Efficient Constrained Optimization by the ε Constrained Rank-Based Differential Evolution

Tetsuyuki Takahama, *Member, IEEE*, and Setsuko Sakai, *Member, IEEE*

***Abstract*—The ε constrained method is an algorithm transformation method, which can convert algorithms for unconstrained problems to algorithms for constrained problems using the ε level comparison, which compares search points based on the pair of objective value and constraint violation of them. We have proposed the ε constrained differential evolution εDE, which is the combination of the ε constrained method and differential evolution (DE), and have shown that the εDE can run very fast and can find very high quality solutions. In this study, we propose the ε constrained rank-based DE (εRDE), which adopts a new and simple scheme of controlling algorithm parameters in DE. In the scheme, different parameter values are selected for each individual. Small scaling factor and large crossover rate are selected for good individuals to improve the efficiency of search. Large scaling factor and small crossover rate are selected for bad individuals to improve the stability of search. The goodness is given by the ranking information. The εRDE is a very efficient constrained optimization algorithm that can find high-quality solutions in very small number of function evaluations. It is shown that the εRDE can find near optimal solutions stably in about half the number of function evaluations compared with various other methods on well known nonlinear constrained problems.**

***Keywords*-constrained optimization; ε constrained method; differential evolution; parameter control**

I. Introduction

Constrained optimization problems, especially nonlinear optimization problems, where objective functions are minimized under given constraints, are very important and frequently appear in the real world. There exist many studies on solving constrained optimization problems using evolutionary algorithms (EAs) [1]–[4]. EAs basically lack a mechanism to incorporate the constraints of a given problem in the fitness value of individuals. Thus, many studies have been dedicated to handle the constraints in EAs. In most successful constraint-handling techniques, the objective function value and the sum of constraint violations, or the constraint violation, are separately handled and an optimal solution is searched with balancing the optimization of the function value and the optimization of the constraint violation.

The ε constrained differential evolution (εDE) adopted one of such techniques called the ε *constrained method* and also adopted differential evolution (DE) as an optimization engine. The εDE can solve constrained problems successfully and stably [5]–[10], including engineering design problems [11]. The ε constrained method [8] is an algorithm transformation method, which can convert algorithms for unconstrained problems to algorithms for constrained problems using the ε level comparison, which compares search points based on the pair of objective value and constraint violation of them. The method has been applied various algorithms such as PSO and GA, and proposed the εPSO [12] and the hybrid algorithm of the εPSO and εGA [13]. It has been shown that the method has general-purpose properties.

In this study, we propose the ε constrained rank-based DE (εRDE) in order to improve the efficiency and stability of the εDE. The efficiency and stability depend largely on the selection of algorithm parameters in DE. The improvement of the efficiency can be attained by strengthening the convergence of search, which can be realized by using small scaling factor and large crossover rate. The improvement of the stability can be attained by maintaining the divergence of search, which can be realized by using large scaling factor and small crossover rate. In order to improve the efficiency and stability, it needs to balance between convergence and divergence. In the rank-based DE, different parameter values are selected to each individual. When the base vector is good, small scaling factor and large crossover rate is selected and the convergence is realized. Also, when the base vector is bad, large scaling factor and small crossover rate are selected and the divergence is realized. The goodness is given by the rank of the base vector in all individuals.

Well known thirteen constrained problems mentioned in [2] are solved by the εRDE within very fewer, or about half, number of function evaluations. The effectiveness of the εRDE is shown by comparing it with various methods on the problems.

In Section II, differential evolution is explained briefly. In Section III, constrained optimization methods and adaptive control methods are reviewed. The ε constrained method is defined in Section IV. The εRDE is explained in Section V. In Section VI, experimental results on thirteen constrained problems are shown and the results of the εRDE are compared with those of other methods. Finally, conclusions are described in Section VII.

II. Differential Evolution

Differential evolution is proposed by Storn and Price [14]. DE is a stochastic direct search method using population or multiple search points. DE has been successfully applied to the optimization problems including non-linear, non-differentiable, non-convex and multi-modal functions. It has been shown that DE is fast and robust to these functions.

There are some variants of DE that have been proposed, such as DE/best/1/bin and DE/rand/1/exp. The variants are classified using the notation DE/*base*/*num*/*cross*. "*base*" indicates the method of selecting a parent that will form the base vector. For example, DE/rand selects the parent for the base vector at random from the population. DE/best selects the best individual in the population.

In DE/rand/1, for each individual $\boldsymbol{x}^i$, three individuals $\boldsymbol{x}^{p1}$, $\boldsymbol{x}^{p2}$ and $\boldsymbol{x}^{p3}$ are chosen from the population without overlapping $\boldsymbol{x}^i$ and each other. Fig. 1 shows that a new vector, or a mutant vector $\boldsymbol{x}'$ is generated by the base vector $\boldsymbol{x}^{p1}$ and the difference vector $\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}$, where F is a scaling factor.

"*num*" indicates the number of difference vectors used to perturb the base vector. "*cross*" indicates the crossover mechanism used to create a child. For example, 'bin' shows that the crossover is controlled by binomial crossover using constant crossover rate, and 'exp' shows that the crossover is controlled by a kind of two-point crossover using exponentially decreasing the crossover rate. Fig. 2 shows the binomial and exponential crossover. A new child $\boldsymbol{x}^{\text{child}}$ is generated from the parent $\boldsymbol{x}^i$ and the mutant vector $\boldsymbol{x}'$, where CR is a crossover rate.

mutation DE/rand/1
```
p1=randint(1,N) s.t. p1 ≠ i;
p2=randint(1,N) s.t. p2 ∉ {i,p1};
p3=randint(1,N) s.t. p3 ∉ {i,p1,p2};
x'=x^{p1}+F(x^{p2} - x^{p3})
```

Fig. 1. Mutation operation, where randint(1,n) generates an integer randomly from $[1, n]$.

binomial crossover DE/·/·/bin
```
j_rand=randint(1,n);
for(k=1; k ≤ n; k++) {
  if(k == j_rand || u(0,1) < CR) x_k^child=x'_k;
  else x_k^child=x_k^i;
}
```
exponential crossover DE/·/·/exp
```
k=1; j=randint(1,n);
do {
    x_j^child=x'_j;
    k=k+1; j=(j+1)%n;
} while(k ≤ n && u(0,1) < CR);
while(k ≤ n) {
    x_j^child=x_j^i;
    k=k+1; j=(j+1)%n;
}
```

Fig. 2. Crossover operations

In this study, DE/rand/1/exp, where the number of difference vector is 1 or $num = 1$, is used.

