INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING

Int. J. Numer. Meth. Engng 2003; **00**:1–6 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

Handling Constraints using Multiobjective Optimization Concepts

Arturo Hernández Aguirre[1], Salvador Botello Rionda[1] Carlos A. Coello Coello[2*],

Giovanni Lizárraga Lizárraga[1], Efrén Mezura Montes[2]

[1] *Center for Research in Mathematics (CIMAT), Department of Computer Science, Guanajuato, Gto. 36240, MEXICO*

[2] *CINVESTAV-IPN, Evolutionary Computation Group, Depto. de Ingeniería Eléctrica, Sección de Computación, Av. Instituto Politécnico Nacional No. 2508, Col. San Pedro Zacatenco, México, D. F. 07300, MEXICO*

SUMMARY

In this paper, we propose a new constraint-handling technique for evolutionary algorithms which we call Inverted-Shrinkable PAES (IS-PAES). This approach combines the use of multiobjective optimization concepts with a mechanism that focuses the search effort onto specific areas of the feasible region by shrinking the constrained search space. IS-PAES also uses an adaptive grid to store the solutions found, but has a more efficient memory-management scheme than its ancestor (the Pareto Archived Evolution Strategy for multiobjective optimization). The proposed approach is validated using several examples taken from the standard evolutionary and engineering optimization literature. Comparisons are provided with respect to the stochastic ranking method (one of the most competitive constraint-handling approaches used with evolutionary algorithms currently available) and with respect to other four multiobjective-based constraint-handling techniques. Copyright © 2003 John Wiley & Sons, Ltd.

*Correspondence to: Dr. Carlos Coello, PO Box 60326-394, Houston, Texas 77205, USA

Received

Revised

KEY WORDS: evolutionary algorithms, constraint-handling, evolutionary multiobjective optimization, evolutionary optimization

1. INTRODUCTION

Evolutionary Algorithms (EAs) in general (i.e., genetic algorithms [14], evolution strategies [31] and evolutionary programming [12]) are search and optimization techniques inspired on the mechanism of natural selection (i.e., the survival of the fittest). However, EAs are unconstrained optimization techniques and therefore lack an explicit mechanism able to bias efficiently the search towards the feasible region in constrained search spaces. Such a mechanism is highly desirable since most real-world problems have (possibly many) constraints which could be of any type (equality, inequality, linear and nonlinear).

The success of EAs in global optimization has triggered a considerable amount of research regarding the development of mechanisms able to incorporate information about the constraints of a problem into the fitness function of the EA used to optimize it [5, 21]. So far, the most common approach adopted in the evolutionary optimization literature to deal with constrained search spaces is the use of penalty functions [28, 32]. When using a penalty function (external penalty functions are normally adopted with EAs [25]), the amount of constraint violation is used to punish or "penalize" an infeasible solution so that feasible solutions are favored by the selection process. Despite the popularity of penalty functions, they have several drawbacks from which the main one is that they require a careful fine tuning of the penalty factors that indicates the degree of penalization to be applied [32, 5].

Recently, some researchers have suggested the use of multiobjective optimization concepts

to handle constraints in EAs (see for example [5]). This paper introduces a new approach that is based on an evolution strategy that was originally proposed for multiobjective optimization: the Pareto Archived Evolution Strategy (PAES) [18]. Our approach (which is an extension of PAES) can be used to handle constraints both of single- and multiobjective optimization problems. One of the main contributions of our approach is that it does not present the scalability problems of the original PAES, which can easily run out of memory. Besides using Pareto-based selection, our approach uses a secondary population (one of the most common notions of elitism in evolutionary multiobjective optimization), and a mechanism that reduces the constrained search space so that our approach can approach the global optimum more efficiently. This mechanism is mostly responsible for the good performance of our approach, which is the first constraint-handling technique based on multiobjective optimization concepts that provides results competitive with respect to state-of-the-art approaches.

The remainder of this paper is organized as follows. Section 2 gives a formal description of the general problem that we want to solve. Section 3 introduces some basic multiobjective optimization concepts that we will be using throughout the paper. Section 4 describes the previous work related to our own. In Section 5, we describe the main algorithm of IS-PAES. Section 6 provides a comparison of results with respect to stochastic ranking (which is representative of the state-of-the-art in constrained evolutionary optimization) and with respect to other multiobjective-based techniques. Finally, Section 7 draws our conclusions and provides some paths of future research.

2. PROBLEM STATEMENT

We are interested in the general non-linear programming problem in which we want to:

$$\text{Find} \quad \vec{x} \text{ which optimizes } f(\vec{x}) \tag{1}$$

subject to:

$$g_i(\vec{x}) \leq 0, \quad i = 1, \ldots, n \tag{2}$$

$$h_j(\vec{x}) = 0, \quad j = 1, \ldots, p \tag{3}$$

where $\vec{x}$ is the vector of solutions $\vec{x} = [x_1, x_2, \ldots, x_r]^T$, n is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or non-linear).

If we denote with $\mathcal{F}$ to the feasible region and with $\mathcal{S}$ to the whole search space, then it should be clear that $\mathcal{F} \subseteq \mathcal{S}$.

For an inequality constaint that satisfies $g_i(\vec{x}) = 0$, then we will say that is active at $\vec{x}$. All equality constraints h_j (regardless of the value of $\vec{x}$ used) are considered active at all points of $\mathcal{F}$.

3. BASIC CONCEPTS

Since we will be using some multiobjective optimization concepts, it is appropriate to define them first. A multiobjective optimization problem (MOP) has the following the form:

$$\text{minimize} \ [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})] \tag{4}$$

subject to the m inequality constraints:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, n \tag{5}$$

and the p equality constraints:

$$h_j(\vec{x}) = 0 \quad j = 1, 2, \ldots, p \tag{6}$$

where k is the number of objective functions $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$. We call $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ the vector of decision variables. We wish to determine from among the set $\mathcal{F}$ of all vectors which satisfy (5) and (6) the particular set of values $x_1^*, x_2^*, \ldots, x_n^*$ which yield the optimum values of all the objective functions.

3.1. Pareto Optimality

It is rarely the case that there is a single point that simultaneously optimizes all the objective functions of a multiobjective optimization problem. Therefore, we normally look for "trade-offs", rather than single solutions when dealing with multiobjective optimization problems. The notion of "optimality" is therefore, different in this case. The most commonly adopted notion of optimality is that originally proposed by Francis Ysidro Edgeworth [10] and later generalized by Vilfredo Pareto [22]. Although some authors call this notion *Edgeworth-Pareto optimality* (see for example [33]), we will use the most commonly accepted term: *Pareto optimality.*

A vector $\vec{u} = (u_1, \ldots, u_k)$ is said to dominate $\vec{v} = (v_1, \ldots, v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if u is partially less than v, i.e., $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.

For a given multiobjective optimization problem, $\vec{f}(x)$, the Pareto optimal set ($\mathcal{P}^*$) is defined

as:

$$\mathcal{P}^* := \{\vec{x} \in \mathcal{F} \mid \neg\exists\ \vec{x}' \in \mathcal{F} \ \ \vec{f}(\vec{x}') \preceq \vec{f}(\vec{x})\}. \tag{7}$$

Thus, we say that a vector of decision variables $\vec{x}^* \in \mathcal{F}$ is *Pareto optimal* if there does not exist another $\vec{x} \in \mathcal{F}$ such that $f_i(\vec{x}) \leq f_i(\vec{x}^*)$ for all $i = 1, \ldots, k$ and $f_j(\vec{x}) < f_j(\vec{x}^*)$ for at least one j.

In words, this definition says that $\vec{x}^*$ is Pareto optimal if there exists no feasible vector of decision variables $\vec{x} \in \mathcal{F}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. Unfortunately, this concept almost always gives not a single solution, but rather a set of solutions called the *Pareto optimal set*. The vectors $\vec{x}^*$ correspoding to the solutions included in the Pareto optimal set are called *nondominated*. The image of the Pareto optimal set under the objective functions is called *Pareto front*.

4. RELATED WORK

Since our approach belongs to the group of techniques in which multiobjective optimization concepts are adopted to handle constraints, we will briefly discuss some of the most relevant work done in this area. The main idea of adopting multiobjective optimization concepts to handle constraints is to redefine the single-objective optimization of $f(\vec{x})$ as a multiobjective optimization problem in which we will have $m + 1$ objectives, where m is the total number of constraints. Then, we can apply any multiobjective optimization technique [8] to the new vector $\bar{v} = (f(\vec{x}), f_1(\vec{x}), \ldots, f_m(\vec{x}))$, where $f_1(\vec{x}), \ldots, f_m(\vec{x})$ are the original constraints of the problem. An ideal solution $\vec{x}$ would thus have $f_i(\vec{x}) \geq 0$ for $1 \leq i \leq m$ and $f(\vec{x}) \leq f(\vec{y})$ for all feasible $\vec{y}$ (assuming minimization).

Three are the mechanisms taken from evolutionary multiobjective optimization that are more frequently incorporated into constraint-handling techniques:

1. Use of Pareto dominance as a selection criterion. Examples of this type of approach are given in [4, 17, 6].
2. Use of Pareto ranking [14] to assign fitness in such a way that nondominated individuals (i.e., feasible individuals in this case) are assigned a higher fitness value. Examples of this type of approach are given in [26, 27, 5].
3. Split the population in subpopulations that are evaluated either with respect to the objective function or with respect to a single constraint of the problem. This is the selection mechanism adopted in the Vector Evaluated Genetic Algorithm (VEGA) [30]. Examples of this type of approach are given in [34, 23, 7].

In order to sample the feasible region of the search space widely enough to reach the global optima it is necessary to maintain a balance between feasible and infeasible solutions. If this diversity is not reached, the search will focus only in one area of the feasible region. Thus, it will lead to a local optimum solution.

A multiobjective optimization technique aims to find a set of trade-off solutions which are considered good in all the objectives to be optimized. In global nonlinear optimization, the main goal is to find the global optimum. Therefore, some changes must be done to those approaches in order to adapt them to the new goal. Our main concern is that feasibility takes precedence, in this case, over nondominance. Therefore, good "trade-off" solutions that are not feasible cannot be considered as good as bad "trade-off" solutions that are feasible. Furthermore, a mechanism to maintain diversity must normally be added to any evolutionary multiobjective optimization technique.

