Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 8250185, 11 pages
https://doi.org/10.1155/2019/8250185

Research Article

Improved Dynamical Particle Swarm Optimization Method for Structural Dynamics

W. D. Annicchiarico [1] and M. Cerrolaza [2,3]

[1] *Dept. Ingeniería Mecánica, Universidad Simón Bolívar, Caracas, Venezuela*
[2] *Aplicaciones en Informática Avanzada (AIA), Barcelona, Spain*
[3] *Universidad Politécnica de Cataluña, Barcelona, Spain*

Correspondence should be addressed to M. Cerrolaza; mcerrola123@gmail.com

Received 18 July 2018; Accepted 2 May 2019; Published 10 June 2019

Academic Editor: Gen Q. Xu

A methodology to the multiobjective structural design of buildings based on an improved particle swarm optimization algorithm is presented, which has proved to be very efficient and robust in nonlinear problems and when the optimization objectives are in conflict. In particular, the behaviour of the particle swarm optimization (PSO) classical algorithm is improved by dynamically adding autoadaptive mechanisms that enhance the exploration/exploitation trade-off and diversity of the proposed algorithm, avoiding getting trapped in local minima. A novel integrated optimization system was developed, called DI-PSO, to solve this problem which is able to control and even improve the structural behaviour under seismic excitations. In order to demonstrate the effectiveness of the proposed approach, the methodology is tested against some benchmark problems. Then a 3-story-building model is optimized under different objective cases, concluding that the improved multiobjective optimization methodology using DI-PSO is more efficient as compared with those designs obtained using single optimization.

1. Introduction

The most popular and used evolutionary computation (EC) techniques, known as Evolution Strategies [1–3], Genetic Algorithms [4, 5], Genetic Programming [6], and Evolutionary Programming [7], are nature-inspired algorithms which mimic natural evolutionary principles such as *The Survival of Species* [8]. In the field of structural engineering, several authors have contributed in some applications, such as damage detection in structures [9, 10], truss optimization [11], identification of dynamical properties [12], structural design [13], vibration controls [14], shape optimization [15], among others.

Particle Swarm Optimization Methods, even though they belong to EC new class of emerging methods, are based on intelligent-sociocognitive behaviours observed in natural species [16]. The PSO methodology was initially proposed by Eberhart and Kennedy in 1995 [17, 18] and shares with EC optimization techniques the way the design space is searched. Both of them use a population of candidate solutions to exploit and explore the design space to find out the optimal or quasi-optimal solution. They differ, however, on how the swarm particles (or individuals) are created. In PSO, the particles have memory, meaning that every single member of the swarm has the ability to remember its best individual position acquired so far while moving through the searching space. Its final position is determined by the combination of this knowledge and the influence exerted by the best individual's position of the topological configuration of the neighbourhood where the particle inhabits or by the best individual position in the whole swarm.

The swarm optimization methodology has been the base of other variants of PSO, thus increasing the number of algorithms that belong to this particular branch of stochastic optimization methodologies. Among them, it is worth mentioning Quantum-behaved Particle Swarm Optimization (QPSO) [19]. QPSO has in common with the original PSO theory the concept of a set of individual best vectors and also uses the global best vector, to propose a global-convergent PSO algorithm. The positions of the particles are obtained by probabilistic sampling throughout the domain and assuming that the particles have quantum behaviour and must converge to their local attractors. The search scope of each particle is based on a normalized probability density function obtained by solving the Schrödinger equation. Also, in QPSO the algorithm does not use recursion, or the inertia term as in the original version of PSO, to promote the perturbation of the new sample points, but it uses a form of reference-point elitism. In [20] the same authors introduced a modified QPSO called MQPSO; the only difference is the way the local attractor is chosen. In this case a randomly selected local attractor vector is selected if its evaluation on the objective function is better than the particle natural local attractor. Recently, in [21] a novel position-update mechanism based on the ring model and combining it with the classical QPSO have been proposed to deal with multiobjective optimization problems.

Several applications to complex problems in diverse fields of engineering show that the methodology is efficient and robust as global optimizer due to its easy implementation, low computational cost, and no gradients, and no additional information about the optimization function is needed. All of the above together with its great ability and good performance in changing and noisy environments has suggested that this methodology would be a very suitable approach in structural dynamics optimization problems discussed herein.

The general advantages offered by optimization algorithms belonging to the class of evolutionary computation over traditional optimization algorithms suggest the development of improved approaches to solve engineering practical problems. Thus, optimization of structural design systems in concordance with best-design practices of modern buildings and infrastructures could be faced. The main objective of this work is to evaluate and propose a design methodology, based on the paradigms of social behaviour, to optimize building structural models which include different criteria and objectives found on all disciplines involved.

This article is divided into the following sections: first, the theoretical formulation of particle swarm optimization and the proposed improvements to make the algorithm more effective in the solution of structural design problems is presented. Then, the multiobjective analysis by means of PSO and the implementation of DI-PSO is described in detail. In the next section, the proposed optimization methodology is assessed by testing against different mathematical benchmark problems. Then, a general description of a multiobjective structural optimization problem is presented and described. After that, the DI-PSO methodology is used to optimize a real structural frame-building model under different optimization cases. Finally, the main recommendations and conclusions will be derived to continue developing the proposed methodology, based on social-cognitive paradigm, in the field of structural optimization.

2. Materials and Methods

2.1. Social Intelligent Paradigm Optimization and Its Improvements. The classical PSO algorithm can be implemented in two different versions. In the global version, the final position of each particle results from the compromise relationship between the particle's best positions reached so far and the best position of the best individual of the whole swarm. In the local variant of the algorithm, instead of taking into account the position of the best individual of the entire swarm, it takes the position of the best individual of the local neighbourhood where the particle belongs. The PSO global approach is described below, as well as some of the developed mechanisms that have been introduced to enhance the optimization algorithm presented in this work (see [23] for details).

The single-objective optimization (for simplicity a minimization is assumed) problem of a function of real values f consists in finding out a vector of design variables, X^*, such that

$$\begin{gathered} f : S \longrightarrow \mathbb{R} \\ f\left(X^*\right) \leq f\left(x\right), \quad \forall X \in S \end{gathered} \tag{1}$$

where $S \subset \mathbb{R}^D$ is a compact not empty set.