III. Constrained Optimization and Previous works

A. Constrained Optimization Problems

In this study, the following optimization problem (P) with inequality constraints, equality constraints, upper bound constraints and lower bound constraints will be discussed.

$$\begin{array}{lll} \text{(P)} & \text{minimize} & f(\boldsymbol{x}) \\ & \text{subject to} & g_j(\boldsymbol{x}) \le 0,\ j = 1, \ldots, q \\ & & h_j(\boldsymbol{x}) = 0,\ j = q+1, \ldots, m \\ & & l_i \le x_i \le u_i,\ i = 1, \ldots, n, \end{array} \qquad (1)$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ is an n dimensional vector, $f(\boldsymbol{x})$ is an objective function, $g_j(\boldsymbol{x}) \le 0$ and $h_j(\boldsymbol{x}) = 0$ are q inequality constraints and $m - q$ equality constraints, respectively. Functions f, g_j and h_j are linear or nonlinear real-valued functions. Values u_i and l_i are the upper bound and the lower bound of x^i, respectively. Also, let the feasible space in which every point satisfies all constraints be denoted by $\mathcal{F}$ and the search space in which every point satisfies the upper and lower bound constraints be denoted by $\mathcal{S}\,(\supset \mathcal{F})$.

B. Constrained optimization methods

EAs for constrained optimization can be classified into several categories according to the way the constraints are treated as follows [3]:

(1) Constraints are only used to see whether a search point is feasible or not. Approaches in this category are usually called death penalty methods. In this category, generating initial feasible points is difficult and computationally demanding when the feasible region is very small.

(2) The constraint violation, which is the sum of the violation of all constraint functions, is combined with the objective function. The penalty function method is in this category [15]–[18]. The main difficulty of the method is the selection of an appropriate value for the penalty coefficient that adjusts the strength of the penalty. In order to solve the difficulty, some methods, where a kind of the penalty coefficient is adaptively controlled [19], [20], are proposed.

(3) The constraint violation and the objective function are used separately. In this category, both the constraint violation and the objective function are optimized by a lexicographic order in which the constraint violation precedes the objective function. Deb [21] proposed a method that adopts the extended objective function, which realizes the lexicographic ordering. Takahama and Sakai proposed the α constrained method [22], and ε constrained method [12] that adopt a lexicographic ordering with relaxation of the constraints. Runarsson and Yao [23] proposed the stochastic ranking method that adopts the stochastic lexicographic order, which ignores the constraint violation with some probability. Mezura-Montes and Coello [24] proposed a comparison mechanism that is equivalent to the lexicographic ordering. Venkatraman and Yen [25] proposed a two-step optimization method, which first optimizes constraint violation and then objective function. These methods were successfully applied to various problems.

(4) The constraints and the objective function are optimized by multiobjective optimization methods. In this category, the constrained optimization problems are solved as the multiobjective optimization problems in which the objective function and the constraint functions are objectives to be optimized [26]–[32]. But in many cases, solving multiobjective optimization

problems is a more difficult and expensive task than solving single objective optimization problems.

(5) Hybridization methods. In this category, constrained problems are solved by combining some of above mentioned methods. Mallipeddi and Suganthan [33] proposed a hybridization of the methods in category (2), (3) and (4).

C. Parameter control methods in DE

The performance of DE is affected by control parameters such as the scaling factor F, the crossover rate CR and the population size N, and is also affected by the type of operations such as mutation strategies and crossover operations. Many researchers have been studying on controlling the parameters and the operations. The methods of the control can be classified into two categories: observation-based and success-based control.

(1) observation-based control: The current search state is observed, proper parameter values are inferred according to the observation, and parameters and/or strategies are dynamically controlled. FADE(Fuzzy Adaptive DE) [34] observes the movement of search points and the change of function values between successive generations, and controls F and CR. DESFC(DE with Speciation and Fuzzy Clustering) [35] adopts fuzzy clustering, observes partition entropy of search points, and controls CR and the mutation strategies between the rand and the species-best strategy.

(2) success-based control: It is recognized as a success case when a better search point than the parent is generated. The parameters and/or strategies are adjusted so that the values in the success cases are frequently used. It is thought that the self-adaptation, where parameters are contained in individuals and are evolved by applying evolutionary operators to the parameters, is included in this category. DESAP(Differential Evolution with Self-Adapting Populations) [36] controls F, CR and N self-adaptively. SaDE(Self-adaptive DE) [37] controls the mean value of CR according to the mean value in success cases and controls the selection probability of the mutation strategies according to the success rates. jDE(self-adaptive DE algorithm) [38] controls F and CR self-adaptively. JADE(adaptive DE with optional external archive) [39] controls the mean values of F and CR according to the mean values in success cases.

In the category (1), it is difficult to select proper type of observation which is independent of the optimization problem and its scale. In the category (2), when a new good search point is found near the parent, parameters are adjusted to the direction of convergence. In problems with ridge landscape or multimodal landscape, where good search points exist in small region, parameters are tuned for small success and big success will be missed. Thus, search process would be trapped at a local optimal solution.

In this study, we propose new observation-based control, which in the category (1) and is independent of the optimization problem and its scale. In the control, F and CR are tuned according to the ranking information of the base vector.