5. IS-PAES ALGORITHM

All of the approaches discussed in the previous section have drawbacks that keep them from producing competitive results with respect to the constraint-handling techniques that represent the state-of-the-art in evolutionary optimization. In a recent technical report [20], four of the existing techniques based on multiobjective optimization concepts (i.e., COMOGA [34], VEGA [7], MOGA [6] and NPGA [5]) have been compared using Michalewicz's benchmark [21] and some additional engineering optimization problems. Although inconclusive, the results indicate that the use of Pareto dominance as a selection criterion gives better results than Pareto ranking or the use of a population-based approach. However, in all cases, the approaches analyzed are unable to reach the global optimum of problems with either high dimensionality, large feasible regions or many nonlinear equality constraints [20].

In contrast, the approach proposed in this paper uses Pareto dominance as the criterion selection, but unlike the previous work in the area, a secondary population is used in this case. The approach, which is a relatively simple extension of PAES [18] provides, however, very good results, which are highly competitive with those generated with an approach that represents the state-of-the-art in constrained evolutionary optimization.

IS-PAES has been implemented as an extension of the Pareto Archived Evolution Strategy (PAES) proposed by Knowles and Corne [18] for multiobjective optimization. PAES's main feature is the use of an adaptive grid on which objective function space is located using a coordinate system. Such a grid is the diversity maintenance mechanism of PAES and it constitutes the main feature of this algorithm. The grid is created by bisecting k times the function space of dimension d (d is the number of objective functions of the problem. In our case, d is given by the total number of contraints plus one. In other words, $d = n + p + 1$,

where n is the number of inequality constraints, and p is the number of equality constraints. Note that we add one to this summation to include the original objective function of the problem). The control of 2^{kd} grid cells means the allocation of a large amount of physical memory for even small problems. For instance, 10 functions and 5 bisections of the space produce 2^{50} cells. Thus, the first feature introduced in IS-PAES is the "inverted" part of the algorithm that deals with this space usage problem. IS-PAES's fitness function is mainly driven by a feasibility criterion. Global information carried by the individuals surrounding the feasible region is used to concentrate the search effort on smaller areas as the evolutionary process takes place. In consequence, the search space being explored is "shrunk" over time. Eventually, upon termination, the size of the search space being inspected will be very small and will contain the solution desired (in the case of single-objective problems. For multi-objective problems, it will contain the feasible region).

The main algorithm of IS-PAES is shown in Figure 1. Its goal is the construction of the Pareto front which is stored in an external memory (called *file*). The algorithm loops over the generation of h children from a random parent c. If the child is better than the parent, that is, the child dominates its parent, then it is inserted in *file*, and its position is recorded. A child is generated by introducing random mutations to the parent, thus, $h = mutate(c)$ will alter a parent with increments whose standard deviation is governed by Equation 8. Most of **main** and the function **test(h,c,file)** in IS-PAES are devoted to three things: (1) decide whether a new child should be inserted in *file*, and if so, (2) how to make room for the new member and (3) who becomes the new parent. Every g new children created, a new parent is randomly picked from *file* for this purpose. Also, every r children generated, the space is shrunk around the current Pareto front represented by the individuals of the external memory.

Here we introduce the following notation: $x_1 \square x_2$ means x_1 is located in a less populated region of the grid than x_2. The pseudo-code of this function is depicted in Figure 2.

5.1. Inverted "ownership"

IS-PAES handles the population *as part of* a grid location relationship, whereas PAES handles a grid location that *contains* a population relationship. In other words, PAES keeps a list of individuals on either grid location, but in IS-PAES each individual knows its position on the grid. Therefore, building a sorted list of the most densely populated areas of the grid only requires to sort the k elements of the external memory. In PAES, this procedure needs to inspect every location of the grid in order to produce an unsorted list, and from there on, the list is sorted. The advantage of the inverted relationship is clear when the optimization problem has many functions (more than 10), and/or the granularity of the grid is fine, for in this case only IS-PAES is able to deal with any number of functions and granularity level.

5.2. Shrinking the objective space

Shrinkspace(file) is the most important function of IS-PAES since its task is the reduction of the search space. The space is reduced every r number of generations. The pseudo-code of **Shrinkspace(file)** is shown in Figure 3.

In the following we describe the four tasks performed by **shrinkspace**.

- The function **select(file)** returns a list whose elements are the best individuals found in *file*. The size of the list is set to 15% of *maxsize*. Thus, the goal of **select(file)** is to create a list with: a) only the best feasible individuals, b) a combination of feasible and partially feasible individuals, or c) the "most promising" infeasible individuals. The

selection algorithm is shown in Figure 4. Note that *validconstraints* (a list of indexes to the problem constraints) indicates the order in which constraints are tested. The loop steps over the constraints removing only one (the worst) individual for each constraint till there is none to delete (all feasible), or 15% of file size is reached (in other words, 85% of the Pareto set will be generated anew using the best 15% individuals as parents). Also, in order to keep diversity, a new parent is randomly chosen from the less populated region of the grid after placing on it g new individuals.

- The function **getMinMax(file)** takes the list *list* (last step in Figure 4) and finds the extreme values of the decision variables represented by those individuals. Thus, the vectors $\underline{\boldsymbol{x}}_{pob}$ and $\overline{\boldsymbol{x}}_{pob}$ are found.
- Function **trim**($\underline{\boldsymbol{x}}_{pob}$, $\overline{\boldsymbol{x}}_{pob}$) shrinks the feasible space around the potential solutions enclosed in the hypervolume defined by the vectors $\underline{\boldsymbol{x}}_{pob}$ and $\overline{\boldsymbol{x}}_{pob}$. Thus, the function **trim**($\underline{\boldsymbol{x}}_{pob}$, $\overline{\boldsymbol{x}}_{pob}$) (see Figure 5) determines the new boundaries for the decision variables. The value of β is the percentage by which the boundary values of either $x_i \in X$ must be reduced such that the resulting hypervolume H is a fraction α of its previous value. The function **trim** first finds in the population the boundary values of each decision variable: $\overline{x}_{pob,i}$ and $\underline{x}_{pob,i}$. Then the new vectors $\overline{x}_i$ and $\underline{x}_i$ are updated by $deltaMin_i$, which is the reduction in each variable that in the overall reflects a change in the volume by a factor β. In IS-PAES all objective variables are reduced at the same rate β, therefore, β can be deduced from α as discussed next. Since we need the new hypervolume be a fraction α of the previous one,

$$H_{\text{new}} \geq \alpha H_{\text{old}}$$

$$\prod_{i=1}^{n}(\overline{x}_i^{t+1} - \underline{x}_i^{t+1}) = \alpha \prod_{i=1}^{n}(\overline{x}_i^{t} - \underline{x}_i^{t})$$

Either x_i is reduced at the same rate β, thus

$$\prod_{i=1}^{n}\beta(\overline{x}_i^{t} - \underline{x}_i^{t}) = \alpha \prod_{i=1}^{n}(\overline{x}_i^{t} - \underline{x}_i^{t})$$

$$\beta^n \prod_{i=1}^{n}(\overline{x}_i^{t} - \underline{x}_i^{t}) = \alpha \prod_{i}^{n}(\overline{x}_i^{t} - \underline{x}_{i=1}^{t})$$

$$\beta^n = \alpha$$

$$\beta = \alpha^{\frac{1}{n}}$$

In short, the new search interval of each decision variable x_i is adjusted as follows (the complete algorithm is shown in Figure 3):

$$width_{new} \geq \beta \times width_{old}$$

It should be clear that the value of α has an important impact on the performance of IS-PAES because it controls the shrinking speed. In order to determine a range within which we could set this parameter for a large variety of problems, we studied the effect of α on the performance of our algorithm for each of the 11 test functions included in Appendix A. From analyzing this effect, we found that in all cases, a range of α between 85% and 97% was always able to generate the best possible solutions to each problem. Values smaller than 0.80 make the algorithm prone to converge to local minima. Values of α too near to 100% slow down convergence, although they increase the probability of success. In order to avoid a fine tuning of α dependent of each test function, we decided to set its value to 0.90, which we considered as a good compromise based on our analysis.

As we will see later on, this value of α provided good results in all the problems solved. Note that also the parameter r (see Figure 1), which controls the shrinkspace rate, plays an important role in the algorithm. To set the value of r, we performed a similar analysis to the one previously described for α. In this analysis, we related the behavior of r with that of α and with the performance of IS-PAES. Our results indicated that a value of $r = 1$ provided convergence to the optimum in most of the problems (in a few cases, a value of $r = 2$ turns out to be better). Thus, we used $r = 1$ in all the experiments reported in this paper.

The variable *slack* is calculated once every new search interval is determined (usually set to 5% of the interval). The role of slack is simply to prevent (upto some extend) against fast decreasing rates of the search interval.

- The last step of **shrinkspace()** is a call to **adjustparameters(file)**. The goal is to re-start the control variable σ through:

$$\sigma_i = (\overline{x}_i - \underline{x}_i)/\sqrt{n} \quad i \in (1, \ldots, n) \tag{8}$$

This expression is also used during the generation of the initial population. In that case, the upper and lower bounds take the initial values of the search space indicated by the problem. The variation of the mutation probability follows the exponential behavior suggested by Bäck [2].

Elitism

A special form of elitism is implemented in IS-PAES to prevent the lost of the best individual. Elitism is implemented as follows: the best individual of the generation is marked and only replaced by another one if it is in the feasible region and with better objective function value.

6. COMPARISON OF RESULTS

We have validated our approach with several problems used as a benchmark for evolutionary algorithms (see [21]) and with several engineering optimization problems taken from the standard literature. In the first case, our results are compared against a technique called "stochastic ranking" [29], which is representative of the state-of-the-art in constrained evolutionary optimization. Stochastic ranking uses a static penalty function and sorts the population based on the penalized value of each individual. However, based on an user-defined parameter called Pf, two individuals can be sorted based only their objective function value, regardless of feasibility. This simple mechanism helps the approach to maintain feasible and infeasible solutions during all the process. The motivation of the approach is to try to balance the effect of the value of the objective function and the value of the penalization in the fitness assignment mechanism. Stochastic ranking has been found to be equally good or even better in some cases than the homomorphous maps of Koziel and Michalewicz [19], which has been considered since its inception, as a landmark in constraint-handling methods used with evolutionary algorithms.