Let us assume that the searching space is J-dimensional and the i-particle of the swarm can be represented as a J-dimensional vector of his position X_i,

$$X_i = \left\langle x_{i1} \quad x_{i2} \quad x_{i3} \quad \cdots \quad x_{ij} \right\rangle \tag{2}$$

and defining $V_{i,j}^{n+1}$ as the change of the particle's velocity (changes over time of its spatial position):

$$\begin{aligned} V_{i,j}^{n+1} = {} & w * V_{i,j}^{n} + c_1 * r_1^n * \left(P_{\mathrm{i\_best,j}}^{n} - x_{i,j}^{n}\right) + c_2 * r_2^n \\ & * \left(P_{\mathrm{Gen\_best,j}}^{n} - x_{i,j}^{n}\right) \end{aligned} \tag{3}$$

then the new position of the particle will be

$$X_{i,j}^{n+1} = X_{i,j}^{n} + V_{i,j}^{n+1} \tag{4}$$

where $V_{i,j}$ is the i-particle's velocity in the jth dimension; w is the inertia factor; c_1 is the individual consciousness, which is a weighting coefficient that measure and control the influence of the best position reached so far for the i-particle ($P_{\mathrm{i\_best,j}}$); and c_2 is the social or collective consciousness parameter of the swarm; it represents a weighting coefficient that measure the influence of the best position reached by the best particle of the whole colony ($P_{\mathrm{Gen\_best,j}}$). Finally, r_1 and r_2 are normal distributed random numbers within the range of (0,1), and the super indexes indicate the iterations.

The swarm behaviour is based on mechanisms which have influence over the temporal-space position of each individual of the swarm as well as the location of the entire colony (those mechanisms are known as the principle of proximity or the rule of simple space movements and time computations; the quality principle or the response of the swarm to quality factors in the environment; principle of diverse response which indicates how the swarm allocates and distributes its resources; the stability principle which regulates how the swarm must react under every fluctuation of the environment; and the adaptability principle that dictates the ability of the swarm to change its behaviour when the benefits surplus are the cost to achieve such behavioural change (see Millonas [24]). These mechanisms take into account quality factors derived from the best position reached so far for each particle and the position obtained by the best particle of the swarm, allocating the response of each particle over the searching space in a way that assures the required diversity in the searching process. Moreover, the behaviour of the colony changes only when the best individual of the colony changes. It can be said that the response of the algorithm is adaptable and stable at the same time.

Despite all these advantages, PSO suffers, as well as others optimization algorithms belonging to EC, from lack of diversity and adaptability in the latest stages of the searching process. In the final stages of the searching process, these kinds of algorithms show off a diminishing capacity of exploration due to concentration of individuals in a particular region of the design space. As a consequent, the algorithms lose the ability to react and to find out better places to be explored. Regarding this matter, the specialized literature offers general recommendations for choosing the values of the swarm's optimization parameters. Kennedy [25] and Shi & Eberhart [26] suggested a linear variation through generations, ranging between 1,2 and 0,4 for the inertia factor (w) and constant acceleration coefficients (c_1; c_2) equal to 2. Despite these recommendations, it is always useful to tune up these parameters to each particular problem.

According to the discussion above and to improve both the diversity and quality of the colony when searching the optimum or quasi-optimum solution, the required autoadaptive mechanisms for a better adjustment of the inertia factor (w) and the acceleration constants associated with the individual consciousness (c_1) and the social consciousness (c_2) are proposed and implemented in this work:

(a) Related to the inertia factor (w): a success index ($\mathrm{Rate\_S}^{(n)}$) of the swarm is evaluated at each generation in order to see in which percentage the swarm particles have improved (or not) their objective values. Thus, the success of an individual of the swarm (particle i) in the iteration n ($s_i^{(n)}$) is defined in terms of the improvement (or not) of its personal score compared to its performance at the previous iteration:

$$s_i^{(n)} = \begin{cases} 1, & \text{Si } f\left(P_{i,j}^{n+1}\right) < f\left(P_{i,j}^{n}\right) \\ 0, & \text{Otherwise} \end{cases} \tag{5}$$

Based on this index, the success ratio in the n-iteration ($\mathrm{Rate\_S}^{(n)}$) can be estimated as a function of the success of the whole swarm (S) by

$$\mathrm{Rate\_S}^{(n)} = \frac{\sum_i s_i^{(n)}}{S} \tag{6}$$

$\mathrm{Rate\_S}^{(n)}$ ($\mathrm{Rate\_S}^{(n)} \in [0,1]$) represents the percentage of the particles which have improved their behaviour in the last iteration. A high value of $\mathrm{Rate\_S}^{(n)}$ indicates a high probability that most of the particles have converged to a nonoptimum point or they are slowly moving to the optimum. On the contrary, a lower value of $\mathrm{Rate\_S}^{(n)}$ indicates that the particles are oscillating without much improvement [27]. In order to dynamically adapt the inertia factor in these cases, an exponential variation of the inertia factor in terms of the number of iterations is used:

$$w^{(n)} = e^{\mathrm{Rate\_S}^{(n)}-1} \tag{7}$$

(b) Related to the acceleration constants (c_1, c_2): it is well known that the optimization process of any evolutionary or sociocognitive algorithms is based on the exploration and exploitation of the design space by a population (or swarm's particles) of candidate solutions. At the initial stages of the searching process it is always useful and recommended to encourage exploration of the design space. And at the final stages of this process, it is a better practice to increase and improve the exploitation of the design space where the possible quasi-optimal or optimal solution can be found. To promote these behaviours, a mechanism that linearly reduces the individual consciousness (c_1) and increases the social consciousness (c_2) while the searching process is running [28] is used in this work, as follows:

$$\begin{gathered} c_1^n = c_{1max} - \frac{n}{n_{\max}}\left(c_{1max} - c_{1min}\right) \\ c_2^n = c_{2max} + \frac{n}{n_{\max}}\left(c_{2max} - c_{2min}\right) \\ 0 < c_1^n + c_2^n < 4 \end{gathered} \tag{8}$$

(c) To preserve an adequate diversity during the searching process: an exploration dynamic operator and a kind of elitist crossover operator inspired by the ideas discussed by Wu *et al.* [29] have been implemented in this work. The authors developed an Improved Chicken Swarm Optimization Method applied to calculate reentry trajectories.

The exploration ability of the classical PSO algorithm is improved, especially in the later stages of the searching process, by implementing the idea of an exploration dynamic operator. This operator is defined through a dynamic mutation factor (P_{mut}) which represents the mutation probability of the particle positions. It has the following expression:

$$P_{mut} = \frac{\left(P_{max} - P_{min}\right) x \left(\mathrm{Inter}_{est} - \mathrm{Iter}_{\mathrm{current}} \mathrm{mod} \mathrm{Inter}_{est}\right)}{\mathrm{Inter}_{est}} + P_{min} \tag{9}$$

where

$P_{\max}$, $P_{\min}$: maximum and minimum value of P_{mut};

$\mathrm{Inter}_{\mathrm{est}}$: not improvement interval;

$\mathrm{Iter}_{\mathrm{current}}$: current iteration.