IV. The ε constrained method

A. Constraint violation and ε level comparisons

In the ε constrained method, constraint violation $\phi(\boldsymbol{x})$ is defined. The constraint violation can be given by the maximum of all constraints or the sum of all constraints.

$$\phi(\boldsymbol{x}) = \max\{\max_j\{0, g_j(\boldsymbol{x})\}, \max_j |h_j(\boldsymbol{x})|\} \quad (2)$$

$$\phi(\boldsymbol{x}) = \sum_j ||max\{0, g_j(\boldsymbol{x})\}||^p + \sum_j ||h_j(\boldsymbol{x})||^p \quad (3)$$

where p is a positive number.

The ε *level comparison* is defined as an order relation on a pair of objective function value and constraint violation $(f(\boldsymbol{x}), \phi(\boldsymbol{x}))$. If the constraint violation of a point is greater than 0, the point is not feasible and its worth is low. The ε level comparisons are defined basically as a lexicographic order in which $\phi(\boldsymbol{x})$ precedes $f(\boldsymbol{x})$, because the feasibility of $\boldsymbol{x}$ is more important than the minimization of $f(\boldsymbol{x})$. This precedence can be adjusted by the parameter ε.

Let $f_1\,(f_2)$ and $\phi_1\,(\phi_2)$ be the function values and the constraint violation at a point $\boldsymbol{x}_1\,(\boldsymbol{x}_2)$, respectively. Then, for any ε satisfying $\varepsilon \geq 0$, ε level comparisons $<_\varepsilon$ and $\leq_\varepsilon$ between (f_1, ϕ_1) and (f_2, ϕ_2) are defined as follows:

$$(f_1, \phi_1) <_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 < f_2, & \text{if } \phi_1, \phi_2 \leq \varepsilon \\ f_1 < f_2, & \text{if } \phi_1 = \phi_2 \\ \phi_1 < \phi_2, & \text{otherwise} \end{cases} \quad (4)$$

$$(f_1, \phi_1) \leq_\varepsilon (f_2, \phi_2) \Leftrightarrow \begin{cases} f_1 \leq f_2, & \text{if } \phi_1, \phi_2 \leq \varepsilon \\ f_1 \leq f_2, & \text{if } \phi_1 = \phi_2 \\ \phi_1 < \phi_2, & \text{otherwise} \end{cases} \quad (5)$$

In case of ε=∞, the ε level comparisons $<_\infty$ and $\leq_\infty$ are equivalent to the ordinary comparisons $<$ and $\leq$ between function values. Also, in case of $\varepsilon = 0$, $<_0$ and $\leq_0$ are equivalent to the lexicographic orders in which the constraint violation $\phi(\boldsymbol{x})$ precedes the function value $f(\boldsymbol{x})$.

B. The properties of the ε constrained method

The ε constrained method converts a constrained optimization problem into an unconstrained one by replacing the order relation in direct search methods with the ε level comparison. An optimization problem solved by the ε constrained method, that is, a problem in which the ordinary comparison is replaced with the ε level comparison, $(\mathrm{P}_{\leq_\varepsilon})$, is defined as follows:

$$(\mathrm{P}_{\leq_\varepsilon}) \quad \text{minimize}_{\leq_\varepsilon} \quad f(\boldsymbol{x}), \quad (6)$$

where minimize$_{\leq_\varepsilon}$ denotes the minimization based on the ε level comparison $\leq_\varepsilon$. Also, a problem (P^ε) is defined such that the constraints of (P), that is, $\phi(\boldsymbol{x}) = 0$, is relaxed and replaced with $\phi(\boldsymbol{x}) \leq \varepsilon$:

$$\begin{array}{lll} (\mathrm{P}^\varepsilon) & \text{minimize} & f(\boldsymbol{x}) \\ & \text{subject to} & \phi(\boldsymbol{x}) \leq \varepsilon \end{array} \quad (7)$$

It is obvious that (P^0) is equivalent to (P).

For the three types of problems, (P^ε), $(\mathrm{P}_{\leq_\varepsilon})$ and (P), the following theorems are given based on the ε constrained method [12].

Theorem 1: If an optimal solution (P^0) exists, any optimal solution of ($\mathrm{P}_{\leq_\varepsilon}$) is an optimal solution of (P^ε).

Theorem 2: If an optimal solution of (P) exists, any optimal solution of ($\mathrm{P}_{\leq_0}$) is an optimal solution of (P).

Theorem 3: Let $\{\varepsilon_n\}$ be a strictly decreasing non-negative sequence and converge to 0. Let $f(\boldsymbol{x})$ and $\phi(\boldsymbol{x})$ be continuous functions of $\boldsymbol{x}$. Assume that an optimal solution $\boldsymbol{x}^*$ of (P^0) exists and an optimal solution $\hat{\boldsymbol{x}}_n$ of ($\mathrm{P}_{\leq_{\varepsilon_n}}$) exists for any ε_n. Then, any accumulation point to the sequence $\{\hat{\boldsymbol{x}}_n\}$ is an optimal solution of (P^0).

Theorem 1 and 2 show that a constrained optimization problem can be transformed into an equivalent unconstrained optimization problem by using the ε level comparison. So, if the ε level comparison is incorporated into an existing unconstrained optimization method, constrained optimization problems can be solved. Theorem 3 shows that, in the ε constrained method, an optimal solution of (P^0) can be given by converging ε to 0 as well as by increasing the penalty coefficient to infinity in the penalty method.

V. The εRDE

In this section, the rank-based DE (RDE) are described first and then the εRDE is defined.

A. Rank-based Differential Evolution (RDE)

It is very important to balance between the convergence and the divergence of search in order to improve the search efficiency.

- Convergence will be improved by generating new points near good search points, which can be realized by adopting the best strategy or selecting small scaling factor. However, search points will often be trapped at a local solution.
- Divergence will be improved by generating new points in wide range, which is realized by adopting the rand strategy or selecting large scaling factor. However, the search efficiency will be lowered.

