Proposal of Adaptive Randomness in Differential Evolution

Junya Tsubamoto
Graduate School of Engineering
Osaka Prefecture University
Osaka, Japan
tsubamoto@hi.cs.osakafu-u.ac.jp

Akira Notsu
Graduate School of Humanities and Sustainable System Sciences
Osaka Prefecture University
Osaka, Japan
notsu@cs.osakafu-u.ac.jp

Seiki Ubukata, Katsuhiro Honda
Graduate School of Engineering
Osaka Prefecture University
Osaka, Japan
{subukata, honda}@cs.osakafu-u.ac.jp

***Abstract*—Differential evolution (DE) is a widely used optimization algorithm, which can achieve high accuracy with a simple mechanism, but sometimes have only limited performances due to its simplicity. In order to mitigate the inappropriate effect of poor initial search points, it is known that adding a random search to DE contributes to obtain better results than normal DE. However, it is inefficient to perform many random searches when the search process is almost converged. In this study, we propose a novel method of DE with Adaptive Randomness (DEAR), which is a hybrid of two promising algorithms of DIEtoDE and SaDE, and can adaptively change the frequency of random search maintaining efficiency. Numerical experiments demonstrated that the proposed method can identify better solutions than other comparative methods.**

***Index Terms*—Optimization problem, differential evolution, adaptive randomness**

I. Introduction

Parameter optimization is a very important problem for many machine learning including neural networks [1]. Many optimization algorithms have been proposed to quickly obtain optimal solutions for these optimization problems. For example, we have several nature-inspired algorithms such as Genetic Algorithm (GA) [2], [3], which is an algorithm expressing human genetics, and Particle Swarm Optimization (PSO) [4], [5], which is an algorithm imitating the movement of a flock of birds.

Differential evolution (DE) [6], [7] is a widely used optimization algorithm because it achieves high accuracy with a simple procedure. However, since the performance of DE varies greatly depending on parameter values and the positions of initial individuals, many improved algorithms have been proposed to address these problems [8]. For example, there are algorithms focusing on DE search methods [9] and automatic parameter adjustment [10], [11].

We previously proposed Differential Evolution with Intervals (DIE) [12] to mitigate the effects of initial search points in DE. DIE is a method of adding random search within confidence intervals in a range according to the number of searches to DE. In addition, we also proposed DIEtoDE [12], which is a method which switches from DIE to DE on the way for the purpose of accelerating the convergence of solution processes. We have confirmed that DIEtoDE can obtain faster and better solutions for many objective functions compared to the conventional DE.

However, DIEtoDE is a one-way algorithm that switches from DIE to DE. In other words, once switching to DE, DIEtoDE does not perform random search thereafter. Therefore, DIEtoDE may not find a good solution because it does not perform the necessary random searches within confidence intervals in the latter stage of the search. For some objective functions, DIEtoDE is unable to properly switch from DIE to DE, and so, it could not make a clear advantage to the solution obtained by DE.

In this study, we propose DE with Adaptive Randomness (DEAR) which incorporates the concept of SaDE [13], [14] to stochastically perform random search even at the last stage of the search. SaDE is a method for adaptively selecting one strategy from multiple individual generation strategies. By introducing these strategies to DE, DIE (DE with random search within the confidence interval), and random search from all areas, DEAR can always offer randomness adaptively.

The structure of this paper is as follows. Section 2 outlines DE and its improved methods. In Section 3, we propose a novel algorithm which improves upon DE. Next, in Section 4, numerical experiments are carried out using the proposed method and the results are examined. Finally, conclusions are drawn in Section 5 including problems for future research.

II. Differential Evolution and Improved Methods

A. Differential Evolution (DE)

In DE, an efficient search is carried out by saving some good search points and generating a next search point using them. Genetic operators such as mutation, crossover, and survivor selection are used to generate and select subsequent search points. DE has been shown to exhibit high speed and high searching performance for various optimization problems [15]. In differential evolution, the focus is on three issues: a) how to select the basic vector, b) the number of difference vectors, and c) the type of crossover. We use the notation of the form (DE/a/b/c) herein to distinguish particular DE algorithms. The algorithm for (DE/rand/1/bin) is shown below.

Step 1 N individuals $\{\boldsymbol{x}_i, i = 1, 2, \ldots, N\}$ are randomly generated as initial search points.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

Step 2 Repeat the following until some stopping conditions are met.

