IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 1, 2019, accepted January 7, 2019, date of publication January 17, 2019, date of current version February 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2893292

HARD-DE: Hierarchical ARchive Based Mutation Strategy With Depth Information of Evolution for the Enhancement of Differential Evolution on Numerical Optimization

ZHENYU MENG, (Member, IEEE), AND JENG-SHYANG PAN, (Senior Member, IEEE)
Fujian Provincial Key Laboratory of Big Data Mining and Applications, Fujian University of Technology, Fuzhou 350000, China

Corresponding author: Jeng-Shyang Pan (jshypan@gmail.com)

This work was supported by Scientific Research Startup Foundation of Fujian University of Technology.

ABSTRACT Differential evolution is a famous and effective branch of evolutionary computation, which aims at tackling complex optimization problems. There are two aspects significantly affecting the overall performance of DE variants, one is trial vector generation strategy and the other is the control parameter adaptation scheme. Here in this paper, a new hierarchical archive-based trial vector generation strategy with depth information of evolution was proposed to get a better perception of landscapes of objective functions as well as to improve the candidate diversity of the trial vectors. Furthermore, novel adaptation schemes both for crossover rate Cr and for population size ps were also advanced in this paper, and consequently, an overall better optimization performance was obtained after these changes. The novel HARD-DE algorithm was verified under many benchmarks of the Congress on Evolutionary Computation (CEC) Competition test suites on real-parameter single-objective optimization as well as two benchmarks on real-world optimization from CEC2011 test suite, and the experiment results showed that the proposed HARD-DE algorithm was competitive with the other state-of-the-art DE variants.

INDEX TERMS Depth information, differential evolution, hierarchical archive, numerical optimization.

I. INTRODUCTION

Different Evolution (DE) is a famous and effective population based trial-and-error method for the tackling of complex single objective numerical optimization problems [37]. Generally, the single objective optimization problem can be mathematically represented as finding the optimal solution X^* of the following set:

$$\Omega^* \equiv \arg\min_{X\in\Omega} f(X) = \{X^* \in \Omega : f(X^*) \leq f(X), \forall X \in \Omega\} \tag{1}$$

where X denotes vector of parameters, Ω denotes the solution space and $f(X)$ denotes the objective function of the optimization problem. Usually, the vector of parameters X, e.g. $\Omega \subseteq \mathbb{R}^D$ for a D-dimensional optimization, is restricted by a lower $X_{\min}$ and upper $X_{\max}$ bound of constraints, $X_{\min} = (x_{\min,1}, x_{\min,2}, ..., x_{\min,D})$ and $X_{\max} = (x_{\max,1}, x_{\max,2}, ..., x_{\max,D})$, then the j^{th} parameter, $j \in \{1, 2, ..., D\}$, of X can be initialized as follows:

$$x_j = x_{\min,j} + rand_j(0, 1) \cdot (x_{\max,j} - x_{\min,j}) \tag{2}$$

Moreover, the parameters in vector X should always satisfy $x_{\min,j} \leq x_j \leq x_{\max,j}, j \in \{1, 2, ..., D\}$, during the whole evolution. After initialization of all vectors in the population (the i^{th} vector of the population in the G^{th} generation is denoted by $X_{i,G}$), DE employs mutation operation to generate donor vector, i.e., $V_{i,G}$, and then employs crossover operation to generate trial vector, i.e., $U_{i,G}$. Fig. 1 presents the relationship between the target vector $X_{i,G}$, the donor vector $V_{i,G}$ and potential trial vectors, i.e., $U_{i,G}$, $U'_{i,G}$ or $U''_{i,G}$, of the canonical DE algorithm in a 2D view [8], [9], [21], [31]. Finally, selection is conducted between the target vector $X_{i,G}$ and the trial vector $U_{i,G}$.

There are two aspects significantly affecting the overall performance of DE algorithm, one is the trial vector generation strategy, and the other is control parameter adaptation scheme. For the trial vector generation strategy, there are also two components within it, one component is mutation strategy and the other component is crossover scheme. As many mutation strategies and crossover schemes had been proposed in the literature since the inception of

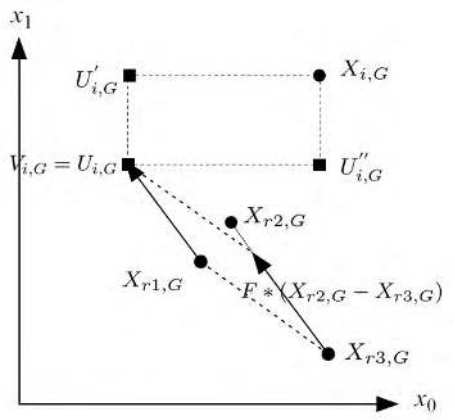

FIGURE 1. Relationship between target vector $X_{i,G}$, donor vector $V_{i,G}$ and trial vector candidates $U_{i,G}$, $U'_{i,G}$ or $U''_{i,G}$ of the canonical DE algorithm in a 2D view.

DE algorithm [32], [33], [36], [37], a notation "DE/$x/y/z$" was introduced to classify these different trial vector generation strategies where x specifies the target vector, y is the number of different vector pair, and z denotes the crossover scheme. Then, the trial vector generation strategy of canonical DE algorithm employing the DE/rand/1/z mutation strategy and a binomial crossover scheme can be written in the form: DE/rand/1/bin. Besides the canonical DE algorithm, Price *et al.* [31], [34], [37] also introduced other trial vector generation strategies, such as DE/best/1/bin, DE/best/2/bin, DE/target-to-best/1/bin, DE/rand/1/either-or, etc. Feoktistov and Janaqi [12] conducted a generalization of these mutation strategies, and they classified these mutation strategies into four groups, however, this classification was not welcomed by DE researchers in comparison with the DE/$x/y/z$ convention. Mezura-Montes *et al.* [25] conducted a deeper comparison among these DE mutation strategies, and experiment results revealed that DE/best/1/bin had an overall better exploitation capacity and performed better on unimodal separable objective functions while mutation strategy DE/rand/1/bin had an overall better exploration capacity and performed better on multi-modal nonseparable objective functions. Later research [8], [35], [43] showed that the mutation strategy DE/target-to-best/1/bin can be considered a balance between exploitation and exploration of the above two mutation strategies, nevertheless, the global best guided mutation strategies, e.g. DE/best/1/bin, DE/target-to-best/1/bin, etc., usually lead to premature convergence. Instead of employing the single global best solution, Zhang and Sanderson [43] proposed a new mutation strategy DE/target-to-$p$best/1/bin employing a handful top superior individuals for evolution while incorporating an optional external archive for solution diversity enhancement. This new mutation strategy achieved a big success, and the recently proposed state-of-the-art DE variants either employed the same mutation strategy [6], [7], [39] or employed some similar variants of this mutation strategy [14], [21], [23].

Besides mutation strategy, crossover scheme also played important role in the trial vector generation strategy. There are mainly two crossover schemes in DE algorithm, one is exponential crossover, and the other is binomial crossover. The exponential crossover is actually a combination of 1-point crossover and 2-point crossover proposed in Genetic Algorithm (GA) [15], and it has a representative/positional bias because of the dependence on parameter separation. The binomial crossover tackles the representative bias by treating each parameter independently, however, bias [19], [21], [24] still exists from a high dimensional perspective of view, and this was discussed in the following paragraph mentioning control parameter Cr of DE algorithm. There were also other crossover-like operations, e.g. disabled crossover (trial vector generation without crossover) [31], blending crossover [14], and arithmetic recombination [8], [9] techniques mentioned in the literature, nevertheless, the binomial crossover was the most popular one on real-parameter single objective optimization.

Control parameters adaptation schemes also play very important role in the overall optimization performance of DE algorithm, and there are three control parameters, population size ps, scale factor F, and crossover rate Cr, restricting selection operation, mutation operation, and crossover operation respectively. ps denotes the population size which defines the number of selection operations in each generation, F denotes the scale factor which constricts the difference vector in mutation operation, and Cr denotes the crossover rate which determines how many parameters in the target vector are changed during the crossover operation. Earlier research on DE algorithm employs fixed control parameters during the whole evolution. The recommended values of the three control parameters by DE inventors Price *et al.* [31] are given as follows: ps is recommended to be set an integer multiple number of dimension D, $ps \in [5D, 10D]$, and $ps = 100$ is a good default setting; F is recommended to be set a real number in the interval [0.4, 1], and $F = 0.5$ is a good default value; There are different recommendations of Cr, and all these recommendations are within the interval [0, 1]. $Cr = 0.1$ is a good initial value for unimodal separable functions while $Cr = 0.9$ is a good initial value for multi-modal and nonseparable functions. In addition, there are also some other recommendations for the control parameter settings mentioned in the literature [1], [13], [25], [30], [31], [41]. These claims and counter claims concerning the rules of the fixed control parameter settings may confuse DE researchers. Furthermore, the binomial crossover scheme with a fixed Cr also has a representative bias from higher dimensional perspective of view. As we know, the performance dependence on parameter separation of exponential crossover is actually due to the unequal selection probability of potential candidates, and this unequal selection probability also exists in binomial crossover with a fixed crossover rate from a higher dimensional view. Meng *et al.* [24] and Pan *et al.* [27] proposed a new Cr parameter reduced DE variant employing an auto-generated crossover matrix to replace the crossover

rate Cr and tackled the high dimensional representative bias, nevertheless, these all lack a good adaptation to the objective function during the whole evolution. Therefore, DE variants with adaptive control parameters became much more popular both for scientific researchers and for engineers [4], [8], [21], [31], [35].

Generally speaking, all the adaptation schemes of control parameters mentioned in the literature can be classified into three categories: the first category is ''Deterministic parameter control'', and the control parameters in this category are renewed according to a certain deterministic rule, e.g. the time based adaptation of mutation rate r in [16], the number of function evaluations based inertia weight iw in [17] and the number of function evaluations based top superior rate p in [6] and [7] etc., can be classified into this category; the second category is named of ''Adaptive parameter control'', which means that the adaptation mechanisms in this group employ some feedbacks of the evolution in the parameter adaptation, the adaptation scheme for Cr in [35] and the adaptation schemes for both F and Cr in [6], [7], [38], [39], and [43] etc., can be classified into this category; the last category is called ''Self-adaptive parameter control'', which means evolution of the evolution. Control parameters in this group are encoded into each individual and they also undergone evolution operations such as mutation, crossover and selection. The self-adaptive mechanisms of F and Cr in [3]–[5] and [42] and the self-adaptive mechanism of population size ps in [40] etc., can be classified into this category.

By reviewing some recently proposed state-of-the-art DE variants, we can see that each powerful DE variant was a combination of both a specified mutation strategy and the associated adaptation schemes for control parameters. For example, Zhang and Sanderson introduced a novel mutation strategy with optional external archive in JADE algorithm [43], meanwhile, new control parameter adaptation schemes both for scale factor F and for crossover rate Cr were also introduced in the JADE algorithm, both of which helped JADE win the competition on scale-invariant optimization at WCCI2008. The LSHADE algorithm [39] inherited the external archive based mutation strategy proposed in JADE algorithm and proposed a success history based control parameter pool to enhance the diversity of control parameters, furthermore, a dynamic linear population size reduction scheme were also incorporated into this algorithm, all of which helped LSHADE algorithm win the competition on real-parameter single objection optimization at CEC2014. Brest et al. proposed a new weighted version of mutation strategy as well as well-tuned control parameter adaptation schemes in jSO algorithm [7], and this new algorithm secured the first rank at CEC2017 competitions. However, all these above mentioned DE variants still had weaknesses both in trial vector generation strategy and parameter adaptation schemes, e.g. stagnation or lack of diversity in a certain given mutation strategy, misleading interaction weakness among control parameters F and Cr [23] in some state-of-the-art DE variants. Therefore, in this paper a novel HARD-DE algorithm is proposed to enhance the overall optimization performance on the commonly used CEC benchmarks. This new variant is based on advantages of some former state-of-the-art DE variants, e.g. JADE [43], LSHADE [39] and LPALMDE [23], etc., and the main contributions of the paper are listed as follows:

1) A novel hierarchical archive based mutation strategy was proposed in the HARD-DE algorithm, and depth information of evolution was firstly taken into consideration in the mutation strategy. The proposed hierarchical archive based mutation strategy with depth information can make a better perception of the landscape of objective functions and consequently obtained an overall better performance on the tested CEC benchmarks.
2) A novel adaptation scheme for crossover rate Cr with grouping strategy was proposed in the HARD-DE algorithm, and this control parameter adaptation scheme performed very well on the tested benchmarks with the above proposed mutation strategy.
3) A novel parabolic population size reduction scheme was also proposed in the HARD-DE algorithm, furthermore, a deep discussion on two implementations of the parabolic reduction scheme was presented in the paper as well.
4) Two test suites (CEC2013 test suite and CEC2017 test suite) containing 58 benchmark functions on real-parameter single objective optimization and two real-world optimization problems from CEC2011 were employed in evaluation of the novel HARD-DE algorithm, and this choice of benchmarks may avoid overfitting problem of a certain algorithm under a single test suite. The experiment results show that the new proposed HARD-DE algorithm was also competitive with the other state-of-the-art DE variants.

The rest of the paper is organized as follows. The related works in Section II reviews several famous DE variants that are closely related to our proposed HARD-DE algorithm. Section III presents the details of the novel HARD-DE algorithm. Section IV conducts the experiment analysis of the HARD-DE algorithm in comparison with DE variants reviewed in Section II. Finally, conclusion is given in Section V.

II. RELATED WORKS

In this part, several famous and powerful DE variants including JADE [43], LSHADE [39], iLSHADE [6], jSO [7] and LPALMDE [23] are briefly reviewed. They are all closely related to the novel HARD-DE algorithm in the paper. By reviewing these former state-of-the-art DE variants, researchers can get a better understanding why and how we proposed the novel HARD-DE algorithm.

A. JADE

Zhang and Sanderson proposed a new optional external archive based mutation strategy in JADE algorithm [43],

the new mutation strategy employed a handful top superior elites of the population rather than the only global best elite in evolution. As we know, the mutation strategies DE/rand/1/bin and DE/target-to-rand/1/bin usually have good exploration capacity and converge slowly in many optimizations while some greedy strategies such as DE/best/1/bin and DE/target-to-best/1/bin have good exploitation capacity and usually converge prematurely [25]. The new proposed mutation strategy in JADE made a good balance between the two characteristics by introducing a handful top superior individuals in the mutation strategy. This mutation strategy achieved a big success and the following reviewed powerful DE variants also employed the same or similar mutation strategy. The detailed equation of the mutation strategy is presented in Eq. 3:

$$V_{i,G} = X_{i,G} + F \cdot (X^{p}_{best,G} - X_{i,G}) + F \cdot (X_{r_1,G} - \widetilde{X}_{r_2,G}) \tag{3}$$

where $X_{i,G}$ denotes the target vector, $V_{i,G}$ denotes the donor vector, $X^{p}_{best,G}$ denotes a certain vector selected from the top 100p% individuals of the population, p is the percentage of top superior individuals. Moreover, an optional external archive was also incorporated into the mutation strategy for diversity enhancement of trial vectors. Symbol A was employed in denoting the external archive that recorded the inferior solutions during the evolution, and $\widetilde{X}_{r_2,G}$ in Eq. 3 is a randomly selected vector from the union $P \cup A$ while $\widetilde{X}_{r_1,G}$ is a randomly selected vector from the current population P. Furthermore, the indices of i, r_1 and r_2 always satisfy $i \neq r_1 \neq r_2$. At the beginning of the evolution, the archive A is initialized empty, $A = \varnothing$, then after each generation of the evolution, the discarded inferior solutions are gradually added into A. When the number of inferior solutions exceeds the fixed maximum size of A, the more solutions are randomly selected out from the inferior solution set and then erased from archive A.

As it is known to all that besides the mutation strategy, a well-designed adaptation scheme for control parameters can also improve the optimization performance of DE variants. The JADE algorithm also introduced novel adaptation schemes both for scale factor F and for control parameter Cr. The adaptation schemes of control parameters F and Cr are given in Eq. 4 and Eq. 5 respectively:

$$\begin{cases} \mu_F = (1-c) \cdot \mu_F + c \cdot mean_L(S_F) \\ mean_L(S_F) = \dfrac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \end{cases} \tag{4}$$

$$\begin{cases} \mu_{Cr} = (1-c) \cdot \mu_{Cr} + c \cdot mean_A(S_{Cr}) \\ mean_A(S_{Cr}) = \dfrac{\sum_{Cr \in S_{Cr}} Cr}{|S_{Cr}|} \end{cases} \tag{5}$$

where μ_F is the location parameter of the Cauchy distribution obeyed by the scale factor F, $F \sim C(\mu_F, \sigma_F)$, $mean_L(S_F)$ denotes the Lehmer mean of the set S_F; μ_{Cr} is the mean of the Normal distribution obeyed by the crossover rate Cr, $Cr \sim N(\mu_{Cr}, \sigma_{Cr})$, $mean_A(S_{Cr})$ denotes the arithmetic mean of the set S_{Cr}. The scale parameter σ_F and the standard deviation σ_{Cr} are set the same value, $\sigma_F = \sigma_{Cr} = 0.1$. When a better solution is found by a trial vector, then the corresponding individual is labeled "success". S_F and S_{Cr} are the sets recording scale factor F and crossover rate Cr of the success individuals respectively. Moreover, $|S_{Cr}|$ denotes the size of set S_{Cr}, parameter c is used for balancing the old μ_F (or μ_{Cr}) and the corresponding mean of the success set in the update scheme. Usually, c is recommended to be set a value that satisfies $1/c \in [5, 20]$, and $c = 0.1$ is the default value in JADE algorithm.