In this study, we propose to balance between the convergence and the divergence using ranking information:

- Convergence: If the base vector is good, the mutant vector is generated near the base vector by adopting small scaling factor. Also, the trial vector is generated near the mutant vector by adopting large crossover rate.
- Divergence: If the base vector is bad, the mutant vector is generated in wide range by adopting large scaling factor. Also, the trial vector is generated near the target vector and far from the mutant vector by adopting small crossover rate.

In order to achieve the idea above, we propose to select different values of F and CR for each individual according to the rank of the base vector, where the rank of the best individual is 1. Let a parent be denoted by $\boldsymbol{x}_i$, the base vector be denoted by $\boldsymbol{x}_b$ and the rank of the base vector be denoted by R_b. The scaling factor F_i and the crossover rate CR_i for $\boldsymbol{x}_i$ can be defined by the following equations:

$$F_i = F_{\min} + (F_{\max} - F_{\min})\frac{R_b - 1}{N - 1} \quad (8)$$

$$CR_i = CR_{\max} - (CR_{\max} - CR_{\min})\frac{R_b - 1}{N - 1} \quad (9)$$

where $F_{\min}$ and $F_{\max}$ are parameters to specify the minimum and maximum value of F, and $CR_{\min}$ and $CR_{\max}$ are parameters to specify the minimum and maximum of CR. If the base vector is the best individual, F becomes the minimum value and CR becomes the maximum value. If the base vector is the worst individual, F becomes the maximum value and CR becomes the minimum value. Thus, the idea is realized.

B. The algorithm of the εRDE

The εRDE is the DE that adopts an observation-based control of algorithm parameters, a simple modification in offspring generation, and the ε constrained method.

The algorithm of the εRDE is as follows:

0. Parameter setup. The range of scaling factor [$F_{\min}$, $F_{\max}$] and the range of crossover rate [$CR_{\min}$, $CR_{\max}$] are given.
1. Initialization of the individuals. Initial N individuals $\{\boldsymbol{x}^i, i = 1, 2, \cdots, N\}$ are generated randomly in search space $\mathcal{S}$ and form an initial population.
2. Initialization of the ε level. An initial ε level is given by the ε level control function $\varepsilon(0)$.
3. Termination condition. If the number of function evaluations exceeds the maximum number of evaluations $FE_{\max}$, the algorithm is terminated.
4. Ranking all individuals. The ranks $\{R_i\}$ of the individuals $\{\boldsymbol{x}^i\}$ are given according to the ε level comparison.
5. DE operation. Each individual $\boldsymbol{x}^i$ is selected as a parent. A new child $\boldsymbol{x}^{\mathrm{new}}$ is generated by DE/rand/1/exp operation with a scaling factor F_i and a crossover rate CR_i that are given by Eqs. (8) and (9), respectively.
6. Survivor selection. If the new one is better than or equal to the parent based on the ε level comparison, the parent $\boldsymbol{x}^i$ is immediately replaced by the trial vector $\boldsymbol{x}^{\mathrm{new}}$ because not discrete generation model but continuous generation model is adopted. Until all individuals are selected, go back to 5 in order to select the next individual as a parent.
7. Control of the ε level. The ε level is updated by the ε level control function $\varepsilon(t)$.
8. Go back to 3.

C. Controlling the ε level

The ε level is controlled according to Eqs. (10) and (11). The initial ε level $\varepsilon(0)$ is the constraint violation of the top θ-th individual in the initial search points. The ε level is updated until the number of iterations t becomes the control generation T_c. After the number of iterations exceeds T_c, the ε level is

```
εRDE/rand/1/exp()
{
// Initialize the individuals
  P=N individuals {x_i} generated randomly in S;
// Initialize the ε level
  ε=ε(0);
  for(t=1; termination condition is false; t++) {
    {R_i}=Ranks of {x_i} according to <_ε;
    for(i=1; i≤N; i++) {
      x_{r1}=Randomly selected from P(r1≠i);
      x_{r2}=Randomly selected from P(r2∉{i,r1});
      x_{r3}=Randomly selected from P(r3∉{i,r1,r2});
      F_i=F_min+(F_max-F_min)(R_{r1}-1)/(N-1);
      CR_i=CR_max-(CR_max-CR_min)(R_{r1}-1)/(N-1);
      x^new=x_i;
      j=select randomly from [1,n];
      k=1;
      do {
        x_j^new=x_{r1,j}+F_i(x_{r2,j}-x_{r3,j});
        j=(j+1)%n;
        k++;
      } while(k≤n && u(0,1)<CR_i);
      if((f(x^new),φ(x^new)) <_ε (f(x^i),φ(x^i)))
        x_i=x^new;
    }
// Control the ε level
    ε=ε(t);
  }
}
```

Fig. 3. The algorithm of the ε constrained rank-based differential evolution, where $\varepsilon(t)$ is the ε level control function, FE is the number of function evaluations, and $u(l,r)$ is a uniform random number generator in $[l,r]$.

set to 0 to obtain solutions with minimum constraint violation.

$$\varepsilon(0) = \phi(\boldsymbol{x}_\theta) \tag{10}$$

$$\varepsilon(t) = \begin{cases} \varepsilon(0)(1-\frac{t}{T_c})^{cp}, & 0<t<T_c, \\ 0, & t \geq T_c \end{cases} \tag{11}$$

where $\boldsymbol{x}_\theta$ is the top θ-th individual and $\theta = 0.2N$. This control is effective to solve problems with equality constraints.

Fig. 3 shows the algorithm of the εRDE.

VI. Solving nonlinear optimization problems

In this paper, thirteen benchmark problems that are mentioned in some studies [3], [23], [24] are optimized, and the results by the εRDE are compared with those results.