2-1 For each individual $\boldsymbol{x}_i$, an individual $\boldsymbol{v}$ is generated from the following equation where $\boldsymbol{x}_a$, $\boldsymbol{x}_b$, and $\boldsymbol{x}_c$ are individuals that do not overlap with each other including $\boldsymbol{x}_i$ in the solution group. F is the mutation coefficient.

$$\boldsymbol{v} = \boldsymbol{x}_a + F(\boldsymbol{x}_b - \boldsymbol{x}_c) \tag{1}$$

2-2 $\boldsymbol{x}_o$ is generated by uniformly crossing $\boldsymbol{x}_i$ and $\boldsymbol{v}$ as in the following equation.

$$x_{oj} = \begin{cases} v_j & (w \leq CR) \\ x_{ij} & (otherwise) \end{cases} \tag{2}$$

where w represents a uniform random variable within the range [0,1] and CR is a crossover rate.

2-3 Evaluate the solution of $\boldsymbol{x}_o$ and replace $\boldsymbol{x}_i$ with $\boldsymbol{x}_o$ if the search result is better than $\boldsymbol{x}_i$.

Step 3 Let the best solution among the searched points be the final solution.

N, F and CR are user-defined parameters. Regarding the generation method of the individual $\boldsymbol{v}$, a plurality of mutation strategies other than (DE/rand/1/bin) are implemented as shown in the following formulas. Here, $\boldsymbol{x}_{best}$ is an individual exhibiting the best solution in the solution group. $\boldsymbol{x}_d$ and $\boldsymbol{x}_e$ are individuals selected so as not to be duplicated in the solution population similarly to $\boldsymbol{x}_a$, $\boldsymbol{x}_b$ and $\boldsymbol{x}_c$. K is a user-defined parameter.

- DE/rand-to-best/2/bin

$$\boldsymbol{v} = \boldsymbol{x}_i + F(\boldsymbol{x}_{best} - \boldsymbol{x}_i) + F(\boldsymbol{x}_a - \boldsymbol{x}_b) + F(\boldsymbol{x}_c - \boldsymbol{x}_d) \tag{3}$$

- DE/rand/2/bin

$$\boldsymbol{v} = \boldsymbol{x}_a + F(\boldsymbol{x}_b - \boldsymbol{x}_c) + F(\boldsymbol{x}_d - \boldsymbol{x}_e) \tag{4}$$

- DE/current-to-rand/1

$$\boldsymbol{v} = \boldsymbol{x}_i + K(\boldsymbol{x}_a - \boldsymbol{x}_i) + F(\boldsymbol{x}_b - \boldsymbol{x}_c) \tag{5}$$

B. Differential Evolution with Intervals (DIE)

In differential evolution, since the initial search points are given randomly, search point generation largely depends on the initial search point set by chance, and the search process may not be successful depending on the shape of the objective function. This motivated the development of DIE.

Assume that there are N search points in a space where each dimension is normalized from 0 to 1 and each search point is, on average, responsible for searching volume $1/N$. Because the decreasing rate of the space to be searched by each point is expected to be the same as that of the confidence intervals in the estimation of the mean and variance, it is proportional to $(1/I)^m$ with the number of searches I and a constant m.

In DIE, the next search point is randomly shifted within the current search range. Let D be the dimensionality of the search space, and make the search range a hypercube of $1/N^{1/D}I^{1/2}$ per side, and add a term which moves randomly in this range in the generation of the next search point. In [12], $m = 1/2$ was set. Equation (1) for the generation of individual $\boldsymbol{v}$ in DE is modified as follows.

$$v_j = x_{a,j} + F\left(x_{b,j} - x_{c,j}\right) + H\left(\frac{rand() - 0.5}{N^{1/D}I^{1/2}}\right) \tag{6}$$

where $rand()$ is a function outputting a uniform random number from 0 to 1, and H is a variable that takes the value of 1 for random search and 0 otherwise.

DIEtoDE is a method which switches from DIE to DE to accelerate solution convergence. If an objective function is unimodal and the solution exists inside a set of search points, the spread of the set of search points becomes exponentially smaller by the DE algorithm. Since the random search range of DIE becomes small in the polynomial order, if the search range of the search point is narrowed to the range of unimodality and the search is continued for a while, the spread of the search point group is expected to become smaller than the random search range of DIE.

Therefore, the DIE random search is applied until the length of one side of the hypercube including the search point group becomes smaller than the length of one side of the range of random movement. The length L of each dimension indicating the extent of the set of search points is calculated by the following equation.

$$L = \frac{1}{D}\sum_{j=1}^{D}(\max_i x_{i,j} - \min_i x_{i,j}) \tag{7}$$

Therefore, the value of variable H that controls the random search of DIEtoDE is switched under the following conditions.

$$H = \begin{cases} 1 & (L \geq \frac{1}{N^{1/D}I^{1/2}}) \\ 0 & (otherwise) \end{cases} \tag{8}$$

C. Self-adaptive Differential Evolution (SaDE)

SaDE is a method that uses memory to store past search information and performs adaptive adjustment of mutation strategy and CR. In normal DE, only one type of mutation strategy is set, and the CR is preset by each user. However, since the performance of DE depends greatly on these strategies and parameters [16], good results may not be obtained depending on the shape of the objective function.

