Use of Multiobjective Optimization Concepts to Handle Constraints in Genetic Algorithms

Efrén Mezura-Montes and Carlos A. Coello Coello

CINVESTAV-IPN
Evolutionary Computation Group
Departamento de Ingeniería Eléctrica
Sección de Computación
Av. Instituto Politécnico Nacional No. 2508
Col. San Pedro Zacatenco
México D.F. 07300, MÉXICO
emezura@computacion.cs.cinvestav.mx
ccoello@cs.cinvestav.mx

Summary. This chapter describes the general multiobjective optimization concepts that can and have been used to incorporate constraints of any type (linear, nonlinear, equality and inequality) into the fitness function of a genetic algorithm used for global optimization. Several approaches reported in the literature are also described and four of them are compared using several test functions. The results obtained are discussed and further ideas about how to devise new approaches are also briefly analyzed.

1 Introduction

Evolutionary Algorithms (EAs) are heuristics that have been successfully applied in a wide set of areas [14, 30, 1, 4, 19, 6, 3, 32, 44, 43, 28], both in single- and in multiobjective optimization. However, EAs lack a mechanism able to bias efficiently the search towards the feasible region in constrained search spaces. This has triggered a considerable amount of research and a wide variety of approaches have been suggested in the last few years to incorporate constraints into the fitness function of an evolutionary algorithm [8, 31].

The most common approach adopted to deal with constrained search spaces is the use of penalty functions. When using a penalty function, 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 can bias the search in an appropriate way [40, 8].

Among the several approaches that have been proposed as an alternative to the use of penalty functiones, there is a group of techniques in which the constraints of a problem are handled as objective functions (i.e., a single-objective constrained problem is restated as an unconstrained multiobjective problem). This chapter precisely focuses on these techniques.

This chapter is organized as follows. Section 2 presents the basic concepts both from global optimization and from multiobjective optimization that are going to be used in the remainder of this chapter. In Section 3, the most popular multiobjective-based constraint-handling techniques are discussed. Section 4 presents a small comparative study in which four of the techniques discussed in the previous section are tested on four benchmark problems taken from the standard constraint-handling literature [31]. Section 4 discusses the results obtained, and Section 6 concludes and presents some possible paths of future research in this area.

2 Basic Concepts

We are interested in the general nonlinear programming problem in which we want to:

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

subject to:

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

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

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

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

Now, we will define some basic concepts from multiobjective optimization.

Definition 1 (General Multiobjective Optimization Problem): *Find the vector* $\mathbf{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ *which will satisfy the* m *inequality constraints:*

$$g_i(\mathbf{x}) \geq 0 \quad i = 1, 2, \ldots, m \tag{4}$$

the p *equality constraints*

$$h_i(\mathbf{x}) = 0 \quad i = 1, 2, \ldots, p \tag{5}$$

and will optimize the vector function

$$\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})]^T \tag{6}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ *is the vector of decision variables.* □

Having several objective functions, the notion of "optimum" changes, because in multiobjective optimization problems, the aim is to find good compromises (or "trade-offs") rather than a single solution as in global optimization. The notion of "optimum" that is most commonly adopted is that originally proposed by Francis Ysidro Edgeworth in 1881 [17] and later generalized by Vilfredo Pareto (in 1896) [33]. This notion is normally referred to as "Pareto optimality" and is defined next.

Definition 2 (Pareto Optimality:): *A point* $\mathbf{x}^* \in \mathcal{F}$ *is* **Pareto optimal** *if for every* $\mathbf{x} \in \mathcal{F}$ *and* $I = \{1, 2, \ldots, k\}$ *either,*

$$\forall_{i \in I}(f_i(\mathbf{x}) = f_i(\mathbf{x}^*)) \tag{7}$$

or, there is at least one $i \in I$ *such that*

$$f_i(\mathbf{x}) > f_i(\mathbf{x}^*) \tag{8}$$

□

In words, this definition says that $\mathbf{x}^*$ is Pareto optimal if there exists no feasible vector $\mathbf{x}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. The phrase "Pareto optimal" is considered to mean with respect to the entire decision variable space unless otherwise specified.

Other important definitions associated with Pareto optimality are the following:

Definition 3 (Pareto Dominance): *A vector* $\mathbf{u} = (u_1, \ldots, u_k)$ *is said to dominate* $\mathbf{v} = (v_1, \ldots, v_k)$ *(denoted by* $\mathbf{u} \preceq \mathbf{v}$*) if and only if* u *is partially less than* v*, i.e.,* $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$. □

Definition 4 (Pareto Optimal Set): *For a given multiobjective optimization problem,* $\mathbf{f}(x)$*, the Pareto optimal set (*$\mathcal{P}^*$*) is defined as:*

$$\mathcal{P}^* := \{x \in \mathcal{F} \mid \neg\exists\, x' \in \mathcal{F}\ \ \mathbf{f}(x') \preceq \mathbf{f}(x)\}. \tag{9}$$

□

3 Multiobjective-based Constraint Handling techniques

The main idea behind using multiobjective techniques to handle constraints is to redefine the single-objective optimization of $f(\mathbf{x})$ 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 [14] to the new vector $\bar{v} = (f(\mathbf{x}), f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$, where $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ are the original constraints of the problem. An ideal solution $\mathbf{x}$ would thus have $f_i(\mathbf{x})$=0 for $1 \leq i \leq m$ and $f(\mathbf{x}) \leq f(\mathbf{y})$ for all feasible $\mathbf{y}$ (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.
2. Use of Pareto ranking [20] to assign fitness in such a way that nondominated individuals (i.e., feasible individuals in this case) are assigned a higher fitness value.
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) [39].

We will now proceed to discuss the different approaches that have been proposed adopting the three main ideas previously indicated.

3.1 COMOGA

Surry & Radcliffe [41] used a combination of the Vector Evaluated Genetic Algorithm (VEGA) [39] and Pareto Ranking to handle constraints in an approach called COMOGA (Constrained Optimization by Multi-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 beginning of the process and it is adapted during the run using as a basis the percentage of feasible individuals that one wishes to have in the population.