We will also provide comparisons with respect to four constraint-handling techniques based on multiobjective optimization concepts, so that there can be a more clear idea of the effectiveness of our proposal. The four approaches selected for this comparative study are:

1. **COMOGA**[†]: Proposed by Surry & Radcliffe [34]. This approach uses a combination of the Vector Evaluated Genetic Algorithm (VEGA) [30] and Pareto Ranking to handle constraints in an approach called COMOGA (Constrained Optimization by Multi-

[†]Dr. Patrick Surry helped us to develop our own C version of COMOGA.

Objective Genetic Algorithms). In this technique, individuals are ranked depending of their sum of constraint violation (number of individuals dominated by a solution). However, the selection process is based not only on ranks, but also on the fitness of each solution. COMOGA uses a non-generational GA and extra parameters defined by the user (e.g., an parameter called ϵ is used to define the change rate of P_{cost}). One of these parameters is P_{cost}, that sets the rate of selection based on fitness. The remaining $1 - P_{cost}$ individuals are selected based on ranking values. P_{cost} is defined by the user at the begining of the process and it is adapted during the evolutionary process using as a basis the percentage of feasible individuals that one wishes to have in the population.

2. **VEGA**: Proposed by Coello [7]. This approach uses a population-based approach similar to VEGA [30] to handle constraints in single-objective optimization problems. At each generation, the population was split into $m + 1$ subpopulations of equal fixed size, where m is the number of constraints of the problem. The additional subpopulation handles the objective function of the problem and the individuals contained within it are selected based on the unconstrained objective function value. The m remaining subpopulations take one constraint of the problem each as their fitness function. The aim is that each of the subpopulations tries to reach the feasible region corresponding to one individual constraint. By combining these different subpopulations, the approach will reach the feasible region of the problem considering all of its constraints.
3. **MOGA**: Proposed by Coello [6]. This approach uses a Pareto dominance-based selection scheme to handle constraints in a genetic algorithm. This is an application of Fonseca and Fleming's Pareto ranking process [13] (called Multi-Objective Genetic Algorithm, or MOGA) to constraint-handling. In this approach, feasible individuals are always

ranked higher than infeasible ones. Based on this rank, a fitness value is assigned to each individual. This technique also includes a self-adaptation mechanism that avoids the usual empirical fine-tuning of the main genetic operators.

4. **NPGA**: Proposed by Coello and Mezura [5]. This is a version of the Niched-Pareto Genetic Algorithm (NPGA) [16] used to handle constraints in single-objective optimization problems. The NPGA is a multiobjective optimization approach in which individuals are selected through a tournament based on Pareto dominance. However, unlike the NPGA, Coello and Mezura's approach does not require niches (or fitnes sharing [9]) to maintain diversity in the population. The NPGA is a more efficient technique than traditional multiobjective optimization algorithms, since it does not compare every individual in the population with respect to each other (as in traditional Pareto ranking), but uses only a sample of the population to estimate Pareto dominance.

6.1. Examples

For the first part of our comparative study, we used eleven test functions described in [29, 21] and included in Appendix A for completeness. These functions contain characteristics that are representative of what can be considered "dificult" global optimization problems for an evolutionary algorithm.

To get a better idea of the difficulty of solving each of these problems, a ρ metric (as suggested by Koziel and Michalewicz [19]) was computed using the following expression:

$$\rho = |F|/|S| \tag{9}$$

where $|F|$ is the number of feasible solutions and $|S|$ is the total number of solutions randomly

generated. In this work, we generated $S = 1,000,000$ random solutions. The different values of ρ for each of the test functions chosen are shown in Table I.

The following parameters were adopted for IS-PAES in all the experiments reported next: $maxsize = 200$, $listsize = 15\%$ of maxsize, $\alpha = 90\%$, $r = 1$, $g = 10$, $slack = 5\%$. IS-PAES was implemented using Borland C++ (under Windows) in a PC with a Pentium IV processor running at 1.7GHz and with 1 Gbyte of RAM. The maximum number of fitness function evaluations was set to 350,000 for all the algorithms compared. This is the number of evaluations used in [29], and such a value was adopted in order to allow a fair comparison with respect to stochastic ranking which is the most competitive constraint-handling technique developed for evolutionary algorithms to date.

The parameters for COMOGA were: population size = 200, crossover rate = 1.0, mutation rate = 0.05, desired proportion of feasible solutions = 10%, and $\epsilon = 0.01$. The parameters for VEGA were: population size = 350, maximum number of generations = 1000, crossover rate = 0.6, mutation rate = 0.05, tournament size = 5. The parameters for NPGA were: population size = 350, number of generations = 1000, crossover rate = 0.6, mutation rate = 0.05, size of sample of the population = 10, selection ratio = 0.8. The parameters for MOGA were: population size = 350, maximum number of generations = 1000, crossover rate = 0.6, mutation rate = 0.05. In all cases, a binary gray-coded representation with two-point crossover and uniform mutation was adopted. Also, the total number of fitness function evaluations was, in all cases, of 350,000.

6.2. Discussion of Results

The results obtained by each method are shown in Tables II, III, IV, V, VI and VII. Note that for each of the approaches compared (i.e., COMOGA, VEGA, NPGA, MOGA, IS-PAES and stochastic ranking), 30 independent runs were performed, of 350,000 fitness function evaluations each. Also note that comparisons with respect to stochastic ranking are only indirect and based on the statistical results reported in [29], and taking advantage of the fact that the authors report results for all the test functions used in the first comparative study included in this paper. Note however that all the other methods compared were implemented by the authors.

From Tables VI and VII we can see that the proposed approach is highly competitive.[‡] The discussion of results for each test function is provided next:

For **g01**, the best solution found by IS-PAES was: $\vec{x}$ = {1, 0.999999939809, 0.999997901977, 1, 0.999981406123, 1, 1, 0.999999242667, 0.999981194574, 2.99987534752, 2.99995011286, 2.99993014684, 0.999982112914} with $F(\vec{x})$ = -14.99968877 (see Table VI). The constraint values of this solution are: $g_i(\vec{x})$ = $\{-0.000174660002, -0.000198701686, -0.000124056728, -5.00012465248, -5.00004940561, -5.00005306898, -0.000106058603, -4.98871399999 \times 10^{-5}, -4.95330679999 \times 10^{-5}\}$. In this case, IS-PAES was less consistent than stochastic ranking in finding the global optimum (see Table VII for the results of stochastic ranking), mainly because the approach was trapped in a local optimum in which $F(\vec{x}) = -13$ during 20% of the runs. However, IS-PAES had a better performance than any of the other multiobjective-based methods. COMOGA is clearly the

[‡]Note that in all cases, we assumed minimization problems. That is the reason why some values of $f(\vec{x})$ are negative in our tables but the global optimum is a positive value.

approach with the worst performance in this test function (see Table II). MOGA was the best of the multiobjective-based methods (see Table V), only below IS-PAES.

For **g02** the best solution found by IS-PAES was: $\vec{x} = \{3.14860401788, 3.10915903011, 3.08909341555, 3.05835689132, 3.04000196011, 3.00100530894, 2.94955289769, 2.94207158769, 0.49907406319, 0.486231653274, 0.49055938302, 0.492879188045, 0.481722447567, 0.471623533316, 0.452037376504, 0.442565813637, 0.451211591495, 0.437863945589, 0.444359423833, 0.437834075871\}$ with $F(\vec{x}) = -0.803375563$ (see Table VI). The constraint values of this solution are: $g_i(\vec{x}) = \{-0.000548020359981, -120.074192395\}$. As we can see, the best result found by stochastic ranking was better than the best result found by IS-PAES (see Table VII). However, the statistical performance measures of IS-PAES were better (particularly the standard deviation which is significantly lower), which seems to indicate that our approach had more robustness in this problem. Regarding the other four multiobjective-based approaches implemented, NPGA had the best performance (see Table IV), but its results were poorer than those produced by IS-PAES. VEGA was the approach with the worst performance in this test function (see Table III).

The best solution found by IS-PAES for **g03** was: $\vec{x} = \{0.316703041297, 0.315651505081, 0.316134923616, 0.316278204068, 0.31704736156, 0.3153235498, 0.316606164753, 0.316219548386, 0.316085870843, 0.316381756491\}$ with $F(\vec{x}) = -1.00047671446$ (see Table VI). The constraint values of this solution are: $g_i(\vec{x}) = \{-2.21822119784 \times 10^{-7}\}$. In can be clearly seen in this case that both IS-PAES and stochastic ranking had an excellent performance (see Table VII). In this case, NPGA was again the best among the other multiobjective-based approaches implemented (see Table IV), but its results were poorer than those produced with IS-PAES. VEGA was again the approach with the worst performance in this case (the approach was

never able to produce a feasible solution in any of the runs performed) (see Table III).

The best solution found by IS-PAES for **g04** was: $\vec{x}$ = {78, 33.00000002, 29.99525605, 45, 36.77581285 } with $F(\vec{x}) = -30665.53867$ (see Table VI). The constraint values of this solution are: $g_i(\vec{x}) = \{-4.56757594941 \times 10^{-9}, -91.9999999954, -11.1594996911, -8.8405003089, -4.9999999992, -8.04911332031 \times 10^{-10}\}$. The behavior of IS-PAES in this test function was practically the same as stochastic ranking (see Table VII). In this case, NPGA was also the best among the other multiobjective-based approaches (see Table IV), but MOGA was not too far from NPGA in terms of performance (see Table V). However, neither MOGA or NPGA were better than IS-PAES. COMOGA occupied the last place in terms of performance for this test function (see Table II).

