A multi-objective constraint-handling method with PSO algorithm for constrained engineering optimization problems

Li, Lily; Li, Xiaodong; Yu, Xinghuo

Li, L., Li, X., & Yu, X. (2008). A multi-objective constraint-handling method with PSO algorithm for constrained engineering optimization problems. 2008 IEEE World Congress on Computational Intelligence, 1528–1535. https://doi.org/10.1109/CEC.2008.4630995

Published Version: https://doi.org/10.1109/CEC.2008.4630995

A Multi-Objective Constraint-Handling Method with PSO Algorithm for Constrained Engineering Optimization Problems

Lily D Li, Xiaodong Li, *Member, IEEE*, and Xinghuo Yu, *Fellow, IEEE*

Abstract — **This paper presents a multi-objective constraint handling method incorporating the Particle Swarm Optimization (PSO) algorithm. The proposed approach adopts a concept of Pareto domination from multi-objective optimization, and uses a few selection rules to determine particles' behaviors to guide the search direction. A goal-oriented programming concept is adopted to improve efficiency. Diversity is maintained by perturbing particles with a small probability. The simulation results on the three engineering benchmark problems demonstrate the proposed approach is highly competitive.**

I. INTRODUCTION

Over the last decade or two, evolutionary algorithms have been extensively studied as search and optimization tools in various problems domains. The primary reasons for their success are their broad applicability, ease of use and global perspective [1]. Although evolutionary algorithms have been successful in many applications, their uses in solving constrained optimization problems remain problematic because their original versions lack a mechanism to incorporate constraints into the fitness function [2-5]. There has been little work on handling constraints by the Particle Swarm Optimization (PSO) algorithm [5]. PSO is a relative new stochastic method for optimizing hard numerical functions based on metaphor of social behaviour of flocks of birds and schools of fish [6]. The PSO technique has proven to be effective and efficient for solving real-valued global unconstrained optimization problems [7, 8]. For constrained optimization problems, there have been only a few attempts. Hu and Eberhart [9] proposed to use a preserving feasibility strategy to handle constraints with PSO. The drawback of this model is that the initialization process may be impractically long or almost impossible for those CNOPs (Constrained Nonlinear Optimization Problems) that have extremely small feasible spaces[10] . Parsopoulos and Vrahatis [11] adopted the penalty functions in a PSO. However, in this model the penalty factors need to be carefully fine-tuned [4] and they are problem-dependent[12, 13]. Pulido and Coello introduced a selection rule based on feasibility checking and constraint violation measurement to handle constraints with PSO[14]. However, it appears that it is not convenient to measure the constraint violation units, and the performance is not consistent. Wei and Wang [15] integrated the multi-objective constraint handling mechanism with PSO. In [15], a selection strategy similar to [14] is used for constraint handling. The approach is only tested by four lower-constrained optimization problems with promising results.

Multi-objective constraint-handling was firstly proposed by Fonseca and Fleming back in 1995 (as cited in [3]). The main idea is to treat the constraints as extra objectives. By doing this, an original single objective constrained optimization problem can be transformed into a multi-objective unconstrained optimization problem. Then the techniques for multi-objective optimization can be employed. Since then, a number of models have been developed using this idea. These models can be classified into two groups - The first group transfers the original problem into an unconstrained bi-objective optimization problem (the original objective function and the sum of constraint violations); and the second group transfers the original problem into an unconstrained multi-objective optimization problem (constraints as separate objectives) [2, 3]. Several representative examples include COMOGA[16], Camponogara & Talukdar [17], Mezura-Montes & Coello [18-20] and Jimenez et al.[21]. Unfortunately, these models have some shortcomings. For example, some of them add extra computational cost [17, 20]; some of them require extra parameters [16, 18, 21]. A detailed review can be found in [3]. It is noticed that most of these models are built on genetic algorithms.

In this research, we propose a new method to integrate the multi-objective constraint handling mechanism with the PSO. By converting a single objective constrained optimization problem into a bi-objective unconstrained optimization problem, the proposed approach aims to minimize the original objective function and the total amount of constraint violations (the second objective). The concept of Pareto domination from multi-objective optimization is adopted in determining a particle's best past experience and the best social experience in the group. The second objective is used as a benchmark to select particles (defined in selection rules). A goal-oriented concept is adopted to improve efficiency. Diversity is maintained by a small perturbation.

The rest of the paper is organized as follows: Section II presents the problem definition, formulation and transformation; Section III describes the proposed multi-objective constraint handling incorporating with the PSO

Lily D Li, is with the School of Computing Sciences, Central Queensland University, Australia. (email: l.li@cqu.edu.au).

Xiaodong Li, is with the School of Computer Science and Information Technology, RMIT University, Australia. (email: xiaodong@cs.rmit.edu.au).

Xinghuo Yu, is with the School of Electrical and Computer Engineering, RMIT University, Australia. (email: x.yu@rmit.edu.au).