A. Test problems and the experimental conditions

In the thirteen benchmark problems, problems g03, g05, g11 and g13 contain equality constraints. In problems with equality constraints, the equality constraints are relaxed and converted to inequality constraints according to Eq. (12), which is adopted in many methods:

$$|h_j(\boldsymbol{x})| \leq \delta,\ \delta > 0, \tag{12}$$

where $\delta = 10^{-4}$. Problem g12 has disjointed feasible regions. Table I shows the outline of the thirteen problems [24], [40]. The table contains the number of variables n, the form of the objective function, the number of linear inequality constraints (LI), nonlinear inequality constraints (NI), linear equality constraints (LE), nonlinear equality constraints (NE) and the number of constraints active at the optimal solution.

TABLE I
SUMMARY OF TEST PROBLEMS

f	n	Form of f	LI	NI	LE	NE	active
g01	13	quadratic	9	0	0	0	6
g02	20	nonlinear	1	1	0	0	1
g03	10	polynomial	0	0	0	1	1
g04	5	quadratic	0	6	0	0	2
g05	4	cubic	2	0	0	3	3
g06	2	cubic	0	2	0	0	2
g07	10	quadratic	3	5	0	0	6
g08	2	nonlinear	0	2	0	0	0
g09	7	polynomial	0	4	0	0	2
g10	8	linear	3	3	0	0	6
g11	2	quadratic	0	0	0	1	1
g12	3	quadratic	0	9^3	0	0	0
g13	5	nonlinear	0	0	1	2	3

The parameters for the ε constrained method are as follows: Every constraint violation is defined as a simple sum of constraints, or $p = 1$ in Eq. (3). The ε level is controlled using Eqs. (10) and (11) with $cp = 5$, $T_c = 1000$ for problems with equality constraints and is 0 for other problems. The parameters for RDE are: The number of search points $N = 40$, the maximum number of evaluations $FE_{\max} = 100,000$, the scaling factor $F_{\min} = 0.6$ and $F_{\max} = 0.95$, the crossover rate $CR_{\min} = 0.85$ and $CR_{\max} = 0.95$. In this paper, 30 independent runs are performed.

B. Experimental results

Table II summarizes the experimental results. The table shows the known "optimal" solution for each problem and the statistics from the 30 independent runs. These include the best, median, mean, and worst values and the standard deviation of the objective values found. Also, the average number of evaluations of the objective function and the constraints to find the best solution in each run is shown in the columns labeled #func and #const respectively. The last column shows how many evaluations of objective function can be omitted.

For problems g01, g04, g05, g06, g08, g09, g11, g12 and g13, the optimal solutions are found consistently in all 30 runs. For all other problems g02, g03, g07 and g10, the optimal or near-optimal solutions are found in all 30 runs. These results show that the εRDE is a very efficient and stable algorithm. As for the problem g02, the problem is a multi-modal problem that has many local optima with peaks near the global optimum within the feasible region. Many other methods cannot constantly obtain high quality solutions, but the εRDE found near-optimal solutions under -0.8036 constantly within 100,000 FEs. Thus, it is thought that the εRDE has a high ability to solve multi-modal problems.

The εRDE is a very fast algorithm. The average execution times ranged from 0.05 seconds to 0.3 seconds using a notebook PC with 2.0GHz Core i7. The execution times are less than 1/8 seconds in all problems, except for g12.

TABLE II

EXPERIMENTAL RESULTS ON 13 BENCHMARK PROBLEMS USING STANDARD SETTINGS; 30 INDEPENDENT RUNS WERE PERFORMED

f	optimal	best	median	mean	worst	st. dev.	#func	#const	omit(%)
g01	-15.000	-15.000000	-15.000000	-15.000000	-15.000000	0.000e+00	35799.8	56508.2	36.7
g02	-0.803619	-0.803618	-0.803615	-0.803614	-0.803605	3.027e-06	55092.2	99741.8	44.8
g03	-1.000	-1.000500	-1.000500	-1.000500	-1.000498	4.372e-07	44910.8	99871.9	55.0
g04	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672	0.0000e+00	28003.3	51613.7	45.7
g05	5126.498	5126.496714	5126.496714	5126.496714	5126.496714	0.000e+00	21801.7	66033.2	67.0
g06	-6961.814	-6961.813876	-6961.813876	-6961.813876	-6961.813876	2.803e-12	5482.9	10152.5	46.0
g07	24.306	24.306209	24.306210	24.306210	24.306215	1.406e-06	29535.8	99829.8	70.4
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0.000e+00	3606.2	4063.4	11.3
g09	680.630	680.630057	680.630057	680.630057	680.630057	0.000e+00	19089.7	42266.1	54.8
g10	7049.248	7049.248021	7049.248021	7049.248021	7049.248022	2.125e-07	17552.8	99820.2	82.4
g11	0.750	0.749900	0.749900	0.749900	0.749900	0.000e+00	26255.5	35536.4	26.1
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	0.000e+00	4012.1	7872.7	49.0
g13	0.053950	0.053942	0.053942	0.053942	0.053942	0.000e+00	23717.9	68253.4	65.3

In the ε constrained method, the objective function and the constraints are treated separately. So, when the order relation of the search points can be decided only by the constraint violation of the constraints, the objective function is not evaluated, or the evaluation of the objective function can often be omitted. Thus, the number of evaluations of the objective function is less than the number of evaluations of the constraints. This nature of the εRDE contributes to the efficiency of the algorithm especially when the objective function is computationally demanding. The number of evaluations of the constraint violations to find the best solution ranged from about 4,000 to 100,000. The number of evaluations of the objective function ranged between about 3,600 and 55,000. For these problems, the εRDE can omit the evaluation of the objective function about 10% to 80%. Therefore, the εRDE can find optimal solutions very efficiently, especially from the viewpoint of the number of evaluations for the objective function.

These results show that the εRDE is a very efficient and stable algorithm.

