Robust PSO-Based Constrained Optimization by Perturbing the Particle's Memory

Angel Muñoz Zavala, Arturo Hernández Aguirre
and Enrique Villa Diharce
Centro de Investigación en Matemáticas (CIMAT)
México

1. Introduction

A successful evolutionary algorithm is one with the proper balance between exploration (searching for good solutions), and exploitation (refining the solutions by combining information gathered during the exploration phase). Diversity maintenance is important in constrained search space algorithms because the additional pressure set on the population to reach the feasible region reduces the diversity. Since reduced diversity promotes premature convergence, new exploration and exploitation techniques have been incorporated into the PSO main paradigm.

In this chapter the authors review the standard PSO algorithm, and several proposals to improve both exploration and exploitation: local and global topologies, particle motion equations, swarm neighbourhoods, and interaction models. For all these approaches the common shared feature is the modification of the PSO main algorithm.

The present chapter, however, describes a rather different approach: the perturbation of the particle memory. In the PSO algorithm, the next particle's position is based on their flying experience (*pbest)*, and the current best individual in either the entire swarm (*gbest*), or in a swarm neighbourhood (*lbest*). Since the values for *gbest* or *lbest* are determined from the pbest values available at any generation, in the end, it is the *pbest* which is mainly responsible for the particle's next position. Therefore, a way to reduce premature convergence is to improve the pbest of each particle.

Our approach aims to prevent convergence to local optima by improving the swarm exploration and exploitation through two perturbation operators. These external operators improve the memory of the best visited locations, and do not modify the main PSO paradigm.

The rest of this Chapter is organized as follows: In Section 2, we introduce the premature convergence problem. We extend this discussion in the context of constrained optimization, in Section 3. Our approach is introduced in Section 4; giving a brief explanation about every component adopted in the PSO algorithm. In Section 5, a well-known benchmark is used to compare our approach against other PSO based methods and evolutionary algorithms representative of the state-of-the-art. The conclusion is given in Section 6, complemented with future work guidelines in Section 7.

Source: Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, Book edited by: Felix T. S. Chan and Manoj Kumar Tiwari, ISBN 978-3-902613-09-7, pp. 532, December 2007, Itech Education and Publishing, Vienna, Austria

2. Premature Convergence

A natural problem in evolutionary computation is the premature convergence. It means that the evolutionary algorithm could stay trapped in a region containing a local optimum. Premature convergence can be caused by the lost of diversity, which occurs when the population reaches a suboptimal state where evolutionary algorithm can no longer produce offspring which outperforms their parents (Fogel, 1994). A way to attain diversity maintenance is by keeping a balance between exploration and exploitation (Holland, 1975).

2.1 Exploration and Exploitation

The balance between exploration and exploitation is a well known issue in evolutionary computation (Michalwicz & Fogel, 2000). But, why is it hard to find an optimal balance? First, we review the classical concepts: exploration *is the act of searching for the purpose of discovery*; and exploitation *is the act of utilizing something for any purpose* (Agnes, 2004). Then, at the context of evolutionary computation, we could define exploration *as the act of searching through all space regions for discovering promissory solutions*; and exploitation *as the act of utilizing local information for refining the solution.*

Accord with the given definition, Downing explained: *exploitation is encouraged by elitist selection and smaller population sizes or by using lower mutation rates to promote correlation between parent and offspring. Conversely, exploration is encouraged by promoting greater population diversity and selecting parents less discerningly, or by increasing mutation rate.* (Downing, 2006).

We can observe that exploration and exploitation are opposite goals; both compete for limited resources, for instance, the number of fitness function evaluations. So, a trade-off between exploration and exploitation is necessary. The search horizon has to be sufficiently close for maintaining exploitation and at the same time sufficiently distant to discover significant novelty (Jacoby, 2005).

The common approach in evolutionary computation about the control of exploration and exploitation is framed in terms of balancing variation and selection processes. There are several works focused to solve the equilibrium dilemma between variation and selection. Even more, there are proposals from other areas (e.g. management, economics) for solving related problems (March, 1991). In the following sections, we review some approaches and propose a new solution for balancing variation and selection in the PSO algorithm.

2.2 Diversity Control in PSO

In PSO, the diversity comes from two sources. One is the difference between the particle's current position and its best neighbor, and the other is the difference between the particle's current position and its best historical value. Although variation provides exploration, it can only be sustained for a limited number of generations because convergence of the flock to the best is necessary to refine the solution (exploitation).

In an early analysis, Angeline shows that PSO may not converge, neither refine solutions when variation is null, that is, when all the particles rest near by the best spot (Angeline, 1998). A few months after Angeline's work, the first formal analysis of a simple PSO was developed by Ozcan and Mohan (Ozcan & Mohan, 1998), which obtained the PSO trajectories. Based on this work, Clerc and Kennedy analyzed a particle's trajectory and determined the relationship between the acceleration parameters that avoid the divergence of the particle (Clerc & Kennedy, 2002). But, when the problem of converge premature seemed solved; Van Den Bergh proves that the PSO trajectories does not converge to the global optimal (Van Den Bergh, 2002). In his Ph.D. thesis, Van Den Bergh explains the attributes that a hybrid PSO must accomplish to become a global search algorithm. There are several proposes related with developing a global search based on the PSO algorithm. In this context, we can find approaches for dealing with constraints, which is the topic of this chapter.

3 Constraint-Handling in PSO

Real optimization problems are subject to a number of equality and inequality constraints, which can be linear or nonlinear. These constraints determine which areas of the search space are feasible and which are infeasible. In addition to these constraints, boundary constraints are usually imposed to the search space (Michalewicz, 1992). Also, there is the possibility that the feasible space is fragmented and separated by infeasible regions, requiring that both the feasible and infeasible regions be searched.

PSO is an unconstrained search technique. Thus, adopting a constraint handling technique into the main PSO algorithm is an open research area. There is a considerable amount of research regarding mechanisms that allow the evolutionary algorithms to deal with equality and inequality constraints. Some constraint-handling approaches tend to incorporate either information about infeasibility or distance to the feasible region, into the fitness function in order to guide the search. These techniques are based on *penalty functions* (Parsopoulos & Vrahatis, 2002). In their work Parsopoulos and Vrahatis used a multi-stage assignment penalty function without diversity control. Other approaches propose a constraint handling technique based on maintaining a feasible population (EI-Gallad et al., 2001), and also some algorithms require a feasible initial population (Hu & Eberhart, 2002; He et al., 2004). In 2003, Coath and Halgamuge presented a comparison of the two constraint-handling methods in PSO: *penalty function* and *feasibility preservation* (Coath & Halgamuge, 2003). Their experiments clearly detect the need of some form of diversity control.