B. LSHADE

LSHADE algorithm [39] is an enhanced version of SHADE algorithm by incorporating linear population size reduction scheme, and the SHADE algorithm is also an improved JADE algorithm by introducing both a diversity entry pool and fitness value based adaptation schemes [29] for control parameters while employing the same external archive based mutation strategy. There are H entries in the diversity entry pool, and each entry records a μ_F and μ_{Cr} pair within it. All the control parameter pairs in the pool are assigned equal values, $\mu_F = \mu_{Cr} = 0.5$, at the initialization stage. The control parameters of each individual employed in mutation operation during the evolution are randomly selected from a certain entry, and only one entry is updated in each generation. The update sequence of the H entries are in a circle, from the 1^{st} to the last and then back to the 1^{st}, and then a new circle begins until the terminal of the evolution. The H-entry pool enhances the robustness of LSHADE and voids probabilistic errors that may lead to the undesirable values of the control parameters. The detailed fitness value based adaptation schemes for control parameters μ_F and μ_{Cr} are presented in Eq. 6 and 7 respectively:

$$\begin{cases} w_k = \dfrac{\Delta f_j}{\sum_{k=1}^{|S_F|} \Delta f_j} \\ \Delta f_j = f(X_{j,G}) - f(U_{j,G}) \\ mean_{WL}(S_F) = \dfrac{\sum_{k=1}^{|S_F|} w_k \cdot S^2_{F,k}}{\sum_{k=1}^{|S_F|} w_k \cdot S_{F,k}} \\ \mu_{F,G+1} = \begin{cases} mean_{WL}(S_F), & if\ S_F \neq \varnothing \\ \mu_{F,G}, & otherwise \end{cases} \end{cases} \tag{6}$$

$$\begin{cases} w_k = \dfrac{\Delta f_j}{\sum_{k=1}^{|S_{Cr}|} \Delta f_j} \\ \Delta f_j = f(X_{j,G}) - f(U_{j,G}) \\ mean_{WL}(S_{Cr}) = \dfrac{\sum_{k=1}^{|S_{Cr}|} w_k \cdot S^2_{Cr,k}}{\sum_{k=1}^{|S_{Cr}|} w_k \cdot S_{Cr,k}} \\ \mu_{Cr,k,G+1} = \begin{cases} mean_{WL}(S_{Cr}), & if\ S_{Cr} \neq \varnothing \\ \mu_{Cr,k,G}, & otherwise \end{cases} \\ Cr_i = \begin{cases} 0, & if\ \mu_{Cr,r_i} = 0; \\ randn_i(\mu_{Cr,r_i}, 0.1) & otherwise \end{cases} \end{cases} \tag{7}$$

where $|S_F|$ denotes the size of set S_F, $|S_{Cr}|$ denotes the size of set S_{Cr}. Both S_F and S_{Cr} also denote the sets of the corresponding control parameter F and Cr of success individuals.

Δf_k denotes the fitness difference of the k^{th} individual in the success individual set, moreover, the k^{th} individual in the success individual set is also associated with a j^{th} index of the whole population in the G^{th} generation. r_i denotes a random index of the H-entry pool, and if μ_{Cr,r_i} of the r_i^{th} entry equals to zero, then the crossover rate Cr of the i^{th} individual is always locked to 0 during the evolution, which means forcing the i^{th} individual searching toward a certain coordinate direction each generation, and this setting usually performs better on some multi-modal functions.

LSHADE algorithm also incorporated a linear population size reduction scheme for the adaptation of control parameter ps. The dynamic change of ps is presented in Eq. 8:

$$ps_G = \begin{cases} round[\frac{ps_{min} - ps_{ini}}{nfe_{\max}} \cdot nfe + ps_{ini}], & if\ G > 1 \\ ps_{ini}, & otherwise \end{cases} \tag{8}$$

where ps_{ini} denotes the initial population size, ps_{min} denotes the minimum population size, $nfe_{\max}$ denotes the maximum number of function evaluations, and nfe denotes the current number of function evaluations, $round[\cdot]$ means the "round to the nearest integer" operation. Obviously, we can see that the population size decreases from ps_{ini} to ps_{min} during the evolution, accordingly, the size of the external archive $\boldsymbol{A}$ is decreased adaptively according to population size ps, and always satisfying the following equation:

$$|\boldsymbol{A}| = r^{arc} \cdot ps_G \tag{9}$$

where $|\boldsymbol{A}|$ denotes the size of the external archive, r^{arc} denotes a constant ratio of the external archive size to population size, and ps_G denotes the population size of the G^{th} generation.

C. iLSHADE

The iLSHADE algorithm [6] is a further extension of the LSHADE algorithm, five small modifications are involved in it, and these modifications are usually beneficial to tackle multi-modal optimization problems. For the first modification, a larger μ_{Cr} value and smaller ps value, $\mu_{Cr} = 0.8, ps = 12 \cdot D$, are employed in iLSHADE algorithm at the initial stage; Second, the control parameter pair in the H^{th} entry of the entry pool of iLSHADE is set constant value, $\mu_F = \mu_{Cr} = 0.9$, during the whole evolution. Third, if a terminal value or a negative value of μ_{Cr} is selected from the entry pool, then the corresponding Cr value of the individual is set to zero. Fourth, the adaptation schemes for control parameter μ_F and μ_{Cr} are changed to new schemes shown in Eq. 10 and Eq. 11 respectively:

$$\begin{cases} w_k = \dfrac{\Delta f_j}{\sum_{k=1}^{|S_F|} \Delta f_j} \\ \Delta f_j = f(X_{j,G}) - f(U_{j,G}) \\ mean_{WL}(S_F) = \dfrac{\sum_{k=1}^{|S_F|} w_k \cdot S_{F,k}^2}{\sum_{k=1}^{|S_F|} w_k \cdot S_{F,k}} \\ \mu_{F,k,G+1} = \begin{cases} (mean_{WL}(S_F) + \mu_{F,k,G+1})/2, & if\ S_F \neq \varnothing \\ \mu_{F,k,G}, & otherwise \end{cases} \end{cases} \tag{10}$$

$$\begin{cases} w_k = \dfrac{\Delta f_j}{\sum_{k=1}^{|S_{Cr}|} \Delta f_j} \\ \Delta f_j = f(X_{j,G}) - f(U_{j,G}) \\ mean_{WL}(S_{Cr}) = \dfrac{\sum_{k=1}^{|S_{Cr}|} w_k \cdot S_{Cr,k}^2}{\sum_{k=1}^{|S_{Cr}|} w_k \cdot S_{Cr,k}} \\ \mu_{Cr,k,G+1} = \begin{cases} (mean_{WL}(S_{Cr}) + \mu_{Cr,k,G+1})/2, & if\ S_{Cr} \neq \varnothing \\ \mu_{Cr,k,G}, & otherwise \end{cases} \\ Cr_i = \begin{cases} 0, & if\ \mu_{Cr,r_i} = 0; \\ randn_i(\mu_{Cr,r_i}, 0.1) & otherwise \end{cases} \end{cases} \tag{11}$$

where the same symbols have the same meanings as the ones in LSHADE algorithm. Furthermore, the generated control parameters Cr and F of each individual of the population are also conducted a readjustment according to Eq. 12 and Eq. 13.

$$F_{i,G} = \begin{cases} \min(F_{i,G}, 0.7), & if\ nfe < 0.25 \cdot nfe_{\max} \\ \min(F_{i,G}, 0.8), & if\ nfe < 0.5 \cdot nfe_{\max} \\ \min(F_{i,G}, 0.9), & if\ nfe < 0.75 \cdot nfe_{\max} \end{cases} \tag{12}$$

$$Cr_{i,G} = \begin{cases} \max(Cr_{i,G}, 0.5), & if\ nfe < 0.25 \cdot nfe_{\max} \\ \max(Cr_{i,G}, 0.25), & if\ nfe < 0.5 \cdot nfe_{\max} \end{cases} \tag{13}$$

where nfe denotes the number of function evaluations in the current generation, $nfe_{\max}$ denotes the maximum number of function evaluations allowed during the evolution. The last modification is that instead of employing a constant p value that specifies the top 100p% superior individuals, iLSHADE algorithm employs a dynamic changed p value with the renewing scheme shown in Eq. 14:

$$p = \frac{p_{\max} - p_{\min}}{nfe_{\max}} \cdot nfe + p_{\min} \tag{14}$$

where $p_{\max}$ is the initial value of parameter p, $p_{\max} = 0.2$, and $p_{\min}$ is the terminal value of parameter p, $p_{\min} = 0.1$.

D. jSO

The jSO algorithm [7] is actually an improved iLSHADE algorithm by incorporating a new inertia weight into the mutation strategy as well as incorporating some modifications of the control parameters. The modified mutation strategy in jSO is presented in the following equation:

$$V_{i,G} = X_{i,G} + F_w \cdot F \cdot (X_{best,G}^p - X_{i,G}) + F \cdot (X_{r_1,G} - \widetilde{X}_{r_2,G}) \tag{15}$$

where same symbols have the same meaning as the ones in Eq. 3 except for the new symbol F_w. Here in Eq. 15, F_w denotes a new incorporated inertia weight and it satisfies:

$$Fw = \begin{cases} 0.7, & if\ nfe < 0.2 \cdot nfe_{\max} \\ 0.8, & if\ 0.2 \leq nfe < 0.4 \cdot nfe_{\max} \\ 1.2. & otherwise \end{cases} \tag{16}$$

Furthermore, the generated control parameters F and Cr of each individual in the population are also conducted a

readjustment according to Eq. 17 and Eq. 18 respectively during the evolution.

$$F = \begin{cases} F, & if\ F < 0.7 \& nfe < 0.6nfe_{\max} \\ 0.7 & otherwise \end{cases} \tag{17}$$

$$Cr = \begin{cases} 0.7. & if\ Cr < 0.7 \& nfe < 0.25nfe_{\max} \\ 0.6, & if\ Cr < 0.6 \& 0.25nfe_{\max} \le nfe < 0.5nfe_{\max} \\ Cr, & otherwise \end{cases} \tag{18}$$

Besides control parameter adaptation schemes and mutation strategy, there is still an extra difference lying in the initialization stage of the jSO in comparison with iLSHADE algorithm. This difference can be named of initialization difference, and the initial parameters ps, μ_F, $p_{\max}$ and $p_{\min}$ are recommended to be set values as follows: $ps = 25log(D)\sqrt{D}$, $\mu_F = 0.3$, $p_{\max} = 0.25$ and $p_{\min} = \frac{p_{\max}}{2}$.

E. LPALMDE

The LPALMDE algorithm [23] can be considered as an improved LSHADE algorithm, both the mutation strategy and the adaptation schemes for control parameters were all enhanced with advantages in LSHADE and jSO incorporated into it. For the mutation strategy in LPALMDE algorithm, a time stamp based external archive was advanced with the equation shown in Eq. 19:

$$V_{i,G} = X_{i,G} + F \cdot (X^p_{best,G} - X_{i,G}) + F \cdot (X_{r_1,G} - \widehat{X}_{r_2,G}) \tag{19}$$

The same symbols in the above equation has the same meanings as the ones in Eq. 3. Moreover, symbol $\widehat{X}_{r_2,G}$ is different from $\widetilde{X}_{r_2,G}$ as $\widehat{X}_{r_2,G}$ is selected from a different union $P \cup \widehat{A}$, and $\widehat{A}$ denotes the time stamp based external archive while $\widetilde{X}_{r_2,G}$ is selected from union $P \cup A$ where A denotes the external archive without time stamp mechanism. The time based mechanism avoids too old inferior solutions being archive residents during the whole evolution, furthermore, it achieves a dynamic change from the mutation strategy with external archive to mutation strategy without archive.

For the adaptation schemes of control parameters, a Parameters with Adaptive Learning Mechanism (PALM mechanism) was introduced in LPALMDE algorithm. Both the scale factor F and crossover rate Cr were updated according to the PALM mechanism in which a new grouping strategy based adaptation was proposed. By separating the control parameters into different groups, all the three control parameters including F, Cr and ps were updated in independent manners which tackled the misleading interaction among parameters. The adaptation scheme for population size ps in LPALMDE algorithm is the same as LSHADE algorithm with the adaptation scheme shown in Eq. 8, the adaptation schemes for F and Cr were new introduced ones with the equation shown in Eq. 20 and Eq. 21 respectively.

$$\begin{cases} S = \bigcup_{j=1}^{k} S_{F_j} \\ w_n = \dfrac{\Delta f_n}{\sum_{n=1}^{|S|} \Delta f_n} \\ mean_{WL}(S) = \dfrac{\sum_{n=1}^{|S|} w_n \cdot S_n^2}{\sum_{n=1}^{|S|} w_n \cdot S_n} \\ F_j = \begin{cases} mean_{WL}(S), & if\ S \neq \varnothing \\ F_j & otherwise, \end{cases} \quad j \in [1, k]. \end{cases} \tag{20}$$

$$\begin{cases} r_j = \begin{cases} \dfrac{ns_j^2}{nSucc \cdot (ns_j + nf_j)}, & if\ ns_j > 0, \\ \epsilon, & otherwise. \end{cases} \\ P(j) = \dfrac{r_j}{\sum_{j=1}^{k}(r_j)}. \\ \mu_{Cr} = \dfrac{\sum_{j=1}^{k}(P(j) \cdot Cr_j^2)}{\sum_{j=1}^{k}(P(j) \cdot Cr_j)} \end{cases} \tag{21}$$

where S denotes the scale factor set of success individuals in all groups while S_{F_j} denotes the scale factor set of success individuals in the j^{th} group; Δf_n denotes the fitness difference of the n^{th} individual in set S; $mean_{WL}(S)$ denotes the weighted Lehmer mean of set S, F_j denotes the location parameter of a Cauchy distribution for the scale factor in the j^{th} group; ns_j and nf_j denote the number of success and failure individuals in the j^{th} group and $nSucc$ denotes the number of success individuals in the population. Moreover, a new parameter called selection probability of groups $P(\cdot)$ was also introduced in the LPALMDE algorithm, and this parameter was also employed in the adaptation of crossover rate Cr and updated adaptively during the evolution.

To summarize, the external archive based mutation strategy and the fitness difference $f(U_{i,G} - f(X_{i,G})$ based parameter adaptation schemes are empirically very useful in the enhancement of DE algorithm. Consequently, these two advantages are also inherited into our new HARD-DE algorithm.

III. THE PROPOSED HARD-DE ALGORITHM

In this section, we give a thorough description of the new proposed HARD-DE algorithm. The description of the whole algorithm is separated into three parts: in the first part, the hierarchical archive based trial vector generation strategy with depth information of evolution is given in detail; In the second part, the novel control parameter adaptation schemes with grouping strategy both for scale factor F and for crossover rate Cr are clearly illustrated; In the third part, novel parabolic population size reduction schemes including three different approaches are presented, moreover, the default HARD-DE algorithm is also defined.

A. HIERARCHICAL ARCHIVE BASED MUTATION STRATEGY WITH DEPTH INFORMATION

As is known to all that trial vector generation strategy significantly affects the overall optimization performance of DE

variants. The strategy DE/target-to-*p*best/1/bin proposed in JADE [43] made a big success and recent state-of-the-art DE variants, e.g. LSHADE [39], iLSHADE [6], jSO [7], LPALMDE [23] and QUATRE-EAR [21] etc., all employed the same or similar trial vector generation strategy as JADE algorithm. However, this trial vector generation strategy still has some weaknesses which can be found in papers [21], [23]. Here we mainly focus on a new weakness, the ignored depth information of the evolution in DE/target-to-*p*best/1/bin.

The depth information of a Kinect frame is about the distance between the sensor and the object, the depth information of a multi-layer neural network is about the data transformed during a number of layers. Here the depth information of evolution in HARD-DE algorithm is about the linkage of more than three different generation of populations, and the linkage reflecting depth information is incorporated into the mutation strategy, and it is implemented by a hierarchical external archive. The proposed hierarchical archive based mutation strategy with depth information is given in Eq. 22:

$$V_{i,G} = X_{i,G} + F \cdot (X^{p}_{best,G} - X_{i,G}) + F_1 \cdot (X_{r_1,G} - \widetilde{X}_{r_2,G}) + F_2 \cdot (X_{r_1,G} - \widetilde{X}_{r_3,G}) \quad (22)$$

where same symbols have the same meanings as the ones in Eq. 3 of JADE algorithm. The new symbol $\widetilde{X}_{r_3,G}$ here denotes a randomly selected vector from the union $P \cup B$ while $\widetilde{X}_{r_2,G}$ still denotes a randomly selected vector from the union $P \cup A$, where P denotes the current population of individuals, A denotes the inferior solutions stored in the first level storage of the hierarchical archive and B denotes the former generations of individuals stored in the second level storage of the hierarchical archive. Actually, A mainly stores inferior solutions generated in the recent passed generations while B stores solutions of the population in several former generations. The indices r_1 and r_2 in each generation are the same as JADE algorithm, and the index r_3 is randomly chosen from the union $P \cup B$ by employing random selection with restriction [28], [31] referring to r_1 and r_2. Moreover, the relationship among the three scale factors F, F_1 and F_2 satisfy the following equation:

$$\begin{cases} F_1 = 0.9 \cdot F \\ F_2 = 0.7 \cdot F \end{cases} \quad (23)$$

The hierarchical archive can be easily established without much computation expense, and it is initialized empty at the beginning of the evolution. Then in each generation during the evolution, the discarded inferior solutions from the current population are inserted into the first-level storage and the individuals in current population are inserted into the second-level storage of the hierarchical archive respectively. The fixed maximum size of the first-level storage in the hierarchical archive is the same as population size, the fixed maximum size of the second-level storage is several times bigger than population size. Here we use $ps \cdot r^{arc}$ to denote the maximum size of the second-level storage of the external archive, r^{har} is the amplification factor, and its default setting is $r^{har} = 3$. When the number of solutions exceeds either the fixed maximum of first-level storage or the second level storage in hierarchical archive, readjustment of the solutions in the corresponding storage is activated and then some of the solutions are randomly removed from the storage to keep the total number of solutions equaling to the fixed maximum size of the storage, the readjustment scheme is the same as that in JADE algorithm. The hierarchical external archive in HARD-DE algorithm is illustrated in Fig. 2.