978-1-4244-1823-7/08/$25.00©2008 IEEE

algorithm. It includes a brief description to the PSO algorithm, selection rules and diversity control scheme. Section IV presents the simulation results for the three well-known engineering benchmark functions. Section V concludes the paper and indicates some future work.

II. PROBLEM DEFINITION, TRANSFORMATION AND FORMULATION

A general single objective constrained optimization problem can be stated as to:

$$\left.\begin{array}{ll} & \text{Find } \mathbf{X} \text{ which optimizes } f(\mathbf{X}) \\ \text{subject to} & g_i(\mathbf{X}) \le 0, \quad i = 1,2,\ldots,m; \\ & h_j(\mathbf{X}) = 0, \quad j = 1,2,\ldots,p; \end{array}\right\} \quad (1)$$

where $\mathbf{X} = [x_1, x_2, ..., x_n]^T$, with each x_i $(i = 1,...n)$ is bounded by lower and upper limits $L_i \le x_i \le U_i$; m is the number of inequality constraints and p is the number of equality constraints. In both cases, constraints can be linear or non-linear.

Note that a maximization problem can be converted into a minimization problem by setting $\max\{f(X)\} = -\min\{-f(X)\}$, and equality constraints $h_j(\mathbf{X}) = 0$ can be translated into inequality constraints $|h_j(\mathbf{X})| - \delta \le 0$, where δ is a allowed tolerance. The equation (1) then can be transformed into:

$$\left.\begin{array}{ll} \text{minimize} & f(\mathbf{X}) \\ \text{subject to} & g_i(\mathbf{X}) \le 0, \quad i = 1,2,....m. \end{array}\right\} \quad (2)$$

where m is the total number of constraints.

Camponogara & Talukar [17] proposed an approach in which a global optimization problem was transformed into a bi-objective problem where the first objective is to optimize the original objective function and the second is to minimize

$$\Phi(\mathbf{X}) = \sum_{i=1}^{m} \max(0, g_i(\mathbf{X})) \quad (3)$$

where $\Phi(\mathbf{X})$ is a total amount of constraint violations. From the equation (3), if a solution vector $\mathbf{X}$ satisfies all constraints, i.e., $g_i(\mathbf{X}) \le 0$ (i=1,2,…m), $\Phi(\mathbf{X})$ returns a zero, otherwise, it returns a positive number (total amount of constraint violations). Thus, the optimum value for $\Phi(\mathbf{X})$ is zero.

Therefore, the equation (2) can be transformed into the following:

$$\left.\begin{array}{ll} \text{minimize} & \mathrm{F}(\mathbf{X}) = (f(\mathbf{X}), \Phi(\mathbf{X})) \\ \text{where} & \Phi(\mathbf{X}) = \sum_{i=1}^{m} \max(0, g_i(\mathbf{X})) \end{array}\right\} \quad (4)$$

Now, the original single objective constrained optimization problem is transformed into a bi-objective unconstrained optimization problem.

For a general multi-objective optimization problem, the ideal procedure is to find a set of Pareto-optimal solutions first and then choose one solution from the set by using some other higher-level information [12]. For global constrained optimization, constraint satisfaction is a must and it is more important than real objective function minimization. That is, if a solution is not feasible, no matter how fit its objective function is, it is of little use. In other words, if a solution is feasible, even if it is not fit enough, it can be still considered as a candidate solution. Therefore, the second objective $\Phi = 0$ (totally constraint satisfied) or $\Phi \le \varepsilon$ (total constraint nearly satisfied), can be used as higher-level information to guide decision making during the search. The ε is a small positive number which indicates the feasibility tolerance. This is the so called "decision making during the search" approach in multi-objective optimization [22]. Fig. 1 below is an example to illustrate the Pareto-front, feasible solutions and the desired solution to the established bi-objective optimization problem as described by equation (4), the final solution will fall between A to B depending on how ε is selected.

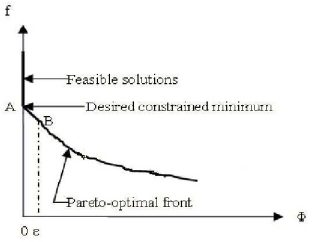

Fig.1. The Pareto-front, feasible solutions and desired constrained minimum for a bi-objective constraint handling optimization problem

Another concept adopted in the proposed algorithm is the so called goal oriented programming. The main idea in goal programming is to find solutions which attain a predefined target for one or more objective functions [12]. If there is no solution that has achieved pre-specified targets in all objective functions, the task is to find solutions which minimize deviations from the targets[12]. Suppose t is a predefined target for $f(\mathrm{X})$, the optimization problem in equation (4) can be transformed into

$$\left.\begin{array}{ll} \text{minimize} & \mathrm{F(X)} = (f(\mathbf{X}) - t, \Phi(\mathrm{X})) \\ \text{goal} & \Phi(\mathrm{X}) \le \varepsilon \text{ and } f(\mathbf{X}) - t \le \Delta \end{array}\right\} \quad (5)$$

where Δ and ε are two small positive numbers which indicate how close a solution to the predefined target. Please note there is no need to use $|f(\mathbf{X}) - t| \le \Delta$ because any better-than-target solutions are allowed to be found.

