Looking Inside Particle Swarm Optimization in Constrained Search Spaces

Jorge Isacc Flores-Mendoza and Efrén Mezura-Montes

Laboratorio Nacional de Informática Avanzada (LANIA A.C.)
Rébsamen 80, Centro, Xalapa, Veracruz, 91000, México
jflores@lania.edu.mx, emezura@lania.mx

Abstract. In this paper, the behavior of different Particle Swarm Optimization (PSO) variants is analyzed when solving a set of well-known numerical constrained optimization problems. After identifying the most competitive one, some improvements are proposed to this variant regarding the parameter control and the constraint-handling mechanism. Furthermore, the on-line behavior of the improved PSO and some of the most competitive original variants are studied. Two performance measures are used to analyze the capabilities of each PSO to generate feasible solutions and to improve feasible solutions previously found i.e. how able is to move inside the feasible region of the search space. Finally, the performance of this improved PSO is compared against state-of-the-art PSO-based algorithms. Some conclusions regarding the behavior of PSO in constrained search spaces and the improved results presented by the modified PSO are given and the future work is established.

1 Introduction

The use of nature-inspired heuristics to solve complex search problems, like numerical optimization problems, has been extended in recent years. Evolutionary Computing (EC) [1], which emulates natural selection and survival of the fittest comprises the first set of these kind of techniques. However, in the mid 1990's the emulation of emergent social behaviors among insects and bird flocks has been proposed as a new area known as Swarm Intelligence (SI) [2]. The two SI initial paradigms are Ant Colony Optimization (ACO) [3] and Particle Swarm Optimization [2]. ACO is based on the foraging behavior of ants and has been mainly used to solve combinatorial optimization problems. On the other hand, PSO is based on the social behavior of bird flocks when moving from one place to another and was proposed mainly to solve numerical optimization problems.

PSO is based on social relationships established among simple individuals in a group. There is a leader in the flock which is followed by the others members. However, each member has a memory about its position in the group. In this way, the individual is able to take a decision based on its own knowledge (cognitive element) and also based on the behavior of its neighbors (social element). PSO is easy to implement and its performance is sometimes better than other nature-inspired heuristics [2].

In PSO, an initial swarm of solutions called particles $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]$ are generated at random. These particles will "fly" in the search space as to locate promising areas and reach a good (optimal) solution. Particles find their search direction by combining social (the position of the best particle in the swarm called "leader") and cognitive (its best position reached so far, called "pbest") information. The position of a particle is changed by adding an updated velocity to the current position: $\boldsymbol{x}(t+1) = \boldsymbol{x}(t) + \boldsymbol{v}(t+1)$, called flight formula. Two different approaches to calculate the velocity vector are presented in this study. The first is the PSO with inertia weight and the second is the PSO with constriction factor:

PSO with inertia weight. The inertia weight [2] was added to the velocity update formula to calibrate the influence of the previous particle's velocity $\boldsymbol{v}_i(t)$. The formula to calculate the new velocity $\boldsymbol{v}_i(t+1)$ is the following: $\boldsymbol{v}_i(t+1) = w * \boldsymbol{v}_i(t) + c_1 * rand() * (\boldsymbol{x}_{pbest_i} - \boldsymbol{x}_i) + c_2 * rand() * (\boldsymbol{x}_{gBest_i} - \boldsymbol{x}_i)$, where w is the inertia weight, $\boldsymbol{x}_{pbest_i}$ is the particle's pbest, $\boldsymbol{x}_{gBest_i}$ is the position of the leader, c_1 and c_2 are the acceleration constants (user-defined) which control the influence of the cognitive (memory of the particle) and social (position of the leader) elements respectively, $rand()$ is a function that generates a uniform-distributed random real number between 0 and 1.

PSO with constriction factor. It was proposed by Clerc and Kennedy in [4] in order to improve the exploration-exploitation capabilities of PSO. The constriction factor k is included in the velocity update formula as follows: $\boldsymbol{v}_i(t+1) = k * [\boldsymbol{v}_i(t) + c_1 * rand() * (\boldsymbol{x}_{pbest_i} - \boldsymbol{x}_i) + c_2 * rand() * (\boldsymbol{x}_{gBest_i} - \boldsymbol{x}_i)]$, where the constriction factor k is calculated by means of the acceleration constants c_1 and c_2 (the remaining elements of the formula are the same of the inertia weight velocity update formula). The authors claim that the constriction factor PSO variant, under certain parameters offers a better velocity control [4]. However, these conclusions are based on unconstrained numerical optimization problems.