When the swarm stagnates during the searching process, detected by no improvement in the values of the objective function in the span of several iterations (e.g., 5 generations), P_{mut} probability is calculated at the current iteration and is applied independently to each coordinates of the particles, updating their positions randomly in the range of values allowed for the design variables of each parameter. The mutation probability is dynamically adjusted within the interval $[P_{max}, P_{min}]$. If in the subsequent iterations P_{mut} value gets less than P_{min}, its value is locked at P_{min} ($P_{mut}=P_{min}$) and the positions of the particles of the colony are modified. The next iteration P_{mut} is restored to its maximum value and the iteration counter is reset to one.

Elitist Crossover Operator: it has been shown [30] that the performances of swarm intelligence algorithms in solving high-dimensional optimization problems are prone to easily fall into local optimum and suffer from premature convergence. To overcome these drawbacks, a kind of real crossover operator is implemented in the following way: after updating the particles spatial position, a random number a ($a \in [0,1]$) is generated. If a is less than the default crossover probability p_c, the best two particles of the swarm (without taking in consideration the best particle of the swarm ($P_{Gen\_best,j}$)) are chosen to perform the elitist crossover operation. The new particles are then created by using the following equations:

$$\begin{aligned} NewPart_1 &= IPI * BestPart_1 + (1 - IPI) \\ &\quad * BestPart_2 \\ NewPart_2 &= (1 - IPI) * BestPart_1 + IPI \\ &\quad * BestPart_2 \end{aligned} \tag{10}$$

where the parameter IPI represents the percentage of information that each new particles can inherit from their parents. After the crossover operation, the two new particles will substitute the two particles with the worst fitness values and the mingling process will continue with the rest of the particles. When the algorithm is trapped into the local optimum, the elitist crossover operator can change the position of some of the best particles trapped into local optimum. Therefore and within certain probability, the fitness value of the new particles can be better than the best particle of the swarm ($p_{g,j}$), replacing it and taking the algorithm out of the local optimum.

Following the recommendations of Van den Bergh [31], a series of experiments have been done to select and tune the values of the different optimization parameters needed to implement the above-mentioned operators. Accordingly with these results, the maximum and minimum values of the dynamical mutation factor are P_{max}=0,8 and P_{min}=0,4. The stagnation interval has been limited to ten iterations ($Inter_{est}$=10), the stagnation threshold to five generations, the default crossover probability p_c (p_c=0,8), and the information percentage inheritance IPI=0,5 or 50%.

2.2. Swarm Intelligence and Multiobjective Optimization Problems. Most real-world search and optimization problems are naturally posed as multiobjective optimization problems (MOOPs). They pose a real challenge that has attracted the attention of scientific researchers in different disciplines, especially in engineering and science, due to their inherent complexities and due to the lack of suitable and efficient solution techniques.

To solve MOOPs many evolutionary-based and swarm intelligence-based optimization algorithms have been proposed. The first ones adopt the evolutionary computation strategy that simulates principles observed in biological evolution, such as selection, mating, mutation, cloning, among others techniques, and mechanisms inspired from natural evolution. A variety of these approaches can be found elsewhere [32, 33]. The second kind of algorithms follows the computational intelligence paradigm in imitating the swarm behaviour by adjusting position and velocity for each particle as well as its neighbours. Their application to solve multiobjective optimization problems is investigated in [13, 34].

The solution of a multiobjective optimization problem results in a number of optimal solutions due to the presence of conflicting multiple objectives. The complexities of this problem arise in the fact that an ideal MOOP algorithm must be able to find multiple optimal solutions and to seek for optimal solutions with a good diversity in objective and/or decision variable space(s). Traditional solution approaches are based in a decomposition concept or strategy; they transform the multiobjective problem in several single-objective optimization problems that are simultaneously optimized. There are several methodologies for constructing aggregation functions of decomposition (e.g., Weighted Sum Method, ε-Constraint Method, Weighted Metric Methods, Tschebyscheff, Weighted Tschebyscheff, and penalty based boundary intersection (PBI)). On the other hand, there are approaches based on the concept of domination in their search. These methods use the domination concept as a way to compare solutions with multiple objectives, casting this information into the fitness function to correlatively treat all objectives to find optimal solutions, known as Pareto-optimal solutions. Dominance guarantees the impossibility to find a solution that improves an objective without degrading at least another one.

Due to the advantages of dominance strategies to deal with multiobjective problem (MOP), i.e., there is no need to transform MOP to one objective problem and their capability of generating a diverse set of Pareto-optimal solutions in a single run, a NSGA-II [35] inspired strategy is now used to customize the dynamically improved particle swarm optimization (DI-PSO) methodology proposed herein. A general view of the MOP problem is introduced below as well as the Pareto dominance criterion and the elitist nondominated sorting methodology used in DI-PSO. In general, MOOPs consist to simultaneously optimize a set of objective functions which are to be minimized or maximized. The problem is constrained by a number of equality and inequality functions that must be satisfied by any feasible solution (including the optimal solution). The problem formulation is written as

$$\frac{\text{Minimize}}{\text{Maximize}} \quad \mathrm{f_m}(\mathrm{x}) = \left| f(x)_1, f(x)_2, \cdots, f(x)_M \right|$$
$$\text{Subjected to} \quad \begin{cases} \mathrm{g}_l(x) \le 0, & \mathrm{l} = 1, 2, \cdots, \mathrm{L} \\ \mathrm{h}_k(x) = 0, & \mathrm{k} = 1, 2, \cdots, \mathrm{K} \\ \mathrm{x}_j^{(\mathrm{L})} \le \mathrm{x}_j \le \mathrm{x}_j^{(\mathrm{U})}, & j = 1, 2, \cdots, \mathrm{J} \end{cases} \tag{11}$$

The idea is to find a design vector of variables, X^*, such that

$$f_{\mathrm{i}}(\mathrm{X}^*) = \min f_{\mathrm{i}}(\mathrm{X}), \quad \mathrm{i} = 1, 2, \cdots, \mathrm{M} \tag{12}$$

However, this is not the common situation and the objective functions behave among them in the opposite sense. One way to solve this problem is to find the greater number of solutions which fulfilled domination criteria of Pareto optimization for multiobjective problems. In Pareto optimization, a solution vector, $\mathrm{X}^{(1)}$, is said to dominate a solution vector $\mathrm{X}^{(2)}$, if the following conditions are simultaneously satisfied:

(1) Evaluation of solution $\mathrm{X}^{(1)}$ is no worse than evaluation of solution $\mathrm{X}^{(2)}$ in all objectives (in an objective function minimization "no worse than" means "is less or equal than..."):

$$f_{\mathrm{i}}\left(\mathrm{X}^{(1)}\right) \le f_{\mathrm{i}}\left(\mathrm{X}^{(2)}\right), \quad \mathrm{i} = 1, 2, \cdots, \mathrm{M} \tag{13}$$

(2) Evaluation of solution $\mathrm{X}^{(1)}$ is strictly better than solution $\mathrm{X}^{(2)}$ in at least one objective function:

$$\exists \overline{\mathrm{m}} \in \{1, 2, \cdots, \mathrm{M}\}: \quad f_{\overline{\mathrm{m}}}\left(\mathrm{X}^{(1)}\right) < f_{\overline{\mathrm{m}}}\left(\mathrm{X}^{(2)}\right) \tag{14}$$

Applying the Pareto criteria to a set of solutions P, it is possible to find out the nondominated set of solutions P' that are not dominated by any member of the set P. When the set P is the entire feasible search space, the resulting nondominated set P' is called the Pareto-optimal set or the Pareto frontier (PF).

Generally, in MOP there is not a common minimum (or maximum) for all objective functions. Strictly speaking, there is not a minimization (or maximization) at all, so that the task of the designer is to identify the greater number of possible Pareto minimum and in terms of them select the most suitable solution that in a compromise way allows solving the set of objective functions.

An elitist nondominated sorting strategy is implemented inside the architecture of DI-PSO in order to deal with multiple-objective problems as a way to find out the nondominated set of solutions. This strategy was inspired on the well-known NSGA-II algorithm formulated by Deb *et al.* [22]. It was chosen due to its efficiency and faster speed to deal with multiobjective problems and also by incorporating elitism and assigning characteristics based on the concepts of dominance and density. Elitism is possible within the algorithm due to the combination of frontiers of nondominated solutions extracted from two successive iterations of the algorithm, parent, and offspring populations in the case of the GA. The adaptation process is firstly evaluated by using a Pareto nondomination classification procedure [36] of the total population created by the combination of parent and offspring, followed by a procedure that measures the density of solutions surrounding a particular solution in the population. The first step belongs to the nondomination classification procedure initially suggested by Goldberg [4] while the last one consists in the assignation of a density index based on the Manhattan distance between the nearest neighbours of a particular solution that belongs to the same rank or frontier.

2.3. Nondominated Sorting Strategy.

The sorting strategy and how to obtain the Pareto frontier, which is saved in an external file for later evaluation and processing, are described in a step-by-step format. Initially, a random N-sized swarm population ($\mathrm{Sw\_P_t}$) is created. A N-sized offspring population ($\mathrm{Sw\_Q_t}$) is then obtained from the previous one by applying intelligence-swarm operators. The populations are sorted into nondomination levels or frontiers. Each solution is assigned a fitness equal to its nondomination level (1 is the best level). Thus minimization of the fitness is assumed.

Step 1. Combine swarm populations ($\mathrm{Sw\_P_t}$) and ($\mathrm{Sw\_Q_t}$) and create $\mathrm{Sw\_R_t}$ ($\mathrm{Sw\_R}_t = \mathrm{Sw\_P_t} \cup \mathrm{Sw\_Q_t}$). Perform a nondominated sorting to $\mathrm{Sw\_R_t}$ and identify different fronts: F_i, i= 1,2,. . ., etc.

Step 2. Set new swarm $\mathrm{Sw\_P_{t+1}} = 0$. Set counter i=1. Until $|\mathrm{Sw\_P_{t+1}}| + |F_i| < \mathrm{N}$, perform $\mathrm{Sw\_P}_{t+1} = \mathrm{Sw\_P}_{t+1} \cup F_i$ and i= i+1.

Step 3. Perform the Crowding-sort procedure (Crowd_Sort) on the last frontier which cannot be completely assigned in the remaining slots remain in $\mathrm{Sw\_P_{t+1}}$ and include the most widely spread ($\mathrm{N}-|\mathrm{Sw\_P}_{t+1}|$) solutions by using the crowding distance values in the sorted F_{i} to $\mathrm{Sw\_P_{t+1}}$. Such procedure is based on the crowding distance metric ($\mathrm{Crowd\_d_i}$), explained below.

Step 4. Create a new offspring swarm population ($\mathrm{Sw\_Q_{t+1}}$) derived from swarm population $\mathrm{Sw\_P_{t+1}}$ by using position and velocity swarm equations.

In the above procedure, it is assumed that every solution has two attributes: a nondominated rank ($\mathrm{r_i}$) that is the nondominated front where the solution lies, and a local crowding distance ($\mathrm{Crowd\_d_i}$) which is a measure of the search space around the solution i and is not occupied by any other solution in the population.

2.4. Evaluation of the Crowding Distance Metric (Crowd_di).

$\mathrm{Crowd\_d_i}$ is an estimate of the density of solutions surrounding a particular solution i that belongs to a frontier (F) of the swarm population sorted in nondomination levels. To get this metric, the average distance of two solutions on either side of solution i along each of the objectives is taken. The distance assignment procedure is described below step by step.

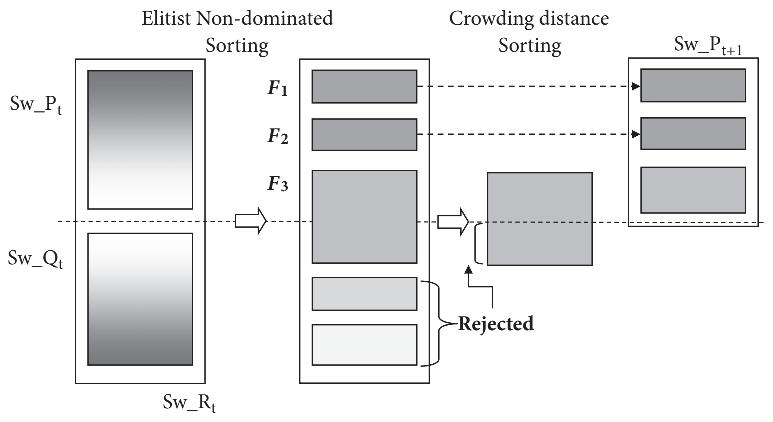

FIGURE 1: Schematic view of elitist nondominated sorting strategy and crowding distance sorting processes (based on NSGA-II [22]).

Step Crw_1. Call the number of solutions in F as $l = |F|$. For each i-solution in the set, first assign $\mathrm{Crow\_d_i} = 0$.

Step Crw_2. For each objective function $m = 1, 2, \cdots, M$, sort the set in the worse order of their function objective values (f_m).