In a more sophisticated approach, Zhang et al. introduced a special technique, called *periodic mode*, to handle inequality constraints. This method consists in keeping the global-best near the boundary thus the flock which is constantly pulled to the border, can sustain exploration (Zhang et al., 2004). A few more sophisticated approaches include applying multi-objective optimization techniques to handle constraints. For instance Toscano and Coello (Toscano & Coello, 2004), use a *feasibility tournament* proposed by Deb (Deb 2000) to handle constraints with PSO. The *feasibility tournament* applies a set of rules similar to the Pareto dominance concept used in multi-objective optimization.

Notably, equality and inequality constraints demand an intelligence exploration of the search space to find the global optimum region. Likewise, an efficient and effective exploitation is required in the boundaries of the feasible region, whenever the inequality constraints are active or equality constraints are present. PSO should find a solution that both optimizes the objective function and satisfies all constraints.

4. Constrained Optimization via Particle Swarm Optimization

A brief analysis of the state-of-the-art in PSO to solve constrained optimization problems was presented. Now, we are going to introduce our approach called Particle Evolutionary Swarm Optimization (PESO) (Muñoz et al., 2005). In this section we explain our approach;

and in the next section we perform a comparison with another PSO-based constraint optimization works.

4.1 Interaction Model

First we should choose an appropriate interaction model for solving constrained optimization problems. In an early analysis, Kennedy provided empirically evidence that the *social-only* model is faster and more efficient than the *full* and *cognitive-only* models (Kennedy, 1997). These models were defined by omitting components of the velocity formula. The *full* model is composed by the cognition component and the social component. Dropping the social component results in the *cognition-only* model, whereas dropping the cognition component defines the *social-only* model. In a fourth model, *selfless* model, the neighbourhood best is chosen only from the neighbours, without considering the current individual. Carlisle and Dozier tested these four models in dynamic changing environments (Carlisle & Dozier, 2000). They empirically prove that the *social-only* model consistently found solutions faster than the *full* model, but the reliability of the *social-only* model is lower than the *full* model.

We test the four models proposed by Kennedy, (Kennedy, 1997). We confirm that the *social-model* is faster than the *full* model, but it is not enough robust due to its premature convergence behaviour. Therefore, we adopt the *full* model which is more reliable for constrained optimization.

4.2 Social Network Structure

In the PSO topology, each particle moves following a leader; this fact is modelled by one of three components of the velocity formula. A leader can be global to all the flock, or local to a flock's neighbourhood. In the latter case there are as many local leaders as neighbourhoods. Having more than one leader in the flock translates into more attractors or good spots in space. Therefore, the use of neighbourhoods is a natural approach to fight premature convergence (Mendes et al., 2004).

Particles in the same neighbourhood communicate with one another by exchanging information for moving towards a better position. The flow of information through the flock, depends on the neighbourhood structure. Figure 1 presents a few neighbourhood structures developed for PSO.

In a highly connected neighbourhood structure, the information about the best particle in the swarm is quickly transmitted through the whole flock. This means faster convergence, which implies a higher risk to converge to a local minimum. Also, Kennedy & Mendes empirically shows that the *star neighbourhood* is faster than the other topologies, but it meets the optimal fewer times than any other one (Kennedy & Mendes, 2002). They suggest trying the *Von Neumann neighbourhood* structure, which performed more consistently in their experiments than the topologies commonly found in current practice. However, in the experiments developed by Kennedy & Mendes, they used a set of unconstrained optimization problems. However, based on their recommendation, we propose a new neighbourhood structure, which we define as *singly-linked ring*.

The *singly-linked ring* rises from analysing the *ring neighbourhood* as a double-linked list; like it is showed in Figure 2-a. Suppose that every particle is assigned a permanent label which is used to construct the neighbourhoods. Then, a particle ***k*** has two neighbours, particles ***k-1*** and ***k+1***. In turn, particles ***k-1*** and ***k+1*** have particle ***k*** as a neighbour. In this way, there is a mutual attraction between consecutive particles, forming overlapped clusters. Also, the slow convergence of the ring structure has been empirically showed (Kennedy, 1999; Kennedy & Mendes, 2002; Carlisle & Dozier, 2000).

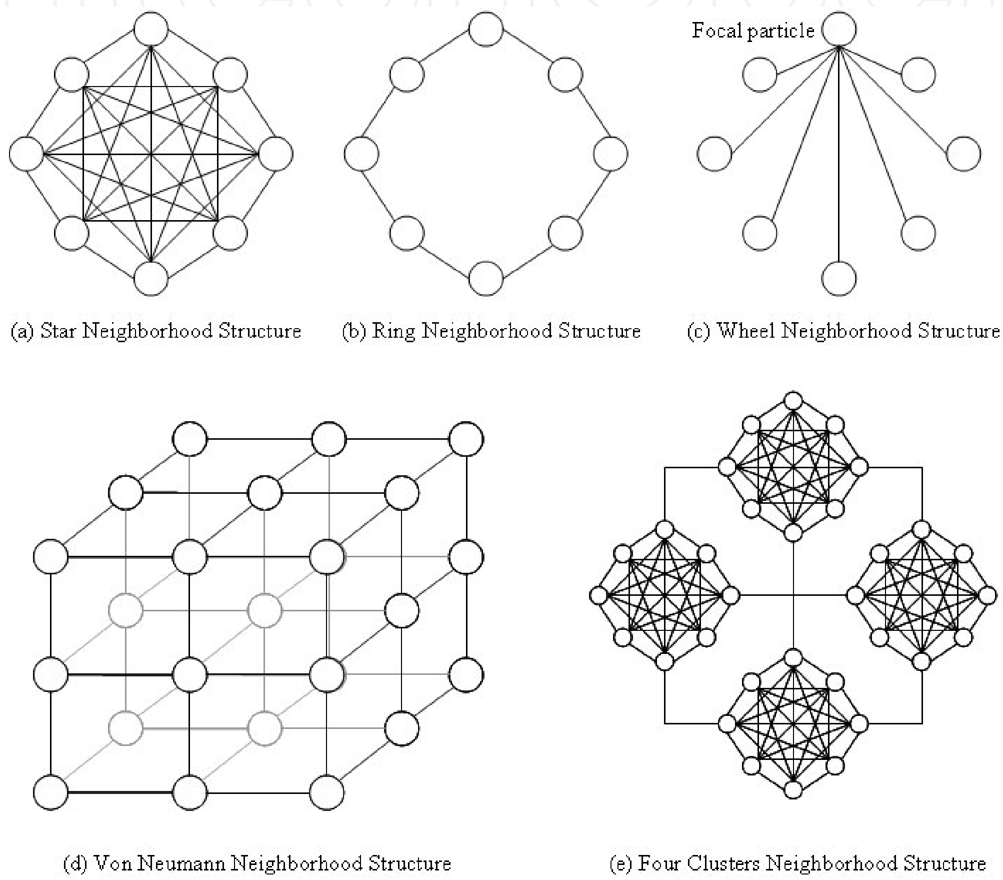


Figure 1. Neighbourhood Structures. A representation of the social networks applied in PSO

The successful of the *Von Neumann neighbourhood* is due to the interaction that each particle has with other particles, an average of 5 neighbours. This promotes the exploitation, but unfortunately fails to provide the exploration required by the constrained optimization problems. Thus, we propose the topology presented in Figure 2-b. The *singly-linked ring* keeps two neighbours for each particle, but breaks the mutual attraction between neighbours. Besides, the information through the whole swarm is transmitted faster than in the original *ring* topology. Therefore, the *singly-linked ring* keeps the exploration at the search space, and increases the exploitation of the best solutions (Hernández et al., 2007).

