Constrained Optimization by Combining the α Constrained Method with Particle Swarm Optimization

Tetsuyuki Takahama
Department of Intelligent Systems
Hiroshima City University
Hiroshima, 731-3194 Japan
email:takahama@its.hiroshima-cu.ac.jp

Setsuko Sakai
Faculty of Commercial Sciences
Hiroshima Shudo University
Hiroshima, 731-3195 Japan
email:setuko@shudo-u.ac.jp

***Abstract*— Recently, Particle Swarm Optimization (PSO) has been applied to various application fields. In this paper, a new optimization method "α Constrained Particle Swarm Optimizer (αPSO)", which is the combination of the α constrained method and PSO, is proposed. The αPSO is applied to several test problems such as nonlinear programming problems and problems with non-convex constraints. It is compared with GENOCOP5.0 which is one of the famous constrained optimization systems. It is shown that the αPSO is a very fast and good optimization algorithm for constrained optimization 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.

Simplex method is well known as an efficient method to solve linear programming problems. Also, there are some efficient methods to solve nonlinear optimization problems, such as a recursive quadratic programming, a projection method, and a generalized reduced gradient method [1]. These methods assume the differentiability of the objective function. However, it is difficult to apply these methods to problems, of which the objective function is not differentiable or the feasible set is not convex.

Generally, the constrained problems are solved by the combination of a transformation method and a direct search method. The transformation method [2] converts a constrained problem into an unconstrained one, and the direct search method optimizes the objective function by using only the value of it. There are some transformation methods such as a penalty method and a multiplier method. The penalty method is often used to solve optimization problems, because the solutions are often near the boundary of the feasible set and the method is used easily for its simplicity. However, it is difficult to know what value of the penalty coefficient leads to a feasible solution and how much a search point satisfies the constraints [3].

Recently, Particle Swarm Optimization (PSO) was proposed [4], [5] and has been applied to various application fields [6], [7]. In this paper, we propose a new constrained optimization method "*α Constrained Particle Swarm Optimizer* (αPSO)", which is the combination of the α constrained method [8], [9], [10], [11] and PSO. PSO was inspired by the movement of a group of animals such as a bird flock or fish school, in which the animals avoid predators and seek foods and mates as a group (not as an individual) while maintaining proper distance between themselves and their neighbors. PSO imitates the movement to solve optimization problems and is considered as a probabilistic multi-point search method like genetic algorithms (GAs) [12]. Several approaches to the constrained optimization have been proposed using GAs [13], [14], [15], [16], [17].

In the α constrained method, the *satisfaction level* of constraints is introduced to indicate how much a search point satisfies the constraints. Also, α *level comparison* that is an order relation and gives priority to satisfaction level over the value of objective function is defined. The α constrained method is a special transformation method. Usually transformation methods convert a constrained optimization problem into an unconstrained problem by changing the objective function. In contrast with the methods, the α constrained method does not transform the objective function, but transforms an algorithm for unconstrained optimization into an algorithms for constrained optimization by replacing the ordinal comparison with the α level comparison in direct search methods. In [8], [9], we proposed the α constrained Powell's method which is the combination of the α constrained method and Powell's direct search method [18]. In [10], [11], we proposed the α constrained Simplex method which is the combination of the α constrained method and Simplex method proposed by Nelder & Mead [19]. In [13], [14], we proposed the α constrained Genetic Algorithm (αGA) which is the combination of the α constrained method and GAs. We showed that the constrained optimization problems, of which the objective functions are not differentiable, are effectively solved by these methods. In this work, we also show that the α constrained method is applicable to PSO.

In αPSO, the agent or point which satisfies the constraints will move to optimize the objective function and the agent which does not satisfy the constraints will move to satisfy the

constraints, naturally. In this paper, the effectiveness of αPSO is shown by comparing αPSO with GENOCOP 5.0 [20], [21], which is the method of developing GENOCOPIII [17], [22], [23], on various types of test problems, such as linear programming problems, nonlinear programming problems, and problems with non-convex constraints.