For **g06**, the best solution found by IS-PAES was: $\vec{x}$ = {14.0950000092, 0.842960808844} with $F(\vec{x})$ = -6961.813854 (see Table VI). The constraint values of this solution are: $g_i(\vec{x}) = \{-4.15492107386 \times 10^{-9}, -1.42450804486 \times 10^{-8}\}$. Note that both approaches (i.e., stochastic ranking and IS-PAES) reached the global optimum in this case, but IS-PAES was more consistent, with very small variations in the results and a much lower standard deviation than stochastic ranking (see Table VII). In this case, MOGA was the best among the other multiobjective-based approaches (see Table V), and the NPGA was the second best (see Table IV). However, none of these two approaches (NPGA or MOGA) were better than IS-PAES. COMOGA occupied again the last place in terms of performance (see Table II).

Stochastic ranking was clearly better in all aspects than IS-PAES for **g07** (see Table VII). The best solution found by IS-PAES was: $\vec{x}$ = {2.16996489702, 2.36701436984, 8.76882720318, 5.07418756668, 0.943992761955, 1.32027308617, 1.31870032997, 9.82673763033, 8.26988778617, 8.36187863755} with $F(\vec{x}) = 24.33817628$ (see Table VI). The constraint values of

this solution are: $g_i(\vec{x}) = \{-0.000530879660006, -0.000896337350003, -8.78072999289 \times 10^{-6}, -0.0453044976875, -0.00211782207532, -0.000269286602308, -6.31896729469, -49.9668859294\}$. NPGA was the best among the other multiobjective-based approaches for **g07** (see Table IV), and MOGA was the second best (see Table V). However, none of these two approaches (NPGA or MOGA) were better than IS-PAES. COMOGA occupied again the last place in terms of performance for this test function (see Table II).

For **g08**, the best solution found by IS-PAES was: $\vec{x} = \{1.227971353, 4.245373368\}$ with $F(\vec{x}) = -0.095825041$ (see Table VI). The constraint values of this solution are: $g_i(\vec{x}) = \{-1.73745972421, -0.167763263276\}$. Both IS-PAES and stochastic ranking had the same performance in this test function (see Table VII). In this case, MOGA was the best among the other multiobjective-based approaches (see Table V), and VEGA and NPGA shared the second place (see Tables III and IV). COMOGA occupied again the last place in terms of performance for this test function (see Table II). Note however that in this case all approaches tried had an acceptable performance.

Both stochastic ranking and IS-PAES reached the global optimum for **g09**, but IS-PAES had better statistical measures (see Tables VII and VI). The best solution found by IS-PAEs was: $\vec{x} = \{2.33155464246, 1.95170473767, -0.4777639392, 4.36501371674, -0.627393647911, 1.04100115201, 1.59671155682\}$ with $F(\vec{x}) = 680.630601799$. The constraint values of this solution are: $g_i(\vec{x}) = \{-0.000156311060875, -252.549012109, -119.289298995, -8.37410965562 \times 10^{-5}\}$. From the other multiobjective-based approaches, NPGA was the best (see Table IV), and MOGA placed second (see Table V). However, none of these two approaches (NPGA or MOGA) were better than IS-PAES. Once again, COMOGA occupied the last place in terms of performance for this test function (see Table II).

Except for the best solution found (which is better for stochastic ranking), the statistical measures of IS-PAES are better than those of stochastic ranking for **g10** (see Tables VI and VII). The best solution found by IS-PAEs was: $\vec{x} = \{105.6345328, 1179.227593, 6070.09281, 122.497943, 257.1979828, 277.4889774, 265.2967614, 357.197398\}$ with $F(\vec{x}) = 7062.019117$. The constraint values of this solution are: $g_i(\vec{x}) = \{3.71167275 \times 10^{-5}, -5.36934575 \times 10^{-5}, -0.00013458717, -7.75297849997, -8.32504183682, -45.013847191\}$. In this case, MOGA was the best of the other multiobjective-based approaches tried (see Table V), and NPGA placed second (see Table IV). However, none of these two approaches (NPGA or MOGA) were better than IS-PAES. VEGA was the worst performer in this test function (see Table III).

For **g11** both stochastic ranking and IS-PAES had a very good performance (see Tables VI and VII). The best solution found by IS-PAES was: $\vec{x} = \{$-0.707871504966, 0.501171903771$\}$ with $F(\vec{x}) = 0.74991153713$. The constraint values of this solution are: $g_i(\vec{x}) = \{-1.01637706393 \times 10^{-5}\}$. In this case, MOGA was again the best of the other multiobjective-based approaches tried (see Table V), and NPGA placed second (see Table IV) (very close to MOGA). All the approaches had a very good performance in this case, with VEGA being the worst performer (see Table III).

Finally, for **g13**, stochastic ranking had a better performance than IS-PAES, in terms of all the statistical measures reported (see Tables VII and VI). The best solution found by IS-PAES was: $\vec{x} = \{$-1.66625191713, 1.53633838622, 1.91898897695, 0.748626384518, 0.787660303986$\}$ with $F(\vec{x}) = 0.0552048784108$. The constraint values of this solution are: $g_i(\vec{x}) = \{3.98920742766 \times 10^{-11}, -1.43917028495 \times 10^{-11}, -6.0520770646 \times 10^{-11}\}$. Note that in this case, none of the other multiobjective-based approaches tried were able to converge to the feasible region of the problem (see Tables II, III, IV, and V). IS-PAES, however, was able

to reach the feasible region in all of the 30 runs performed and it was able to converge to a distance of 3% from the global optimum.

In general, we can see that IS-PAES was superior to any of the other multiobjective-based approaches implemented in all of the test functions tried. MOGA and NPGA competed for the second best performer, but had obvious difficulties with most of the test functions. In fact, these two approaches were not able to reach the global optimum in most of the problems tried, unlike IS-PAES. Also, COMOGA and VEGA competed for the worst performer. Our main conclusion is that Pareto-based techniques perform better than population based techniques (such as VEGA) and hybrids (such as COMOGA), which is something that we had already found in some previous work [20].

As we indicated before, we used 350,000 fitness function evaluations for our experiments, so that we could compare results with respect to stochastic ranking [29]. However, it is also interesting to analyze the correlation between number of iterations and accuracy of our proposed approach. Thus, we performed some experiments to determine, for each of the previously indicated test functions, how many iterations were required by IS-PAES to converge to the global optimum (or the best known solution). Figures 6 and 7 show in the x-axis the distance to the optimum (expressed as a percentage) and on the y-axis the number of fitness function evaluations. Note in Figure 6 that **g06** and **g09** are the two test functions that are more difficult to IS-PAES. However, for the set of test functions included in Figure 6 (i.e., **g04**, **g06**, **g08**, **g09** and **g11**), IS-PAES is able to converge to a distance of 1% of the global optimum using only 55,000 fitness function evaluations. This first set, is composed of "easy" test functions. In Figure 7, we can see that the second set of test functions (i.e., **g01**, **g02**, **g03**, **g07**, **g10**, and **g13**) requires of the full 350,000 fitness function evaluations performed

in order to converge to a distance of 1% of the global optimum. Obviously, this is the set of "difficult" test functions. It is also worth noticing that IS-PAES cannot converge to a distance of 1% of the global optimum for **g13**, which is the most difficult function used. However, it converges to a distance of 3% of the global optimum, whereas none of the other methods based on multiobjective optimization concepts adopted for our comparative study was able to reach the feasible region.

Figure 8 is used to illustrate the consistency of IS-PAES. First, it is important to indicate that IS-PAES was able to converge to the feasible region in all of the 30 runs performed per function. For **g01**, **g03**, **g04**, **g06**, **g08**, **g09**, and **g11**, IS-PAES was able to converge to a distance of 1% from the global optimum in the 30 runs performed. For **g07**, 29 of the 30 runs performed converged to a distance of 1% from the global optimum and the other one converged to a distance of 2% from the global optimum. For **g02**, IS-PAES was able to converge to 1% from the global optimum in 15 of the 30 runs performed. However, note that in 29 of the 30 runs performed, IS-PAES converged to a 2% distance from the global optimum and in the remaining run, it converged to a distance of 3% from the global optimum. For **g10**, IS-PAES was able to converge to a 1% distance from the global optimum only in 6 of the 30 runs performed. However, in 21 runs, it converged to a distance of 2% from the global optimum and in the remaining runs, it converged to a distance of 4% from the global optimum. The most difficult function is **g13**, in which IS-PAES was able to converge to a distance of 3% from the global optimum only in 2 of the runs performed.

To give an indication of computational cost of IS-PAES (this time varies according to the problem being solved), the "easy"[§] test functions such as **g08** take between 5 and 10 minutes

[§] Difficulty in this case is defined in terms of the cost of evaluating the objective function and the constraints

for 30 runs (using 350,000 fitness function evaluations per run). That would be about 20 seconds per run. For a "difficult" problem such as **g10** takes between 25 and 30 minutes for 30 runs (using 350,000 fitness function evaluations per run). That is approximately, one minute per run. Regarding the other multiobjective-based approaches implemented, COMOGA is the slowest, taking, on average, 2 minutes per run when solving **g08**, and 3 minutes per run when solving **g10** (this additional time taken by COMOGA has to do with its nongenerational nature and the ranking of solutions that it requires). VEGA takes, on average, 10 seconds per run when solving **g08** and 42 seconds per run when solving **g10**. NPGA takes, on average, 58 seconds per run when solving **g08** and 88 seconds per run when solving **g10** (the time difference of NPGA has to do with the use of Pareto dominance checkings, which have an $O(kM^2)$ complexity, where k is the number of objectives and M is the population size). Finally, MOGA takes, on average, 15 seconds per run when solving **g08** and 46 seconds per run when solving **g10**. VEGA and MOGA are the fastest algorithms, but they do not provide results as good as IS-PAES. Computational times for stochastic ranking are not available, but, most likely, they will be slower than IS-PAES due to the fact that stochastic ranking was implemented in MATLAB. However, if stochastic ranking is implemented in C/C++, it may take approximately the same CPU time as IS-PAES, because it does not involve any algorithmic process that is significantly more or less expensive than those included in IS-PAES. Note that in the evolutionary computation literature, it is more common to report fitness function evaluations (see for example [19, 29]), because it is well-known that any evolutionary algorithm has a low computational complexity (its computational complexity is obtained by multiplying the population size times the number of iterations performed) and most of the computational time taken by a typical run is normally spent evaluating the fitness function.

Since none of the approaches evaluated performs any computationally expensive operation in the fitness function, we preferred to perform the comparison of approaches assuming the same number of fitness function evaluations for all of them.

