Unified Particle Swarm Optimization for Solving Constrained Engineering Optimization Problems

K.E. Parsopoulos[1] and M.N. Vrahatis[2]

[1] Computational Intelligence Laboratory (CI Lab), Department of Mathematics, University of Patras, GR–26110 Patras, Greece
{kostasp, vrahatis}@math.upatras.gr
[2] University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras, GR–26110 Patras, Greece

Abstract. We investigate the performance of the recently proposed Unified Particle Swarm Optimization method on constrained engineering optimization problems. For this purpose, a penalty function approach is employed and the algorithm is modified to preserve feasibility of the encountered solutions. The algorithm is illustrated on four well–known engineering problems with promising results. Comparisons with the standard local and global variant of Particle Swarm Optimization are reported and discussed.

1 Introduction

Many engineering applications, such as structural optimization, engineering design, VLSI design, economics, allocation and location problems [1], involve difficult optimization problems that must be solved efficiently and effectively. Due to the nature of these applications, the solutions usually need to be constrained in specific parts of the search space that are delimited by linear and/or nonlinear constraints.

Different deterministic as well as stochastic algorithms have been developed for tackling such problems. Deterministic approaches such as Feasible Direction and Generalized Gradient Descent make strong assumptions on the continuity and differentiability of the objective function [1,2]. Therefore their applicability is limited since these characteristics are rarely met in problems that arise in real–life applications. On the other hand, stochastic optimization algorithms such as Genetic Algorithms, Evolution Strategies, Evolutionary Programming and Particle Swarm Optimization (PSO) do not make such assumptions and they have been successfully applied for tackling constrained optimization problems during the past few years [3, 4, 5, 6, 7].

Most of the aforementioned optimization algorithms have been primarily designed to address unconstrained optimization problems. Thus, constraint–handling techniques are usually incorporated in the algorithm in order to direct the search towards the desired (feasible) regions of the search space. The most common constraint–handling technique is the use of penalty functions [3, 8, 9, 7]. In these approaches, the problem is solved as an unconstrained one, where the

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3612, pp. 582–591, 2005.

objective function is designed such that non–feasible solutions are characterized by high function values (in minimization cases). The popularity of penalty–based approaches for constraint–handling is based mostly on their simplicity and direct applicability that does not involve neither modifications of the employed algorithm nor development of specialized operators to tackle constraints.

Unified Particle Swarm Optimization (UPSO) is a recently proposed PSO scheme that harnesses the local and global variant of PSO, combining their exploration and exploitation abilities without imposing additional requirements in terms of function evaluations [10]. Preliminary studies have shown that UPSO can tackle efficiently different optimization problems [10, 11].

We investigate the performance of UPSO on four well–known constrained engineering optimization problems. A penalty function approach is adopted and the obtained results are compared to that of the standard PSO algorithm, providing useful conclusions regarding the efficiency of the unified scheme. The rest of the paper is organized as follows. The employed penalty function is described in Section 2, while Section 3 is devoted to the description of UPSO. The considered test problems as well as the obtained results are reported and discussed in Section 4. The paper closes with conclusions in Section 5.

2 The Penalty Function Approach

The constrained optimization problem can be formulated, in general, as:

$$\min_{X \in S \subset \mathbb{R}^n} f(X), \tag{1}$$

$$\text{subject to} \quad g_i(X) \leqslant 0, \quad i = 1, \ldots, m, \tag{2}$$

where m is the number of constraints. Different inequality and equality constraints can be easily transformed into the form of Eq. (2). The corresponding penalty function can be defined as [3]:

$$F(X) = f(X) + H(X), \tag{3}$$

where $H(X)$ is a *penalty factor* that is strictly positive for all non–feasible solutions. Penalty functions with static, dynamic, annealing and adaptive penalties have been proposed and successfully applied in different applications [3, 7].

In the current study, we employed a penalty function that includes information about both the number of the violated constraints as well as the degree of violation. Thus, the penalty factor is defined as [8]:

$$H(X) = w_1 \, \mathrm{NVC}_X + w_2 \, \mathrm{SVC}_X, \tag{4}$$

where NVC_X is the number of constraints that are violated by X; SVC_X is the sum of all violated constraints, i.e.,

$$\mathrm{SVC}_X = \sum_{i=1}^{m} \max\{0, g_i(X)\},$$

and w_1, w_2, are static weights. The selection of this form of penalties was based on the promising results obtained by using such penalty functions with evolutionary algorithms [8].