II. Constrained optimization problems

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

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

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ is an n dimensional vector, $f(\boldsymbol{x})$ is an objective function, $g_j(\boldsymbol{x}) \leq 0,\ j = 1, \cdots, q$ are q inequality constraints and $h_j(\boldsymbol{x}) = 0,\ j = q+1, \cdots m$ are $m - q$ equality constraints. $f,\ g_j$ and h_j are linear or nonlinear real-valued functions. u_i and $l_i,\ i = 1, \cdots n$ are the upper bounds and the lower bounds of x_i, respectively. Also, let the feasible space in which every point satisfies all constraints be denoted by $\mathcal{F}$ and the set in which every point satisfies the upper and lower bound constraints be denoted by $\mathcal{S}\,(\supset \mathcal{F})$.

III. The α constrained method

We survey briefly the α constrained method [8], [9], [10], [11], [13], [14].

A. The constrained optimization problems and the satisfaction level

In the α constrained method, the *satisfaction level* of constraints $\mu(\boldsymbol{x})$, which indicates how much a search point $\boldsymbol{x}$ satisfies the constraints, is introduced. The satisfaction level $\mu(\boldsymbol{x})$ is the following function:

$$\begin{cases} \mu(\boldsymbol{x}) = 1, & \text{if } g_i(\boldsymbol{x}) \leq 0, h_j(\boldsymbol{x}) = 0 \\ & \quad \text{for all } i, j \\ 0 \leq \mu(\boldsymbol{x}) < 1, & \text{otherwise} \end{cases} \tag{2}$$

In order to define the satisfaction level $\mu(\boldsymbol{x})$ of (P), the satisfaction level of each constraint in (P) is defined and the all satisfaction levels are combined. For example, each constraint in (P) can be transformed into one of the following satisfaction levels that are defined by piecewise linear functions on g_i and h_j:

$$\mu_{g_i}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } g_i(\boldsymbol{x}) \leq 0 \\ 1 - \frac{g_i(\boldsymbol{x})}{b_i}, & \text{if } 0 \leq g_i(\boldsymbol{x}) \leq b_i \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$\mu_{h_j}(\boldsymbol{x}) = \begin{cases} 1 - \frac{|h_j(\boldsymbol{x})|}{b_j}, & \text{if } |h_j(\boldsymbol{x})| \leq b_j \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where b_i and b_j are proper positive real numbers. In order to obtain the satisfaction level $\mu(\boldsymbol{x})$ of (P), the satisfaction levels $\mu_{g_i}(\boldsymbol{x})$ and $\mu_{h_j}(\boldsymbol{x})$ need to be combined. Usually the minimization is used for the combination operator:

$$\text{Minimization} \quad \mu(\boldsymbol{x}) = \min_{i,j}\{\mu_{g_i}(\boldsymbol{x}), \mu_{h_j}(\boldsymbol{x})\} \tag{5}$$

B. The α level comparison

The α *level comparison* is defined as an order relation on the set of $(f(\boldsymbol{x}), \mu(\boldsymbol{x}))$. If the satisfaction level of a point is less than 1, the point is not feasible and its worth is low. The α level comparisons are defined by a lexicographic order in which $\mu(\boldsymbol{x})$ proceeds $f(\boldsymbol{x})$, because the feasibility of $\boldsymbol{x}$ is more important than the minimization of $f(\boldsymbol{x})$.