C. Comparison with other methods

There are some methods that solved the same thirteen problems. In the methods, for comparative studies we chose the simple multimembered evolution strategy (SMES) proposed by Mezura-Montes and Coello [24], the adaptive trade-off model (ATMES) proposed by Wang *et al.* [20], multiobjective method (HCOEA) proposed by Wang *et al.* [32], ECTHT-EP2 proposed by Mallipeddi and Suganthan [33], and the εDE proposed by Takahama and Sakai [8], because the results of these methods are better than the results of the other methods, and they reported good quality statistical information. Also, A-DDE proposed by Mezura-Montes and Palomeque-Ortiz [41], which adopts adaptive parameter control, is included in the comparison.

Table III shows the comparisons of the best, median, average, worst values and the standard deviation for the seven methods. The maximum number of FEs is also shown in "$FE_{\max}$".

All methods found optimal solutions in all 30 runs for g01, g03, g04, g08, g11 and g12. In other problems, from the viewpoint of quality of solutions, it is thought that the εRDE and the εDE are the best methods followed by ECHT-EP2, because the εRDE and the εDE found high-quality near optimal solutions stably in all problems. As for the εRDE, in problem g02, the εRDE found better solutions on average than the εDE and ECHT-EP2. As for the εDE, in problem g07, the εDE found better solutions on average than the εRDE and ECHT-EP2. As for the εRDE and the εDE, in problems g02, g07 and g10, both methods found better solutions on average than ECHT-EP2. Also, the number of FEs in the εRDE is much less than that in the εDE and ECHT-EP2. Thus, it is thought that the εRDE is better than the εDE and ECHT-EP2 from the viewpoint of the efficiency.

VII. CONCLUSIONS

Differential evolution is known as a simple, efficient and robust search algorithm that can solve unconstrained optimization problems. In this study, we proposed a new and simple scheme of controlling parameters in order to improve the efficiency and stability using ranking information, and proposed the εRDE. We showed that the εRDE could solve thirteen benchmark problems most efficiently and stably compared with many other methods.

In the future, we will apply the εRDE to various real world problems that have large numbers of decision variables and constraints.

Acknowledgments

This research is supported in part by Grant-in-Aid for Scientific Research (C) (No. 20500138,22510166) of Japan society for the promotion of science and Hiroshima City University Grant for Special Academic Research (General Studies).

REFERENCES

[1] Z. Michalewicz, "A survey of constraint handling techniques in evolutionary computation methods," in *Proc. of the 4th Annual Conference on Evolutionary Programming*. Cambridge, Massachusetts: The MIT Press, 1995, pp. 135–155.

[2] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, Jan. 2002.

TABLE III
COMPARISON OF STATISTICAL RESULTS AMONG THE εRDE, THE εDE [8], SMES [24], ATMES [20], HCOEA [32], ECHT-EP2 [33] AND A-DDE [41].