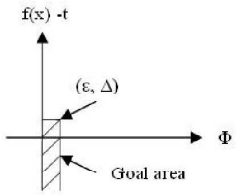

Fig.2. The solutions area for a bi-objective constraint handling optimization problem

Fig. 2 is an example to illustrate the solution area (goal area) for equation (5). By adding the goal to the optimization process, all particles should fly toward the goal area. Once the goals are achieved, no more evolution is needed. In case there is no solution satisfying both goals, the priority is given to $\Phi(\mathbf{X}) \le \varepsilon$, the solution is found from all particles where their $\Phi(\mathbf{X}) \le \varepsilon$.

Most multi-objective optimization methods use a domination concept to search for non-dominated solution, since this concept allows a way to compare solutions with multiple objectives. The definition for domination as stated below.

Definition: A solution $\mathbf{X}^{(1)}$ is said to dominate the other solution $\mathbf{X}^{(2)}$, if both conditions 1 and 2 are true:

1. *The solution $\mathbf{X}^{(1)}$ is no worse than $\mathbf{X}^{(2)}$ in all objectives, (for all j =1,2,…,m).*
2. *The solution $\mathbf{X}^{(1)}$ is strictly better than $\mathbf{X}^{(2)}$ in at least one objective, for at least one $j \in \{1, 2, \ldots, m\}$"* [12].

III. The Proposed Approach

A. The PSO Algorithm in Brief

A particle swarm optimization algorithm consists of individuals, called particles that form a swarm. Each particle represents a candidate solution to the problem. Particles change their positions by flying in a multi-dimensional search space looking for the optimal position. During flight, each particle adjusts its position according to its own experience and the experience its neighboring particles, making use of the best position encountered by itself and the best position in the entire population (or its local neighborhood). The performance of each particle is measured by a predefined fitness function (objective function) which is problem-dependent.

Let *i-th* particle in a D-dimensional search space be represented as $\mathbf{X}_i = (x_{i1}, x_{i2}, \ldots x_{iD})$. The best previous position of the *i-th* particle in the flight history is $PBest_i = (p_{i1}, p_{i2}, \ldots p_{iD})$. The position of the best particle of the neighborhood is $lBest_i = (p_{g1}, p_{g2}, \cdots p_{gD})$. The velocity for particle *i* is $\mathbf{V}_i = (v_{i1}, v_{i2}, \ldots v_{iD})$. In the PSO algorithm, the next position (*t+1*) of particle *i* on the dimension *d* is manipulated by the following equations (*t* denote the iteration):

$$v_{id}^{(t+1)} = \chi[w v_{id}^{(t)} + c_1 r_{1id}^{(t)}(pBest_{id}^{(t)} - x_{id}^{(t)}) + c_2 r_{2id}^{(t)}(lBest_{id}^{(t)} - x_{id}^{(t)})] \quad (6)$$

$$v_{id}^{(t+1)} = V_{max}, \quad if \ v_{id}^{(t+1)} > V_{max}$$

$$v_{id}^{(t+1)} = -V_{max}, \quad if \ v_{id}^{(t+1)} < -V_{max}$$

$$x_{id}^{(t+1)} = x_{id}^{(t)} + v_{id}^{(t+1)} \quad (7)$$

where d = 1,2,… D, D is the search dimension; i = 1,2,…N, and N is the number of particles in the swarm; w is the inertia weight; c_1 and c_2 are two positive constants, called the cognitive and social parameters respectively (or acceleration constants); r_{1id} and r_{2id} are two random numbers within the range [0, 1].

Although Clerc and Kennedy [23] suggested use of a constriction coefficient χ to the velocity formula and shows that the constriction coefficient can converge without using V_{max}. In order to ensure convergence and explore a wider area, in this research, both χ and V_{max} will be used.

As mentioned above, the original PSO algorithm and its variations have no mechanism to incorporate constraints handling into the algorithm. In order to integrate constraints handling with PSO, we introduce a few selection rules to determine particles' behaviour in the next section. A diversity control scheme is also introduced in the next section.

B. Selection Rules and Diversity Control

In PSO algorithm, a particle's best past experience and its group's best experience play a key role in guiding its search direction. For a multi-objective optimization problem, due to many objectives involved, the notion of dominance comparison is adopted [12]. The following selection rules are defined:

- Non-dominated particles are better than dominated ones.
- A particle with lower Φ (constraint violations) is better than a particle with higher Φ.