In general, the penalty function influences heavily the performance of an algorithm in solving constrained optimization problems. Sophisticated and problem–based penalty functions can increase the algorithm's performance significantly. To avoid the possibly large influence of the employed penalty function on the performance of the algorithms, we used static weights w_1 and w_2, although self–adaptive approaches that modify the weights dynamically through co–evolution schemes, as well as more complicated penalty functions, have been successfully applied in relative works [8, 6].

3 Unified Particle Swarm Optimization

PSO is a stochastic, population–based algorithm for solving optimization problems. It was introduced in 1995 by Eberhart and Kennedy for numerical optimization tasks and its dynamic is based on principles that govern socially organized groups of individuals [12].

In PSO's context, the population is called a *swarm* and its individuals (search points) are called *particles*. Each particle has three main characteristics: an adaptable velocity with which it moves in the search space, a memory where it stores the best position it has ever visited in the search space (i.e., the position with the lowest function value), and the social sharing of information, i.e., the knowledge of the best position ever visited by all particles in its neighborhood. The neighborhoods are usually determined based on the indices of the particles, giving rise to the two main variants of PSO, namely the *global* and the *local* variant. In the former, the whole swarm is considered as the neighborhood of each particle, while in the latter strictly smaller neighborhoods are used.

Assume an n–dimensional function, $f : S \subset \mathbb{R}^n \to \mathbb{R}$, and a swarm, $\mathbb{S} = \{X_1, X_2, \ldots, X_N\}$, of N particles. The i–th particle, $X_i \in S$, its velocity, V_i, as well as its best position, $P_i \in S$, are n–dimensional vectors. A neighborhood of radius m of X_i consists of the particles $X_{i-m}, \ldots, X_i, \ldots, X_{i+m}$. Assume b_i to be the index of the particle that attained the best previous position among all the particles in the neighborhood of X_i, and t to be the iteration counter. Then, according to the *constriction coefficient* version of PSO, the swarm is updated using the equations [13],

$$V_i(t+1) = \chi\Big[V_i(t) + c_1 r_1\big(P_i(t) - X_i(t)\big) + c_2 r_2\big(P_{b_i}(t) - X_i(t)\big)\Big], \tag{5}$$

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{6}$$

where $i = 1, 2, \ldots, N$; χ is the constriction coefficient; c_1 and c_2 are positive constants, referred to as *cognitive* and *social* parameters, respectively; and r_1, r_2 are random vectors with components uniformly distributed in $[0, 1]$. Default values for χ, c_1 and c_2 are determined in the theoretical analysis of Clerc and Kennedy [13].

The performance of a population–based algorithm is heavily dependent on the trade–off between its *exploration* and *exploitation* abilities, i.e., its ability to explore wide areas of the search space and its ability to converge rapidly towards the most promising solutions, respectively. The global variant of PSO promotes exploitation since all particles are attracted by the same best position, thereby converging faster towards the same point. On the other hand, the local variant has better exploration properties since the information regarding the best position of each neighborhood is communicated to the rest of the swarm through neighboring particles. Therefore, the attraction to specific points is weaker, thus, preventing the swarm from getting trapped in local minima. Obviously, the proper selection of neighborhood size affects the trade–off between exploration and exploitation. However, the selection of neighborhood size is heavily based on the experience of the user [10].

The *Unified Particle Swarm Optimization* (UPSO) scheme was recently proposed as an alternative that combines the exploration and exploitation properties of both the local and global PSO variant [10]. Let $\mathcal{G}_i(t+1)$ and $\mathcal{L}_i(t+1)$ denote the velocity update of the particle X_i for the global and local PSO variant, respectively [10],

$$\mathcal{G}_i(t+1) = \chi\left[V_i(t) + c_1 r_1\big(P_i(t) - X_i(t)\big) + c_2 r_2\big(P_b(t) - X_i(t)\big)\right], \tag{7}$$

$$\mathcal{L}_i(t+1) = \chi\left[V_i(t) + c_1 r_1'\big(P_i(t) - X_i(t)\big) + c_2 r_2'\big(P_{b_i}(t) - X_i(t)\big)\right], \tag{8}$$

where t denotes the iteration number; b is the index of the best particle of the whole swarm (global variant); and b_i is the index of the best particle in the neighborhood of X_i (local variant). The main UPSO scheme is defined by [10]:

$$\mathcal{U}_i(t+1) = (1-u)\,\mathcal{L}_i(t+1) + u\,\mathcal{G}_i(t+1), \tag{9}$$

$$X_i(t+1) = X_i(t) + \mathcal{U}_i(t+1), \tag{10}$$

where $u \in [0,1]$ is a parameter called the *unification factor*, which balances the influence of the global and local search directions in the unified scheme. The standard global PSO variant is obtained by setting $u = 1$ in Eq. (9), while $u = 0$ corresponds to the standard local PSO variant. All values $u \in (0,1)$, correspond to composite variants of PSO that combine the exploration and exploitation characteristics of the global and local variant.

Besides the aforementioned scheme, a stochastic parameter that imitates mutation in evolutionary algorithms can also be incorporated in Eq. (9) to enhance the exploration capabilities of UPSO [10]. Thus, depending on which variant UPSO is mostly based, Eq. (9) can be written as [10],

$$\mathcal{U}_i(t+1) = (1-u)\,\mathcal{L}_i(t+1) + r_3\,u\,\mathcal{G}_i(t+1), \tag{11}$$

which is mostly based on the local variant, or

$$\mathcal{U}_i(t+1) = r_3\,(1-u)\,\mathcal{L}_i(t+1) + u\,\mathcal{G}_i(t+1), \tag{12}$$

which is mostly based on the global variant, where $r_3 \sim \mathcal{N}(\mu, \sigma^2 I)$ is a normally distributed parameter, and I is the identity matrix. Although r_3 imitates mutation, the obtained scheme is consistent with the PSO dynamics.

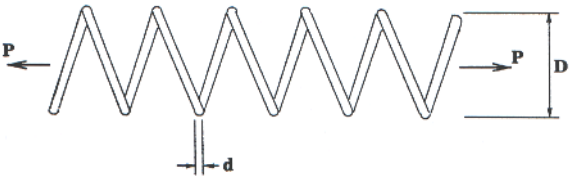

Fig. 1. The tension/compression spring problem

4 Results and Discussion

In the experiments we used four well–known constrained engineering optimization problems:

Problem 1: *Design of a tension/compression spring* [14]. This problem consists of the minimization of the weight of the tension/compression spring illustrated in Fig. 1, subject to constraints on the minimum deflection, shear stress, surge frequency, diameter and design variables. The design variables are the wire diameter, d, the mean coil diameter, D, and the number of active coils, N. The problem is formulated as:

$$\min_X \ f(X) = (N+2)Dd^2,$$

subject to:

$$\begin{aligned} g_1(X): &\quad 1 - \tfrac{D^3 N}{71785 d^4} \leqslant 0, \\ g_2(X): &\quad \tfrac{4D^2 - dD}{12566(Dd^3 - d^4)} + \tfrac{1}{5108 d^2} - 1 \leqslant 0, \\ g_3(X): &\quad 1 - \tfrac{140.45 d}{D^2 N} \leqslant 0, \\ g_4(X): &\quad \tfrac{D+d}{1.5} - 1 \leqslant 0, \end{aligned}$$

where $X = (d, D, N)^\top$. The desired ranges of the design variables are:

$$0.05 \leqslant d \leqslant 2.0, \quad 0.25 \leqslant D \leqslant 1.3, \quad 2.0 \leqslant N \leqslant 15.0.$$

Problem 2: *Design of a welded beam* [15]. This problem consists of the minimization of the cost of a welded beam illustrated in Fig. 2, subject to constraints on the 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, h, l, t and b that will be denoted as x_1, x_2, x_3 and x_4, respectively. The problem is formulated as:

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

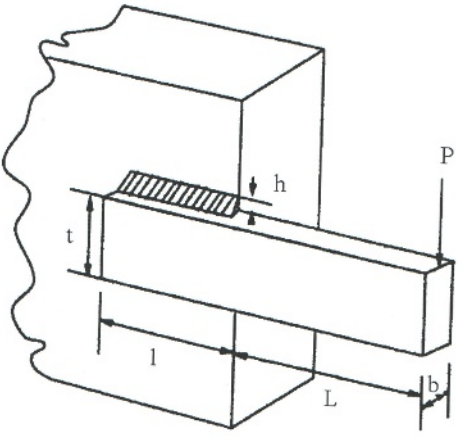

Fig. 2. The welded beam problem

subject to:

$$
\begin{aligned}
g_1(X): &\quad \tau(X) - \tau_{\max} \leqslant 0,\\
g_2(X): &\quad \sigma(X) - \sigma_{\max} \leqslant 0,\\
g_3(X): &\quad x_1 - x_4 \leqslant 0,\\
g_4(X): &\quad 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leqslant 0,\\
g_5(X): &\quad 0.125 - x_1 \leqslant 0,\\
g_6(X): &\quad \delta(X) - \delta_{\max} \leqslant 0,\\
g_7(X): &\quad P - P_c(X) \leqslant 0,
\end{aligned}
$$

where,

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

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J}, \quad M = P\left(L + \frac{x_2}{2}\right),$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},$$

$$\sigma(X) = \frac{6PL}{x_4x_3^2}, \quad \delta(X) = \frac{4PL^3}{Ex_3^3x_4}, \quad P_c = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),$$

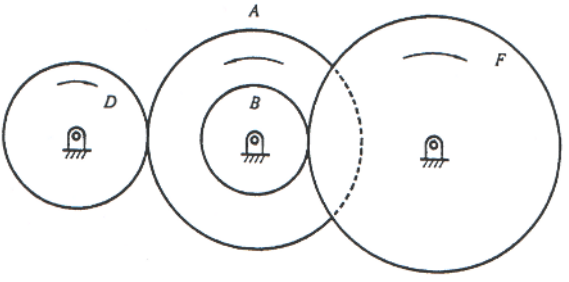

Fig. 3. The gear train problem

$$P = 6000 \text{ lb}, \quad L = 14 \text{ in}, \quad E = 30 \times 10^6 \text{ psi}, \quad G = 12 \times 10^6 \text{ psi},$$

$$\tau_{\max} = 13600 \text{ psi}, \quad \sigma_{\max} = 30000 \text{ psi}, \quad \delta_{\max} = 0.25 \text{ in},$$

and $X = (x_1, x_2, x_3, x_4)^\top$. The desired ranges of the design variables are:

$$0.1 \leqslant x_1, x_4 \leqslant 2.0, \quad 0.1 \leqslant x_2, x_3 \leqslant 10.0.$$

Problem 3: *Design of a gear train* [16]. This problem consists of the minimization of the cost of the gear ratio of the gear train illustrated in Fig. 3. The gear ratio is defined as:

$$\text{gear ratio} = \frac{n_B n_D}{n_F n_A},$$

where n_j denotes the number of teeth of the gearwheel j, with $j = A, B, D, F$. The design variables, n_A, n_B, n_D and n_F will be denoted as x_1, x_2, x_3 and x_4, respectively, and they are all integers in the range $[12, 60]$. The problem is formulated as:

$$\min_X \ f(X) = \left(\frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4} \right)^2,$$

subject to:

$$12 \leqslant x_i \leqslant 60, \quad i = 1, \ldots, 4.$$

Problem 4: *Design of a pressure vessel* [16]. This problem consist of the minimization of the cost of the pressure vessel illustrated in Fig. 4. The design variables are the shell's thickness, T_s, the thickness of the head, T_h, the inner radius, R, and the length, L, of the cylindrical section of the vessel, and they will be denoted as x_1, x_2, x_3 and x_4, respectively. The variables T_s and T_h are integer multiples of 0.0625, which represent the available thicknesses of rolled steel plates. The problem is formulated as:

$$\min_X \ f(X) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3,$$

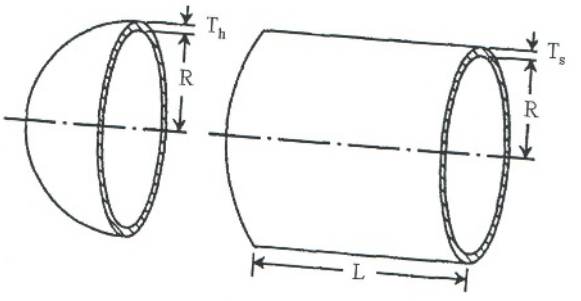

Fig. 4. The pressure vessel problem

subject to:

$$\begin{aligned} g_1(X): &\quad -x_1 + 0.0193x_3 \leqslant 0, \\ g_2(X): &\quad -x_2 + 0.00954x_3 \leqslant 0, \\ g_3(X): &\quad -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \leqslant 0, \\ g_4(X): &\quad x_4 - 240 \leqslant 0, \end{aligned}$$

where $X = (x_1, x_2, x_3, x_4)^\top$. The desired ranges of the design variables are:

$$1 \leqslant x_1, x_2 \leqslant 99, \quad 10.0 \leqslant x_3, x_4 \leqslant 200.0.$$

