Efficient Constrained Optimization by the ε Constrained Adaptive 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 adaptive DE (εADE), which adopts a new and stable way of controlling the ε level and adaptive control of algorithm parameters in DE. The εADE is very efficient constrained optimization algorithm that can find high-quality solutions in very small number of function evaluations. It is shown that the εADE 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.**

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]–[3]. 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 [4]–[7], including engineering design problems [8]. The ε constrained method [4] 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 [9] and the hybrid algorithm of the εPSO and εGA [10]. It has been shown that the method has general-purpose properties.

T. Takahama is with the Department of Intelligent Systems, Hiroshima City University, Asaminami-ku, Hiroshima, 731-3194 Japan (e-mail: takahama@info.hiroshima-cu.ac.jp).

S. Sakai is with the Faculty of Commercial Sciences, Hiroshima Shudo University, Asaminami-ku, Hiroshima, 731-3195 Japan (e-mail: setuko@shudo-u.ac.jp).

In this study, we propose the ε constrained adaptive DE (εADE) in order to improve the efficiency, stability and usability of the εDE as follows:
(1) The efficiency depends largely on the selection of algorithm parameters in DE. In order to improve the efficiency, a simple but powerful scheme, which is a modification of the scheme proposed in JADE [11], is adopted. In the scheme, a scaling factor and a crossover rate are adaptively controlled according to their values in successful DE operation, where the generated child by the operation is better than the parent.
(2) In order to improve the stability, it is effective that all individuals approach to an optimal solution at a similar speed while maintaining diversity. To attain this speed, a simple modification to DE is introduced: When a parent generates a child and the child is not better than the parent, the parent can generate another child. This modification will average the approaching speed of individuals.
(3) In the εDE, users need to decide whether enabling or disabling the control of the ε level which keeps a balance between the optimization of the function value and the optimization of the constraint violation. If the feasible region is very small such as in problems with equality constraints, users should enable the control. Otherwise, users should disable the control to avoid the convergence of individuals out of the feasible region. However, it is not easy to decide whether the feasible region is very small or not when the feasible region is completely unknown. In this study, a simple scheme, where the ε level is adjusted with referring the constraint violation values of individuals, is introduced. This scheme can be thought as an adaptive control of the ε level. By introducing this scheme, users can always enable the control of the ε level even in problems with large feasible region. Thus, the usability is improved.

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

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

II. Previous works

A. 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 [12]–[15]. 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 [16], [17], 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 [18] proposed a method that adopts the extended objective function, which realizes the lexicographic ordering. Takahama and Sakai proposed the α constrained method [19], and ε constrained method [9] that adopt a lexicographic ordering with relaxation of the constraints. Runarsson and Yao [20] proposed the stochastic ranking method that adopts the stochastic lexicographic order, which ignores the constraint violation with some probability. Mezura-Montes and Coello [21] proposed a comparison mechanism that is equivalent to the lexicographic ordering. Venkatraman and Yen [22] 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 [23]–[29]. 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 [30] proposed a hybridization of the methods in category (2), (3) and (4).

B. Parameter control methods

Parameter control methods can be classified three categories [11]:
(1) Deterministic parameter control: The control parameters are altered by some deterministic rules without feedback from states of the evolutionary search, or individuals.
(2) adaptive parameter control: Feedback from the search states is used to dynamically alter the control parameters. Some DE-based algorithms are proposed [31]–[33]. In Section IV-B, the adaptation mechanism of JADE [11] is described briefly because the εADE adopts the similar adaptation.
(3) self-adaptive parameter control: The control parameters are associated with individuals and undergo mutation and recombination. Since better parameter values tend to generate good individuals, the values can be inherited to more offspring. Some DE-based algorithms are proposed [34].

It is thought that a good combination of constraint-handling techniques and parameter control techniques can achieve great improvement of the efficiency for constrained optimization algorithms. There are some studies on parameter control in DE for constrained optimization [35].

III. The ε constrained method

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.

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

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. 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.

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

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

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:

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

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

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 $f(\boldsymbol{x})$.

C. 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, (P$_{\leq_\varepsilon}$), is defined as follows:

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

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

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

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

For the three types of problems, (P$^\varepsilon$), (P$_{\leq_\varepsilon}$) and (P), the following theorems are given based on the ε constrained method [9].