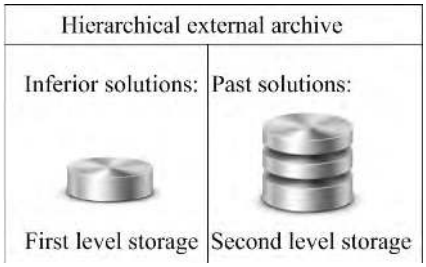

FIGURE 2. The storage in the hierarchical external archive in HARD-DE algorithm.

B. CONTROL PARAMETERS ADAPTATION SCHEMES FOR F AND Cr

In this part, the adaptation schemes of all the three control parameters F, Cr and ps in the novel HARD-DE algorithm are clearly presented. Besides incorporating above mentioned advantages in Section II, this part also introduced a novel adaptation scheme for crossover rate Cr with grouping strategy, and also introduced a novel parabolic population size reduction scheme. As is known to all that DE variants belong to probabilistic evolutionary algorithms. Due to the probabilistic nature, a better solution can be found by a good F and a bad Cr and vice versa in DE variants, e.g. JADE, LSHADE, iLSHADE and jSO etc., then the bad parameter would be mistaken for a good one which may consequently be propagated into next generation [23]. LPALMDE algorithm pointed out this weakness, named it misleading interaction weakness among control parameters, and tackled this weakness by separating these control parameters into different groups and then updated them independently during the evolution. However, the update scheme of Cr in LPALMDE algorithm was heavily dependent on the number of individuals in each group, which may fall into a bad adaptation of Cr when population size became relative small in a population size reduction scheme.

Here in the HARD-DE algorithm, there are k groups maintained during the whole evolution, the grouping strategy of the population is depicted in Fig. 3. From the figure we can see that each group is associated with two parameters, control parameter μ_{Cr} and selection probability $P(\cdot)$. The control parameter μ_{Cr} denotes the mean value of a normal distribution that the crossover rate parameter of individuals obey and the initial μ_{Cr} values for all the groups are the same, $\mu_{Cr_1} = \mu_{Cr_2} = \ldots = \mu_{Cr_j} = \ldots = \mu_{Cr_k} = 0.8$.

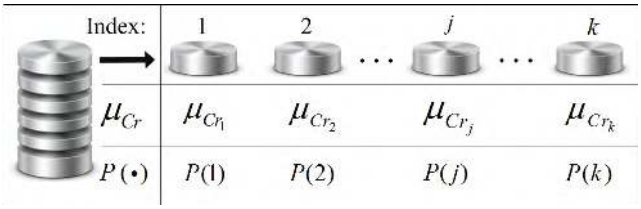

FIGURE 3. Classification of individuals in groups and group parameters illustration in HARD-DE algorithm.

Algorithm 1 Pseudo Code of Stochastic Universal Selection

Input: Population size ps and selection probability of each group $P(\cdot) = (P(1), P(2), ..., P(k))$;
Output: The group indices that the individuals are categorized into;
Initialization $space = \frac{1}{ps}$, $rnd = rand()$,
$rndn = (rnd : 1 : ps) \times space$, $sumP = 0$,
$label = zeros(ps, 1)$, $index = zeros(ps, 1)$;
for $j = 1; j \leq k; j++$ **do**
  $sumP = sumP + P(j)$;
  $nlabel$ records binary inverted $label$ value.
  $label = nlabel\&(rndn < sumP)'$;
  $index = index + label \times j$;
  $label = label|index$;
$index = index(randperm(ps))$;
return $index$;

The selection probability $P(\cdot)$ denotes the possibility that a certain individual of the population is categorized into the group, and the initial values of the k probabilities are also equal, $P(1) = P(2) = ... = P(j) = ... = P(k) = \frac{1}{k}$. Then, all individuals of the population in each generation can be separated into the k groups by employing stochastic universal selection in Algorithm 1. Accordingly, control parameter Cr of the i^{th} individual that categorized into the j^{th} group can be generated according to Eq. 24. Moreover, if the generated Cr_i value is out of the interval [0, 1], then it should be truncated to be the nearest bound, 0 or 1.

$$Cr_i = \begin{cases} N_i(\mu_{Cr_j}, 0.1). & if\ \mu_{Cr_j} > 0 \\ 0. & otherwise \end{cases} \tag{24}$$

Scale factor F of each individual in the population obeys Cauchy distribution, $F \sim C(\mu_F, \sigma_F)$, where μ_F denotes the location parameter of the Cauchy distribution that all scale factors obey, and σ_F denotes the scale parameter of the Cauchy distribution. The initial value of μ_F is set to $\mu_F = 0.3$, and σ_F is set a constant value during the whole evolution, $\sigma_F = 0.1$. Then, the i^{th} individual in the population can be generated according to $F_i \sim C(\mu_F, \sigma_F)$. Furthermore, if the generated F_i value is outside the interval (0, 1], it should be readjusted according to Eq. 25:

$$F_i = \begin{cases} C_i(\mu_F, \sigma_F), & while\ F_i \leq 0 \\ 1, & if\ F_i > 1 \\ F_i, & otherwise \end{cases} \tag{25}$$

All the parameters μ_F, μ_{Cr_j} and $P(j)$, $j \in \{1, 2, ..., k\}$ are renewed adaptively during the evolution in the proposed HARD-DE algorithm, and we present some notations first before the illustration of these adaptation schemes. If a better solution is found by the trial vector which is generated according to the above mentioned control parameters and mutation strategy, a success sign 's' is labeled on the individual, otherwise a failure sign 'f' is labeled on it. Moreover, the set of all 's' individuals is denoted by S (S only stores the indices of the 's' individuals in the population), the set of the control parameter Cr values of the 's' individuals is denoted by S_{Cr}, and the set of control parameter F values of the 's' individuals is denoted by S_F. Obviously, the three sets have the same size and elements with the same index of these sets are about the same individual, therefore, we use the same variable s to index the three sets. Then the adaptation scheme of μ_F can be renewed according to the following Eq. 26:

$$\begin{cases} w_s = \dfrac{\Delta f_{S_s}}{\sum_{s=1}^{|S_F|} \Delta f_{S_s}} \\ \Delta f_i = f(X_{i,G}) - f(U_{i,G}) \\ mean_{WL}(S_F) = \dfrac{\sum_{s=1}^{|S_F|} w_s \cdot S_F^2}{\sum_{s=1}^{|S_F|} w_s \cdot S_F} \\ \mu_{F,G+1} = \begin{cases} mean_{WL}(S_F), & if\ S_F \neq \varnothing \\ \mu_{F,G}, & otherwise \end{cases} \end{cases} \tag{26}$$

where S_s denotes the index of a certain 's' individual in the population, Δf_i denotes the fitness difference of the i^{th} individual, w_s denotes the weight of control parameter F, and $mean_{WL}(S_F)$ denotes the weighted Lehmer means. In the adaptation of control parameter μ_{Cr}, only the group with lowest selection probability is renewed in each generation. Therefore, the selection probability of each group should be renewed first before the update of control parameter μ_{Cr}. The update scheme of selection probability is presented in Eq. 27:

$$\begin{cases} r_j = \begin{cases} \dfrac{ns_j^2}{ns \cdot (ns_j + nf_j)}, & if\ ns_j > 0, \\ \epsilon, & otherwise. \end{cases} \\ ns = \sum_{j=1}^{k} ns_j, \\ P(j) = \dfrac{r_j}{\sum_{j=1}^{k}(r_j)}. \end{cases} \tag{27}$$

where ns_j and nf_j denote the number of 's' individuals and 'f' individuals in the j^{th} group respectively while ns denotes the number of 's' individuals in the population. ϵ is assigned a small value, e.g. $\epsilon = 0.01$, which is used to avoid possible null values of probability. After the selection probability of all groups are renewed, we can find the index idx of the

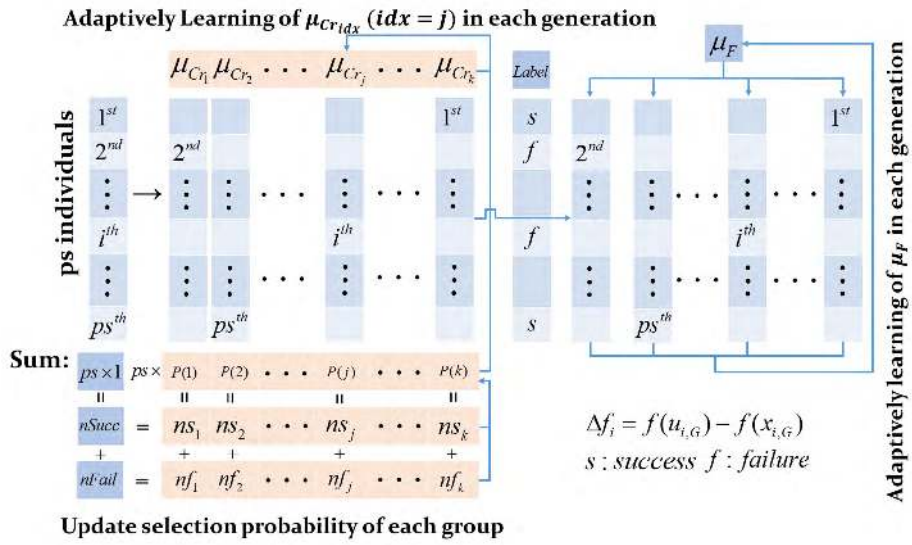

FIGURE 4. Illustration of adaptation schemes for crossover rate *Cr* and scale factor *F* in HARD-DE algorithm.

group with smallest selection probability. If there are more than one index, *idx* will be assigned a random index from these indices. Then the parameter μ_{Cr} in the *idx*-group is to renewed according to Eq. 28. Fig. 4 illustrates the novel adaptation schemes both for crossover rate *Cr* and for the scale factor *F* in the HARD-DE algorithm in detail.

$$\begin{cases} w_s = \dfrac{\Delta f_s}{\sum_{s=1}^{|S_{Cr}|} \Delta f_s} \\ \Delta f_i = f(X_{i,G}) - f(U_{i,G}) \\ mean_{WL}(S_{Cr}) = \dfrac{\sum_{s=1}^{|S_{Cr}|} w_s \cdot S_{Cr}^2(s)}{\sum_{s=1}^{|S_{Cr}|} w_s \cdot S_{Cr}(s)} \\ \mu_{Cr_{idx},G+1} = \begin{cases} mean_{WL}(S_{Cr}), & if\ S_{Cr} \neq \varnothing \,\&\, \max\{S_{Cr}\} > 0 \\ 0, & if\ S_{Cr} \neq \varnothing \,\&\, \mu_{Cr_{idx},G} = 0 \\ \mu_{Cr_{idx},G}. & otherwise \end{cases} \\ Cr_i = \begin{cases} randn_i(\mu_{Cr_{idx}}, 0.1), & if\ \mu_{Cr_{idx}} > 0; \\ 0. & otherwise \end{cases} \end{cases} \tag{28}$$

C. PARABOLIC POPULATION SIZE REDUCTION SCHEME

In this part, the novel parabolic population size reduction scheme employed in HARD-DE algorithm is introduced. Empirically, the quick reduction of population size at the beginning of the evolution usually leads to a bad perception of the landscape of most objective functions. Therefore, we proposed a parabolic population size reduction scheme in the paper to slow down the decrease of population size at the beginning of the evolution. Generally, there are two points $[ps_{ini}, ps_{ini}]$ and $[nfe_{max}, ps_{min}]$ that the parabola should pass through, therefore, if we take one of the point as the convex, then the two different parabolas can be illustrated in Fig. 5. We adopt the convex downward parabolic

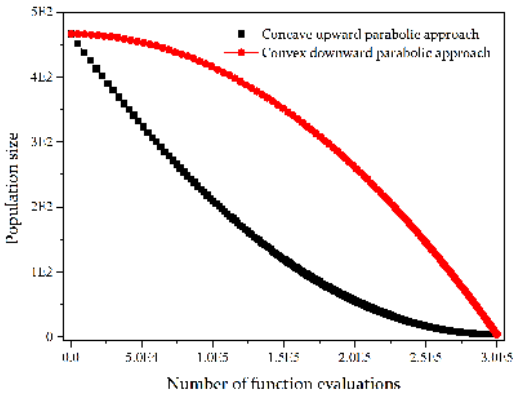

FIGURE 5. Illustration of the population size reduction schemes between the novel parabolic approach and linear approach.

population reduction scheme in the HARD-DE algorithm because population size decreases slowly at the beginning of the evolution in this approach. The detailed equation of this approach is given in Eq. 29:

$$ps_{G+1} = round[\frac{ps_{\min} - ps_{ini}}{(nfe_{\max} - ps_{ini})^2} \cdot (nfe - ps_{ini})^2 + ps_{ini}] \tag{29}$$

where $ps_{\min}$ and ps_{ini} denote the minimum and initial value of population size, *nfe* and $nfe_{\max}$ denote the current number of function evaluation and maximum number of function evaluation respectively, $round[\cdot]$ denotes rounding to the nearest value operation. As it is known to all that when fixed maximum number of function evaluation is employed in the evaluation of algorithms, the slower decrease of population size at the beginning of the evolution means fewer generations of the evolution, which may result in bad optimization performance in some multi-modal benchmark functions, and this is

Algorithm 2 Pseudo Code of HARD-DE Algorithm

Input: Bound constraints $[R^D_{min}, R^D_{max}]$, the fixed maximum number of function evaluations $nfe_{\max}$, benchmark functions $f(\boldsymbol{X})$;
Output: Best fitness value $f(X_{gbest})$, best individual X_{gbest}, number of function evaluations nfe;
Initialize the population size $ps = ps_{ini}$, all individuals $\boldsymbol{X} = \{X_1, X_2, ..., X_{ps}\}$, $k = 4$, $\boldsymbol{A} = \varnothing$, $\mu_F = 0.3$, $\mu_{Cr_1} = \mu_{Cr_2} ... = \mu_{Cr_k} = \mu_{Cr} = 0.8$, $p = 0.11$, $r^{arc} = 3$, $P(1) = P(2) = ... = P(k) = \frac{1}{k}$, $G = 1$;
for $i = 1; i \leq ps; i++$ **do**
  $X_{i,G} = X_i$, calculate fitness value $f(X_{i,G})$;
Label the global best $X_{gbest,G}$;
Calculate the corresponding fitness value $f(X_{gbest,G})$;
$nfe = ps$, $G = 2$;
while $nfe \leq nfe_{\max}$ **do**
  for $i = 1; j \leq ps; i++$ **do**
    Generate $X^p_{best,G}$, $X_{r_1,G}$, $\widetilde{X}_{r_2,G}$ and $\widetilde{X}_{r_3,G}$;
  if $G > 2$ **then**
    Adjust the individuals of the population;
    Adjust storage A and B according to Eq. 32;
  Categorize ps individuals into k-groups by stochastic universal selection in Algorithm 1;
  for $j = 1; j \leq k; j++$ **do**
    Generate F and Cr of individuals in the j^{th} group: $F \sim C(\mu_F, 0.1)$, $Cr \sim N(\mu_{Cr_j}, 0.1)$;
    Readjust F and Cr into the bound constraints if necessary;
  for $i = 1; i \leq ps; i++$ **do**
    Generate $X_{r_1,G}$, $X_{r_2,G}$ and $X_{r_3,G}$;
    Generate donor vector $V_{i,G}$ and trial vector $U_{i,G}$;
    Calculate fitness value $f(U_{i,G})$;
  $nfe = nfe + ps$;
  for $i = 1; i \leq ps; i++$ **do**
    if $f(U_{i,G}) \leq f(X_{i,G})$ **then**
      $X_{i,G+1} = U_{i,G}$;
    else
      $X_{i,G+1} = X_{i,G}$;
  if $\boldsymbol{S}_F \neq \varnothing$ **then**
    Update μ_F according to Eq. 26;
    Update $P(1), P(2), ..., P(k)$ according to Eq. 27;
    Update $\mu_{Cr_{idx}}$ according to Eq. 28;
  $G++$;
  Update storage $\boldsymbol{A}$ and $\boldsymbol{B}$;
  Label $X_{gbest,G}$ and the corresponding $f(X_{gbest,G})$;
  Adjust population size according to Eq. 29;
$f(X_{gbest}) = f(X_{gbest,G})$, $X_{gbest} = X_{gbest,G}$;
return $f(X_{gbest})$, X_{gbest}, and nfe;

a dilemma. Here in the paper, we also presented a pivot based reduction approach to balance the above mentioned dilemma. The introduced pivot pt is defined in this form: $pt = (x, y)$, where x denotes the current number of function evaluations and y denotes the current population size, then the population size reduction scheme can be changed to Eq. 30:

$$ps_{G+1} = \begin{cases} ceil[\frac{y - ps_{ini}}{(x - ps_{ini})^2} \cdot (nfe - ps_{ini})^2 + ps_{ini}], \\ \qquad\qquad if\ nfe \leq x \\ floor[\frac{y - ps_{ini}}{(x - ps_{\min})} \cdot (nfe - nfe_{\max}) + ps_{min}]. \\ \qquad\qquad otherwise \end{cases} \tag{30}$$

which can be considered as a combined parabolic-linear population size reduction scheme. When $pt = [nfe_{\max}, ps_{min}]$, the pivot based population size reduction scheme is degraded into the parabolic population size reduction scheme and when $pt = [ps_{ini}, ps_{ini}]$, the pivot based population size reduction scheme is degraded into the linear population size reduction scheme. Here we use "HARD-DE-Para" and "HARD-DE-Linear" to denote the parabolic population size reduction scheme and the linear population size reduction scheme respectively. Furthermore, the mutation strategy proposed in this paper, shown in Eq. 22, has a better perception of the landscape as a whole while mutation strategy in JADE algorithm, shown in Eq. 3, has a better exploitation characteristic and performed very well especially on some multi-modal functions. Therefore, we further extends the pivot based population size reduction scheme in Eq. 30 by employing two different mutation strategies which are listed as follows:

$$V_{i,G} = \begin{cases} According\ to\ Eq.\ 22, & if\ nfe \leq x \\ According\ to\ Eq.\ 3, & otherwise \end{cases} \tag{31}$$

"HARD-DE-Pivot" is used in denoting this approach, and it is also taken as the default trial vector generation strategy with $pt = [2/3 \cdot nfe_{\max}, 1/3 \cdot ps_{ini}]$ in HARD-DE algorithm. The initial values of F and Cr for the second mutation strategy are $F = 0.6$, $Cr = 0.8$, and they are also renewed adaptively according to Eq. 26 and Eq. 28 respectively. Moreover, the size of the first level storage and the second level storage in the hierarchical external archive is also changed adaptively according to the decrease of population size. The size of the first level storage and the size of the second level storage satisfies the following equation during the evolution:

$$\begin{cases} |\boldsymbol{A}| = ps_G \\ |\boldsymbol{B}| = r^{arc} \cdot ps_G \end{cases} \tag{32}$$

where $|\boldsymbol{A}|$ and $|\boldsymbol{B}|$ denote the size of the first and second storage of the hierarchical external archive, r^{arc} denotes the ratio of external archive size to population size, and ps_G denotes the current population size. The pseudo code of the HARD-DE is presented in Algorithm 2.