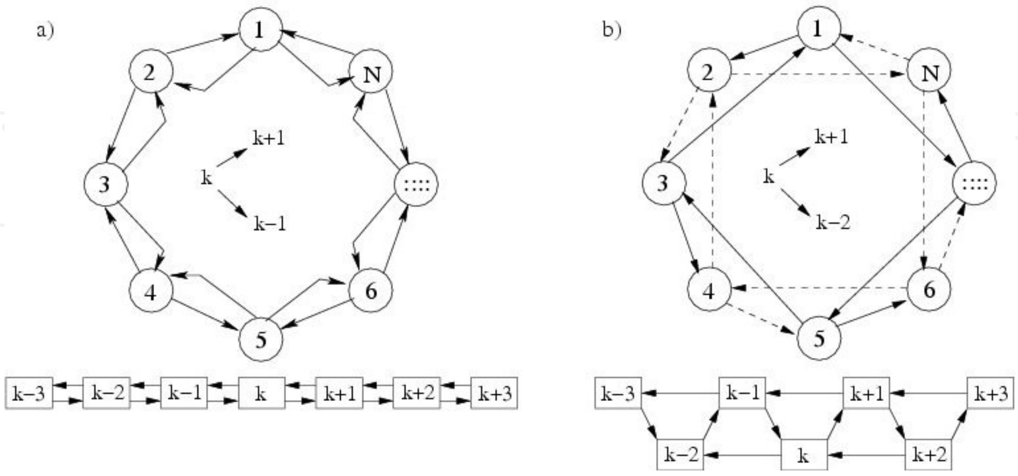


Figure 2. Ring Neighbourhood Structures. a) Original ring topology, b) Singly-Linked ring

4.3 Feasibility Tournament

The first step for developing a PSO-based constraint optimization is to choose a constraint-handling method. There are three methods that have been used by other approaches: *penalty function*, *feasibility preservation* and *feasibility tournament*.

The *penalty function* method involves a number of parameters which must be set right in any problem to obtain good solutions. This fact has motivated sophisticated penalty function approaches and extensive experimentation for setting up appropriate parameters (Michalewicz & Schoenauer, 1996). According with Deb (Deb, 2000), there are two problems associated with the *static penalty function*: the optimal solution depends on penalty parameters and the inclusion of the penalty term distorts the objective function.

Now, we give the details of the *feasibility preservation* method. There are two main problems associated with this method: it needs a feasible initial population and special operators to keep the population into the feasible region. Also, the method could be unreliable handling problems with active constraints, since it does not allow unfeasible solutions and has not information about the boundaries.

The *feasibility tournament* proposes to use a tournament selection operator, where two solutions are compared at time, and the following criteria are always applied:

1. Any feasible solution is preferred to any infeasible solution.
2. Among two feasible solutions, the one having better objective function value is preferred.
3. Among two infeasible solutions, the one having smaller sum of constraint violation is preferred.

The *feasibility tournament* does not require tuning parameters or applying special operators. Just a simple comparison is used to choose the best individual. Even, in any of the above three scenarios, solutions are never compared in terms of both objective function and sum of constraint violation. This method was implemented by Toscano and Coello in a PSO with global topology, obtaining competitive results (Toscano & Coello, 2004). Our approach applies this method in a local topology, allowing feasible and infeasible solutions in the *pbest* particles. It enriches the information about the search space, especially at boundaries. Nevertheless, for handling equality constraints, it is not enough just converting them into inequality constraints:

$$g(x) = \delta - |h(x)| \tag{1}$$

Our approach applies a *dynamic tolerance* for handling equality constraints. First, we rewrite them as inequality constraints of the form $|h(x)| \leq \delta$, where δ is called the tolerance. Then, the tolerance is linearly decremented from 1.0 to a specified target value (1E-06 in our experiments) during the first 90% of function evaluations. For the last 10% the tolerance is kept fixed; thus, the particles have additional time to achieve convergence. This technique proved to be very effective in the test problems that we present in the Section 5.

4.4 Perturbing the PSO Memory

In Section 2, we mention the Van Den Bergh's PhD thesis and his contributions in the PSO context. He gives a set of requirements that an evolutionary algorithm must accomplish to be a global search algorithm. Also, he shows that the PSO algorithm is not in fact a global search algorithm (Van Den Bergh, 2002). Nevertheless, Van Den Bergh gives a theorem which specifies under which conditions an algorithm can be considered a global optimization method. The theorem implies that a general algorithm, without *a priori* knowledge, must be able to generate an infinite number of samples distributed throughout the whole of S in order to guarantee that it will find the global optimum with asymptotic probability 1 (Van Den Bergh, 2002).

This can be achieved by periodically adding randomised particles to the swarm. Nevertheless, resetting the position of the particles is not a trivial task; a bad decision affects directly in the exploitation of the best solutions. We propose, based on the observation that the *pbest* particles drive the swarm, perturbing the *pbest* of each particle.

Our approach has three stages. In the first stage, an iteration of the standard PSO algorithm with the features described in this Section 4 is applied. Then the perturbations are applied to *pbest* in the next two stages. The goal of the second stage is to add a perturbation generated from the linear combination of three different particles for every dimension. This perturbation is preferred over other operators because it preserves the distribution of the population. This operator is used for reproduction by the Differential Evolution algorithm (Price et al., 2005). In our approach this perturbation is called *C-Perturbation*. It is applied to the members of *pbest* to yield a set of temporal particles *tempC*. Then each member of *tempC* is compared with its corresponding father and *pbest* is updated applying the feasibility tournament. Figure 3 shows the pseudo-code of the *C-Perturbation* operator.