Step Crw_3. For $m = 1, 2, \cdots, M$, assign a large distance to the boundary solutions: $\mathrm{Crow\_d}_{l_1^m} = \mathrm{Crow\_d}_{l_L^m} = \infty$, and for all other solutions j = 2 to (l − 1), assign

$$\mathrm{Crow\_d}_{l_j^m} = \mathrm{Crow\_d}_{l_j^m} + \frac{f_m^{(l_{j+1}^m)} - f_m^{(l_{j-1}^m)}}{f_m^{\max} - f_m^{\min}} \tag{15}$$

The index l_j denotes the solution index of the j-th member in the sorted list. Thus, for any objective, the indexes l_1 and l_L denote the lowest and highest objective function values, respectively. The parameters $f_m^{\max}$ and $f_m^{\min}$ can be set as the population-maximum and population-minimum values of the m-th objective function. To illustrate the above described mechanisms, Figure 1 depicts a schematic view of the elitist nondominated sorting mechanism and the crowding distance sorting processes developed in this work.

The basic DI-PSO algorithm can be written as follows:

```
Begin
Parameter settings and initialize swarm
Evaluate fitness and initialize leaders ($P^n_{\mathrm{i\_GenBest,j}}$) keeps them in a leader_pool
Identify the top best leader from the leader_pool through Crowd_d$_i$.quality measure
K = 0                         // K = Iteration count
While (the stopping criterion is not met, say, K < Kmax)
  For each particle in ($\mathrm{Sw\_P}^t_{i,j}$) do (to obtain $\mathrm{Sw\_Q}^t_{i,j}$)
    Select leader in the leader_pool ($P_{\mathrm{i\_GenBest,j}}$)
    Update velocity
    Update position
    Apply auto-adaptive mechanisms and operators
    Evaluate fitness
    Update $P_{\mathrm{i\_best,j}}$
  End For
  With ($\mathrm{Sw\_P}^t_{i,j}$ and $\mathrm{Sw\_Q}^t_{i,j}$) do
    Elitist Non-dominated Sorting
    Crowding distance Sorting
    Obtain $Sw\_P^{t+1}$
  End with
  If Stagnation then do
    Apply exploration dynamic operator
    Elitist Crossover Operator
  End do
  Update the top best into the external archive
  K = K + 1.
End While
Report results in the external archive
End
```

As it can be observed, the first step is the swarm initialization. Then, a set of leaders is also initialized with the nondominated particles from the swarm. This set of leaders is stored in an external file called leader_pool. Later on, the attributes of each particle (its rank (r_i) and local crowding distance ($\mathrm{Crowd\_d_i}$)) are calculated for all particles and leaders to select the best among them. At each generation and for each particle, a leader is selected and the flight is performed. Next the autoadaptive mechanisms discussed above are applied. Then, the particle is evaluated and its corresponding $P_{\mathrm{i\_best,j}}$ is updated. A new particle replaces its $P_{\mathrm{i\_best,j}}$ particle usually when this particle is dominated or if both are incomparable (i.e., they are both nondominated with respect to each other). After all the particles have been updated, the set of leaders is updated, too. Finally, the quality measure of the set of leaders is recalculated. This process is repeated for a certain fixed number of iterations.

TABLE 1: Mathematical test problem formulation.

Problem	Type	n	Variable Bounds	Objective Functions
ZDT3	Unconstrained	30	$[0, 1]$	$f_1(x) = x_1$; $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)} \sin(10\pi x_1)\right]$; $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
TNK	Constrained	2	$[0, \pi]$	$f_1(x) = x_1 \quad f_2(x) = x_2$; $C_1(x) \equiv x_1^2 + x_2^2 - 1 - 0.1\cos\left(16 * \arctan\frac{x_1}{x_2}\right) \geq 0$; $C_2(x) \equiv (x_1 - 0.5)^2 + (x_2 - 0.5)x_1^2 \leq 0.5$

TABLE 2: Comparison of the average convergence and diversity metrics between NSGA-II and the proposed methodology DI-PSO.

	Convergence		Diversity	
Problem	NSGA-II [22]	DI-PSO	NSGA-II [22]	DI-PSO
ZDT3	0.00139	0.00137	0.55060	0.54492
TNK	0.02470	0.02350	0.56080	0.55010

3. Results and Discussion

3.1. Application on Mathematical Test Cases. The algorithm is applied to test cases to evaluate its performance on closed-form mathematical functions, before to apply the developed method in real situations. Technical literature describes a great number of test cases for single and multiple objectives, almost as varied as the number of EC algorithms that have been written. It is of course impossible to be completely general when establishing a test suite, so some subjective viewing of the subject is required and some generalizing assumptions must be made. In this sense and for illustration purposes two test cases were chosen: test case ZDT3 from benchmark problems presented in Zitzler *et al.* [37] and test case TNK from Coello *et al.* [32]. Table 1 shows their formulation.

The first case (ZDT3) is an example of an optimal discontinuous Pareto front and thus would not be able to be treated by a deterministic optimizer. Among some of its characteristics its PF has five separate convex bands. It was solved with 150-particles swarm. The second case (TNK) also has a discontinuous PF and was chosen due to the nature of its nonlinear constraints. It was solved with 100-particles swarm. Even though it is evident when the entire population has converged to the PF which can be seen by inspection in Figures 2 and 3, the performances of the proposed algorithm are also measured in terms of convergence towards the optimal Pareto front and diversity of solutions along that front by using the two metrics introduced by Deb *et al.* [22]. Convergence is measured as the average minimum Euclidean distance between last generation individuals and uniformly sampled points on the optimal Pareto front. As reference points for the Pareto front, a 1000-points set is used for each problem, provided elsewhere (see, for instance, [38]) or in the website of the System Optimization group of the Swiss Federal Institute of Technology of Zurich. The smaller the value for this metric is, the better the convergence towards the optimal Pareto front is. Diversity is measured as the nonuniformity of the average distance between adjacent

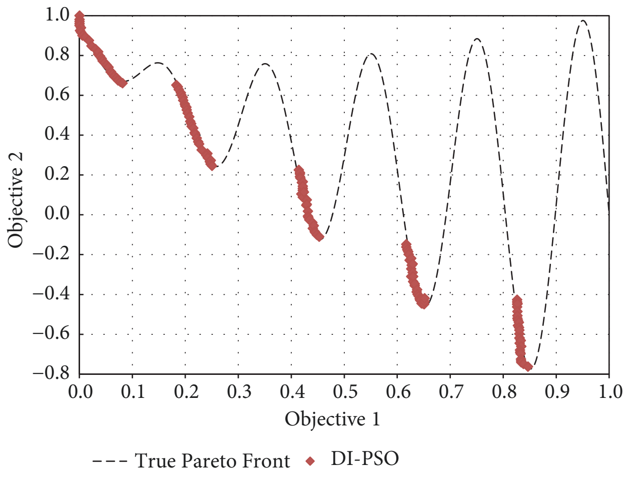

FIGURE 2: Pareto front obtained for ZDT3 test case.

solutions of the optimal Pareto front. The closer the metric is to 0, the better the diversity of solutions are (see Table 2).