COMOGA was applied on a gas network design problem and it was compared against a penalty function approach. Although COMOGA showed a slight improvement in the results with respect to a penalty function, its main

advantage is that it does not requiere a fine tuning of penalty factors or any other additional parameter. The main drawback of COMOGA is that it requires several extra parameters, although its authors argue that the technique is not particularly sensitive to their values [41].

The algorithm of COMOGA is the following [41]:

Begin
1. Calculate constraint violation for all solutions.
2. Rank solutions based on constraint violation (nondominance checking).
3. Evaluate the fitness of solutions.
4. Select a P_{cost} proportion of parents based on fitness and the remaining $1 - P_{cost}$ based on constraint ranking.
6. Apply genetic operators
7. Adjust P_{cost}: Decreasing it favors feasible solutions; Increasing it favors lower cost solutions (high fitness)

End

3.2 VEGA

Parmee & Purchase [34] proposed to use VEGA [39] to guide the search of an evolutionary algorithm to the feasible region of an optimal gas turbine design problem with a heavily constrained search space. After having a feasible point, they generated an optimal hypercube around it in order to avoid leaving the feasible region after applying the genetic operators. Note that this approach does not really use Pareto dominance or any other multiobjective optimization concepts to exploit the search space. Instead, it uses VEGA just to reach the feasible region. The use of special operators that preserve feasibility make this approach highly specific to one application domain rather than providing a general methodology to handle constraints.

Coello [12] used a population-based approach similar to VEGA [39] 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.

The algorithm of this approach is the following:

Begin
Create M random solutions for the initial population.

Split the population into $m + 1$ sub-populations
Evaluate all M individuals
Assign a fitness value to all M individuals depending of their corresponding subpopulation.
While stopping criterion is not satisfied **Do**
  Insert the best individual of the current population into the next population
  While the next population is not full **Do**
    Select 2 parents p_1 and p_2 based on **tournament selection with n candidates** from all the M individuals of the main population
    Apply crossover to p_1 and p_2 to generate 2 offspring c_1 and c_2
    Apply mutation to offspring c_1 and c_2
    Insert c_1 and c_2 into the next population
  End While
  Split the population into $m + 1$ subpopulations
  Evaluate all M new individuals
  Assign a fitness value to all M individuals depending of their corresponding subpopulation.
End While
End

The fitness assignment scheme of the approach is the following:

if $g_j(\mathbf{x}) < 0.0$	**then**	fitness $= g_j(\mathbf{x})$
else if $v \neq 0$	**then**	fitness $= -v$
else		fitness $= f(\mathbf{x})$

where $g_j(\mathbf{x})$ refers to the jth constraint of the problem, v is the number of violated constraints ($v \leq m$) and $f(\mathbf{x})$ is the value of the objective function of the individual.

As can be seen above, each subpopulation tries to satisfy one single constraint. If the encoded solution does not violate the constraint of its corresponding subpopulation, then the fitness of an individual will be determined by the total number of constraints violated. Finally, if the solution is feasible, then the feasible criterion is to optimize the objective function. Therefore, any feasible individuals will be merged with the subpopulation on charge of optimizing the original (unconstrained) objective function.

The genetic operators are applied to the entire population and it is allowed to every individual in a subpopulation to mate with any other in any subpopulation (including its own, of course). In this way, individuals who satisfy constraints are combined with individuals with a good fitness value. At the end, it is expected to have a population of feasible individuals with high fitness values.

This approach was tested with some engineering problems [12] in which it produced competitive results. It has also been successfully used to solve

combinational circuit design problems [13]. The main drawback of this approach is that the number of subpopulations required increases linearly with the number of constraints of the problem. This has some obvious scalability problems. Furthermore, it is not clear how to determine appropriate sizes for each of the subpopulations used.

3.3 MOGA

Coello [11] proposed the use of Pareto dominance selection to handle constraints in EAs. This is an application of Fonseca and Fleming's Pareto ranking process [18] (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.

Coello's approach uses a real-coded GA with universal stochastic samplig selection (to reduce the selection pressure caused by the Pareto ranking process).

The algorithm of this approach is the following:

Begin
  Create M random solutions for the initial population.
  Evaluate the M individuals in the population.
  Calculate the rank for each of the M individuals in the population.
  Assign a fitness value to all M individuals depending on rank
  While stopping criterion is not satisfied **Do**
    Insert the best individual of the current
    population into the next population
    While the next population is not full **Do**
      Select 2 parents p_1 and p_2 using **Universal Stochastic Sampling**
      Apply crossover to p_1 and p_2 to generate 2 offspring c_1 and c_2
      Apply mutation to offspring c_1 and c_2
      Insert c_1 and c_2 into the next population
    End While
    Evaluate the M new individuals in the population
    Calculate the rank for each one of the M individuals in the population.
    Assign a fitness value to all M individuals depending on rank
  End While
End

To compute the **rank** of an individual $\mathbf{x}_i$ this approach uses the following procedure:

- Evaluate:

$$\text{rank}(\mathbf{x}_i) = \text{count}(\mathbf{x}_i) + 1 \tag{10}$$

  where $\text{count}(\mathbf{x}_i)$ is computed according to the following rules:

1. Compare $\mathbf{x}_i$ against every other individual in the population. Assuming pairwise comparisons, we will call $\mathbf{x}_j$ ($j = 1, \ldots, pop\_size$ and $j \neq i$) the other individual against which x_i is being compared at any given time.
2. Initialize count($\mathbf{x}_i$)(for $i = 1, \ldots, pop\_size$) to zero.
3. If both $\mathbf{x}_i$ and $\mathbf{x}_j$ are feasible, then both are given a rank of zero and count($\mathbf{x}_i$) remains without changes.
4. If $\mathbf{x}_i$ is infeasible and $\mathbf{x}_j$ is feasible, then count($\mathbf{x}_i$) is incremented by one.
5. If both $\mathbf{x}_i$ and $\mathbf{x}_j$ are infeasible, but $\mathbf{x}_i$ violates more constraints than $\mathbf{x}_j$, then count($\mathbf{x}_i$) is incremented by one.
6. If both $\mathbf{x}_i$ and $\mathbf{x}_j$ are infeasible, and both violate the same number of constraints, but $\mathbf{x}_i$ has a total amount of constraint violation larger than the constraint violation of $\mathbf{x}_j$, then count($\mathbf{x}_i$) is incremented by one.
   If any constraint $g_k(\mathbf{x})$ ($k = 1, \ldots, m$, where m is the total amount of constraints) is considered satisfied if $g_i(\mathbf{x}) \leq 0$, then, the total amount of constraint violation for an individual $\mathbf{x}_i$ (denoted as coef($\mathbf{x}_i$)) is given by:

$$\text{coef}(\mathbf{x}_i) = \sum_{k=1}^{p} g_k(\mathbf{x}_i) \quad \text{for all } g_k(\mathbf{x}_i) > 0 \tag{11}$$

To compute **fitness**, the following rules are adopted:

1. If $\mathbf{x}_i$ is feasible, then rank($\mathbf{x}_i$) = $fitness(\mathbf{x}_i)$, else
2. rank($\mathbf{x}_i$) = $\frac{1}{\text{rank}(\mathbf{x}_i)}$

Then, individuals are selected based on rank($\mathbf{x}_i$) (stochastic universal sampling is used). Note that the values produced by $fitness(\mathbf{x}_i)$ must be normalized to ensure that the rank of feasible individuals is always higher than the rank of infeasible ones.

This approach has been used to solve some engineering design problems [11] in which it produced very good results. Furthermore, the approach showed great robustness and a relatively low number of fitness function evaluations with respect to traditional penalty functions. Additionally, it does not require any extra parameters. Its main drawback is the computational cost ($O(M^2)$, where M is the population size) derived from the Pareto ranking process.

3.4 NPGA

Coello and Mezura [9] implemented a version of the Niched-Pareto Genetic Algorithm (NPGA) [23] 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 fitness sharing [16]) 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. This is the main advantage of this approach with respect to Coello's proposal [11].

Note however that Coello and Mezura's approach requires an additional parameter called S_r that controls the diversity of the population. S_r indicates the proportion of parents selected by four comparison criteria described below. The remaining $1 - S_r$ parents will be selected by a pure probabilistic approach. Thus, this mechanism is responsible for keeping infeasible individuals in the population (i.e., the source of diversity that keeps the algorithm from converging to a local optimum too early in the evolutionary process).

A graphical illustration of the role of the parameter S_r is shown in Figure 1.

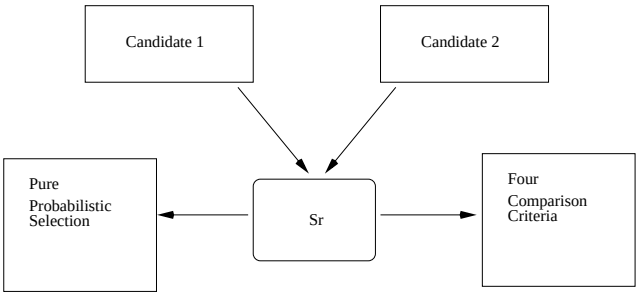

Fig. 1. Diagram that illustrates the role of S_r in the selection process of Coello and Mezura's algorithm.

Tournaments in this approach are decided using as a basis four comparison criteria:

If

1. both individuals are feasible, the individual with the higher fitness wins.
2. one is feasible and the other is infeasible, the feasible individual wins.
3. both are infeasible: Nondominance checking is applied (tournament selection as in the NPGA [23]).
4. both are nondominated or dominated, the individual with the lowest amount of constraint violation wins.

The algorithm of this approach is the following:

Begin

Create M random solutions for the initial population.

Evaluate the M individuals in the population.

While stopping criterion is not satisfied **Do**
Insert the best individual of the current
population into the next population
While the next population is not full **Do**
Select 2 parents p_1 and p_2 based on $\mathbf{S_r}$ value
Apply crossover to p_1 and p_2 to generate 2 offspring c_1 and c_2
Apply mutation to offspring c_1 and c_2
Insert c_1 and c_2 into the next population
End While
Evaluate the M new individuals in the population
End While
End

This approach has been tested with several benchmark problems and was compared against several types of penalty functions [29]. Results indicated that the approach was robust, efficient and effective. However, it was also found that the approach had scalability problems (its performance degrades as the number of decision varibles increases).

3.5 Pareto Set and Line Search

Camponogara & Talukdar [5] proposed an approach in which a global optimization problem was transformed into a bi-objective problem where the first objective is to optimize the original objective function and the second is to minimize:

$$\Phi(\mathbf{x}) = \sum_{i=1}^{n} \max(0, g_i(\mathbf{x})) \tag{12}$$

Equation 12 tries to minimize the total amount of constraint violation of a solution (i.e., it tries to make it feasible). At each generation of the process, several Pareto sets are generated. An operator that substitutes crossover takes two Pareto sets S_i and S_j where $i < j$ and two solutions $x_i \in S_i$ and $x_j \in S_j$ where x_i dominates x_j. With these two points a search direction is defined using:

$$d = \frac{(x_i - x_j)}{|x_i - x_j|} \tag{13}$$

Line search begins by proyecting d over one variable axis on decision variable space in order to find a new solution x which dominates both x_i and x_j. At pre-defined intervals, the worst half of the population is replaced with new random soutions to avoid premature convergence. This indicates some of the problems of the approach to maintain diversity. Additionally, the use of line search within a GA adds some extra computational cost.