6.3. Optimization of a 49-bar plane truss

The first engineering optimization problem chosen is the optimization of the 49-bar plane truss shown in Figure 9. The goal is to find the cross-sectional area of each member of the truss, such that the overall weight is minimized, subject to stress and displacement constraints. The weight of the truss is given by $F(\vec{x}) = \sum_{j=1}^{49} \gamma A_j L_j$, where A_j is the cross-sectional area of the j_{th} member, L_j is the corresponding length of the bar, and γ is the volumetric density of the material. We used a catalog of *Altos Hornos de México, S.A.*, with 65 entries for the cross-sectional areas available for the design. Other relevant data are the following: Young modulus = 2.1×10^6 kg/cm^2, maximum allowable stress = 3500.00 kg/cm^2, $\gamma = 7.4250 \times 10^{-3}$ kg/cm^3, and a horizontal load of 4994.00 kg applied to the nodes: 3, 5, 7, 9, 12, 14, 16, 19, 21, 23, 25 y 27. We solved this problem for two cases:

1. **Case 1. Stress constraints only:** Maximum allowable stress = 3500.00 kg/cm^2. A total of 49 constraints, thus 50 objective functions.
2. **Case 2. Stress and displacement constraints:** Maximum allowable stress = 3500.00 kg/cm^2, maximum displacement per node = 10 cm. There is a total of 72 constraints, thus 73 objective functions.
3. **Case 3. Real-world problem:** The design problem considers traction and compression stress on the bars, as well as their proper weight. Maximum allowable stress = 3500.00 kg/cm^2, maximum displacement per node =10 cm. A total of 72 constraints, thus 73

objective functions.

The average result of 30 runs for each case are shown in Tables VIII, IX and X. We compare IS-PAES with previous results reported by Botello *et al.* [3] using other heuristics with a penalty function [28] (SA: Simulated Annealing, GA50: Genetic Algorithm with a population of 50, and GSSA: General Stochastic Search Algorithm with populations of 50 and 5).

We can clearly see that in all the cases tried, IS-PAES produced the lowest average weight.

6.4. Optimization of a 10-bar Plane Truss

The second engineering optimization problem chosen is the optimization of the 10-bar plane truss shown in Figure 10. We want to find the coss-sectional area of each bar of this truss such that its weight is minimized, subject to stress and displacement constraints. The weight of the truss is given by:

$$F(\vec{x}) = \sum_{j=1}^{10} \gamma A_j L_j \tag{10}$$

where: $\vec{x}$ is a candidate solution, A_j is the cross-sectional area of the jth member. L_j is the length of member j and γ is the volumetric weight of the material.

The maximum allowable displacement for each node (vertical and horizontal) is assumed as 5.08 cm. There are 10 stress constraints and 8 displacement constraints in total. The maximum and minimum allowable value for the cross-sectional areas are 0.5062 cm^2 and 999.0 cm^2, respectively. The remaining assumed data are: Young's modulus $E = 7.3\times10^5$ kg/cm^2, maximum allowable stress = 1742.11 kg/cm^2, $\gamma = 7.4239\times10^{-3}$ kg/cm^3, and a vertical load of -45454.0 kg applied at nodes 2 and 4.

Table XI shows the minimum value found for this problem by different heuristic algorithms

[3]: GSSA (general stochastic search algorithm with a population size of five, crossover rate of zero, and mutation rate $0 \ldots 10/(\text{number\_of\_bars})$, and simulated annealing with $\alpha = 1.001$), VGA (variable-length genetic algorithm of Rajeev and Krishnamoorthy [24], with population size of 50), MC (Monte-Carlo annealing algorithm of Elperin [11]), SAARSR (Simulated Annealing with Automatic Reduction of Search Range, proposed by Tzan and Pantelides [35]), ISA (Iterated Simulated Annealing, of Ackley [1], and SSO (State Space Optimal [15]).

We can see in Table XI that IS-PAES found better results than any of the other methods. Note that MC found a solution with a lower weight than IS-PAES, but such a solution violates stress and displacement constraints, as can be seen in Tables XII and XIII.

7. CONCLUSIONS AND FUTURE WORK

We have introduced a constraint-handling approach that combines multiobjective optimization concepts with an efficient reduction mechanism of the search space and a secondary population. We have shown how our approach overcomes the scalability problem of the original PAES (which was proposed exclusively for multiobjective optimization) from which it was derived, and we also showed that the approach is highly competitive with respect to the state-of-the-art technique in the area and with respect to other multiobjective-based constraint-handling techniques.

The main contribution of the proposed approach is that this is the first constraint-handling technique based on Pareto dominance that provides competitive results with respect to stochastic ranking (which is currently the best constraint-handling techniques used with evolutionary algorithms reported in the literature). IS-PAES not only was able to match many of the results of stochastic ranking, but was also more consistent (in terms of

statistical performance) than stochastic ranking. Additionally, the main advantage of IS-PAES with respect to stochastic ranking is that the former can be extended for multiobjective optimization problems in a more natural way (in fact, this is part of our future work and some preliminary results indicate its viability). The application of IS-PAES to multiobjective optimization problems is straightforward, because even single-objective problems are handled as multiobjective problems by the approach. In contrast, stochastic ranking would require a re-definition that considers the use of Pareto dominance in the selection of solutions in order to work with multiobjective optimization problems.

Another aspect that we want to explore in the future is the elimination of all of the parameters of IS-PAES using on-line or self-adaptation. This, however, may interfere with the mechanism used to reduce the search space and therefore requires a careful study.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers whose comments greatly helped us to improve the contents of this paper.

The first author acknowledges support from CONACyT project I-39324-A. The second author acknowledges support from CONACyT project No. 34575-A. The third author acknowledges support from the NSF-CONACyT project No. 32999-A.

REFERENCES

1. D. Ackley. An empirical study of bit vector function optimization. In Lawrence Davis, editor, *Genetic Algorithms and Simulated Annealing*, pages 170–271. Morgan Kaufmann Publishers, Los Altos, California, 1987.

2. Thomas Bäck. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, New York, 1996.
3. Salvador Botello, José Luis Marroquín, Eugenio Oñate, and Johan Van Horebeek. Solving Structural Optimization problems with Genetic Algorithms and Simulated Annealing. *International Journal for Numerical Methods in Engineering*, 45(8):1069–1084, July 1999.
4. Eduardo Camponogara and Sarosh N. Talukdar. A Genetic Algorithm for Constrained and Multiobjective Optimization. In Jarmo T. Alander, editor, *3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA)*, pages 49–62, Vaasa, Finland, August 1997. University of Vaasa.
5. Carlos A. Coello Coello and Efrén Mezura-Montes. Handling Constraints in Genetic Algorithms Using Dominance-Based Tournaments. In I.C. Parmee, editor, *Proceedings of the Fifth International Conference on Adaptive Computing Design and Manufacture (ACDM 2002)*, volume 5, pages 273–284, University of Exeter, Devon, UK, April 2002. Springer-Verlag.
6. Carlos A. Coello Coello. Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering and Environmental Systems*, 17:319–346, 2000.
7. Carlos A. Coello Coello. Treating Constraints as Objectives for Single-Objective Evolutionary Optimization. *Engineering Optimization*, 32(3):275–308, 2000.
8. Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.
9. Kalyanmoy Deb and David E. Goldberg. An Investigation of Niche and Species Formation in Genetic Function Optimization. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, San Mateo, California, June 1989. George Mason University, Morgan Kaufmann Publishers.
10. F. Y. Edgeworth. *Mathematical Physics.* P. Keagan, London, England, 1881.
11. T. Elperin. Monte-carlo structural optimization in discrete variables with annealing algorithm. *International Journal for Numerical Methods in Engineering*, 26:815–821, 1988.
12. Lawrence J. Fogel. *Artificial Intelligence through Simulated Evolution. Forty Years of Evolutionary Programming.* John Wiley & Sons, Inc., New York, 199.
13. Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. University

of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

14. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.
15. Edward J. Haug and Jasbir S. Arora. *Applied Optimal Design: Mechanical and Structural Systems.* Wiley, New York, 1979.
16. Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey, June 1994. IEEE Service Center.
17. F. Jiménez, A.F. Gómez-Skarmeta, and G. Sánchez. How Evolutionary Multi-objective Optimization can be used for Goals and Priorities based Optimization. In E. Alba, F. Fernández, J.A. Gómez, F. Herrera, J.I. Hidalgo, J. Lanchares, J.J. Merelo, and J.M. Sánchez, editors, *Primer Congreso Español de Algoritmos Evolutivos y Bioinspirados (AEB'02)*, pages 460–465, Mérida España, 2002. Universidad de la Extremadura, España.
18. Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.
19. Slawomir Koziel and Zbigniew Michalewicz. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary Computation*, 7(1):19–44, 1999.
20. Efrén Mezura-Montes and Carlos A. Coello Coello. A Numerical Comparison of some Multiobjective-based Techniques to Handle Constraints in Genetic Algorithms. Technical Report EVOCINV-03-2002, Evolutionary Computation Group at CINVESTAV-IPN, México, D.F. 07300, September 2002. available at: `http://www.cs.cinvestav.mx/~EVOCINV/`.
21. Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.
22. Vilfredo Pareto. *Cours D'Economie Politique*, volume I and II. F. Rouge, Lausanne, 1896.
23. I. C. Parmee and G. Purchase. The development of a directed genetic search technique for heavily constrained design spaces. In I. C. Parmee, editor, *Adaptive Computing in Engineering Design and Control-'94*, pages 97–102, Plymouth, UK, 1994. University of Plymouth, University of Plymouth.
24. S. Rajeev and C.S. Krishnamoorthy. Genetic Algorithms-Based Methodologies for Design Optimization of Trusses. *Journal of Structural Engineering*, 123(3):350–358, 1997.
25. Singiresu S. Rao. *Engineering Optimization. Theory and Practice.* John Wiley & Sons, Inc., third edition,

1996.