A perturbation with a minor probability of p is also introduced to maintain diversity and prevent premature convergence. It can be empirically derived.

C. The Proposed Algorithm

Fig. 3 illustrates the proposed algorithm. Comparing with the original PSO, the algorithm has the following features:

- Whenever calculating fitness, both objectives F=*f(X)-t* and Φ need to be evaluated;
- If a particle's new location is better than its best past location, the *pBest* is updated (decided by selection rules);
- A particle's best neighboring particle is determined by the two steps:

  a) Find all the non-dominated particles in the neighborhood;
  b) If there is only one non-dominated particle in the neighborhood, select it as *lBest*; otherwise

select one with the lowest Φ as *lBest* (the lower Φ means closer to the feasible region).

- A minor perturbation with the probability of p is introduced after the calculating the next particle position.
- If perturbed particle is better than the particle before perturbation, replace the particle with the perturbed one.
- After each iteration, check whether the goal is obtained; if obtained, the iteration ends.

```
GlobalF = POSITIVE_INFINITY;
P0 = URand (Li, Ui)
V0 = 0
F0 = Fitness_F (P0)
Φ0 = Fitness_Φ(P0)
pBest0 = P0
For i = 0 To n
   lBesti = LocalBest (Pi-1, Pi, Pi+1)      (Selection rules)
End for
Do
   For i = 0 To n
      Vi+1 = Speed ( Pi,Vi, pBesti, lBesti)   (Equation (6))
      Pi+1 = Pi+ Vi+1                         (Equation (7))
      r = URand (0,1)
      If (r ≤ p)
        TempPi+1 = Rand (Li, Ui)
        If (TempPi+1 isBetterThan Pi+1)   (Selection rules)
           Pi+1 = TempPi+1
      Fi+1 = Fitness_F(Pi+1)
      Φi+1 = Fitness_Φ(Pi+1)
      If (Pi+1  isBetterThan pBesti )     (Selection rules)
         pBesti = Pi+1
      If (Φi+1 ≤ ε)
         If (Fi+1 < GlobalF)
           GlobalF = Fi+1
   End For
   For i = 0 To n
      lBesti = LocalBest (Pi-1, Pi, Pi+1)   (Selection rules)
   End for
   If (GlobalF<= Δ)
      Output GlobalF and Pi+1
      Stop
End Do
```

Fig. 3. Pseudo code of the proposed multi-objective constraint-handling method with PSO algorithm

IV. ENGINEERING OPTIMIZATION EXAMPLES

This section reports the experiment results to the three well-known engineering optimization problems - the welded beam design problem (E01), the pressure vessel design problem (E02) and the spring design problem (E03). The proposed approach has been implemented in multithreaded Java programming language. The PSO neighborhood topology is set to ring topology with the neighbor size of 2. For example, if the neighbor size is 2, a particle with index i will have the particle index $i-1$ and particle $i+1$ as its neighbors. For each case, 30 independent runs (implemented in 30 threads) have been performed. The PSO parameters are: $w = 0$ *(Empirically derived);* $c_1=c_2=2$; $\chi=0.63$; $V_{max}=0.5*$*(decision variable range);* $p=0.1\%$; *number of particles* is *100;* the maximum iteration is set to 10,000. The feasibility tolerance allowed $\varepsilon = 1.0E\text{-}9$, that is, if a solution's total amount of constraint violation $\Phi \le \varepsilon$, the solution is feasible.

In order to see the improvement made from different approaches, results from other three most recent approaches are included in this section. The three approaches, CPSO [24], Hu et al. [25] (denoted as HES-PSO) and Coello&Montes [26] have been selected for comparison for the following reasons. Both CPSO and HES-PSO are based on the PSO algorithm as we adopted, however, they use the different constraint handling methods. CPSO adopted a penalty function method and HES-PSO adopted a preserving feasibility method. The Coello&Montes' approach uses the similar multi-objective constraint handling method as we adopted, but their implementation is through genetic algorithm. Therefore, we can evaluate the algorithm from different perspectives.

A. E01: Welded beam design problem

A welded beam is designed for minimum cost subject to constraints of shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c), end deflection of the beam (δ), and side constraints. There are four design variables: the thickness of the weld $h=x_1$, the length of the welded joint $l=x_2$, the width of the beam $t=x_3$ and the thickness of the beam $b=x_4$.

Please note the welded beam problem included in this paper is not exactly same as the original version proposed by Reklaitis et al in 1983 [27]. There are five constraints in the original version. For some reason, many researchers have studied another version of this problem as following with seven constraints. We include this version for comparison purpose.