There are two main communication variants in PSO: (1) Global best (GBPSO), where all particles can communicate in the swarm (star social network structure) and there is just one global leader ($gBest$), and (2) Local best (LBPSO), where particles can only communicate with others in their vicinities (ring social network structure) and there are several leaders ($lBest_i$) depending of the number of neighborhoods defined. Usually, GBPSO converges faster than LBPSO, but the first has a higher tendency to get trapped in local optima while the second may be more robust to avoid them [2]. The general PSO pseudocode is presented in Figure 1.

The problem of interest in this paper is the numerical constrained optimization problem (NCOP), which can be defined as follows: Find $\boldsymbol{x}$ which minimizes $f(\boldsymbol{x})$ subject to: $g_i(\boldsymbol{x}) \leq 0, \;\; i = 1, \ldots, m$, and $h_j(\boldsymbol{x}) = 0, \;\; j = 1, \ldots, q$ where $\boldsymbol{x} \in \mathbb{R}^n$ is the vector of solutions $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$, where each $x_i, \;\; i = 1, \ldots, n$ is bounded by lower and upper limits $L_i \leq x_i \leq U_i$ which define the search space $\mathcal{S}$, $\mathcal{F}$ is the feasible region and $\mathcal{F} \subseteq \mathcal{S}$; m is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or nonlinear). Equality constraints are transformed into inequalities constraints of the form: $\mid h_j(\boldsymbol{x}) \mid - \epsilon \leq 0$, where ϵ is the tolerance allowed (a very small value).

PSO, as other bio-inspired heuristics like EC, in their original versions, are designed to solve unconstrained optimization problems i.e. they lack a mechanism to incorporate feasibility information of solutions in their fitness values. There are excellent surveys about the research about constraint-handling techniques used in EC and also in SI [5]. The most popular approach to deal with constraints is the use of penalty functions, which aim is to decrease the fitness (i.e. quality measure) of infeasible solutions. There are other methods based on preserving feasible solutions, methods which make a clear distinction between feasible and infeasible solutions and also hybrid methods [5].

Regarding the use of PSO to solve NCOPs there are works reported in the specialized literature. They can be classified in (1) approaches based on penalty function and (2) based on methods of separation of objective and constraints. Examples of methods based on penalty functions are the following: Parsopoulos and Vrahatis [6] used a static penalty function and stochastic parameter in a combined GBPSO variant with constriction factor. Li, Tian and Kong [7] proposed a GBPSO with inertia weight using an adaptive penalty function and a mutation strategy based on the population diversity. Krohling and Do Santos Coehlo [8] used a co-evolutionary approach in a Lagrangian function with two sub-swarms, one of them optimizes the original problem and the other one aims to find the optimal values for the Lagrangian values. He, Prempain and Wu [9] proposed a GBPSO with inertia weight and with a "fly-back" (i.e. death penalty) mechanism which only lets the PSO fly inside the feasible region of the search space. On the other hand, approaches based on separation of objective function and constraints are the following: Toscano and Coello [10] proposed a turbulence operator in a GBPSO with inertia weight where the leader is chosen based on the lowest number of violated constraints (when infeasible particles are in the current swarm). Liang and Suganthan [11] used a LBPSO where each sub swarm (i.e. neighborhood) is focused on either satisfying a single constraint or optimizing the objective function, regardless of feasibility information. Cagnina et al. [12] presented a combination of global-local best PSO with inertia weight and a constraint handling method based on feasibility rules which prefer feasible solutions over infeasible ones and among infeasible solutions chooses those with a lower amount of constraint violation. Lu and Chen [13] proposed the use of a bi-objective approach based on a GBPSO with inertia weight, one objective is the original objective function and the second one is the sum of constraint violation.