26. Tapabrata Ray, Tai Kang, and Seow Kian Chye. An Evolutionary Algorithm for Constrained Optimization. In Darrell Whitley et al., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2000)*, pages 771–777, San Francisco, California, 2000. Morgan Kaufmann.
27. Tapabrata Ray and K.M. Liew. A Swarm Metaphor for Multiobjective Design Optimization. *Engineering Optimization*, 34(2):141–153, March 2002.
28. Jon T. Richardson, Mark R. Palmer, Gunar Liepins, and Mike Hilliard. Some Guidelines for Genetic Algorithms with Penalty Functions. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms (ICGA-89)*, pages 191–197, San Mateo, California, June 1989. George Mason University, Morgan Kaufmann Publishers.
29. T.P. Runarsson and X. Yao. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.
30. J. David Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.
31. Hans-Paul Schwefel. *Evolution and Optimum Seeking*. John Wiley & Sons, New York, 1995.
32. Alice E. Smith and David W. Coit. Constraint Handling Techniques—Penalty Functions. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, chapter C 5.2. Oxford University Press and Institute of Physics Publishing, 1997.
33. W. Stadler. Fundamentals of multicriteria optimization. In W. Stadler, editor, *Multicriteria Optimization in Engineering and the Sciences*, pages 1–25. Plenum Press, New York, NY, 1988.
34. Patrick D. Surry and Nicholas J. Radcliffe. The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms. *Control and Cybernetics*, 26(3):391–412, 1997.
35. S. Tzan and C.P. Pantelides. Annealing strategy for optimal structural design. *Journal of Structural Engineering*, 122(7):815–827, 1996.

APPENDIX A: Test Functions Adopted

1. **g01**:

   Minimize:

$$f(\vec{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

   subject to:

$$g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$$

$$g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$$

$$g_3(\vec{x}) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$$

$$g_4(\vec{x}) = -8x_1 + x_{10} \leq 0$$

$$g_5(\vec{x}) = -8x_2 + x_{11} \leq 0$$

$$g_6(\vec{x}) = -8x_3 + x_{12} \leq 0$$

$$g_7(\vec{x}) = -2x_4 - x_5 + x_{10} \leq 0$$

$$g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \leq 0$$

$$g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \leq 0$$

   where the bounds are $0 \leq x_i \leq 1$ $(i = 1, \ldots, 9)$, $0 \leq x_i \leq 100$ $(i = 10, 11, 12)$ and $0 \leq x_{13} \leq 1$. The global optimum is at $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$ where $f(x^*) = -15$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

2. **g02**:

   Maximize:

$$f(\vec{x}) = \left| \frac{\sum_1^n \cos^4(x_i) - 2\prod_{i=1}^n \cos^2(x_i)}{\sqrt{\sum_1^n i x_i^2}} \right| \tag{11}$$

   subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= 0.75 - \prod_{i=1}^{n} x_i \leq 0 \\
g_2(\vec{x}) &= \sum_{i=1}^{n} x_i - 7.5n \leq 0
\end{aligned} \tag{12}
$$

where $n = 20$ and $0 \leq x_i \leq 10$ $(i = 1, \ldots, n)$. The global maximum is unknown; the best reported solution is [29] $f(x^*) = 0.803619$. Constraint g_1 is close to being active ($g_1 = -10^{-8}$).

3. **g03**:

   Maximize:

$$
f(\vec{x}) = \left(\sqrt{n}\right)^n \prod_{i=1}^{n} x_i \tag{13}
$$

   subject to:

$$
h(\vec{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0 \tag{14}
$$

   where $n = 10$ and $0 \leq x_i \leq 1$ $(i = 1, \ldots, n)$. The global maximum is at $x_i^* = 1/\sqrt{n}$ $(i = 1, \ldots, n)$ where $f(x^*) = 1$.

4. **g04**:

   Minimize:

$$
f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \tag{15}
$$

subject to:

$$\begin{aligned}
g_1(\vec{x}) &= 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0 \\
g_2(\vec{x}) &= -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \leq 0 \\
g_3(\vec{x}) &= 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0 \\
g_4(\vec{x}) &= -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0 \\
g_5(\vec{x}) &= 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0 \\
g_6(\vec{x}) &= -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0
\end{aligned} \tag{16}$$

where: $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$, $27 \leq x_i \leq 45$ $(i = 3, 4, 5)$. The optimum solution is $x^* = (78, 33, 29.995256025682, 45, 36.775812905788)$ where $f(x^*) = -30665.539$. Constraints g_1 and g_6 are active.

5. **g06**

Minimize:

$$f(\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \tag{17}$$

subject to:

$$\begin{aligned}
g_1(\vec{x}) &= -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \\
g_2(\vec{x}) &= (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0
\end{aligned} \tag{18}$$

where $13 \leq x_1 \leq 100$ and $0 \leq x_2 \leq 100$. The optimum solution is $x^* = (14.095, 0.84296)$ where $f(x^*) = -6961.81388$. Both constraints are active.

6. **g07**

Minimize:

$$\begin{aligned} f(\vec{x}) = \quad & x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\ & + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 \\ & + (x_{10} - 7)^2 + 45 \end{aligned} \tag{19}$$

Subject to:

$$\begin{aligned} g_1(\vec{x}) &= -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0 \\ g_2(\vec{x}) &= 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\ g_3(\vec{x}) &= -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\ g_4(\vec{x}) &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0 \\ g_5(\vec{x}) &= 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0 \\ g_6(\vec{x}) &= x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \leq 0 \\ g_7(\vec{x}) &= 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0 \\ g_8(\vec{x}) &= -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0 \end{aligned} \tag{20}$$

where $-10 \leq x_i \leq 10$ $(i = 1, \ldots, 10)$. The global optimum is $x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$ where $f(x^*) = 24.3062091$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

7. **g08**

Maximize:

$$f(\vec{x}) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)} \tag{21}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= x_1^2 - x_2 + 1 \leq 0 \\ g_2(\vec{x}) &= 1 - x_1 + (x_2 - 4)^2 \leq 0 \end{aligned} \tag{22}$$

where $0 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 10$. The optimum solution is located at $x^* = (1.2279713, 4.2453733)$ where $f(x^*) = 0.095825$.

8. **g09**

Minimize:

$$\begin{aligned} f(\vec{x}) = \quad & (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 \\ & +10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 \end{aligned} \tag{23}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0 \\ g_2(\vec{x}) &= -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0 \\ g_3(\vec{x}) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0 \\ g_4(\vec{x}) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned} \tag{24}$$

where $-10 \leq x_i \leq 10$ $(i = 1, \ldots, 7)$. The optimum solution is $x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$ where $f(x^*) = 680.6300573$. Two constraints are active (g_1 and g_4).

9. **g10**

Minimize:

$$f(\vec{x}) = x_1 + x_2 + x_3 \tag{25}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2(\vec{x}) &= -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \\ g_3(\vec{x}) &= -1 + 0.01(x_8 - x_5) \leq 0 \\ g_4(\vec{x}) &= -x_1 x_6 + 833.33252 x_4 + 100 x_1 - 83333.333 \leq 0 \\ g_5(\vec{x}) &= -x_2 x_7 + 1250 x_5 + x_2 x_4 - 1250 x_4 \leq 0 \\ g_6(\vec{x}) &= -x_3 x_8 + 1250000 + x_3 x_5 - 2500 x_5 \leq 0 \end{aligned} \tag{26}$$

where $100 \leq x_1 \leq 10000$, $1000 \leq x_i \leq 10000$, $(i = 2, 3)$, $10 \leq x_i \leq 1000$, $(i = 4, \ldots, 8)$. The global optimum is: $x^* = (579.3167, 1359.943, 5110.071,$ $182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$, where $f(x^*) = 7049.3307$. g_1, g_2 and g_3 are active.

10. **g11**

Minimize:

$$f(\vec{x}) = x_1^2 + (x_2 - 1)^2 \tag{27}$$

subject to:

$$h(\vec{x}) = x_2 - x_1^2 = 0 \tag{28}$$

where: $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$. The optimum solution is $x^* = (\pm 1/\sqrt{2}, 1/2)$ where $f(x^*) = 0.75$.

11. **g13**

Minimize:

$$f(\vec{x}) = e^{x_1 x_2 x_3 x_4 x_5} \tag{29}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0 \\ g_2(\vec{x}) &= x_2 x_3 - 5 x_4 x_5 = 0 \\ g_3(\vec{x}) &= x_1^3 + x_2^3 + 1 = 0 \end{aligned} \tag{30}$$

where $-2.3 \leq x_i \leq 2.3$ $(i = 1, 2)$ and $-3.2 \leq x_i \leq 3.2$ $(i = 3, 4, 5)$. The optimum solution is $x^* = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.763645)$ where $f(x^*) = 0.0539498$.

```
maxsize: maximum size of file
c: current parent ∈ X (decision variable space)
h: child of c ∈ X, a_h: individual in file that dominates h
a_d: individual in file dominated by h
current: current number of individuals in file
cnew: number of individuals generated thus far
g: pick a new parent from less densely populated region every g new individuals
r: shrink space at every r new individuals
current = 1; cnew=0;
c = newindividual();
add(c);
While cnew≤MaxNew do
    h = mutate(c); cnew+ =1;
    if (c≺h) then Label A
    else if (h⪯c) then { remove(c); add(g); c=h; }
    else if (∃ a_h ∈ file | a_h ⪯ h) then Label A
    else if (∃ a_d ∈ file | h ⪯ a_d) then {
        add( h ); ∀ a_d { remove(a_d); current− =1 }
    else test(h,c,file)
Label A
    if (cnew % g==0) then c = individual in less densely populated region
    if (cnew % r==0) then shrinkspace(file)
End While
```

Figure 1. Main algorithm of IS-PAES

```
if (current < maxsize) then {
    add(h);
    if (h ◻ c) then c=h }
else if (∃a_p∈file | h ◻ a_p) then {
    remove(a_p); add(h)
    if (h ◻ c) then c = h; }
```

Figure 2. Pseudo-code of **test(h,c,file)** (called by **main** of IS-PAES)