SaDE obtains better approximate solutions than normal DE in many objective functions by implementing a function for stochastically selecting four strategies: (DE/rand/1/bin), (DE/rand-to-best/2/bin), (DE/rand/2/bin), and (DE/current-to-rand/1) for each individual in each generation and leading to adaptively adjustment of parameters. In the following, the case where a child individual is better than the parent individual is called "success", otherwise it is called "failure".

After setting LP as the number of generations, the numbers of successful and failed searches in each strategy during the LP generations are stored. The probability of selecting each strategy in the next trial vector generation is calculated from the number of failures and successes during LP generations. When preparing K strategies, the initial value of the probability of selecting each strategy is set $1/K$. After saving

the LP generations, if the probability that the kth strategy is selected in the Gth generation is $p_{k,G}$, $p_{k,G}$ is represented by the following equation:

$$p_{k,G} = \frac{S_{k,G}}{\sum_{k=1}^{K} S_{k,G}} \tag{9}$$

where

$$S_{k,G} = \frac{\sum_{g=G-LP}^{G-1} ns_{k,g}}{\sum_{g=G-LP}^{G-1} ns_{k,g} + \sum_{g=G-LP}^{G-1} nf_{k,g}} + \varepsilon \tag{10}$$

in which $S_{k,G}$ represents the success rate of the trial vectors generated by the kth strategy entering the next generation within the previous LP generations with respect to generation G. $ns_{k,g}$ and $nf_{k,g}$ are respectively the numbers of successes and failures when strategy k is selected in generation g. The small constant value $\varepsilon = 0.01$ is used to avoid possible null success rates.

In normal DE, parameters F and CR are defined by the user, but in SaDE, they change adaptively from their initial values. F is generated randomly using normal random numbers, and no adaptive adjustment is performed. On the other hand, CR records the values that succeeded in the past in memory for each strategy, and random numbers are generated based on a normal distribution having the same average value with CR stored in memory.

III. Proposed algorithm

DIEtoDE, which we proposed in an early study, is a one-way algorithm. Thus, once DIE switches to DE, DE does not implement random searches and, as such, sometimes cannot find a good solution. Therefore, we propose a novel algorithm of DE with adaptive randomness (DEAR). This algorithm can perform a random search even in the latter stage of the search supported by the idea of the strategy selection of SaDE.

SaDE adaptively selects a DE strategy to generate the trial vector $\boldsymbol{v}$. In DEAR, instead of the DE individual generation strategy, a new individual is generated by probabilistic selection from three methods: DE, DIE, and random search, i.e., SaDE-like strategy selection is performed with $K = 3$ using different candidates. In DEAR, the probability of choosing one of these three strategies is calculated by the above formula (9) using the numbers of successes and failures in each strategy, as in the case of SaDE. When DE is selected based on the probability calculated in (9), an individual is generated using (1) and (2). If DIE is selected, an individual is generated using (6) and (2). When random search is selected, one individual is generated randomly from the entire range of the domain.

The parameters in DEAR are set so that they can be adaptively changed. F is generated randomly using normal random numbers, and no adaptive adjustment is performed. CR records the values of CRs that succeeded in the past in memory for each strategy, and random numbers are generated based on a normal distribution having the same average value with CR stored in memory.

The main difference between DEAR and SaDE is the responsibility of random search. The performance of SaDE is

Algorithm 1 DEAR algorithm

[Step 1] Set the generation counter $G = 0$, initialize the average value of CR and strategy selection probabilities, and generate initial search points $\{\boldsymbol{x}_i, i = 1, 2, \ldots, N\}$ randomly.
[Step 2] Evaluate the population.
[Step 3]
while some stopping conditions are not met **do**
  [Step 3.1] Calculate strategy probability $p_{k,G}$ and update the Success and Failure Memory.
  if $G > LP$ **then**
    for $k = 1$ to K **do**
      Update the $p_{k,G}$ by equation (9).
      Remove $ns_{k,G-LP}$ and $nf_{k,G-LP}$ out of the Success and Failure Memory, respectively.
    end for
  end if
  [Step 3.2] Select one strategy k from the candidate strategies such as DE, DIE and random search for each target vector $\boldsymbol{x}_i$ using stochastic universal sampling.
  for $i = 1$ to N **do**
    $F_i = Normrnd(0.5, 0.3)$
  end for
  if $G >= LP$ **then**
    for $k = 1$ to K **do**
      $CRm_k = average(CRMemory_k)$
    end for
  end if
  for $k = 1$ to K **do**
    for $i = 1$ to N **do**
      $CR_{k,i} = Normrnd(CRm_k, 0.1)$
    end for
  end for
  [Step 3.3] Generate a new population where each trial vector $\boldsymbol{v}$ is generated according to associated trial vector generation strategy k and parameters F_i and $CR_{k,i}$ in previous step.
  [Step 3.4]
  for $i = 1$ to N **do**
    Evaluate the trial vector $\boldsymbol{v}$.
    if $f(\boldsymbol{v}) < f(\boldsymbol{x}_i)$ **then**
      $\boldsymbol{x}_i = \boldsymbol{v}$, $f(\boldsymbol{x}_i) = f(\boldsymbol{v})$
      $ns_{k,G} = ns_{k,G} + 1$
      Store $CR_{k,i}$ into $CRMemory_k$.
    else
      $nf_{k,G} = nf_{k,G} + 1$
    end if
  end for
  Store $ns_{k,G}$ and $nf_{k,G}$ into the Success and Failure Memory, respectively.
  [Step 3.5] Increment the generation count $G = G + 1$.
end while