As it can be seen from the related work, the design is centered on the constraint-handling technique and, most of the time, the PSO variant is chosen without knowing

- Begin
  - Generate a swarm of random solutions.
  - Evaluate the fitness of each particle in the swarm
  - do
    - Select the leader(s) of the swarm.
    - For each particle, update its position with the flight formula.
    - Evaluate the fitness of the new position of each particle
    - Update the $pbest$ (memory) value of each particle.
  - Until $t = Gmax$.
- End

Fig. 1. Basic PSO algorithm. GMAX is the number of generations of the algorithm.

if, indeed, is the most suitable for constrained search spaces and for the constraint-handling technique used. Consequently, it is common to see additional operators like mutation [10], combination of communication models e.g. global-local best PSO [12, 6] and mechanisms like co-evolution [8]. In this work, a performance analysis of original PSO variants when solving NCOPs is presented. A simple parameterless constraint-handling technique was used as to do not introduce any additional bias to the original PSO. Based on the most competitive variant, two small improvements are made to it regarding parameter control and constraint-handling. After that, the on-line behavior of this improved variant is compared with respect to the original ones by using two performance measures. The hypothesis of this work is that the more knowledge about the original PSO behavior may lead to design competitive but less complex approaches, in this case, to solve NCOPs by using the search abilities provided by the search engine itself.

The paper is organized as follows: Section 2 presents an empirical comparison of four basic PSO variants. Based on the results obtained, the most competitive variant is improved in Section 3. The on-line behavior analysis of this new approach, as well as other PSO variants, is presented in Section 4. Finally, in Section 5 the conclusions of this work are enumerated and the future work is established.

2 Empirical Comparison of PSO variants

As a first step in this proposal, the four most used PSO variants in constrained optimization, in their original versions, were compared in a benchmark found in the specialized literature [13] which comprises minimization problems with different features (See Table 1). Each PSO variant used in this comparison was implemented.

Table 1. Main features for each benchmark problem used in the experiments. ρ is the estimated size of the feasible region with respect to the whole search space [5], n is the number of decision variables, LI is the number of linear inequality constraints, NI the number of nonlinear inequality constraints, LE is the number of linear equality constraints and NE is the number of nonlinear equality constraints.

Problem	**n**	**Type of function**	ρ	**LI**	**NI**	**LE**	**NE**
g01	13	quadratic	0.0003%	9	0	0	0
g02	20	nonlinear	99.9973%	0	2	0	0
g03	10	nonlinear	0.0026%	0	0	0	1
g04	5	quadratic	27.0079%	0	6	0	0
g05	4	nonlinear	0.0000%	2	0	0	3
g06	2	nonlinear	0.0057%	0	2	0	0
g07	10	quadratic	0.0000%	3	5	0	0
g08	2	nonlinear	0.8581%	0	2	0	0
g09	7	nonlinear	0.5199%	0	4	0	0
g10	8	linear	0.0020%	3	3	0	0
g11	2	quadratic	0.0973%	0	0	0	1
g12	3	quadratic	4.7697%	0	1	0	0
g13	5	nonlinear	0.0000%	0	0	0	3

The goal of this experiment is to detect which original variant (without additional operators and complex mechanisms) is able to solve this set of test problems [13] with the best performance. The four variants chosen were: (1) GBPSO with inertia weight, (2) LBPSO with inertia weight, (3) GBPSO with constriction factor and (4) LBPSO with constriction factor.

The constraint-handling mechanism used in this work was originally proposed by Deb [14] and it consists on three feasibility rules: (1) Between 2 feasible solutions, the one with the highest fitness value wins, (2) if one solution is feasible and the other one is infeasible, the feasible solution wins and (3) if both solutions are infeasible, the one with the lowest sum of normalized constraint violation is preferred $\left(\sum_{i=1}^{m+q} max(0, g_i(\boldsymbol{x}))\right)$.

The parameter values for each PSO variant were defined as follows: 80 particles and 2000 generations (160,000 evaluations), $c_1 = 2.7$ and $c_2 = 2.5$ for all PSO variants, for the two local best variants we used 8 neighborhoods, $w = 0.7$ for both inertia weight variants and $k = 0.729$ [8] for both constriction factor variants. Equality constraints tolerance was set to ϵ =0.0001. These values were chosen empirically (by using set of independent runs per different parameter values as to find better statistical results for all variants) favoring the best performance of the PSO variants. 30 independent runs per variant per problem were performed. The statistical results are summarized in Table 2, first six columns. The best (B), mean (M) and standard deviation (SD) values are reported. Bks is the best known solution per test problem. Quality of a solution is measured by the B result, while consistency is measured by a better M and/or lower SD values. From those results, the variant Local best with constriction factor was clearly the most competitive (better statistical values on 11 test problems and similar good results just in 2 test problems). Furthermore, it was the only variant which reached the feasible region in all 13 problems. Global best variants presented premature convergence and failed to generate feasible solutions in two problems (g05 and g13). Finally, the LBPSO with inertia weight also presented convergence to local optima and failed to reach the feasible region in problem g13.