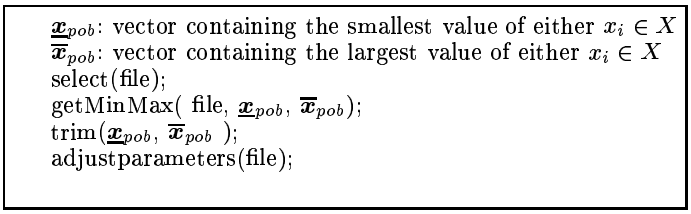

Figure 3. Pseudo-code of **Shrinkspace(file)** (called by **main** of IS-PAES)

```
m: number of constraints
i: constraint index
maxsize: max size of file
listsize: 15% of maxsize
constraintvalue(x,i): value of individual at constraint i
sortfile(file): sort file by objective function
worst(file,i): worst individual in file for constraint i
validconstraints={1,2,3,...,m};
i=firstin(validconstraints);
While (size(file) > listsize and size(validconstraints) > 0) {
    x=worst(file,i)
    if (x violates constraint i)
        file=delete(file,x)
    else validconstraints=removeindex(validconstraints,i)
if (size(validconstraints) > 0) i=nextin(validconstraints)
}
if (size(file) == listsize))
    list=file
else
    file=sort(file)
    list=copy(file,listsize) *pick the best listsize elements*
```

Figure 4. Pseudo-code of **select(file)** (called by **shrinkspace**)

n: size of decision vector;
$\overline{x}_i$: actual upper bound of the i_{th} decision variable
$\underline{x}_i$: actual lower bound of the i_{th} decision variable
$\overline{x}_{pob,i}$: upper bound of i_{th} decision variable in population
$\underline{x}_{pob,i}$: lower bound of i_{th} decision variable in population
$\forall i$: i $\in$ { 1, ..., n }
$slack_i = 0.05 \times (\overline{x}_{pob,i} - \underline{x}_{pob,i})$
$width\_pob_i = \overline{x}_{pob,i} - \underline{x}_{pob,i}$; $width_i^t = \overline{x}_i^t - \underline{x}_i^t$
$deltaMin_i = \frac{\beta * width_i^t - width\_pob_i}{2}$
$\text{delta}_i = \max(\text{slack}_i, \text{deltaMin}_i)$;
$\overline{x}_i^{t+1} = \overline{x}_{pob,i} + delta_i$; $\underline{x}_i^{t+1} = \underline{x}_{pob,i} - delta_i$;
if $(\overline{x}_i^{t+1} > \overline{x}_{original,i})$ **then**
$\underline{x}_i^{t+1} -= \overline{x}_i^{t+1} - \overline{x}_{original,i}$; $\overline{x}_i^{t+1} = \overline{x}_{original,i}$;
if $(\underline{x}_i^{t+1} < x_{original,i})$ **then** $\overline{x}_i^{t+1} += \underline{x}_{original,i} - \underline{x}_i^{t+1}$;
$\underline{x}_i^{t+1} = \underline{x}_{original,i}$;
if $(\overline{x}^{t+1} > \overline{x}_{original,i})$ **then** $\overline{x}_i^{t+1} = \overline{x}_{original,i}$;

Figure 5. Pseudo-code of **trim** (called by **shrinkspace**)

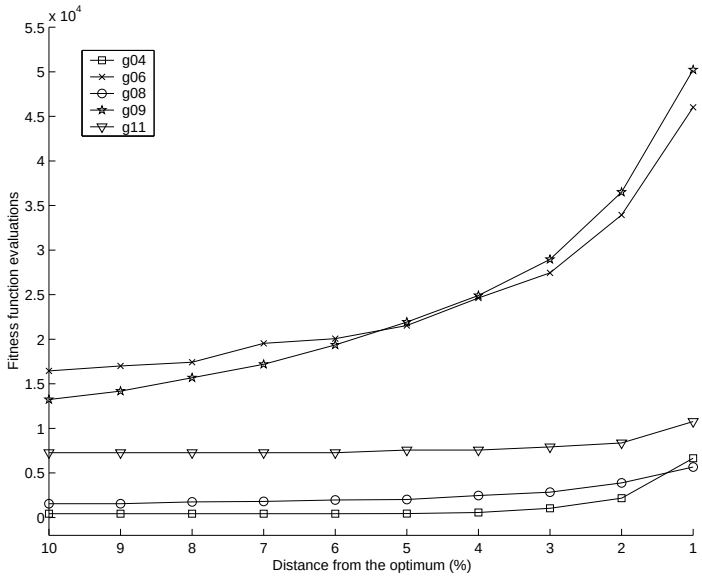

Figure 6. Graphical display of the distance to the optimum (expressed as a percentage) achieved by IS-PAES with respect to the number of fitness function evaluations (values in the y-axis are multiplied by 1×10^4). This figure includes only test functions **g04**, **g06**, **g08**, **g09**, and **g11**.

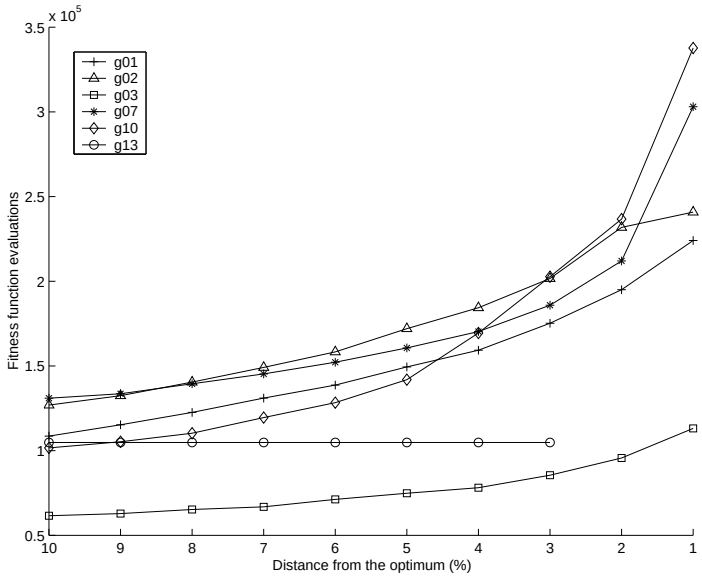

Figure 7. Graphical display of the distance to the optimum (expressed as a percentage) achieved by IS-PAES with respect to the number of fitness function evaluations (values in the y-axis are multiplied by 1×10^5). This figure includes only test functions **g01**, **g02**, **g03**, **g07**, **g10** and **g13**.

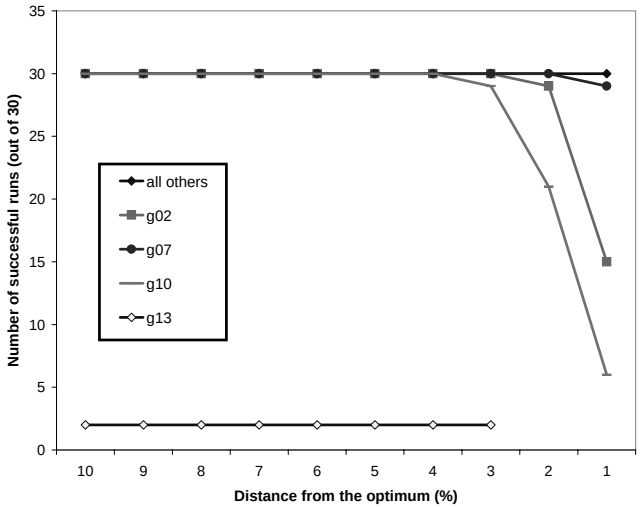

Figure 8. Graphical representation of the distance from the optimum (expressed as a percentage) achieved by IS-PAES with respect to the successful runs performed (over the total of 30 runs performed on each test function).

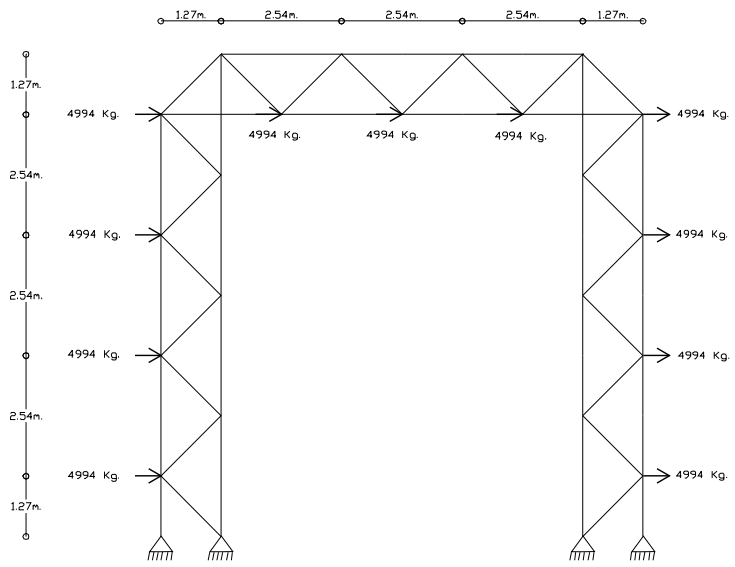

Figure 9. 49-bar plane truss used as the first engineering optimization example.

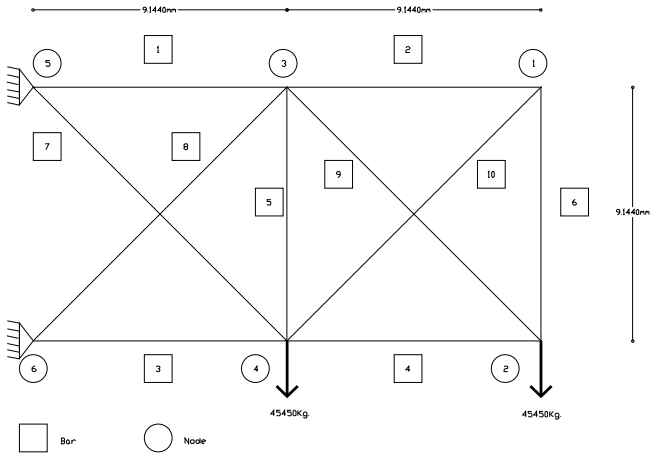

Figure 10. 10-bar plane truss used as the second engineering optimization example.

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

Table I. Values of ρ for the eleven test problems chosen. n is the number of decision variables, LI is the number of linear inequalities, NI the number of nonlinear inequalities, NE is the number of nonlinear equalities, and LE is the number of linear equalities.

