{g+1,p,d}$ is the offspring individual of the parent individual $x_{g,p,d}$. $x_{g,best,d}$ and $x_{g,worst,d}$ represent the best individual and the worst individual of the population in gth iteration, respectively. r_1, r_2 are random numbers in the range of 0 and 1.

3.2. Framework of SDH-Jaya algorithm

According to the operation mechanism of the SDH-Jaya algorithm, the pseudocode and flow chart are shown in Algorithm 2 and Fig. 1.

3.2.1. An improved surrogate-assisted Jaya algorithm

A simple meta-heuristic algorithmic framework which is based on a surrogate model without using model error to guide the construction of the model, easily leads the algorithm into a locally optimal solution due to model inaccuracy. The following improvements have been made to address this shortcoming as is shown in Eq. (2). The exact fitness function is replaced by a surrogate model. Continuous iterations and evaluations are required to acquire an optimal solution in evolutionary algorithms for complex optimization problems. Surrogate-assisted evolutionary algorithms are effective in reducing fitness evaluations and simplifying the complexity of problems.

$$y = \hat{f}(x) + \varepsilon(x) \tag{2}$$

where $\hat{f}(x)$ is the surrogate model, $\varepsilon(x)$ is the error of the surrogate model.

Three types of points are sampled to establish a precise surrogate model. The process of building the surrogate model is shown in Fig. 2. Firstly, taking two-dimensional space as an example, individuals with exact fitness are all stored in database $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. The first kind of points which are sampled are the dispersed points among the database. According to the diversity metric for dispersed points introduced (Solow et al., 1993), $5d$ points are selected to constitute database D_t. The characteristics of search space are described by the $5d$ dispersed points and the sampling process is shown as follows in Algorithm 3.

Secondly, the most uncertain point (x_u, y_u) is sampled. D_t is unitized to establish a polynomial response surface model $\hat{f}_p(x)$ and a radial basis function model $\hat{f}_r(x)$, respectively. e_p is the error of $\hat{f}_p(x)$ and e_r is the error of $\hat{f}_r(x)$. The gap between $\hat{f}_p(x)$ and $\hat{f}_r(x)$ is the biggest among all individuals is evaluated fitness function to improve the accuracy of the surrogate model. The method to choose the most uncertain point is shown in Eqs. (3) and (4). The most uncertain point which revealed the discordance of $\hat{f}_p(x)\, and\, \hat{f}_r(x)$ is added to D_t to optimize the surrogate model. D_t is updated to $D_t \cup (x_u, y_u)$.

$$U_{ens}(x_{new}) = max(\hat{f}_p(x_{new}) - \hat{f}_r(x_{new})) \tag{3}$$

$$x_u = arg \max_{x_{new}} u_{ens}(x_{new}) \tag{4}$$

where x_{new} is the new individual, U_{ens} is the maximum of $\hat{f}_p(x_{new}) - \hat{f}_r(x_{new})$, x_u is the index of U_{ens}.

Finally, the third kind of points, the most excellent point (x_b, y_b), is introduced to describe local details of the exact fitness functions by the surrogate model which combines $\hat{f}_p(x)$ and $\hat{f}_r(x)$ as shown in

Algorithm 1: The Jaya algorithm

Input: P, the termination condition
The initial population is generated randomly and the fitness of each individual is calculated
While the termination condition is not met
  For p in 1 to P
    The offspring individual x_{i+1} of x_i is produced according to Eq. (1)
    Calculate the fitness $f_{x_{i+1}}$ of x_{i+1}
    If $f_{x_{i+1}} < f_{x_i}$
      $x_{i+1} = x_{i+1}$
    Else
      $x_{i+1} = x_i$
    End if
  End for
Output: the optimal individual and fitness

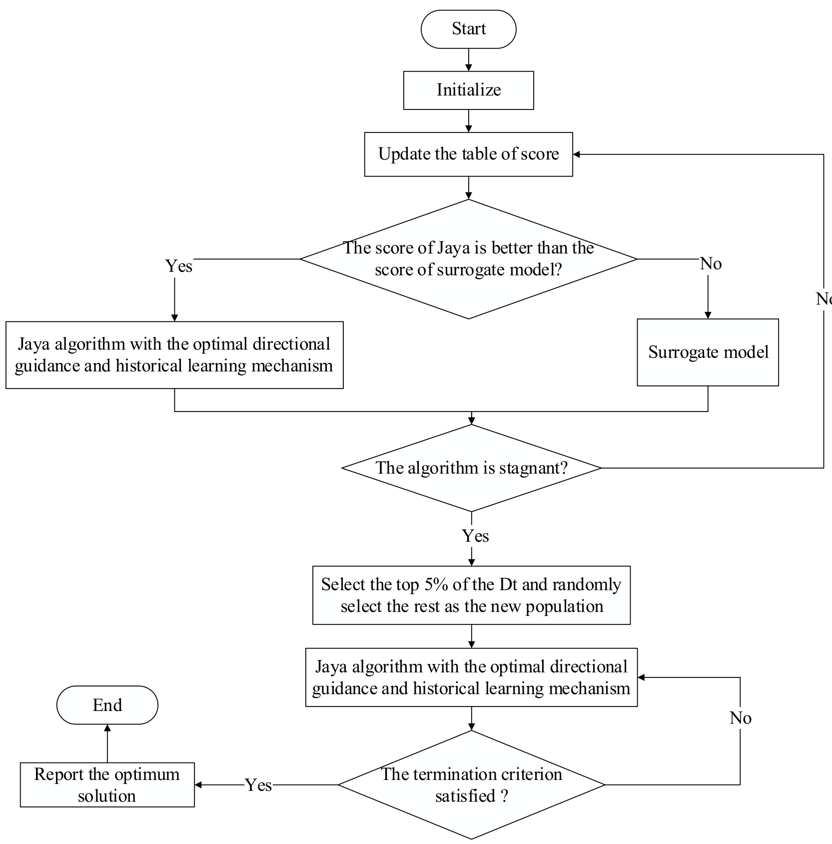


Fig. 1. The flow chart of SDH-Jaya.

Eq. (5). The surrogate model $\hat{f}(x)$ is weighted according to the error. To decrease $\varepsilon(x)$, ω_p and ω_r is introduced to combine $\widehat{f_p}(x)$ and $\hat{f}_r(x)$. The calculation method of ω_p and ω_r are shown in Eq. (6). After one cycle of surrogate model management and optimization, all three kinds of sampling points are stored in D, D is updated to $D \cup \left\{ \left(x_u, y_u\right), \left(x_b, y_b\right) \right\}$.

$$\hat{f}(x) = w_p \widehat{f_p}(x) + w_r \hat{f}_r(x) \tag{5}$$

$$w_i = 0.5 - (e_i / (2\left(e_p + e_r\right))), i = p, r \tag{6}$$

3.2.2. Two co-evolutionary mechanisms

Two co-evolutionary mechanisms between the Jaya algorithm and surrogate model are introduced, which include the self-learning evolutionary mechanism and the assisted co-evolutionary mechanism. The primary framework is the Jaya algorithm for the assisted co-evolutionary mechanism. Unlike the Jaya algorithm, the part of exact fitness evaluations is replaced by the surrogate model and the output is replaced by an uncertain point or an excellent point. The population mutates to produce new candidates, and the surrogate model is utilized to evaluate the performance of each candidate in Fig. 3. To find an uncertain point $\left(x_u, y_u\right)$, the population just updates one generation to save computing resources. The maximum number of stagnations in the surrogate (SMs) model is introduced as a condition of termination during finding the desirable point. The best individual $\left(x_{Gb}, y_{Gb}\right)$ in the current population is produced. When y_{Gb} does not change in SMs consecutive times, the algorithm terminates and outputs the optimal point $\left(x_b, y_b\right)$. A criterion is designed to decide which operation

Algorithm 2: The SDH-Jaya algorithm

Input: p, d, the maximal fitness evaluations ($MFEs$), the maximum of algorithmic stagnations (Ms), the maximum of stagnations in the surrogate model (SMs) and the score table
While the termination condition is not met
  If $Score_{Jaya} > Score_{surrogate\ assisted}$
    Optimization using directional guidance and historical learning mechanism
  Else
    Optimization using the surrogate-assisted mechanism
  End
  Update the value of the score table by the self-learning co-evolutionary strategy
  Select the top 5% D_t and randomly select the rest of $p - 5\%p$ solutions as new population
  Optimization using DH-Jaya algorithm
End for
Output the optimal solution

Algorithm 3: The sampling process

Initialize D_t to be empty and put the point with the best fitness in D to D_t
While the termination condition is not met **do**
  For i in 1 to i
    Calculate the distance between each point in D_t and D
    Record the point i
    dis_i into a matrix $Dis = \{(1, dis_1), (2, dis_2), \ldots, (i, dis_i)\}$
    Move the largest distance from D to D_t, according to Dis
  End for

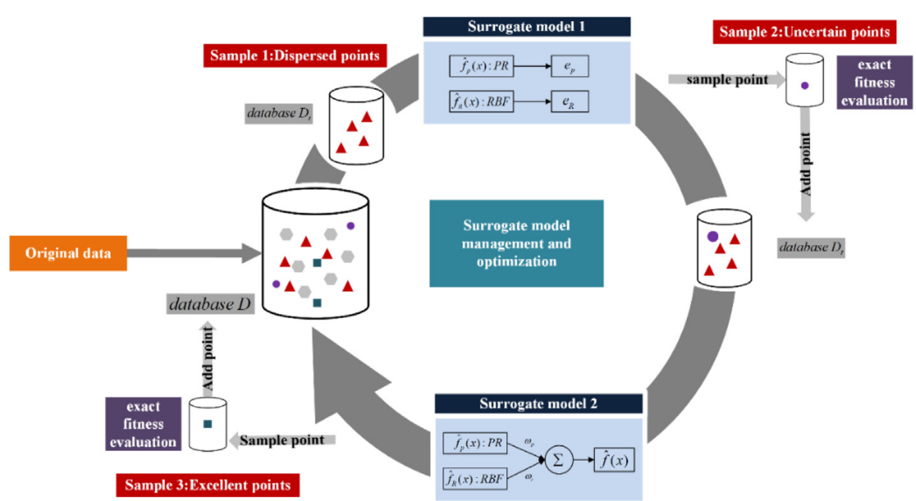


Fig. 2. Surrogate model management and optimization.

is chosen in the management layer in the self-learning evolutionary mechanism. The rewards and punishments rules are proposed as shown in Table 1. In the Jaya algorithm, P exact fitness evaluations are wasted in each generation. Nevertheless, only two exact fitness evaluations are consumed in $SMs + 1$ generations evaluated by the surrogate model. During the modeling process, the dispersion of $FEs(5d - 1)$ individuals is evaluated, and P exact fitness evaluations are evaluated in the Jaya algorithm. Therefore, the criteria of speed and evaluation in the Jaya algorithm are P and $P(SMs+1)$, respectively. The criteria of speed and evaluation in the surrogate model are $FEs(5d - 1)$ and 2 respectively. A penalty term is obtained by normalizing the speed and evaluation indexes. The reward item is set to 0.5.

The operation layer, which includes the surrogate model system and the Jaya algorithm system, provides alternative operations illustrated in the action-space (A) and state-space (a) in Fig. 3. Whether a better individual is found at the end of each operation is set as the evaluation indicator to punish or reward the operation. The reward is given according to the reward table when the operation finds a better individual. The punishment is given according to the punishment table when the operation does not find a better individual. The score

Action-space (A): ⟨the surrogate model, the Jaya algorithm⟩

State-space (a): ⟨find a better individual, not find a better individual⟩

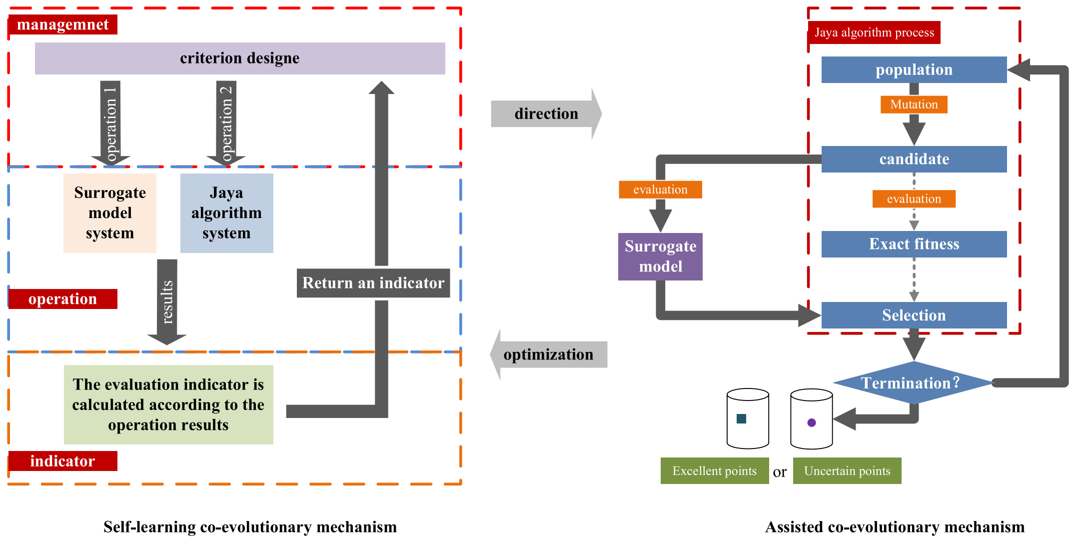


Fig. 3. Two co-evolutionary mechanisms.

Table 1
Rewards and punishments rules and initial score.

	Jaya algorithm system	Surrogate model system
Speed	P	$FEs(5d-1)$
Evaluation	$P(SMs+1)$	2
Punishment	$\frac{p}{p+FEs(5d-1)} + \frac{p(SMs+1)}{2+pMs}$	$\frac{FEs(5d-1)}{p+FEs(5d-1)} + \frac{2}{2+p(SMs+1)}$
Reward	0.5	0.5
Initial score	1	0

table is updated and returned to the management layer for guiding the next evolutionary according to the result of the operation. The self-learning co-evolutionary mechanism directs the use of the assisted co-evolutionary mechanism. The optimal individual is produced by the assisted co-evolutionary mechanism with less fitness evaluations to accelerate the convergence and to optimize the evolutionary population.

3.2.3. The optimal directional guidance and historical learning mechanism

The diversity of the population declines rapidly during the search process. The item $r_1(x_{g,best,d} - |x_{g,p,d}|)$ and $r_2(x_{g,worst,d} - |x_{g,p,d}|)$ are not changed or changed randomly without guidance to lead the premature convergence of population on complex optimization problems. Therefore, the optimal directional guidance and the historical mechanism are proposed in this section. A new mutational formula is introduced to replace the basal mutational formula of the Jaya algorithm, and the historical population is preserved in an archive to improve the diversity of the population when the algorithm stagnates. Eq. (7) is utilized to replace the basal mutational formula of the Jaya algorithm for taking advantage of the orientation and step of the differential vector between the best individual $x_{g,best,d}$ and the worst individual $x_{g,worst,d}$ which is the optimal direction of evolution in the current generation.

$$x_{g+1,p,d} = x_{g,p,d} + r(x_{g,best,d} - x_{g,worst,d}) \tag{7}$$

where $x_{g+1,p,d}$ is a candidate produced by $x_{g,p,d}$. r is a random number in the range of 0 and 1 to enhance the diversity of the population.

The major idea is that the candidates are continually closed to the optimal solution and far away from the worst solution. α_1 is the angle between the new search direction and the worst solution, and α_2 is the angle between the new search direction and the best solution. The relationship between α_1 and α_2 indicates that the new solution is generated away from the worst solution and towards the best solution in Fig. 4(a). According to the differential vector between the best solution and the worst solution, search direction and adaptive evolutionary steps are provided for the candidates to balance the exploration and exploitation of the algorithm. x^l_{worst}, x^l_{best}, and x^l are individuals in late evolutionary populations as shown in Fig. 4(b). The diversity of the population is decreased during the search process, and the algorithm enters the explorative stage, which leads the population to premature convergence and stagnation. x is not escaped the yellow area. The algorithm is stagnant, and the individuals escaped from the yellow area to find the optimal solution in Fig. 4(b). The new search direction and search step are provided for the stagnant population according to the mutational vectors. As shown in Fig. 4(c), $x_{history}$ is an individual in the historical population. x_{pop} is an individual of the current population. The difference vector between $x_{history}$ and x_{pop} is utilized to guide the search direction and step to make candidates jump out of the yellow position. The formula is shown in Eq. (8).

$$x^{mutation}_{g,p,d} = x_{g,r_1,d} + r(x_{g,r_2,d} - x^{history}_{g,r_3,d}) \tag{8}$$

where r_1, r_2, r_3 and r are random numbers. $x^{history}_{g,r_3,d}$ is a historical individual. $x_{g,r_1,d}$ and $x_{g,r_2,d}$ are individuals in the current population. $x^{mutation}_{g,p,d}$ is a mutational vector.

The archive is updated when ρ_x is smaller than ρ_{xh}. The replacement rate is decreased with the increase of evolutionary generation to provide abundant historical information as shown in Eq. (9). ρ_x is a random number between 0 and 1. The selective method of the historical population is shown in Algorithm 4.

The crossover operator based on a crosstab is utilized to improve the diversity of the population after the mutational vector is obtained. The crosstab is initialized as 0, the 0 in the crosstab is replaced with 1 based on the values of *num* and *u*. According to the crosstab, the dimensions in the candidate where the values are 1 in the crosstab are replaced by the

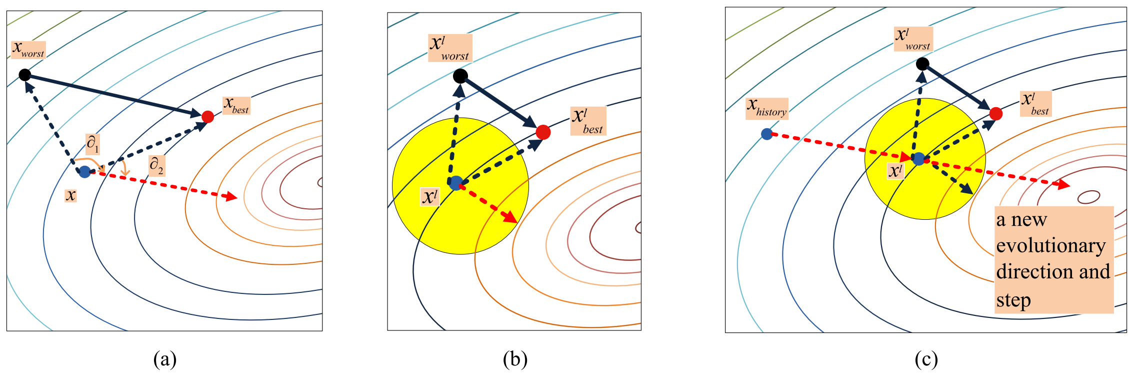


Fig. 4. Evolutionary stage based on the optimal directional guidance and historical learning mechanism.