f & optimal	Statistics	εRDE	εDE	SMES	ATMES	HCOEA	ECHT-EP2	A-DDE
	$FE_{\max}$	**100,000**	200,000	240,000	240,000	240,000	240,000	180,000
g01 -15.000	best	-15.000000	-15.000000	-15.000	-15.000	-15.000000	-15.0000	-15.000
	median	-15.000000	-15.000000	-15.000	-15.000	-15.000000	-15.0000	-15.000
	mean	-15.000000	-15.000000	-15.000	-15.000	-15.000000	-15.0000	-15.000
	worst	-15.000000	-15.000000	-15.000	-15.000	-14.999998	-15.0000	-15.000
	σ	0.00e+00	0.00e+00	0.00e+00	1.6e-14	4.297e-07	0.00e+00	7.00e-06
g02 -0.803619	best	-0.803618	-0.803618	-0.803601	-0.803388	-0.803241	-0.8036191	-0.803605
	median	-0.803615	-0.803614	-0.792549	-0.792420	-0.802556	-0.8033239	-0.777368
	mean	**-0.803614**	-0.803613	-0.785238	-0.790148	-0.801258	-0.7998220	-0.771090
	worst	-0.803605	-0.803588	-0.751322	-0.756986	-0.792363	-0.7851820	-0.609853
	σ	**3.03-06**	5.59e-06	1.67e-02	1.3e-02	3.832e-03	6.29e-03	3.66e-02
g03 -1.0005	best	-1.000500	-1.000500	-1.000	-1.000	-1.000000	-1.0005	-1.000
	median	-1.000500	-1.000500	-1.000	-1.000	-1.000000	-1.0005	-1.000
	mean	-1.000500	-1.000500	-1.000	-1.000	-1.000000	-1.0005	-1.000
	worst	-1.000498	-1.000500	-1.000	-1.000	-1.000000	-1.0005	-1.000
	σ	4.37e-07	6.457e-09	2.09e-04	5.9e-05	1.304e-12	**0.0e+00**	9.30e-12
g04 -30665.5387	best	-30665.538672	-30665.538670	-30665.539	-30665.539	-30665.539	-30665.5387	-30665.539
	median	-30665.538672	-30665.538670	-30665.539	-30665.539	-30665.539	-30665.5387	-30665.539
	mean	-30665.538672	-30665.538670	-30665.539	-30665.539	-30665.539	-30665.5387	-30665.539
	worst	-30665.538672	-30665.538670	-30665.539	-30665.539	-30665.539	-30665.5387	-30665.539
	σ	0.00e+00	0.00e+00	0.00e+00	7.4e-12	5.404e-07	0.0e+00	3.20e-13
g05 5126.4967	best	5126.496714	5126.496714	5126.599	5126.498	5126.4981	5126.4967	5126.497
	median	5126.496714	5126.496714	5160.198	5126.776	5126.4981	5126.4967	5126.497
	mean	5126.496714	5126.496714	5174.492	5127.648	5126.4981	5126.4967	5126.497
	worst	5126.496714	5126.496714	5304.167	5135.256	5126.4984	5126.4967	5126.497
	σ	**0.00e+00**	1.82e-12	5.006e+01	1.8e+00	1.727e-07	**0.0e+00**	2.10e-11
g06 -6961.8139	best	-6961.813876	-6961.813876	-6961.814	-6961.814	-6961.81388	-6961.8139	-6961.814
	median	-6961.813876	-6961.813876	-6961.814	-6961.814	-6961.81388	-6961.8139	-6961.814
	mean	-6961.813876	-6961.813876	-6961.284	-6961.814	-6961.81388	-6961.8139	-6961.814
	worst	-6961.813876	-6961.813876	-6952.482	-6961.814	-6961.81388	-6961.8139	-6961.814
	σ	2.80e-12	**0.00e+00**	1.85e+00	4.6e-12	8.507e-12	**0.00e+00**	2.11e-12
g07 24.3062	best	24.306209	24.306209	24.327	24.306	24.3064582	24.3062	24.306
	median	24.306210	24.306209	24.426	24.313	24.3073055	24.3063	24.306
	mean	24.306210	**24.306209**	24.475	24.316	24.3073989	24.3063	24.306
	worst	24.306215	24.306209	24.843	24.359	24.3092401	24.3063	24.306
	σ	1.41e-06	**4.27e-09**	1.32e-01	1.1e-02	7.118e-04	3.19e-05	4.20e-05
g08 -0.095825	best	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504	-0.095825
	median	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504	-0.095825
	mean	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504	-0.095825
	worst	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.09582504	-0.095825
	σ	0.00e+00	0.00e+00	0.00e+00	2.8e-17	2.417e-17	0.0e+00	9.10e-10
g09 680.630057	best	680.630057	680.630057	680.632	680.630	680.6300574	680.630057	680.63
	median	680.630057	680.630057	680.642	680.633	680.6300574	680.630057	680.63
	mean	680.630057	680.630057	680.643	680.639	680.6300574	680.630057	680.63
	worst	680.630057	680.630057	680.719	680.673	680.6300578	680.630057	680.63
	σ	**0.00e+00**	**0.00e+00**	1.55e-02	1.0e-02	9.411e-08	2.61e-08	1.15e-10
g10 7049.248	best	7049.248021	7049.248021	7051.903	7052.253	7049.286598	7049.2483	7049.248
	median	7049.248021	7049.248021	7253.603	7215.357	7049.486145	7049.2488	7049.248
	mean	**7049.248021**	**7049.248021**	7253.047	7250.437	7049.525438	7049.2490	7049.248
	worst	7049.248022	7049.248021	7638.366	7560.224	7049.984208	7049.2501	7049.248
	σ	2.15e-07	**0.00e+00**	1.36e+02	1.2e+02	1.502e-01	6.60e-04	3.23e-4
g11 0.749900	best	0.749900	0.749900	0.75	0.75	0.750000	0.7499	0.75
	median	0.749900	0.749900	0.75	0.75	0.750000	0.7499	0.75
	mean	0.749900	0.749900	0.75	0.75	0.750000	0.7499	0.75
	worst	0.749900	0.749900	0.75	0.75	0.750000	0.7499	0.75
	σ	**0.00e+00**	**0.00e+00**	1.52e-04	3.4e-04	1.546e-12	**0.0e+00**	5.35e-15
g12 -1.000	best	-1.000000	-1.000000	-1.0000	-1.000	-1.000000	-1.0000	-1.000
	median	-1.000000	-1.000000	-1.0000	-1.000	-1.000000	-1.0000	-1.000
	mean	-1.000000	-1.000000	-1.0000	-1.000	-1.000000	-1.0000	-1.000
	worst	-1.000000	-1.000000	-1.0000	-0.994	-1.000000	-1.0000	-1.000
	σ	0.00e+00	0.00e+00	0.00e+00	1.0e-03	0.000e+00	0.0e+00	4.10e-11
g13 0.0539415	best	0.0539415	0.053942	0.053986	0.053950	0.0539498	0.053941514	0.053942
	median	0.0539415	0.053942	0.061873	0.053952	0.0539498	0.053941514	0.053942
	mean	**0.0539415**	**0.053942**	0.166385	0.053959	0.0539498	**0.053941514**	0.079627
	worst	0.0539415	0.053942	0.468294	0.053999	0.0539499	0.053941514	0.438803
	σ	**0.00e+00**	**0.00e+00**	1.77e-01	1.3e-05	8.678e-08	1.00e-12	9.60e-02

[3] T. Takahama and S. Sakai, "Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations," *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 5, pp. 437–451, Oct. 2005.

[4] E. Mezura-Montes and C. A. C. Coello, "Constraint-handling in nature-inspired numerical optimization: Past, present and future," *Swarm and Evolutionary Computation*, vol. 1, pp. 173–194, 2011.

[5] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *Proc. of the 2006 IEEE Congress on Evolutionary Computation*, Jul. 2006, pp. 308–315.

[6] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with dynamic ε-level control," in *Advances in Differential Evolution*, U. Chakraborty, Ed. Springer-Verlag, 2008, pp. 139–154.

[7] T. Takahama and S. Sakai, "Solving difficult constrained optimization problems by the ε constrained differential evolution with gradient-based mutation," in *Constraint-Handling in Evolutionary Optimization*, E. Mezura-Montes, Ed. Springer-Verlag, 2009, pp. 51–72.

[8] T. Takahama and S. Sakai, "Fast and stable constrained optimization by the ε constrained differential evolution," *Pacific Journal of Optimization*, vol. 5, no. 2, pp. 261–282, May 2009.

[9] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation," in *Proc. of the 2010 IEEE Congress on Evolutionary Computation*, Jul. 2010, pp. 1680–1688.

[10] T. Takahama and S. Sakai, "Efficient constrained optimization by the ε constrained adaptive differential evolution," in *Proc. of the 2010 IEEE Congress on Evolutionary Computation*, Jul. 2010, pp. 2052–2059.

[11] T. Takahama, S. Sakai, and N. Iwane, "Solving nonlinear constrained optimization problems by the ε constrained differential evolution," in *Proc. of the 2006 IEEE Conference on Systems, Man, and Cybernetics*, Oct. 2006, pp. 2322–2327.