Let $f_1\,(f_2)$ and $\mu_1\,(\mu_2)$ be the function values and the satisfaction levels at a point $\boldsymbol{x}_1\,(\boldsymbol{x}_2)$, respectively. Then, for any α satisfying $0 \leq \alpha \leq 1$, α level comparison $<_\alpha$and $\leq_\alpha$between (f_1, μ_1) and (f_2, μ_2) is defined as follows:

$$(f_1, \mu_1) <_\alpha (f_2, \mu_2) \Leftrightarrow \begin{cases} f_1 < f_2, \text{ if } \mu_1, \mu_2 \geq \alpha \\ f_1 < f_2, \text{ if } \mu_1 = \mu_2 \\ \mu_1 > \mu_2, \text{ otherwise} \end{cases} \tag{6}$$

$$(f_1, \mu_1) \leq_\alpha (f_2, \mu_2) \Leftrightarrow \begin{cases} f_1 \leq f_2, \text{ if } \mu_1, \mu_2 \geq \alpha \\ f_1 \leq f_2, \text{ if } \mu_1 = \mu_2 \\ \mu_1 > \mu_2, \text{ otherwise} \end{cases} \tag{7}$$

In case of α=0, α level comparison $<_0$ and $\leq_0$ are equivalent to the ordinal comparison $<$ and $\leq$ between function values. Also, in case of $\alpha = 1$, $<_1$ and $\leq_1$ are equivalent to the lexicographic order in which the satisfaction level $\mu(\boldsymbol{x})$ precedes the function value $f(\boldsymbol{x})$.

C. The properties of the α constrained method

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

$$(\text{P}_{\leq_\alpha}) \quad \text{minimize}_{\leq_\alpha} \quad f(\boldsymbol{x}), \tag{8}$$

where $\text{minimize}_{\leq_\alpha}$ means the minimization based on the α level comparison $\leq_\alpha$. Also, a problem (P^α) is defined that the constraints of (P), that is, $\mu(\boldsymbol{x}) = 1$, is relaxed and replaced with $\mu(\boldsymbol{x}) \geq \alpha$:

$$\begin{aligned} (\text{P}^\alpha) \quad \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \mu(\boldsymbol{x}) \geq \alpha \end{aligned} \tag{9}$$

It is obvious that (P^1) is equivalent to (P).

For the three types of problems, (P^α), ($\text{P}_{\leq_\alpha}$) and (P), the following theorems are given [8], [9].

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

Theorem 2: If an optimal solution of (P) exists, any optimal solution of ($\text{P}_{\leq_1}$) is an optimal solution of (P).

Theorem 3: Let $\{\alpha_n\}$ be a strictly increasing non-negative sequence and converge to 1. Let $f(\boldsymbol{x})$ and $\mu(\boldsymbol{x})$ be continuous functions of $\boldsymbol{x}$. Assume that an optimal solution $\boldsymbol{x}^*$ of (P^1) exists and an optimal solution $\hat{\boldsymbol{x}}_n$ of ($P_{\leq_\alpha}$) exists for any α_n. Then, any accumulation point to the sequence $\{\hat{\boldsymbol{x}}_n\}$ is an optimal solution of (P^1).

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

IV. The α constrained particle swarm optimizer αPSO

In this section, the α constrained swarm optimizer αPSO is defined by combining the α constrained method with the direct search method PSO.

A. Particle swarm optimization PSO

Some kinds of animals avoid predators, and seek foods and mates as a group, not as an individual or an agent. In a group of animals, such as a bird flock, a fish school and a insect swarm, they synchronously move with maintaining proper distance between themselves and their neighbors. The group often changes the direction suddenly, scatters and regroups. It is thought that the behavior of the group can be explained by some simple rules. The agents in the group share information among all members in searching food. They profit from not only the private history but also the experience of all other members. The information sharing brings great advantage into the agents when the resource is unpredictably distributed in the searching field.

PSO is the multipoint search method that is developed through the simulation of above simplified social models. For PSO is based on such a very simple concept, it can be realized by primitive mathematical operators. It computationally is very efficient, runs very fast, and requires few memories.