Algorithm 4: Generate the historical population

Input: x, $x^{history}$ // x is the current population. $x^{history}$ is the historical population

　If $\rho_x < \rho_{xh}$

　　$x^{history} = x$

　Else

　　$x^{history} = x^{history}$

　End

Output: $x^{history}$

Algorithm 5: Generate the offspring population

Input: p, d, $x^{history}$

　$crosstab_{(1:p,1:d)} = 0$

　$u = randperm(p)$

　For i from 1 to p

　　$num = ceil(C_{xh} \times d)$

　　$crosstab_{i,u(1:num)} = 1$

　　For j from 1 to d

　　　$x^{mutation}_{i,j} = x_{r_1,j} + r \times (x_{r_2,j} - x^{history}_{r_3,j})$

　　　If $crosstab_{i,j} = 1$

　　　　$x'_{i,j} = x^{mutation}_{i,j}$

　　　End

　　End

　End

Output: x' // x' is the offspring population

---

values in a mutational vector. The execution of crossover operations ρ_c is related to the crossover rate. C_{xh} is defined in Eq. (10). The crossover operator and the generation of crosstab are performed in Algorithm 5.

$$\rho_{xh} = (cos((FEs/maxFEs) \times \pi) + 1) \times 0.5 \tag{9}$$

where ρ_{xh} is the replacement rate of the historical archive.

$$C_{xh} = (cos(FEs/maxFEs \times 2\pi) + 1) \times 0.5 \tag{10}$$

The improved Jaya algorithm based on optimal directional guidance and historical learning (DH-Jaya) is utilized for optimization. p new individuals are generated according to Eq. (7). The population stored in the historical archive is selected according to ρ_x and ρ_{xh}. The historical learning mechanism strategy is utilized randomly to keep the diversity of the population. ρ_m and ρ_{nm} are random numbers. The fitness values of the parent individuals and the offspring individuals are compared to input better ones to the next generation.

3.2.4. Population recombination strategy

The convergence of the Jaya algorithm is accelerated by the surrogate model. However, the surrogate model is subject to error, which causes surrogate-assisted evolutionary algorithms to converge on locally optimal solutions. Hence, the population recombination strategy is introduced. When a better solution is not found in Ms consecutive times, the population is recombined in this study. Ms is the maximum of stagnation. Individuals in the top $5\%p$ of the dataset D are selected preferentially. The remaining $p - 5\%p$ individuals have randomly selected from dataset D.

3.2.5. Convergence analysis and iterations analysis of SDH-Jaya

The convergence analysis of SDH-Jaya is shown in this section. The Markov chain analysis is utilized to analyze the convergence of SDH-Jaya (Suzuki et al., 1995). The Markov model abides by the assumptions that all the candidates are generated by old individuals and the old individuals are replaced by the candidates.

Definition 1. The sequence of the population $\{x(t), t = 0, 1, 2\ldots\}$ is generated by a population-based stochastic algorithm, $x(t)$ is the stochastic sequence that weakly converges in probability to the global optimum. If and only if.

$$\lim_{t\to\infty} p\left\{x(t) \cap B^* \neq \varnothing\right\} = 1$$

For the optimization problem, a series of the global optimum is B^*.

Definition 2. For a random sequence $\{X_n, X_n \in S, n \geq 0\}$, the state space is S. For $\forall n \geq 0$ and any state $i, j, i_0, \ldots i_{n-1}$, if.

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots X_0 = i_0) = P(X_{n+1} = j | X_n = i)$$

The Markov chain is described as $\{X_n, X_n \in S, n \geq 0\}$, the conditional distribution is that the future state X_{n+1} is independent of the past state and depends on the current state X_n.

Definition 3. The search space Ψ, which is continuous, is mapped to a finite discrete set Φ due to the limitations of the numerical calculation accuracy in computers. Suppose the population scale is m, the state space is.

$$\Phi^{\mathrm{m}} = \underbrace{\Phi \times \Phi \times \cdots \times \Phi}_{\mathrm{m}}$$

Definition 4. Suppose that $\mathrm{M}^0, \mathrm{HM}^0, \mathrm{HC}^0$ and S^0 are the mutation operator, the mutation operator in the historical learning strategy, the crossover operator in the historical learning strategy, and the selection operator of SDH-Jaya, respectively. $f(X^*)$ is the minimum of the population X.

Theorem 1. *The population sequence generated by SDH-Jaya is* $\{x(t), t = 0, 1, 2\ldots\}$. *The finite homogeneous Markov chain in the state space* Φ^m *is* $\{x(t), t = 0, 1, 2\ldots\}$.

Proof. According to Definition 3, the population generated by SDH-Jaya is independent of the past state and the current state. As Definition 2, the $\{x(t), t = 0, 1, 2\ldots\}$ is a finite homogeneous Markov chain in the state space Φ^m.

Property 1. *The population space is* Φ^m *for SDH-Jaya without the selection operator. For all states, the one-step transition probabilities from X to Y are greater than 0.*

$$P\left\{M^0(X)\right\} = Y > 0$$

Proof. Suppose Φ^m indicate the population space of the SDH-Jaya without the selection are n possible individuals $y_i, (i = 1, 2, \ldots, n)$. For SDH-Jaya, the population contains N candidates $x, (j = 1, 2, \ldots, N)$, $\mathcal{J}(s)$ is denoted the set of population, $\mathcal{J}(s)$ as.

$$\mathcal{J}(s) = \left\{j : RA\left(x_i(s) + r\left(x_b(s) - x_w(s)\right)\right) = x_i(s)\right\}$$

where r is a Gaussian random number, X_b, X_w are respectively the best and worst solutions in the current population and the RA rounds the absolute value of x to the nearest integer. According to defining the $y_k(s)$ as the sth feature of y_k and $x_k(s)$ as the sth feature of x_k, if the historical learning operator does not occur. $y_k(s) = x_k(s)$ is the probability described as follows.

$$P\left(y_k(s) = x_k(s) | no\ historical\ learning\right) = 1$$

If the historical learning operator occurs the probability of $P\{HM^0 \times HC^0\}$ is expressed as.

$$P\left\{HM^0 \times HC^0 = Y\right\} = \rho_c \times P\left\{MH^0(X) = Y\right\} + r \times P\{MC^0(X) = Y\}$$
$$P\left\{MH^0(X) = Y\right\} = 1$$
$$P\left\{MC^0(X) = Y\right\} = (\sum_{i=1}^{p}\sum_{j=1}^{d} map_{i,j} \neq 1)/pd$$

where ρ_c is the probability of crossover. r is a Gaussian random value. p is the population size. d is the number of design variables. According to the above description:

$$P\left(y_k(s) = x_k(s) | with\ historical\ learning\right) > 0$$
$$P\left\{M^0(X)\right\} = Y > 0$$

Property 2. *The states* $X, Y, Z \in \Phi^m$, $Z \subset X \cup Y$, *the selection operator is a part of the f classes: if* $f(Z^*) \neq min(f(X^*), f(Y^*))$, *Z cannot be generated by X and Y according to the select operator, that is.*

$$P\{S^0(X, Y) = Z\} = 0$$

If $f(Z^*) = min(f(X^*), f(Y^*))$, Z is generated by X and Y according to select operator, as follow.

$$P\{S^0(X, Y) = Z\} > 0$$

Theorem 2. *Suppose that the population sequence* $\{x(t), t = 0, 1, 2, \ldots\}$ *is generated by SDH-Jaya.* $\{x(t), t = 0, 1, 2, \ldots\}$ *converges to the global optimum in probability.*

Proof. $\forall X, Y, Z \in \Phi^m$, the transition probability is as follow,

$$\begin{aligned} P\{x(t+1) = Z | x(t) = X\} &= P\{S^0 \cdot M^0(X) = Z \\ &= \sum_{Y\in\Phi^m} P\left\{M^0 = Y\right\} P\{S^0(X, Y) = Z\}\} \end{aligned}$$

where B^0 is a population sequence consist of populations that one individual is optimum at least. $B_0 \subset \Phi^m$.

$$B_0 \subset \{X = (x_1, x_2, \ldots, x_m) \in \Phi^m | x_1 \in B^*, \exists i \in \{1, 2, \ldots, m\}\}$$

The transition probability is discussed to be two classes.

Suppose $X \in B_0, Z \notin B_0, f(Z^*) > min(f(X^*), f(Y^*))$. Based on Property 2, $P\{S^0(X, Y) = Z\} = 0$. Then $P\{x(t+1) = Z | x(t) = X\} = 0$.

Suppose $X \in B_0, Z \in B_0, f(Z^*) = min(f(X^*), f(Y^*))$. From Property 2, $P\{S^0(X, Y) = Z\} > 0$. According to Property 1, $P\{M^0(X) = Y\} > 0$. So, $P\{x(t+1) = Z | x(t) = X\} > 0$. Then, all states of B_0 communicate.

B^0 is a positive recurrent, irreducible, aperiodic, and closed set. Based on the properties of the periodic, the Markov chain $\{x(t), t = 1, 2, \ldots\}$ exists a distribution of limiting $\pi(Y)$,

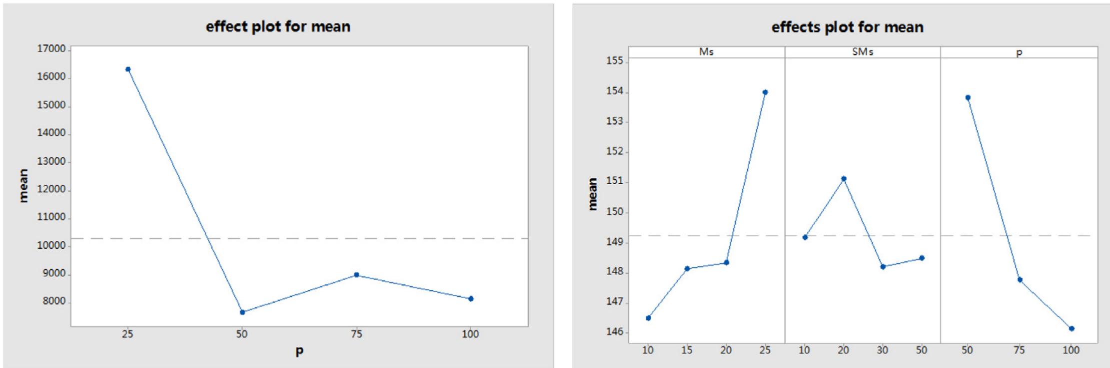
$$\lim_{t\to\infty} P\{x(t) = Y\} = \begin{cases} \pi(Y), & Y \in B^0 \\ 0, & otherwise \end{cases}$$

Then, $\lim_{t\to\infty} P\left\{x(t) = B^0\right\} = 1$. So, $\lim_{t\to\infty} P\{x(t) \cap B^* = \varnothing\} = 1$. DH-Jaya converges to the global optimum in probability according to Definition 1.

The iterations of SDH-Jaya are analyzed. Suppose that the using frequency of the surrogate model is S. The maximal number of stagnations in the surrogate model is SMs. The fitness evaluations saved by the surrogate model are $Fes = S \times SMs$.

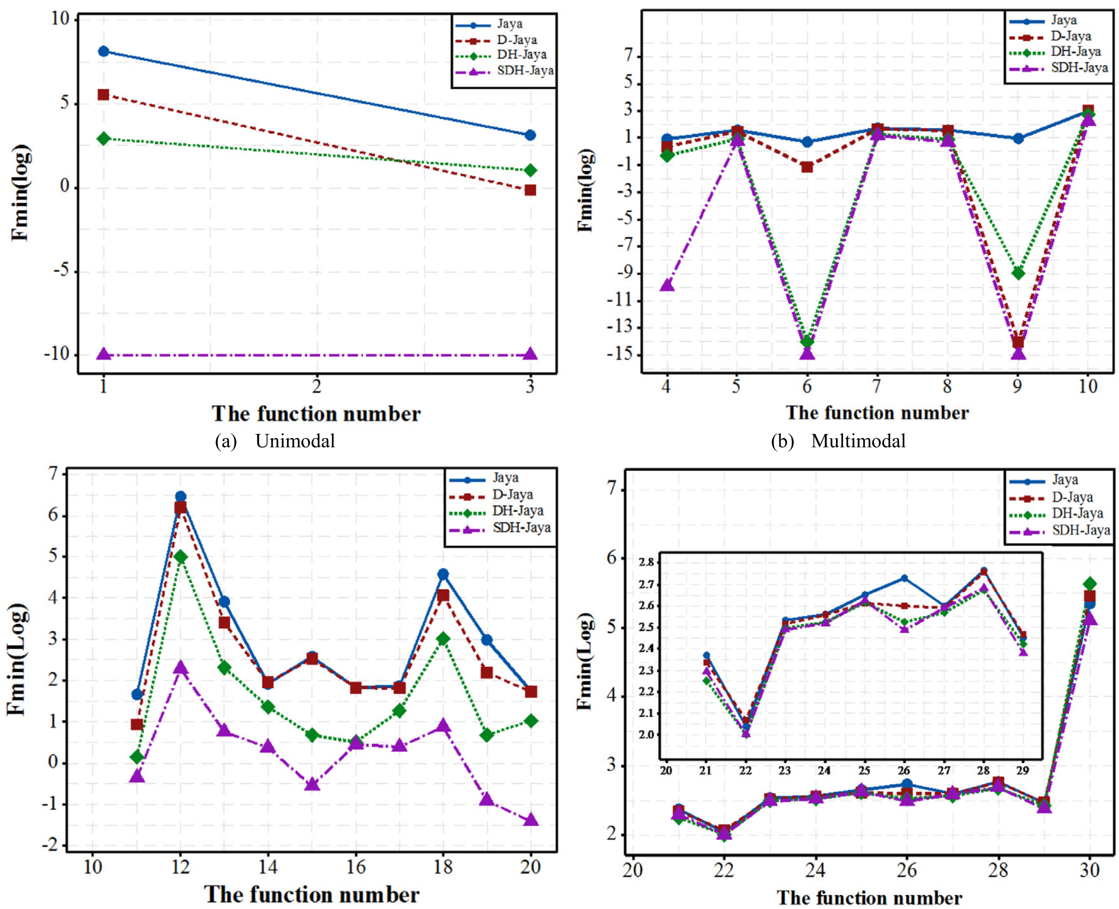
4. The numerical results of the DH-Jaya and SDH-Jaya

The numerical results of the DH-Jaya and SDH-Jaya are analyzed in this section on CEC2017 benchmark problems (Wu et al., 2016). The CEC2017 benchmark function consists of four different categorizations:



(a) Main effects plot of DH-Jaya (b) Main effects plot of SDH-Jaya

Fig. 5. Main effects plot of parameters.



(a) Unimodal (b) Multimodal

(c) Hybrid (d) Composition

Fig. 6. Point plot of Jaya, D-Jaya, DH-Jaya, and SDH-Jaya.

$f_1 - f_3$ are unimodal functions, $f_4 - f_{10}$ are multimodal functions, $f_{11} - f_{20}$ are hybrid functions, and $f_{21} - f_{30}$ are composition functions. f_2 has been excluded because it shows unstable behavior especially for higher dimensions and significant performance variations for the identical algorithm implemented in MATLAB. To ensure the fairness of the experiment, all experiments are implemented by MATLAB and run 51 times independently with $10000D$ fitness evaluations on the PC with a 3.4 GHZ Intel(R), Core (TM) i7-6700 CPU, 8 GB of RAM, and 64-bit OS. For communicating with other interested researchers, the code of the SDH-Jaya is published on GitHub. The link is https://github.com/zzzhhh-320/SDHJAYA.git.

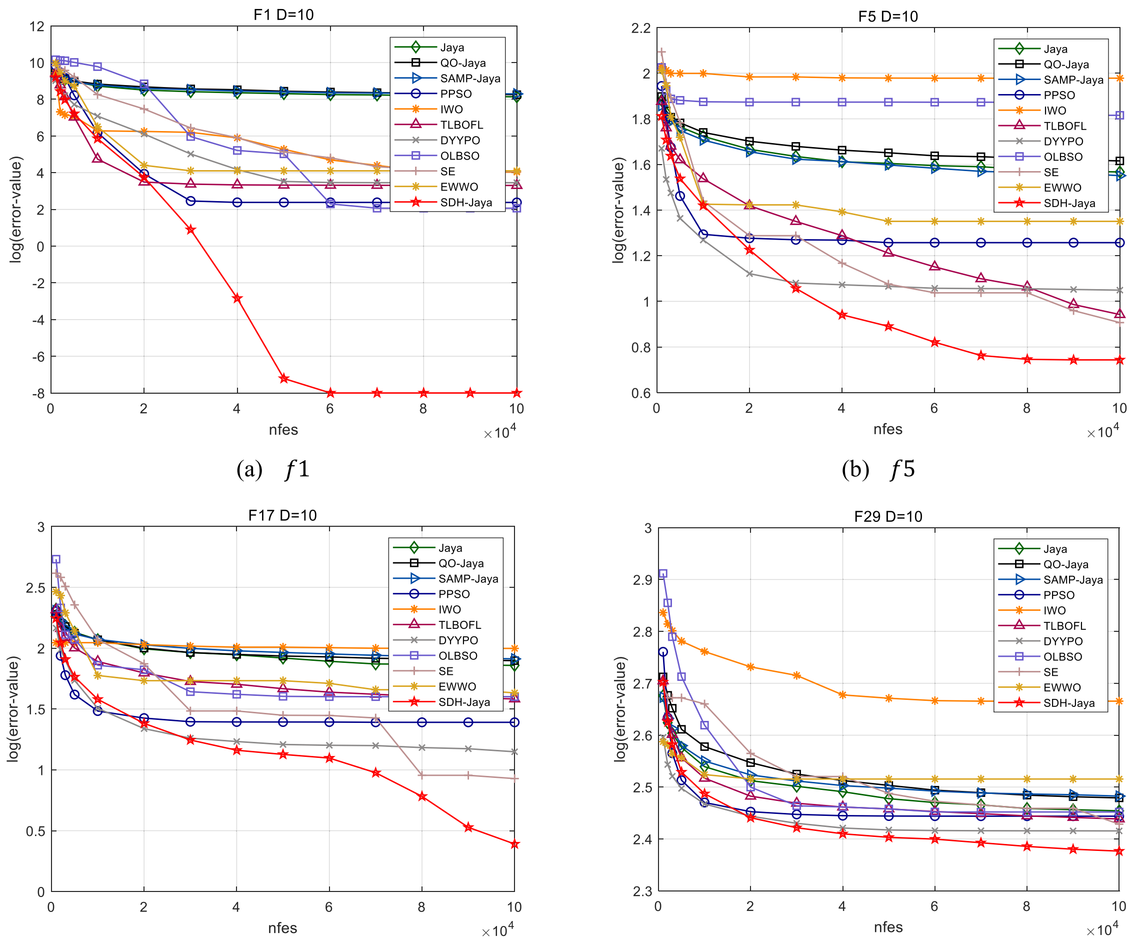
4.1. Parameters analysis

Parameters of the DH-Jaya and SDH-Jaya are analyzed as follows. In DH-Jaya, only one parameter and the selected parameters of DH-Jaya are $p \in (25, 50, 75, 100)$. The parameters in SDH-Jaya include p (the size of the population), Ms (the maximal number of algorithmic stagnations), and SMs (the maximal number of stagnations in the surrogate model).

The selected parameters in SDH-Jaya are as follows, $p \in (50, 75, 100)$, $Ms \in (10, 15, 20, 25)$, $SMs \in (10, 20, 30, 50)$. The best combination of parameters in SDH-Jaya is selected by the Design of Experiments (DOE)

Table 2
ANOVA results for parameter settings of SDH-Jaya.