3 Proposed algorithm: Modified PSO (MPSO)

The results obtained in the previous empirical comparison showed that the LBPSO with constriction factor was the most competitive variant. Two collateral conclusions from the previous study were that (1) LBPSO performs better in constrained search spaces and (2) as other nature-inspired heuristics, PSO is very sensitive to its parameter values (it was quite difficult to fine-tune them). Therefore, a parameter control mechanism was added to the most competitive variant. Two parameters have a strong influence in the velocity update formula used in the LBPSO with constriction factor: The constriction factor k and the acceleration constant c_2 (related with the social element in the swarm i.e. the position of the leader). Thus, as to promote different behaviors in the particles of the swarm, regardless of the neighborhood they belong, a dynamic parameter control for these two parameters is proposed for a subset of particles in the swarm. Hence, at each generation, some particles will use the k and c_2 static values and the remaining ones will use dynamic values, which will be ascending as the generations go on, until

these values are the same of the static ones. The expected effect is that some particles will move at a pre-defined ratio (based on the static values) and others will move slowly at the beginning and faster as the process advances. The proposed function to dynamically adapt the values is the following: $f(y) = y^4$, where y is the number of current generation divided by $Gmax$ of generations. The expressions to modify k' and c_2' (dynamic values) are: $k' = k * f(y)$ and $c_2' = c_2 * f(y)$ (see left graph on Figure 2).

The percentage of particles that will use the dynamic values is also dynamic. An oscillatory effect provided a better performance (based on results of previous set of runs with different effects). It follows that the rate of particles that will use the dynamic values will vary between 60% and 80% during a single run based on a probability p calculated as follows: $p = k + \frac{(\sin(4.0 \times \pi \times y))}{10.3}$, where y is defined as mentioned before and k is the fixed value for the constriction factor (see right graph in Figure 2). In this modified PSO, constraints are handled with the same constraint mechanism used in the empirical comparison presented in Section 2. However, the sum of constraint violation was calculated separately for the equality and inequality constraints. Therefore, Pareto dominance [15] is used to select the best between two infeasible solutions (criterion 3). MPSO pseudocode is shown in Figure 3.

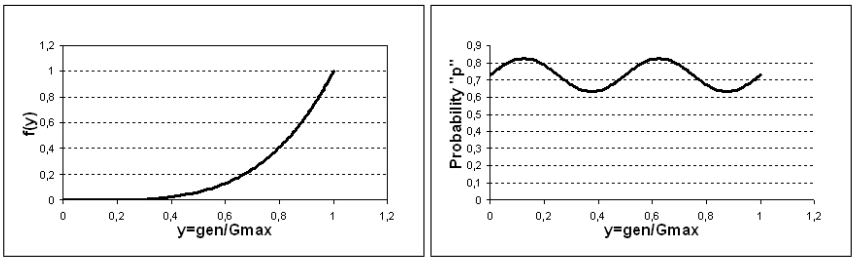

Fig. 2. Left: Behavior of the dynamic function to adapt the k' and c_2' values. y is the number of the current generation divided by $Gmax$. This behavior promotes some particles to move slowly at the beginning and quickly increasing its velocity as to equal the velocities of the remaining particles. Right: Behavior of the dynamic adaptation for the percentage of particles that uses the increasing parameter values.

4 On-line behavior

To analyze more in-depth the behavior of this modified PSO (MPSO), two performance measures were used. The first one is the number of solution evaluations required to generate the first feasible solution (EVALS) [16]. A lower value is preferred because it means a low computational cost to reach the feasible region of the search space. The second one measures the capacity of an algorithm to improve the first feasible solution i.e. it measures the ability of an algorithm to move inside the feasible region of the search space. The formula, called PROGRESS RATIO and proposed in [16] is the following: $P = |\ln \sqrt{\frac{f_{min}(G_{ff})}{f_{min}(GMAX)}}|$, where $f_{min}(G_{ff})$ is the value of the objective function of the first feasible solution found and $f_{min}(GMAX)$ is the value of the objective function of the best feasible solution at the end of the process. A higher value