IV. EXPERIMENT ANALYSIS OF HARD-DE ALGORITHM

A. EXPERIMENT ENVIRONMENT DESCRIPTION

In this section 58 benchmarks from the CEC2013, CEC2017 test suites and 2 benchmarks from CEC2011 test

TABLE 1. Minimum, Median, Mean/Std of fitness error Δf^* over 51 runs under each benchmark on 30-D optimization are presented here for comparisons. ">", "=" and "<" denote better performance, similar performance and worse performance in comparison with the default HARD-DE algorithm (HARD-DE-Pivot).

30D	HARD-DE-Linear			HARD-DE-Para			HARD-DE-Pivot		
No.	best	median	mean/std	best	median	mean/std	best	median	mean/std
1	0(=)	0(=)	0/0(=)	0(=)	0(=)	0/0(=)	0	0	0/0
2	0(>)	2.2737E-013(>)	2.6304E-013/1.7830E-013(>)	0(>)	2.2737E-013(>)	1.6942E-013/1.0008E-013(>)	2.2737E-013	1.3642E-012	2.3362E-012/2.3555E-012
3	0(>)	2.2737E-013(>)	9.2648E-007/6.5380E-006(<)	0(>)	2.2737E-013(>)	7.4454E-013/3.7273E-012(>)	2.2737E-013	6.8212E-013	2.2960E-012/3.8844E-012
4	0(=)	2.2737E-013(=)	1.5158E-013/1.0825E-013(>)	0(=)	0(>)	9.8083E-014/1.1373E-013(>)	0	2.2737E-013	2.9425E-013/1.5283E-013
5	1.1369E-013(=)	1.1369E-013(=)	1.1369E-013/0(=)	1.1369E-013(=)	1.1369E-013(=)	1.1369E-013/0 (=)	1.1369E-013	1.1369E-013	1.1369E-013/0
6	0(>)	2.2737E-013(>)	2.2532E-011/1.0630E-010(>)	0(>)	5.1159E-012(<)	9.8159E-008/4.5454E-007(<)	1.1369E-013	9.0949E-013	1.1049E-009/4.2715E-009
7	6.6667E-005(>)	3.3944E-002(<)	7.6372E-002/1.2851E-001(<)	1.9065E-005(>)	3.1799E-003(<)	3.9057E-002/7.1947E-002(>)	1.3629E-004	2.2691E-003	4.3935E-002/1.0781E-001
8	2.0780E+001(<)	2.0935E+001(<)	2.0932E+001/5.3953E-002(<)	2.0810E+001(<)	2.0950E+001(<)	2.0949E+001/4.2174E-002(<)	2.0528E+001	2.0864E+001	2.0834E+001/1.2659E-001
9	2.1830E+001(<)	2.6008E+001(<)	2.5612E+001/1.4271E+000(<)	2.3186E+001(<)	2.6313E+001(<)	2.6089E+001/1.4887E+000(<)	1.5043E+001	2.5591E+001	2.5028E+001/2.5150E+000
10	0(=)	0(=)	0/0(=)	0(=)	0(=)	0/0(=)	0	0	0/0
11	1.1369E-013(<)	1.7053E-013(=)	1.8168E-013/3.2195E-014(<)	1.4330E-010(<)	2.7123E-009(<)	4.3016E-009/4.0589E-009(<)	5.6843E-014	1.7053E-013	1.6161E-013/3.9989E-014
12	6.4390E+000(>)	1.1254E+001(<)	1.1256E+001/1.8784E+000(<)	1.0739E+001(<)	1.5167E+001(<)	1.5117E+001/2.2503E+000(<)	7.9985E+000	1.1024E+001	1.1127E+001/1.6733E+000
13	1.0161E+001(<)	2.1642E+001(<)	2.1944E+001/7.1526E+000(<)	1.0889E+001(<)	2.1251E+001(<)	2.2414E+001/6.2374E+000(<)	5.0389E+000	1.8602E+001	1.9049E+001/5.9231E+000
14	3.2766E-001(<)	4.4879E+000(<)	4.5588E+000/2.4997E+000(<)	2.2359E+001(<)	3.9016E+001(<)	3.9965E+001/1.0101E+001(<)	1.8190E-012	3.6380E-012	8.5726E-003/1.3283E-002
15	1.8618E+003(>)	2.8955E+003(>)	2.8439E+003/3.2559E+002(>)	2.5446E+003(<)	3.3373E+003(<)	3.2998E+003/3.2555E+002(<)	2.1160E+003	2.9393E+003	2.8906E+003/2.8315E+002
16	6.1609E-001(<)	1.0250E+000(<)	1.3321E+000/6.8024E-001(<)	6.9645E-001(<)	1.2502E+000(<)	1.4825E+000/5.3749E-001(<)	1.4066E-001	7.1281E-001	7.3832E-001/4.4400E-001
17	3.0489E+001(=)	3.0587E+001(<)	3.0602E+001/6.5886E-002(<)	3.1313E+001(<)	3.1964E+001(<)	3.1952E+001/2.9898E-001(<)	3.0434E+001	3.0434E+001	3.0434E+001/4.4578E-014
18	5.5944E+001(<)	6.2779E+001(<)	6.3376E+001/4.1931E+000(<)	6.2652E+001(<)	7.5692E+001(<)	7.6239E+001/6.4455E+000(<)	4.9390E+001	6.1283E+001	6.1596E+001/4.8775E+000
19	1.0374E+000(<)	1.1863E+000(<)	1.1786E+000/7.2075E-002(<)	1.2114E+000(<)	1.5241E+000(<)	1.5317E+000/1.2920E-001(<)	9.5289E-001	1.1770E+000	1.1775E+000/9.3495E-002
20	8.7821E+000(<)	9.7981E+000(<)	1.0081E+001/1.1632E+000(<)	8.8021E+000(<)	1.0017E+001(<)	1.0402E+001/1.3477E+000(<)	8.6393E+000	9.6668E+000	9.7054E+000/4.4241E-001
21	2.0000E+002(=)	3.0000E+002(=)	2.9105E+002/3.9120E+001(>)	2.0000E+002(=)	3.0000E+002(=)	3.0367E+002/3.1788E+001(<)	2.0000E+002	3.0000E+002	2.9949E+002/5.3368E+001
22	1.0745E+002(<)	1.1318E+002(<)	1.1314E+002/2.6504E+000(<)	1.2557E+002(<)	1.4247E+002(<)	1.4286E+002/7.8433E+000(<)	1.0484E+002	1.0598E+002	1.0601E+002/6.2829E-001
23	2.3202E+003(<)	3.0327E+003(<)	3.0037E+003/3.3141E+002(<)	2.4965E+003(<)	3.2711E+003(<)	3.3271E+003/3.2300E+002(<)	2.3118E+003	3.0168E+003	2.9878E+003/2.9060E+002
24	2.0000E+002(=)	2.0000E+002(>)	2.0000E+002/6.6728E-003(>)	2.0000E+002(=)	2.0000E+002(>)	2.0000E+002/1.4722E-003(>)	2.0000E+002	2.0001E+002	2.0001E+002/7.2523E-003
25	2.0000E+002(=)	2.0000E+002(>)	2.0631E+002/1.6059E+001(>)	2.0000E+002(=)	2.0000E+002(>)	2.0630E+002/1.4874E+001(>)	2.0000E+002	2.0001E+002	2.0677E+002/1.5845E+001
26	2.0000E+002(=)	2.0000E+002(=)	2.0000E+002/1.4352E-013(=)	2.0000E+002(=)	2.0000E+002(=)	2.0000E+002/1.4352E-013(=)	2.0000E+002	2.0000E+002	2.0000E+002/1.4352E-013
27	3.0000E+002(=)	3.0005E+002(>)	3.0015E+002/4.4662E-001(=)	3.0000E+002(>)	3.0002E+002(>)	3.0004E+002/6.2398E-002(>)	3.0002E+002	3.0010E+002	3.0015E+002/1.4219E-001
28	3.0000E+002(=)	3.0000E+002(=)	3.0000E+002/1.5814E-013(=)	3.0000E+002(=)	3.0000E+002(=)	3.0000E+002/2.3066E-013(=)	3.0000E+002	3.0000E+002	3.0000E+002/2.7047E-013
> / = / <	6/11/11	7/8/13	7/6/15	5/9/14	6/6/16	7/5/16	-/-/-	-/-/-	-/-/-

TABLE 2. Recommended Parameter settings of all these contrasted algorithms.

Algorithms.	Parameters initial settings
JADE	$\mu_F = 0.5, F \sim C(\mu_F, 0.1), \mu_{Cr} = 0.5, Cr \sim N(\mu_{Cr}, 0.1), ps = 100, p = 0.05, c = 0.1$
LSHADE	$H = 6, \mu_F = 0.5, F \sim C(\mu_F, 0.1), \mu_{Cr} = 0.5, Cr \sim N(\mu_{Cr}, 0.1), ps = 18D \sim 4, r^{arc} = 2.6, p = 0.11$
iLSHADE	$H, F, Cr\&r^{arc}$ same as LSHADE, $\mu_F = 0.8, \mu_{Cr} = 0.5, \mu_{F_H} = \mu_{Cr_H} = 0.9, ps = 12D \sim 4, p = 0.2 \sim 0.1$
jSO	$F, Cr\&r^{arc}$ same as iLSHADE, $\mu_F = 0.3, \mu_{Cr} = 0.8, H = 5, ps = 25log(D)\sqrt{D} \sim 4, p = 0.25 \sim 0.125$
LPALMDE	$F_j = 0.5, F_{ji} \sim C(F_j, 0.2), \mu_{Cr} = 0.5, Cr \sim N(\mu_{Cr}, 0.1), k = 8, ps = 23D \sim k, p = 0.11, r^{arc} = 1.6, T_0 = 70$
HARD-DE	$\mu_F = 0.3, \mu_{Cr} = 0.8, F\&Cr$ same as LSHADE, $H = 4, p = 0.11, ps = 25log(D)\sqrt{D} \sim 4, r^{arc} = 3$

suite are employed in verifying the proposed HARD-DE algorithm. Generally, there are several commonly used CEC test suites for real-parameter single objective optimization, e.g. CEC2005 test suite, CEC2013 test suite, CEC2014 test suite and CEC2017 test suite. Only one test suite for algorithm verification may have over-fitting problem, therefore, the paper suggests to use more than one test suites for algorithm comparison. Moreover, the reason why we adopt CEC2013 and CEC2017 test suites is that CEC2013 further enhances the eight year ago CEC2005 test suite and CEC2017 test suite is directly improved from CEC2014 test suite rather than CEC2013 test suite. Therefore, our test suite contains the benchmarks both from CEC2013 and CEC2017, and benchmarks in CEC2013 test suite are mapped into our test suite and labeled as fa_1–fa_{28} while benchmarks in CEC2017 test suite are mapped into our test suite and labeled as fb_1–fb_{30}. Finally, we also employed two real-world benchmarks from CEC2011 test suite in the verification of the proposed HARD-DE, the first real-world problem is parameter estimation for frequency-modulated (FM) sound waves, and the second problem is spread spectrum radar polly phrase code design. The two real-world benchmarks are denoted as fc_1 and fc_2 herein the paper.

All the experiments were conducted on RedHat Linux Enterprise Edition 5.5 x64 Operating System of a PC with Intel(R) Core(TM) $i5-4590$ CPU @ 3.3Hz. The implementation of all the contrasted algorithms were under Matlab 2011b Unix version. 51 runs were conducted on each benchmark and the fitness error $\Delta f^* = f - f^*$ (f was the fitness value obtained by an algorithm under a benchmark function and f^* was the optimal value of the benchmark) was collected of the total 51 runs for experiment analysis. Values that is smaller than eps, $eps = 2.220446\text{e-}016$, was considered as zeros in the analysis.

B. ANALYSIS OF DIFFERENT POPULATION SIZE REDUCTION SCHEMES

In this part, CEC2013 test suites for real-parameter single-objective optimization benchmarks are employed in analyzing the above mentioned three different population size reduction schemes, including the HARD-DE-Para, HARD-DE-Linear, and HARD-DE-Pivot (the default HARD-DE algorithm). The ''Minimum'' (also the best), ''Median'', ''Mean and Standard Deviation'' of fitness error Δf^* over 51 runs on each benchmark are contrasted in Table 1 with the maximum number of function evaluation equaling to $10000 \cdot D$. Symbols ''$>$'', ''$=$'' and ''$<$'' in the parentheses behind the values denote ''Better Performance'', ''Similar Performance'' and ''Worse Performance'' respectively. The ''Minimum'' and ''Median'' values are measured by their arithmetic values with the rule ''the smaller the better'' while the ''Mean/Std'' values are measured under Wilcoxon's signed rank test with a level of significant $\alpha = 0.05$. From the table we can see that HARD-DE-Para algorithm performs best of the three on the unimodal benchmarks of CEC2013 test suite; the default HARD-DE algorithm (HARD-DE-Pivot algorithm) obtains ''Better Performance'' on many basic multi-modal benchmarks (most of the benchmarks from f_8 to f_{20}) of CEC2013 test suite; Both HARD-DE-Linear and HARD-DE-Para obtain better or at least similar performance in comparison with the default HARD-DE algorithm on some of the composition functions, e.g. f_{24}–f_{28}. Generally, the default HARD-DE algorithm reveals an overall better performance in comparison with HARD-DE-Linear and HARD-DE-Para from ''Minimum'', ''Median'' and ''Mean/Std'' perspective of view.

C. PARAMETER SETTINGS OF THE CONTRASTED ALGORITHMS

Here in this part, several former state-of-the-art DE variants including JADE [43], LSHADE [39], iLSHADE [6], jSO [7] and LPALMDE [23] are contrasted with the new proposed HARD-DE algorithm. All these contrasted DE variants employ the default parameter settings, and these are summarized as follows: In JADE, scale factor F and crossover rate Cr obey semi-fixed Cauchy distribution and semi-fixed Normal distribution, $F \sim C(\mu_F, \sigma_F)$ and $Cr \sim N(\mu_{Cr}, \sigma_{Cr})$ respectively. The initial μ_F and μ_{Cr} is set to $\mu_F = \mu_{Cr} = 0.5$, and they are dynamically renewed during the evolution. σ_F, σ_{Cr}, the population size ps, the ratio of top superior individuals p and the balance parameter c are all fixed constants, $\sigma_F = \sigma_{Cr} = 0.1, ps = 100, p = 0.05$, and $c = 0.1$, during the whole evolution. In LSHADE algorithm, parameters F and Cr are obeyed the same distribution and initialized the same values as JADE, moreover, a linear population size reduction is employed and the population size dynamic decreased from ps_{ini}, $ps_{ini} = 18 \cdot D$, to the fixed minimum ps_{min}, $ps_{min} = 4$. Moreover, historical success values of μ_F and μ_{Cr} are recorded in a H-entry pool, $H = 6$, a larger p defining the ratio of top superior individuals and a bigger r^{arc} defining the factor of external archive are employed in the algorithm, $p = 0.11$ and $r^{arc} = 2.6$. In iLSHADE algorithm,

the same settings of μ_F and μ_{Cr} in the first $H-1$ entries are employed except for the initial value of μ_F in comparison with LSHADE algorithm, and a larger initial μ_F is used in the first $H-1$ entries, $\mu_F = 0.8$. The initial values of μ_F and μ_{Cr} in the last entry are set fixed constant values, $\mu_{F_H} = \mu_{Cr_H} = 0.9$, which are different from LSHADE algorithm. The same population size reduction scheme is employed in iLSHADE, but with a smaller initial value, $ps_{ini} = 12 \cdot D$, in comparison with LSHADE algorithm. Furthermore, instead of employing fixed ratio p in LSHADE algorithm, the iLSHADE algorithm employed a dynamic decreased ratio p, from 0.2 to 0.1, during the evolution. For jSO, the distribution of control parameters μ_F and μ_{Cr} and the archive factor r^{arc} are also the same as LSHADE algorithm though the initial values of μ_F and μ_{Cr} are different, a bigger μ_F and a smaller μ_{Cr} are used in the initialization, $\mu_F = 0.3$, $\mu_{Cr} = 0.8$. Moreover, a smaller H,ps_{ini} and a different decreasing interval of p are employed in jSO algorithm, $H = 5$, $ps_{ini} = 25 \cdot \log D\sqrt{D}$ and $p \in [0.25, 0.125]$. For the LPALMDE algorithm, control parameters F and Cr also obey semi-fixed distributions, $F \sim C(\mu_F, 0.2)$, $Cr \sim N(\mu_{Cr}, 0.1)$ and the initial values of μ_F and μ_{Cr} are equal, $\mu_F = \mu_{Cr} = 0.5$. Group number k is set a constant value, $k = 8$, and population size is dynamically decreased from $ps_{ini} = 23 \cdot D$ to $ps_{min} = k$. The ratio of top superior individuals p is set constant value, $p = 0.11$, which is the same as LSHADE algorithm. The new parameter time stamp T_0 in LPALMDE algorithm is also set a constant value, $T_0 = 70$. The factor of the external archive r^{arc} is different from the one in LSHADE algorithm, $r^{arc} = 1.6$. For the proposed HARD algorithm, control parameters F and Cr obey the same distribution as LSHADE, however, the initial values of μ_F and μ_{Cr} are different from the ones in LSHADE, a smaller μ_F and a bigger μ_{Cr}, $\mu_F = 0.3$, $\mu_{Cr} = 0.8$, are employed in the HARD-DE algorithm. Moreover, the setting of parameter p is the same as LSHADE algorithm, $p = 0.11$, the initial population size is same as jSO, $ps = 25log(D)\sqrt{D}$, and the number of groups in HARD-DE algorithm is different from that in LPALMDE, a smaller group number, $H = 4$ is employed in HARD-DE algorithm. All these settings are listed in Table 2.