strongly affected by the nature of DE because SaDE uses only DE in individual generation. Therefore, in the case where DE cannot provide good results for a function, SaDE may also fail to obtain good results for the function. However, the proposed method, DEAR, is a method that can control the responsibility of random search, so it is expected that global search can be performed without being restricted by the nature of DE.

The procedure of the proposed method, DEAR, is shown as Algorithm 1 in a pseudo code following [13], where the number of strategies is generalized as K but can be set as $K = 3$ for the three strategies: DE, DIE and random search in the above proposal.

IV. Numerical experiment

A. Test functions and experiment setting

A comparative experiment was carried out between the conventional methods and the proposed method: DE, DIEtoDE, SaDE, DEAR. We used 8 test functions referring to the literature [17], [18]. In this experiment, we only considered minimization problems, in which the goal is to search for the minimum values of the objective functions. Table I shows the mathematical expressions of the functions.

Brief descriptions of the functions used in this experiment are as follows: F_1 function is a unimodal function with only one local solution. F_2 function is a multimodal function with many local solutions in the domain. F_3 function is a unimodal function, but the optimal solution can be obtained only when the search points are located over an extremely narrow range. In F_4 function, the optimal solution is in a narrow parabolic valley and it is very difficult to find the optimal solution. F_5 function is a multimodal function that has large values at the edge of the domain but has small deviations near the optimal solution. F_6 function is a multimodal function that has many local solutions at the edge of the domain. F_7 function is a multimodal function, and the search points need to enter over an extremely narrow range to obtain the optimal solution. In F_8 function, there are many large peaks around the optimal solution, and it is difficult to narrow the search range.

Regarding parameter settings, for DIE and DE, the crossover rate CR and mutation F were both set to 0.5. For SaDE and DEAR, the parameter F is randomly set with a normal distribution having an average value of 0.5 and a standard deviation of 0.3 for each time an individual is generated. Regarding CR, a value is stored when the individual is updated for each strategy, and the average is set as CRm. The CR is randomly set from a normal distribution with a mean CRm and a standard deviation of 0.3. Finally, for both SaDE and DEAR, $LP = 50$.

B. Experimental result

Figs. 1-8 show the experimental results of the above test functions using the four algorithms. D represents the number of decision variables of the function, and N represents the number of individuals per generation in the search. The horizontal axis indicates the number of searches and the vertical axis indicates the objective function values, and the average value of 1000 trials of the best solution in each search is plotted. Since this experiment deals with minimization problems, a smaller value is better. In addition, Table II shows the average of the best solutions obtained by each method when the 1000th individual was generated.

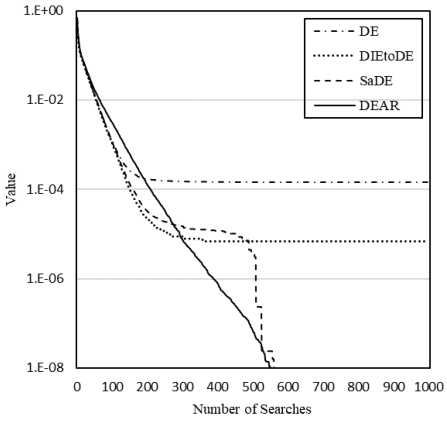

Fig. 1. F_1 function : D2, N10

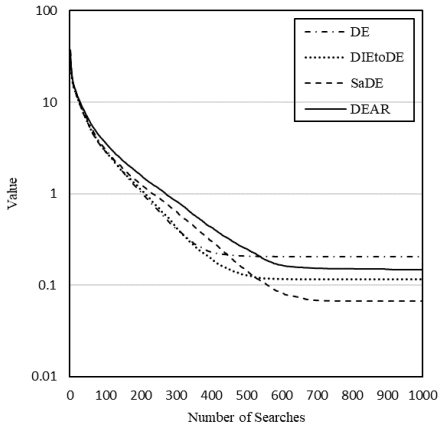

Fig. 2. F_2 function : D2, N10

In F_1 function, DEAR performed better than the other methods. In normal DE, individuals sometimes gathered at a position other than the optimal solution in some trials out of 1,000 trials. Therefore, when averaging, the accuracy was worse than other methods. However, it is considered that DEAR was able to prevent individuals from gathering in the first half of the search by random search. In F_2 function, DEAR got worse results than DIEtoDE and SaDE. Since this function has a very large number of local solutions in the domain, it is considered that a random search has converged to a local solution at a position away from the optimal solution.