In the third stage every vector is perturbed again so a particle could be deviated from its current direction as responding to external, maybe more promissory, stimuli. This perturbation is implemented by adding small random numbers to every design variable. The perturbation, called M-Perturbation, is applied to every member of *pbest* to yield a set of temporal particles *tempM*. Then each member of *tempM* is compared with its corresponding father and *pbest* is updated applying the feasibility tournament. Figure 4 shows the pseudo-code of the *M-Perturbation* operator, where *LL* and *UL* are the lower and upper limits of the search space. The perturbation is added to every dimension of the decision vector with probability **1/d,** where **d** is the dimension of the decision variable vector.

```
For  k = 0  To  n
  For  j = 0  To  d
    r = U ( 0 , 1 )
    p1 = k
    p2 = random ( n )
    p3 = random ( n )
    Temp [ k , j ] = P_{i+1} [ p1 , j ] + r * ( P_{i+1} [ p2 , j ] - P_{i+1} [ p3 , j ] )
  End For
End For
```

Figure 3. C-Perturbation Operator. Pseudo-code of the C-Perturbation applies by PESO

These perturbations have the additional advantage of keeping the self-organization potential of the flock since they only work on the *pbest* particles, as we can observe in Figure 5. The current PSO population is not perturbed, but few or may be several *pbest* particles have been moved to a better position. This is an improved form for adding randomised particles to the swarm, compared with those propose by Van Den Bergh (Van Den Bergh, 2002). Our approach not only resets the position of the *pbest* particles, also improves them; probably driving the swarm to a promise region.

```
For  k = 0  To  n
  For  j = 0  To  d
    r = U ( 0 , 1 )
    If  r ≤ 1/d  Then
      Temp [ k , j ] = Random ( LL , UL )
    Else
      Temp [ k , j ] = P_{i+1} [ k , j ]
  End For
End For
```

Figure 4. M-Perturbation Operator. Pseudo-code of the M-Perturbation applies by PESO

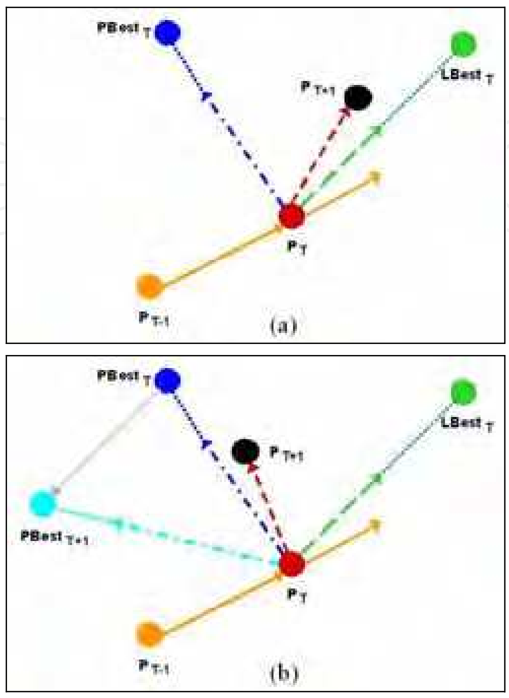


Figure 5. Effects of the perturbation operators. (a) Shows the movement of particle at iteration T+1 without perturbing the PBest in the iteration T. (b) Shows the movement of particle at iteration T+1 with the influence of PBest at iteration T+1, which is the final position after applying the perturbation operators on the PBest in the iteration T

4.5 PSO Parameters

The parameters play an important roll in the successful of any evolutionary algorithm. There are several works that discuss a number of control parameters like swarm size (Van Den Bergh, 2001), neighbourhood size (Suganthan, 1999), or acceleration coefficients (Ratnaweera et al., 2002a; Ratnaweera et al., 2002b).

Our approach generally uses a swarm size of n=100 particles, a neighbourhood size of k=2, and the following set of acceleration coefficients: w=U(0.5, 1.0), c1=1.0, c2=1.0, where U is a uniform distribution. These parameters have not been deeply studied; only the neighbourhood size, which has been explained in this Section. Nevertheless, the acceleration coefficients accomplish the mathematical model gave by Clerc & Kennedy to avoid divergence of the particle trajectories (Clerc & Kennedy, 2002).

4.6 Our Approach PESO

In summary the proposed algorithm, PESO, is a local PSO with a *singly-linked* ring neighbourhood. PESO handles constraints adopting a *feasibility tournament* complemented with a *dynamic tolerance* for handling equality constraints. The main components of PESO are the *C-Perturbation* and *M-Perturbation* operators applied to the *pbest* population.

5. Experiments

PESO is applied to solve the benchmark used in the Special Session on Constrained Real-Parameter Optimization, CEC-06 (Liang et al., 2006). The benchmark is an extended version of 24 functions from the original benchmark of Runnarson and Yao, with 13 functions (Runnarson & Yao, 2000). It is integrated by linear and non-linear functions with linear and non-linear constraints. The benchmark was proposed by Mezura, in his Ph.D. thesis (Mezura, 2004).

TP	Optimal	Best	Median	Mean	Worst	S.D.
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	0
g02	-0.803619	-0.803619	-0.803617	-0.801320	-0.786566	4.59E-03
g03	-1.000000	-1.000005	-1.000005	-1.000005	-1.000003	3.15E-07
g04	-30665.538	-30665.53867	-30665.53867	-30665.53867	-30665.53867	0
g05	5126.49811	5126.498096	5126.498096	5126.498096	5126.498096	0
g06	-6961.8138	-6961.813876	-6961.813876	-6961.813876	-6961.813876	0
g07	24.306209	24.306209	24.306210	24.306212	24.306219	3.34E-06
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0
g09	680.630057	680.630057	680.630057	680.630057	680.630057	0
g10	7049.248	7049.248020	7049.248638	7049.250087	7049.263662	3.61E-03
g11	0.750000	0.749999	0.749999	0.749999	0.749999	0
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	0
g13	0.053950	0.053950	0.053950	0.053950	0.053965	2.76E-06
g14	-47.764411	-47.761108	-47.747212	-47.741430	-47.670921	2.15E-02
g15	961.715172	961.715171	961.715171	961.715171	961.715171	0
g16	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155	0
g17	8876.98068	8856.502344	8863.875542	8877.812811	8941.344349	30.1195
g18	-0.8660	-0.866025	-0.866025	-0.866001	-0.865568	8.74E-05
g19	32.386	32.349645	32.386872	32.411596	32.571543	6.30E-02
g20	0.096737	*0.204095	*0.209711	*0.212003	*0.233281	6.94E-03
g21	193.778349	205.852693	279.309106	273.298016	303.454837	23.8556
g22	382.902205	*157.5136	*3161.1026	*5376.2265	*18732.7838	5.01E+03
g23	-400.0025	-361.856637	-136.564268	-138.407772	3.775736	84.5217
g24	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013	0

Table 1. PESO results in the Benchmark. *Infeasible Solution

5.1 PESO Results

In Table 1, we present the results of PESO in the benchmark problems, where *S.D.* means standard deviation. For every test problem 30 runs was developed, and in each run 350,000 fitness function evaluations were applied to test our. In only two test problems, g20 and g22, PESO did not find a feasible solution. These problems have 14 and 19, equality constraints respectively. Also, PESO presents a poor performance in test problems g21 and g23, where it did not reach the optimal value, but always found a feasible solution at the 30 runs. In the rest of the benchmark, PESO attains the global optimal. PESO was able to outperform the best know solution in test problems g03, g05, g11, g13, g17 and g19, due the conversion of equality constraints to inequality constraints with a tolerance value of 1E-06. The whole benchmark was resolved using the same parameters, but there are several test problems, which were solved with less than 350000 fitness function evaluations. This fact is showed in Table 2.