The algorithm is the following [5]:

Begin
Let S be a random initial population.
Let $F = \{f_0(), \ldots, f_k()\}$ be a set of objectives to be minimized.
While stopping criterion is not satisfied **Do**
Let $L = \{S_1, \ldots, S_t\}$ be the Pareto list for S with respect to F
If $t = 1$ **Then**
Replace half of the points in S by random points
Rebuild the Pareto List L
End If
Let $N = \emptyset$ be the set of new points to be generated
While $|N| < m$ **Do**
Choose two Pareto sets S_i and S_j with $i < j$
Choose two points sets $x_i \in S_i$ and $x_j \in S_j$ with $i < j$
Let $d = \frac{(x_i - x_j)}{|x_i - x_j|}$ be the search direction
With probability α project d onto the axis of one variable j in the solution space
Execute line search through the line defined by the point x_i and by the direction d
Let $U = \{u_o, \ldots, u_k\}$ be the set of points obtained in the line search such that u_j is the best point with evaluation $f_j()$
Let $N = N \bigcup U$
End While
Let $S = S \bigcup N$
Let $L = \{S_1, \ldots, S_l\}$ be the Pareto list for S
Remove $|N|$ points from the last Pareto sets in List L
End While
End

The authors of this approach validated it using a benchmark consisting of five test functions. The results obtained were either optimal or very close to it. The main drawback of this approach is its additional computational cost. Also, it is not clear what is the impact of the segment chosen to search in the overall performance of the algorithm.

3.6 Min-Max

An approach similar to a min-max formulation used in multiobjective optimization [7] combined with tournament selection was proposed by Jiménez and Verdegay [25].

The algorithm is the following:

Begin
Create M random solutions for the initial population.
Evaluate the M individuals in the population.

While stopping criterion is not satisfied **Do**
Insert the best individual of the current
population into the next population
While the next population is not full **Do**
Select 2 parents p_1 and p_2 based on **tournament selection** and
based on the criteria shown below
Apply crossover to p_1 and p_2 to generate 2 offspring c_1 and c_2
Apply mutation to offspring c_1 and c_2
Insert c_1 and c_2 into the next population
End While
Evaluate the M new individuals in the population
End While
End

The selection criteria is based on the following rules:

- Between two feasible individuals, that one with a higher fitness wins.
- A feasible individual wins over a infeasible individual.
- Between two infeasible individuals, that one with the lowest amount of constraint violation wins.

This approach was validated using four test functions, and the results obtained in most cases were very close to the optima. A subtle problem with this approach is that the evolutionary process first concentrates only on the constraint satisfaction problem and therefore it samples points in the feasible region essentially at random [42]. This means that in some cases (e.g., when the feasible region is disjoint) we might land in an inappropriate part of the feasible region from which we will not be able to escape. However, this approach (as in the case of Parmee and Purchase's [34] technique) may be a good alternative to find a feasible point in a heavily constrained search space. The relative simplicity of this approach is another advantage of this technique.

3.7 Pareto Ranking and Domain Knowledge

Ray et al. [36] proposed the use of a Pareto ranking approach that operates on three spaces: objective space, constraint space and the combination of the two previous spaces. This approach also uses mating restrictions to ensure better constraint satisfaction in the offspring generated and a selection process that eliminates weaknesses in any of these spaces. To maintain diversity, a niche mechanism based on Euclidean distances is used. This approach can solve both constrained or unconstrained optimization problems with one or several objective functions.

The algorithm is the following [36]:

Begin
Let M be a random initial population.

Do
Compute Pareto Ranking based on objective matrix to yield
a vector **RankObj**
Compute Pareto Ranking based on constraint matrix to yield
a vector **RankCon**
Compute Pareto Ranking based on the combined matrix to yield
a vector **RankCom**
If problem is multiobjective optimization **Then**
Select individuals from the population in this generation if
RankCom=1 and Feasible and put them into the population for
the next generation
End If
If problem is single objective optimization **Then**
Select individuals from the population in this generation if
RankCom is better than allowable rank and Feasible
and put them into the population for the next generation
End If
Do
Select an individual A and its partner from the population at
this generation
Mate A with its partner
Put parents and children into the population of the next generation
While the population is not full
Remove duplicate points in parametric space and shrink population
While the maximum number of generations is not attained
End

From the previous pseudo-code, it can be seen that three different matrices must be computed: the objective matrix (containing the objective function values of each solution), the constraint matrix (that contains the constraint values of each individual), and a matrix that combines the two previous. These matrices are illustrated in equations 14, 15 and 16.

$$\begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1k} \\ f_{21} & f_{22} & \cdots & f_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ f_{M1} & f_{M2} & \cdots & f_{Mk} \end{pmatrix} \tag{14}$$

$$\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1s} \\ c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & \cdots & c_{Ms} \end{pmatrix} \tag{15}$$

$$\begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1k} & c_{11} & c_{12} & \cdots & c_{1s} \\ f_{21} & f_{22} & \cdots & f_{2k} & c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{M1} & f_{M2} & \cdots & f_{Mk} & c_{M1} & c_{M2} & \cdots & c_{Ms} \end{pmatrix} \tag{16}$$

The mating restrictions used by this method are based on the information that each individual has about its own feasibility. Such a scheme is based on an idea proposed by Hinterding and Michalewicz [22].

The main advantage of this approach is that it requires a very low number of fitness function evaluations (between 2% and 10% of the number of evaluations required by the homomorphous maps of Koziel and Michalewicz [27], which is one of the best constraint-handling techniques known to date). The technique has some problems to reach the global optima, but it produces very good approximations considering its low computation cost. The main drawback of the approach is that its implementation is considerably more complex than any of the other techniques previously discussed.

3.8 Pareto Dominance and Preselection

Jiménez et al. [24] proposed an algorithm that uses Pareto dominance inside a preselection scheme to solve several types of optimization problems (multiobjective, constraint satisfaction, global optimization, and goal programming problems). The approach redefines the problem as an unconstrained multiobjective optimization problem in which objectives are given priorities. Feasible solutions with a good objective function value are given the highest priority. The authors use a real-coded nongenerational GA with two types of crossover operators (uniform and arithmetic) and two mutation operators (uniform and nonuniform).