D. OPTIMIZATION PERFORMANCE EVALUATION UNDER BENCHMARKS FOR NUMERICAL OPTIMIZATION

The comparison results of the algorithms mentioned in last subsection are presented here. There are two parts in the comparison: in the first part, all these algorithms are evaluated under CEC2013 test suite; In the second part, the HARD-DE is also contrasted with the jSO algorithm which secured the first rank of the CEC2017 competition. Therefore, the first part comparison is conducted on fa_1–fa_{28} benchmarks of our test suite on 10-D and 30-D optimization respectively. Table 3 and Table 4 present the mean/std (mean value and the corresponding standard deviation) of fitness error over the 51 runs on 10D and 30D optimization respectively. Symbols "$>$", "$=$" and "$<$" in the parentheses behind the mean/std pair denote "Better Performance", "Similar Performance" and "Worse Performance" respectively, all of which are measured under Wilcoxon's signed rank test with a level of significant $\alpha = 0.05$.

TABLE 3. Mean/std fitness error $\Delta f^* = f - f^*$ comparison on 10D optimization among JADE, LSHADE, iLSHADE, jSO, LPALMDE and HARD-DE is presented here. The results are calculated under 51 independent runs with the fixed maximum number of function evaluations nfe_{max} equaling to 10000 · D. The overall performance ($>$, $=$ or $<$) of each algorithm is measured under Wilcoxon's signed rank test with the significant level $\alpha = 0.05$ in comparison with the new proposed HARD-DE. The bottom line summarized the overall performance on the 28 benchmarks.

D = 10	JADE	LSHADE	iLSHADE	jSO	LPALMDE	HARD-DE
fa_1	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0
fa_2	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0
fa_3	3.7651E+001/7.7285E+001(<)	1.9091E-001/9.7404E-001(<)	1.1193E-002/2.6209E-002(<)	1.3992E-003/9.9919E-003(>)	4.1975E-003/1.6957E-002(>)	5.5967E-003/1.9375E-002
fa_4	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0
fa_5	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0(=)	0.0
fa_6	4.4252E+000/4.9312E+000(<)	5.1948E+000/4.9465E+000(<)	6.1568E+000/4.7914E+000(<)	1.3468E+000/3.4102E+000(<)	1.1544E+000/3.1929E+000(<)	3.3804E-003/6.3293E-003
fa_7	1.0759E-001/1.6301E-001(<)	1.4181E-005/2.3149E-005(>)	1.4712E-005/4.1995E-005(>)	3.4153E-005/1.0479E-004(>)	2.0074E-005/3.7632E-005(>)	2.0208E-001/1.2112E-001
fa_8	2.0329E+001/6.4992E-002(<)	2.0230E+001/1.5088E-001(<)	2.0338E+001/9.3426E-002(<)	2.0358E+001/8.3752E-002(<)	2.0079E+001/1.4596E-001(>)	2.5237E+000/1.1282E+000
fa_9	3.7562E+000/6.2099E-001(<)	2.3294E+000/1.6767E+000(>)	5.8850E-001/8.7988E-001(>)	7.0117E-001/8.6243E-001(>)	4.0449E-001/6.3400E-001(>)	2.5237E+000/1.1282E+000
fa_{10}	1.8488E-002/8.3409E-003(<)	1.0334E-002/1.2181E-002(<)	6.0844E-003/8.2879E-003(<)	1.7401E-003/3.8597E-003(<)	1.1630E-002/1.6007E-002(<)	5.4299E-006/3.8271E-005
fa_{11}	0/0(>)	0/0(>)	1.9509E-002/1.3932E-001(<)	0/0(>)	0/0(>)	7.80203E-015/1.9755E-014
fa_{12}	4.2590E+000/1.3539E+000(<)	2.1393E+000/7.9528E-001(<)	2.0906E+000/7.7796E-001(>)	2.3801E+000/8.2236E-001(<)	3.6872E+000/1.5579E+000(<)	2.1331E+000/8.6644E-001
fa_{13}	5.4677E+000/2.2796E+000(<)	2.0873E+000/1.1803E+000(>)	1.7814E+000/8.5561E-001(>)	2.1201E+000/1.0770E+000(>)	3.1493E+000/1.9315E+000(<)	2.5298E+000/9.7643E-001
fa_{14}	1.2246E-002/2.7985E-002(<)	2.8166E-002/5.0451E-002(<)	2.8030E-001/6.4621E-001(<)	3.6075E-002/4.3136E-002(<)	1.3327E+000/2.7736E+000(<)	6.5983E-013/4.0992E-013
fa_{15}	4.8334E+002/1.2449E+002(<)	3.0419E+002/1.1924E+002(>)	2.5327E+002/1.7148E+002(>)	2.8348E+002/1.1125E+002(>)	4.7538E+002/1.7888E+002(<)	3.6552E+002/1.5347E+002
fa_{16}	1.1555E+000/2.1311E-001(<)	2.9314E-001/1.6262E-001(>)	8.3311E-001/3.0579E-001(<)	1.0942E+000/2.0374E-001(<)	1.1155E-001/1.1431E-001(>)	2.9639E-001/1.6617E-001
fa_{17}	1.0122E+001/7.9877E-015(=)	1.0122E+001/7.9877E-015(=)	1.0126E+001/6.6650E-003(<)	1.0123E+001/8.5962E-004(<)	1.0147E+001/9.5300E-002(<)	1.0122E+001/1.7940E-015
fa_{18}	1.8796E+001/1.6743E+000(<)	1.3860E+001/1.2415E+000(>)	1.3379E+001/1.2681E+000(>)	1.6434E+001/1.9716E+000(<)	1.5273E+001/2.6917E+000(<)	1.5253E+001/1.7140E+000
fa_{19}	3.3622E-001/4.2728E-002(<)	2.2556E-001/3.2446E-002(>)	3.1047E-001/5.9523E-002(<)	2.7388E-001/4.9580E-002(<)	5.3609E-001/1.5695E-001(<)	2.4165E-001/5.5838E-002
fa_{20}	2.2740E+000/4.6883E-001(<)	1.9943E+000/3.8533E-001(<)	1.8281E+000/5.2566E-001(>)	1.7248E+000/3.1470E-001(>)	1.7195E+000/4.0531E-001(>)	1.8424E+000/3.7433E-001
fa_{21}	4.0019E+002/0.0000E+000(<)	4.0019E+002/0.0000E+000(<)	4.0019E+002/0.0000E+000(<)	3.9627E+002/2.8033E+001(<)	4.0019E+002/0.0000E+000(<)	3.8842E+002/4.7573E+001
fa_{22}	3.2721E+000/4.2043E+000(>)	1.1607E+001/2.3918E+001(<)	2.2267E+001/3.2619E+001(<)	6.5426E+000/4.8560E+000(<)	2.3400E+001/2.3195E+001(>)	3.4106E+000/3.8802E+000
fa_{23}	5.2172E+002/1.7317E+002(<)	2.8705E+002/1.5536E+002(>)	2.2308E+002/1.1810E+002(>)	2.2153E+002/1.1252E+002(>)	4.1630E+002/2.0770E+002(<)	3.3962E+002/1.7191E+002
fa_{24}	1.9932E+002/9.7873E+000(>)	2.0074E+002/1.4231E+001(<)	2.0109E+002/1.1946E+001(<)	1.9992E+002/1.1073E+001(>)	1.9812E+002/1.3405E+001(>)	2.0000E+002/9.4284E-004
fa_{25}	2.0033E+002/5.1683E+000(<)	1.9963E+002/1.3963E+001(<)	2.0054E+002/1.7517E+000(<)	2.0009E+002/6.3574E-001(<)	1.9813E+002/1.3359E+001(=)	1.9813E+002/1.3267E+001
fa_{26}	1.4147E+002/4.5620E+001(<)	1.6054E+002/4.7715E+001(<)	1.2384E+002/4.0550E+001(<)	1.0258E+002/1.8040E+000(>)	1.1345E+002/2.8903E+001(<)	1.0278E+002/1.1268E+000
fa_{27}	3.0230E+002/1.5159E+001(<)	3.0000E+002/0.0000E+000(=)	3.1415E+002/4.9002E+001(<)	3.0000E+002/0.0000E+000(=)	3.0000E+002/0.0000E+000(=)	3.0000E+002/0.0000E+000
fa_{28}	2.9608E+002/2.8006E+001(>)	2.9216E+002/3.9208E+001(>)	3.0000E+002/0.0000E+000(=)	3.0000E+002/0.0000E+000(=)	2.9608E+002/2.8006E+001(>)	3.0000E+002/0
w/d/l	4/5/19	10/6/12	8/5/15	10/6/12	10/6/12	-

TABLE 4. Mean/std fitness error $\Delta f = f - f^*$ comparison on 30D optimization among JADE, LSHADE, iLSHADE, jSO, LPALMDE and HARD-DE is presented here. The results are calculated under 51 independent runs with the fixed maximum number of function evaluations $nfe_{\max}$ equaling to 10000D. The overall performance of each algorithm is measured under Wilcoxon's signed rank test with the significant level $\alpha = 0.05$ in comparison with the new proposed HARD-DE.

D = 30	JADE	LSHADE	iLSHADE	jSO	LPALMDE	HARD-DE
fa_1	0/0(=)	0/0(=)	0/0(=)	0/0(=)	0/0(=)	0/0
fa_2	7.0462E+003/5.8745E+003(<)	2.8399E-012/1.2034E-011(<)	7.5088E-007/5.0803E-006(<)	3.7115E-010/7.7174E-010(<)	3.5191E-011/1.8040E-009(<)	2.3362E-012/2.3555E-012
fa_3	3.1464E+005/1.0965E+006(<)	1.7272E-002/8.8023E-002(<)	4.9453E-003/3.2508E-002(<)	2.3793E-009/1.5886E-008(<)	7.3958E-003/3.6215E-002(<)	2.2960E-012/3.8844E-012
fa_4	3.5640E+003/1.0960E+004(<)	1.6496E-013/1.0248E-013(>)	3.9679E-013/2.5255E-013(<)	2.5457E-012/1.7191E-012(<)	3.2991E-013/1.1481E-013(<)	2.9425E-013/1.5283E-013
fa_5	1.1146E-013/1.5919E-014(=)	1.0923E-013/2.2287E-014(=)	1.1369E-013/0.0000E+000(=)	1.1369E-013/0.0000E+000(=)	1.1369E-013/0.0000E+000(=)	1.1369E-013/0
fa_6	2.0712E+000/7.1703E+000(<)	5.1779E-001/3.6978E+000(<)	1.0356E+000/5.1769E+000(<)	1.0356E+000/5.1769E+000(<)	3.6853E-004/1.9134E-003(<)	1.1049E-009/4.2715E-009
fa_7	2.6358E+000/2.3447E+000(<)	6.7711E-001/3.8630E-001(<)	2.8039E-001/2.2748E-001(<)	4.9068E-002/1.5007E-001(<)	1.0292E-001/1.2628E-001(<)	4.3935E-002/1.0781E-001
fa_8	2.0942E+001/5.3025E-002(<)	2.0890E+001/1.1297E-001(<)	2.0833E+001/1.0683E-001(=)	2.0957E+001/4.8471E-002(<)	2.0717E+001/1.2151E-001(>)	2.0833E+001/1.2659E-001
fa_9	2.6891E+001/1.8036E+000(<)	2.7053E+001/1.1305E+000(<)	2.0158E+001/4.9504E+000(>)	2.3704E+001/2.6676E+000(>)	1.5636E+001/3.7837E+000(>)	2.5028E+001/2.5150E+000
fa_{10}	4.2114E-002/2.4790E-002(<)	5.3153E-004/2.2122E-003(<)	2.9004E-004/1.4499E-003(<)	0/0(=)	9.6611E-004/3.1950E-003(<)	0/0
fa_{11}	0/0(>)	7.8020E-014/3.4032E-014(>)	6.5760E-014/2.0878E-014(>)	1.6496E-013/7.3021E-014(<)	3.8060E+000/2.2312E+000(<)	1.6161E-013/3.9989E-014
fa_{12}	2.4090E+001/4.8150E+000(<)	5.4969E+000/1.6086E+000(>)	6.6331E+000/2.1494E+000(>)	9.0538E+000/2.4992E+000(>)	1.1296E+001/3.2566E+000(<)	1.1127E+001/1.6733E+000
fa_{13}	4.8920E+001/1.1661E+001(<)	5.3495E+000/2.7391E+000(>)	8.5465E+000/4.5166E+000(>)	1.0715E+001/5.2616E+000(>)	2.2353E+001/1.0266E+001(<)	1.9049E+001/5.9231E+000
fa_{14}	2.7759E-002/2.3431E-002(<)	2.9395E-002/2.2487E-002(<)	4.6537E-002/2.9907E-002(<)	8.4381E+000/4.4500E+000(<)	2.1041E+002/1.1899E+002(<)	8.5726E-003/1.3283E-002
fa_{15}	3.2918E+003/3.0778E+002(<)	2.6298E+003/2.8989E+002(>)	2.5137E+003/2.7530E+002(>)	2.6816E+003/3.3684E+002(>)	3.1328E+003/4.4921E+002(>)	2.8906E+003/2.8315E+002
fa_{16}	1.9783E+000/6.2751E-001(<)	8.2157E-001/1.4440E-001(<)	9.2240E-001/4.3183E-001(<)	2.3393E+000/3.0407E-001(<)	3.4416E-001/2.2456E-001(>)	7.3832E-001/4.4400E-001
fa_{17}	3.0434E+001/8.0389E-015(=)	3.0434E+001/1.3202E-006(=)	3.0434E+001/2.1913E-006(=)	3.0669E+001/1.0979E-001(<)	3.7692E+001/2.4068E+000(<)	3.0434E+001/4.4578E-014
fa_{18}	7.5750E+001/7.6650E+000(<)	5.1783E+001/3.1302E+000(>)	4.4237E+001/4.0750E+000(>)	5.6911E+001/6.2991E+000(>)	4.6128E+001/5.9005E+000(>)	6.1596E+001/4.8775E+000
fa_{19}	1.4356E+000/1.2019E-001(<)	1.1878E+000/8.7252E-002(<)	1.0496E+000/1.3757E-001(>)	1.2898E+000/1.0311E-001(<)	2.3760E+000/6.0265E-001(<)	1.1775E+000/9.3495E-002
fa_{20}	1.0161E+001/4.9428E-001(<)	1.1526E+001/2.1028E+000(<)	9.5020E+000/3.9815E-001(>)	9.7472E+000/3.8117E-001(<)	9.2988E+000/5.5861E-001(>)	9.7054E+000/4.4241E-001
fa_{21}	3.1161E+002/7.0509E+001(<)	2.9608E+002/1.9604E+001(>)	3.0316E+002/6.2148E+001(<)	3.0708E+002/5.8493E+001(<)	3.0281E+002/2.0100E+001(<)	2.9949E+002/5.3368E+001
fa_{22}	1.0324E+002/1.8223E+001(>)	1.0829E+002/2.5583E+000(<)	1.0676E+002/1.1981E+000(<)	1.1976E+002/3.4337E+000(<)	2.1300E+002/6.6635E+001(<)	1.0601E+002/6.2829E-001
fa_{23}	3.4268E+003/3.8445E+002(>)	2.5185E+003/2.8743E+002(>)	2.1826E+003/4.0411E+002(>)	2.3659E+003/3.3190E+002(>)	3.0503E+003/4.6513E+002(<)	2.9878E+003/2.9060E+002
fa_{24}	2.0857E+002/1.1226E+001(<)	2.0082E+002/1.7428E+000(<)	2.0006E+002/2.7178E-001(<)	2.0003E+002/3.2709E-002(<)	2.0003E+002/3.3724E-002(<)	2.0001E+002/7.2523E-003
fa_{25}	2.8170E+002/6.3689E+000(<)	2.4080E+002/4.0815E+000(<)	2.4000E+002/4.6140E+000(<)	2.3924E+002/6.8915E+000(<)	2.0806E+002/1.7642E+001(<)	2.0677E+002/1.5845E+001
fa_{26}	2.1581E+002/4.4406E+001(<)	2.0000E+002/0(=)	2.0000E+002/0(=)	2.0000E+002/0(=)	2.0000E+002/0(=)	2.0000E+002/0
fa_{27}	6.6268E+002/2.4023E+002(<)	3.0201E+002/4.2899E+000(<)	3.0104E+002/2.6188E+000(<)	3.0083E+002/9.8159E-001(<)	3.0180E+002/3.7048E+000(<)	3.0015E+002/1.4219E-001
fa_{28}	3.0000E+002/0(=)	3.0000E+002/0(=)	3.0000E+002/0(=)	3.0000E+002/0(=)	3.0000E+002/0(=)	3.0000E+002/2.7047E-013
w/d/l	2/4/22	8/5/15	9/6/13	6/5/17	5/4/19	-

TABLE 5. Mean/std fitness error $\Delta f = f - f^*$ comparison on 50D optimization among JADE, LSHADE, iLSHADE, jSO, LPALMDE and HARD-DE is presented here. The results are calculated under 51 independent runs with the fixed maximum number of function evaluations $nfe_{\max}$ equaling to 10000D. The overall performance of each algorithm is measured under Wilcoxon's signed rank test with the significant level $\alpha = 0.05$ in comparison with the new proposed HARD-DE.