TP	Best	Median	Mean	Worst	S.D.	F.R.	S.R.
g01	90800	95000	95396.67	99400	2613.29	30	30
g02	142900	175800	179395.45	232100	28120.18	30	22
g03	315100	315100	315123.33	315600	97.14	30	30
g04	59600	65100	65086.67	70000	2713.28	30	30
g05	315100	315100	315256.67	315900	245.91	30	30
g06	47100	54200	53410.00	57000	2577.80	30	30
g07	185500	227600	233400.00	304500	32253.97	30	30
g08	3600	6850	6470.00	8500	1381.94	30	30
g09	69900	78500	79570.00	102400	7154.65	30	30
g10	167200	221300	224740.00	307200	38407.87	30	30
g11	315100	315100	315100.00	315100	0	30	30
g12	400	6900	6646.67	10400	2606.98	30	30
g13	315100	315150	315546.67	318100	710.87	30	30
g14	326900	326900	326900.00	326900	0	30	1
g15	315100	315100	315100.00	315100	0	30	30
g16	37200	41000	40960.00	45400	2210.88	30	30
g17	315100	316100	316608.70	318800	1061.69	30	23
g18	102200	153600	167088.89	252900	43430.30	30	27
g19	206800	259650	264414.29	331000	36456.84	30	14
g20	NR	NR	NR	NR	NR	0	0
g21	NR	NR	NR	NR	NR	30	0
g22	NR	NR	NR	NR	NR	0	0
g23	NR	NR	NR	NR	NR	30	0
g24	14900	19350	19156.67	22200	1927.24	30	30

Table 2. Convergence of PESO in the Benchmark. NR Optimal not reached

In Table 2, we present the number of fitness function evaluations that PESO requires to attain a value within 1E-4 of the optimal. Also, the number of feasible runs, *F.R.* and the number of successful runs, *S.R.* are showed. We define like *F.R.* that run, which finds at least one feasible solution in less than 350000 fitness evaluations. On the other hand, when the best value found is within 1E-4 of the optimal the run is successful. Only the successful runs were used to calculate the measures presented in Table 2. Test problems with equality constraints require at least 315000 fitness function evaluations, due the dynamic tolerance applied in PESO. The experiments show a poor performance of PESO in test problems g20, g21, g22 and g23. These problems have several equality constraints; in fact the problems g20 and g22 have more than 10 of them. Now, we compare our approach against other PSO based methods and evolutionary algorithms representative of the state-of-the-art.

5.2 Comparison PESO versus Turbulent PSO

First we compare PESO against another PSO approach which applies *feasibility tournament* to handle constrained optimization problems. In Section 3, we mention that Toscano and Coello proposed a constraint handling technique for PSO (Toscano & Coello). Their approach handles constraints through a *feasibility tournament*, and keeps diversity by adding mutations to the velocity vector using a *turbulence* operator. They test the original benchmark with 13 test functions (Runnarson & Yao, 2000). The comparison is shown in Table 3. TC-PSO (Toscano and Coello's PSO) performed 340,000 fitness function evaluations, 10,000 less than PESO, but it is not significative for the comparison. The performance of PESO is better than TC-PSO on test problems g02, g05, g07, g09, g10 and g13.

TP	Optimal	PESO	TC-PSO
g01	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803619	-0.803432
g03	-1.000000	-1.000005	-1.004720
g04	-30665.538	-30665.53867	-30665.500000
g05	5126.49811	5126.498096	5126.640000
g06	-6961.8138	-6961.813876	-6961.810000
g07	24.306209	24.306209	24.351100
g08	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.638000
g10	7049.248	7049.248020	7057.590000
g11	0.750000	0.749999	0.749999
g12	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.068665

Table 3. Comparison of two PSO with feasibility tournament for handling constraints. Best results of PESO and TC-PSO into 30 runs

5.3 Comparison PESO versus Feasible PSO

Now, we compare PESO against the approach proposes by Hu and Eberhart (Hu & Eberhart, 2002). They apply a global PSO based on *feasibility preservation* for handling constraint problems. They only test the first 12 test functions of the benchmark. The comparison is shown in Table 4. HE-PSO (Hu and Eberhart's PSO) performed 100,000 fitness function evaluations. For developing a real comparison, PESO performed 100,000 fitness function evaluations. Nevertheless, the comparison is not equal since we do not take into account the number of constraint evaluations that HE-PSO performs to preserve feasibility. Also, we must mention that the randomly initialized particles are not always in the feasible space. So initialization may take a longer time. The performance of PESO is better than HE-PSO on test problems g02, g06, g07, g09 and g10. Even, there is not available information about the performance of HE-PSO at test problem g05. We should observe that PESO is robust to the number of fitness function evaluations. It is not a surprise, since in Table 2 we can observe the convergence rate of PESO for every test problem. In the first 12 test problems there are 3 with equality constraints; therefore, their convergence rate is driven by the dynamic tolerance. In the rest, there are 6 test problems with a best convergence rate lower than 100,000 fitness function evaluations. Only test problems g02, g07 and g10 have a best convergence rate upper than 100,000, which cause a little decrease in their best solution found around 30 runs.

TP	Optimal	PESO	HE-PSO
g01	-15.000000	-15.000000	-15.0
g02	-0.803619	-0.803613	-0.7130
g03	-1.000000	-1.000005	-1.0
g04	-30665.538	-30665.53867	-30665.5
g05	5126.49811	5126.498096	-
g06	-6961.8138	-6961.813876	-6961.7
g07	24.306209	24.306248	24.4420
g08	-0.095825	-0.095825	-0.0958250
g09	680.630057	680.630057	680.657
g10	7049.248	7049.250802	7131.01
g11	0.750000	0.749999	0.75
g12	-1.000000	-1.000000	-1.0

Table 4. Comparison of PESO against HE-PSO. Best results of PESO and HE-PSO into 30 runs

5.4 Comparison PESO versus Periodic Mode PSO

In Section 3, we mention the special technique for handling inequality constraints introduced by Zhang et al., called *periodic mode* (Zhang et al., 2004). Their method keeps the global-best near the boundary thus the flock which is constantly pulled to the border, can sustain exploration. They only tested the 9 functions with inequality constraints of the benchmark proposed by Runnarson and Yao (Runnarson & Yao, 2000). The comparison is shown in Table 5.

TP	Optimal	PESO	PM-PSO
g01	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803613	-0.64330
g04	-30665.538	-30665.53867	-30665.54
g06	-6961.8138	-6961.813876	-6961.814
g07	24.306209	24.306248	24.306
g08	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.630
g10	7049.248	7049.250802	7049.5

Table 5. Comparison of PESO against PM-PSO. Best results of PESO and PM-PSO into 30 runs

The PM-PSO algorithm (periodic mode PSO) performed 1,500,000 fitness function evaluations. Although, PESO performed four times less fitness function evaluations than PM-PSO, we did not increase the number of fitness function evaluations, because PESO performance is competitive with the general parameters applied in these experiments. The performance of PESO is better than PM-PSO on test problems g02 and g10.