Searching procedures by PSO can be described as follows: A group of agents optimizes a certain objective function $f(\cdot)$. At any time t, each agent i knows its current position $\boldsymbol{x}_i^t$. It also remembers its private best value until now $pbest_i$ and the position $\boldsymbol{x}_i^*$.

$$pbest_i = \min_{t=1,\cdots,k} f(\boldsymbol{x}_i^t) \tag{10}$$

$$\boldsymbol{x}_i^* = \arg\min_{t=1,\cdots,k} f(\boldsymbol{x}_i^t) \tag{11}$$

Moreover, every agent knows the best value in the group until now $gbest$ and the position $\boldsymbol{x}_G^*$.

$$gbest = \min_i pbest_i \tag{12}$$

$$\boldsymbol{x}_G^* = \arg\min_i f(\boldsymbol{x}_i^*) \tag{13}$$

The modified velocity $\boldsymbol{v}_i^{k+1}$ of each agent can be calculated by using the current velocity $\boldsymbol{v}_i^k$ and the difference among x_i^k, x_i^* and x_G^* as shown below:

$$\boldsymbol{v}_i^{k+1} = w\boldsymbol{v}_i^k + c_1\, rand\,(\boldsymbol{x}_i^* - \boldsymbol{x}_i^k) + c_2\, rand\,(\boldsymbol{x}_G^* - \boldsymbol{x}_i^k) \tag{14}$$

where $rand$ is a random number in the interval [0, 1]. The parameters c_1 and c_2 are called "cognitive" and "social" parameters, respectively, and they are used to bias the agent's search towards its own best previous position and towards the best experience of the group. The parameter w is called "inertia" weight and it is used to control the trade-off between the global and the local searching ability of the group.

Using the above equation (14), a certain velocity that gradually get close to $pbest_i$ and $gbest$ can be calculated. The position of agent i, $\boldsymbol{x}_i^k$, is replaced with $\boldsymbol{x}_i^{k+1}$ as follows:

$$\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k + \boldsymbol{v}_i^{k+1} \tag{15}$$

B. Algorithm of αPSO

The algorithm of αPSO can be defined by replacing the ordinary comparisons with the α level comparisons in PSO as follows:

1) Initializing agents
   Each agent is initialized with a point $\boldsymbol{x}$ and a velocity $\boldsymbol{v}$. Each point is generated as a random point in S. Each element of the velocity is generated as a random number in $[-v_{max}, v_{max}]$.
2) Finding the best agent
   All agents are compared by the α level comparison $<_\alpha$ and the best point $\boldsymbol{x}_G^*$ is decided.
3) Updating agents
   For each agent, the velocity $\boldsymbol{v}$ is updated based on the current velocity, the best point until now of the agent $\boldsymbol{x}_i^*$, and the best point until now in the group $\boldsymbol{x}_G^*$. The new point for the agent is updated by eqs.(14) and (15). If the current point is better than the best visited point of the agent, the best point is replaced by the current point. If the current point is better than the best visited point in the group, the best point is replaced by the current point.
4) Iteration
   If the number of iteration is smaller than the predefined number T, go back to 2. Otherwise the execution is stopped.

Following is the sample algorithm of αPSO coded by C-like language based on Shi's PSO code [24], [25].

```
αPSO()
{
  w=w^0;
  Initialize P(0);
  x_G^*=arg min_{<α} f(x), x in P(0);
  for(t=1;t≤T;t++) {
    for(each agent i in P(t)) {
      v_i=wv_i+φ_1(x_i^*-x_i)+φ_2(x_G^*-x_i);
```

```
        x_i=x_i+v_i;
        if(f(x_i)<_α f(x_i^*)) {
          x_i^*=x_i;
          if(f(x_i)<_α f(x_G^*)) x_G^*=x_i;
        }
      }
      w=w + (w^T − w^0)/T;
    }
  }
```

where w is the weight for the current velocity and is decreased from w^0 to w^T linearly, and ϕ_1 and ϕ_2 are uniform random variables in $[-2, 2]$.

V. Constrained nonlinear programming problems

In this paper, some test problems are optimized, and the results by αPSO are compared with those by GENOCOP5.0.