Source	Sum of squares	Degrees of freedom	Mean square	*F-ratio*	*p-value*
Ms	383.9	3	127.965	3.01	0.0571
SMs	62.9	3	20.965	0.49	0.6912
p	534.5	2	267.25	6.29	**0.0085**
$Ms * SMs$	443.19	9	49.243	1.16	0.3752
$Ms * p$	562.17	6	93.694	2.21	0.0903
$SMs * p$	222.17	6	37.028	0.87	0.5343
$Error$	764.5	18	42.472		
$Total$	2973.31	47			



(a) $f1$ (b) $f5$ (c) $f17$ (d) $f29$

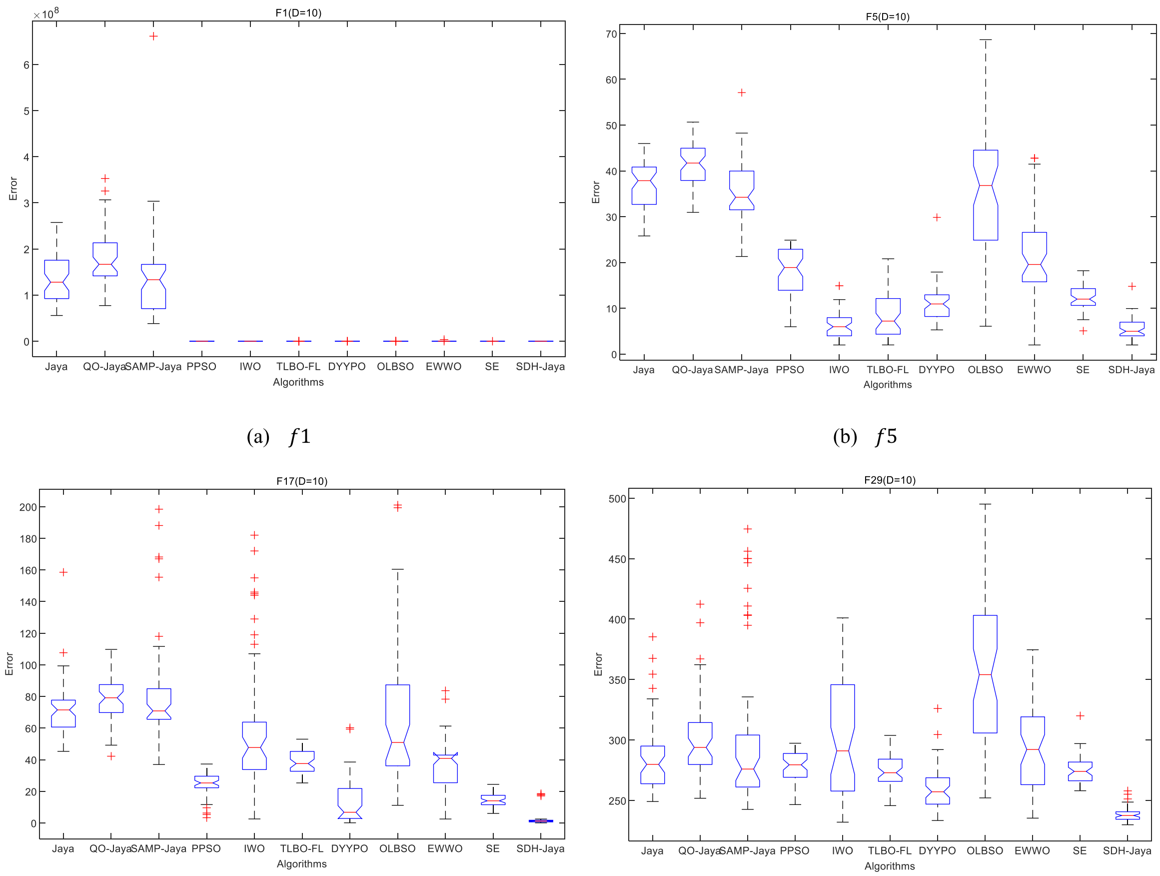
Fig. 7. Convergence curves of eleven algorithms on $f1$, $f5$, $f17$, and $f29$ ($10D$).

approach. All possible combinations are $3 \times 4 \times 4 = 48$. The experimental results are analyzed by the multivariate Analysis of Variance (ANOVA) (Shao et al., 2019). The effect of population size in the DH-Jaya algorithm is analyzed and the result is shown in Fig. 5(a), which means that the best population size is 50. Before ANOVA is conducted, three main assumptions are tested, i.e., normality, homoscedasticity, and independence of residuals. After careful examination, we find no significant deviations from these assumptions. For reasons of space, the above verification procedure is not included in this paper. The results of the ANOVA are shown in Table 2. According to the results from Table 2, the *p-value* of p is less than the confidence level ($\alpha = 0.05$), which indicates that the parameter p is more sensitive than others in SDH-Jaya. Meanwhile, the *F-ratio* of p is the maximum corresponded to others, which suggests that p has the greatest effect on the average performance of the SDH-Jaya among all factors. The best combination of parameters is selected as follows, $p = 100$, $Ms = 10$, and $SMs = 30$ according to Fig. 5(b).

4.2. The effectiveness of mechanisms

To analyze the effectiveness of the optimal directional guidance mechanism, the historical learning mechanism, and the co-evolutionary mechanism, the Jaya based on the optimal directional guidance mechanism (D-Jaya), the Jaya based on the optimal directional guidance mechanism and historical learning mechanism (DH-Jaya), surrogate-assisted Jaya based on the optimal directional guidance mechanism and historical learning mechanism (SDH-Jaya), and the classic Jaya are compared on the CEC 2017 benchmark problem in the $10D$ with 100000 fitness evaluation. As shown in Table 3, the accuracy of the algorithm is improved with each strategy utilized and SDH-Jaya has better performance than other comparison algorithms for addressing CEC 2017 problems.

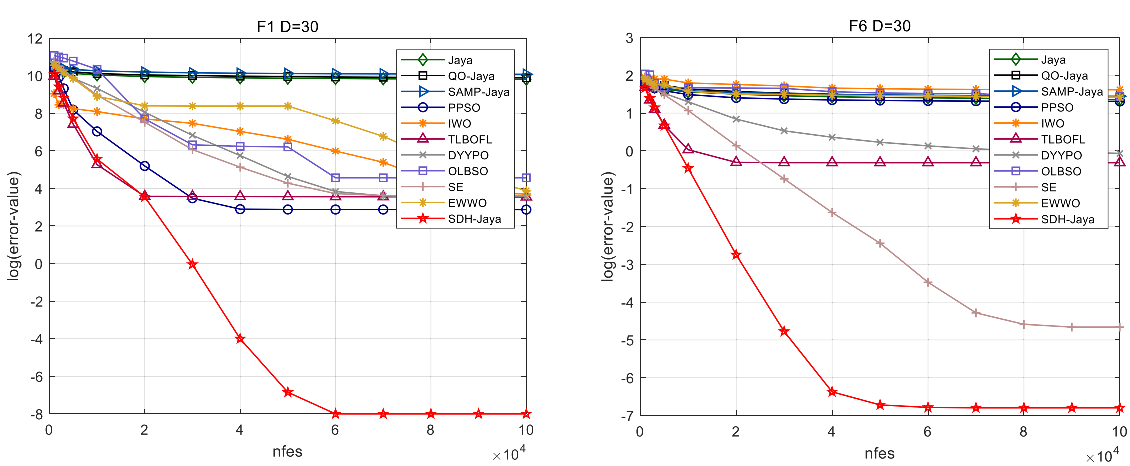
According to the results in Table 3 and the best result is bold, the optimal directional guidance mechanism, the historical learning mechanism, and the co-evolutionary mechanism are effective to improve



(a) $f1$ (b) $f5$

(c) $f17$ (d) $f29$

Fig. 8. Boxplots of eleven algorithms on $f1$, $f5$, $f17$, and $f29$ ($10D$).



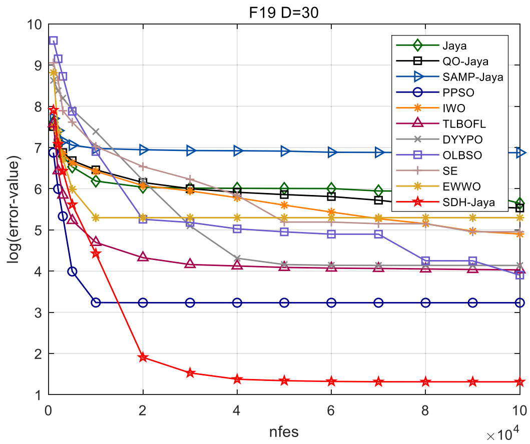
(e) $f1$ (f) $f6$

Fig. 9. Convergence curves of eleven algorithms on $f1$ and $f6$ ($30D$).

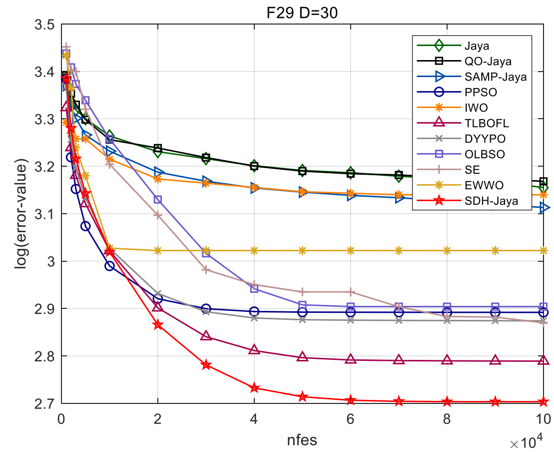
precision. The point plot of Jaya, D-Jaya, DH-Jaya, and SDH-Jaya are shown in Fig. 6. The purple line at the bottom of all the lines means SDH-Jaya has better precision than other comparison algorithms in addressing 29 benchmark problems.

5. CEC 2017 benchmark test suite experiment

In this section, the basic Jaya algorithm (Venkata Rao, 2016), QO-Jaya (Warid et al., 2018), SAMP-Jaya (Venkata (Venkata Rao and Saroj, 2017), and state-of-the-art algorithms such as invasive weed optimization (IWO) (Mehrabian and Lucas, 2006), proactive particles in swarm optimization (PPSO) (Tangherloni et al., 2017), teaching-learning based optimization with focused learning (TLBO-FL) (Kommadath et al., 2017), dynamic Yin–Yang pair optimization (DYYPO) (Maharana et al., 2017), orthogonal learning of brain storm optimization (OLBSO) (Ma et al., 2020), enhanced water wave optimization (EWWO) (Zhao et al., 2019), and spherical evolution (SE) (Tang, 2019)

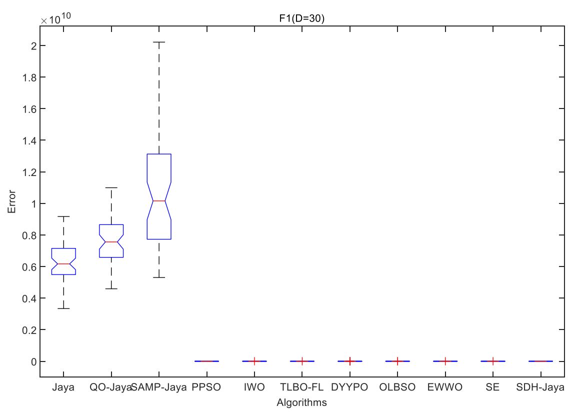


(g) $f19$

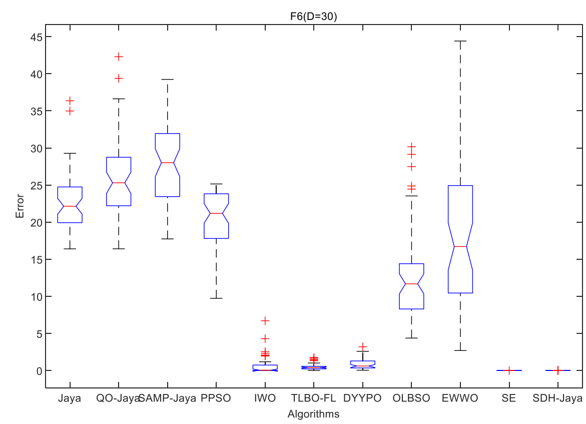


(h) $f29$

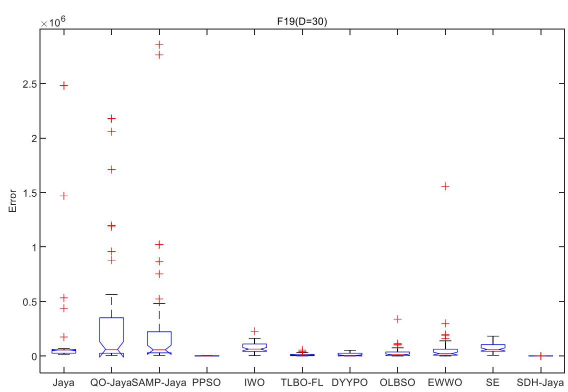
Fig. 10. Convergence curves of eleven algorithms on $f19$ and $f29$ ($30D$).



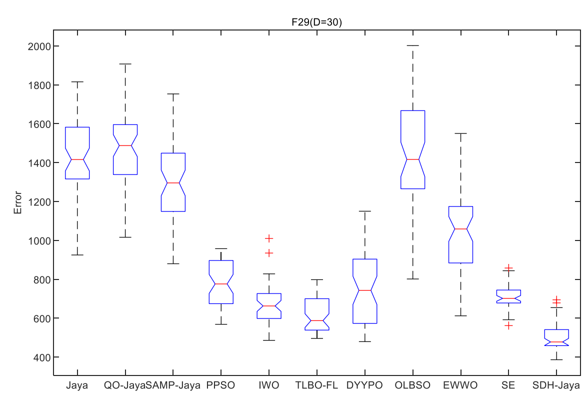
(e) $f1$



(f) $f6$



(g) $f19$



(h) $f29$

Fig. 11. Boxplots of eleven algorithms on $f1$, $f6$, $f19$, and $f29$ ($30D$).

are utilized for comparison. The parameters of the Jaya algorithm, QO-Jaya, SAMP-Jaya, DH-Jaya, and SDH-Jaya in this experiment are set in Table 4. The remaining parameters of the comparison algorithm are set to the values recommended in the original paper. For ensuring the fairness of the experiment, each benchmark function is run independently 51 times on $D = 10$, $D = 30$, $D = 50$, and $D = 100$ with the fitness evaluation of $10000D$. The fitness value and variance of less than $1e-8$ are regarded as 0.

5.1. Numerical result

The results of eleven algorithms are shown in Tables 5–8, and the optimal fitness value is highlighted in bold. Convergence curves are

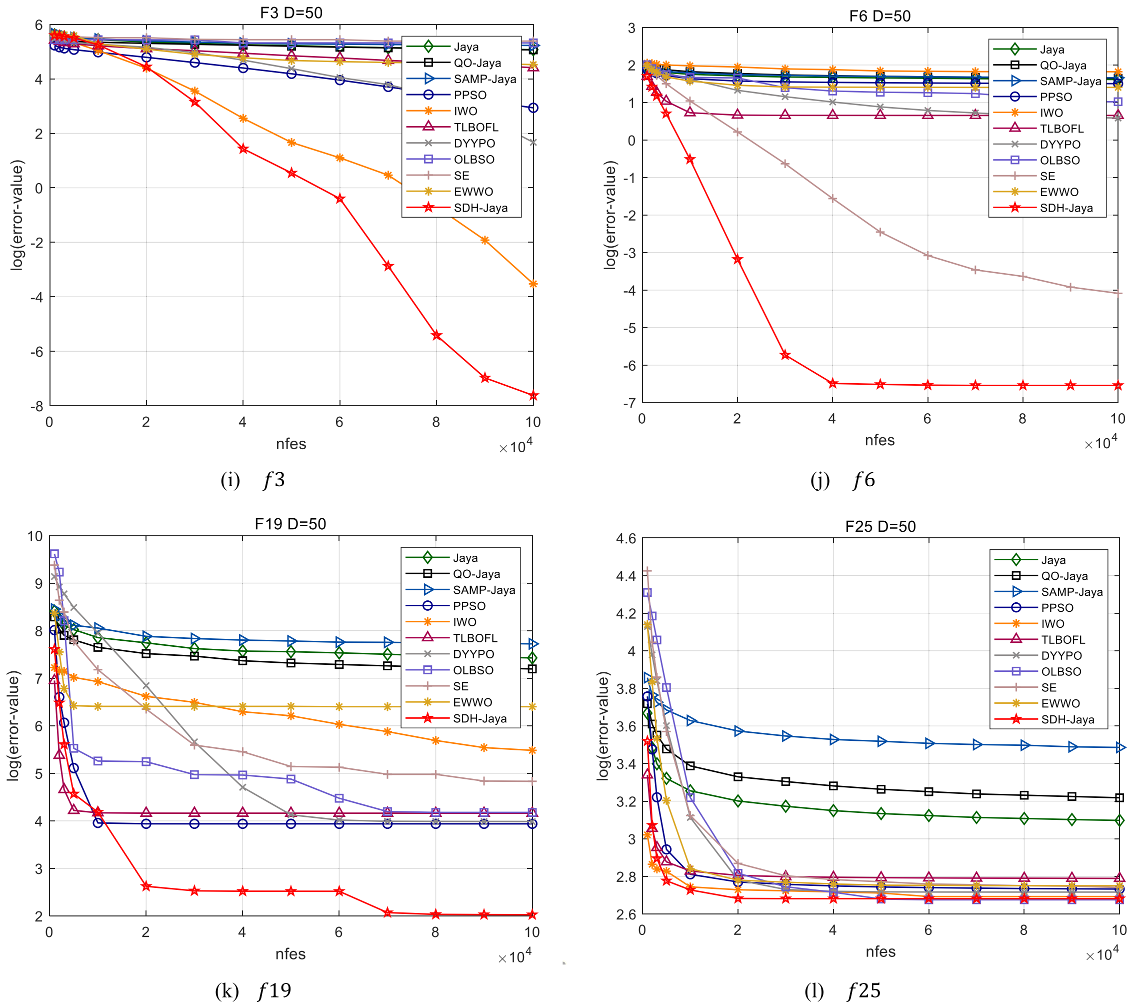


Fig. 12. Convergence curves of eleven algorithms on $f3$, $f6$, $f19$, and $f25$ ($50D$).

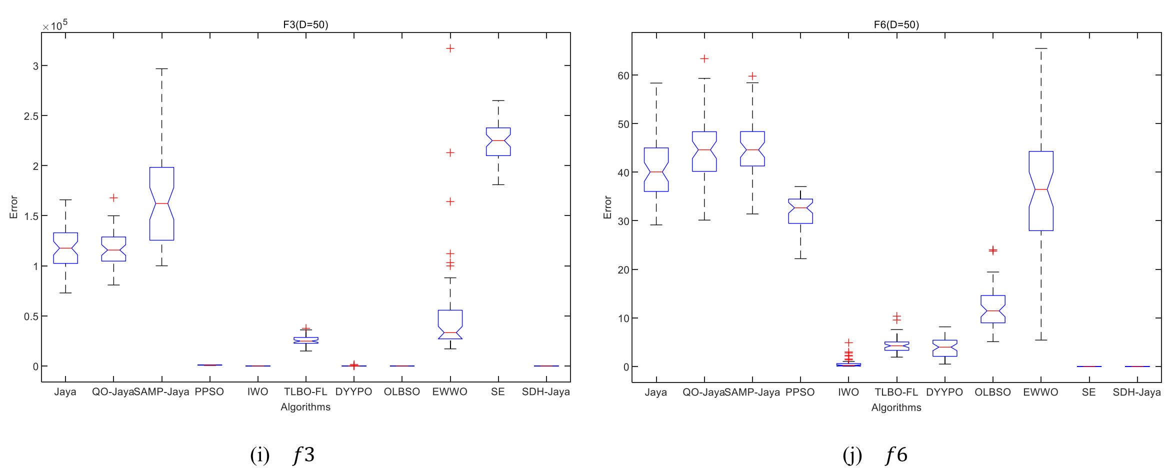
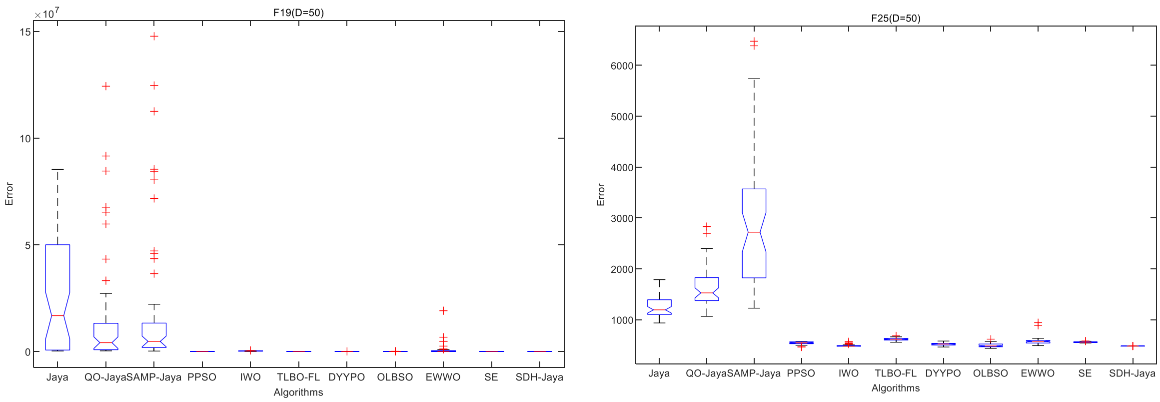


Fig. 13. Boxplots of eleven algorithms on $f3$and $f6$ ($50D$).

shown in Figs. 7, 9, 10, 12, and 15. Boxplots shown in Figs. 8, 11, 13, 14, and 16 are utilized to reveal the stability of algorithms.