The preselection scheme of this approach works as follows [24]:

Begin
At each iteration of the algorithm
select two parents p_1 and p_2 at random.
Apply the crossover operators NC times
to produce NC offspring
Apply the mutation operators to produce
a total of $2 \times NC$ offspring
Obtain the best c_1 individuals of the first NC
offspring based on Pareto dominance
Obtain the best c_2 individuals of the second NC
(mutated) offspring based on Pareto dominance
If c_1 dominates p_1 **Then**
c_1 replaces p_1 in the population
EndIf
If c_2 dominates p_2 **Then**

c_2 replaces p_2 in the population
EndIf
End

The authors argue that this preselection mechanism is an implicit niche formation technique because individuals are replaced only by similar ones (i.e., their offspring). As only the best individuals are inserted into the new population, this scheme is also an elitist strategy.

This approach was validated with eleven test functions, producing very good results. Note however, that the authors do not specify the computational cost of the approach and it is not clear if the approach is competitive with other techniques in that regard.

3.9 Pareto Ranking and Robust Optimization

Ray [35] explored an extension of his previous work on constraint-handling [36] in which the emphasis was robustness. A robust optimized solution is not sensitive to parametric variations due to incomplete information of the problem or to changes on it. This approach is capable of handling constraints and finds feasible solutions that are robust to parametric variations produced over time. This is achieved using the individual's self-feasibility and its neighborhood feasibility. Thus, a new matrix called "modified constraint matrix" is used to replace both the constraint matrix and the combined matrix of Ray's original proposal (see equations 15 and 16). An example of this modified constraint matrix is the following:

$$\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1s} & c_{1s+1} & c_{1s+2} & \cdots & c_{12s} \\ c_{21} & c_{22} & \cdots & c_{2s} & c_{2s+2} & c_{2s+2} & \cdots & c_{22s} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & \cdots & c_{Ms} & c_{Ms+1} & c_{Ms+2} & \cdots & c_{M2s} \end{pmatrix} \tag{17}$$

where $c_{is+1}, c_{is+2} \ldots c_{i2s}$ denotes the number of violations of the first constraint among k neighbors and so on. A mechanism based on ranking values in both spaces (objective space and constraint space) is used to select the best individuals and copy them into the next population. The remaining portion of the new population is filled by mating two parents p_1 and p_2. One parent (p_1) is selected based on a crowding factor using roulette wheel selection. The partner of this individual (p_2) is the result of the competition between two individuals (A and B) picked up by a roulette wheel selection process. To decide between A and B in order to obtain p_2 the following criteria are adopted:

1. If A is feasible and B is not: Select A as the partner p_2.
2. If B is feasible and A is not: Select B as the partner p_2.
3. If both A and B are feasible: Select one with a minimum **Rank Objective**. If the **Rank Objective** of A and B are the same, select one with minimum number of neighbors in the parametric space as a mate.

4. If both A and B are infeasible: Select A or B with a minimum **Rank Constraint**. If the **Rank Constraints** are the same, randomly select between A and B.

The general algorithm is the following [35]:

Begin
Let M be a random initial population.
Do
Generate the Objective and Constraint matrices
Rank individuals based on these matrices
Select elite solutions and copy them into the population
to be used in the next generation
Pick an elite individual
Choose its partner
Mate them and generate two children
Put these children into the population
to be used in the next generation
While termination condition=true
End

A real-coded GA with Simulated Binary Crossover [15] was used to implement this technique. The results reported in two well-known design problems [35] showed that the proposed approach did not reach solutions as good as the other techniques against which it was compared, but it turned out to be less sensitive to parametric variations, which was the main goal of the approach. In constrast, the other techniques analyzed showed significant changes when the parameters were perturbed. The main drawback of this approach is, again, its relative complexity (i.e., its difficulty to implement it), and it would also be desirable that the approach is further refined so that it can get closer to the global optimum than the current available version.

4 A Small Comparative Study

Four techniques were selected from those discussed before to perform a small comparative study that aims to illustrate some practical issues of constraint-handling techniques. The techniques selected are the following: COMOGA [41], the use of VEGA proposed by Coello [12], the NPGA to handle constraints [9] and the approach that uses MOGA [11]. In order to simplify our notation, the last three techniques previously indicated will be called HCVEGA, HCNPGA and HCMOGA, respectively.

To evaluate the performance of the techniques selected, we decided to use two of the well-known benchmark proposed in [31] plus two engineering design problems used in [11]. The full description of the four test functions is the following:

1. **g04**:
   Minimize:

$$f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \quad (18)$$

   subject to:

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

   where: $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$, $27 \leq x_i \leq 45$ $(i = 3, 4, 5)$.
2. **g11**
   Minimize:

$$f(\mathbf{x}) = x_1^2 + (x_2 - 1)^2 \quad (19)$$

   subject to:

$$h(\mathbf{x}) = x_2 - x_1^2 = 0$$

   where: $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$.
3. **Design of a Pressure Vessel**

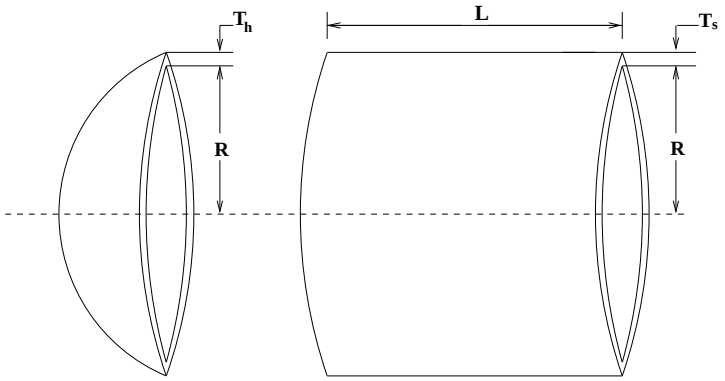

Fig. 2. Center and end section of the pressure vessel used for the third example.