The formal statement of the problem is the following:

Minimize

$$f(X) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2)$$

Subject to

$$g_1(X) = \tau(X) - \tau_{\max} \le 0$$
$$g_2(X) = \sigma(X) - \sigma_{\max} \le 0$$
$$g_3(X) = x_1 - x_4 \le 0$$
$$g_4(X) = 0.10471 x_1^2 + 0.04811 x_3 x_4 (14 + x_2) - 5 \le 0$$
$$g_5(X) = 0.125 - x_1 \le 0$$
$$g_6(X) = \delta(X) - \delta_{\max} \le 0$$
$$g_7(X) = P - P_c(X) \le 0$$

Where:

$$\tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1 x_2}$$

$$\tau'' = \frac{MR}{J}, \quad M = P(L + \frac{x_2}{2})$$

$$R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2}$$

$$J = 2\left\{\sqrt{2}x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\delta(X) = \frac{4PL^3}{Ex_3^3 x_4}, \qquad \sigma(X) = \frac{6PL}{x_4 x_3^2}$$

$$P_c(X) = \frac{4.013E\sqrt{(x_3^2 x_4^6)/36}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$P = 6000lb,\ L = 14in.,\ E = 30\times10^6\ psi,\ G = 12\times10^6\ psi$

$\tau_{\max} = 13,600psi,\ \sigma_{\max} = 30,000psi,\ \delta_{\max} = 0.25in.$

The ranges for the design variables are given as follows:

$0.1 \le x_1 \le 2.0,\quad 0.1 \le x_2 \le 10,$

$0.1 \le x_3 \le 10,\quad 0.1 \le x_4 \le 2.0.$

The best solutions found from our simulations and the three approaches discussed above are listed in Table I. In our approach, the best result is 1.724852321, which is close to the best-known result 1.724852; the mean result for 30 independent runs is 1.724861948; and the standard deviation is 2.05462E-05. According to the Table I, the solution found from our approach is better than those achieved by the other three approaches (the result presented in CPSO [24] seems incorrect because they used $\tau_{\max} = 13,000\ psi$ rather than $\tau_{\max} = 13,600\ psi$; the results presented in HES-PSO [25] have some constraints violated). The statistical data demonstrate our approach performs well in both search quality and consistency.

TABLE I
OPTIMAL SOLUTION OF E01 – WELDED BEAM DESIGN

Design variables	Best solution found			
	This paper	CPSO [24] (2007)	HES-PSO[25] (2003)	Coello& Montes[26] (2002)
$x_1(h)$	0.205729642	0.202369	0.20573	0.205986
$x_2(l)$	3.470488637	3.544214	3.47049	3.471328
$x_3(t)$	9.036623843	9.048210	9.03662	9.020224
$x_4(b)$	0.205729643	0.205723	0.20573	0.206480
$g_1(X)$	-1.67E-09	-12.839796	0.0	-0.103050
$g_2(X)$	-1.32E-05	-1.247467	0.0	-0.231748
$g_3(X)$	-6.08E-10	-0.001498	-5.55E-17	-0.000495
$g_4(X)$	-3.43298378	-3.429347	-3.432983785	-3.430043
$g_5(X)$	-8.07E-02	-0.079381	-0.0807296	-0.080986
$g_6(X)$	-2.36E-01	-0.235536	-0.2355403	-0.235514
$g_7(X)$	-2.47E-04	-11.681355	-9.09494E-13	-58.64688
$f(X)$	1.724852321	1.728024	1.72485084	1.728226

TABLE II
STATISTIC RESULTS TO DIFFERENT APPROACHES (WELDED BEAM DESIGN)

Test Case	Approach	Best	Mean	Std. Dev.
Case 1	This paper	1.72485231	1.73612022	2.46E-02
	CPSO [24]	1.728024	1.748831	1.29E-02
Case 2	This paper	1.72485231	1.73612022	2.46E-02
	HES-PSO[25]	1.72485084	NA	NA
Case 3	This paper	1.72485747	1.76521069	4.40E-02
	Coello& Montes[26]	1.728226	1.792654	7.47E-02

Case 1 and Case 2: 40 particles, 5000 iterations; Case 3: 40 particles, 2000 iterations

It needs to be mentioned that our solutions are generated based on 100 particles and 10,000 maximum iterations, which forms a total maximum number of 1,000,000 function evaluations. In fact, since a goal oriented programming concept is adopted in the program, (i.e., once the goal is reached, evolution stops), the total number of function evaluations, in most cases, is less than this maximum number setting. An experiment result will be presented later in this section.

To compare the results with others, we simulated our approach by adjusting the total maximum number of function evaluations to 200,000 (to match CPSO[24] and HES-PSO[25]), 80,000 (to match Coello&Montes[26]) respectively, based on the 30 independent runs, the simulation results are listed in Table II.

From the Table II, the proposed approach in this paper performs better (best solution found) than any of other three approaches. The mean results obtained by our approach are also better than others. Using standard deviation as a measure of consistency, our approach performs better than Coello&Montes's approach and slightly worse than the other two. However, they are still small and acceptable.

