Constrained Optimization via Multiobjective Evolutionary Algorithms

Efrén Mezura-Montes[1] and Carlos A. Coello Coello[2*]

[1] Laboratorio Nacional de Informática Avanzada (LANIA A.C.)
Rébsamen 80, Centro, Xalapa, Veracruz, 91000 MEXICO
emezura@lania.mx

[2] CINVESTAV-IPN (Evolutionary Computation Group)
Departamento de Computación
Av. IPN No. 2508, Col. San Pedro Zacatenco, México D.F. 07360, MEXICO
ccoello@cs.cinvestav.mx

Summary. In this chapter, we present a survey of constraint-handling techniques based on evolutionary multiobjective optimization concepts. We present some basic definitions required to make this chapter self-contained, and then we introduce the way in which a global (single-objective) nonlinear optimization problem is transformed into an unconstrained multiobjective optimization problem. A taxonomy of methods is also proposed and each of them is briefly described. Some interesting findings regarding common features of the approaches analyzed are also discussed.

1 Introduction

Nowadays, evolutionary algorithms (EAs) have become a popular choice to solve different types of optimization problems [20, 33, 45]. Indeed, this chapter points out the application of some ideas originally designed to solve an specific type of optimization problems using EAs, and which are now applied to solve a different type of problems. Despite being considered powerful search engines, EAs, in their original versions, lack a mechanism to incorporate constraints into the fitness function in order to solve constrained optimization problems. Hence, several approaches have been proposed to deal with this issue. Michalewicz & Schoenauer [36] and Coello Coello [7] have presented comprehensive surveys about constraint-handling techniques used with EAs. As indicated in such surveys, the most popular method adopted to handle constraints in EAs was taken from the mathematical programming literature: penalty functions (mostly exterior). Penalty functions were originally proposed by Courant in the 1940s [11] and later expanded by Carroll [4] and Fiacco & McCormick [17]. The idea of this method is to transform a constrained

[*] Corresponding author

optimization problem into an uncontrained one by adding (or subtracting) a certain value to/from the objective function based on the amount of constraint violation present in a certain solution. This aims to favor feasible solutions over infeasible ones during the selection process. The main advantage of the use of penalty functions is their simplicity. However, their main shortcoming is that penalty factors, which determine the severity of the punishment, must be set by the user and their values are problem-dependent [50, 7]. This has motivated the design of alternative techniques like those based on special encodings and operators [34, 48] and on repair algorithms [35].

Unlike penalty functions, which combine the objective function and the constraints values into one fitness value, there are other approaches which handle these two values separately. The most representative approaches, which work based on this idea are two: (1) the methods based on the superiority of feasible points [41, 15] and (2) the methods based on evolutionary multiobjective optimization concepts.

This chapter focuses on the last type of techniques (i.e., those based on multiobjective optimization concepts) and describes, tests and criticizes them.

In order to present our discussion of methods in a more organized way, we propose a simple taxonomy of techniques, based on the way they transform the nonlinear programming problem (NLP) into a multiobjective optimization problem (MOP):

1. Approaches which transform the NLP into an unconstrained bi-objective optimization problem (the original objective function and the sum of constraint violation).
2. Techniques which transform the NLP into an unconstrained MOP where the original objective function and each constraint of the NLP are treated as separate objectives. From this category, we observed two further subcategories:
   a) Methods which use non-Pareto concepts (mainly based on multiple populations) and
   b) techniques which use Pareto concepts (ranking and dominance) as their selection criteria.

The remainder of this chapter is organized as follows. In Section 2 we present the general NLP, and we recall some multiobjective optimization concepts used in this survey; we also show the transformation of a NLP into a MOP. After that, in Section 3 the approaches which solve the NLP as a bi-objective problem (using the original objective function and the sum of constraint violation) are presented. Later on, Section 4 shows techniques based on solving the problem by taking the original objective function and each of the constraints of the problem as different objectives, either by using Pareto and non-Pareto concepts. In Section 5, we provide some highlights of the methods previously discussed. A small comparative experiment using four representive approaches (from those previously discussed) is presented in Section 6. Finally, Section 7 presents our conclusions and some possible future paths of research in the area.