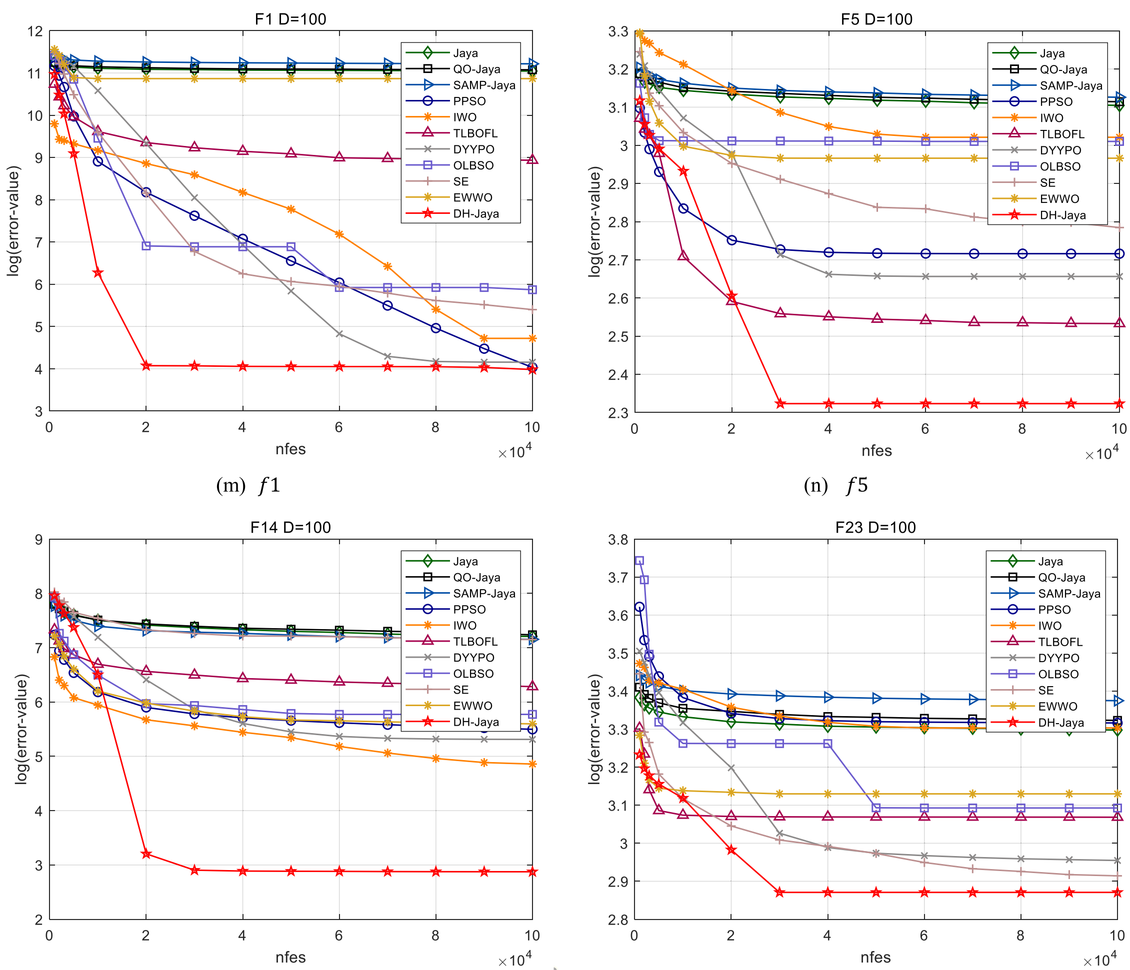
5.2. Non-parametric test

In this section, the Friedman test is utilized to analyze the significant difference between the mean errors of the eleven algorithms on the CEC2017 benchmark problems. The average rank of SDH-Jaya in the dimension of 10, 30, 50 and DH-Jaya in the dimension of 100 are the smallest of the eleven algorithms with $\alpha = 0.05$ and $\alpha = 0.1$. SDH Jaya and DH Jaya have significant differences with other comparison algorithms. The results are shown in Tables 9–12, and Figs. 17–20. In addition, the Wilcoxon symbolic rank test is utilized to analyze the significant difference between the mean errors of SDH-Jaya and DH-Jaya with other comparison algorithms in the dimension of 10, 30, 50, 100 respectively. R^{+} is the sum of the rank that SDH-Jaya or DH-Jaya



(k) $f19$ (l) $f25$

Fig. 14. Boxplots of eleven algorithms on $f19$ and $f25$ ($50D$).



(m) $f1$ (n) $f5$

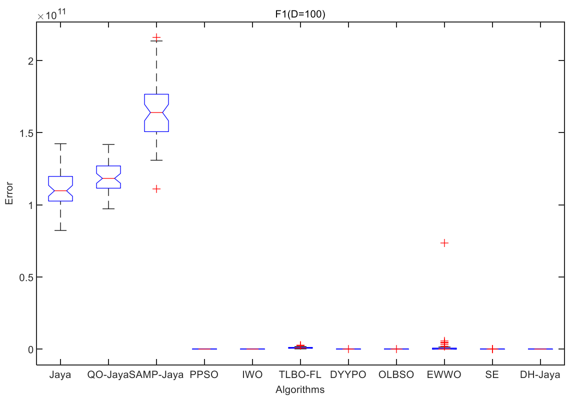
(o) $f14$ (p) $f23$

Fig. 15. Convergence curves of eleven algorithms on $f1$, $f5$, $f14$, and $f23$ ($100D$).

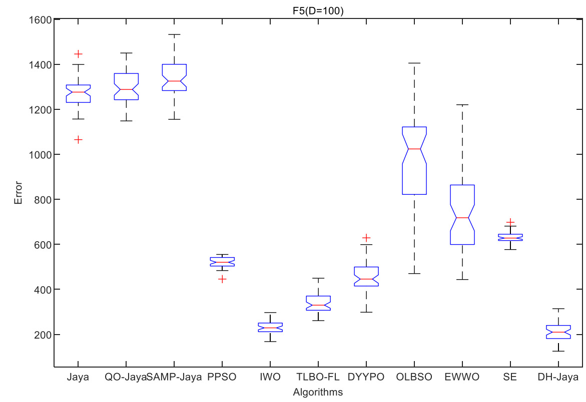
is superior to the current comparison algorithms. R^- is the sum of the rank that the current comparison algorithm is superior to SDH-Jaya or DH-Jaya. The rank sum of R^+ of SDH-Jaya or DH-Jaya is higher than the rank sum of R^- in the ten groups of comparison results. The *p-value* represents asymptotic significance. When *p-value* $< \alpha$, the corresponding α position of the algorithm is "Yes", which represents that the two algorithms have significant differences. The results are shown in Table 13.

5.3. Analysis and discussion

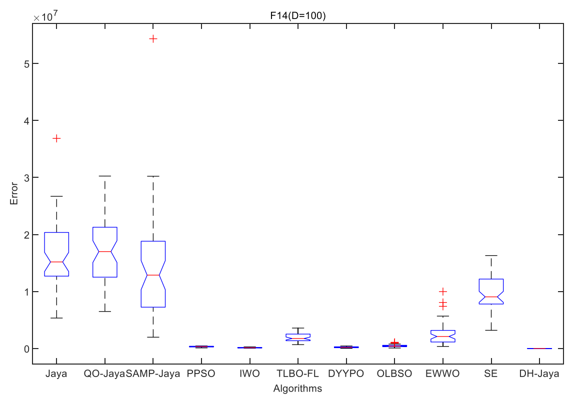
According to the mean and standard deviation values of eleven algorithms are shown in Tables 5–8 in the dimension of 10, 30, 50, 100 respectively. The results of SDH-Jaya are better than the other ten comparison algorithms in 20 benchmark problems in the $10D$, 15 benchmark problems in the $30D$, and 16 benchmark problems in the $50D$. The DH-Jaya has 15 benchmark problems in $100D$ that are better



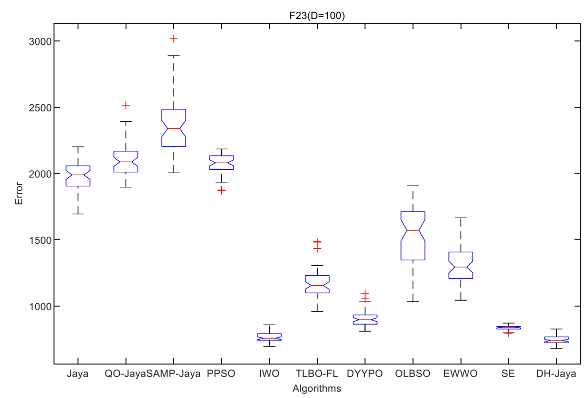
(m) $f1$



(n) $f5$



(o) $f14$



(p) $f23$

Fig. 16. Boxplots of eleven algorithms on $f1$, $f5$, $f14$, and $f23$ (100D).

Table 3
Results of Jaya, D-Jaya, DH-Jaya, and SDH-Jaya (10D).

Type	Jaya		D-Jaya		DH-Jaya		SDH-Jaya	
	mean	std	mean	std	mean	std	mean	std
Unimodal	6.8E+07	2.6E+07	1.7E+05	1.9E+05	4.2E+02	9.4E+02	**0.0E+00**	**0.0E+00**
Multimodal	1.5E+02	3.7E+01	1.6E+02	3.1E+01	7.6E+01	2.7E+01	**2.7E+01**	**1.6E+01**
Hybrid	2.9E+05	3.3E+05	1.5E+05	1.3E+05	1.0E+04	1.9E+04	**2.1E+01**	**1.8E+01**
Composition	2.2E+04	3.3E+04	2.9E+04	4.2E+04	4.3E+04	**1.8E+04**	**1.3E+04**	2.8E+04
Average	1.7E+07	6.6E+06	9.0E+04	9.4E+04	1.3E+04	9.7E+03	**3.2E+03**	**7.0E+03**

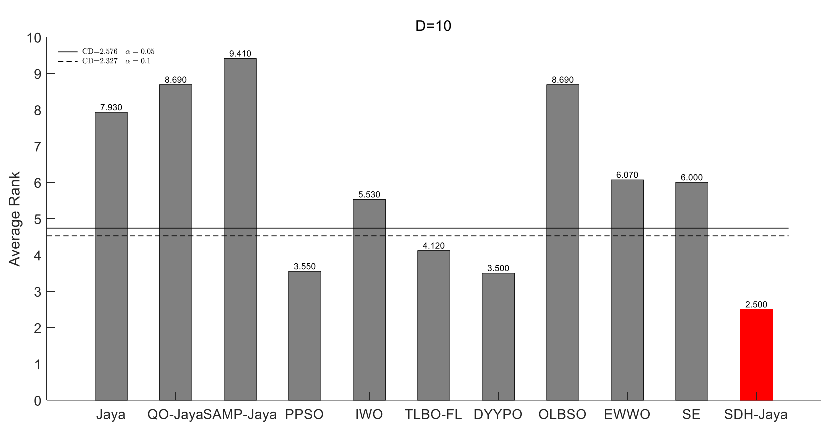


Fig. 17. Rankings obtained through Friedman test (10D).

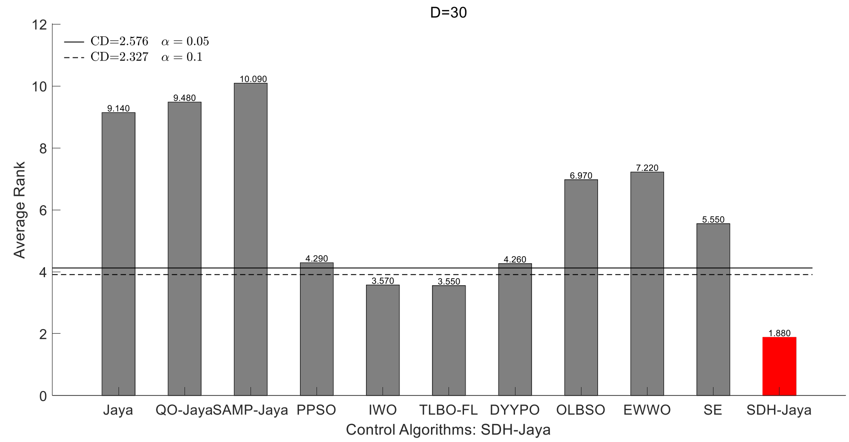


Fig. 18. Rankings obtained through Friedman test (30D).

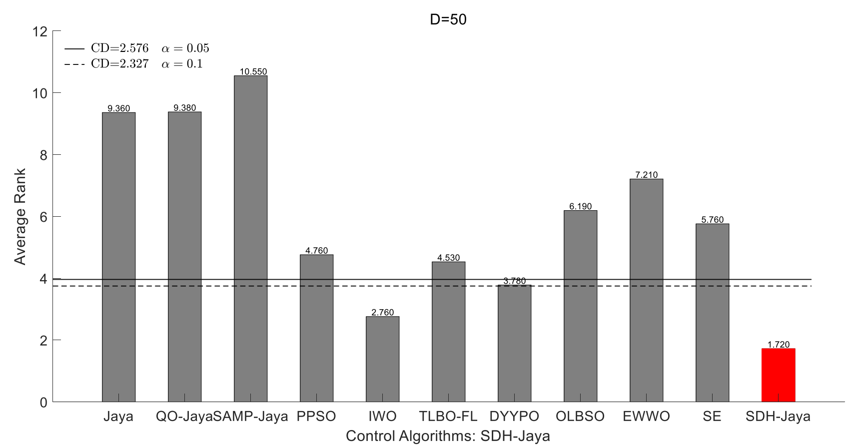


Fig. 19. Rankings obtained through Friedman test (50D).

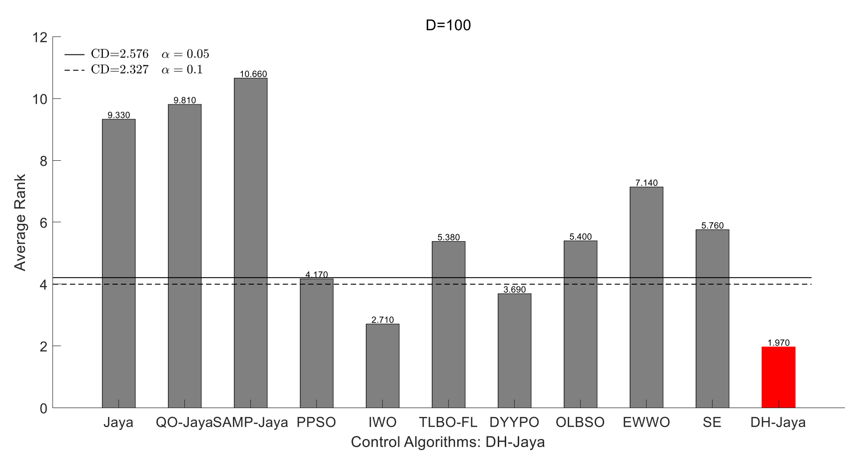


Fig. 20. Rankings obtained through Friedman test (100D).

Table 4
The parameters setting.

Algorithm	Parameters
Jaya	$p = 25$
QO-Jaya	$p = 20$
SAMP-Jaya	$p = 20; m = 2$
DH-Jaya	$p = 50$
SDH-Jaya	$p = 100; SMs = 30; Ms = 10$

than the comparison algorithms. The optimization results are better than other algorithms. The SDH-Jaya is significantly better than other comparison algorithms at a confidence interval of 90 except for the SE in the 10D. At a confidence interval of 95, the SDH-Jaya significantly outperforms other comparison algorithms except for the PPSO and SE in the 10D, the IWO in the 50D according to Table 13.

The radial basis model is not good for solving high-dimensional problems. Therefore, DH-Jaya is tested in the 100D. The optimization results of DH-Jaya are better than other algorithms and significantly better than other comparison algorithms at a confidence interval of 90

Table 5
The results of eleven algorithms (10D).