A cylindrical vessel is capped at both ends by hemispherical heads as shown in Figure 2. The objective is to minimize the total cost, including the cost of the material, forming and welding. There are four design

variables: T_s (thickness of the shell), T_h (thickness of the head), R (inner radius) and L (length of the cylindrical section of the vessel, not including the head). T_s and T_h are integer multiples of 0.0625 inch, which are the available thicknesses of rolled steel plates, and R and L are continuous. Using the same notation given by Kannan and Kramer [26], the problem can be stated as follows:
Minimize :

$$F(\mathbf{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (20)$$

Subject to :

$$g_1(\mathbf{x}) = -x_1 + 0.0193x_3 \leq 0 \quad (21)$$

$$g_2(\mathbf{x}) = -x_2 + 0.00954x_3 \leq 0 \quad (22)$$

$$g_3(\mathbf{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0 \quad (23)$$

$$g_4(\mathbf{x}) = x_4 - 240 \leq 0 \quad (24)$$

4. **Design of a 10-bar plane truss**

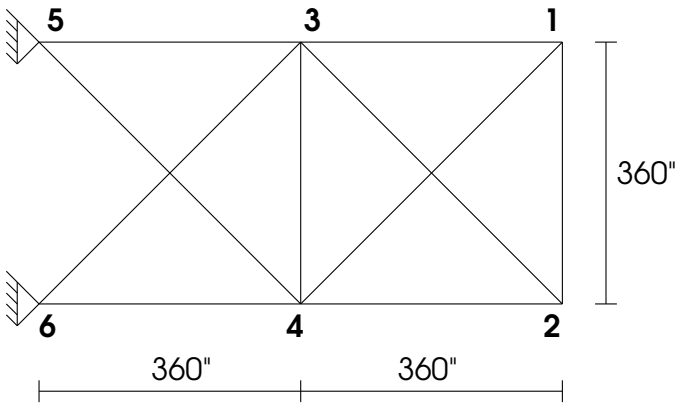

Fig. 3. 10-bar plane truss used for the fourth example.

Consider the 10-bar plane truss shown in Figure 3 [2]. The problem is to find the moment of inertia of each member of this truss, such that we minimize its weight, subject to stress and displacement constraints. The weight of the truss is given by:

$$f(x) = \sum_{j=1}^{10} \rho A_j L_j \quad (25)$$

where x is the candidate solution, A_j is the cross-sectional area of the jth member, L_j is the length of the jth member, and ρ is the weight density of the material.
The assumed data are: modulus of elasticity, $E = 1.0 \times 10^4$ ksi 68965.5 MPa), $\rho = 0.10$ lb/in^3 (2768.096 kg/m^3), and a load of 100 kips (45351.47 Kg) in the negative y-direction is applied at nodes 2 and 4. The maximum allowable stress of each member is called σ_a, and it is assumed to be ± 25 ksi (172.41 MPa). The maximum allowable displacement of each node (horizontal and vertical) is represented by u_a, and it is assumed to be 2 inches (5.08 cm).
There are 10 stress constraints, and 12 displacement constraints (we can really assume only 8 displacement constraints because there are two nodes with zero displacement, but they will nevertheless be considered as additional constraints). The moment of inertia of each element can be different, thus the problem has 10 design variables.

To get a measure of the difficulty of solving each of these problems, a ρ metric (as suggested by Koziel and Michalewicz [27]) was computed using the following expression:

$$\rho = |F|/|S| \tag{26}$$

where $|F|$ is the number of feasible solutions and $|S|$ is the total number of solutions randomly generated. In this work $S = 1,000,000$ random solutions.

Problem	n	Type of function	ρ	LI	NI	LE	NE
1	5	quadratic	27.0079%	0	0	4	2
2	2	quadratic	0.0973%	0	1	0	0
3	4	quadratic	39.6762%	0	0	3	1
4	10	non linear	46.8070%	0	0	0	22

Table 1. Values of ρ for the four test problems chosen.

The different values of ρ for each of the functions chosen are shown in Table 1, where n is the number of decision variables, LI is the number of linear inequalities, NI the number of nonlinear inequalities, LE is the number of linear equalities and NE is the number of nonlinear equalities. It can be clearly seen that the second test function should be the most difficult to solve since it presents the lowest value of ρ.

In our comparative study, we used a binary-coded GA with two-point crossover and uniform mutation. Equality constraints were transformed into inequalities using a tolerance value of 0.001 (see [8] for details of this transformation). The number of fitness function evaluations is the same for all the approaches under study (80,000). The parameters adopted for each of the methods were the following:

- **COMOGA**:
  - Population Size = 200
  - Crossover rate = 1.0
  - Mutation rate = 0.05
  - Desired proportion of feasible solutions = 10 %
  - $\epsilon = 0.01$
- **HCVEGA**:
  - Population Size = 200
  - Number of generations = 400
  - Crossover rate = 0.6
  - Mutation rate = 0.05
  - Tournament size= 5
- **HCNPGA**:
  - Population Size = 200
  - Number of generations = 400
  - Crossover rate = 0.6
  - Mutation rate = 0.05
  - Size of sample of the population = 10
  - Selection Ratio = 0.8
- **HCMOGA**:
  - Population Size = 200
  - Number of generations = 400
  - Crossover rate = 0.6
  - Mutation rate = 0.05

A total of 100 runs per technique per problem were performed. Statistical results are presented in Tables 2, 3, 4 and 5, where P_i refers to the problem solved ($1 \leq i \leq 4$), $0.0 \leq F_p \leq 1.0$ is the average rate of feasible solutions found during a single run (with respect to the full population).

	COMOGA						
P	Optimal	Best	Median	Mean	St. Dev.	Worst	F_p
P_1	−30665.539	−30533.056641	−30328.199219	−30329.563398	74.793290	−30141.033203	0.002358
P_2	0.750	0.749058	0.749787	0.749829	0.000495	0.751753	0.000286
P_3	6059.946341	6369.428223	7889.838867	7795.411538	701.363966	9147.520508	0.003989
P_4	5152.636136	6283.198730	6675.126709	6660.455649	126.289250	6968.627441	0.006516

Table 2. Experimental results using COMOGA with the four test problems.