TABLE III
OPTIMAL SOLUTION OF E02 – PRESSURE VESSEL DESIGN

Design variables	Best solution found			
	This paper	CPSO [24] (2007)	HES-PSO [25] (2003)	Coello& Montes[26] (2002)
$x_1(T_s)$	0.79641436	0.812500	0.8125	0.812500
$x_2(T_h)$	0.39944942	0.437500	0.4375	0.437500
$x_3(R)$	41.0039194	42.091266	42.09845	42.097398
$x_4(L)$	190.801191	176.746500	176.6366	176.654047
$g_1(X)$	-5.04E-03	-0.000139	0.0	-0.000020
$g_2(X)$	-8.27E-03	-0.035949	-0.03588	-0.035891
$g_3(X)$	-595.450104	-116.382700	-5.8208E-11	-27.886075
$g_4(X)$	-49.1988089	-63.253500	-63.3634	-63.345953
$f(X)$	5971.4003	6061.0777	6059.1313	6059.94634

B. E02: Pressure vessel design problem

The pressure vessel design problem is a cylindrical vessel capped at both ends by hemispherical heads. The objective is to minimize the total cost, including the cost of the materials forming the welding. There are four design variables: Thickness of the shell $T_s = x_1$, thickness of the head $T_h = x_2$, the inner radius $R = x_3$, and the length of the cylindrical section of the vessel $L = x_4$. T_s and T_h are discrete values which are integer multiples 0.0625 in., in accordance with the available thickness of rolled steel plates, R and L are continuous.

The optimization problem can be expressed as follows:

Minimize

$$f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

Subject to

$$g_1(X) = 0.0193x_3 - x_1 \le 0$$

$$g_2(X) = 0.00954x_3 - x_2 \le 0$$

$$g_3(X) = 1296000 - \pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 \le 0$$

$$g_4(X) = x_4 - 240 \le 0$$

Where the design variables have to be in the following ranges:

$$0.0625 \le x_1 \le 6.1875, \quad 0.0625 \le x_2 \le 6.1875,$$
$$10 \le x_3 \le 200, \quad 10 \le x_4 \le 200.$$

The best solution found from our approach and the other three approaches are listed in Table III. The best result found from our approach is 5971.4003, which is the best solution ever reported. Before this paper, the best-known result was 6059.94634; the mean result for 30 independent runs is 6049.1590; and the standard deviation is 22.841537. As we can see, our approach performs well and is reasonably consistent. As above, the best results are generated with a particle size of 100 and the maximum iteration is 10,000.

From the Table IV, the proposed approach in this paper performs better in search quality than the other three. The mean results and standard deviations obtained by our approach are not as good as the others. By looking up the best solution generated from the original setting (100 particles, 10,000 maximum iterations) as stated above, it would appear that our approach needs more iterations to achieve more consistent results for this problem.

TABLE IV
STATISTIC RESULTS TO DIFFERENT APPROACHES (PRESSURE VESSEL DESIGN)

Test Case	Approach	Best	Mean	Std. Dev.
Case 1	This paper	5990.105542	6160.11071	145.152
	CPSO [24]	6061.0777	6147.1332	86.4545
Case 2	This paper	5990.105542	6160.11071	145.152
	HES-PSO[25]	6059.1313	NA	NA
Case 3	This paper	6035.857686	6362.82540	266.134
	Coello& Montes[26]	6059.9463	6177.2533	130.93

Case 1 and Case 2: 40 particles, 5000 iterations; Case 3: 40 particles, 2000 iterations

C. E03: Spring design problem

This problem consists of minimizing the weight of a tension/compression spring, subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. The three design variables are: the wire diameter $d = x_1$, the mean coil diameter $D = x_2$ and the number of active coils $N = x_3$.

The formal statement of the problem is as follows:

Minimize $f(X) = (x_3 + 2)x_2 x_1^2$

Subject to

$$g_1(X) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0$$

$$g_2(X) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \le 0$$

$$g_3(X) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0$$

$$g_4(X) = \frac{x_2 + x_1}{1.5} - 1 \le 0$$

The boundaries of the design variables are as follows:

$$0.05 \le x_1 \le 2, \quad 0.25 \le x_2 \le 1.3, \quad 2 \le x_3 \le 15.$$

The best solution found from our approach and other three approaches are listed in Table V. The best result found from our approach is 0.012665236, which is very close to the best-known solution 0.012665 and better than any of the three comparing approaches; the mean result from our approach is 0.012714543; and the standard deviation is 6.28E-05. These data demonstrate our approach performs really well in both search quality and consistency.

Table VI is a collection of data obtained by comparing with other approaches.

The data from Table VI indicate the proposed approach in this paper performs better than the three other approaches. The mean results and the standard deviations obtained by our approach are slightly worse than the others. However, they are quite small and in an acceptable range.