3.2. Multiobjective Structural Optimization Problem. A seismic-structural problem is presented and discussed in this section. It consists in finding out a compromise solution between the building structural topology and the seismic input energy that the building can safely take. The above problem can be restated as how the elements dimensions, the allowed elements displacements, and story drifts of a building should be selected in order to obtain an optimized building seismic design which represents the better trade-off between the building minimum weight and the maximum seismic input energy that the building can resist.

In this sense, the structural-seismic design system requires the definition of a merit function that includes the different variables needed to perform the seismic design of the building structure. The technical literature is plenty of different evaluation functions to optimize the distribution

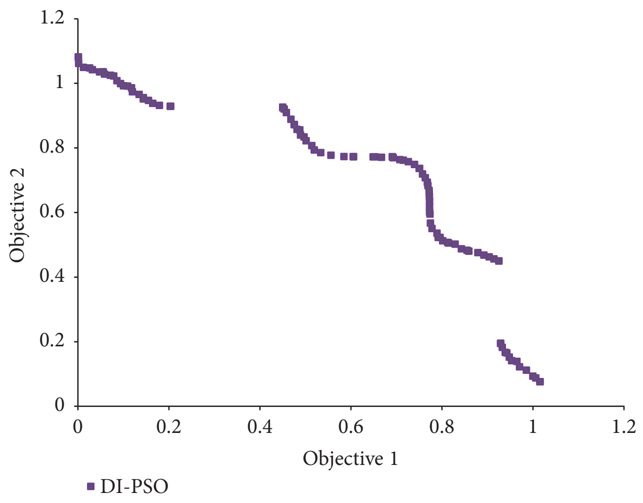

Figure 3: Pareto front obtained for TNK test case.

and localization of the building masses as well as criteria to improve the safety of the resulting structure [39]. In the present work, the building structure is optimized using the following functions:

$$F_{obj\_1} = W = \sum_{i=1}^{Nels} \rho_i A_i l_i \tag{16}$$

where ρ is the specific weight, A is the cross-section area, and l is element length. Nels is the number of elements of the structure.

When a structure is subjected to seismic actions, the design's objective seeks to decrease the dynamic response of the building in order to reduce its damage due to seismic loads. This objective can be accomplished by either minimizing the acceleration taken by the masses of the building or by reducing the seismic energy absorbed by the building structure. According to [39], the input energy (E_i) provided by the earthquake is the work done by the shear base force of the structure (S_{vi}) through soil displacements due to seismic waves. It can be obtained as

$$F_{obj\_2} = E_i = \sum_{i=1}^{Nmode} \left(\frac{\{\phi_i\}^T \{m\}}{\sqrt{M}} \right) \left(\frac{1}{2} M S_{vi}^2 \right) \tag{17}$$

where $\{\phi^i\}$ is the ith normalized mode; $\{m\} = \{m_1, m_2, \ldots, m_{Nmode}\}$ is the vector of masses assigned to each dynamic degree of freedom of the structural model, and $M = \Sigma m_i$ is the total mass of the structure. The seismic design is based on the International Building Code [40], and the following design criteria and stress limit are assumed:

(a) The maximum stress in the columns will be bounded by the maximum allowed combined stress of the columns material. This stress is obtained as

$$\sigma = \frac{P}{A} + \frac{M_x y}{I_x} + \frac{M_y x}{I_y} \tag{18}$$

(b) The drift of i-storey (δ_i) is limited to

$$|\delta_i| \le \frac{1}{400} h_i \tag{19}$$

where h_i is the height between floors.

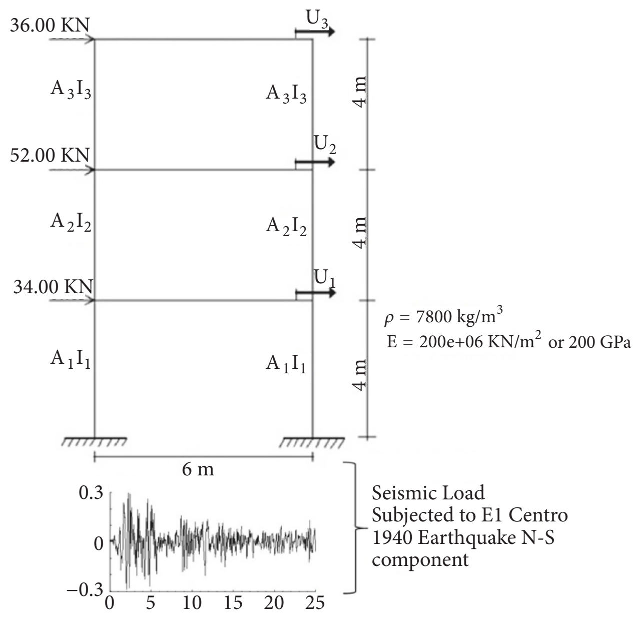
36.00 KN
52.00 KN
34.00 KN
U3
U2
U1
A3I3
A2I2
A1I1
4 m
6 m
ρ = 7800 kg/m3
E = 200e+06 KN/m2 or 200 GPa
Seismic Load Subjected to E1 Centro 1940 Earthquake N-S component

Figure 4: Structural model, elements dimensions, material properties, and loading.

(c) The fundamental period of the example structure should be greater than 0,3 s.

$$T_1 \ge 0{,}3\ \mathrm{s} \tag{20}$$

(d) The stiffness relation between two consecutives floors is bounded by

$$0{,}5 \le \frac{K_{i+1}}{K_i} \le 1 \tag{21}$$

Several techniques have been developed to deal with constraints and to drive the algorithm to the feasible design space. Among them, some can be mentioned [41]: penalization techniques, repair techniques, separation techniques, and hybrid techniques. In this work, the external penalization technique is used to allow the optimization process to start in any region of the design space, making this technique more flexible than the internal penalization technique, where the searching process must start in feasible regions of the design space. The basic idea of the external penalization method is to move solutions from the nonfeasible region to the feasible region adding extra weight (penalizing) to objective function values of the solutions that violate constraints. A detailed description on how to handle constraints can be seen in Annicchiarico *et al.* [42].

3.3. Real Example Application.

3.3. Real Example Application. Figure 4 depicts the structural model of a shear three-story steel building. Rigid floor diaphragms are considered in every building storey while column axial deformations are neglected. Each floor is loaded with a vertical dead and live combinational load of 56 kN/m. The weight of columns was neglected.