Fun	Jaya		QO-Jaya		SAMP-Jaya		PPSO		IWO		DYYPO		TLBO-FL		OLBSO		EWWO		SE		SDH-Jaya	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
1	1.36E+08	5.24E+07	1.78E+08	5.92E+07	1.96E+08	3.12E+08	2.39E+02	2.01E+02	4.95E+03	3.92E+03	2.86E+03	3.27E+03	2.02E+03	2.46E+03	1.36E+03	1.88E+03	8.07E+04	5.11E+05	5.63E+03	5.67E+03	**0.00E+00**	**0.00E+00**
3	1.35E+03	4.69E+02	1.21E+03	4.37E+02	2.68E+03	4.18E+03	**0.00E+00**	**0.00E+00**	3.18E−04	9.00E−05	1.17E−05	4.30E−05	1.11E−04	7.84E−04	1.73E+03	1.59E+03	1.65E+01	6.12E+01	4.45E+03	1.58E+03	**0.00E+00**	**0.00E+00**
4	8.30E+00	1.75E+00	8.22E+00	1.18E+00	4.39E+01	7.59E+01	1.20E+00	9.36E−01	4.50E−01	2.34E−01	2.07E+00	8.25E+00	3.03E+00	1.17E+00	1.25E+01	1.79E+01	6.33E+00	1.15E+00	6.55E+00	6.33E−01	**0.00E+00**	**0.00E+00**
5	3.69E+01	4.97E+00	4.13E+01	4.61E+00	3.56E+01	6.89E+00	1.81E+01	5.10E+00	6.28E+00	2.79E+00	1.12E+01	4.23E+00	8.75E+00	5.57E+00	3.60E+01	1.45E+01	2.12E+01	9.42E+00	1.22E+01	2.74E+00	**5.54E+00**	**2.45E+00**
6	5.19E+00	1.55E+00	6.50E+00	1.59E+00	5.80E+00	2.23E+00	2.26E−01	3.08E−01	2.85E−01	1.04E+00	6.36E−05	6.02E−05	**0.00E+00**	4.43E−07	7.57E+00	5.33E+00	7.04E−02	3.11E−01	1.29E−07	6.09E−08	**0.00E+00**	**0.00E+00**
7	4.94E+01	6.03E+00	5.29E+01	7.44E+00	4.98E+01	1.50E+01	1.69E+01	**2.21E+00**	**1.40E+01**	3.51E+00	2.18E+01	6.02E+00	2.76E+01	3.97E+00	5.71E+01	1.96E+01	3.45E+01	1.02E+01	2.48E+01	2.97E+00	1.60E+01	2.54E+00
8	3.65E+01	4.42E+00	3.72E+01	7.01E+00	3.29E+01	5.98E+00	9.95E+00	2.37E+00	5.42E+00	2.34E+00	1.32E+01	4.50E+00	1.23E+01	4.40E+00	3.14E+01	1.21E+01	2.07E+01	9.05E+00	1.25E+01	2.20E+00	**4.96E+00**	**2.14E+00**
9	9.32E+00	3.65E+00	1.79E+01	3.75E+01	1.91E+01	3.28E+01	**0.00E+00**	**0.00E+00**	7.98E−05	1.83E−05	1.96E−02	9.82E−02	8.91E−03	6.36E−02	1.99E+02	1.55E+02	5.20E+00	3.21E+01	4.10E−07	9.01E−07	**0.00E+00**	**0.00E+00**
10	9.48E+02	2.41E+02	1.26E+03	1.73E+02	7.18E+02	3.31E+02	5.03E+02	1.55E+02	3.74E+02	1.29E+02	3.67E+02	1.69E+02	9.55E+02	2.14E+02	9.92E+02	3.44E+02	7.31E+02	2.86E+02	5.25E+02	1.29E+02	**1.63E+02**	**1.07E+02**
11	4.69E+01	4.31E+01	6.47E+01	5.06E+01	8.79E+01	9.15E+01	1.69E+01	5.31E+00	9.54E+00	7.71E+00	9.28E+00	4.79E+00	4.12E+00	1.46E+00	5.50E+01	4.83E+01	1.68E+01	1.41E+01	6.42E+00	1.57E+00	**4.49E−01**	**5.38E−01**
12	2.87E+06	3.34E+06	2.87E+06	3.33E+06	4.00E+06	7.85E+06	4.55E+03	2.51E+03	2.11E+04	1.85E+04	1.35E+04	1.20E+04	6.56E+04	5.49E+04	1.30E+06	1.19E+06	5.85E+05	1.66E+06	8.34E+05	5.78E+05	**1.92E+02**	**1.58E+02**
13	8.25E+03	1.02E+04	1.18E+04	9.58E+03	1.19E+04	1.11E+04	1.39E+03	1.33E+03	6.78E+03	8.94E+03	5.08E+03	5.64E+03	2.45E+03	2.16E+03	9.29E+03	7.57E+03	9.68E+03	1.04E+04	1.44E+03	1.04E+03	**5.68E+00**	**1.87E+00**
14	8.42E+01	2.78E+02	1.19E+03	5.35E+03	3.72E+03	8.74E+03	3.73E+01	1.17E+01	8.53E+01	5.85E+01	2.07E+01	2.20E+01	6.73E+01	1.83E+01	2.23E+03	3.07E+03	1.24E+02	9.92E+01	8.40E+01	8.18E+01	**2.38E+00**	**3.92E+00**
15	3.82E+02	1.94E+02	4.93E+02	2.55E+02	2.61E+03	7.44E+03	5.33E+01	2.27E+01	6.33E+02	6.83E+02	4.36E+01	1.11E+02	1.26E+02	4.34E+01	3.34E+03	4.44E+03	4.84E+02	6.31E+02	3.10E+02	3.20E+02	**2.81E−01**	**4.25E−01**
16	6.70E+01	2.88E+01	8.73E+01	4.20E+01	8.71E+01	9.72E+01	8.30E+01	7.25E+01	1.96E+02	1.20E+02	4.38E+01	5.79E+01	8.91E+00	2.19E+01	2.52E+02	1.37E+02	5.89E+01	5.16E+01	5.61E+00	**3.56E+00**	**2.79E+00**	7.79E+00
17	7.22E+01	1.73E+01	7.90E+01	1.44E+01	8.17E+01	3.54E+01	2.46E+01	7.43E+00	6.19E+01	4.37E+01	1.41E+01	1.39E+01	3.83E+01	7.83E+00	6.76E+01	4.48E+01	3.81E+01	1.45E+01	1.47E+01	**4.58E+00**	**2.46E+00**	**4.58E+00**
18	3.85E+04	9.68E+03	4.70E+04	2.59E+04	3.73E+04	8.76E+03	8.78E+02	7.14E+02	8.99E+03	1.01E+04	8.76E+03	6.42E+03	6.15E+03	5.63E+03	1.84E+04	1.73E+04	2.26E+04	1.52E+04	8.65E+03	5.06E+03	**7.53E+00**	**9.56E+00**
19	9.56E+02	3.44E+03	1.28E+03	4.72E+03	1.54E+03	5.52E+03	2.25E+01	1.56E+01	2.98E+02	8.70E+02	9.27E+01	2.94E+02	6.06E+01	3.18E+01	4.81E+03	6.53E+03	2.66E+03	4.21E+03	1.21E+02	2.83E+02	**1.22E−01**	**2.42E−01**
20	5.44E+01	1.17E+01	8.24E+01	5.27E+01	4.80E+01	2.23E+01	2.78E+01	8.96E+00	8.33E+01	7.41E+01	8.01E+00	9.07E+00	1.46E+01	9.45E+00	7.57E+01	4.85E+01	4.02E+01	3.47E+01	8.49E+00	2.71E+00	**3.79E−02**	**1.56E−01**
21	2.35E+02	6.09E+00	1.15E+02	2.19E+01	2.25E+02	3.08E+01	1.04E+02	2.15E+01	1.95E+02	3.51E+01	**1.00E+02**	**9.03E−01**	1.42E+02	5.17E+01	1.50E+02	5.90E+01	1.48E+02	5.74E+01	1.66E+02	4.59E+01	1.96E+02	3.55E+01
22	1.10E+02	2.29E+01	1.17E+02	1.53E+01	1.35E+02	8.91E+01	9.67E+01	1.68E+01	1.37E+02	1.34E+02	9.75E+01	1.99E+01	**9.33E+01**	2.27E+01	1.06E+02	4.25E+00	9.61E+01	1.94E+01	9.98E+01	6.13E+00	1.00E+02	**2.90E−01**
23	3.42E+02	5.25E+00	3.46E+02	5.98E+00	3.43E+02	8.15E+00	3.42E+02	1.05E+01	3.09E+02	4.45E+00	3.09E+02	4.46E+01	**3.07E+02**	3.84E+00	3.48E+02	1.82E+01	3.22E+02	8.16E+00	3.14E+02	**2.58E+00**	3.08E+02	**2.58E+00**
24	3.63E+02	**3.71E+01**	3.23E+02	9.65E+01	3.55E+02	5.45E+01	2.27E+02	1.35E+02	2.61E+02	1.12E+02	**1.17E+02**	4.97E+01	3.10E+02	6.90E+01	3.19E+02	1.15E+02	3.25E+02	8.72E+01	3.11E+02	7.81E+01	3.30E+02	4.71E+01
25	4.48E+02	2.28E+01	4.40E+02	2.04E+01	4.47E+02	2.53E+01	**4.04E+02**	**1.45E+01**	4.05E+02	5.91E+01	4.23E+02	2.33E+01	4.26E+02	2.25E+01	4.31E+02	2.06E+01	4.27E+02	2.54E+01	4.16E+02	2.04E+01	4.20E+02	2.38E+01
26	5.35E+02	3.55E+02	4.30E+02	8.66E+01	5.89E+02	3.59E+02	**2.67E+02**	7.66E+01	4.30E+02	3.64E+02	3.03E+02	3.14E+01	3.01E+02	4.63E+01	4.89E+02	3.07E+02	3.68E+02	3.89E+01	3.19E+02	2.32E+01	3.07E+02	**1.91E+01**
27	3.97E+02	2.18E+00	4.04E+02	1.90E+01	4.17E+02	3.24E+01	4.27E+02	1.35E+01	3.98E+02	2.00E+01	3.96E+02	3.76E+00	3.93E+02	3.28E+00	4.17E+02	6.84E+01	3.92E+02	1.89E+00	3.90E+02	4.04E−01	**3.89E+02**	**3.82E−01**
28	5.82E+02	1.36E+02	5.23E+02	9.69E+01	6.08E+02	1.43E+02	**2.94E+02**	**4.20E+01**	4.38E+02	1.73E+02	3.01E+02	5.11E+01	4.47E+02	1.58E+02	4.96E+02	1.16E+02	4.39E+02	9.60E+01	4.46E+02	1.03E+02	4.82E+02	1.44E+02
29	2.85E+02	2.98E+01	3.01E+02	3.44E+01	3.04E+02	6.73E+01	2.78E+02	1.35E+01	3.00E+02	4.98E+01	2.60E+02	1.89E+01	2.74E+02	1.38E+01	3.55E+02	5.67E+01	2.93E+02	3.47E+01	2.76E+02	1.18E+01	**2.38E+02**	**5.63E+00**
30	2.24E+05	3.33E+05	3.77E+05	4.27E+05	6.84E+05	1.35E+06	**2.99E+03**	**8.99E+02**	3.62E+05	5.66E+05	6.70E+03	6.52E+03	2.79E+05	4.92E+05	6.10E+05	4.81E+05	2.49E+05	2.85E+05	7.37E+04	5.60E+04	1.28E+05	2.82E+05

Table 6
The results of eleven algorithms (30*D*).

Fun	Jaya		QO-Jaya		SAMP-Jaya		PPSO		IWO		DYYPO		TLBO-FL		OLBSO		EWWO		SE		SDH-Jaya	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
1	6.40E+09	1.22E+09	7.69E+09	1.68E+09	1.18E+10	6.24E+09	7.48E+02	6.06E+02	7.67E+03	5.90E+03	3.71E+03	5.04E+03	3.51E+03	3.61E+03	4.70E+04	3.22E+04	6.92E+08	4.94E+09	5.87E+03	4.57E+03	**0.00E+00**	**0.00E+00**
3	5.24E+04	1.18E+04	3.39E+04	6.18E+03	7.01E+04	2.19E+04	1.13E+00	4.83E−01	8.16E−03	1.51E−03	5.27E+02	3.76E+03	2.99E+03	1.08E+03	4.82E+03	8.85E+03	1.10E+04	1.00E+04	9.98E+04	1.44E+04	**0.00E+00**	**0.00E+00**
4	2.83E+02	1.89E+02	3.11E+02	1.27E+02	1.02E+03	1.05E+03	**4.39E+01**	3.19E+01	8.84E+01	7.05E+00	9.12E+01	2.49E+01	9.01E+01	2.37E+01	7.22E+01	2.09E+01	1.04E+02	2.97E+01	9.83E+01	6.51E+00	5.93E+01	**4.49E+00**
5	2.31E+02	1.73E+01	2.53E+02	2.26E+01	2.40E+02	2.87E+01	1.12E+02	1.33E+01	**3.60E+01**	1.01E+01	9.09E+01	2.43E+01	3.95E+01	2.07E+01	1.85E+02	4.73E+01	1.53E+02	3.24E+01	9.94E+01	**6.64E+00**	4.35E+01	1.02E+01
6	2.26E+01	4.07E+00	2.60E+01	5.47E+00	2.79E+01	5.35E+00	2.03E+01	4.15E+00	6.12E−01	1.23E+00	8.59E−01	7.17E−01	4.87E−01	4.24E−01	1.31E+01	6.37E+00	1.78E+01	1.01E+01	**7.14E−08**	**2.39E−08**	1.60E−07	3.00E−07
7	3.45E+02	3.09E+01	3.64E+02	3.00E+01	4.52E+02	1.10E+02	1.35E+02	1.63E+01	**6.69E+01**	8.77E+00	1.44E+02	3.08E+01	1.39E+02	4.75E+01	2.28E+02	3.80E+01	1.96E+02	4.66E+01	1.37E+02	8.86E+00	7.53E+01	1.19E+01
8	2.41E+02	1.42E+01	2.55E+02	2.06E+01	2.55E+02	2.79E+01	8.10E+01	1.04E+01	3.75E+01	9.72E+00	9.63E+01	2.45E+01	**3.67E+01**	1.84E+01	1.73E+02	4.39E+01	1.67E+02	3.46E+01	1.04E+02	**8.16E+00**	4.58E+01	1.07E+01
9	3.02E+03	1.37E+03	4.17E+03	1.64E+03	4.02E+03	1.77E+03	1.36E+03	2.82E+02	1.02E−02	6.37E−02	6.54E+02	7.73E+02	3.45E+01	2.71E+01	5.16E+03	2.21E+03	4.05E+03	2.34E+03	9.69E+01	4.96E+01	**1.76E−03**	**1.25E−02**
10	6.86E+03	3.55E+02	6.14E+03	7.34E+02	6.50E+03	5.60E+02	3.13E+03	3.46E+02	**1.73E+03**	4.13E+02	2.84E+03	6.02E+02	6.69E+03	**2.77E+02**	4.73E+03	7.02E+02	3.98E+03	7.38E+02	4.30E+03	2.93E+02	2.69E+03	5.89E+02
11	8.03E+02	1.69E+02	8.88E+02	1.96E+02	1.26E+03	6.75E+02	8.43E+01	**1.84E+01**	1.25E+02	4.58E+01	1.16E+02	4.11E+01	8.16E+01	4.14E+01	1.77E+02	6.21E+01	2.72E+02	1.80E+02	2.27E+02	6.19E+01	**4.63E+01**	2.87E+01
12	7.34E+07	3.65E+07	1.13E+08	8.55E+07	3.13E+08	3.88E+08	2.77E+04	8.55E+03	1.03E+06	9.41E+05	1.50E+06	1.19E+06	5.75E+04	8.99E+04	3.59E+06	2.85E+06	5.45E+06	5.21E+06	5.21E+06	2.03E+06	**1.42E+03**	**1.88E+03**
13	8.52E+06	1.44E+07	1.08E+07	1.18E+07	8.00E+07	2.51E+08	3.21E+03	2.88E+03	1.53E+05	8.62E+04	9.58E+03	1.28E+04	2.02E+04	1.79E+04	2.40E+04	1.68E+04	1.80E+05	4.73E+05	1.99E+05	1.14E+05	**1.81E+02**	**2.68E+02**
14	8.57E+04	5.61E+04	1.16E+05	7.77E+04	5.41E+05	2.42E+06	2.32E+03	1.52E+03	3.00E+03	3.19E+03	2.11E+03	2.28E+03	7.10E+03	5.85E+03	2.51E+04	3.67E+04	7.38E+04	8.93E+04	1.04E+05	6.13E+04	**4.83E+01**	**1.07E+01**
15	4.38E+06	2.98E+06	3.89E+06	4.16E+06	1.64E+06	1.70E+06	2.13E+03	1.63E+03	6.55E+04	3.19E+04	1.06E+04	9.40E+03	2.16E+04	2.27E+04	8.89E+03	7.39E+03	3.96E+04	3.06E+04	7.34E+04	4.22E+04	**3.38E+01**	**3.22E+01**
16	1.62E+03	1.97E+02	1.80E+03	2.30E+02	1.60E+03	2.85E+02	8.46E+02	1.53E+02	5.08E+02	1.93E+02	6.66E+02	2.26E+02	4.92E+02	3.53E+02	1.48E+03	3.08E+02	1.11E+03	3.33E+02	7.21E+02	**1.40E+02**	**4.85E+02**	2.32E+02
17	5.76E+02	1.31E+02	6.79E+02	1.35E+02	4.60E+02	1.97E+02	3.31E+02	1.13E+02	2.83E+02	1.37E+02	2.55E+02	1.55E+02	1.41E+02	6.59E+01	7.29E+02	2.51E+02	5.53E+02	2.29E+02	2.10E+02	**6.31E+01**	**1.11E+02**	7.51E+01
18	1.81E+06	1.30E+06	2.78E+06	3.45E+06	2.45E+06	3.57E+06	6.99E+04	3.06E+04	1.07E+05	5.66E+04	1.25E+05	1.01E+05	3.67E+05	1.67E+05	2.70E+05	2.07E+05	8.54E+05	9.29E+05	5.09E+05	2.08E+05	**6.29E+01**	**3.93E+01**
19	4.40E+05	2.17E+06	3.40E+05	5.79E+05	7.49E+06	3.50E+07	1.71E+03	1.69E+03	7.46E+04	4.61E+04	1.37E+04	1.56E+04	1.07E+04	1.10E+04	2.98E+04	5.22E+04	9.01E+05	5.86E+06	7.37E+04	4.31E+04	**2.05E+01**	**7.55E+00**
20	6.12E+02	1.22E+02	5.88E+02	1.27E+02	5.68E+02	1.89E+02	3.48E+02	9.16E+01	3.23E+02	1.09E+02	2.55E+02	1.47E+02	2.21E+02	1.25E+02	5.73E+02	2.21E+02	4.66E+02	1.78E+02	2.12E+02	**8.08E+01**	**1.29E+02**	1.11E+02
21	4.21E+02	2.33E+01	4.02E+02	7.39E+01	4.27E+02	1.80E+01	3.05E+02	3.30E+01	2.40E+02	1.04E+01	2.98E+02	2.31E+01	**2.34E+02**	1.16E+01	4.06E+02	6.11E+01	3.66E+02	4.60E+01	3.10E+02	**8.09E+00**	2.50E+02	1.33E+01
22	6.23E+03	2.08E+03	7.68E+02	1.35E+02	5.77E+03	2.05E+03	**1.00E+02**	5.05E−07	1.48E+03	8.12E+02	**1.00E+02**	9.87E−01	1.01E+02	1.94E+00	3.03E+03	2.56E+03	3.41E+03	1.94E+03	3.62E+02	1.83E+02	1.29E+03	1.39E+03
23	6.02E+02	2.02E+01	6.29E+02	3.17E+01	6.69E+02	6.44E+01	6.81E+02	3.79E+01	**3.91E+02**	1.26E+01	4.53E+02	3.19E+01	3.96E+02	1.62E+01	6.10E+02	6.73E+01	4.96E+02	3.10E+01	4.55E+02	**8.91E+00**	4.00E+02	1.30E+01
24	6.65E+02	1.99E+01	6.86E+02	3.39E+01	7.38E+02	7.23E+01	7.39E+02	4.57E+01	**4.54E+02**	**8.09E+00**	5.65E+02	5.05E+01	4.69E+02	1.62E+01	6.78E+02	6.89E+01	5.53E+02	2.96E+01	5.56E+02	1.19E+01	4.75E+02	1.41E+01
25	4.72E+02	4.09E+01	4.81E+02	3.43E+01	6.46E+02	2.06E+02	**3.85E+02**	1.77E+00	3.87E+02	6.57E−01	3.86E+02	1.41E+00	4.02E+02	1.76E+01	4.07E+02	2.26E+01	4.09E+02	4.93E+01	3.88E+02	4.79E−01	3.87E+02	**1.19E−01**
26	3.80E+03	2.63E+02	2.78E+03	1.22E+03	4.54E+03	9.49E+02	2.04E+03	1.73E+03	**1.15E+03**	4.84E+02	2.17E+03	7.28E+02	1.42E+03	4.69E+02	3.38E+03	1.27E+03	2.78E+03	4.23E+02	2.06E+03	**8.90E+01**	1.54E+03	1.32E+02
27	5.40E+02	1.21E+01	5.55E+02	2.95E+01	5.83E+02	5.37E+01	7.08E+02	5.42E+01	5.04E+02	1.18E+01	5.39E+02	1.62E+01	5.32E+02	2.07E+01	**4.96E+02**	7.63E+00	5.34E+02	1.44E+01	5.15E+02	**2.87E+00**	5.03E+02	7.38E+00
28	8.66E+02	3.07E+02	8.33E+02	1.13E+02	1.39E+03	6.11E+02	**3.27E+02**	3.17E+01	3.85E+02	4.41E+01	3.87E+02	4.33E+01	4.30E+02	2.67E+01	4.33E+02	2.44E+01	5.06E+02	8.68E+01	4.56E+02	**7.57E+00**	3.32E+02	5.14E+01
29	1.43E+03	1.93E+02	1.47E+03	2.01E+02	1.30E+03	2.06E+02	7.80E+02	1.21E+02	6.70E+02	1.05E+02	7.48E+02	1.96E+02	6.15E+02	9.09E+01	1.45E+03	2.71E+02	1.04E+03	2.14E+02	7.09E+02	**6.62E+01**	**5.05E+02**	7.21E+01
30	5.98E+06	3.13E+06	8.54E+06	4.38E+06	3.19E+07	1.74E+08	3.32E+03	3.89E+02	4.72E+05	2.83E+05	3.31E+04	3.13E+04	2.57E+04	2.78E+04	7.18E+05	4.50E+05	2.42E+05	4.00E+05	5.07E+04	2.59E+04	**2.01E+03**	**1.56E+02**

Table 7
The results of eleven algorithms (50D).