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

Theorem 2: If an optimal solution of (P) exists, any optimal solution of (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 (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.

IV. The εADE

In this section, differential evolution and JADE are described first and then the εADE is defined.

A. Differential Evolution

Differential evolution (DE) is proposed by Storn and Price [36]. 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. $p1 \neq i$;
  $p2$=randint(1,$N$) s.t. $p2 \notin \{i, p1\}$;
  $p3$=randint(1,$N$) s.t. $p3 \notin \{i, p1, p2\}$;
  $\boldsymbol{x}'$=$\boldsymbol{x}^{p1}$+$F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3})$
```

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

In this study, DE/rand/1/exp and DE/rand/1/bin variants are used.

B. JADE

In JADE, the mean of the scaling factor μ_F and the crossover rate μ_{CR} are given, where initial values are μ_F=μ_{CR}=0.5. The scaling factor F_i and the crossover rate CR_i for each individual $\boldsymbol{x}^i$ are independently generated according to the following equations.

$$F_i = \text{rand}c_i(\mu_F, 0.1) \tag{8}$$

$$CR_i = \text{rand}n_i(\mu_{CR}, 0.1) \tag{9}$$

where randc_i is a random variable according to Cauchy distribution of location parameter μ_F and scale parameter 0.1,

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

Fig. 2. Crossover operations

randn_i is a random variable according to Normal distribution of mean μ_{CR} and standard deviation 0.1. CR_i is truncated to $[0,1]$ and F_i is truncated to be 1 if $F_i \geq 1$ or regenerated if $F_i \leq 0$. The means μ_F and μ_{CR} are updated as follows:

$$\mu_F = (1-c)\mu_F + cS_{F^2}/S_F \quad (10)$$

$$\mu_{CR} = (1-c)\mu_{CR} + cS_{CR}/S_N \quad (11)$$

where S_N is the number of successful operation, S_F, S_{F^2} and S_{CR} are the sum of F, F^2 and CR in successful operations, respectively. A constant c is a weight of update in (0,1] and the recommended value is 0.1.

C. The algorithm of the εADE

The εADE is DE that adopts an adaptive control of algorithm parameters, a simple modification in offspring generation, and the ε constrained method with an adaptive ε level control. As mentioned in Section I, in order to improve the efficiency and the stability, when a parent generates a child and the child is not better than the parent, the parent generate another child using an adaptive parameter control. In JADE, parameters are generated according to Cauchy or Normal distribution. In this study, uniform distribution in a narrow range is used to avoid unstable parameter change caused by wide perturbation.

The algorithm of the εADE[1] is as follows:

- Step0 Parameter setup. A fixed scaling factor F_0 and its perturbation width w_F, and a fixed crossover rate CR_0 and its perturbation width w_{CR} are given. The mean value of scaling factor $\mu_F = F_0$ and the mean value of crossover rate $\mu_{CR} = CR_0$.
- Step1 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.
- Step2 Initialization of the ε level. An initial ε level is given by the ε level control function $\varepsilon(0)$.
- Step3 Termination condition. If the number of function evaluations exceeds the maximum number of evaluations $FE_{\max}$, the algorithm is terminated.
- Step4 DE operation with fixed parameters. Each individual $\boldsymbol{x}^i$ is selected as a parent. If all individuals are selected, go to Step6. DE/rand/1/exp operation with a fixed scaling factor F_0 and a fixed crossover rate CR_0 is used and a new child $\boldsymbol{x}^{\text{child}}$ is generated. If the new one is not better than the parent based on the ε level comparison, or the operation is failed, go to Step5. Otherwise, the parent $\boldsymbol{x}^i$ is immediately replaced by the trial vector $\boldsymbol{x}^{\text{child}}$ because not discrete generation model but continuous generation model is adopted. Go back to the beginning of Step4 and the next individual is selected as a parent.
- Step5 DE operation with adaptive parameters. New F and CR are generated randomly by perturbing μ_F and μ_{CR} in a small range defined by w_F and w_{CR}. DE/rand/1/bin operation with F and CR is used and a new individual is generated again. If the new one is not better than the parent based on the ε level comparison, go back to Step4. Otherwise, the parent $\boldsymbol{x}^i$ is immediately replaced by the trial vector $\boldsymbol{x}^{\text{child}}$. Also, the operation is treated as a successful operation. The number of success S_N, sum of successful F, S_F and sum of successful CR, S_{CR} are updated. Go back to Step4.
- Step6 Update of adaptive parameters. The mean of the scaling factor μ_F and the mean of crossover rate μ_{CR} are updated using S_N, S_F and S_{CR}.
- Step7 Control of the ε level. The ε level is updated by the ε level control function $\varepsilon(t)$ with a truncation mechanism.
- Step8 Go back to Step3.

[1]downloadable at http://www.ints.info.hiroshima-cu.ac.jp/˜takahama/eng

D. Controlling the ε level

Usually, the ε level is controlled according to the equation (12). 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 set to 0 to obtain solutions with minimum constraint violation.

$$\begin{aligned} \varepsilon(0) &= \phi(\boldsymbol{x}_\theta) \\ \varepsilon(t) &= \begin{cases} \varepsilon(0)(1-\frac{t}{T_c})^{cp}, & 0 < t < T_c, \\ 0, & t \geq T_c \end{cases} \end{aligned} \quad (12)$$

where $\boldsymbol{x}_\theta$ is the top θ-th individual and $\theta = 0.2N$.

In this study, as mentioned in Section I, in order to improve the usability, a truncation mechanism is proposed. In order to recognize the states of individuals, $\phi_{\min}$, $\phi_{\max}$ and ϕ_0 are obtained, where $\phi_{\min}$ and $\phi_{\max}$ are the minimum and maximum value of constraint violations in the population, and ϕ_0 is a number of feasible individuals whose constraint violation is zero. Then, the value of $\varepsilon(t)$ is truncated to $[ap\,\phi_{\min}, ap\,\phi_{\max}]$, where ap is a positive constant. If the number of feasible individuals exceeds $ap\,\phi_0$, the value of $\varepsilon(t)$ is set to zero. The recommended value of ap is 0.9.

Fig. 3 shows the algorithm of the εADE.