In F_3 and F_4 functions, DEAR achieved slightly better results than DIEtoDE and SaDE. Since DEAR includes more random searches in the first half of the search than the other

TABLE I
TEST FUNCTIONS USED IN THIS EXPERIMENT

function	formula, optimal solution
F1: Sphere function	$f(x_1, \cdots, x_D) = \sum_{j=1}^{D} x_j^2$ $-1 \leq x_j \leq 1$ $f_{min}(0, \cdots, 0) = 0$
F2: Rastrigin function	$f(x_1, \cdots, x_D) = 10D + \sum_{j=1}^{D}(x_j^2 - 10\cos(2\pi x_j))$ $-5 \leq x_j \leq 5$ $f_{min}(0, \cdots, 0) = 0$
F3: Easom function	$f(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-((x_1-\pi)^2 + (x_2-\pi)^2))$ $-100 \leq x_j \leq 100$ $f_{min}(\pi, \pi) = -1$
F4: Rosenbrock function	$f(x_1, \cdots, x_D) = \sum_{j=1}^{D-1}(100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2)$ $-5 \leq x_j \leq 5$ $f_{min}(1, \cdots, 1) = 0$
F5: Beale function	$f(x_1, x_2) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$ $-4.5 \leq x_j \leq 4.5$ $f_{min}(3, 0.5) = 0$
F6: Xin-She Yang function	$f(x_1, \cdots, x_D) = \left(\sum_{j=1}^{D}\lvert x_j\rvert\right)\exp\left(-\sum_{j=1}^{D}\sin(x_j^2)\right)$ $-2\pi \leq x_j \leq 2\pi$ $f_{min}(0, \cdots, 0) = 0$
F7: Ackley function	$f(x_1, \cdots, x_D) = 20 - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D} x_j^2}\right) + e - \exp\left(\frac{1}{D}\sum_{j=1}^{D}\cos(2\pi x_j)\right)$ $-32.768 \leq x_j \leq 32.768$ $f_{min}(0, \cdots, 0) = 0$
F8: Schaffer function	$f(x_1, x_2) = 0.5 + \frac{\sin^2(x_1^2 - x_2^2) - 0.5}{\left(1 + 0.001(x_1^2 + x_2^2)\right)^2}$ $-100 \leq x_j \leq 100$ $f_{min}(0, 0) = 0$

TABLE II
MEAN $\pm$ SD OF THE BEST SOLUTIONS IN 1000 SEARCHES (BEST PERFORMANCES ARE DEPICTED IN BOLD.)

	DE	DIEtoDE	SaDE	DEAR
F_1	0.0001468 ± 0.002184	$0.000006697 \pm 0.0001222$	$\mathbf{0.000000 \pm 0.000000}$	$\mathbf{0.000000 \pm 0.000000}$
F_2	0.2052 ± 0.5630	0.1164 ± 0.4638	$\mathbf{0.06666 \pm 0.2716}$	0.1493 ± 0.3822
F_3	-0.9649 ± 0.1778	-0.9934 ± 0.07371	-0.9937 ± 0.07188	$\mathbf{-0.9998 \pm 0.003659}$
F_4	0.2128 ± 0.6884	0.08517 ± 0.3205	0.02411 ± 0.1078	$\mathbf{0.01630 \pm 0.05686}$
F_5	0.09236 ± 0.3909	0.06041 ± 0.2143	0.02189 ± 0.1218	$\mathbf{0.002884 \pm 0.04521}$
F_6	0.05721 ± 0.1215	0.04851 ± 0.1182	$\mathbf{0.02704 \pm 0.09046}$	0.05719 ± 0.1247
F_7	0.1416 ± 1.0115	0.008857 ± 0.1416	0.01290 ± 0.1820	$\mathbf{0.002636 \pm 0.08154}$
F_8	0.001673 ± 0.01624	0.001089 ± 0.01578	$\mathbf{0.0003043 \pm 0.003210}$	0.0008764 ± 0.002644