In order to show the flexibility and versatility of the dynamically improved particle swarm optimization methodology proposed herein the three-story building is subjected

Table 3: Test cases for the single and multiobjective design of the 3-story shear building.

Design case	Objective(s)	Constraints
1	Min. weight: (F_{obj_1})	Structural and seismic
2	Min. weight: (F_{obj_1}) Min. input energy: (F_{obj_2})	Structural and seismic

to a set of loading cases and analysis. Table 3 collects the different optimization cases. Case 1 deals with the single-objective optimization to obtain a structure with minimum weight. In case 2 the structure is optimized to find out a design able to absorb the maximum seismic input energy without failing and finally the multiobjective optimization case to find out a compromise solution between minimum weight and minimum input seismic energy.

In all cases the structure is subjected to both vertical and lateral loads due to seismic action. For the design case 1 the sectional cross areas (A_i) of the columns were considered as design variables, but in case 2 the inertias (I_i) were used instead. To consider the variability of the mass of each story (mi) and its stiffness (EI_i), the mass of floor i of the structure is obtained as

$$m_i = \frac{(56\,\mathrm{KN/m}) * 6\mathrm{m}}{g} + 2 * (4\mathrm{m}) * A_i * \rho \qquad (22)$$

where g is the gravity acceleration; ρ and E are the specific weight and elasticity module of steel, respectively (ρ= 7800 kg/m^3 and E=200$*10^6$ kN/m^2). The columns are designed as "I" wide flange following the AISC-2005 design criteria while the cross-sectional area (A) and section modulus (S) are expressed in terms of the inertia moment:

$$A = 0{,}80\sqrt{I}$$
$$S = 0{,}78\sqrt[4]{I^3} \qquad (23)$$

By means of a spectral analysis of N-S direction of El Centro earthquake 1940, the seismic lateral load was obtained as shown in Figure 4 (F_1= 34 kN, F_2= 52 kN, and F_3= 36 kN). In real-life problems there is no certainty that the real Pareto frontier can be found. Whether such solution exists or not, it is clear that by using evolutionary and social-cognitive algorithms, supported by many research studies in this matter (see, for example, [22, 43, 44]) the solutions given by these kinds of new methodologies are very close to the exact solution (if they are not in the exact solution) and this solution can be taken as the best known solution of the problem.

In this sense, the proposed methodology herein allows obtaining a high-quality set of compromise solutions and the greater possible certainty, allowing the final user (the decision maker) make the best decision to solve the problem. Also, due to the inherent heuristics in EC algorithms when exploring the design space, the reported results are the best average compromise solution from a set of 20 different runs of the algorithm. The final weight of the building, considering only its minimum weight (case 1) in it design, was 1,5605 kN.

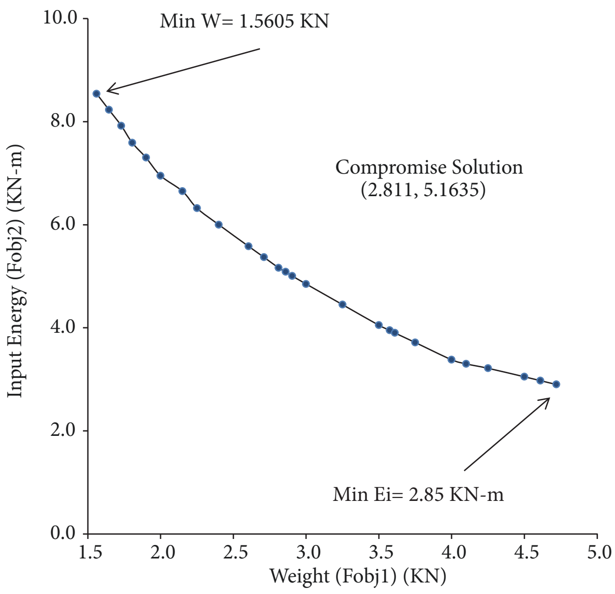

Figure 5: Pareto frontier in design case 2: multiobjective optimization minimum weight-minimum absorbed energy.

Table 4 shows also the cross-sectional areas of the columns of the building.

Figure 5 shows the Pareto frontier obtained in design case 2. It can be observed how the structure minimum weight (W_i) is 1,5605kN, while the minimum input energy (E_i) is 2,85kNm. The inverse relationship between both objectives can also be noted, which means when the weight of the structure (Wi) is reduced, it tends to be more flexible, making the structure more prone to be adversely affected by the seismic loading that causes higher values of the input energy (Ei).

Finally, Table 5 contains the quality metrics (convergence and diversity) for the PF obtained in design case 2. Both values are close to zero which shows the good quality of the solution sought.

The compromise solution in this case, calculated from the Pareto frontier's elbow, is (2,8111 kN; 5,1635 KNm). Aiming for a minimum weight objective could only lead to more damage to the structure, causing higher reparation cost or even worse, and be prone to the structure in risk of collapse.

4. Concluding Remarks

An improved multiobjective methodology, based in an autoadaptive social intelligent algorithm for seismic-structural optimization, was presented and discussed among some of

Table 4: Results for the design case of minimum weight of the 3-story shear building.

Case	$A_1(cm^2)$	$A_2(cm^2)$	$A_3(cm^2)$	Final weight(kN)
1	1020,81	866,56	612,94	1,5605

Table 5: Convergence and diversity metrics for the multiobjective optimization of the 3-story shear building.

Problem	Convergence	Diversity
3-story building	0.01227	0.3271

the new characteristics introduced in the proposed optimization methodology included: a dynamically adapted inertia factor based on particle success index, social and individual particle factors varying according to exploration/exploitation relationship as the searching process of the optimum advance through iterations, dynamic mutation factor, and elitist crossover operator were introduced to promote diversity among swarm particles avoiding the algorithm to get trapped in local optimum. The proposed structural optimization methodology was tested in some test and real problems showing its efficiency and applicability to solve structural-seismic design problems.

It is needed to consider conflicting design objectives to get a better understanding of the behaviour of real structures. Thus, structures found in the real world have to be designed under multiple objectives and constraints. Its optimization considering one goal usually leads to ideal structural designs that do not reproduce their actual observed behaviour. In this sense, the study and development of new tools for optimizing the structural design to get closer to reality, where goals are often contradictory among themselves (as that presented in this work), must be considered an important task, which will help designers to conceive more efficient structures in different scenarios or objectives. The incorporation of this type of tool ensures that the engineering design process would be more reliable and robust.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research work was possible due to support of the Research Council of Simón Bolívar University (DID-USB) and the Venezuelan National Council for Science and Technology (FONACIT).