```
εADE()
{
  $\mu_F$=$F_0$; $\mu_{CR}$=$CR_0$;
// Initialize the individuals
  $P$=$N$ individuals $\{\boldsymbol{x}^i\}$ generated randomly in $\mathcal{S}$;
  $FE$=$N$;
// Initialize the ε level
  $\varepsilon$=$\varepsilon(0)$;
  for($t$=1; $FE \le FE_{max}$; $t$++) {
    $S_N$=$S_F$=$S_{CR}$=0;
    for($i$=1; $i \le N$; $i$++) {
// DE operation with fixed parameters
      $F$=$F_0$; $CR$=$CR_0$;
      $\boldsymbol{x}^{\rm new}$=trial vec. generated by DE/rand/1/exp;
      $FE$=$FE$+1;
      if$\big((f(\boldsymbol{x}^{\rm new}), \phi(\boldsymbol{x}^{\rm new})) <_\varepsilon (f(\boldsymbol{x}^i), \phi(\boldsymbol{x}^i))\big)$
        $\boldsymbol{x}^i$=$\boldsymbol{x}^{\rm new}$;
      else {
// DE operation with adaptive parameters
        $F$=$\mu_F$+$w_F$ $u(-0.5, 0.5)$;
        Truncate $F$ to $[0.4,\ 0.9]$;
        $CR$=$\mu_{CR}$+$w_{CR}$ $u(-0.5, 0.5)$;
        Truncate CR to $[0,\ 1]$;
        $\boldsymbol{x}^{\rm new}$=trial vec. generated by DE/rand/1/bin;
        $FE$=$FE$+1;
        if$\big((f(\boldsymbol{x}^{\rm new}), \phi(\boldsymbol{x}^{\rm new})) <_\varepsilon (f(\boldsymbol{x}^i), \phi(\boldsymbol{x}^i))\big)$ {
          $\boldsymbol{x}^i$=$\boldsymbol{x}^{\rm new}$;
          $S_N$=$S_N$+1; $S_F$=$S_F$+$F$; $S_{CR}$=$S_{CR}$+$CR$;
        }
      }
    }
// Update adaptive parameters
    if($S_N$>0) {
      $\mu_F$=$(1-c)$ $\mu_F$+$c$ $S_F/S_N$;
      $\mu_{CR}$=$(1-c)$ $\mu_{CR}$+$c$ $S_{CR}/S_N$;
    }
// Control the ε level
    $\varepsilon$=$\varepsilon(t)$;
    if$\big(|\{\boldsymbol{x}^i|\phi(\boldsymbol{x}^i)=0\}| > ap\,N\big)$ $\varepsilon$=0;
    else Truncate $\varepsilon$ to $[ap\,\phi_{\min},\ ap\,\phi_{\max}]$;
  }
}
```

Fig. 3. The algorithm of the ε constrained adaptive differential evolution with adaptive control of the ε level, where F_0 and CR_0 are a fixed scaling factor and a fixed crossover rate, w_F and w_{CR} are a width of perturbation in F and CR, FE is the number of function evaluations, and $u(l,r)$ is a uniform random number generator in $[l,r]$.

V. Solving nonlinear optimization problems

In this paper, thirteen benchmark problems that are mentioned in some studies [3], [20], [21] are optimized, and the results by the εADE 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 the equation (13), which is adopted in many methods:

$$|h_j(\boldsymbol{x})| \le \delta,\ \delta > 0, \tag{13}$$

where $\delta = 10^{-4}$. Problem g12 has disjointed feasible regions. Table I shows the outline of the thirteen problems [21], [37]. 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 the equation (3). The ε level is controlled using the equation (12) with $cp = 5$, $T_c = 500$ and the truncation mechanism with $ap = 0.9$. The parameters for adaptive DE are: The number of search points $N = 40$, the maximum number of evaluations $FE_{\max} = 100,000$, the scaling factor $F_0 = 0.7$, the crossover rate $CR_0 = 0.9$, the perturbation width $w_F = 0.05$ and $w_{CR} = 0.05$ are common settings. In the εADE, it is difficult to specify the maximum number of generations, because a parent generates one or two children in one generation. Thus, only the maximum number of function evaluations (FEs) is specified. 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. Last two columns show the success rate (%) when fixed parameters (fix) and adaptive parameters (adp) are used, respectively.

For problems g01, g04, g05, g06, g08, g09, g11, g12 and g13, the optimal solutions are found consistently in all 30 runs. For other problems g03, g07 and g10, the optimal or near-optimal solutions are found in all 30 runs. These results show that the εADE is a very efficient and stable algorithm. As for the problem g02, the problem is a multimodal 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,