In all cases, the constriction coefficient PSO version was used with $\chi = 0.729$, $c_1 = c_2 = 2.05$. The neighborhood radius for the determination of the velocities in the local PSO variant was always equal to 1 (smallest possible neighborhood) in order to take full advantage of its exploration capabilities. For each test problem we applied the standard UPSO algorithm with $u = 0.2$ and 0.5, as well as UPSO with mutation (denoted as UPSOm) with $u = 0.1$, $\mu = (0, \ldots, 0)^\top$ and $\sigma = 0.01$. These choices were based on prior good performance on static optimization problems [10]. Also, the standard global and local PSO versions (derived for $u = 1$ and $u = 0$, respectively), were applied. In all problems, the swarm size was equal to 20, and the algorithm was allowed to perform 5000 iterations per experiment. We conducted 100 independent experiments per algorithm per problem, recording at each experiment the best solution detected by the swarm.

In order to preserve feasibility of the solutions, the update of the best positions of the particles was performed according to the scheme adopted by Hu *et al.* in [4]. More specifically, the best position of a particle was updated only if the new candidate best position was feasible, otherwise, it remained unchanged. Regarding the weights w_1 and w_2 of the penalty function in Eq. (4), the values $w_1 = w_2 = 100$ were used.

Table 1. The obtained results

Pr.		Standard UPSO				UPSOm
		$u = 0$	$u = 0.2$	$u = 0.5$	$u = 1$	
1	Mean	2.32563×10^{-2}	1.19291×10^{-1}	4.67351×10^{-2}	4.19581×10^{-2}	2.29478×10^{-2}
	StD	7.48230×10^{-3}	5.42710×10^{-1}	2.14505×10^{-1}	2.84724×10^{-2}	7.20571×10^{-3}
	Min	1.28404×10^{-2}	1.31269×10^{-2}	1.28158×10^{-2}	1.30803×10^{-2}	1.31200×10^{-2}
	Max	4.87550×10^{-2}	4.12260×10^{0}	1.57998×10^{0}	1.98921×10^{-1}	5.03651×10^{-2}
2	Mean	2.58869×10^{0}	2.29718×10^{0}	1.96820×10^{0}	4.27985×10^{0}	2.83721×10^{0}
	StD	5.01437×10^{-1}	4.10969×10^{-1}	1.55415×10^{-1}	1.36945×10^{0}	6.82980×10^{-1}
	Min	1.83008×10^{0}	1.82440×10^{0}	1.76558×10^{0}	1.91853×10^{0}	1.92199×10^{0}
	Max	4.13207×10^{0}	4.17382×10^{0}	2.84406×10^{0}	8.91270×10^{0}	4.88360×10^{0}
3	Mean	3.92135×10^{-8}	7.55581×10^{-8}	2.83820×10^{-7}	1.64225×10^{-6}	3.80562×10^{-8}
	StD	7.71670×10^{-8}	1.83057×10^{-7}	6.87035×10^{-7}	8.28521×10^{-6}	1.09631×10^{-7}
	Min	2.70085×10^{-12}	2.70085×10^{-12}	2.30781×10^{-11}	8.88761×10^{-10}	2.70085×10^{-12}
	Max	6.41703×10^{-7}	8.94899×10^{-7}	5.69940×10^{-6}	8.19750×10^{-5}	8.94899×10^{-7}
4	Mean	9.19585×10^{3}	8.66971×10^{3}	8.01637×10^{3}	1.35035×10^{5}	9.03255×10^{3}
	StD	9.60268×10^{2}	6.24907×10^{2}	7.45869×10^{2}	1.51116×10^{5}	9.95573×10^{2}
	Min	7.56796×10^{3}	6.77080×10^{3}	6.15470×10^{3}	7.52706×10^{3}	6.54427×10^{3}
	Max	1.26720×10^{4}	1.01895×10^{4}	9.38777×10^{3}	5.59300×10^{5}	1.16382×10^{4}