D = 50	JADE	LSHADE	iLSHADE	jSO	LPALMDE	HARD-DE
f_1	1.6942E-013/1.0008E-013(<)	2.2292E-014/6.8286E-014(>)	4.4583E-014/9.1172E-014(>)	2.6750E-014/7.3986E-014(>)	6.2416E-014/1.0248E-013(>)	1.1146E-013/1.1480E-013
f_2	2.7583E+004/1.5474E+004(<)	1.4036E+003/1.6680E+003(<)	3.4366E+003/3.9672E+003(<)	2.2197E+002/1.6572E+002(<)	6.5623E+002/1.0106E+003(<)	1.9177E+002/6.0687E+002
f_3	5.7894E+006/1.3374E+007(<)	6.6993E+004/1.7655E+005(<)	4.9184E+003/1.7221E+004(<)	1.3171E+002/8.0333E+002(>)	7.6501E+004/4.4019E+005(<)	1.8254E+003/1.0164E+004
f_4	6.2245E+003/1.7270E+004(<)	3.8092E-011/8.0764E-011(>)	1.6106E+009/3.6941E+009(<)	4.7411E+009/7.7628E+009(<)	4.1006E-010/1.3387E-009(>)	1.3238E+009/3.1395E+009
f_5	1.4267E-013/5.0038E-014(<)	1.8279E-013/5.6058E-014(<)	1.7164E-013/6.1737E-014(<)	2.0954E-013/4.7545E-014(<)	1.7164E-013/6.1737E-014(<)	1.3821E-013/4.7224E-014
f_6	4.3559E+001/7.9535E-001(<)	4.3447E+001/0(=)	4.3447E+001/0(=)	4.3447E+001/0(=)	4.3447E+001/0(=)	4.3447E+001/0
f_7	1.9311E+001/8.6851E+000(<)	1.6596E+000/1.2973E+000(<)	4.5763E-001/4.7477E-001(<)	1.4960E-001/1.2100E-001(<)	1.5700E+000/1.5700E+000(<)	2.1876E-001/3.3636E-001
f_8	2.1137E+001/3.8018E-002(<)	2.1091E+001/8.5035E-002(<)	2.1062E+001/7.5472E-002(<)	2.1123E+001/4.4397E-002(<)	2.1026E+001/7.6054E-002(>)	2.1030E+001/1.1946E-001
f_9	5.4444E+001/2.0762E+000(<)	5.3319E+001/1.9029E+000(<)	4.0393E+001/8.6307E+000(>)	4.7942E+001/5.2101E+000(<)	4.3446E+001/4.8093E+000(>)	4.6825E+001/1.8323E+000
f_{10}	3.6398E-002/3.0424E-002(<)	6.0887E-003/7.2157E-003(<)	4.1078E-003/4.8959E-003(>)	4.5698E-014/2.2793E-014(>)	7.6337E-003/6.0158E-003(<)	4.4938E-003/5.8245E-003
f_{11}	0/0(>)	8.3785E-011/9.8714E-011(<)	8.4373E-013/6.2676E-013(<)	5.9902E-009/1.9493E-008(<)	3.5568E+001/7.2111E+000(<)	3.5221E-013/4.8260E-014
f_{12}	5.6905E+001/8.7542E+000(<)	1.2194E+001/2.4613E+000(>)	1.2747E+001/2.9177E+000(>)	1.5065E+001/3.4463E+000(>)	1.7738E+001/5.1633E+000(>)	2.5027E+001/3.4523E+000
f_{13}	1.2639E-002/2.6752E+001(<)	1.6538E+001/6.5736E+000(>)	1.8351E+001/8.3932E+000(>)	1.8860E+001/1.0259E+001(>)	4.2395E+001/2.0835E+001(>)	5.9910E+001/1.0352E+001
f_{14}	4.4333E-002/2.4391E-002(<)	2.2459E-001/4.8029E-002(<)	2.5666E-001/6.8396E-002(<)	5.9348E+001/1.4388E+001(<)	2.0103E+003/5.7532E+002(<)	4.0487E-002/1.8009E-002
f_{15}	6.8822E+003/3.7540E+002(<)	6.2702E+003/3.6566E+002(>)	5.3857E+003/5.6668E+002(>)	5.9500E+003/5.6668E+002(>)	6.8413E+003/8.9682E+002(<)	6.5896E+003/3.8730E+002
fa_{16}	2.2429E+000/7.3534E-001(<)	1.2660E+000/1.5780E-001(<)	1.3932E+000/6.1530E-001(<)	3.1480E+000/3.3212E-001(<)	1.1265E+000/4.3821E-001(>)	1.1268E+000/5.1483E-001
fa_{17}	5.0786E+001/5.7856E-014(=)	5.0787E+001/2.0207E-003(<)	5.0787E+001/2.1396E-003(<)	5.2434E+001/4.6550E-001(<)	1.0593E+002/1.2071E+001(<)	5.0786E+001/1.1487E-010
fa_{18}	1.4211E+002/9.3899E+000(<)	1.0623E+002/6.4447E+000(>)	8.4367E+001/8.8521E+000(>)	1.1175E+002/8.8481E+000(>)	8.3629E+001/1.2214E+001(>)	1.1937E+002/9.2856E+000
fa_{19}	2.7351E+000/2.1437E-001(<)	2.5057E+000/1.7443E-001(<)	2.3122E+000/2.0796E-001(>)	2.6998E+000/1.4630E-001(<)	5.0112E+000/8.4388E-001(<)	2.4163E+000/1.5435E-001
fa_{20}	1.9702E+001/5.9173E-001(<)	1.8545E+001/5.3900E-001(>)	1.8579E+001/4.4491E-001(>)	1.9189E+001/4.2809E-001(<)	1.8735E+001/7.6499E-001(<)	1.8629E+001/3.2887E-001
fa_{21}	8.0741E+002/3.9372E+002(>)	8.5095E+002/4.2437E+002(<)	7.9671E+002/4.4508E+002(>)	6.8389E+002/4.4920E+002(>)	7.4934E+002/4.4904E+002(>)	8.4663E+002/4.0890E+002
fa_{22}	1.5049E+001/1.4570E+001(<)	1.3775E+001/1.4722E+000(<)	1.5967E+001/1.2125E+001(<)	1.5583E+001/1.6033E-001(<)	2.0521E+003/5.4303E+002(<)	1.5583E+001/6.0933E-001
fa_{23}	7.2668E+003/7.9041E+002(<)	5.3110E+003/4.4341E+002(>)	4.7427E+003/5.5371E+002(>)	5.0592E+003/6.1531E+002(>)	5.9216E+003/9.4959E+002(>)	6.6375E+003/5.1804E+002
fa_{24}	2.5468E+002/2.6106E+001(<)	2.1139E+002/7.3310E+000(<)	2.0943E+002/1.3038E+001(<)	2.0453E+002/1.0014E+001(<)	2.0916E+002/5.8199E+000(<)	2.0039E+002/7.6989E-001
fa_{25}	3.5901E+002/1.8605E+001(<)	3.7692E+002/5.5531E+000(>)	2.7591E+002/7.7462E+000(>)	2.7417E+002/7.4624E+000(>)	2.7745E+002/5.9261E+000(>)	2.8358E+002/7.3162E+000
fa_{26}	3.3556E+002/1.0323E+002(<)	2.4876E+002/5.2270E+001(<)	2.9033E+002/3.3377E+001(<)	2.3174E+002/7.7404E+001(<)	2.3266E+002/4.8714E+001(>)	2.3860E+002/5.0620E+001
fa_{27}	1.3112E+003/3.4144E+002(<)	4.5605E+002/9.9901E+001(<)	4.9832E+002/1.4892E+002(<)	4.4705E+002/1.4816E+002(<)	3.7714E+002/4.3588E+001(<)	3.1654E+002/1.5274E+001
fa_{28}	4.5842E+002/4.1718E+002(<)	4.0000E+002/2.8433E-013(=)	4.0000E+002/2.8705E-013(=)	4.0000E+002/2.8705E-013(=)	4.0000E+002/2.8705E-013(=)	4.0000E+002/2.8159E-013
w/d/l	2/1/25	9/2/17	12/2/14	12/2/14	12/2/14	-

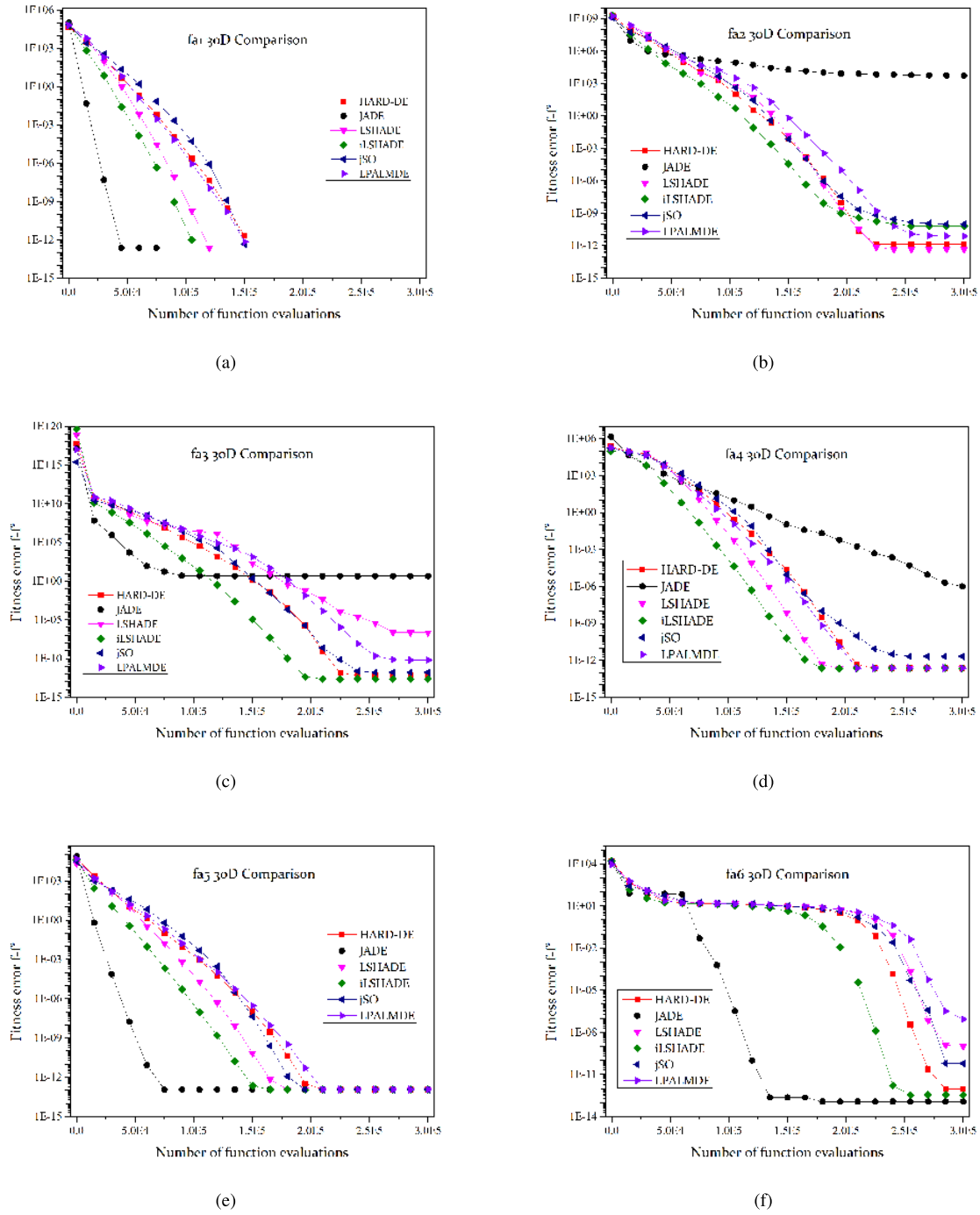

FIGURE 6. Here presents the convergence speed comparison by employing the median value of 51 runs obtained by each algorithm on 30-D optimization. There are total 28 comparison figures and the first 6 figures are presented here. (a) f_1. (b) f_2. (c) f_3. (d) f_4. (e) f_5. (f) f_6.

global optima each run. The proposed HARD-DE algorithm outperforms other DE variants on benchmark fa_6, on which other DE variants often converged into some local optima. Furthermore, the proposed HARD-DE algorithm reveals 19 better performances and 5 similar performances out of 28 benchmarks in comparison with JADE algorithm; it also reveals 12 better performances and 6 similar performances in comparison with LSHADE algorithm; reveals 13 better performances and 5 similar performances in comparison with iLSHADE algorithm; reveals 12 better performances and 6 similar performances in comparison with jSO algorithm; reveals 12 better performance and 6 similar performances in comparison with LPALMDE algorithm. In a word, the new proposed HARD-DE algorithm secures an overall better

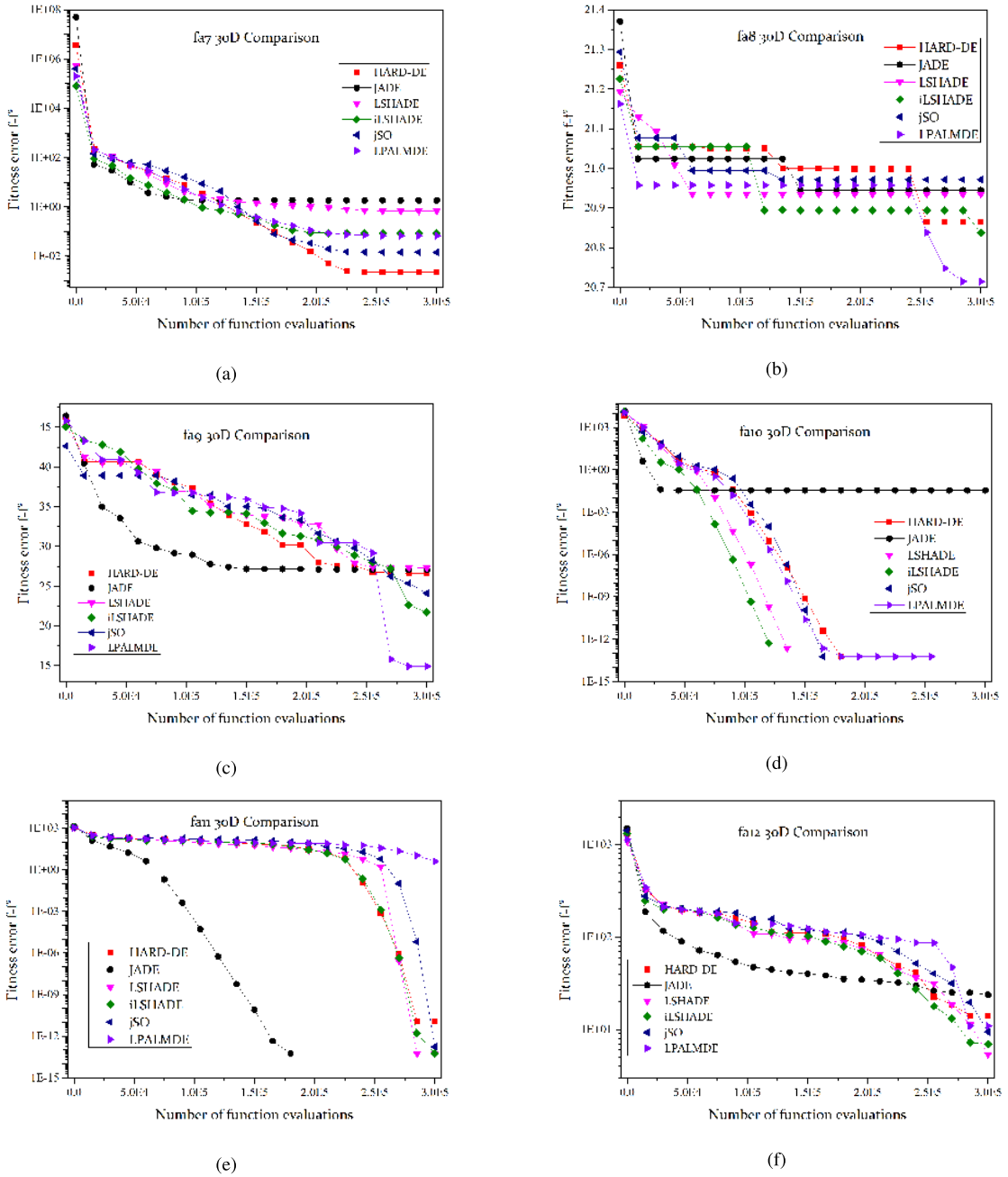

FIGURE 7. Continued from Fig. 6, comparisons on f_7–f_{12} are presented here. (a) f_7. (b) f_8. (c) f_9. (d) f_{10}. (e) f_{11}. (f) f_{12}.

performance on 10D optimization under the tested benchmark functions.

For 30D optimization with the results shown in Table 4, we can see that all the six DE variants perform equally well on f_1, f_5, f_{28}. The JADE algorithm outperforms other DE variants on f_{11}, and it can also find the global optima during each run. The jSO algorithm and the HARD-DE algorithm outperform other DE variants on benchmark f_{10}, and they can find the global optima during each run as well. Moreover, the proposed HARD-DE algorithm obtains 22 better performances and 4 similar performances out of 28 benchmarks in comparison with JADE algorithm; it also obtains 15 better performances and 5 similar performances in comparison with LSHADE algorithm; obtains 13 better performances and 6 similar performances in comparison with iLSHADE algorithm, 17 better performances and 5 similar performances in comparison with jSO algorithm, 19 better performances and 4 similar performances in comparison with LPALMDE algorithm. In a word, the proposed HARD-DE algorithm secures an overall much better performance on 30D optimization under the tested benchmark functions.