Fun	Jaya		QO-Jaya		SAMP-Jaya		PPSO		IWO		DYYPO		TLBO-FL		OLBSO		EWWO		SE		SDH-Jaya	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
1	2.52E+10	5.52E+09	2.80E+10	5.29E+09	4.18E+10	1.18E+10	3.89E+02	2.95E+02	1.56E+04	9.19E+03	6.57E+03	7.09E+03	6.06E+05	2.21E+06	2.82E+05	1.72E+05	1.47E+08	7.37E+08	6.25E+04	6.93E+04	**5.01E−06**	**1.15E−05**
3	1.18E+05	2.12E+04	1.16E+05	1.79E+04	1.69E+05	5.12E+04	8.65E+02	1.86E+02	2.94E−02	4.58E−03	4.67E+01	2.30E+02	2.57E+04	4.88E+03	2.32E+01	8.06E+00	6.36E+04	8.79E+04	2.24E+05	1.92E+04	**0.00E+00**	**0.00E+00**
4	1.86E+03	6.50E+02	2.28E+03	7.25E+02	4.37E+03	1.62E+03	9.13E+01	3.61E+01	1.26E+02	5.04E+01	1.37E+02	4.97E+01	1.90E+02	4.58E+01	1.51E+02	4.72E+01	2.16E+02	7.89E+01	1.64E+02	**1.39E+01**	**6.93E+01**	4.42E+01
5	5.02E+02	3.30E+01	5.45E+02	3.62E+01	5.33E+02	4.69E+01	2.01E+02	1.37E+01	**8.68E+01**	1.58E+01	1.95E+02	3.95E+01	9.67E+01	1.73E+01	3.95E+02	1.03E+02	3.66E+02	9.04E+01	2.19E+02	**1.35E+01**	1.01E+02	2.43E+01
6	4.10E+01	6.36E+00	4.43E+01	6.75E+00	4.57E+01	6.68E+00	3.18E+01	3.91E+00	5.76E−01	9.32E−01	3.84E+00	2.03E+00	4.51E+00	1.70E+00	1.24E+01	4.57E+00	3.57E+01	1.34E+01	**6.19E−08**	**1.76E−08**	2.88E−07	2.90E−07
7	7.94E+02	8.02E+01	8.31E+02	9.14E+01	1.17E+03	2.37E+02	2.78E+02	3.42E+01	**1.32E+02**	1.57E+01	2.61E+02	4.27E+01	1.74E+02	4.38E+01	4.36E+02	9.00E+01	4.23E+02	8.53E+01	2.76E+02	**1.29E+01**	1.47E+02	2.00E+01
8	5.42E+02	3.04E+01	5.67E+02	3.13E+01	5.69E+02	4.26E+01	1.99E+02	**1.52E+01**	**8.37E+01**	1.87E+01	1.90E+02	4.70E+01	9.33E+01	1.58E+01	3.36E+02	9.26E+01	3.47E+02	8.59E+01	2.16E+02	1.59E+01	9.18E+01	2.11E+01
9	1.33E+04	4.29E+03	2.10E+04	6.50E+03	1.73E+04	5.25E+03	6.06E+03	7.28E+02	1.57E+00	2.49E+00	3.51E+03	1.86E+03	1.30E+03	1.05E+03	1.28E+04	4.78E+03	1.26E+04	4.62E+03	1.33E+03	3.86E+02	**3.08E−01**	**7.34E−01**
10	1.34E+04	3.45E+02	1.25E+04	4.45E+02	1.32E+04	5.30E+02	5.20E+03	5.51E+02	**3.17E+03**	5.98E+02	4.80E+03	6.41E+02	1.27E+04	3.97E+02	7.76E+03	8.50E+02	7.40E+03	1.38E+03	7.98E+03	**3.13E+02**	4.91E+03	5.98E+02
11	2.23E+03	7.60E+02	2.56E+03	7.46E+02	5.55E+03	3.40E+03	1.27E+02	**1.45E+01**	2.33E+02	4.44E+01	1.90E+02	5.19E+01	1.69E+02	4.79E+01	3.10E+02	8.46E+01	1.33E+03	5.92E+03	1.02E+03	4.95E+02	**6.61E+01**	3.00E+01
12	3.86E+09	1.42E+09	4.27E+09	1.01E+09	6.87E+09	2.56E+09	5.52E+05	2.03E+05	7.71E+06	4.48E+06	7.75E+06	5.08E+06	9.16E+05	8.94E+05	1.24E+07	5.88E+06	9.53E+07	2.76E+08	3.97E+07	9.86E+06	**3.01E+04**	**3.31E+04**
13	5.66E+08	3.73E+08	6.47E+08	2.48E+08	1.65E+09	2.16E+09	**8.47E+02**	**5.63E+02**	1.84E+05	1.08E+05	7.55E+03	7.38E+03	8.01E+03	5.14E+03	3.99E+04	2.49E+04	9.25E+05	4.32E+06	3.37E+05	1.65E+05	1.05E+03	6.21E+02
14	8.98E+05	1.13E+06	9.42E+05	6.36E+05	2.66E+06	5.88E+06	1.95E+04	1.00E+04	1.81E+04	2.09E+04	2.89E+04	2.75E+04	8.63E+04	4.96E+04	8.54E+04	7.16E+04	3.96E+05	3.02E+05	8.33E+05	4.09E+05	**1.26E+02**	**3.81E+01**
15	1.16E+08	7.72E+07	6.00E+07	5.49E+07	2.12E+08	3.85E+08	1.19E+03	7.80E+02	8.39E+04	5.52E+04	8.24E+03	7.00E+03	6.88E+03	5.91E+03	1.24E+04	1.04E+04	1.62E+06	1.00E+07	3.86E+04	2.44E+04	**2.18E+02**	**2.26E+02**
16	3.36E+03	2.57E+02	3.57E+03	2.84E+02	3.19E+03	5.29E+02	1.24E+03	2.28E+02	**7.70E+02**	2.10E+02	1.31E+03	4.08E+02	8.47E+02	3.05E+02	2.42E+03	5.35E+02	2.28E+03	4.63E+02	1.55E+03	**1.84E+02**	1.21E+03	4.19E+02
17	2.38E+03	2.14E+02	2.42E+03	2.45E+02	2.36E+03	4.11E+02	1.03E+03	**1.54E+02**	7.62E+02	1.97E+02	8.88E+02	2.65E+02	8.53E+02	4.12E+02	1.94E+03	3.78E+02	1.81E+03	4.27E+02	1.01E+03	1.80E+02	**7.38E+02**	2.37E+02
18	9.58E+06	4.88E+06	9.35E+06	5.06E+06	1.51E+07	4.55E+07	2.09E+05	8.67E+04	1.85E+05	7.99E+04	1.81E+05	8.14E+04	1.15E+06	5.13E+05	8.30E+05	6.53E+05	2.51E+06	2.78E+06	2.85E+06	1.28E+06	**3.36E+02**	**1.37E+02**
19	2.67E+07	2.77E+07	1.57E+07	2.72E+07	5.27E+07	1.84E+08	8.67E+03	3.91E+03	2.12E+05	1.06E+05	9.64E+03	8.85E+03	1.45E+04	9.40E+03	2.09E+04	2.14E+04	9.08E+05	2.92E+06	5.38E+04	1.87E+04	**1.06E+02**	**6.32E+01**
20	1.85E+03	1.87E+02	1.44E+03	2.88E+02	1.83E+03	2.46E+02	7.70E+02	1.92E+02	5.56E+02	1.40E+02	6.54E+02	2.79E+02	1.04E+03	3.87E+02	1.41E+03	3.85E+02	1.26E+03	3.56E+02	7.73E+02	**1.39E+02**	**4.57E+02**	2.56E+02
21	6.85E+02	3.18E+01	6.94E+02	3.58E+01	7.27E+02	4.37E+01	4.33E+02	2.14E+01	2.85E+02	1.74E+01	4.02E+02	4.05E+01	**2.81E+02**	1.52E+01	5.35E+02	1.03E+02	5.52E+02	6.45E+01	4.33E+02	1.55E+01	2.93E+02	**8.75E+00**
22	1.33E+04	**3.96E+02**	5.21E+03	4.11E+03	1.30E+04	6.88E+02	5.97E+03	9.89E+02	**3.71E+03**	6.59E+02	4.78E+03	2.04E+03	6.56E+03	6.40E+03	8.43E+03	1.47E+03	7.98E+03	1.27E+03	7.19E+03	2.78E+03	5.99E+03	1.06E+03
23	1.02E+03	5.04E+01	1.08E+03	5.84E+01	1.20E+03	1.23E+02	1.06E+03	7.09E+01	**5.11E+02**	2.08E+01	6.46E+02	5.66E+01	5.66E+02	3.40E+01	9.42E+02	1.13E+02	7.91E+02	5.66E+01	6.61E+02	**1.35E+01**	5.33E+02	1.43E+01
24	1.02E+03	4.28E+01	1.06E+03	6.22E+01	1.21E+03	1.19E+02	1.08E+03	7.07E+01	**5.67E+02**	**1.53E+01**	7.31E+02	7.36E+01	6.75E+02	4.75E+01	9.77E+02	1.06E+02	7.67E+02	5.74E+01	7.98E+02	1.74E+01	6.08E+02	2.35E+01
25	1.25E+03	2.10E+02	1.65E+03	4.24E+02	3.06E+03	1.70E+03	5.41E+02	2.77E+01	4.93E+02	2.33E+01	5.21E+02	3.03E+01	6.16E+02	2.68E+01	4.93E+02	4.08E+01	5.82E+02	7.65E+01	5.59E+02	9.06E+00	**4.81E+02**	**5.87E−01**
26	7.63E+03	7.82E+02	4.64E+03	1.57E+03	9.17E+03	1.71E+03	5.45E+03	2.59E+03	**1.82E+03**	4.22E+02	3.42E+03	7.82E+02	2.93E+03	6.02E+02	5.27E+03	2.22E+03	4.82E+03	5.94E+02	3.43E+03	**1.32E+02**	2.20E+03	1.71E+02
27	8.68E+02	9.27E+01	9.34E+02	9.39E+01	1.08E+03	1.69E+02	1.46E+03	1.70E+02	5.45E+02	2.74E+01	6.72E+02	7.26E+01	8.68E+02	1.76E+02	**5.05E+02**	2.03E+01	7.76E+02	8.97E+01	6.53E+02	**1.83E+01**	5.85E+02	4.88E+01
28	3.53E+03	1.74E+03	1.93E+03	4.46E+02	4.50E+03	1.71E+03	4.89E+02	1.79E+01	4.80E+02	2.28E+01	4.80E+02	2.49E+01	6.11E+02	4.22E+01	4.90E+02	**1.05E+01**	2.14E+03	1.99E+03	5.25E+02	1.28E+01	**4.78E+02**	2.43E+01
29	2.80E+03	6.44E+02	2.93E+03	4.79E+02	2.73E+03	5.98E+02	1.52E+03	2.09E+02	9.27E+02	1.78E+02	9.82E+02	3.06E+02	1.02E+03	2.48E+02	2.52E+03	4.75E+02	1.98E+03	5.02E+02	9.27E+02	**1.48E+02**	**7.16E+02**	1.66E+02
30	7.48E+07	6.46E+07	1.11E+08	7.31E+07	1.23E+08	2.40E+08	7.81E+05	**4.82E+04**	1.31E+07	2.77E+06	1.54E+06	3.23E+05	1.16E+06	3.15E+05	1.00E+07	2.42E+06	8.03E+06	7.64E+06	1.19E+06	1.84E+05	**6.47E+05**	7.57E+04

Table 8
The results of eleven algorithms (100D).

Fun	Jaya		QO-Jaya		SAMP-Jaya		PPSO		IWO		DYYPO		TLBO-FL		OLBSO		EWWO		SE		DH-Jaya	
	mean	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
1	1.11E+11	1.30E+10	1.19E+11	1.06E+10	1.65E+11	2.12E+10	1.08E+04	**3.31E+03**	5.24E+04	1.77E+04	1.43E+04	1.71E+04	8.59E+08	5.17E+08	1.26E+06	4.77E+05	2.03E+09	1.03E+10	1.05E+05	9.46E+04	**9.59E+03**	9.82E+03
3	4.12E+05	4.45E+04	2.59E+05	1.97E+04	4.90E+05	8.15E+04	7.25E+04	6.60E+03	1.11E+04	3.79E+03	3.45E+04	1.34E+04	1.77E+05	1.97E+04	3.60E+02	**6.87E+01**	1.64E+05	5.97E+04	5.81E+05	4.25E+04	**2.47E+02**	8.03E+02
4	1.31E+04	4.31E+03	1.53E+04	3.68E+03	2.46E+04	7.90E+03	2.40E+02	3.12E+01	2.38E+02	2.73E+01	2.59E+02	4.10E+01	6.99E+02	1.20E+02	2.40E+02	5.16E+01	4.78E+02	5.49E+02	2.66E+02	1.48E+01	**2.05E+02**	**8.60E+00**
5	1.27E+03	6.70E+01	1.30E+03	8.27E+01	1.34E+03	7.79E+01	5.20E+02	**2.31E+01**	2.30E+02	2.98E+01	4.53E+02	7.46E+01	3.41E+02	4.51E+01	9.57E+02	2.37E+02	7.50E+02	2.04E+02	6.29E+02	2.54E+01	**2.11E+02**	4.10E+01
6	6.84E+01	6.97E+00	7.89E+01	9.85E+00	7.17E+01	7.23E+00	4.01E+01	2.34E+00	3.78E+00	2.20E+00	1.35E+01	3.71E+00	1.96E+01	3.01E+00	1.57E+01	6.02E+00	5.75E+01	9.56E+00	**5.43E−08**	**1.19E−08**	1.82E−04	1.30E−04
7	2.71E+03	2.64E+02	2.43E+03	1.62E+02	3.74E+03	4.46E+02	7.38E+02	8.90E+01	3.49E+02	2.88E+01	6.59E+02	9.90E+01	7.80E+02	9.56E+01	9.42E+02	1.90E+02	1.37E+03	1.96E+02	7.44E+02	**2.87E+01**	**3.07E+02**	3.54E+01
8	1.34E+03	5.82E+01	1.37E+03	7.38E+01	1.45E+03	8.61E+01	5.82E+02	3.07E+01	2.38E+02	3.09E+01	4.78E+02	9.01E+01	3.71E+02	5.59E+01	9.20E+02	2.67E+02	8.24E+02	2.37E+02	6.19E+02	**2.27E+01**	**2.22E+02**	4.04E+01
9	5.57E+04	8.35E+03	6.72E+04	6.00E+03	6.20E+04	8.74E+03	1.48E+04	8.12E+02	7.23E+01	1.49E+02	1.40E+04	5.45E+03	1.97E+04	4.93E+03	2.81E+04	9.66E+03	3.94E+04	8.57E+03	2.06E+04	3.34E+03	**4.74E+01**	**4.02E+01**
10	3.02E+04	**5.19E+02**	2.82E+04	1.47E+03	3.00E+04	6.96E+02	1.14E+04	8.94E+02	**8.59E+03**	8.70E+02	1.16E+04	1.87E+03	2.90E+04	5.93E+02	1.61E+04	1.43E+03	1.62E+04	1.73E+03	1.91E+04	5.90E+02	1.23E+04	1.12E+03
11	4.41E+04	1.28E+04	5.61E+04	1.47E+04	9.10E+04	3.79E+04	8.96E+02	**7.09E+01**	1.41E+03	2.26E+02	1.14E+03	2.27E+02	1.05E+03	1.98E+02	1.45E+03	1.94E+02	8.89E+03	2.52E+04	4.21E+04	8.96E+03	**4.98E+02**	9.82E+01
12	1.98E+10	4.23E+09	2.30E+10	4.63E+09	3.83E+10	1.41E+10	4.12E+06	7.35E+05	2.91E+07	1.28E+07	3.35E+07	1.70E+07	2.72E+07	1.52E+07	2.70E+07	8.18E+06	5.10E+08	8.47E+08	1.09E+08	2.22E+07	**1.72E+05**	**8.65E+04**
13	2.66E+09	7.40E+08	3.17E+09	8.77E+08	5.00E+09	2.36E+09	**1.53E+03**	**7.24E+02**	1.08E+05	4.59E+04	1.49E+04	8.05E+03	1.40E+04	6.47E+03	3.07E+04	1.07E+04	3.54E+07	1.05E+08	1.76E+05	6.94E+04	5.53E+03	5.50E+03
14	1.61E+07	5.84E+06	1.73E+07	5.91E+06	1.42E+07	8.99E+06	3.14E+05	8.93E+04	1.32E+05	6.50E+04	2.05E+05	1.06E+05	1.92E+06	7.10E+05	4.49E+05	2.18E+05	2.47E+06	1.97E+06	9.61E+06	2.92E+06	**7.50E+02**	**1.24E+02**
15	8.01E+08	4.50E+08	9.92E+08	4.24E+08	1.55E+09	9.89E+08	**4.69E+02**	**1.58E+02**	9.10E+04	3.95E+04	7.46E+03	5.29E+03	3.73E+03	3.58E+03	1.12E+04	4.27E+03	4.81E+06	2.08E+07	9.44E+04	4.91E+04	4.61E+03	5.46E+03
16	9.45E+03	4.73E+02	9.60E+03	5.15E+02	9.65E+03	8.66E+02	3.18E+03	3.25E+02	**1.97E+03**	4.95E+02	2.77E+03	5.97E+02	2.57E+03	5.79E+02	4.98E+03	7.67E+02	5.93E+03	9.86E+02	4.07E+03	**2.89E+02**	3.61E+03	6.19E+02
17	8.54E+03	6.39E+02	8.83E+03	8.63E+02	1.18E+04	6.43E+03	2.63E+03	3.34E+02	**1.66E+03**	3.32E+02	2.11E+03	4.35E+02	2.19E+03	4.74E+02	4.02E+03	6.34E+02	5.22E+03	7.57E+02	2.96E+03	**2.70E+02**	2.24E+03	3.86E+02
18	3.06E+07	1.11E+07	3.06E+07	1.20E+07	2.95E+07	3.28E+07	6.89E+05	1.75E+05	2.93E+05	1.00E+05	3.72E+05	1.37E+05	4.59E+06	1.64E+06	7.19E+05	2.81E+05	4.01E+06	3.04E+06	8.72E+06	2.38E+06	**1.46E+04**	**1.12E+04**
19	9.76E+08	2.65E+08	1.06E+09	2.10E+08	1.31E+09	5.41E+08	**5.42E+02**	**3.33E+02**	9.12E+05	2.64E+05	7.49E+03	7.29E+03	4.15E+03	5.31E+03	1.18E+05	1.21E+05	4.45E+07	2.51E+08	2.04E+05	9.54E+04	3.28E+03	3.94E+03
20	5.23E+03	3.15E+02	4.89E+03	2.42E+02	5.15E+03	2.90E+02	2.35E+03	2.58E+02	**1.49E+03**	2.94E+02	2.09E+03	5.38E+02	4.23E+03	3.18E+02	4.03E+03	6.09E+02	3.50E+03	6.82E+02	2.68E+03	**1.90E+02**	2.21E+03	4.23E+02
21	1.52E+03	7.30E+01	1.58E+03	8.21E+01	1.67E+03	1.16E+02	1.06E+03	5.52E+01	4.68E+02	3.21E+01	6.94E+02	9.76E+01	5.99E+02	4.53E+01	1.21E+03	1.78E+02	1.15E+03	3.11E+02	8.56E+02	**2.32E+01**	**4.63E+02**	4.00E+01
22	3.10E+04	6.07E+02	2.37E+04	8.70E+03	3.08E+04	7.13E+02	1.37E+04	8.94E+02	**9.72E+03**	1.07E+03	1.31E+04	1.95E+03	2.95E+04	4.23E+03	1.77E+04	1.34E+03	1.78E+04	1.44E+03	2.02E+04	**5.78E+02**	1.37E+04	1.34E+03
23	1.98E+03	1.12E+02	2.10E+03	1.33E+02	2.37E+03	2.16E+02	2.07E+03	8.16E+01	7.66E+02	3.50E+01	9.01E+02	6.03E+01	1.17E+03	1.11E+02	1.53E+03	2.18E+02	1.31E+03	1.30E+02	8.37E+02	**1.67E+01**	**7.43E+02**	3.09E+01
24	2.66E+03	1.58E+02	2.95E+03	2.47E+02	3.57E+03	4.55E+02	1.89E+03	1.04E+02	**1.08E+03**	3.55E+01	1.38E+03	8.86E+01	2.02E+03	2.61E+02	1.98E+03	2.25E+02	1.82E+03	1.39E+02	1.43E+03	**2.25E+01**	1.12E+03	4.16E+01
25	7.98E+03	1.28E+03	9.18E+03	1.57E+03	1.28E+04	3.23E+03	7.59E+02	3.37E+01	**7.14E+02**	6.50E+01	7.69E+02	7.02E+01	1.32E+03	1.09E+02	7.20E+02	5.61E+01	1.01E+03	1.80E+02	9.24E+02	**1.62E+01**	7.60E+02	4.63E+01
26	2.33E+04	1.68E+03	2.52E+04	3.25E+03	2.92E+04	3.14E+03	1.56E+04	5.19E+03	**5.38E+03**	3.12E+02	8.03E+03	8.66E+02	1.09E+04	1.57E+03	1.38E+04	4.25E+03	1.39E+04	1.41E+03	9.38E+03	**2.35E+02**	5.93E+03	4.33E+02
27	1.84E+03	2.33E+02	1.93E+03	2.58E+02	2.27E+03	3.43E+02	1.34E+03	9.82E+01	6.40E+02	3.28E+01	7.68E+02	4.97E+01	1.17E+03	1.47E+02	**5.00E+02**	**2.19E−04**	1.04E+03	1.68E+02	7.64E+02	2.13E+01	6.71E+02	2.52E+01
28	1.76E+04	1.99E+03	1.33E+04	1.49E+03	2.05E+04	3.02E+03	5.86E+02	1.49E+01	5.56E+02	3.27E+01	6.15E+02	4.00E+01	1.48E+03	2.70E+02	**5.00E+02**	**8.74E+00**	1.10E+04	5.41E+03	7.04E+02	1.54E+01	5.47E+02	3.56E+01
29	9.52E+03	8.17E+02	9.86E+03	6.87E+02	1.28E+04	1.44E+04	3.73E+03	2.98E+02	**2.51E+03**	3.31E+02	2.85E+03	4.99E+02	3.44E+03	4.97E+02	5.45E+03	6.05E+02	5.94E+03	1.07E+03	3.18E+03	**2.48E+02**	**2.51E+03**	3.58E+02
30	1.75E+09	6.10E+08	1.87E+09	5.59E+08	3.14E+09	1.42E+09	7.17E+03	1.17E+03	6.37E+06	2.80E+06	1.18E+06	8.20E+05	6.18E+04	6.36E+04	1.46E+06	4.94E+05	4.12E+07	7.85E+07	4.82E+04	1.27E+04	**2.93E+03**	**2.09E+02**