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-4.806906	-1.203723	0.000000	0.000000	1.638475
g02	-0.803619	0.021716	0.016409	0.017607	0.007805	0.003410
g03	-1.0	-0.022967	-0.001121	-0.000042	-0.000000	-0.004195
g04	-30665.539	-30483.474609	-30397.133659	-30389.091797	-30320.294922	37.640510
g06	-6961.814	-6622.280273	-6058.865120	-6157.552979	-4859.331055	436.786555
g07	24.306	468.216675	1173.033939	1690.598938	1933.539917	610.679874
g08	-0.095825	-0.095813	-0.095447	-0.095652	-0.093345	0.000499
g09	680.630	723.854919	873.917936	877.980988	995.981873	68.385708
g10	7049.331	11129.170898	15875.698828	15952.260254	20528.048828	2371.513317
g11	0.750	0.749014	0.749308	0.749297	0.749880	0.000186
g13	0.053950	N.F.	N.F.	N.F.	N.F.	N.F.

Table II. Results produced by COMOGA. N.F. = Not Feasible (i.e., the method was unable to reach the feasible region).

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-11.136517	-10.249717	-10.148078	-9.534448	0.349041
g02	-0.803619	0.000212	0.000077	0.000048	0.000008	0.000057
g03	-1.0	N.F.	N.F.	N.F.	N.F.	N.F.
g04	-30665.539	-30652.330078	-30638.775977	-30637.970703	-30625.042969	5.337246
g06	-6961.814	-6941.932129	-6873.139681	-6871.179932	-6743.495117	46.609326
g07	24.306	28.631790	32.014552	32.026387	35.525009	1.736919
g08	-0.095825	-0.095826	-0.095826	-0.095826	-0.095826	0.000000
g09	680.630	693.252319	717.023442	717.820679	744.084412	11.384411
g10	7049.331	11259.611328*	14046.409651*	13283.869629*	22271.488281*	2773.263053*
g11	0.750	0.749426	0.760147	0.752852	0.811940	0.018649
g13	0.053950	N.F.	N.F.	N.F.	N.F.	N.F.

Table III. Results produced by VEGA. N.F. = Not Feasible (i.e., the method was unable to reach the feasible region). * One of the runs of VEGA converged to a non-feasible solution. Therefore, these statistics were computed using only the results of 29 runs.

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-11.007717	-8.033259	-7.654563	-4.719421	1.718908
g02	-0.803619	0.790404	0.769520	0.772691	0.739923	0.012923
g03	-1.0	-0.981203	-0.928032	-0.927160	-0.884223	-0.021715
g04	-30665.539	-30659.656250	-30653.059961	-30654.006836	-30639.603516	5.022099
g06	-6961.814	-6956.971680	-6776.418766	-6818.011475	-6310.125488	176.181100
g07	24.306	26.232813	28.296385	28.167288	30.784266	1.106179
g08	-0.095825	0.095826	0.095826	0.095826	0.095826	0.000000
g09	680.630	680.872986	681.473741	681.515137	682.188416	0.326890
g10	7049.331	8812.435547	11134.727006	9896.345215	15609.163086	2381.940642
g11	0.750	0.749007	0.749099	0.749049	0.749713	0.000142
g13	0.053950	N.F.	N.F.	N.F.	N.F.	N.F.

Table IV. Results produced by NPGA. N.F. = Not Feasible (i.e., the method was unable to reach the feasible region)

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-14.504487	-13.981660	-13.997746	-13.306435	0.320086
g02	-0.803619	0.680874	0.58471	0.569982	0.499295	0.048400
g03	-1.0	-0.868598	-0.561975	-0.578549	-0.263762	0.157852
g04	-30665.539	-30659.845703	-30615.247591	-30614.80273	-30552.658203	28.262409
g06	-6961.814	-6957.950684	-6903.774691	-6906.598389	-6845.432129	29.742030
g07	24.306	27.512201	36.427887	33.668530	80.891251	10.502327
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0.000000
g09	680.630	681.324036	686.101556	684.921814	700.550598	4.505219
g10	7049.331	7372.459961	8566.307992	8316.973145	12552.230469	1159.513051
g11	0.750	0.749002	0.749063	0.749041	0.749312	0.000072
g13	0.053950	N.F.	N.F.	N.F.	N.F.	N.F.

Table V. Results produced by MOGA. N.F. = Not Feasible (i.e., the method was unable to reach the feasible region)

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-14.9997	-14.494	-14.997	-12.446	9.3×10^{-1}
g02	-0.803619	-0.803376	-0.793281	-0.793342	-0.768291	9.0×10^{-3}
g03	-1.0	-1.000	-1.000	-1.000	-1.000	9.7×10^{-5}
g04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	0.0
g06	-6961.814	-6961.814	-6961.813	-6961.814	-6961.810	8.5×10^{-5}
g07	24.306	24.338	24.527	24.467	24.995	1.7×10^{-1}
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0.0
g09	680.630	680.630	680.631	680.631	680.634	8.1×10^{-4}
g10	7049.331	7062.019	7342.944	7448.014	7588.054	1.4×10^{2}
g11	0.750	0.750	0.750	0.750	0.751	2.6×10^{-4}
g13	0.053950	0.05517	0.28184	0.2779	0.5471	1.776×10^{-1}

Table VI. Results produced by our IS-PAES algorithm.

TF	optimal	Best	Mean	Median	Worst	Std Dev
g01	-15.0	-15.0	-15.0	-15.0	-15.0	0.0
g02	-0.803619	-0.803515	-0.781975	-0.785800	-0.726288	2×10^{-2}
g03	-1.0	-1.000	-1.000	-1.000	-1.000	1.9×10^{-4}
g04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	2.0×10^{-5}
g06	-6961.814	-6961.814	-6875.940	-6961.814	-6350.262	1.6×10^{2}
g07	24.306	24.307	24.374	24.357	24.642	6.6×10^{-2}
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	2.6×10^{-17}
g09	680.630	680.630	680.656	680.641	680.763	3.4×10^{-2}
g10	7049.331	7054.316	7559.192	7372.613	8835.655	5.3×10^{2}
g11	0.750	0.750	0.750	0.750	0.750	8.0×10^{-5}
g13	0.053950	0.053957	0.0675543	0.057006	0.216915	3.1×10^{-2}

Table VII. Results produced by the stochastic ranking algorithm [29] using $Pf = 0.45$.

Algorithm	Average Weight (Kg)
IS-PAES	610
SA	627
GA50	649
GSSA50	619
GSSA5	625

Table VIII. Comparison of different algorithms on the 49-bar struss, case 1.

Algorithm	Average Weight (Kg)
IS-PAES	725
SA	737
GA50	817
GSSA50	748
GSSA5	769

Table IX. Comparison of different algorithms on the 49-bar struss, case 2.

Algorithm	Average Weight (Kg)
IS-PAES	2603
SA	2724
GA50	2784
GSSA50	2570
GSSA5	2716

Table X. Comparison of different algorithms on the 49-bar struss, case 3.

Element	IS-PAES	GSSA	VGA	MC	SSO	ISA	SAARSR
1	190.53	205.17	206.46	200.01	193.75	269.48	201.35
2	0.6466	0.6452	0.6452	0.6452	0.6452	79.810	0.6452
3	146.33	134.20	151.62	129.04	150.15	178.45	161.55
4	95.07	90.973	103.23	90.328	98.62	152.90	95.68
5	0.6452	0.6452	0.6452	0.6452	0.6452	70.390	0.6452
6	3.0166	0.6452	0.6452	0.6452	3.23	10.260	4.19
7	47.677	55.487	54.84	51.616	48.18	147.87	49.16
8	129.826	127.75	129.04	145.17	136.64	14.710	131.55
9	133.282	133.56	132.27	96.78	139.47	156.06	134.32
10	0.6452	0.6452	0.6452	0.6452	0.6452	87.740	0.6452
Vol. (cm^3)	801624.5	805777	833258	765710	828956	1313131	833258
Weight (kg)	5951	6186	6186	5685	6155	9750	6187

Table XI. Comparison of weights for the 10-bar plane truss of the second engineering example.

Element	IS-PAES	GSSA	VGA	MC	SSO	ISA	SAARSR
1	483.27	-447.65	-444.75	-460.10	-475.31	-209.75	-476.58
2	-73.37	0.41	3.41	-15.30	91.98	-111.35	43.99
3	-613.26	670.31	593.43	695.72	597.46	449.90	569.04
4	-478.62	499.60	440.30	503.06	461.46	239.13	485.80
5	1741.30	-1464.09	-1428.68	**-1757.16**	-1754.88	362.13	-1641.04
6	-15.72	0.41	3.41	-15.30	18.37	-866.13	14.83
7	1313.54	-1134.31	-1148.24	-1214.48	-1299.10	-763.45	-1311.60
8	-507.89	513.60	508.24	453.71	482.74	1064.60	528.83
9	482.80	-481.25	-485.97	-664.00	-461.46	-331.34	-492.79
10	103.985	0.58	-4.82	21.64	-130.07	143.23	-65.61

Table XII. Comparison of stresses for the 10-bar plane truss of the second engineering example. We indicate in **boldface** the elements in which the stress constraints are being violated.

Element	IS-PAES	GSSA	VGA	MC	SSO	ISA	SAARSR
1	0.5134	0.5602	0.5528	0.5954	0.4802	0.4022	0.5419
2	-5.080	-5.0798	-4.9040	**-5.4352**	-4.9056	-3.8008	**5.0889**
3	-1.368	-1.4654	-1.2948	-1.5016	-1.3264	-0.8631	-1.3213
4	-5.060	-5.0792	-4.8997	**5.4543**	-4.8826	-4.8857	-5.0746
5	0.6053	0.5607	0.5571	0.5763	0.5954	0.2627	0.5970
6	-1.878	-1.8474	-1.8303	-1.7130	-1.8047	-2.9298	-1.9303
7	-0.768	-0.8396	-0.7433	-0.8715	-0.7484	-0.5636	-0.7129
8	-4.059	-3.6813	-3.6199	-3.9140	-4.0030	-2.4762	-3.9901

Table XIII. Comparison of displacements for the 10-bar plane truss of the second engineering example. We indicate in **boldface** the elements in which the displacement constraints are being violated.