5 Discussion of Results

Although this study is too limited to derive general conclusions, there are a few interesting issues that deserve discussion. First, it is important to emphasize that all of the approaches tested were able to reach the feasible region in

P	HCVEGA						
	Optimal	Best	Median	Mean	St. Dev.	Worst	F_p
P_1	−30665.539	−30647.246094	−30628.587891	−30628.468711	7.877054	−30607.240234	0.408108
P_2	0.750	0.749621	0.812369	0.798690	0.025821	0.847242	0.011206
P_3	6059.946341	6064.723633	6238.489746	6259.963745	170.254024	6820.944824	0.425354
P_4	5152.636136	5327.418457	5453.446045	5455.871895	56.744241	5569.240723	0.676408

Table 3. Experimental results using HCVEGA to handle constraints with the four test problems.

P	HCNPGA						
	Optimal	Best	Median	Mean	St. Dev.	Worst	F_p
P_1	−30665.539	−30661.033203	−30635.346680	−30630.883145	20.466057	−30544.324219	0.345432
P_2	0.750	0.749001	0.749613	0.753909	0.012147	0.832940	0.025842
P_3	6059.946341	6059.926270	6127.618408	6172.527373	123.897547	6845.770508	0.330693
P_4	5152.636136	5179.740723	5256.108154	5259.013174	37.658930	5362.890625	0.503566

Table 4. Experimental results using HCNPGA to handle constraints with the four test problems.

Problem	HCMOGA						
	Optimal	Best	Median	Mean	St. Dev.	Worst	F_p
P_1	−30665.539	−30649.958984	−30570.754883	−30568.917930	53.531272	−30414.773438	0.344896
P_2	0.750	0.749001	0.749155	0.749393	0.000595	0.752445	0.016913
P_3	6059.946341	6066.969727	6561.483154	6629.064048	385.110736	7547.403320	0.452672
P_4	5152.636136	5336.618652	5745.238281	5748.839526	210.696096	6474.041992	0.603598

Table 5. Experimental results using HCMOGA to handle constraints with the four test problems.

all of the test functions. We also found that HCNPGA was the most robust approach. This technique was able to find the best results in all of the test functions, except for **g11** in which there was a tie with HCMOGA. Note that the mean values obtained by the HCNPGA is not always the best of all the approaches in all of the test functions (see for example **g04** and **g11**), but it tends to have the lowest standard deviations, which is an indicative of a more robust performance.

COMOGA presented its best performance in function **g11**, which is the one with the most heavily constrained search space. HCMOGA normally presented the best statistical performance (i.e., lowest standard deviations) of all the approaches. HCVEGA presented its best performance in **g04** but its best result was still relatively far from the global optimum.

The high selection pressure of the nongenerational GA used in COMOGA made it difficult to avoid premature convergence. This is reflected by the F_p values produced by this approach, which are lower than those generated by the three other approaches.

The population-based approach used by the HCVEGA was the most robust in the test function with the highest number of decision variables. This seems to suggest that this sort of approach may be more effective in problems with high dimensionality. In fact, we have found in other studies that most constraint-handling techniques based on Pareto dominance tend to degrade their performance as the number of decision variables increases.

HCMOGA performed well in all four problems but its overall performance was not as good as that of the HCNPGA. However, in problem 4 (**g11**) the HCMOGA presented the most consistent behavior and the best solution that it found was the same as the one produced by the HCNPGA. This seems to suggest that ranking all the population may be useful to deal with highly constrained spaces. The obvious drawback is its computational cost.

Dealing with a high number of constraints (e.g., problem 4) was not difficult for any of the approaches, except for COMOGA which could not find good results.

Although not conclusive, this study seems to indicate that Pareto dominance, Pareto ranking and population-based mechanisms are promising approaches to handle constraints. These results also seem to suggest that a traditional (i.e., generational) GA performs better in optimization problems than nongenerational GAs.

Regarding diversity, the four approaches exhibited a good performance, which seems to indicate that the search space is well sampled by all of them. However, note that none of the four approaches could reach the global optimum. This indicates that further refinements are required to improve the effectiveness of these approaches.

The use of a sample of the population to determine the Pareto dominance of an individual was found to be appropriate in this context. This is consistent with the behavior reported for the NPGA in multiobjective optimization problems [10]. However, in problems with highly constrained search spaces, Pareto ranking of the entire population seems to be advantageous. On the other hand, population-based approaches seem to be less sensitive to scalability of the decision variable space. An open question is if the advantages of each of these techniques can be combined into a single approach.

6 Conclusions and Future Work

A set of constraint-handling techniques based on multiobjective concepts were presented in this chapter. In each case, advantages and disadvantages were discussed. We also presented a small comparative study in which four of the techniques discussed were implemented and evaluated using four test functions. Our results provided some insights regarding the behavior of each type of technique. Note however, that comparisons with respect to traditional penalty functions [37, 40] and with the most competitive constraint-handling techniques used with EAs (e.g., stochastic ranking [38], the homomorphous maps [27], and the adaptive segregational constrained handling evolutionary algorithm (ASCHEA) [21]) are still lacking.

The results obtained seem to indicate that techniques based on multiobjective optimization can properly deal with constrained search spaces. However, such results also seem to indicate that additional mechanisms should be used

to improve the effectiveness of these approaches, since they had obvious difficulties to reach the global optimum in all the test functions used.

Some of the most promising paths of future research in this area are the following:

- Incorporation of self-adaptation or on-line adaptation to make unnecessary the fine tuning of additional parameters.
- Exploitation of domain knowledge extracted from the evolutionary process to improve the results obtained by the EA.
- Hybridization with classical global optimization techniques and/or with other heuristics.

Acknowledgments

This work is representative of the research performed by the Evolutionary Computation Group at CINVESTAV-IPN (EVOCINV). The first author acknowledges support from the mexican Consejo Nacional de Ciencia y Tecnología (CONACyT) through a scholarship to pursue graduate studies at CINVESTAV-IPN's Electrical Engineering Department. The second author acknowledges support from (CONACyT) through project number 32999-A.