indicates a better improvement inside the feasible region. The MPSO is compared with respect to two of the original variants used in the first experiment: GBPSO and LBPSO, both with constriction factor. The parameters used are the same reported in the first experiment. The only differences with respect to the MPSO is that k and c_2 are dynamically controlled as well as the percentage of particles using these dynamic values. 30 independent runs per PSO per problem per performance measure were performed. Statistical values (best, mean and standard deviation) are summarized in Table 2 (last six columns). In order to get more statistical support to the results, nonparametric statistical tests (Kruskal-Wallis and Mann-Whitney) were computed for each sample compared as to verify that the differences shown in the samples are significant (95%-level of confidence). The results of these tests showed that the differences are significant in all cases except in problems g08, g11 and g12 for EVALS and in problems g04, g08, g12 and g13 for PROGRESS RATIO.

- Begin
  - Generate a swarm of random solutions.
  - Evaluate the fitness of each particle in the swarm.
  - do
    - Select the leader(s) of the swarm **by using the modified feasibility rules**
    - For each particle, update its position with the flight formula **A subset of the swarm uses increasing parameters**
    - Evaluate the fitness of the new position of each particle.
    - Update the $pbest$ (memory) value of each particle **by using the modified feasibility rules**.
  - Until $t = Gmax$
- Stop

Fig. 3. Modified PSO, the modifications are remarked in **boldface**. GMAX is the number of generations of the algorithm.

EVALS. The GBPSO with constriction factor quickly locates the feasible region in problems g01, g06, g07 and g10, but in g13 it fails to find it. The MPSO reaches faster the feasible region in problems g03, g05 and g13. The LBPSO with constriction factor (the base variant for the MPSO) did not provide better results in any function. In the remaining problems, the behavior was very similar in the three approaches compared. The results suggest that the GBPSO is able to generate feasible solutions faster in problems with a combination of linear and nonlinear inequality constraints which define a very small feasible region with respect to the search space (problems g01, g06, g07 and g10). On the other hand, the MPSO was faster in problems with only equality constraints (g03, slightly better in g13) and problems with a combination of inequality and equality constraints (g05).

PROGRESS RATIO. The MPSO could obtain a better improvement (based on quality and consistency) inside the feasible region in problem g01. Furthermore, MPSO provided the best quality improvement in problems g09 and g10. The LBPSO with constriction factor was clearly better in problem g06 and it was more robust in problem g10. The GBPSO with constriction factor did not provide a better result in any test problem. The above results show that MPSO has slightly better capabilities to maintain a progress inside the feasible region, regardless the features of the problem being solved in this benchmark. The dynamic parameter adaptation may be useful as to promote a better exploration of the search space, and mostly of the feasible region.

Table 2. First section: Statistical results of 30 independent runs on the 13 test problems for the four PSO variants compared. Second section: Statistics of EVALS and PROGRESS RATIO on 30 independent runs on the same 13 test problems. "(n)" means that in "n" runs, out of 30, the feasible region was found. "-" means that no feasible solutions were found in any single run. In **boldface** the best obtained result is remarked.