5.5 Comparison PESO versus Diversity-DE

Now, we compare PESO against other evolutionary algorithms. We believe that could be interesting a comparison against a Differential Evolution algorithm, since PESO applies a *C-Perturbation* similar to the operator used for reproduction in this algorithm. Mezura et al. modified the Differential Evolution algorithm in a way that every parent may have more than one offspring (Mezura et al., 2005). The winner is the best child but then the child is compared to the current parent. Another tournament is performed but this time the winner is found by tossing a coin and comparing by fitness value, or by constraint violation; similar to Stochastic Ranking (Runnarson & Yao, 2000). The comparison of the first 13 test functions is shown in Table 6; the number of fitness evaluations for both algorithms is 225,000. The performance of PESO and Diversity-DE is very similar. A little advantage is shown by PESO on test problems g09 and g13.

TP	Optimal	PESO	Diversity-DE
g01	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803618	-0.803619
g03	-1.000000	-1.000005	-1.000
g04	-30665.538	-30665.53867	-30665.539
g05	5126.49811	5126.498096	5126.497
g06	-6961.8138	-6961.813876	-6961.814
g07	24.306209	24.306211	24.306
g08	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.638
g10	7049.248	7049.248435	7049.248
g11	0.750000	0.749999	0.75
g12	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053941

Table 6. Comparison of PESO against Diversity-DE. Best results of PESO and Diversity-DE into 30 runs

5.6 Comparison PESO versus SMES

The extend benchmark was proposed by Mezura in his Ph.D. thesis (Mezura, 2004). Therefore, it is interesting to compare PESO against the approach developed by Mezura, called SMES. SMES works over a simple multimembered evolution strategy: ($\mu+\lambda$)-ES. The modifications introduced into SMES are the reduction of the initial step size of the sigma values to favour finer movements in the search space. A panmictic recombination operator based on a combination of the discrete and intermediate recombination operators. Also, SMES changes the original deterministic replacement of the ES, sorting the solutions by applying a comparison mechanism based on feasibility. This allows remaining in the next generation, the best infeasible solution, from either the parents or the offspring population. In Table 7 we show the comparison of PESO and SMES. In this case both algorithms performed 240,000 fitness function evaluations. It can be seen that PESO is clearly better than SMES in problems g05, g07, g10, g13, g14, g15, g17, g19, g21 and g23. PESO and SMES were unable to find feasible solutions for test problems g20 and g22. But, PESO finds feasible solutions for test problems g17, g21 and g23, where SMES could not find feasible solutions in any single run.

TP	Optimal	PESO	SMES
g01	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803618	-0.803601
g03	-1.000000	-1.000005	-1.000000
g04	-30665.538	-30665.53867	-30665.539
g05	5126.49811	5126.498096	5126.599
g06	-6961.8138	-6961.813876	-6961.814
g07	24.306209	24.306211	24.327
g08	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.632
g10	7049.248	7049.248871	7051.903
g11	0.750000	0.749999	0.750000
g12	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053986
g14	-47.764411	-47.760600	-47.535
g15	961.715172	961.715171	*961.698
g16	-1.905155	-1.905155	-1.905
g17	8876.98068	8860.030895	*8890.1826
g18	-0.8660	-0.866025	-0.866
g19	32.386	32.351376	34.223
g20	0.096737	*0.204095	*0.211364
g21	193.778349	236.928359	*347.9809
g22	382.902205	*157.5136	*2340.6166
g23	-400.0025	-369.765012	*-1470.1525
g24	-5.508013	-5.508013	-5.508

Table 7. Comparison of PESO against SMES. Best results of PESO and SMES into 30 runs.
*Infeasible Solution

5.7 Comparison PESO versus ISRES

An algorithm representative of the state-of-the-art is the Stochastic Ranking algorithm was proposed by Runarsson and Yao (Runnarson & Yao, 2000). Later, the authors provided a new improved version, called Improved Stochastic Ranking Evolution Strategy, (ISRES) (Runnarson & Yao, 2005). The algorithm is a simple evolution strategy enhanced with a stochastic sorting, which decides, through a probability fixed value, performing a comparison using only the function value or the constraint violation. The ISRES's code is available at Runarsson's page, and we used it, for developing the experiments for test problems g14 through g24. The parameters used were the same as the suggested by the authors (Runnarson & Yao, 2005). The comparison is shown in Table 8. Both algorithms performed the same number of fitness function evaluations, 350000. Note that ISRES finds the best values for test problems g21 and g23. But PESO is better in problems g13 and g17. In test problem g21, PESO found feasible solutions in all 30 runs, whereas ISRES only had 5 successful runs. Both PESO and ISRES were unable to find feasible solutions for test problems g20 and g22.

TP	Optimal	PESO	ISRES
g01	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803619	-0.803619
g03	-1.000000	-1.000005	-1.001
g04	-30665.538	-30665.53867	-30665.539
g05	5126.49811	5126.498096	5126.497
g06	-6961.8138	-6961.813876	-6961.814
g07	24.306209	24.306209	24.306
g08	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.630
g10	7049.248	7049.248020	7049.248
g11	0.750000	0.749999	0.750
g12	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053942
g14	-47.764411	-47.761180	-47.761129
g15	961.715172	961.715171	961.715171
g16	-1.905155	-1.905155	-1.905155
g17	8876.98068	8856.502344	8889.9003
g18	-0.8660	-0.866025	-0.866025
g19	32.386	32.349645	32.348689
g20	0.096737	*0.204095	-
g21	193.778349	205.852693	193.785034
g22	382.902205	*157.5136	-
g23	-400.0025	-361.856637	-400.000551
g24	-5.508013	-5.508013	-5.508013

Table 8. Comparison of PESO against ISRES. Best results of PESO and ISRES into 30 runs.
*Infeasible Solution

6. Conclusion

In this chapter, we described a robust PSO for solving constrained optimization problems. We discussed the premature convergence problem, which still is an issue in evolutionary computation. A brief trip was made through several proposals to attain a balance between exploration and exploitation. Also, we briefly review recent works that contribute with interesting ideas for handling-constraints in PSO.

This work presents an algorithm called PESO to handle constrained optimization problems. Based on the empirical and theoretical results of several works, we explain and validate every component applied in PESO. We empirically show the performance of PESO in a well-know benchmark. PESO has shown high performance in constrained optimization problems of linear or nonlinear nature. Three important contributions of PESO are worth to mention: A new neighbourhood structure for PSO, the incorporation of perturbation operators without modifying the essence of the PSO, and a special handling technique for equality constraints.

The first contribution is the singly-linked neighbourhood structure. It increases the exploitation of the algorithm, breaking the double-link that exists between the particles using the original ring neighbourhood structure. PESO implements a singly-linked ring with a neighbourhood of size n = 2, but a general algorithm to build neighbourhoods of size n is given by Hernández et al. (Hernández et al., 2007).

Another relevant idea developed by PESO, is the perturbation of the target to keep flock's diversity and space exploration. Two perturbation operators, *C-perturbation* and *M-perturbation* are applied to the *pbest*. It is equivalent to perturb the particle's memory and not its behaviour; as it is performed by other approaches that tend to destroy the flock's organization capacity.