References

1. Dirk V. Arnold. *Noisy Optimization with Evolution Strategies*. Kluwer Academic Publishers, New York, June 2002. ISBN 1-4020-7105-1.
2. Ashok Dhondu Belegundu. *A Study of Mathematical Programming Methods for Structural Optimization*. Department of civil and environmental engineering, University of Iowa, Iowa, Iowa, 1982.
3. Jürgen Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, New York, December 2001. ISBN 0-7923-7631-5.
4. Martin V. Butz. *Anticipatory Learning Classifier Systems*. Kluwer Academic Publishers, New York, December 2001. ISBN 0-7923-7630-7.
5. 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.
6. Erick Cant-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, New York, November 2000. ISBN 0-7923-7221-2.
7. V. Chankong and Y.Y Haimes. Multiobjective Decision Making: Theory and Methodology. In Andrew P. Sage, editor, *Systems Science and Engineering*. North-Holland, 1983.
8. Carlos A. Coello Coello. Theoretical and Numerical Constraint Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, January 2002.

9. 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.
10. Carlos A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.
11. Carlos A. Coello Coello. Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering and Environmental Systems*, 17:319–346, 2000.
12. Carlos A. Coello Coello. Treating Constraints as Objectives for Single-Objective Evolutionary Optimization. *Engineering Optimization*, 32(3):275–308, 2000.
13. Carlos A. Coello Coello and Arturo Hernández Aguirre. Design of Combinational Logic Circuits through an Evolutionary Multiobjective Optimization Approach. *Artificial Intelligence for Engineering, Design, Analysis and Manufacture*, 16(1):39–53, 2002.
14. Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, June 2002. ISBN 0-3064-6762-3.
15. Kalyanmoy Deb and R. W. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148, 1995.
16. 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.
17. Francis Ysidro Edgeworth. *Mathematical Physics*. P. Keagan, London, England, 1881.
18. 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 Kaufmann Publishers.
19. David Goldberg. *The Design of Innovation*. Kluwer Academic Publishers, New York, June 2002. ISBN 1-4020-7098-5.
20. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Co., Reading, Massachusetts, 1989.
21. S. Ben Hamida and Marc Schoenauer. An Adaptive Algorithm for Constrained Optimization Problems. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 529–538, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.
22. Robert Hinterding and Zbigniew Michalewicz. Your Brains and My Beauty: Parent Matching for Constrained Optimisation. In *Proceedings of the 5th International Conference on Evolutionary Computation*, pages 810–815, Anchorage, Alaska, May 1998.
23. 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.

24. F. Jiménez, A.F. Gómez-Skarmeta, and G. Sánchez. How Evolutionary Multiobjective 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 Espaõl de Algoritmos Evolutivos y Bioinspirados (AEB'02)*, pages 460–465, Mérida España, 2002. Universidad de la Extremadura, España.
25. Fernando Jiménez and José L. Verdegay. Evolutionary techniques for constrained optimization problems. In *7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99)*, Aachen, Germany, 1999. Springer-Verlag.
26. B. K. Kannan and S. N. Kramer. An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design. *Journal of Mechanical Design. Transactions of the ASME*, 116:318–320, 1994.
27. Slawomir Koziel and Zbigniew Michalewicz. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary Computation*, 7(1):19–44, 1999.
28. Nivedita Sumi Majumdar and Dipankar Dasgupta. Determining Optimal Configuration for Turbine Generator Cooler. In *Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001)*, volume 2, pages 1009–1014, Piscataway, New Jersey, May 2001. IEEE Service Center.
29. Efrén Mezura-Montes. Uso de la Técnica Multiobjetivo NPGA para el Manejo de Restricciones en Algoritmos Genéticos. Master's thesis, Universidad Veracruzana, Xalapa, México, 2001. (In Spanish).
30. Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, third edition, 1996.
31. Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.
32. Shigeru Obayashi. Pareto Solutions of Multipoint Design of Supersonic Wings using Evolutionary Algorithms. In I.C. Parmee, editor, *Adaptive Computing in Design and Manufacture V*, pages 3–15, London, 2002. Springer-Verlag.
33. Vilfredo Pareto. *Cours D'Economie Politique*, volume I and II. F. Rouge, Lausanne, 1896.
34. 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.
35. Tapabrata Ray. Constraint Robust Optimal Design using a Multiobjective Evolutionary Algorithm. In *Proceedings of the Congress on Evolutionary Computation 2002 (CEC'2002)*, volume 1, pages 419–424, Piscataway, New Jersey, May 2002. IEEE Service Center.
36. Tapabrata Ray, Tai Kang, and Seow Kian Chye. An Evolutionary Algorithm for Constrained Optimization. In Darrell Whitley, David Goldberg, Erick Cantú-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2000)*, pages 771–777, San Francisco, California, July 2000. Morgan Kaufmann.

37. 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.
38. Thomas P. Runarsson and Xin Yao. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.
39. 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.
40. 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.
41. Patrick D. Surry and Nicholas J. Radcliffe. The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms. *Control and Cybernetics*, 26(3):391–412, 1997.
42. Patrick D. Surry, Nicholas J. Radcliffe, and Ian D. Boyd. A Multi-Objective Approach to Constrained Optimisation of Gas Supply Networks : The COMOGA Method. In Terence C. Fogarty, editor, *Evolutionary Computing. AISB Workshop. Selected Papers*, Lecture Notes in Computer Science, pages 166–180. Springer-Verlag, Sheffield, U.K., 1995.
43. K.C. Tan, T.H. Lee, J. Cai, and Y. H. Chew. Automating the Drug Scheduling of Cancer Chemotherapy via Evolutionary Computation. In *Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001)*, volume 2, pages 908–913, Piscataway, New Jersey, May 2001. IEEE Service Center.
44. K. Tesh, M.A. Atherton, and M.W. Collins. Genetic Algorithms Search for Stent Design Improvements. In I.C. Parmee, editor, *Adaptive Computing in Design and Manufacture V*, pages 99–107, London, 2002. Springer-Verlag.