COMPARISON OF 4 PSO VARIANTS						PERFORMANCE MEASURES					
						EVALS			PROGRESS RATIO		
Problem		global best (w=0.7)	global best (k=0.729)	local best (w=0.7)	local best (k=0.729)	global best (k)	local best (k)	MPSO	global best (k)	local best (k)	MPSO
g01	B	-14.961	-14.951	-14.999	**-15.000**	**162**	246	252	0.302	0.346	**0.368**
	M	-11.217	-11.947	-12.100	**-13.363**	**306**	368	419	0.196	0.266	**0.295**
	SD	2.482	1.813	3.055	**1.397**	64.096	**54.107**	77.781	0.059	0.050	**0.038**
g02	B	-0.655973	-0.634737	-0.614785	**-0.790982**	0	0	0	**1.388**	1.373	1.218
	M	-0.606774	-0.559591	-0.543933	**-0.707470**	0	0	0	0.884	**1.015**	1.013
	SD	0.026463	0.030346	**0.020048**	0.059237	0.000	0.000	0.000	0.123	0.107	**0.099**
g03	B	-0.080	-0.019	-0.045	**-0.126**	**189**	366	457	0.346	0.346	0.334
	M	-9.722E-03	-1.721E-03	-0.010	**-0.017**	3568	2118	**1891**	0.037	0.026	**0.067**
	SD	0.016	**4.642E-03**	0.012	0.027	3509.766	1354.926	**982.427**	**0.065**	0.725	0.081
g04	B	-30655.331	-30665.439	-30665.539	-30665.539	0	0	0	0.110	0.120	**0.124**
	M	-30664.613	-30664.606	-30665.539	-30665.539	4	2	2	0.070	**0.080**	0.071
	SD	0.573	0.549	7.400E-012	7.400E-012	5.149	3.115	**2.921**	0.023	0.023	0.025
g05	B	-	-	5126.646 (18)	**5126.496**	33459 (12)	16845	**13087**	4.273E-07 (12)	0.087	0.087
	M	-	-	6057.259	**5140.060**	86809	23776	**17037**	1.250E-07	**0.056**	0.036
	SD	-	-	232.251	**15.525**	45359.759	3683.613	**2211.676**	**1.643E-07**	0.037	0.032
g06	B	-6959.517	-6959.926	-6958.704	**-6961.814**	**180**	256	254	0.799	**0.807**	0.772
	M	-6948.937	-6948.121	-6941.207	**-6961.814**	**440**	513	562	0.306	**0.348**	0.296
	SD	6.312	6.415	9.059	2.679E-04	273.858	258.324	**186.856**	0.207	**0.188**	0.193
g07	B	43.731	38.916	41.747	**24.444**	**178**	484	812	2.117	**2.504**	2.499
	M	68.394	64.186	59.077	**25.188**	**873**	1164	1316	1.656	1.919	**1.963**
	SD	40.698	17.156	7.653	**0.599**	711.998	401.181	**252.411**	0.363	0.369	**0.350**
g08	B	-0.095825	-0.095825	-0.095825	-0.095825	4	**0**	3	0.494	0.451	**0.556**
	M	-0.095824	-0.095825	-0.095825	-0.095825	**56**	78	76	0.317	0.304	**0.356**
	SD	1.751E-07	7.258E-08	4.234E-17	4.234E-17	**39.709**	52.159	57.963	0.091	0.092	**0.072**
g09	B	692.852	693.878	696.947	**680.637**	**5**	8	17	4.685	4.394	**4.768**
	M	713.650	708.274	728.730	**680.671**	**88**	94	107	**2.622**	2.209	2.510
	SD	12.969	10.155	15.804	**0.021**	50.144	**48.131**	69.187	1.298	**1.126**	1.246
g10	B	8024.273	8769.477	8947.646	**7097.001**	**242**	579	522	0.598	0.665	**0.678**
	M	8931.263	9243.752	9247.134	**7641.849**	**861**	972	1202	0.360	**0.482**	0.468
	SD	390.577	229.449	**184.691**	361.366	329.899	**269.375**	456.381	0.138	**0.107**	0.117
g11	B	0.749	0.749	0.750	**0.749**	**85**	249	364	0.143	0.143	0.143
	M	0.752	0.755	0.799	**0.749**	1662	1152	**1009**	0.088	**0.113**	0.101
	SD	9.273E-03	0.014	0.057	**1.999E-03**	2101.228	832.708	**572.870**	**0.047**	0.049	0.052
g12	B	-0.999	-0.999	-0.999	**-1.000**	2	**0**	1	**0.342**	0.281	0.285
	M	-0.999	-0.999	-0.999	**-1.000**	19	**15**	25	**0.136**	0.119	0.096
	SD	6.967E-07	5.137E-07	2.596E-05	**0.000**	**16.012**	17.784	21.712	0.072	**0.067**	0.074
g13	B	-	-	-	0.081	-	11497	**8402**	-	1.165	**2.327**
	M	-	-	-	0.454	-	17553	**12874**	-	0.410	**0.549**
	SD	-	-	-	0.258	-	2820.497	**1826.789**	-	0.346	**0.576**

Based on the overall obtained results, the GBPSO with constriction factor is faster to reach the feasible region, but fails in some problems. However, MPSO is more consistent and in presence of equality constraints, MPSO is more competitive. Furthermore, the dynamic parameter control added to MPSO was beneficial because the original LBPSO with constriction factor was unable to reach the feasible region faster in all test problems. Finally, the ability of PSO to move inside the feasible region was improved in MPSO. Another interesting aspect here is that local best PSOs (both MPSO and LBPSO) were more competitive to move inside the feasible region than GBPSO.