References

[1] T. Bäck, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, New York, NY, USA, 1996.

[2] H. Beyer and H. Schwefel, “Evolution strategies—a comprehensive introduction,” *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.

[3] I. Rechenberg, “Evolution Strategy,” in *Computational Intelligence: Imitating Life*, J. M. Zurada, R. J. Marks and, and C. Robinson, Eds., IEEE Press, Piscataway, NJ, USA, 1994.

[4] D. E. Goldberg, *Genetics Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, USA, 1989.

[5] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, Germany, 3rd edition, 1996.

[6] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, Mass, USA, 1992.

[7] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, Piscataway, NJ, USA, 1996.

[8] C. Darwin, *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*, John Murray, London, UK, 1859.

[9] Z. H. Ding, M. Huang, and Z. R. Lu, “Structural damage detection using artificial bee colony algorithm with hybrid search strategy,” *Swarm and Evolutionary Computation*, vol. 28, pp. 1–13, 2016.

[10] J. J. Zhu, M. Huang, and Z. R. Lu, “Bird mating optimizer for structural damage detection using a hybrid objective function,” *Swarm and Evolutionary Computation*, vol. 35, pp. 41–52, 2017.

[11] H. Assimi and A. Jamali, “A hybrid algorithm coupling genetic programming and Nelder–Mead for topology and size optimization of trusses with static and dynamic constraints,” *Expert Systems with Applications*, vol. 95, pp. 127–141, 2018.

[12] W. Annicchiarico and M. Cerrolaza, “Identification of the dynamical properties of structures using free vibration data and distributed genetic algorithms,” *Engineering Optimization*, vol. 39, no. 8, pp. 969–980, 2007.

[13] A. Kaveh, “Particle Swarm Optimization,” in *Advances in Metaheuristic Algorithms for Optimal Design of Structures*, Springer, Switzerland, 2016.

[14] M. N. Hadi and M. E. Uz, “Investigating the optimal passive and active vibration controls of adjacent buildings based on performance indices using genetic algorithms,” *Engineering Optimization*, vol. 47, no. 2, pp. 265–286, 2015.

[15] W. Annicchiarico, “Metamodel-assisted distributed genetic algorithms applied to structural shape optimization problems,” *Engineering Optimization*, vol. 39, no. 7, pp. 757–772, 2007.

[16] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann, Burlington, Mass, USA, 2001.

[17] R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in *Proceedings of the 6th International Symposium on Micromachine and Human Science*, pp. 39–43, Nagoya, Japan, October 1995.

[18] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks (ICNN ’95)*, vol. 4, pp. 1942–1948, Perth, Western Australia, November-December 1995.

[19] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, vol. 1, pp. 325–331, Portland, Ore, USA, June 2004.

[20] J. Sun, CH. Lai, W. B. Xu, and Z. Chai, "A novel and more efficient search strategy of quantum behaved particle swarm optimization," in *Proceedings of the 8th International Conference on Adaptive and Natural Comp Algorithms*, pp. 394–403, 2007.

[21] D. Zhou, Y. Li, B. Jiang, and J. Wang, "A novel multiobjective quantum-behaved particle swarm optimization based on the ring model," *Mathematical Problems in Engineering*, vol. 2016, Article ID 4968938, 15 pages, 2016.

[22] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[23] W. D. Annicchiarico, "Multidisciplinary-multiobjective structural design optimization by using dynamical particle swarm algorithm," *J. UCV Eng. School*, vol. 27, pp. 51–64, 2012 (Spanish).

[24] M. M. Millonas, "Swarms, phase transitions, and collective intelligence," in *Artificial Life III*, C. G. Langton, Ed., pp. 417–445, Addison-Welsey, Reading, Mass, USA, 1994.

[25] J. Kennedy, "The behavior of particles," in *Evolutionary Programming VII*, V. W. Porto, N. Saravanan, D. Waagen and, and A. E. Eiben, Eds., Springer, Berlin, Germany, 1998.

[26] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary Programming VII*, V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben, Eds., Springer, Berlin, Germany, 1998.

[27] B. K. Panigrahi, V. R. Pandi, and S. Das, "Adaptive particle swarm optimization approach for static and dynamic economic load dispatch," *Energy Conversion and Management*, vol. 46, pp. 1407–1415, 2008.

[28] W. D. Annicchiarico, "Development and application of an optimization general system using distributed evolutive algorithms and metamodels," Internal report 2018/002-12, Department of Mechanics, Simón Bolívar University, Venezuela, 2018 (Spanish).

[29] Y. Wu, B. Yan, and X. Qu, "Improved chicken swarm optimization method for reentry trajectory optimization," *Mathematical Problems in Engineering*, vol. 2018, Article ID 8135274, 13 pages, 2018.

[30] D. Wu, F. Kong, W. Gao, Y. Shen, and Z. Ji, "Improved chicken swarm optimization," in *Proceedings of the 5th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015*, pp. 681–686, China, June 2015.

[31] F. Van den Bergh, *An analysis of particle swarm optimization [Ph.D. dissertation]*, Faculty of Natural and Agricultural Science, University of Pretoria, South Africa, 2002.

[32] C. Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer, New York, NY, USA, 2002.

[33] C. Coello, A. H. Aguirre, and E. Zitzler, "Evolutionary multicriterion optimization," in *Proceedings of the Third International Conference*, Springer, Berlin, Germany, 2005.

[34] M. Reyes-Sierra and C. A. Coello Coello, "Multi-objective particle swarm optimizers: a survey of the state-of-the-art," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 287–308, 2006.

[35] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: NSGA-II," Technical Report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[36] E. Zitzler, M. Laumanns, and S. Bleuler, *A Tutorial on Evolutionary Multiobjective Optimization*, Springer, Berlin, Germany, 2002.

[37] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[38] F. Fortin and M. Parizeau, "Revisiting the NSGA-II crowding-distance computation," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, Amsterdam, The Netherlands, July 2013.

[39] F. Y. Cheng and D. Li, "Multiobjective optimization of structures with and without control," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 2, pp. 392–397, 1996.

[40] International Code Council, *Uniform Building Code-Struct Engng Design Provisions*, vol. 2, 1997.

[41] C. A. Coello Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," *Computer Methods Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, 2002.

[42] W. D. Annicchiarico, J. Périaux, M. Cerrolaza, and G. Winter, Eds., >*Evolutionary Algorithms and Intelligent Tools in Engineering Optimization*, WIT Press, Southampton, UK, 2005.

[43] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[44] E. Zitzler, M. Laummans, and L. Thiele, "SPEA2: improving the strength pareto evolutionary algorithm," Tech. Rep. 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 2001.