A. Test problems and the experimental conditions

The six problems are tested. Five test problems $G_1 - G_5$ are given by Michalewicz [16], [23] and a test problem S_1 is given by Sakawa [3]. These test problems are shown in Table I. All problems are nonlinear constrained programming problems. All constraints of G_1 are linear functions and the objective function of G_2 is a linear function. Also, G_4 has the equality constraints and S_1 is a nonlinear problem with non-convex constraints.

The standard settings in GENOCOP 5.0 are as follows: The population size of search points and that of reference points are 70 respectively. The maximum generations $T = 5000$ and the parameter of nonlinear ranking selection $c = 0.1$. Each crossover rate for the simple crossover, the arithmetical crossover and the heuristic crossover is $4/70$. Also, each mutation rate for the boundary mutation, the uniform mutation, the non-uniform mutation and the whole non-uniform mutation is $4/70$.

In GENOCOP 5.0, reference points that satisfy all constraints are searched randomly. However, when the feasible region is very narrow as in G_4, any reference point cannot be found, or even if some reference points were found, it couldn't be optimized sufficiently. Thus, based on Michalewicz's proposal, the equality constraints are relaxed, that is, all equality constraints $h_j(\boldsymbol{x}) = 0,\ j = q+1, \cdots, m$ are replaced by inequalities:

$$|h_j(\boldsymbol{x})| \leq \epsilon,\ \epsilon > 0 \tag{16}$$

In this experiments, $\epsilon = 10^{-3}$. Furthermore, when any reference point cannot be found, reference points are searched additionally, by minimizing the sum of squares of the functions in the equality constraints, $\sum_j h_j(\boldsymbol{x})^2$, with the simplex method of Nelder & Mead.

In αPSO, the same settings are used for all problems. The parameters for the α constrained method are defined as follows: Every satisfaction level are defined as piecewise linear functions like eqs.(3) and (4), and set the constants $b_i = b_j = 10000$. The satisfaction levels are combined by minimization eq.(5). The α level is fixed as $\alpha = 1$ for

TABLE I

TEST PROBLEMS

$G_1(\boldsymbol{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i,$
subject to
$2(x_1 + x_2) + x_{10} + x_{11} \leq 10,$
$2(x_1 + x_3) + x_{10} + x_{12} \leq 10,$
$2(x_2 + x_3) + x_{11} + x_{12} \leq 10,$
$-8x_1 + x_{10} \leq 0,\ -8x_2 + x_{11} \leq 0,$
$-8x_3 + x_{12} \leq 0,\ -2x_4 - x_5 + x_{10} \leq 0,$
$-2x_6 - x_7 + x_{11} \leq 0,\ -2x_8 - x_9 + x_{12} \leq 0,$
$0 \leq x_i \leq 1, i = 1, \ldots, 9,$
$0 \leq x_i \leq 100, i = 10, 11, 12,\ 0 \leq x_{13} \leq 1.$

$G_2(\boldsymbol{x}) = x_1 + x_2 + x_3,$
subject to
$1 - 0.0025(x_4 + x_6) \geq 0,$
$1 - 0.0025(x_5 + x_7 - x_4) \geq 0,$
$1 - 0.01(x_8 - x_5) \geq 0,$
$x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 \geq 0,$
$x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 \geq 0,$
$x_3x_8 - 1250000 - x_3x_5 + 2500x_5 \geq 0,$
$100 \leq x_1 \leq 10000,\ 1000 \leq x_i \leq 10000, i = 2, 3,$
$10 \leq x_i \leq 1000, i = 4, \cdots, 8.$

$G_3(\boldsymbol{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2$
$+10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7,$
subject to
$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0,$
$282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0,$
$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0,$
$-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0,$
$-10 \leq x_i \leq 10, i = 1, \ldots, 7.$