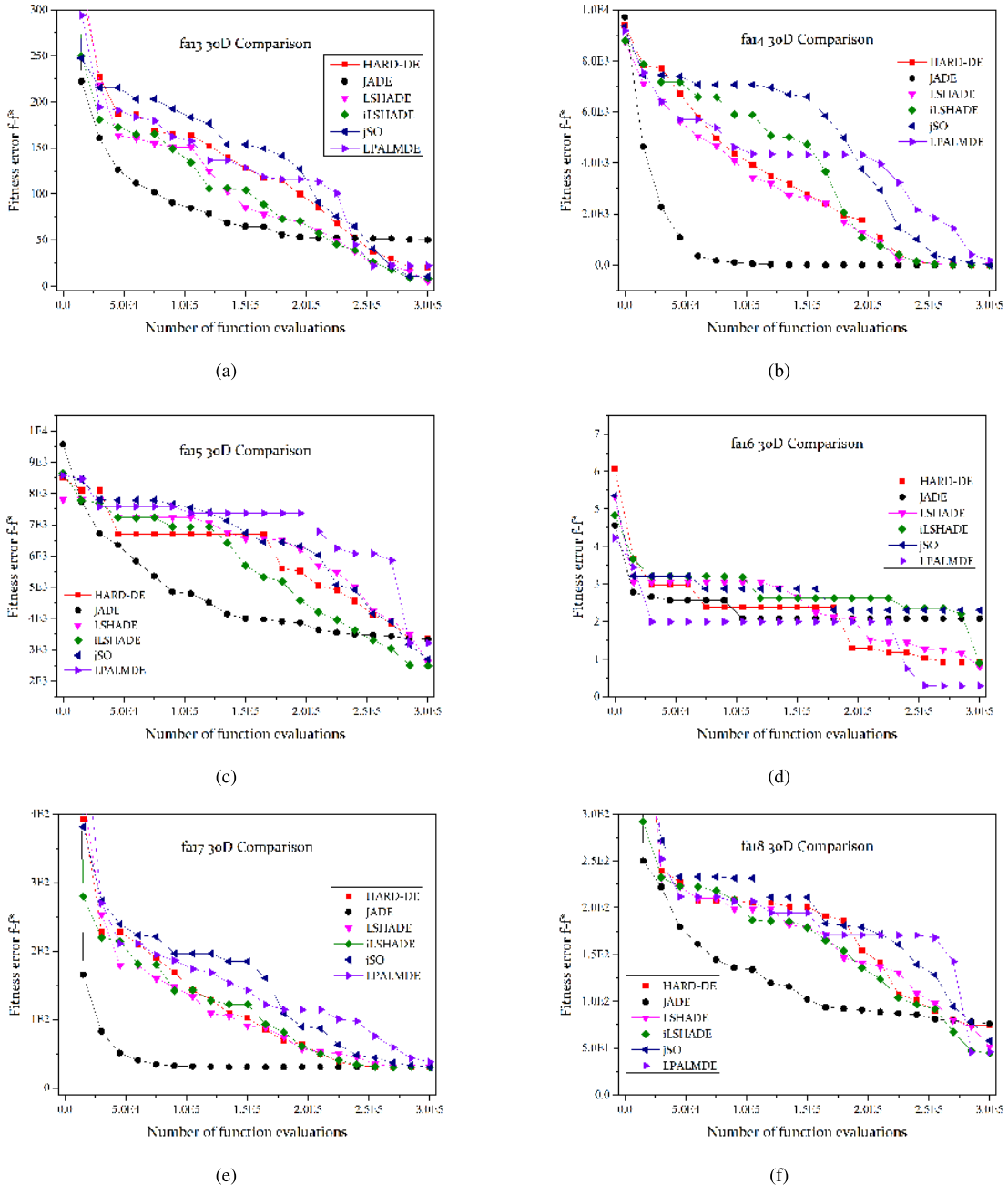

FIGURE 8. Continued from Fig. 7, comparisons on f_{13}–f_{18} are presented here. (a) f_{13}. (b) f_{14}. (c) f_{15}. (d) f_{16}. (e) f_{17}. (f) f_{18}.

For 50D optimization with the results shown in Table 5, we also can see that all the DE variants except JADE perform equally well on f_6 and f_{28}. The JADE algorithm outperforms the other DE variants on f_{11}, and it also obtains the same best performance as HARD-DE algorithm on f_{17}. The LSHADE algorithm outperforms the other DE variants on f_1, f_4, f_{12}, f_{13} and f_{20}; The iLSHADE algorithm outperforms the other DE variants on f_9, f_{15}, f_{19} and f_{23}; The jSO algorithm outperforms the other DE variants on f_3, f_7, f_{10}, f_{21}, f_{25} and f_{26}; The LPalmDE algorithm outperforms the other DE variants on f_8, f_{16} and f_{18}; The HARD-DE algorithm outperforms the other DE variants on f_2, f_5, f_6, f_{14}, f_{17}, f_{22}, $f24$ and f_{27}. Furthermore, the new proposed HARD-DE algorithm obtains 26 better or similar performance in comparison with JADE algorithm, it obtains 19 better or similar performance in comparison with LSHADE algorithm, it obtains 16 better or similar performance in comparison with iLSHADE, jSO and LPALMDE algorithm. In a word, the proposed HARD-DE algorithm secures an overall better performance on 50D optimization under the tested benchmark functions.

The proposed HARD-DE algorithm is also verified from convergence speed perspective of view. All the convergence

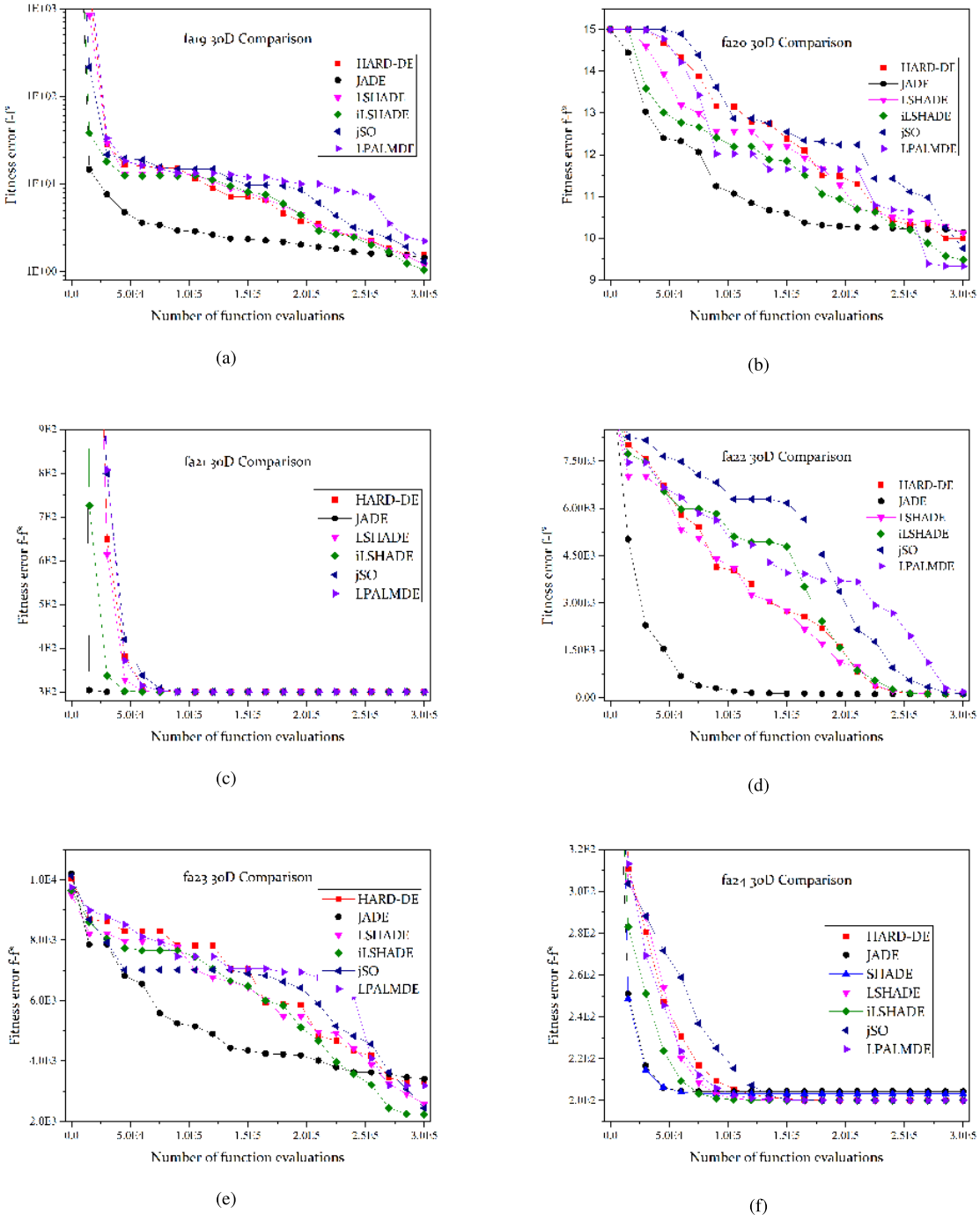

FIGURE 9. Continued from Fig. 8, comparisons on f_{19}–f_{24} are presented here. (a) f_{19}. (b) f_{20}. (c) f_{21}. (d) f_{22}. (e) f_{23}. (f) f_{24}.

speed comparisons that employ the median value of the 51 runs are also presented here in Fig. 6–Fig. 10 for algorithm evaluation on 30-D optimization. Fig. 6 presents the comparison under the first 6 benchmark functions in which the first 5 benchmarks are uni-modal functions, and we can see that JADE algorithm with fixed population size outperforms other DE variants on benchmarks f_1, f_5 and f_6, the population size reduction scheme based DE variants have similar convergence curves on all of the six benchmark functions though some of them have a little faster converge speed while others have a little slower convergence speed. What should be emphasized more is that most of these DE variants except for the new proposed HARD-DE algorithm usually converge into local optima on benchmark f_6 during the 51 runs, however, this cannot be reflected by the converge speed perspective. Fig. 7–Fig. 8 mainly presents the convergence speed comparisons on multi-modal functions f_7–f_{18}. We can see that JADE algorithm outperforms other DE variants on

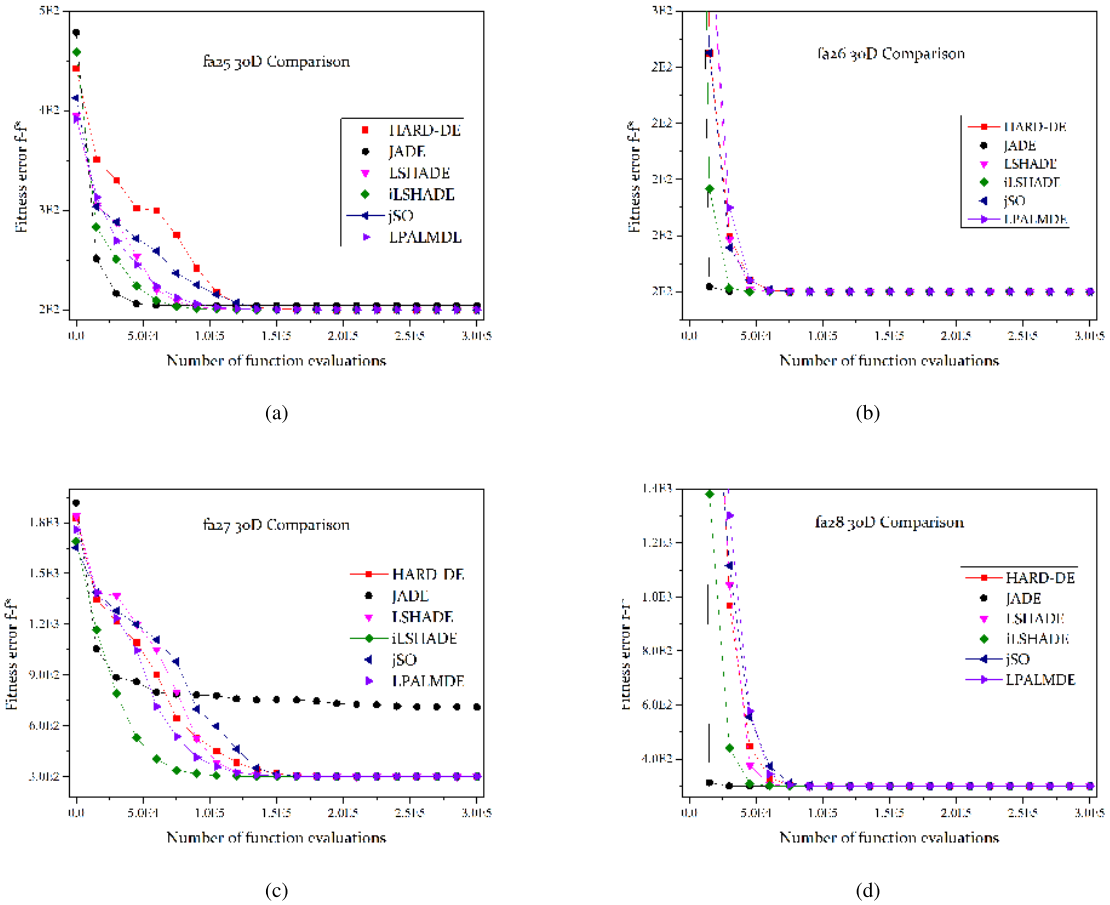

FIGURE 10. Continued from Fig. 9, the last four comparisons are presented here for analysis. (a) f_{25}. (b) f_{26}. (c) f_{27}. (d) f_{28}.

benchmark f_{11}, all the DE variants can converge into the global best on f_{10} except for JADE. The novel HARD-DE algorithm outperforms others on benchmark f_7, and it is also competitive in comparison with other DE variants on benchmarks f_8, f_9, f_{12}, f_{14}, f_{16} and f_{17}. Fig. 9 and Fig. 10 present the convergence speed comparison under the left two multi-modal functions and six composition functions. From these figures we can see that the novel HARD-DE algorithm performs extremely well on these composition functions rather than the left two multi-modal functions, and the HARD-DE algorithm is also competitive with the other DE variants on f_{22} and f_{24}–f_{28}. From all the 28 convergence figures, we can see that JADE algorithm with fixed population size usually converges quickly at the beginning of the evolution, however, other DE variants with population size reduction schemes usually obtain better performance at the end of the evolution, and our proposed HARD-DE algorithm, as a result, is competitive with other state-of-the-art DE variants from convergence speed perspective of view under these tested benchmarks.

The proposed HARD-DE algorithm is also contrasted with the state-of-the-art DE variant jSO under CEC2017 benchmark functions which are mapped into our test suite and labeled as fb_1–fb_{30} as the jSO algorithm secured the first rank on CEC2017 competition. The "Best", and "Mean/std" values are selected/calculated from 51 runs with the maximum number of function evaluation equaling to $10000 \cdot D$. All these results are presented in Table 6. Symbols ">", "=" and "<" in the parentheses of the "Best" column are measured by the arithmetic values while symbols in the "Mean/std" column are measured under Wilcoxon's signed rank test with a level of significance $\alpha = 0.05$, all these symbols has the same meanings as the ones mentioned earlier in the paper. The overall performances are summarized in last row of the table. From the table we can see that the novel HARD-DE and jSO can find the global optima during the 51-run on benchmarks fb_1–fb_3, fb_6 and fb_9. Both the new proposed HARD-DE and jSO can find tier performance on benchmarks fb_{22} and fb_{25}. From the "Best" perspective, the HARD-DE and jSO have similar performance under the 30 benchmark functions; From the "Mean/std" perspective, jSO performs a litter better than the HARD-DE algorithm. Taking all the 58 benchmark functions including fa_1–fa_{28} and fb_1–fb_{30} into consideration, the novel HARD-DE algorithm secures 30 better performances and 8 similar performances in comparison with the jSO algorithm, therefore, the new

TABLE 6. Best, mean and standard deviation comparison under benchmarks fb_1–fb_{30} between the state-of-the-art DE variant jSO and the new proposed HARD-DE algorithm. All these values are selected/calculated from 51 runs with the maximum function evaluation equaling to $nfe_{\max} = 10000 \times D$. Symbols ">", "=" and "<" in the parentheses of the "Best" column are measured by the arithmetic values while symbols in the "Mean/std" column are measured under Wilcoxon's signed rank test with a level of significance $\alpha = 0.05$, all these symbols denotes "Better Performance", "Similar Performance" and "Worse Performance" respectively. The overall performances are summarized in last row of the table.