All results are reported in Table 1. More specifically, the mean, standard deviation, minimum and maximum value of the function values of the best solutions obtained in 100 experiments for each algorithm and problem are reported. In Problem 1, UPSOm (UPSO with mutation) had the overall best performance with respect to the mean objective function value of the best solutions as well as the standard deviation, although, the lowest minimum function value was obtained for the standard UPSO scheme with $u = 0.5$. In Problem 2, UPSO with $u = 0.5$ had the smallest mean, standard deviation and minimum of the objective function value of the best solutions, which is also true for Problem 4 with the exception of the standard deviation. In Problem 3, UPSOm had again the best mean, although the local PSO variant (UPSO with $u = 0$) was more robust, exhibiting the smallest standard deviation, while they had the same minimum value. In all cases except Problem 1, the global PSO variant had the worst mean and maximum value.

Summarizing the results, UPSO with $u = 0.5$ and UPSOm proved to be the most promising schemes, conforming with results obtained for different unconstrained optimization problems [10, 11]. The global PSO variant had the worst overall performance, while the local variant was competitive, however only in Problem 3 it outperformed UPSO with respect to the standard deviation and the minimum objective function value.

5 Conclusions

We investigated the performance of the recently proposed Unified Particle Swarm Optimization method on four well–known constrained engineering optimization problems, using a penalty function approach and a feasibility preserving mod-

ification of the algorithm. The results were very promising, with UPSO outperforming the standard PSO algorithm, conforming with previous results for different unconstrained optimization problems.

Further work will consider the investigation of the effect of the penalty function on the algorithm's performance as well as different feasibility preserving mechanisms.

Acknowledgment

This work was partially supported by the PENED 2001 Project awarded by the Greek Secretariat of Research and Technology.

References

1. Floudas, C.A., Pardalos, P.M.: A collection of test problems for constrained global optimization algorithms. In: LNCS. Vol. 455. Springer-Verlag (1987)
2. Himmelblau, D.M.: Applied Nonlinear Programming. McGraw–Hill (1972)
3. Coello Coello, C.A.: A survey of constraint handling techniques used with evolutionary algorithms. Techn. Rep. Lania–RI–99–04, LANIA (1999)
4. Hu, X., Eberhart, R.C., Shi, Y.: Engineering optimization with particle swarm. In: Proc. 2003 IEEE Swarm Intelligence Symposium. (2003) 53–57
5. Joines, J.A., Houck, C.R.: On the use of non–stationary penalty functions to solve nonlinear constrained optimization problems with ga's. In: Proc. IEEE Int. Conf. Evol. Comp. (1994) 579–585
6. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method for constrained optimization problems. In Sincak et al., eds.: Intelligent Technologies–Theory and Application: New Trends in Intelligent Technologies. Volume 76 of Frontiers in Artificial Intelligence and Applications. IOS Press (2002) 214–220
7. Yeniay, Ö.: Penalty function methods for constrained optimization with genetic algorithms. Mathematical and Computational Applications **10** (2005) 45–56
8. Coello Coello, C.A.: Self–adaptive penalties for ga–based optimization. In: Proc. 1999 IEEE CEC. Volume 1., Washington, D.C., USA (1999) 573–580
9. Coello Coello, C.A.: Use of a self–adaptive penalty approach for engineering optimization problems. Computers in Industry **41** (2000) 113–127
10. Parsopoulos, K.E., Vrahatis, M.N.: UPSO: A unified particle swarm optimization scheme. In: Lecture Series on Computer and Computational Sciences, Vol. 1, Proc. Int. Conf. Computational Methods in Sciences and Engineering (ICCMSE 2004), VSP International Science Publishers, Zeist, The Netherlands (2004) 868–873
11. Parsopoulos, K.E., Vrahatis, M.N.: Unified particle swarm optimization in dynamic environments. LNCS **3449** (2005) 590–599
12. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings Sixth Symposium on Micro Machine and Human Science, Piscataway, NJ, IEEE Service Center (1995) 39–43
13. Clerc, M., Kennedy, J.: The particle swarm–explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6** (2002) 58–73
14. Arora, J.S.: Introduction to Optimum Design. McGraw–Hill, New York (1989)
15. Rao, S.S.: Engineering Optimization–Theory and Practice. Wiley (1996)
16. Sandgen, E.: Nonlinear integer and discrete programming in mechanical design optimization. Journal of Mechanical Design (ASME) **112** (1990) 223–229