$G_4(\boldsymbol{x}) = \mathrm{e}^{x_1x_2x_3x_4x_5},$
subject to
$x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 = 10,$
$x_2x_3 - 5x_4x_5 = 0,\ x_1^3 + x_2^3 = -1,$
$-2.3 \leq x_i \leq 2.3, i = 1, 2,\ -3.2 \leq x_i \leq 3.2, i = 3, 4, 5.$

$G_5(\boldsymbol{x}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2$
$+4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2$
$+7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$
subject to
$105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \geq 0,$
$-10x_1 + 8x_2 + 17x_7 - 2x_8 \geq 0,$
$8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \geq 0,$
$-3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \geq 0,$
$-5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \geq 0,$
$-x_1^2 - 2(x_2 - 2)^2 + 2x_1x_2 - 14x_5 + 6x_6 \geq 0,$
$-0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \geq 0,$
$3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \geq 0,$
$-10 \leq x_i \leq 10, i = 1, \ldots, 10.$

$S_1(\boldsymbol{x}) = x_1^3 + (x_2 - 5)^2 + 3(x_3 - 9)^2 - 12x_3 + 2x_4^3$
$+4x_5^2 + (x_6 - 5)^2 - 6x_7^2 + 3(x_7 - 2)x_8^2$
$-x_9x_{10} + 4x_9^3 + 5x_1x_3 - 3x_1x_7 + 2x_8x_7,$
subject to
$-3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4$
$-2x_5x_6x_8 + 120 \geq 0,$
$-5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \geq 0,$
$-x_1^2 - 2(x_2 - 2)^2 + 2x_1x_2 - 14x_5 - 6x_5x_6 \geq 0,$
$-0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_5x_8 + 30 \geq 0,$
$3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \geq 0,$
$4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 105,$
$10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0,$
$-8x_1 + 2x_2 + 5x_9 - 2x_{10} \leq 12,$
$-5 \leq x_i \leq 10, i = 1, \ldots, 10.$

TABLE II

EXPERIMENTAL RESULTS

f	Optimal	Item	αPSO	GCOP5.0
G_1	-15.000	best	-15.000	-15.000
		average	-14.938	-11.523
		worst	-12.983	-3.442
		σ	0.340	4.447
		CPU(s)	2.337	24.810
G_2	7049.331	best	7061.810	7087.337
		average	7674.143	8114.309
		worst	12034.500	11107.814
		σ	720.713	966.195
		CPU(s)	1.644	19.470
G_3	680.630	best	680.631	680.634
		average	680.646	680.750
		worst	680.679	683.255
		σ	0.010	0.289
		CPU(s)	1.555	18.590
G_4	0.05395	best	0.06856	0.05583
		average	1.23149	1.40557
		worst	10.18960	18.68213
		σ	1.39802	2.53356
		CPU(s)	1.466	28.240
		#feasible	86	0
		const_err	4.552e-17	8.836e-4
G_5	24.306	best	24.319	24.529
		average	25.182	28.362
		worst	27.672	48.058
		σ	0.778	3.926
		CPU(s)	2.079	27.530
S_1	-216.602*	best	-216.655	-216.402
		average	-170.050	-84.652
		worst	9.085	513.794
		σ	80.8358	165.486
		CPU(s)	2.081	24.710

* The best value of S_1 that was found by [3].

all problems. Also, an equality constraint isn't transformed into eq.(16), but eq.(4) is used as the satisfaction level. The parameters for PSO are as follows: The number of agents $N = 70$, v_{max}=2, $w^0 = 1.0$, $w^T = 0.0$. The maximum number of iterations is $T = 5000$.

In works of Michalewicz, 10 trials were tested for each problem. Since good results might continue in 10 trials by chance, 100 trials are tested in order to evaluate more exactly in this paper.

B. Experimental results

Experimental results on the test problems are shown in Table II, in which each value is the average of 100 trials. "Optimal" is the optimal value in each problem. Also, "best", "average", "worst", and "σ" are the best value, the average value, the worst value, and the standard deviation of the objective function's values for the best agent in each trial, respectively. CPU(s) is the average execution time (seconds) for a trial using a computer with UltraSPARCIII (750MHz). #feasible shows the number of trials when feasible solutions are found. "const_err" shows how far the best agent is away from the feasible region, that is, $\max_{i,j}\{0, g_i(\boldsymbol{x}), |h_j(\boldsymbol{x})|\}$.