2 Problem definition and transformation

In the following definitions we will assume (without loss of generality) minimization. The general NLP is defined as to:

$$\text{Find } \mathbf{X} \text{ which minimizes } f(\mathbf{X}) \tag{1}$$

subject to:

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

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

where $\mathbf{X} \in \mathbb{R}^n$ is the vector of solutions $\mathbf{X} = [x_1, x_2, \ldots, x_n]^T$, where each $x_i, \quad i = 1, \ldots, n$ is bounded by lower and upper limits $L_i \leq x_i \leq U_i$ which define the search space $\mathcal{S}$, $\mathcal{F}$ is the feasible region and $\mathcal{F} \subseteq \mathcal{S}$; m is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or nonlinear).

When solving NLPs with EAs, equality constraints are usually transformed into inequality constraints of the form:

$$g_j(\mathbf{X}) = |h_j(\mathbf{X})| - \epsilon \leq 0, \quad j = m+1, m+2, \ldots, m+p \tag{4}$$

where ϵ is the tolerance allowed (a very small value). In the rest of the chapter we will refer only to inequality constraints because we will assume this transformation.

As it was discussed in Chapter **??**, in a multiobjective problem, the optimum solution consists on a set of ("trade-off") solutions, rather than a single solution as in global optimization. This optimal set is known as the Pareto Optimal set.

Based on the review of the literature that we undertook, we found that researchers have adopted two different ways to transform the NLP into a MOP:

1. The first approach transforms the NLP into an unconstrained bi-objective problem. The first objective is the original objective function and the second one is the sum of constraint violation as follows: optimize $\mathbf{F}(\mathbf{X}) = (f(\mathbf{X}), G(\mathbf{X}))$, where $G(\mathbf{X}) = \sum_{i=1}^{m+p} \max(0, g_i(\mathbf{X}))$ and each $g_i(\mathbf{X}), i = 1, \ldots, m+p$ must be normalized.
   Note however, that in this case when solving a transformed NLP, we are not looking for a set of solutions (as described in Chapter **??**). Instead, we seek a single solution, the global constrained optimum, where: $f(\mathbf{X}) \leq f(\mathbf{Y})$ for all feasible $\mathbf{Y}$ and $G(\mathbf{X}) = 0$.

2. The second approach transforms the problem into an unconstrained MOP, in which we will have $k+1$ objectives, where k is the total number of constraints $(m+p)$ and the additional objective is the original NLP objective function. Then, we can apply a multiobjective evolutionary algorithm to the new vector $\mathbf{F}(\mathbf{X}) = (f(\mathbf{X}), g_1(\mathbf{X}), \ldots, g_{m+p}(\mathbf{X}))$, where $g_1(\mathbf{X}), \ldots, g_{m+p}(\mathbf{X})$ are the original constraints of the problem.
   As indicated before, we are looking again for the global constrained optimum instead of a set of trade-off solutions. Thus, we require the following: $g_i(\mathbf{X}) = 0$ for $1 \leq i \leq (m+p)$ and $f(\mathbf{X}) \leq f(\mathbf{Y})$ for all feasible $\mathbf{Y}$.

These apparently subtle differences in the way of stating a MOP prompt to changes in the way multiobjective concepts are applied (i.e., it influences the way in which nondominated, Pareto ranking and multipopulation-based techniques are actually used). In the following sections, we will describe the approaches reported in the specialized literature to deal with this special type of MOP that arises from a transformed NLP.