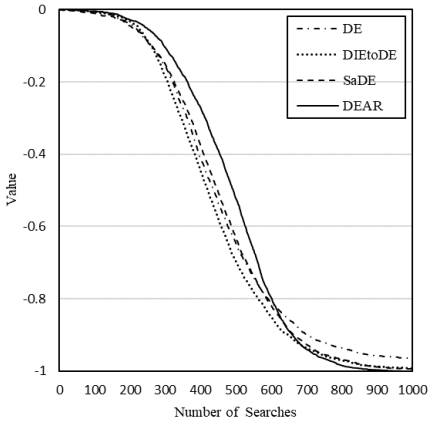

Fig. 3. F_3 function : D2, N10

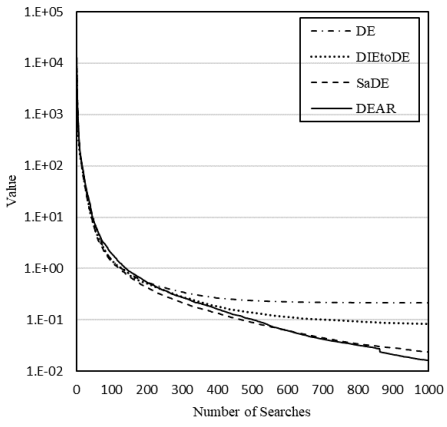

Fig. 4. F_4 function : D2, N10

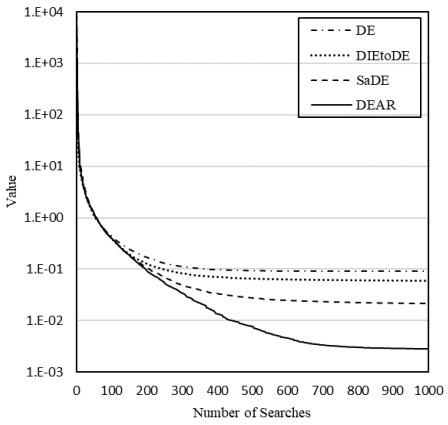

Fig. 5. F_5 function : D2, N10

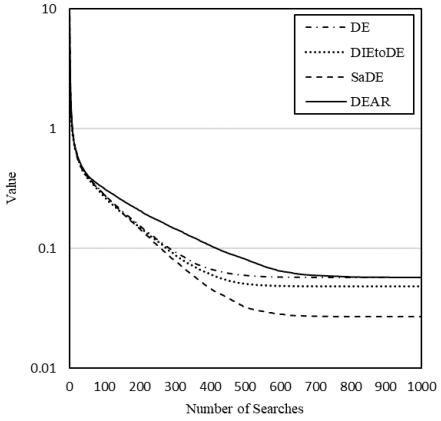

Fig. 6. F_6 function : D2, N10

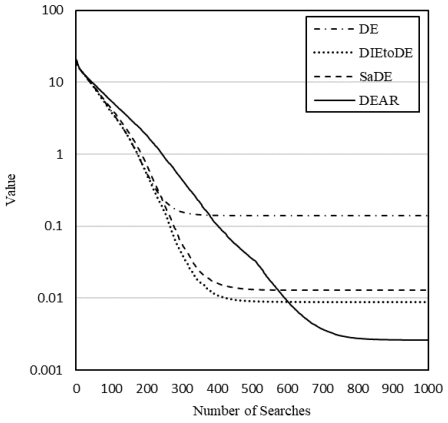

Fig. 7. F_7 function : D2, N10

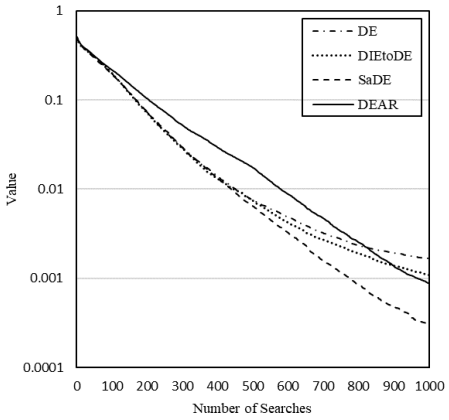

Fig. 8. F_8 function : D2, N10

methods, it can obtain better results in these functions, which are difficult to obtain optimal solutions in DE.

DEAR achieved much better results in F_5 and F_7 functions than the other methods. Since the F_5 function is a multimodal function with a small height difference near the optimal solution, DIEtoDE and SaDE fell into local solutions. However, DEAR has found better solutions than these comparative methods. Since the F_7 function has a shape like a unimodal function but is a multimodal function with many local solutions in a fine range, SaDE and DIEtoDE were tricked such that the function looks like unimodal and chose inappropriate parameters and selection strategies. However, the proposed method, DEAR, performed better than these methods by adopting random search even in the last stage.

Conversely, for the F_6 function, DEAR could not provide a better solution than DIEtoDE and SaDE. The F_8 function has a large number of local solutions far from the optimal solution. It is probable that good results could not be obtained because a new individual was generated at a position far from the optimal solution by the random search at the beginning of the search.