In all problems, αPSO found the better solutions than GENOCOP 5.0 on average. In G_1, αPSO and GENOCOP 5.0 found the optimal solution. In G_2, G_3, G_5 and S_1, the best values found by αPSO are smaller than those by GENOCOP 5.0, or αPSO found better solutions than GENOCOP 5.0. Especially in S_1, αPSO found the solutions better than the solution that was found by Sakawa.

TABLE III

RATIO OF FEASIBLE REGIONS

G_1	G_2	G_3	G_4	G_5	S_1
0.00023%	0.00064%	0.52685%	0.0%	0.0001%	0.00077%

In all problems, αPSO could find feasible solutions. In G_4, αPSO found the feasible solutions 86 times, but GENOCOP5.0 cannot find any feasible solution of G_4. The solutions obtained by GENOCOP 5.0 are away about $3 \times 10^{-4} \sim 1 \times 10^{-3}$ from the feasible region because the constraints are relaxed. Thus, it is thought that the ability of αPSO searching feasible solutions is higher for the problems with the equality constraints.

As for the execution time, αPSO is more than 10 times faster than GENOCOP5.0. Therefore, it is thought that αPSO is a very faster optimization algorithm than GENOCOP 5.0.

Table III shows ρ=$|\mathcal{F} \cap \mathcal{S}|/|\mathcal{S}|$, which is the ratio between the size of the feasible space and the whole search space. The ratio ρ for each problem is obtained by generating 10,000,000 random points within S and counting the points within F. From the table, the feasible region of G_3 is very wide and the feasible region of G_5 is very narrow. In both of the problems with wide and narrow feasible region, the considerably better solutions were found by αPSO than by GENOCOP 5.0. Thus, αPSO is a method that the solution with high precision can be obtained even if the problem has the wide and narrow feasible region.

VI. CONCLUSIONS

We proposed αPSO that combined the α constrained method with PSO. By applying αPSO to the six constrained optimization problems, it was shown that αPSO obtained an approximate solution near the optimal one for every problem by the numerical experiments and αPSO was a high precision and stable optimization algorithm. Also, by comparing αPSO with GENOCOP 5.0 which is known as an efficient algorithm for the constrained optimization problems, it was shown that αPSO was a very fast and good algorithm.

In this paper, only simple version of PSO is used. In the future, we are planning to introduce some operators such as the boundary mutation in order to improve the performance of αPSO. Also, we will apply αPSO to various application fields.

Acknowledgement

This research is supported in part by Grant-in-Aid for Scientific Research (C)(2)(No. 14580498, 16500083) of Japan society for the promotion of science.

REFERENCES

[1] D.G.Luenberger. *Linear and nonlinear programming*, Addison-Wesley, 1984.

[2] A.V.Fiacco, G.P.McCormick. *Nonlinear programming: sequential unconstrained minimization techniques*, Society for Industrial and Applied Mathematics, 1990.

[3] M.Sakawa and K.Yauchi. Floating Point Genetic Algorithms for Nonconvex Nonlinear Programming Problems: Revised GENOCOPIII, *IEICE Trans. on Information and systems*, vol.J81-A, no.1, pp90-97, Jan. 1998, In Japanese.

[4] J.Kennedy and R.Eberhart. Particle Swarm Optimization, *Proc. of IEEE International Conference on Neural Networks*, vol.IV, pp.1942–1948, Perth, Australia, 1995.

[5] J.Kennedy and R.C.Eberhart. *Swarm Intelligence*, Morgan Kaufmann, San Francisco, 2001.