30D	Best		Mean/std	
No.	jSO	HARD-DE	jSO	HARD-DE
fb_1	0(=)	0	4.1797E-015/6.5395E-015(<)	1.9505E-015/4.9388E-015
fb_2	0(=)	0	1.3932E-014/2.6887E-014(<)	2.7864E-015/8.5358E-015
fb_3	0(=)	0	5.2385E-014/1.5434E-014(<)	1.4489E-014/2.5019E-014
fb_4	5.8562E+001(<)	5.6843E-014	5.8562E+001/3.2700E-014(<)	5.5444E+001/1.4061E+001
fb_5	4.9748E+000(>)	7.1432E+000	8.6497E+000/2.0934E+000(>)	1.2287E+001/1.7066E+000
fb_6	0(=)	0	3.2367E-008/9.3835E-008(<)	1.9526E-008/8.0458E-008
fb_7	3.4600E+001(>)	3.7162E+001	3.9339E+001/1.8841E+000(>)	4.3095E+001/2.2645E+000
fb_8	4.9752E+000(>)	9.1220E+000	9.4583E+000/2.1279E+000(>)	1.3101E+001/2.1779E+000
fb_9	0(=)	0	0/0(=)	0/0
fb_{10}	9.6500E+002(<)	8.5613E+002	1.4808E+003/2.5423E+002(>)	1.5698E+003/2.5486E+002
fb_{11}	1.1369E-012(<)	1.1348E+000	7.1131E+000/1.6029E+001(<)	6.8710E+000/1.1953E+001
fb_{12}	1.1244E+001(>)	1.2244E+002	2.2700E+002/1.6507E+002(>)	5.0225E+002/2.6225E+002
fb_{13}	3.8406E+000(<)	9.9498E-001	1.7375E+001/2.6842E+000(<)	1.4127E+001/6.2371E+000
fb_{14}	2.0020E+001(>)	4.9899E+000	2.1897E+001/1.0315E+000(<)	2.0402E+001/6.1033E+000
fb_{15}	4.3465E-001(<)	4.3380E-001	1.2503E+000/9.0304E-001(>)	2.0888E+000/1.2407E+000
fb_{16}	1.3400E+001(>)	2.4671E+001	5.0255E+001/6.3397E+001(>)	1.8204E+002/8.7798E+001
fb_{17}	1.3509E+001(>)	1.7136E+001	3.1718E+001/7.5669E+000(>)	3.5099E+001/7.8986E+000
fb_{18}	4.6254E-001(>)	1.3274E+000	2.0419E+001/2.8814E+000(<)	2.0390E+001/6.3642E-001
fb_{19}	2.9251E+000(<)	1.9122E+000	5.0249E+000/1.7148E+000(>)	6.6457E+000/1.8452E+000
fb_{20}	1.3195E+001(>)	1.8183E+001	2.8954E+001/5.8323E+000(>)	4.3625E+001/1.7770E+001
fb_{21}	2.0580E+002(<)	2.0579E+002	2.1030E+002/1.8239E+000(>)	2.1276E+002/2.0880E+000
fb_{22}	1.0000E+002(=)	1.0000E+002	1.0000E+002/1.2158E-013(=)	1.0000E+002/1.1677E-013
fb_{23}	3.4976E+002(<)	3.4178E+002	3.5735E+002/3.3985E+000(<)	3.4910E+002/3.3390E+000
fb_{24}	4.2673E+002(<)	4.1438E+002	4.3131E+002/2.8783E+000(<)	4.2096E+002/3.0635E+000
fb_{25}	3.8669E+002(=)	3.8669E+002	3.8669E+002/5.1663E-003(=)	3.8670E+002/1.7593E-002
fb_{26}	8.7006E+002(<)	8.1704E+002	9.9050E+002/4.5187E+001(<)	9.1856E+002/4.2476E+001
fb_{27}	4.9150E+002(<)	4.8186E+002	5.0229E+002/5.7289E+000(<)	4.9749E+002/7.2899E+000
fb_{28}	3.0000E+002(=)	3.0000E+002	3.1341E+002/3.7087E+001(>)	3.2556E+002/4.6611E+001
fb_{29}	3.4949E+002(>)	3.8640E+002	4.3420E+002/1.5194E+001(>)	4.3852E+002/1.1164E+001
fb_{30}	1.9417E+003(>)	1.9551E+003	1.9694E+003/1.7286E+001(>)	1.9962E+003/2.6450E+001
$w/d/l$	11/8/11	-	14/3/13	-

proposed HARD-DE outperforms all other contrasted DE variants under our test suite containing 58 benchmark functions. Furthermore, the jSO algorithm suffers an over-fitting problem under CEC2017 test suites, and that's why we recommend to employ more than one test suites for algorithm validation.

E. OPTIMIZATION PERFORMANCE EVALUATION ON REAL-WORLD PROBLEMS

In this section, two real-world optimization problems from CEC2011 [10] test suite were also employed for algorithm validation. The first problem fc_1 is a 6-D parameter estimation for frequency-modulated (FM) sound waves, and the second problem fc_2 is a 20-D spread spectrum radar polly phrase code design, both of which are all bound constrained problem with same range on each dimension. The maximum number of function evaluation is set to $nfe_{\max} = 150000$, and the optimization results of all the DE variants are presented in Table 7 and Table 8 respectively.

We can see that the proposed HARD-DE algorithm performs best of all the contrasted algorithms on fc_1, and all these algorithms occasionally converged into some local optima during the 25 runs. Moreover, the proposed HARD-DE algorithm also outperforms other contrasted algorithms including JADE, LSHADE, iLSHADE and jSO algorithm on fc_2 though it performs relative worse than LPalmDE algorithm. Generally, the canonical DE [31] and some other algorithms, e.g. Genetic Algorithm variants [11], [15], constriction coefficient Particle Swarm Optimization (ccPSO) [2], Ebb-tide-fish [22], [26] and Monkey King Evolution [20] etc., usually performed very well on lower dimensional optimization problems rather than these sophisticated state-of-the-art DE variants mentioned in the paper. This is also the reason why we didn't choose the benchmarks with low dimension in CEC2011 test suite.

To summarize, the proposed HARD-DE is also competitive with the other state-of-the-art DE variants when tackling some real-world optimization problems.

F. EVALUATING THE TIME COMPLEXITY OF THE HARD-DE ALGORITHM

The evaluation of time complexity of the new proposed HARD-DE is presented here in this section. Generally, the proposed HARD-DE algorithm consumes more time in comparison with the other state-of-the-art DE variants, and there are two components responsible for the more time consumed, one is the stochastic universal selection operation (with the pseudo code shown in Algorithm 1) and the other is the extra difference operation $(X_{r_1,G} - \widetilde{X}_{r_3,G})$ in the mutation strategy (shown in Eq.22 on Page 6 of the paper), which presented the depth information of the evolution. The stochastic universal selection operation of HARD-DE is inherited from LPalmDE algorithm, therefore, the HARD-DE may consumes a little more time in comparison with LPalmDE because of the mutation strategy. Furthermore, the time complexity of trial vector generation and parameter adaptation in

TABLE 7. Best and Median comparison among JADE, LSHADE, iLSHADE, jSO, LPALMDE and HARD-DE is presented here. The results are calculated under 25 independent runs with the fixed maximum number of function evaluations nfe_{max} equaling to 150000 under two benchmarks fc_1 and fc_2 selected from CEC2011 test suite for real-parameter optimization.

30D	JADE		LSHADE		iLSHADE		jSO		LPalmDE		HARD-DE	
No.	best	median	best	median	best	median	best	median	best	median	best	median
fc_1	1.0345E-005	3.5692E-002	0	0	0	0	0	0	0	0	0	0
fc_2	9.6176E-001	1.1565E+000	1.0302E+000	1.1586E+000	7.5287E-001	9.7419E-001	9.3667E-001	1.1279E+000	5.0000E-001	6.5583E-001	5.0000E-001	1.0642E+000

TABLE 8. Mean and standard deviation (Std) comparison among JADE, LSHADE, iLSHADE, jSO, LPALMDE and HARD-DE is presented here. The results are calculated under 25 independent runs with the fixed maximum number of function evaluations nfe_{max} equaling to 150000 under two benchmarks fc_1 and fc_2 selected from CEC2011 test suite for real-parameter optimization.

30D	JADE	LSHADE	iLSHADE	jSO	LPalmDE	HARD-DE
No.	Mean/Std	Mean/Std	Mean/Std	Mean/Std	Mean/Std	Mean/Std
fc_1	8.5151E-002/1.4074E-001	1.7975E-001/6.0458E-001	2.0335E+000/4.1509E+000	4.0670E-001/2.0335E+000	1.2757E+000/3.5766E+000	4.7219E-004/1.5926E-003
fc_2	1.1540E+000/9.5060E-002	1.1635E+000/7.0991E-002	9.9951E-001/1.3101E-001	1.1437E+000/1.2197E-001	6.2824E-001/9.7360E-002	9.8147E-001/1.9977E-001

TABLE 9. Algorithm time complexity comparison on 30D optimization under CEC2013 benchmark 14.

Algorithms.	T_0	T_1	$\widehat{T}_2$	$\frac{\widehat{T}_2-T_1}{T_0}$
JADE	0.1181	0.8298	1.4417	5.1812
LSHADE			1.8385	8.5411
iLSHADE			1.9526	9.5072
jSO			1.8928	9.0008
LPalmDE			2.0978	10.7367
HARD-DE			2.1900	11.5174

LSHADE, iLSHADE, jSO and LPalmDE are more or less the same, therefore, we can simply sort the time complexity of these algorithms by qualitative analysis.

Beside the qualitative analysis, we also conducts quantitative analysis according to the recommendation of CEC2013 competition [18], and the time consumption comparison is presented in Table 9. Time consumption of basic arithmetic expressions in CEC2013 competition recommendation is recorded as T_0, the time consumption of 200000 function evaluations for 30D optimization on benchmark fa_{14} from CEC2013 test suite is recorded as T_1, and the overall cost of a certain algorithm optimizing fa_{14} is recorded as T_2. We run 11 times to get the average T_0, T_1 and T_2, and then collect $\frac{\widehat{T}_2-T_1}{T_0}$ for complexity evaluation. Table 9 illustrated the time complexity comparison among these six algorithms, and we can see that the proposed HARD-DE algorithm consumes more time especially in comparison with the JADE algorithm.

V. CONCLUSION

Differential evolution variants are famous, easy-coding and powerful stochastic algorithms for many complex optimization problems, however, there still exist many weaknesses within them. This paper proposed a new hierarchical archive based differential evolution called HARD-DE algorithm, and two enhancements were involved in this DE variant. The first enhancement was that a new hierarchical archive based trial vector generation strategy with depth information of evolution was proposed to get better perception of the landscape of the objective functions. The second enhancement was that novel adaptation schemes for all the three control parameters were advanced in the paper to tackle the weaknesses of parameter adaptations in some former famous DE variants. Then the new HARD-DE algorithm is verified under two test suites, CEC2013 test suite and CEC2017 test suite containing 58 benchmark functions, for real-parameter numerical optimization and 2 benchmarks from CEC2011 test suite for real-world optimization. Both the optimization accuracy (measured by "Mean/Std" values under Wilcoxon's singed rank test with a level of significance $\alpha = 0.05$) and the convergence speed (measured by the convergence curve of the "Median" value from 51 runs) of all the six DE variants are contrasted for algorithm verification. From the experiment results, we can see that: First, the novel HARD-DE-Para algorithm secured an overall better performance on all tested unimodal benchmarks in comparison with HARD-DE-Linear and HARD-DE-Pivot (the default HARD-DE algorithm), and the HARD-DE-Linear secured an overall better performance on many multi-modal benchmarks in comparison with HARD-DE-Para. The default HARD-DE secured an overall better performance on the tested benchmarks in comparison with HARD-DE-Linear and HARD-DE-Pare. Second, the default HARD-DE algorithm obtained an overall better performance in comparison with other famous DE variants including JADE, LSHADE, iLSHADE, jSO and LPALMDE on real-parameter optimization benchmarks selected from CEC2013 and CEC2017 test suites. Third, algorithm verification on a single test suite containing a relative small number of benchmarks may have over-fitting problem, therefore, more than one test suites containing a large number of benchmarks were recommended to be employed in algorithm evaluation in this paper. Fourth, the proposed HARD-DE algorithm was also competitive with the contrasted DE variants on real-world optimization problems selected from CEC2011 test suite. To summarize, the novel HARD-DE algorithm proposed in the paper secured an overall better performance in

comparison with other famous DE variants under our test suite though it might consume more time.

REFERENCES

[1] M. Z. Ali, N. H. Awad, P. N. Suganthan, and R. G. Reynolds, "An adaptive multipopulation differential evolution with dynamic population reduction," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2768–2779, Sep. 2017.

[2] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp. (SIS)*, Apr. 2007, pp. 120–127.

[3] J. Brest, A. Zamuda, B. Boskovic, M. S. Maucec, and V. Zumer, "High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction," in *Proc. IEEE Congr. Evol. Comput. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 2032–2039.

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

[5] J. Brest and M. S. Mau ec, "Population size reduction for the differential evolution algorithm," *Appl. Intell.*, vol. 29, no. 3, pp. 228–247, 2008.

[6] J. Brest, M. S. Mau ec, and B. Bošković, "iL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 1188–1195.

[7] J. Brest, M. S. Mau ec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2017, pp. 1311–1318.

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

[9] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[10] S. Das and P. N. Suganthan, *Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems*. Kolkata, India: Jadavpur University, 2010.

[11] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "GA with a new multi-parent crossover for solving IEEE-CEC2011 competition problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2011, pp. 1034–1040.

[12] V. Feoktistov and S. Janaqi, "Generalization of the strategies in differential evolution," in *Proc. 18th Int. Parallel Distrib. Process. Symp.*, Apr. 2004, p. 165.

[13] R. Gämperle, S. D. Müller, and P. Koumoutsakos, "A parameter study for differential evolution," *Adv. Intell. Syst., fuzzy Syst., Evol. Comput.*, vol. 10, no. 10, pp. 293–298, 2002.

[14] A. Gosh, S. Das, R. Mallipeddi, A. K. Das, and S. S. Dash, "A modified differential evolution with distance-based selection for continuous optimization in presence of noise," *IEEE Access*, vol. 5, pp. 26944–26964, 2017.

[15] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI, USA: University of Michigan Press, 1975.

[16] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI, USA: University of Michigan, 1975.

[17] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[18] J. Liang, B. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," Nanyang Technological University, Singapore, Tech. Rep. 201212, 2013.

[19] Z. Meng and J.-S. Pan, "QUasi-affine TRansformation Evolutionary (QUATRE) algorithm: A parameter-reduced differential evolution algorithm for optimization problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 4082–4089.

[20] Z. Meng and J.-S. Pan, "Monkey king evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization," *Knowl.-Based Syst.*, vol. 97, pp. 144–157, Apr. 2016.

[21] Z. Meng and J.-S. Pan, "QUasi-Affine TRansformation Evolution with External ARchive (QUATRE-EAR): An enhanced structure for differential evolution," *Knowl.-Based Syst.*, vol. 155, pp. 35–53, Sep. 2018.

[22] Z. Meng, J.-S. Pan, and A. Alelaiwi, "A new meta-heuristic ebb-tide-fish-inspired algorithm for traffic navigation," *Telecommun. Syst.*, vol. 62, no. 2, pp. 403–415, 2016.

[23] Z. Meng, J.-S. Pan, and L. Kong, "Parameters with adaptive learning mechanism (PALM) for the enhancement of differential evolution," *Knowl.-Based Syst.*, vol. 141, pp. 92–112, Feb. 2018.

[24] Z. Meng, J.-S. Pan, and H. Xu, "QUasi-Affine TRansformation Evolutionary (QUATRE) algorithm: A cooperative swarm based algorithm for global optimization," *Knowl.-Based Syst.*, vol. 109, pp. 104–121, Oct. 2016.

[25] E. Mezura-Montes and J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Proc. 8th Annu. Conf. Genetic Evol. Comput.*, 2006, pp. 485–492.

[26] J.-S. Pan, Z. Meng, S.-C. Chu, and H.-R. Xu, "Monkey king evolution: An enhanced ebb-tide-fish algorithm for global optimization and its application in vehicle navigation under wireless sensor network environment," *Telecommun. Syst.*, vol. 65, no. 3, pp. 351–364, 2017.

[27] J.-S. Pan, Z. Meng, H. Xu, and X. Li, "QUasi-Affine TRansformation Evolution (QUATRE) algorithm: A new simple and accurate structure for global optimization," in *Proc. Int. Conf. Ind., Eng. Other Appl. Appl. Intell. Syst.* Morioka, Japan: Springer, 2016, pp. 657–667.

[28] J.-S. Pan, Z. Meng, H. Xu, and X. Li, "A matrix-based implementation of de algorithm: The compensation and deficiency," in *Proc. Int. Conf. Ind., Eng. Other Appl. Appl. Intell. Syst.* Arras, France: Springer, 2017, pp. 72–81.

[29] F. Peng, K. Tang, G. Chen, and X. Yao, "Multi-start jade with knowledge transfer for numerical optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 1889–1895.

[30] A. P. Piotrowski, "Review of differential evolution population size," *Swarm Evol. Comput.*, vol. 32, pp. 1–24, Feb. 2017.

[31] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer, 2006.

[32] K. V. Price, "Differential evolution: A fast and simple numerical optimizer," in *Proc. North Amer. Fuzzy Inf. Process.*, Jun. 1996, pp. 524–527.

[33] K. V. Price, "Differential evolution vs. the functions of the 2^{nd} ICEO," in *Proc. IEEE Int. Conf. Evol. Comput.*, Indianapolis, IN, USA, Apr. 1997, pp. 153–157.

[34] K. V. Price, "Eliminating drift bias from the differential evolution algorithm," in *Advances in Differential Evolution*. Berlin, Germany: Springer, 2008, pp. 33–88.

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

[36] R. Storn and K. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces," Int. Comput. Sci. Inst., Berkeley, CA, USA, Tech. Rep., 1995.

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

[38] R. Tanabe and A. S. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2013, pp. 71–78.

[39] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 1658–1665.

[40] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Comput.*, vol. 10, no. 8, pp. 673–686, 2006.

[41] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants," *Inf. Sci.*, vol. 423, pp. 172–186, Jan. 2018.

[42] A. Zamuda, J. Brest, B. Boskovic, and V. Zumer, "Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution," in *Proc. IEEE Congr. Evol. Comput. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 3718–3725.

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

ZHENYU MENG (M'17) received the B.S. degree from Shandong Normal University, in 2008, the M.Phil. degree from the Harbin Institute of Technology Shenzhen Graduate School, in 2011, and the Ph.D. degree from the Harbin Institute of Technology Shenzhen, in 2018, all in computer science.

After completing the M.Phil. degree, he was with the Guangzhou Institute of Advanced Technology, Chinese Academy of Sciences, as a Research Assistant, from 2012 to 2013, before pursuing the Ph.D. degree. He is currently a Professor with the Fujian Key Provincial Key Laboratory of Data Mining and Application, Fujian University of Technology, and also a Research Fellow with the Harbin Institute of Technology Shenzhen. His research interests include evolutionary computation, computer vision, and vehicle navigation.

JENG-SHYANG PAN (SM'09) received the B.S. degree in electronic engineering from the National Taiwan University of Science and Technology, in 1986, the M.S. degree in communication engineering from National Chiao Tung University, Taiwan, in 1988, and the Ph.D. degree in electrical engineering from the University of Edinburgh, U.K., in 1996.

He is currently the Dean of the College of Information Science and Engineering, Fujian University of Technology, and the Director of the Innovative Information Industry Research Center, Harbin Institute of Technology, Shenzhen, China. He joined the Editorial Board of *LNCS Transactions on Data Hiding and Multimedia Security*, the *Journal of Computers*, the *Journal of Information Hiding*, and *Multimedia Signal Processing*. His current research interests include soft computing, information security, and signal processing.