It is important to note that the use of multiobjective optimization concepts improves the solution procedure of a constrained problem with respect to a typical penalty function in two aspects: (1) No penalty factors must be tuned and (2) the way to approach the feasible region becomes more robust because of the trade-offs among objective function and constraints of the problem. In constrast, in a typical penalty function this path to the constrained optimum is rather rigid and fixed. In fact, a penalty function is forcing the search to generate feasible solutions. On the other hand, by using a multiobjective approach the aim is to decrease the violation of constraints, but at the same time we look for objective function improvement, and this behavior may lead to reaching the feasible region from different (and maybe promising) directions.

3 Transforming the NLP into a Bi-Objective problem

Surry & Radcliffe [51] proposed COMOGA (Constrained Optimization by Multiobjective Optimization Genetic Algorithms) where individuals are Pareto-ranked based on the sum of constraint violation. Then, solutions can be chosen using binary tournament selection based either on their rank or their objective function value. This decision is based on a parameter called P_{cost} whose value is modified dynamically. The aim of the proposed approach to solve this bi-objective problem is based on reproducing solutions which are good in one of the two objectives with other competitive solutions in the other objective i.e., constraint violation (as Shaffer's Vector Evaluated Genetic Algorithm (VEGA) promoted to solve MOPs [47]). COMOGA was tested on a gas network design problem providing slightly better results than those obtained with a penalty function approach. Its main drawback is that it requires several extra parameters. Also, to the authors' best knowledge, this approach has not been used by other researchers.

Camponogara & Talukdar [3] proposed to solve the bi-objective problem in the following way: A set of Pareto fronts in the bi-objective space is generated by the EA. Two of them (S_i and S_j, where $i < j$) are selected. After that, two solutions $x_i \in S_i$ and $x_j \in S_j$ where x_i dominates x_j are chosen. Based in these two points, a search direction is generated as follows:

$$d = \frac{(x_i - x_j)}{|x_i - x_j|} \tag{5}$$

A line search begins by projecting 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. Other mechanism of the approach allows, at pre-defined intervals, to replace the worst half of the population with new random solutions 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. Furthermore, it is not clear what is the impact of the segment chosen to search on the overall performance of the algorithm.

Zhou et al. [55] proposed a ranking procedure based on Pareto strength [56] for the bi-objective problem, i.e. to count the number of individuals which are dominated for a given solution. Ties are solved by the sum of constraint violation (second objective in the problem). The Simplex crossover (SPX) operator is used to generate a set of offspring where the individual with the highest Pareto strength and also the solution with the lowest sum of constraint violation are both selected to take part in the population for the next generation. The approach was tested on a subset of a well-known benchmark for evolutionary constrained optimization [30]. The results were competitive but the authors had to use different sets of parameters for different functions, which made evident the high sensitivity of the approach to the values of its parameters.

Wang and Cai [54] used a framework similar to the one proposed by Zhou et al. [55] because they also used the SPX with a set of parents to generate a set of offspring. However, instead of using just two individuals from the set of offspring, all nondominated solutions (in the bi-objective space) are used to replace the dominated solutions in the parent population. Furthermore, they use an external archive to store infeasible solutions with a low sum of constraint violation in order to replace some random solutions in the current population. The idea is to maintain infeasible solutions close to the boundaries of the feasible region in order to perform a better sampling of this region as to find optimum solutions located there (i.e., when dealing with active constraints) [28]. The approach provided good results in 13 well-known test problems. However, different sets of values for the parameters were used, depending of the dimensionality of the problem.

Venkatraman and Yen [52] proposed a generic framework to solve the NLP. The approach is divided in two phases: The first one treats the NLP as a constraint satisfaction problem i.e., the goal is to find at least one feasible solution. To achieve that, the population is ranked based only on the sum of constraint violation. The second phase starts when the first feasible solution has been found. Now both objectives (the original objective function and the sum of constraint violation) are taken into account and nondominated sorting [16] is used to rank the population (alternatively, the authors proposed a preference scheme based on feasibility rules [15], but in their experiments, they found that nondominated sorting provided better results). Also, to favor diversity, a niching scheme based on the distance of the nearest neighbors to each solution is applied. To decrease the effect of differences in values, all constraints are normalized before calculating the sum of those which are violated. The approach used a typical GA as a search engine with 10% elitism. The approach provided good quality results in 11 well-known benchmark problems and in some problems generated with the Test-Case Generator tool [32], but lacked consistency due to the fact that the way to approach the feasible region is mostly at random because of the first phase which only focuses on finding a feasible solution, regardless of the region from which the feasible region is approached.