[12] T. Takahama and S. Sakai, "Constrained optimization by ε constrained particle swarm optimizer with ε-level control," in *Proc. of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology (WSTST'05)*, May 2005, pp. 1019–1029. [Online]. Available: http://www.ints.info.hiroshima-cu.ac.jp/˜takahama/eng/papers/ePSO05c.pdf

[13] T. Takahama, S. Sakai, and N. Iwane, "Constrained optimization by the ε constrained hybrid algorithm of particle swarm optimization and genetic algorithm," in *Proc. of the 18th Australian Joint Conference on Artificial Intelligence*, Dec. 2005, pp. 389–400, Lecture Notes in Computer Science 3809.

[14] R. Storn and 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.

[15] A. Homaifar, S. H. Y. Lai, and X. Qi, "Constrained optimization via genetic algorithms," *Simulation*, vol. 62, no. 4, pp. 242–254, 1994.

[16] J. Joines and C. Houck, "On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs," in *Proc. of the first IEEE Conference on Evolutionary Computation*, D. Fogel, Ed. Orlando, Florida: IEEE Press, 1994, pp. 579–584.

[17] Z. Michalewicz and N. Attia, "Evolutionary optimization of constrained problems," in *Proc. of the 3rd Annual Conference on Evolutionary Programming*, A. Sebald and L. Fogel, Eds. River Edge, NJ: World Scientific Publishing, 1994, pp. 98–108.

[18] C. A. C. Coello, "Use of a self-adaptive penalty approach for engineering optimization problems," *Computers in Industry*, vol. 41, no. 2, pp. 113–127, 2000.

[19] B. Tessema and G. Yen, "A self adaptive penalty function based algorithm for constrained optimization," in *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, G. G. Yen, S. M. Lucas, G. Fogel, G. Kendall, R. Salomon, B.-T. Zhang, C. A. C. Coello, and T. P. Runarsson, Eds. Vancouver, BC, Canada: IEEE Press, 16-21 July 2006, pp. 246–253.

[20] Y. Wang, Z. Cai, Y. Xhau, and W. Zeng, "An adaptive tradeoff model for constrained evolutionary computation," *IEEE Trans. on Evolutionary Computation*, vol. 12, no. 1, pp. 80–92, 2008.

[21] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2/4, pp. 311–338, 2000.

[22] T. Takahama and S. Sakai, "Tuning fuzzy control rules by the α constrained method which solves constrained nonlinear optimization problems," *Electronics and Communications in Japan, Part3: Fundamental Electronic Science*, vol. 83, no. 9, pp. 1–12, Sep. 2000.

[23] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

[24] E. Mezura-Montes and C. A. C. Coello, "A simple multimembered evolution strategy to solve constrained optimization problems," *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 1, pp. 1–17, Feb. 2005.

[25] S. Venkatraman and G. G. Yen, "A generic framework for constrained optimization using genetic algorithms," *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 4, pp. 424–435, Aug. 2005.

[26] E. Camponogara and S. N. Talukdar, "A genetic algorithm for constrained and multiobjective optimization," in *3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA)*, J. T. Alander, Ed. Vaasa, Finland: University of Vaasa, Aug. 1997, pp. 49–62.

[27] P. D. Surry and N. J. Radcliffe, "The COMOGA method: Constrained optimisation by multiobjective genetic algorithms," *Control and Cybernetics*, vol. 26, no. 3, pp. 391–412, 1997.

[28] C. A. C. Coello, "Constraint-handling using an evolutionary multiobjective optimization technique," *Civil Engineering and Environmental Systems*, vol. 17, pp. 319–346, 2000.

[29] T. Ray, K. M. Liew, and P. Saini, "An intelligent information sharing strategy within a swarm for unconstrained and constrained optimization problems," *Soft Computing – A Fusion of Foundations, Methodologies and Applications*, vol. 6, no. 1, pp. 38–44, Feb. 2002.

[30] T. P. Runarsson and X. Yao, "Evolutionary search and constraint violations," in *Proc. of the 2003 Congress on Evolutionary Computation*, vol. 2. Piscataway, New Jersey: IEEE Service Center, Dec. 2003, pp. 1414–1419.

[31] A. H. Aguirre, S. B. Rionda, C. A. C. Coello, G. L. Lizárraga, and E. M. Montes, "Handling constraints using multiobjective optimization concepts," *International Journal for Numerical Methods in Engineering*, vol. 59, no. 15, pp. 1989–2017, Apr. 2004.

[32] Y. Wang, Z. Cai, G. Cuo, and Z. Zhou, "Multiobjective optimization and hybrid evolutionary algorthm to solve constrained optimization problems," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 3, pp. 560–575, 2007.

[33] R. Mallipeddi and P. N. Suganthan, "Ensemble of constraint handling techniques," *IEEE Transactions on Evolutionary Computation*, 2010, to appear.

[34] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput.*, vol. 9, no. 6, pp. 448–462, 2005.

[35] T. Takahama and S. Sakai, "Fuzzy c-means clustering and partition entropy for species-best strategy and search mode selection in nonlinear optimization by differential evolution," in *Proc. of the 2011 IEEE International Conference on Fuzzy Systems*, Jun. 2011, pp. 290–297.

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

[37] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. of the 2005 IEEE Congress on Evolutionary Computation*, 2005, pp. 1785–1791.

[38] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Transaction on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[39] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[40] R. Farmani and J. A. Wright, "Self-adaptive fitness formulation for constrained optimization," *IEEE Trans. on Evolutionary Computation*, vol. 7, no. 5, pp. 445–455, Oct. 2003.

[41] E. Mezura-Montes and A. G. Palomeque-Ortiz, "Parameter control in differential evolution for constrained optimization," in *Proc. of the 2009 IEEE Congress on Evolutionary Computation*, 2009, pp. 1375–1382.