The last feature of PESO is its special technique to handle equality constraints. It is performed through a *dynamic tolerance* that allows unfeasible particles at the first generations, but it decreases the tolerance value until reach a desired error. The *dynamic tolerance* helps to keep the flock near the feasible region, while exploring promising regions.

The results on the benchmark problems provide evidence that PESO is highly competitive. So far, PESO performed very well at solving the current state-of-the-art problems, but it should be improved to handle problems with a higher number of equality constraints.

7. Future Research

PESO shows a competitive performance solving constrained optimization problems, so global (unconstrained) optimization and multi-objective optimization problems are attractive topics for future research. But, there are other research areas that could be explored in this approach.

As we mention in Section 4, the acceleration parameters have not been studied yet. A set of sub-swarms could improve the robustness of the PSO (Liang & Suganthan, 2006). One of the main research areas in evolutionary computation is the application to real optimization problems. In that field, we used PESO to solve system reliability optimization problems (Muñoz, 2004).

8. References

Agnes, M. (2004). *Webster's New World College Dictionary*, Webster's New World, ISBN 0764571257, May 2004.

Angeline, P. (1998). Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences, *Proceedings of the Seventh Annual Conference on Evolutionary Programming, EP VII*, pp. 601–610, ISBN 3-540-64891-7, March 1998, Springer, San Diego, California, USA.

Carlisle, A. & Dozier, G. (2000). Adapting Particle Swarm Optimization to Dynamic Environments, *Proceedings of the International Conference on Artificial Intelligence, ICAI2000*, pp. 429-434, ISBN 1892512564, Monte Carlo Resort Hotel, June 2000, CSREA Press, Las Vegas, Nevada, USA.

Clerc, M. & Kennedy, J. (2002). The Particle Swarm: Explosion, Stability, and Convergence in a Multi-Dimensional Complex Space, *IEEE Transactions on Evolutionary Computation,* Vol. 6, No. 1, February 2002, pp. 58-73, ISSN 1089-778X.

Coath, G. & Halgamuge, S. (2003). A comparison of Constraint-Handling Methods for the Application of Particle Swarm Optimization to Constrained Nonlinear Optimization Problems, *Proceedings of the 2003 Congress on Evolutionary Computation*, pp. 2419–2425, ISBN 0-7803-7804-0, Rydges Lakeside Hotel, December 2003, IEEE-Press, Canberra, Australia.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms, *Computer Methods in Applied Mechanics and Engineering*, Vol. 186, No. 2-4, June 2000, pp. 311-338, ISSN 0045-7825.

Downing, R. (2006). Neutrality and Gradualism: Encouraging Exploration and Exploitation Simultaneously with Binary Decision Diagrams, *Proceedings of IEEE Congress on Evolutionary Computation 2006, CEC 2006*, pp. 615-622, ISBN 0-7803-9487-9, Sheraton Vancouver Wall Centre Hotel, July 2006, IEEE-Press, Vancouver, BC, Canada.

El-Gallad, A.; El-Hawary, M. & Sallam, A. (2001). Swarming of intelligent Particle for Solving the Nonlinear Constrainted Optimization Problem. *International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications*, Vol. 9, No. 3, 2001, pp. 155–163, ISSN 0969-1170.

Fogel, D. (1994). An Introduction to Simulated Evolutionary Optimization. *IEEE Transaction on Neural Networks*, Vol. 5, No. 1, January 1994, pp. 3–14, ISSN 1045-9227.

He, S.; Prempain, E. & Wu, Q. (2004). An Improved Particle Swarm Optimizer for Mechanical Design Optimization Problems, *Engineering Optimization*, Vol. 36, No. 5, October 2004, pp. 585–605, ISSN 1029-0273.

Hernández, A.; Muñoz, A.; Villa, E. & Botello, S. (2007). COPSO: Constrained Optimization via PSO Algorithm, *Technical Report of the Computer Sciences Department*, I-07-04 (CC), February 2007, Centro de Investigación en Matemáticas, Guanajuato, México.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*, University of Michigan Press, ISBN 0-262-08213-6, Ann Arbor, MI, USA.

Hu, X. and Eberhart, R. (2002). Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization, *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics 2002 (SCI 2002)*, pp. 203-206, ISBN 980-07-8150-1, Sheraton World, July 2002, The International Institute of Informatics and Systemics, Orlando, Florida, USA.

Jacoby, N. (2005). Exploration and Exploitation Strategies. What kind of analytical models?, *Technical Report Cahiers de la Maison des Sciences Economiques*, May 2005, Université Panthéon-Sorbonne (Paris 1).

Kennedy, J. (1997). The Particle Swarm: Social Adaptation of Knowledge, *Proceedings of the IEEE International Conference on Evolutionary Computation, ICEC'97,* pp. 303-308, ISBN 0-7803-3949-5, University Place Hotel, April 1997, IEEE-Press, Indianapolis, IN, USA.

Kennedy, J. (1999). Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance, *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1931–1938, ISBN 0-7803-5536-9, Mayflower Hotel, July 1999, IEEE-Press, Washington, DC, USA.

Kennedy, J. & Mendes, R. (2002). Population Structure and Particle Swarm Performance, *Proceedings of the 2002 Congress on Evolutionary Computation, CEC02*, pp. 1671-1676, ISBN 0-7803-7282-4, Hilton Hawaiian Village Hotel, May 2002, IEEE-Press, Honolulu, HI, USA.

Liang, J.; Runarsson, T.; Mezura-Montes, E.; Clerc, M.; Suganthan, P.; Coello, C. & Deb, K. (2006). Problem Definitions and Evaluation Criteria for the CEC 2006, *Special Session on Constrained Real-Parameter Optimization, Technical Report*, Nanyang Technological University, Singapore, 2006.

Liang, J. & Suganthan, P. (2006). Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism, *Proceedings of the 2006 IEEE Congress on Evolutionary Computation, CEC 2006*, pp. 9-16, ISBN 0-7803-9487-9, Sheraton Vancouver Wall Centre Hotel, July 2006, IEEE Press, Vancouver, Canada.

March J. (1991). Exploration and Exploitation in Organizational Learning, *Organization Science*, Vol. 2, No. 1, February 1991, pp. 71-87, ISSN 1047-7039.

Mendes, R.; Kennedy, J. & Neves, J. (2004). The Fully Informed Particle Swarm: Simpler, Maybe Better, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, June 2004, pp. 204–210, ISSN 1089-778X.

Mezura, E. (2004). Alternatives to Handle Constraints in Evolutionary Optimization, Ph.D. thesis, CINVESTAV-IPN, DF, Mexico.

Mezura, E.; Velazquez, J. & Coello, C. (2005). Promising Infeasibility and Multiple Offspring Incorporated to Differential Evolution for Constrained Optimization, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2005*, pp. 225-232, ISBN 1-59593-010-8, L'Enfant Plaza Hotel, June 2005, ACM, Washington DC, USA.