In F_8 function, DEAR performed better than DIEtoDE and performed worse than SaDE. However, DEAR did not converged in some trials when 1000 individuals are generated. Therefore, DEAR may get better results than SaDE by increasing the number of searches.

The challenge with DEAR is that it adopts random search more frequently than the other methods, so it takes time to get plausible solutions for many objective functions. In the proposed method, the initial probability in each strategy is $1/K$ as in SaDE. Therefore, it is considered that this issue can be alleviated by reducing the initial probability in the random search.

Finally, Fig. 9 to Fig. 16 show changes in the probability that DE, DIE, and random search are selected in each function when DEAR is used. Since this probability is updated every time the generation is updated, the horizontal axis represents the change of the generation. Since $LP = 50$ for all functions,

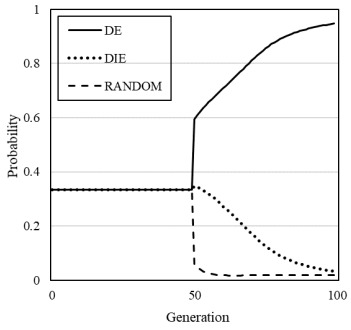

Fig. 9. Selection probability in F_1 function

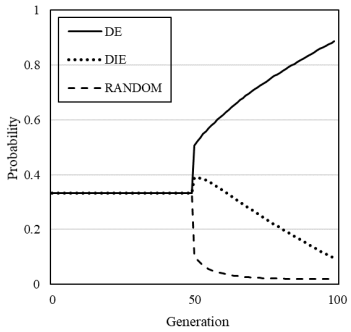

Fig. 10. Selection probability in F_2 function

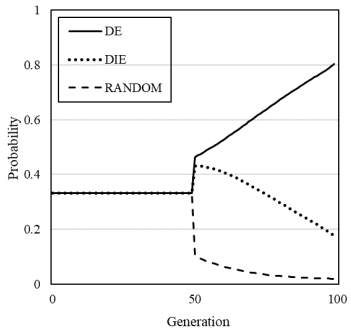

Fig. 11. Selection probability in F_3 function

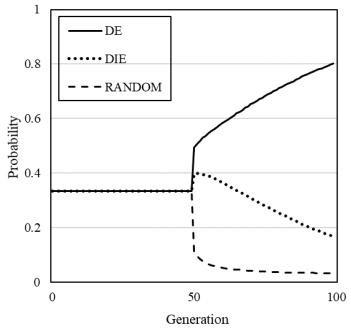

Fig. 12. Selection probability in F_4 function

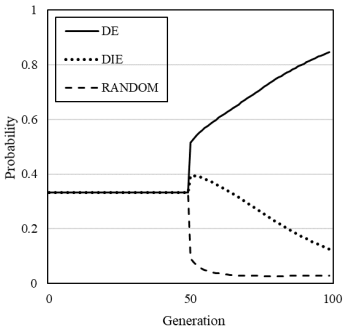

Fig. 13. Selection probability in F_5 function

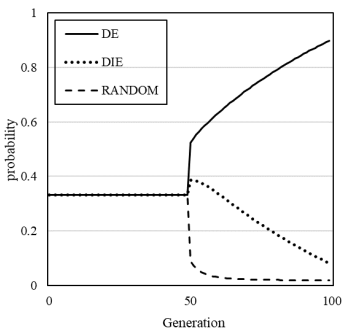

Fig. 14. Selection probability in F_6 function

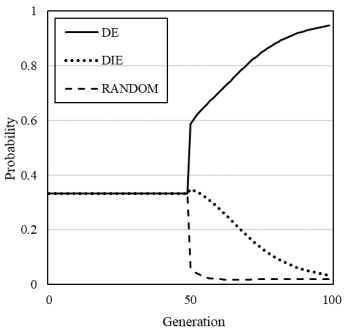

Fig. 15. Selection probability in F_7 function

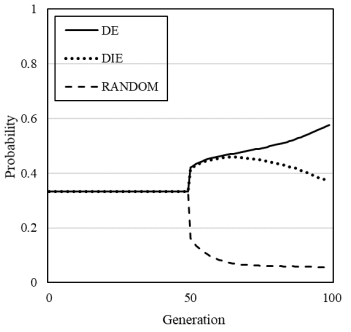

Fig. 16. Selection probability in F_8 function

the probability of selecting each method is $1/3$ until the 50th generation. Basically, after 50 generations, random search decreases rapidly, and DIE gradually decreases as the solution converges. Therefore, the probability of DE greatly increases.

Since the F_1 function is a unimodal function, a random search is not required at the latter of the search, and both DIE and random search are sharply reduced as compared to other functions. On the other hand, since the F_3 and F_4 functions require a certain amount of random search, the probability of random search decreases less than other functions, and the probability of DIE temporarily increases.