Table 9
Friedman test ($10D$).

Algorithm	Mean rank
Jaya	7.930
QO-Jaya	8.690
SAMP-Jaya	9.410
PPSO	3.550
IWO	5.530
TLBO-FL	4.120
DYYPO	3.500
OLBSO	8.690
EWWO	6.070
SE	6.000
SDH-Jaya	2.500
Crit. Diff. $\alpha = 0.05$	2.576
Crit. Diff. $\alpha = 0.1$	2.327

Table 10
Friedman test ($30D$).

Algorithm	Mean rank
Jaya	9.140
QO-Jaya	9.480
SAMP-Jaya	10.090
PPSO	4.290
IWO	3.570
TLBO-FL	3.550
DYYPO	4.260
OLBSO	6.970
EWWO	7.220
SE	5.550
SDH-Jaya	1.880
Crit. Diff. $\alpha = 0.05$	2.576
Crit. Diff. $\alpha = 0.1$	2.327

Table 11
Friedman test ($50D$).

Algorithm	Mean rank
Jaya	9.360
QO-Jaya	9.380
SAMP-Jaya	10.550
PPSO	4.760
IWO	2.760
TLBO-FL	4.530
DYYPO	3.780
OLBSO	6.190
EWWO	7.210
SE	5.760
SDH-Jaya	1.720
Crit. Diff. $\alpha = 0.05$	2.576
Crit. Diff. $\alpha = 0.1$	2.327

Table 12
Friedman test ($100D$).

Algorithm	Mean rank
Jaya	9.330
QO-Jaya	9.810
SAMP-Jaya	10.660
PPSO	4.170
IWO	2.710
TLBO-FL	5.380
DYYPO	3.690
OLBSO	5.400
EWWO	7.140
SE	5.760
DH-Jaya	1.970
Crit. Diff. $\alpha = 0.05$	2.576
Crit. Diff. $\alpha = 0.1$	2.327

and 95 except for the IWO in Table 13. From Tables 9–12, the results of the Friedman test, the SDH-Jaya outperforms the other ten algorithms in the dimensional of 10, 30, 50, and DH-Jaya outperforms the other ten algorithms in the $100D$. From the convergence curves shown in Figs. 7, 9, 10, 12, and 15, the convergence speed of SDH Jaya and DH Jaya is faster than that of the other ten comparison algorithms. From the boxplots shown in Figs. 8, 11, 13, 14, and 16, the stability of SDH Jaya and DH Jaya is better than the other ten comparison algorithms. For reducing the fitness evaluations, in this experiment, the parameters of SDH-Jaya are set as $p = 100$, $SMs = 30$, $Ms = 10$. Supporting the using time of the surrogate model is num_s. For the same optimization results, the fitness evaluations of SDH-Jaya are $500num_s$ less than DH-Jaya. In terms of the quality of solution and execution time, the experimental results reveal that the effectiveness of SDH-Jaya and DH-Jaya algorithm outperforms classical Jaya algorithm, the variants of Jaya algorithm, and state-of-the-art algorithms.

6. SDH-Jaya and DH-Jaya for engineering problems

In this section, two categories of engineering optimization problems, the continuous engineering optimization problem and the discrete engineering optimization problem, are utilized to test the performance of the SDH-Jaya. In addition, the performance of the SDH-Jaya and DH-Jaya are compared with classical methods to analyze the advantages of the SDH-Jaya and DH-Jaya in addressing engineering optimization problems. For each engineering problem, the algorithm is run independently 10 times.

6.1. Gear train engineering design problem

In this section, the gear train engineering design problem is utilized to verify the performance of SDH-Jaya in addressing engineering problems. There are four types of parameters in the gear train engineering design problem as shown in Fig. 21. The optimization problem is described as follows. More details on Sandgren, GeneAS, and ABC are provided in (Sadollah et al., 2013). Jaya, QO-Jaya runs 10000 fitness evaluations and SDH-Jaya runs 1000 fitness evaluations. It is observed from Table 14 that the SDH-Jaya outperforms the other five algorithms.

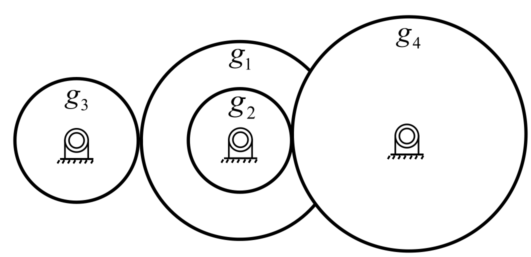


Fig. 21. Geat train design problem.

Consider $\vec{g} = [g_1, g_2, g_3, g_4] = [M_A, M_B, M_C, M_D]$
Minimize $f(\vec{g}) = (\frac{1}{6.931} - \frac{g_2 g_3}{g_1 g_4})^2$
Subjected to: $12 \le g_1, g_2, g_3, g_4 \le 60$

6.2. The no-idle flowshop scheduling problem

The no-idle flow shop scheduling problem (NIFSP) (Zhao et al., 2020a) is one of the NP-hard problems described as follows. There are N jobs processed in M machines. The processing sequence of N jobs is the same as on M machines. No idle time is required between two jobs processed on the same machine. There is an infinite buffer between machines, one job is not processed by more than one machine at the same time, a machine does not process more than one job at the same time. The processing time of the jobs on each machine is known.

Table 13
Rankings obtained through the Wilcoxon test.

D		vs	+	−	=	R^{+}	R^{-}	Z	*p-value*	$\alpha = 0.05$	$\alpha = 0.1$
		Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		QO-Jaya	27	2	0	4.17E+02	1.80E+01	−4.31E+00	1.60E−05	Yes	Yes
		SAMP-Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		PPSO	20	7	2	2.65E+02	1.14E+02	−1.81E+00	6.97E−02	No	Yes
10	SDH-Jaya	IWO	24	5	0	3.76E+02	5.95E+01	−3.42E+00	6.34E−04	Yes	Yes
		DYYPO	23	6	0	3.25E+02	1.10E+02	−2.32E+00	2.01E−02	Yes	Yes
		TLBO-FL	22	6	1	3.34E+02	7.25E+01	−2.97E+00	2.96E−03	Yes	Yes
		OLBSO	27	2	0	4.20E+02	1.55E+01	−4.37E+00	1.30E−05	Yes	Yes
		EWWO	25	4	0	3.95E+02	4.00E+01	−3.84E+00	1.24E−04	Yes	Yes
		SE	19	10	0	2.88E+02	1.47E+02	−1.52E+00	1.27E−01	No	No
		Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		QO-Jaya	28	1	0	4.22E+02	1.30E+01	−4.42E+00	1.00E−05	Yes	Yes
		SAMP-Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		PPSO	25	4	0	4.05E+02	3.00E+01	−4.05E+00	5.00E−05	Yes	Yes
30	SDH-Jaya	IWO	20	8	1	3.22E+02	8.40E+01	−2.71E+00	6.73E−03	Yes	Yes
		DYYPO	27	2	0	4.12E+02	2.30E+01	−4.21E+00	2.60E−05	Yes	Yes
		TLBO-FL	22	7	0	3.75E+02	6.00E+01	−3.41E+00	6.60E−04	Yes	Yes
		OLBSO	28	1	0	4.34E+02	1.00E+00	−4.68E+00	3.00E−06	Yes	Yes
		EWWO	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		SE	27	2	0	4.15E+02	2.00E+01	−4.27E+00	1.90E−05	Yes	Yes
		Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		QO-Jaya	28	1	0	4.26E+02	9.00E+00	−4.51E+00	7.00E−06	Yes	Yes
		SAMP-Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		PPSO	27	2	0	4.21E+02	1.40E+01	−4.40E+00	1.10E−05	Yes	Yes
50	SDH-Jaya	IWO	18	11	0	2.94E+02	1.41E+02	−1.65E+00	9.81E−02	No	Yes
		DYYPO	27	2	0	4.05E+02	3.00E+01	−4.05E+00	5.00E−05	Yes	Yes
		TLBO-FL	26	3	0	4.14E+02	2.10E+01	−4.25E+00	2.10E−05	Yes	Yes
		OLBSO	28	1	0	4.30E+02	5.00E+00	−4.60E+00	4.00E−06	Yes	Yes
		EWWO	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		SE	28	1	0	4.34E+02	1.00E+00	−4.68E+00	3.00E−06	Yes	Yes
		Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		QO-Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		SAMP-Jaya	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		PPSO	22	6	1	3.20E+02	8.60E+01	−2.66E+00	7.72E−03	Yes	Yes
100	DH-Jaya	IWO	19	9	1	2.80E+02	1.26E+02	−1.75E+00	7.95E−02	No	Yes
		DYYPO	24	5	0	3.72E+02	6.30E+01	−3.34E+00	8.35E−04	Yes	Yes
		TLBO-FL	26	3	0	4.03E+02	3.20E+01	−4.01E+00	6.00E−05	Yes	Yes
		OLBSO	26	3	0	4.22E+02	1.30E+01	−4.42E+00	1.00E−05	Yes	Yes
		EWWO	29	0	0	4.35E+02	0.00E+00	−4.70E+00	3.00E−06	Yes	Yes
		SE	28	1	0	4.34E+02	1.00E+00	−4.68E+00	3.00E−06	Yes	Yes

The complexity of the NIFSP is $N!$. The integer programming model is shown as follows. The problem features are described in Fig. 22.

Consider $x_{j,k} = \begin{cases} 1, & if\ j\ is\ the\ kth\ job\ of\ \pi \\ 0, & else \end{cases}, j,k \in \{1,2,\ldots,n\}$

Minimize $min(Cmax) = min(\max_{k\in\{1,2,\ldots,n\}} C_{k,m})$

Subjected to:

$$\sum_{k=1}^{n} X_{j,k} = 1, j \in \{1,2,\ldots,n\}$$

$$\sum_{j=1}^{n} X_{j,k} = 1, k \in \{1,2,\ldots,n\}$$

$$C_{1,1} = \sum_{j=1}^{n} X_{j,1} \cdot p_{j,1}$$

$$C_{k+1,i} = C_{k.i} + \sum_{j=1}^{n} X_{j,k+1} \cdot p_{j,i}, \quad k \in \{1,2,\ldots,n-1\}, i \in \{1,2,\ldots,m\}$$

$$C_{k,i+1} \geq C_{k,i} + \sum_{j=1}^{n} X_{j,k} \cdot p_{j,i+1}, \quad k \in \{1,2,\ldots,n\}, i \in \{1,2,\ldots,m-1\}$$

$$C_{k,i} \geq 0, k \in \{1,2,\ldots,n\}, i \in \{1,2,\ldots,m\}$$

$$X_{j,k} \in 0, k, j \in \{1,2,\ldots,n\}$$

where π is the processing sequence. $C_{k,i} is$ the completion time of kth job in ith machine. $p_{j,i}$ is the processing time of jth job in ith machine, $k \in \{1,2,\ldots,n\}, i \in \{1,2,\ldots,m\}, j \in \{1,2,\ldots,n\}$.

The average relative percentage deviation (ARPD) is utilized to measure the performance of the algorithm and the calculation method is shown in Eq. (11). As the ARPD value is smaller, the better the algorithm performance. The DH-Jaya with minimum ARPD outperforms other algorithms. The DH-Jaya runs for $10M \cdot N$ milliseconds. The comparison algorithms are NEH (Ham, 1983), Saasani–Guinet–Moalla (SGM) (Saadani et al., 2005), hybrid discrete teaching-learning based meta-heuristic (HDTLM) (Shao et al., 2018), and hybrid discrete water wave optimization (HWWO) (Zhao et al., 2020b). The test problem is the Taillard benchmarks (Taillard, 1993) which include problems of different scales. $TA_1 - TA_{60}$ are selected to be tested from the Taillard benchmarks. The scale of $TA_1 - TA_{10}$ is 20×5, the scale of $TA_{11} - TA_{20}$ is 20×10, the scale of $TA_{21} - TA_{30}$ is 20×20, the scale of $TA_{31} - TA_{40}$ is 50×5, the scale of $TA_{41} - TA_{50}$ is 50×10, and the scale of $TA_{51} - TA_{60}$ is 50×20. The ARPD results of NEH, SGM, HDTLM, HWWO, and DH-Jaya are shown in Tables 15 and 16. The best results of the five algorithms are highlighted in bold.

$$ARPD = (\sum_{i=1}^{R} (C_i - C_{opt})/C_{opt} \cdot 100)/R \quad (11)$$

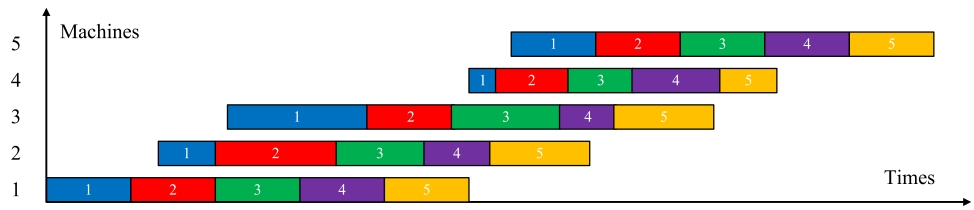


Fig. 22. The Gantt chart of NIFSP.

Table 14
The results of six algorithms.

Algorithms	Sandgren	GeneAS	ABC	Jaya	QO-Jaya	SDH-Jaya
g_1	6.00E+01	5.00E+01	4.90E+01	6.00E+01	6.00E+01	6.00E+01
g_2	4.50E+01	3.00E+01	1.60E+01	4.33E+01	4.33E+01	4.33E+01
g_3	2.20E+01	1.40E+01	1.90E+01	1.20E+01	1.20E+01	1.20E+01
g_4	1.80E+01	1.70E+01	4.30E+01	6.00E+01	6.00E+01	6.00E+01
$f(\vec{g})$	1.47E−01	1.44E−01	2.70E−12	1.08E−19	1.33E−13	**7.57E−22**

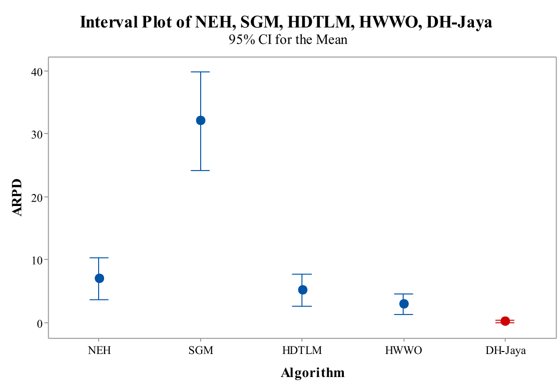


Fig. 23. Interval plot of ARPD of five algorithms.

Table 15
The ARPD results of five algorithms on Taillard benchmark.

$N \times M$	NEH	SGM	HDTLM	HWWO	DH-Jaya
20 × 5	7.66	34.61	5.42	4.49	**0.24**
20 × 10	9.92	36.58	7.65	3.19	**0.27**
20 × 20	8.73	33.22	6.21	2.78	**0.42**
50 × 5	1.01	17.01	0.56	0.22	**0.00**
50 × 10	5.95	34.08	4.77	2.66	**0.00**
50 × 20	8.29	36.57	6.27	4.18	**0.00**

where C_i represents the solution generated by the specific algorithm in the ith experiment of the given instance. R is the number of runs. C_{opt} is the minimum makespan found by all algorithms.

6.3. Analysis and discussion

The comparison of gear train engineering design problems is shown in Table 14 which indicates that the SDH-Jaya has a desirable performance. The results of the computation of the five algorithms in the Taillard benchmark are recorded in Tables 15 and 16. In terms of the ARPD, the ARPD value of DH-Jaya is lower than NEH, SGM, HDTLM, and HWWO. The ARPD of the results obtained by the four algorithms on 60 instances respectively in Fig. 24. The ARPD value of DH-Jaya is lower than the other four comparison algorithms. The same conclusion is obtained from Fig. 23 where the ARPD values for DH-Jaya are the smallest compared to the other four algorithms. It is indicative that solving NIFSP, DH-Jaya performs better than the four other algorithms. As a consequence, the DH-Jaya is utilized to address the discrete problem effectively.

According to the above experimental results, the SDH-Jaya and the DH-Jaya outperform other classical algorithms in the continuous engineering optimization problem and the discrete engineering optimization problem. The feature of algorithm-specific parameter-free properties in the SDH-Jaya and DH-Jaya makes the algorithms simply to be applied in addressing various engineering optimization problems. In addressing the continuous engineering optimization problem, optimization results are found with fewer fitness evaluations than other algorithms in SDH-Jaya. The DH-Jaya performs better than the NEH, SGM, HDTLM, and HWWO when solving the NIFSP. The proposed algorithms SDH-Jaya and DH-Jaya are simple and effective in addressing various engineering optimization problems in this study.