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	fix(%)	adp(%)
g01	-15.000	-15.000000	-15.000000	-15.000000	-15.000000	0.000e+00	44337.9	74267.8	18.4	11.4
g02	-0.803619	-0.803617	-0.803605	-0.803568	-0.803307	8.466e-05	54755.9	99639.2	5.1	3.7
g03	-1.000	-1.000500	-1.000500	-1.000500	-1.000500	5.404e-08	50722.5	97707.3	9.3	17.0
g04	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672	0.000e+00	22060.1	38972.9	6.0	4.7
g05	5126.498	5126.496714	5126.496714	5126.496714	5126.496714	0.000e+00	20037.7	62936.4	11.0	9.3
g06	-6961.814	-6961.813876	-6961.813876	-6961.813876	-6961.813876	1.827e-12	15845.7	36110.5	17.9	9.8
g07	24.306	24.306209	24.306209	24.306211	24.306220	3.227e-06	38401.7	99895.7	8.8	14.1
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0.000e+00	20604.7	30581.6	15.9	7.8
g09	680.630	680.630057	680.630057	680.630057	680.630057	0.000e+00	25399.7	50766.4	6.0	5.9
g10	7049.248	7049.248021	7049.248022	7049.248086	7049.249527	2.721e-04	24815.3	99865.7	8.6	12.4
g11	0.750	0.749900	0.749900	0.749900	0.749900	0.000e+00	8431.6	11749.3	4.9	2.5
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	0.000e+00	4210.5	6179.3	2.2	1.1
g13	0.053950	0.053942	0.053942	0.053942	0.053942	0.000e+00	22462.9	65999.9	8.1	7.7

but the εADE found near-optimal solutions under -0.803 constantly within 100,000 FEs. Thus, it is thought that the εADE has a high ability to solve multi-modal problems.

The εADE is a very fast algorithm. The average execution times ranged from 0.03 seconds to 0.46 seconds using a notebook PC with 2.0GHz Core2 Duo. The execution times are less than 1/5 seconds in all problems, except for g12.

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 εADE 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 6,000 to 100,000. The number of evaluations of the objective function ranged between about 4,000 and 55,000. For these problems, the εADE can omit the evaluation of the objective function about 30% to 75%. Therefore, the εADE can find optimal solutions very efficiently, especially from the viewpoint of the number of evaluations for the objective function. Also, it is shown that the adaptive parameters F and CR are more effective than the fixed parameters in g03, g07 and g10 although the success rate of the second child is clearly worse than that of the first child in case of using fixed parameters only.

These results show that the εADE 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 [21], the adaptive trade-off model (ATMES) proposed by Wang *et al.* [17], multiobjective method (HCOEA) proposed by Wang *et al.* [29], ECTHT-EP2 proposed by Mallipeddi and Suganthan [30], and the εDE proposed by Takahama and Sakai [4], 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 [35], 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 for 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 εDE is the best method followed by the εADE and ECHT-EP2, because the εDE found high-quality near optimal solutions stably in all problems. As for the εADE and ECHT-EP2, in problems g02, g07 and g10 the εADE found better solutions on average than ECHT-EP2. Also, the number of FEs in the εADE is much less than that in ECHT-EP2. Thus, it is thought that εADE is better than ECHT-EP2 from the viewpoint of the efficiency. As for the εDE and εADE, the εDE found better solutions than the εADE did. However, the difference is small enough. Also, the εDE needs 200,000 FEs and users must specify whether enabling the control of the ε level or not. Thus, from the viewpoint of the efficiency and the usability, the εADE is better than the εDE.

VI. CONCLUSIONS