[6] H.Yoshida, K.Kawata, Y.Fukuyama, S.Takayama and Y.Nakanishi. A Particle Swarm Optimization for Reactive Power and Voltage Control Considering Voltage Security Assessment, *IEEE Trans. on Power Systems*, vol.15, no.4, pp.1232–1239, Nov. 2001.

[7] T.Bartz-Beielstein, P.Limbourg, J.Mehnen, K.Schmitt, K.E.Parsopoulos and M.N.Vrahatis. Particle Swarm Optimizers for Pareto Optimization with Enhamced Archiving Techniques, *Proceedings of IEEE Congress on Evolutionary Computation 2003*, Canbella, Australia, pp.1780–1787, 2003.

[8] T.Takahama and S.Sakai. Tuning Fuzzy Control Rules by the α Constrained Method Which Solves Constrained Nonlinear Optimization Problems, *IEICE Trans. on Information and Systems*, vol.J82-A, no.5, pp.658–668, May 1999, in Japanese.

[9] 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*, vol.83, no.9, pp.1–12, 2000.

[10] T.Takahama and S.Sakai. Learning Fuzzy Control Rules by α-Constrained Simplex Method, *IEICE Trans. on Information and Systems*, vol.J83-D-I, no.7, pp.770–779, July 2000, in Japanese.

[11] T.Takahama and S.Sakai. Learning Fuzzy Control Rules by α-Constrained Simplex Method, *System and Computers in Japan*, vol.34, no.6, pp.80–90, 2003.

[12] D.E.Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, 1989.

[13] T.Takahama and S.Sakai. Constrained Optimization by α Constrained Genetic Algorithm (αGA), *Systems and Computers in Japan*, vol.35, no.5, pp.11–22, May 2004.

[14] T.Takahama and S.Sakai. Constrained Optimization by α Constrained Genetic Algorithm (αGA), *IEICE Trans. on Information and Systems*, vol.J86-D-I, no.4, pp.198–207, Apr. 2003.

[15] Z.Michalewicz. A Survey of Constraint Handling Techniques in Evolutionary Computation Methods, *Proc. of the 4th Annual Conference on Evolutionary Programming*, pp. 135–155, MIT Press, Cambridge, MA, 1995.

[16] Z.Michalewicz. Genetic Algorithms, Numerical Optimization and Constraints, *Proc. of the 6th International Conference on Genetic Algorithms*, pp.151–158, Pittsburgh, July 1995.

[17] Z.Michalewicz. *Genetic algorithm + data structures = evolution programs 3rd ed.*, Springer-Verlag, Berlin, 1996.

[18] M.J.D.Powell. An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives, *Computer J.*, vol.7, pp.155–162, 1964.

[19] J.A.Nelder and R.Mead. A Simplex Method for Function Minimization, *J. Computer*, vol.7, pp.308–313, 1965.

[20] S.Koziel and Z.Michalewicz. A Decoder-based Evolutionary Algorithm for Constrained Parameter Optimization Problems, *Proc. of the 5th Parallel Problem Solving from Nature*, Lecture Notes in Computer Science, vol.1498, pp.231–240, Amsterdam, Sep. 1998.

[21] S.Koziel and Z.Michalewicz. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization, *Evolutionary Computation*, vol.7, no.1, pp.19–44, 1999.

[22] Z.Michalewicz and G.Nazhiyath. Genocop III: A Co-evolutionary Algorithm for Numerical Optimization Problems with Nonlinear Constraints, *Proc. of the 2nd IEEE International Conference on Evolutionary Computation*, vol.2, pp.647–651, Perth, 1995.

[23] Z.Michalewicz and M.Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems, *Evolutionary Computation*, vol.4, no.1, pp.1–32, 1996.

[24] Y.Shi and R.Eberhart. A Modified Particle Swarm Optimizer, *Proc. of IEEE International Conference on Evolutionary Computation*, pp.69–73, Anchorage, May 1998.

[25] Y.Shi, http://www.engr.iupui.edu/˜eberhart/web/PSObook.html