7. Conclusions and future work

The surrogate-assisted Jaya algorithm, which is based on optimal directional guidance and historical learning mechanism (SDH-Jaya), is introduced to improve the performance of the classical Jaya algorithm for optimization problems in this paper. Optimal directional guidance, historical learning mechanism, and an ensemble surrogate model are introduced. In addition, the assisted co-evolutionary mechanism and the self-learning co-evolutionary mechanism are proposed. The experimental results show that the SDH-Jaya algorithm has the better performance compared to the classical Jaya algorithm, its variants, and state-of-the-art algorithms. The accuracy of the solution, the convergence speed, and the stability of the algorithm are improved. The costs are saved by SDH-Jaya for solving expensive optimization problems. The results confirm that the efficiency and quality of the proposed algorithm are superior to the classical Jaya algorithm and its variants. In conclusion, the proposed surrogate-guided Jaya algorithm is a fast, robust, and trustworthy algorithm for solving single-objective continuous optimization problems.

The self-learning co-evolutionary mechanism based on stratification will be adopted to improve the performance of the basic Jaya algorithm. A new sampling method will be researched to construct an accurate surrogate model with as few sampling points as possible. In addition, the sorting model and classification model are combined to preprocess the candidates.

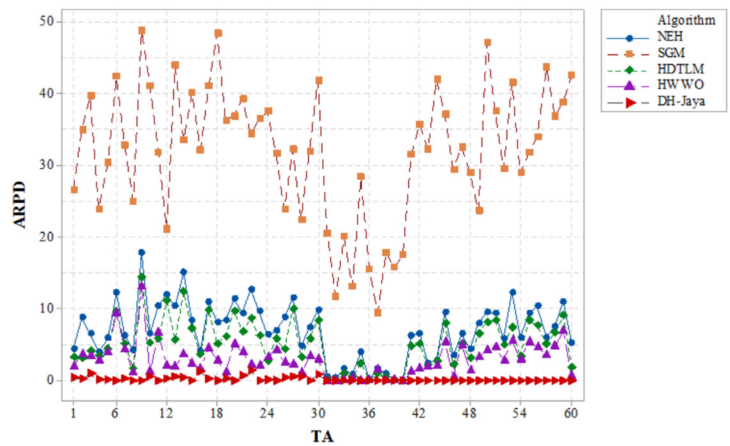


Fig. 24. The ARPD of five algorithms on 60 instances.

Table 16
The ARPD results of NEH, SGM, HDTLM, HWWO, and DH-Jaya.

TA	NEH	SGM	HDTLM	HWWO	DH-Jaya	TA	NEH	SGM	HDTLM	HWWO	DH-Jaya
1	4.43	26.61	3.18	1.99	**0.30**	31	0.46	20.54	0.23	**0.00**	**0.00**
2	8.78	35.07	3.02	3.62	**0.25**	32	0.38	11.74	0.06	**0.00**	**0.00**
3	6.47	39.70	4.07	3.40	**0.93**	33	1.59	20.10	0.88	0.14	**0.00**
4	4.01	23.90	3.34	2.74	**0.03**	34	0.82	13.05	0.49	0.33	**0.00**
5	5.90	30.45	4.44	4.01	**0.13**	35	3.97	28.36	2.37	**0.00**	**0.00**
6	12.23	42.46	9.70	9.39	**0.00**	36	0.10	15.60	0.10	0.10	**0.00**
7	6.28	32.81	5.11	4.44	**0.26**	37	1.71	9.42	0.92	1.48	**0.00**
8	4.21	25.04	1.72	1.02	**0.00**	38	0.96	17.89	0.50	0.10	**0.00**
9	17.78	48.85	14.40	13.12	**0.00**	39	0.07	15.78	0.04	0.04	**0.00**
10	6.53	41.22	5.26	1.21	**0.53**	40	**0.00**	17.59	**0.00**	**0.00**	**0.00**
11	10.45	31.90	5.77	6.65	**0.00**	41	6.28	31.56	4.82	1.17	**0.00**
12	12.03	21.10	11.06	2.07	**0.26**	42	6.58	35.74	5.08	1.68	**0.00**
13	10.34	43.99	5.72	1.87	**0.44**	43	2.38	32.26	2.10	1.87	**0.00**
14	15.07	33.17	12.34	3.71	**0.33**	44	2.82	42.09	2.60	2.03	**0.00**
15	8.37	36.94	7.23	2.37	**0.00**	45	9.48	37.20	7.94	5.45	**0.00**
16	4.14	30.68	3.68	1.64	**1.17**	46	3.47	29.44	2.19	0.54	**0.00**
17	10.93	35.12	9.76	4.54	**0.27**	47	6.47	32.61	5.11	5.00	**0.00**
18	8.12	44.50	5.15	2.80	**0.00**	48	4.40	29.05	3.15	1.40	**0.00**
19	8.40	35.98	6.13	1.12	**0.25**	49	8.01	23.68	6.53	3.18	**0.00**
20	11.32	30.23	9.68	5.11	**0.00**	50	9.56	47.20	8.16	4.23	**0.00**
21	9.40	39.35	6.86	3.97	**0.64**	51	9.33	37.53	8.40	4.67	**0.00**
22	12.69	34.42	8.66	2.23	**1.35**	52	5.83	29.52	5.00	2.83	**0.00**
23	9.62	36.55	6.22	2.08	**0.00**	53	12.26	41.63	7.32	5.54	**0.00**
24	6.34	37.55	2.72	3.16	**0.11**	54	6.00	28.98	3.32	2.96	**0.00**
25	6.91	31.80	5.82	4.29	**0.00**	55	9.37	31.81	8.38	5.31	**0.00**
26	8.79	23.79	4.35	2.55	**0.30**	56	10.32	34.04	7.64	4.67	**0.00**
27	11.51	32.32	9.95	2.23	**0.49**	57	6.01	43.71	5.12	3.55	**0.00**
28	4.77	22.44	3.21	1.02	**0.57**	58	7.55	36.92	6.62	4.78	**0.00**
29	7.41	32.04	5.86	3.30	**0.00**	59	11.00	38.90	9.09	6.91	**0.00**
30	9.82	41.92	8.39	2.95	**0.75**	60	5.21	42.67	1.83	0.59	**0.00**

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by the Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), and Project of Gansu Natural Science Foundation (21JR7RA204), respectively. All authors approved the version of the manuscript to be published.

References

Amouzgar, K., Bandaru, S., Ng, A.H.C., 2018. Radial basis functions with a priori bias as surrogate models:A comparative study. Eng. Appl. Artif. Intell. 71, 28–44. http://dx.doi.org/10.1016/j.engappai.2018.02.006.

Bhattacharya, A., Chattopadhyay, P.K., 2010. Biogeography-based optimization for different economic load dispatch problems. IEEE Trans. Power Syst 25, 1064–1077. http://dx.doi.org/10.1109/tpwrs.2009.2034525.

Bilal Pant, M., Zaheer, H., Garcia-Hernandez, L., Abraham, A., 2020. Differential evolution: A review of more than two decades of research. Eng. Appl. Artif. Intell. 90, 103479. http://dx.doi.org/10.1016/j.engappai.2020.103479.

Cai, X., Gao, L., Li, X., 2020. Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. Ieee Trans. Evol. Comput 24, 365–379. http://dx.doi.org/10.1109/tevc.2019.2919762.

Cai, X., Gao, L., Li, X., Qiu, H., 2019. Surrogate-guided differential evolution algorithm for high dimensional expensive problems. Swarm Evol. Comput 48, 288–311. http://dx.doi.org/10.1016/j.swevo.2019.04.009.

Cai, X., Qiu, H., Gao, L., Shao, X., 2017a. Metamodeling for high dimensional design problems by multi-fidelity simulations. Struct. Multidiscip. Optim 56, 151–166. http://dx.doi.org/10.1007/s00158-017-1655-y.

Cai, X., Qiu, H., Gao, L., Wei, L., Shao, X., 2017b. Adaptive radial-basis-function-based multifidelity metamodeling for expensive black-box problems. Aiaa J 55, 2424–2436. http://dx.doi.org/10.2514/1.J055649.

Chen, X., Cao, W., Gan, C., Ohyama, Y., She, J., Wu, M., 2021. Semi-supervised support vector regression based on data similarity and its application to rock-mechanics parameters estimation. Eng. Appl. Artif. Intell. 104, 104317. http://dx.doi.org/10.1016/j.engappai.2021.104317.

Chugh, T., Jin, Y.C., Miettinen, K., Hakanen, J., Sindhya, K., 2018. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. Ieee Trans. Evol. Comput 22, 129–142. http://dx.doi.org/10.1109/tevc.2016.2622301.

Deng, W., Xu, J.J., Zhao, H.M., 2019. An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem. IEEE Access 7, 20281–20292. http://dx.doi.org/10.1109/access.2019.2897580.

Emmerich, M.T.M., Giannakoglou, K.C., Naujoks, B., 2006. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. Ieee Trans. Evol. Comput 10, 421–439. http://dx.doi.org/10.1109/TEVC.2005.859463.

Gao, Kaizhou, Yang, Fa jun, Zhou, MengChu, Pan, Quanke, Suganthan, Nagaratnam, P., 2019. Flexible job-shop rescheduling for new job insertion by using discrete jaya algorithm. IEEE Trans. Cybern..

Goel, T., Haftka, R.T., Shyy, W., Queipo, 2007. Ensemble of surrogates. Struct. Multidiscip. Optim 33, 199–216. http://dx.doi.org/10.1007/s00158-006-0051-9.

Ham, J.I., 1983. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega.

Huang, D., Allen, T.T., Notz, W.I., Zeng, N., 2006. Global optimization of stochastic black-box systems via sequential Kriging meta-models. J. Glob. Optim 34, 441–466. http://dx.doi.org/10.1007/s10898-005-2454-3.

Jin, Y., 2011. Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm Evol. Comput 1, 61–70. http://dx.doi.org/10.1016/j.swevo.2011.05.001.

Jin, Y., Olhofer, M., Sendhoff, B., 2001. Managing approximate models in evolutionary aerodynamic design optimization. http://dx.doi.org/10.1109/CEC.2001.934445.

Jin, Y., Olhofer, M., Sendhoff, B., 2002. A framework for evolutionary optimization with approximate fitness functions. Ieee Trans. Evol. Comput 6, 481–494. http://dx.doi.org/10.1109/TEVC.2002.800884.

Johnson, M.E., Moore, L.M., Ylvisaker, D., 1990. Minimax and maximin distance designs. J. Statist. Plann. Inference 26, 131–148. http://dx.doi.org/10.1016/0378-3758(90)90122.

Jqla, B., Jwd, A., Cyl, A., Yyh, A., Jie, T.B., Bz, A., Cgw, A., 2020. An improved jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times - ScienceDirect. Knowl-Based Syst. 200.

Kitayama, S., Arakawa, M., Yamazaki, K., 2011. Sequential approximate optimization using radial basis function network for engineering optimization. Optim. Eng 12, 535–557. http://dx.doi.org/10.1007/s11081-010-9118-y.

Kommadath, R., Kotecha, P., Ieee, 2017. Teaching Learning Based Optimization with Focused Learning and Its Performance on CEC2017 Functions, 2017 Ieee Congress on Evolutionary Computation. Ieee, New York.

Liang, Y., Ren, Z., Yao, X., Feng, Z., Chen, A., Guo, W., 2020. Enhancing Gaussian estimation of distribution algorithm by exploiting evolution direction with archive. IEEE Trans. Cybern 50, 140–152. http://dx.doi.org/10.1109/TCYB.2018.2869567.

Liu, B., Zhang, Q., Gielen, G.G.E., 2014. A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. Ieee Trans. Evol. Comput 18, 180–192. http://dx.doi.org/10.1109/tevc.2013.2248012.

Long, W., Wu, T., Jiao, J., Tang, M., Xu, M., 2020. Refraction-learning-based whale optimization algorithm for high-dimensional problems and parameter estimation of PV model. Eng. Appl. Artif. Intell. 89, 103457. http://dx.doi.org/10.1016/j.engappai.2019.103457.

Ma, L., Cheng, S., Shi, Y., 2020. Enhancing learning efficiency of brain storm optimization via orthogonal learning design. IEEE Trans. Syst. Man, Cybern. Syst. 1–20.

Maharana, D., Kommadath, R., Kotecha, P., Ieee, 2017. Dynamic Yin-Yang Pair Optimization and Its Performance on Single Objective Real Parameter Problems of CEC 2017, 2017 Ieee Congress on Evolutionary Computation. Ieee, New York.

Mehrabian, A.R., Lucas, C., 2006. A novel numerical optimization algorithm inspired from weed colonization. Ecol. Inform 1, 355–366.

Mishra, A.K., Shrivastava, D., Bundela, B., Sircar, S., 2020. An efficient jaya algorithm for multi-objective permutation flow shop scheduling problem. Adv. Eng Opt Through Intell Tech.

Rao, R.V., More, K.C., 2017. Design optimization and analysis of selected thermal devices using self-adaptive Jaya algorithm. Energy Convers. Manag 140, 24–35. http://dx.doi.org/10.1016/j.enconman.2017.02.068.

Rao, R.V., Saroj, A., 2019. An elitism-based self-adaptive multi-population Jaya algorithm and its applications. Soft Comput 23, 4383–4406. http://dx.doi.org/10.1007/s00500-018-3095-z.

Ratle, A., 1998. Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (Eds.), Parallel Problem Solving from Nature — PPSN V. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 87–96.

Ravipudi, J.L., Neebha, M., 2018. Synthesis of linear antenna arrays using jaya, self -adaptive jaya and chaotic jaya algorithms. Aeu-International J. Electron. Commun 92, 54–63. http://dx.doi.org/10.1016/j.aeue.2018.05.022.

Saadani, N., Guinet, A., Moalla, M., 2005. A travelling salesman approach to solve the F/no-idle/Cmax problem. Eur. J. Oper. Res 161, 11–20.

Sadollah, A., Bahreininejad, A., Eskandar, H., Hamdi, M., 2013. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. Appl. Soft Comput 13, 2592–2612. http://dx.doi.org/10.1016/j.asoc.2012.11.026.

Shao, W., Pi, D., Shao, Z., 2018. A hybrid discrete teaching-learning based meta-heuristic for solving no-idle flow shop scheduling problem with total tardiness criterion. Comput. Oper. Res 94, 89–105. http://dx.doi.org/10.1016/j.cor.2018.02.003.

Shao, Z., Pi, D., Shao, W., Yuan, P., 2019. An efficient discrete invasive weed optimization for blocking flow-shop scheduling problem. Eng. Appl. Artif. Intell. 78, 124–141. http://dx.doi.org/10.1016/j.engappai.2018.11.005.

Simon, D., 2008. Biogeography-based optimization. Ieee Trans. Evol. Comput 12, 702–713. http://dx.doi.org/10.1109/tevc.2008.919004.

Solow, A., Polasky, S., Broadus, J., 1993. On the measurement of biological diversity. J. Environ. Econ. Manage 24, 60–68. http://dx.doi.org/10.1006/jeem.1993.1004.

Son, N.N., Van Kien, C., Anh, H.P.H., 2020. Parameters identification of Bouc–Wen hysteresis model for piezoelectric actuators using hybrid adaptive differential evolution and Jaya algorithm. Eng. Appl. Artif. Intell. 87, 103317. http://dx.doi.org/10.1016/j.engappai.2019.103317.

Suzuki, J., Man, J.S., on, C.I.T., 1995. A Markov chain analysis on simple genetic algorithms. 25, pp. 655–659.

Taillard, E.D., 1993. Benchmarks for basic scheduling problems. Eur. J. Oper. Res 64, 278–285.

Tang, D., 2019. Spherical evolution for solving continuous optimization problems. Appl. Soft Comput. J 81, 105499. http://dx.doi.org/10.1016/j.asoc.2019.105499.

Tang, K., Li, Z., Luo, L., Liu, B., 2015. Multi-strategy adaptive particle swarm optimization for numerical optimization. Eng. Appl. Artif. Intell. 37, 9–19. http://dx.doi.org/10.1016/j.engappai.2014.08.002.

Tangherloni, A., Rundo, L., Nobile, M.S., 2017. Proactive particles in swarm optimization: A settings-free algorithm for real-parameter single objective optimization problems. In: 2017 IEEE Congress on Evolutionary Computation (CEC). pp. 1940–1947. http://dx.doi.org/10.1109/CEC.2017.7969538.

Venkata Rao, R., 2016. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int. J. Ind. Eng. Comput 7, 19–34. http://dx.doi.org/10.5267/j.ijiec.2015.8.004.

Venkata Rao, R., Saroj, A., 2017. A self-adaptive multi-population based jaya algorithm for engineering optimization. Swarm Evol. Comput 37, 1–26. http://dx.doi.org/10.1016/j.swevo.2017.04.008.

Wang, H., Jin, Y., Doherty, J., 2017. Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems. Ieee Trans. Cybern 47, 2664–2677. http://dx.doi.org/10.1109/tcyb.2017.2710978.

Wang, H., Jin, Y., Yang, C., Jiao, L., 2020. Transfer stacking from low-to high-fidelity: A surrogate-assisted bi-fidelity evolutionary algorithm. Appl. Soft Comput. 92, http://dx.doi.org/10.1016/j.asoc.2020.106276.

Warid, W., Hizam, H., Mariun, N., Wahab, N.I.A., 2018. A novel quasi-oppositional modified Jaya algorithm for multi-objective optimal power flow solution. Appl. Soft Comput 65, 360–373. http://dx.doi.org/10.1016/j.asoc.2018.01.039.

Wu, G., Mallipeddi, R., Suganthan, P., 2016. Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization.

Zhang, Yiying, Chi, A., Mirjalili, S., 2021b. Enhanced jaya algorithm: A simple but efficient optimization method for constrained engineering design problems. Knowledge-Based Syst. 107555. http://dx.doi.org/10.1016/j.knosys.2021.107555.

Zhang, H., Wang, Z., Chen, W., Heidari, A.A., Wang, M., Zhao, X., Liang, G., Chen, H., Zhang, X., 2021a. Ensemble mutation-driven salp swarm algorithm with restart mechanism: Framework and fundamental analysis. Expert Syst. Appl 165, 113897. http://dx.doi.org/10.1016/j.eswa.2020.113897.

Zhao, F., Hu, X., Wang, L., Zhao, J., Tang, J., Jonrinaldi, 2021a. A reinforcement learning brain storm optimization algorithm (BSO) with learning mechanism. Knowl.-Based Syst. 235, 107645. http://dx.doi.org/10.1016/j.knosys.2021.107645.

Zhao, Feng, Zeng, Z., Liu, H., Lan, R., Fan, J., 2020a. Semisupervised approach to surrogate-assisted multiobjective kernel intuitionistic fuzzy clustering algorithm for color image segmentation. Ieee Trans. Fuzzy Syst 28, 1023–1034. http://dx.doi.org/10.1109/tfuzz.2020.2973121.

Zhao, F., Zhang, L., Zhang, Y., Ma, W., Zhang, C., Song, H., 2019. An improved water wave optimisation algorithm enhanced by CMA-ES and opposition-based learning. Conn. Sci http://dx.doi.org/10.1080/09540091.2019.1674247.

Zhao, Fuqing, Zhang, L., Zhang, Y., Ma, W., Zhang, C., Song, H., 2020b. A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion. Expert Syst. with Appl 146, 113166.

Zhao, F., Zhao, J., Wang, L., Cao, J., Tang, J., 2021b. A hierarchical knowledge guided backtracking search algorithm with self-learning strategy. Eng. Appl. Artif. Intell. 102, 104268. http://dx.doi.org/10.1016/j.engappai.2021.104268.

Zhou, Z.Z., Ong, Y.S., Nguyen, M.H., Lim, D., Ieee, 2005. A study on polynomial regression and Gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm. In: 2005 Ieee Congress on Evolutionary Computation, Vols 1-3, Proceedings.