Differential evolution is known as a simple, efficient and robust search algorithm that can solve unconstrained optimization problems. In this study, we proposed an adaptive control scheme of parameters and a simple idea of generating children for DE in order to improve the efficiency and stability. We proposed an adaptive control of the ε level using a truncation mechanism in the ε constrained method in order to improve the usability. By combining these ideas, we proposed the εADE. We showed that the εADE could solve thirteen benchmark problems most efficiently and stably compared with many other methods.

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

TABLE III

COMPARISON OF STATISTICAL RESULTS AMONG THE εADE, THE εDE [4], SMES [21], ATMES [17], HCOEA [29], ECHT-EP2 [30] AND A-DDE [35].

f & optimal	Statistics	εADE	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	-0.803617	-0.803618	-0.803601	-0.803388	-0.803241	-0.8036191	-0.803605
	median	-0.803605	-0.803614	-0.792549	-0.792420	-0.802556	-0.8033239	-0.777368
	mean	-0.803568	**-0.803613**	-0.785238	-0.790148	-0.801258	-0.7998220	-0.771090
	worst	-0.803307	-0.803588	-0.751322	-0.756986	-0.792363	-0.7851820	-0.609853
	σ	8.47e-05	**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	-1.000500	-1.000500	-1.000	-1.000	-1.000000	-1.0005	-1.000
	σ	5.40e-08	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	1.83e-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	24.306209	24.306209	24.426	24.313	24.3073055	24.3063	24.306
	mean	24.306211	**24.306209**	24.475	24.316	24.3073989	24.3063	24.306
	worst	24.306220	24.306209	24.843	24.359	24.3092401	24.3063	24.306
	σ	3.23e-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	7049.248022	7049.248021	7253.603	7215.357	7049.486145	7049.2488	7049.248
	mean	7049.248086	**7049.248021**	7253.047	7250.437	7049.525438	7049.2490	7049.248
	worst	7049.249527	7049.248021	7638.366	7560.224	7049.984208	7049.2501	7049.248
	σ	2.72e-04	**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

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) 7111.

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.

[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] 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.

[5] 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.

[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, "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*, July 2006, pp. 308–315.

[8] 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.

[9] 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

[10] 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.

[11] Z. Jingqiao 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.

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

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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, Sept. 2000.

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

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

[26] 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.

[27] 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.

[28] 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.

[29] 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.

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

[31] 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.

[32] 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.

[33] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proc. of the 2008 IEEE Congress on Evolutionary Computation*, 2008, pp. 1110–1116.

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

[35] 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.

[36] 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.

[37] 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.