TABLE V
OPTIMAL SOLUTION OF E03 – TENSION/COMPRESSION STRING DESIGN PROBLEM

Design variables	Best solution found: This paper	CPSO [24] (2007)	HES-PSO[25] (2003)	Coello& Montes[26] (2002)
$x_1(d)$	0.051702169	0.051728	0.05146637	0.051989
$x_2(D)$	0.357033166	0.357644	0.35138395	0.363965
$x_3(N)$	11.27049760	11.244543	11.6086592	10.890522
$g_1(X)$	-4.07E-8	-0.000845	-0.0033366	-0.000013
$g_2(X)$	5.14E-9	-1.26E-05	-1.0970128	-0.000021
$g_3(X)$	-4.05440797	-4.051300	-4..026318	-4.061338
$g_4(X)$	-0.72750978	-0.727090	-0.7312393	-0.722698
$f(X)$	0.012665236	0.0126747	0.01266614	0.012681

TABLE VI
STATISTIC RESULTS TO DIFFERENT APPROACHES (TENSION/COMPRESSION STRING DESIGN PROBLEM)

Test Case	Approach	Best	Mean	StD. Dev.
Case 1	This paper	0.012666062	0.012812823	2.28E-04
	CPSO [24]	0.0126747	0.012730	5.20E-05
Case 2	This paper	0.012666062	0.012812823	2.28E-04
	HES-PSO[25]	0.01266614	NA	NA
Case 3	This paper	0.012666626	0.012964163	3.67E-04
	Coello& Montes[26]	0.0126810	0.0127420	5.90E-05

Case 1 and Case 2. 40 particles, 5000 iterations, Case 3. 40 particles, 2000 iterations

D. Discussion

As mentioned above, the best solutions generated from our approach are based on 100 particles and 10,000 maximum iterations, which forms a total number of 1,000,000 function evaluations. Although the maximum number of function evaluations looks larger, the actual numbers of function evaluations are less than this maximum because a goal oriented programming concept is adopted in

the program. That is, once the goal is reached, evolution stops. This approach suits real world applications better. Therefore, we have performed an experiment to illustrate how many minimum iterations/generations are needed to reach the best-known solutions or to achieve even better solutions. Table VII is a summary of the experiment results based on 30 independent runs and with a particle population size of 100. In Table VII, a goal result is the existing best-known result. The minimum tolerance allowed Δ indicates how close a solution is to a goal solution. If a solution is less than the goal solution (better than the best-known) or if its $f - goal \le \Delta$ (close enough to the goal), evolution stops. Since the problem E02 has a larger magnitude than E01 and E03, the Δ is set to a relative larger value.

According to the Table VII, the lowest number of iterations needed to reach the goal solution for the E01, E02 and E03 problems are 610, 680 and 183 respectively; the average number of iterations needed are 1597, 4210 and 1158 respectively. This mechanism makes the computation cost more rational. Under our multithreaded Java programming implementation, the computation costs to these three engineering problems are not a big issue. The feasibility tolerance ε impacts on the results. A larger ε can sometime make the constraints not fully satisfied but the search will be easier. We used a fairly small ε = 1.0E-9 to ensure the constraints are fully satisfied.

TABLE VII
ITERATIONS NEEDED TO REACH THE BEST-KNOWN SOLUTION

Problem	Goal (Best-Known)	Tolerance allowed Δ	Lowest*	Average*	Highest*
E01	1.724852	1.0E-04	610 (61000)	1597 (159700)	3324 (332400)
E02	6059.946	1.0E-01	680 (68000)	4210 (421000)	9998 (999800)
E03	0.012665	1.0E-04	183 (18300)	1158 (115800)	4590 (459000)

* indicate lowest iteration, average iteration and highest iteration needed (number in brackets indicates the number of function evaluation)

V. CONCLUSIONS

This paper has presented a multi-objective constraint handling method with the PSO algorithm for tackling constrained global optimization problems. The proposed approach adopts the concept of domination from multi-objective optimization, and uses a few selection rules and a goal oriented programming concept to improve search performance. Diversity is maintained by perturbing particles with a small probability. The simulation results to the three well-known engineering problems have been presented.

Compared with other work, our approach has achieved a better or very similar performance on the three well-known engineering problems. Remarkably, a best ever solution has been found for the engineering problem E02- *Pressure vessel design problem*. The proposed approach also performed well in consistency. Since the proposed approach has advantages of no problem-dependent parameters, and rational computation cost, it has potential to be applied to other constrained engineering optimization problems.

Our future work includes an extensive study of multi-objective constraint handling methods with the PSO algorithms and application of the proposed approach to more challenging real-world constrained engineering problems.

REFERENCES

[1] D. E. Goldberg, *Genetic Algorithms for Search, Optimization, and Machine Learning*: Addison-Wesley, 1989.

[2] C. A. C. Coello, "A Survey of Constraint Handling Techniques used with Evolutionary Algorithms, Technical Report Lania-RI-9904, Laboratorio, Nacional de Informtica Avanzada," 1999.