As a final experiment, a comparison against state-of-the-art PSO-based approaches is presented in Table 3. Best (B), Mean (M) and worst (W) results from a set of 30 independent runs by MPSO, using the same parameter values of the previous experiments are presented. The results are compared against: A) Toscano & Coello PSO [10], B) Lu & Chen PSO [13] and C) Cagnina et al. PSO [12]. The results used in this experiment were obtained from their original papers i.e. we conducted an indirect comparison.

The performance of MPSO is more consistent with respect to the results of the compared approaches in problems g05 and g13, both with nonlinear equality constraints. Problem g05 has a combination of equality and inequality constraints. Thus, it seems that the separation of sums of constraint violation provides more specific information to guide the search. However, this statement requires further studies. MPSO reached consistently the best known solution in five problems: g01, g04, g06, g08 and g12 and was very competitive in five more: g02, g07, g09, g10, g11. The test function where MPSO did not provide competitive results was problem g03 (ten decision variables and one nonlinear equality constraint). It seems that the combination of high dimensionality with equality constraints may lead to a premature convergence of the approach. The results of this last experiment show that MPSO provides a competitive performance against PSO-based state-of-the-art approaches. It is worth mentioning that MPSO does not use additional operators and/or parameters like the compared approaches and the number of evaluations required to reach these results was 160,000, compared to 340,000 required by Toscano & Coello [10] and Cagnina et al. [12], only above the 50,000 required by Lu & Chen [13] where an additional parameter is added to the flight formula and there is no clear evidence about its fine-tuning.

5 Conclusions and future work.

In this paper, an analysis of the behavior of PSO in constrained search spaces was presented. Four original variants were compared in a well-known benchmark by using a simple parameterless constraint-handling technique. Based on statistical results, the most competitive variant was detected (Local best with constriction factor). A dynamic parameter control mechanism was added to update two PSO parameters (k and c_2) and the constraint-handling mechanism was improved in its third criterion by using separate sums of constraint violation for equality and inequality constraints. This modified PSO (MPSO) was compared with two original PSO variants using two performance measures as to analyze how fast they reach the feasible region and also their ability to move inside it. Finally, the MPSO was compared with three state-of-the-art PSO-based approaches for constrained optimization. The study of PSO original variants, as a first

Table 3. Statistical results of MPSO and PSO-based approaches. **Boldface** remarks the best result per function. The results with a "$*$" seem to be infeasible, but they could not be verified in their original sources.

STATE-OF-THE-ART ALGORITHMS AND MPSO					
Problem & Bks.		Toscano & Coello [10]	Lu & Chen [13]	Cagnina et al. [12]	MPSO
g01	B	**-15.000**	-15.000	-15.000	**-15.000**
-15.000	M	**-15.000**	-14.418	-15.000	**-15.000**
	W	**-15.000**	-12.453	*-134.219	**-15.000**
g02	B	**-0.803432**	-0.664	-0.801	-0.802629
-0.803619	M	**-0.790406**	-0.413	0.765	-0.713879
	W	**-0.750393**	-0.259	0.091	-0.600415
g03	B	-1.004	-1.005	**-1.000**	-0.641
-1.000	M	-1.003	-1.002	**-1.000**	-0.154
	W	-1.002	-0.934	**-1.000**	-3.747E-03
g04	B	-30665.500	**-30665.539**	-30665.659	**-30665.539**
-30665.539	M	-30665.500	**-30665.539**	-30665.656	**-30665.539**
	W	-30665.500	**-30665.539**	-25555.626	**-30665.539**
g05	B	5126.640	**5126.484**	5126.497	5126.498
5126.498	M	5461.081	5241.054	5327.956	**5135.521**
	W	6104.750	5708.225	*2300.5443	**5169.191**
g06	B	-6961.810	-6961.813	-6961.825	**-6961.814**
-6961.814	M	-6961.810	-6961.813	-6859.075	**-6961.814**
	W	-6961.810	-6961.813	64827.544	**-6961.814**
g07	B	24.351	**24.306**	24.400	24.366
24.306	M	25.355	**24.317**	31.485	24.691
	W	27.316	**24.385**	4063.525	25.15
g08	B	**-0.095825**	**-0.095825**	-0.095825	**-0.095825**
-0.095825	M	**-0.095825**	**-0.095825**	-0.095800	**-0.095825**
	W	**-0.095825**	**-0.095825**	-0.000600	**-0.095825**
g09	B	680.638	**680.630**	680.636	680.638
680.630	M	680.852	**680.630**	682.397	680.674
	W	681.553	**680.630**	18484.759	680.782
g10	B	7057.900	**7049.248**	7052.852	7053.963
7049.248	M	7560.047	**7049.271**	8533.699	7306.466
	W	8104.310	**7049.596**	13123.465	7825.478
g11	B	0.749	**0.749**	0.749	0.749
0.749	M	0.750	**0.749**	0.750	0.753
	W	0.752	**0.749**	0.446	0.776
g12	B	**-1.000**	**-1.000**	-1.000	**-1.000**
-1.000	M	**-1.000**	**-1.000**	-1.000	**-1.000**
	W	**-1.000**	**-1.000**	9386	**-1.000**
g13	B	0.068	**0.053**	0.054	0.066
0.053949	M	1.716	0.681	0.967	**0.430**
	W	13.669	2.042	1.413	**0.948**