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs,* Springer-Verlag, ISBN 3-540-60676-9, Germany

Michalewicz, Z. & Schoenauer, M. (1996). Evolutionary Algorithms for Constrained Parameter Optimization Problems, *Evolutionary Computation,* Vol. 4, No. 1, 1996, pp. 1-32, ISSN 1063-6560.

Michalewicz, Z. & Fogel, D. (2000). *How to Solve It: Modern Heuristics*, Springer-Verlag, ISBN 3-540-22494-7, Germany.

Muñoz, A. (2004). Diseño Optimo para la Confiabilidad, Master thesis, Department of Computer Sciences, CIMAT, Guanajuato, Gto, Mexico.

Muñoz, A.; Hernández, A. & Villa, E. (2005). Constrained Optimization Via Particle Evolutionary Swarm Optimization Algorithm (PESO), *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2005*, pp. 209-216, ISBN 1-59593-010-8, L'Enfant Plaza Hotel, June 2005, ACM, Washington DC, USA.

Munoz-Zavala, A.; Hernandez-Aguirre, A.; Villa-Diharce, E. & Botello-Rionda, S. (2006). PESO+ for Constrained Optimization, *Proceedings of the 2006 IEEE Congress on Evolutionary Computation, CEC 2006*, pp. 231-238, ISBN 0-7803-9487-9, Sheraton Vancouver Wall Centre Hotel, July 2006, IEEE Press, Vancouver, Canada.

Ozcan, E. & Mohan, C. (1998). Analysis of a Simple Particle Swarm Optimization System, Intelligent Engineering Systems Through Artificial Neural Networks: *Proceedings of the 1998 Artificial Neural Networks in Engineering Conference. ANNIE98*, pp. 253-258, ISBN 0-7918-0064-4, Marriott Pavilion Hotel, November 1998, ASME Press, St. Louis, Missouri, USA.

Parsopoulos, K. & Vrahatis, M. (2002). Particle Swarm Optimization Method for Constrained Optimization Problems, In: *Intelligent Technologies from Theory to Applications: New Trends in Intelligent Technologies,* P. Sincak, J. Vascak, V. Kvasnicka, and J. Pospichal, Ed. 76 of Frontiers in Artificial Intelligence and Applications, pp. 214-220, IOS Press, ISBN 1-5860-3256-9, Amsterdam ; Washington DC.

Price, K.; Storn, R. & Lampinen, J. (2005). *Differential Evolution: A practical Approach to Global Optimization*, Springer Verlag, ISBN 3540209506, Berlin, Germany.

Ratnaweera, A.; Halgamuge, S. & Watson, H. (2002a). Particle Swarm Optimization with Time Varying Acceleration Coefficients, *Proceedings of the First International Conference on Soft Computing and Intelligent Systems*, pp. 240-255, At the Auditorium of AIST, October 2002, Tsukuba, Japan.

Ratnaweera, A.; Halgamuge, S. & Watson, H. (2002b). Particle Swarm Optimization with Self-Adaptive Acceleration Coefficients, *Proceedings of the First International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 264-268, ISBN 981-04-7520-9, Orchid Country Club, November 2002, Singapore.

Runarsson, T. & Yao, X. (2000). Stochastic Ranking for Constrained Evolutionary Optimization, *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 3, September 2000, pp. 284–294, ISSN 1089-778X.

Runarsson, T. & Yao, X. (2005). Search Biases in Constrained Evolutionary Optimization, *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, Vol. 35, No. 2, May 2005, pp. 233–243, ISSN 1094-6977.

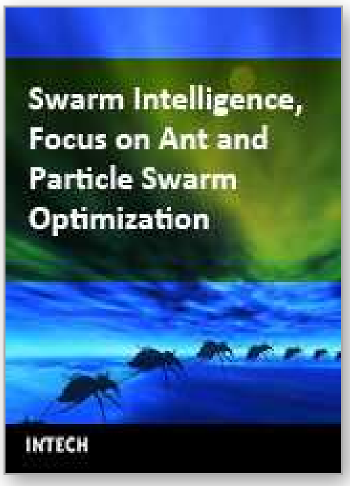
Suganthan, P. (1999). Particle Swarm Optimiser with Neighborhood Operator, *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1958-1962, ISBN 0-7803-5536-9, Mayflower Hotel, July 1999, IEEE-Press, Washington, DC, USA.

Toscano, G. & Coello, C. (2004). A Constraint-Handling Mechanism for Particle Swarm Optimization, *Proceedings of the 2004 Congress on Evolutionary Computation*, pp. 1396–1403, ISBN 0-7803-8515-2, Portland Marriott Downtown, June 2004, IEEE-Press, Portland, Oregon, USA.

Van Den Bergh, F. & Engelbrecht, A. (2001). Effects of Swarm Size on Cooperative Particle Swarm Optimisers, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2001*, pp. 892-899, ISBN 1-55860-774-9, Holiday Inn Golden Gateway Hotel, July 2001, Morgan Kaufmann, San Francisco, California, USA.

Van Den Bergh, F. (2002). An Analysis of Particle Swarm Optimizers, PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa.

Zhang, W.; Xie, X. & Bi, D., Handling Bloundary Constraints for Numerical Optimization by Particle Swarm Flying in Periodic Search Space, *Proceedings of the 2004 Congress on Evolutionary Computation*, pp. 2307–2311, ISBN 0-7803-8515-2, Portland Marriott Downtown, June 2004, IEEE-Press, Portland, Oregon, USA.



Swarm Intelligence, Focus on Ant and Particle Swarm Optimization

Edited by FelixT.S.Chan and Manoj KumarTiwari

ISBN 978-3-902613-09-7
Hard cover, 532 pages
Publisher I-Tech Education and Publishing
Published online 01, December, 2007
Published in print edition December, 2007

In the era globalisation the emerging technologies are governing engineering industries to a multifaceted state. The escalating complexity has demanded researchers to find the possible ways of easing the solution of the problems. This has motivated the researchers to grasp ideas from the nature and implant it in the engineering sciences. This way of thinking led to emergence of many biologically inspired algorithms that have proven to be efficient in handling the computationally complex problems with competence such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), etc. Motivated by the capability of the biologically inspired algorithms the present book on "Swarm Intelligence: Focus on Ant and Particle Swarm Optimization" aims to present recent developments and applications concerning optimization with swarm intelligence techniques. The papers selected for this book comprise a cross-section of topics that reflect a variety of perspectives and disciplinary backgrounds. In addition to the introduction of new concepts of swarm intelligence, this book also presented some selected representative case studies covering power plant maintenance scheduling; geotechnical engineering; design and machining tolerances; layout problems; manufacturing process plan; job-shop scheduling; structural design; environmental dispatching problems; wireless communication; water distribution systems; multi-plant supply chain; fault diagnosis of airplane engines; and process scheduling. I believe these 27 chapters presented in this book adequately reflect these topics.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821