The F_8 function is different from other functions, and it can be confirmed that the probability of DIE and DE is close even after the 50th generation, and a little randomness is still required. As can be seen from Fig. 8, it is considered that this function has not converged yet, and thereafter, the DE increases and the DIE decreases similarly to other functions. As described above, it was confirmed that DEAR could adaptively adjust responsibility of random search for each function.

For the optimization problem of high dimension (Large-scale optimization), the result was not so different from the existing techniques. We would like to examine and add randomness to the technique studied recently [19], [20].

V. Conclusions

We have proposed a new method, DEAR, which adaptively uses DE, DIE (DE with random search within the confidence interval), and random search from all areas in the latter half of the search. Through numerical experiments, we confirmed that DEAR finds a better solution than others. In particular, the proposed algorithm is considered to be sufficiently practical because it has a simpler mechanism and can obtain relatively good solutions with fewer parameters.

In terms of future researches, we intend to explore the effect in the case of wider search ranges with the same functions and study high-dimensional optimization problems. In addition, since it was confirmed that it takes time to obtain a good solution for random search, we would like to propose a method to quickly find a good solution while maintaining accuracy by bandit algorithms such as Thompson Sampling.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP18K11473.

References

[1] G. I. Diaz, A. Fokoue, G. Nannicini, H. Samulowitz “An effective algorithm for hyperparameter optimization of neural networks,” IBM Jounal of Research and Development, vol. 61, issue 4/5, 2017

[2] J. H. Holland, “Adaptation in natural and artificial systems,” Ann Arbor, The University of Michigan Press, 1975

[3] D. Whitley, “A genetic algorithm tutorial,” Statistic and Computing, pp. 65-85, 1994

[4] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” IEEE Int. Conf. on Neural Networks, vol.4, pp.1942-1948, 1995

[5] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, J. R. Pastor,“Particle Swarm Optimization for Hyper-parameter Selection in Deep Neural Networks,” GECCO ’17, pp. 481-488, 2017

[6] R. Storn, K. Price, “Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” Journal of Global Optimization, vol. 11, pp. 341–359, 1997

[7] R. Storn, K. Price “Minimizing the real functions of the ICEC’96 contest by differential evolution,” Proc. of IEEE International Conference on Evolutionary Computation, pp. 842-844, 1996

[8] S. Das, P. N. Suganthan, “Differential Evolution: A Survey of the State-of-the-Art,” IEEE Transactions on Evolutionary Computation, vol. 15, no. 1, pp. 4–31, 2011

[9] M. Shibasaka, A. Hara, T. Ichimura, T. Takahara, “Species-Based Differential Evolution with Switching Search Strategies for Multimodal Function Optimization,” Proc. of 2007 IEEE Congress on Evolutionary Computation, pp. 1183-1190, 2007

[10] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, “Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems,” IEEE Transactions on Evolutionary Computation, vol. 10, No. 6, pp. 646-657, 2006

[11] J. Zhang, A. C. Sanderson, “JADE: Adaptive Differential Evolution with Optional External Archive,” IEEE Transactions on Evolutionary Computation, vol. 13, No. 5, pp. 945-958, 2009

[12] A. Notsu, M. Sakakibara, S. Ubukata, K. Honda, “Setting of Candidate Solutions Considering Confidence Intervals in Differential Evolution,” 2018 International Conference on Fuzzy Theory and Its Applications iFuzzy, pp. 7-11, 2018

[13] A. K. Qin, P. N. Suganthan, “Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization,” IEEE Transactions on Evolutionary Computation, vol. 13, no. 2, pp. 398–417, 2009

[14] R. Mallipeddi, P. N. Suganthan, “Differential Evolution Algorithm with Ensemble of Parameters and Mutation and Crossover Strategies, Swarm, Evolutionary, and Memetic Computing,” Lecture Notes in Computer Science, vol. 6466, pp. 71-78, 2010

[15] M. Ali, M. Pant and A. Abraham, “Simplex Differential Evolution,” Acta Polytechnica Hungarica, vol. 6, no. 5, 2009

[16] R. Gamperle, S. D. Muller, Petros Koumoutsakos, “A Parameter Study for Differential Evolution,” WSEAS Int. Conf. on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, 2002

[17] X.-S. Yang, “Test Problems in Optimization,” arxiv, 1008.0549v1, 2010

[18] M. Jamil, X.-S. Yang, “A Literature Survey of Benchmark Functions for Global Optimization Problems,” arxiv, 1308.4008v1, 2013

[19] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, Z. Zhu, “A survey on cooperative co-evolutionary algorithms,” IEEE Trans. Evol. Comput. vol. 23, pp. 421–441, 2019

[20] D. M. Cabrera, “Evolutionary algorithms for large-scale global optimization: a snapshot, trends and challenges,” Prog. Artif. Intell. vol. 5, pp. 85–89, 2016