step of design, allowed to get a PSO-based approach to solve constrained optimization problems without adding extra parameters or other mechanisms which may eliminate PSO simplicity. The performance measures used in the second experiment showed that MPSO is faster to reach the feasible region of the search space in presence of equality constraints and also that its ability to improve feasible solutions previously found was enhanced. Finally, the performance of MPSO was very competitive against algorithms representative of the state-of-the-art.

The future work consists on testing MPSO in more problems with a combination of equality and inequality constraints and to solve engineering design problems.

References

1. Eiben, A., Smith, J.E.: Introduction to Evolutionary Computing. Springer (2003)
2. Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence. Wiley (2006)
3. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man and Cybernetics-Part B **26** (1996) 29–41
4. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6** (2002) 58–73
5. Michalewicz, Z., Schoenauer, M.: Evolutionary Algorithms for Constrained Parameter Optimization Problems. Evolutionary Computation **4** (1996) 1–32
6. Parsopoulos, K., Vrahatis, M.: Unified Particle Swarm Optimization for solving constrained engineering optimization problems. Advances in Natural Computation, Pt. 3 (2005) 582–591 Lecture Notes in Computer Science Vol. 3612.
7. Li, X., Tian, P., Kong, M.: Novel particle swarm optimization for constrained optimization problems. In: Proceedings of the Australian Conference on Artificial Intelligence, Heidelberg, Germany, Springer-Verlag (2005) 1305–1310 LNAI No. 3809.
8. Krohling, R.A., dos Santos Coelho, L.: Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems. IEEE Transactions on Systems, Man and Cybernetics Part B **36** (2006) 1407–1416
9. He, S., Prempain, E., Q.H.Wu: An Improved Particle Swarm Optimizer for Mechanical Design Optimization Problems. Engineering Optimization **36** (2004) 585–605
10. Toscano-Pulido, G., Coello Coello, C.A.: A Constraint-Handling Mechanism for Particle Swarm Optimization. In: Proceedings of the Congress on Evolutionary Computation 2004. Volume 2., Piscataway, New Jersey, IEEE Service Center (2004) 1396–1403
11. Liang, J.J., Suganthan, P.N.: Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constrain-Handling Mechanism. In: 2006 IEEE Congress on Evolutionary Computation (CEC'2006), Vancouver, BC, Canada, IEEE (2006) 316–323
12. Cagnina, L.C., Esquivel, S.C., Coello, C.A.C.: A Particle Swarm Optimizer for Constrained Numerical Optimization. In: Parallel Problem Solving from Nature (PPSN IX), Reykjavik, Iceland, Springer (2006) 910–919 LNCS Vol. 4193.
13. Lu, H., Chen, W.: Dynamic-objective particle swarm optimization for constrained optimization problems. Journal of Combinatorial Optimization **12** (2006) 409–419
14. Deb, K.: An Efficient Constraint Handling Method for Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering **186** (2000) 311–338
15. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Wiley (2001)
16. Mezura-Montes, E., Coello, C.A.C.: Identifying On-line Behavior and Some Sources of Difficulty in Two Competitive Approaches for Constrained Optimization. In: 2005 IEEE Congress on Evolutionary Computation. Volume 2., IEEE Press (2005) 1477–1484