[3] E. Mezura-Montes and C. A. C. Coello, "A Survey of Constraint-Handling Techniques Based on Evolutionary Multiobjective Optimization." vol. 2007, 2006.

[4] E. Mezura-Montes and C. A. C. Coello, "Multiobjective-Based Concepts to Handle Constraints in Evolutionary Algorithms," in *Proceedings of the Fourth Mexican International Conference on Computer Science* Apizaco, MEXICO, 2003, pp. 192-199.

[5] G. T. Pulido and C. A. C. Coello, "A Constraint-Handling Mechanism for Particle Swarm Optimization," in *Proceedings of the 2004 Congress on Evolutionary Computation*. vol. 2 Portland, OR, USA, 2004, pp. 1396 - 1403.

[6] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedins of IEEE International Conference on Neural Networks*. vol. 4 Perth, Australia: IEEE Service Centerr, 1995, pp. 1942-1948.

[7] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*: San Francisco: Morgan Kaufmann Publisher, 2001.

[8] Y. Shi and R. Eberhart, "Parameter selection in particle swarm optimization," in *The 7th Ann. Conf. on Evolutionary Programming*, San Diego, CA, 1998.

[9] X. Hu and R. Eberhart, "Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization," in *Proceedings of the Sixth World Multiconference on Systemics, Cybernetics and Informatics*, 2002.

[10] G. Coath and S. K. Halgamuge, "A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems," in *Proceedings of the 2003 Congress on Evolutionary Computation* Newport Beach, CA, USA, 2003, pp. 2419-2425.

[11] K. E. Parsopoulos and M. N. Vrahatis, "Paricle Swarm Optimization Method for Constrained Optimization Problem," *Intelligent Technologies--Theory and Application: New Trends in Intelligent Technologies, vol. 76 of Frontiers in Artificial Intelligence and Applications, pp. 214-220, IOS Press,* pp. 214-220, 2002.

[12] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*: John Wiley & Sons, 2001.

[13] Ruhul Sarker, Masoud Mohammadian, and X. Yao, "Evolutionary Optimization," in *International Series in Operations Research & Management Science*, F. S. Hillier, Ed., 2002.

[14] G. T. Pulido and C. A. C. Coello, "A Constraint-Handling Mechanish for Particle Swarm Optimization," in *proceeding of CEC 2004* San Diego, CA, USA, 2004, pp. 1396-1403.

[15] J. Wei and Y. Wang, "A Novel Multi-objective PSO Algorithm for Constrained Optimization Problems," *LNCS,* vol. 4247, pp. 174-180, 2006.

[16] P. D. Surry and N. J. Radcliffe, "The COMOGA Method: Constrained Optimization by Multi-Objective Genetic Algorithms," *Control and Cybernetics,* vol. 26, 1997.

[17] E. Camponogara and S. N. Talukdar, "A Genetic Algorithn For Constrained and Multiobjective optimization," in *Proceeding of 3rd Nordic Workshop on Genetic Algorithms and Their Applications* Vaasa, Finland, 1995, pp. 49-62.

[18] C. A. C. Coello and E. Mezura-Montes, "Handling Constraints in Genetic Algorithms Using Dominance-Based Tournaments," in *Proceedings of the Fifth International Conference on Adaptive Computing Design and Manufacture* Devon, UK, 2002, pp. 273-284.

[19] C. A. C. Coello, "The use of a multiobjective optimization technique

to handle constraints."
[20] C. A. C. Coello, "Constraint-handling using an evolutionary multiobjective optimization technique," *Civil Engineering and Environmental Systems,* vol. 17, pp. 319-346, 2000.
[21] F. Jimenez, A. F. Gomez-Skarmeta, and G. Sanchez, "How Evolutionary Multiobjective Optimization can be used for Goals and Priorities based Optimization," in *Proceeding of AEB'02* Merida Espana., 2002.
[22] J. Horn, "Evolutionary Computation Applications F1.9 Multicriterion decision making," in *Handbook of Evolutionary Computation*: IOP Publishing LTD and Oxford University Press, 1997.
[23] M. Clerc and J. Kennedy, "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION,* vol. VOL. 6, NO.1, pp. 58-73, 2002.
[24] Q. He and L. Wang, "An effective co-evolutionary particle swarm optimization for constrainted engineering design problems," *Engineering Application of Artificial Intelligence,* vol. 20, pp. 89-99, 2007.
[25] X. Hu, R. C. Eberhart, and Y. Shi, "Engineering Optimization with Particle Swarm. Available: http://www.swarmintelligence.org/papers/SIS2003Engineering.pdf,
[26] C. A. C. Coello and E. M. Montes, "Constraint-handling in genetic algorithms through the use of dominance-based tournament selection," *Advanced Engineering Informatics,* vol. 16 pp. 193--203, 2002.
[27] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization Methods and Applications*. New York: Wiley, 1983.