Wang et al. [53] also solved the bi-objective problem but using selection criteria based on feasibility very similar to those proposed by Deb [15], where a feasible solution is preferred over an infeasible one; between two feasible solutions, the one with the best objective function value is selected and, finally, between two infeasible solutions, the one with the lowest sum of constraint violation is chosen. Furthermore, they proposed a new crossover operator based on uniform design methods [53]. This operator is able to explore regions closer to the parents. Finally, Gaussian noise is used as a mutation operator. The approach was tested on a subset of a well-known benchmark used to test evolutionary algorithms in constrained optimization [30]. No details are given by the authors about the influence of the extra parameters required to control the crossover operator (q) and the number of offspring generated (r).

4 Transforming the NLP into a multiobjective problem with objective function and constraints as separated objectives

As indicated before, in this case we may use non-Pareto schemes or Pareto schemes. Each of these two sub-classes of methods will be discussed next.

4.1 Techniques based on non-Pareto schemes

Parmee & Purchase [40] used the idea proposed in VEGA [47] 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. The aim of VEGA is to divide the population into sub-populations, and each sub-population has then the goal of optimizing only one objective. In this case, the set of objectives are the constraints of the problem. Genetic operators are applied to all solutions regardless of the sub-population of each solution. In Parmee's approach, once the feasible region is reached, special operators are used to improve the feasible solutions found. The use of these special operators that preserve feasibility makes this approach highly specific to one application domain rather than providing a general methodology to handle constraints.

Schoenauer & Xanthakis [49] proposed a constraint-handling technique based on the notion of *behavioral memory* [13], which takes into account the information contained in the whole population after some genetic evolution. As it turns out, this approach consists of a form of "lexicographic ordering" [10]. The main idea is to satisfy each constraint of the problem in a sequential order. Once a certain number of solutions in the population satisfy the first constraint (based on a parameter of the approach), the second constraint is added in order to be also satisfied, but always enforcing that the solutions satisfy the first one. In this way, solutions which satisfy the second constraint but not the first one, will be removed from the population (like in a death-penalty approach [7]). The success of the approach normally depends on the order in which constraints are processed. Besides, it may not be appropriate when solving problems with a large feasible region (with respect to the whole search space). However, this technique may be very effective to solve problems where constraints have a natural hierarchy to be evaluated.

Coello Coello [6] also used VEGA's idea [47] to solve NLPs. At each generation, the population was split into $m+1$ sub-populations of equal (fixed) size, where m is the number of constraints of the problem. The additional sub-population handled the objective function of the problem and the individuals contained within it were selected based on the unconstrained objective function value. The m remaining sub-populations took one constraint of the problem each as their fitness function. The aim was that each of the sub-populations tried to reach the feasible region corresponding to one individual constraint. By combining these different sub-populations, the approach would then reach the feasible region of the problem considering all of its constraints. The main drawback of the approach is that the number of sub-populations increases linearly with respect to the number of constraints.

This issue was further tackled by Liang and Suganthan [27], where a dynamic particle multi-swarm optimization was proposed. They also used VEGA's idea to split the swarm into sub-swarms and each sub-swarm optimized one objective. However, in this case, the sub-swarms are assigned dynamically. In this way, the number of sub-swarms depends on the complexity of the constraints to be satisfied instead of depending on the number of constraints. The authors also included a local search mechanism based on sequential quadratic programming to improve values of a set of randomly chosen *pbest* values. The approach provided competitive results in the extended version of a well-known benchmark adopted for evolutionary constrained optimization [27]. The main drawbacks of the approach are that it requires extra parameters to be tuned by the user and it also presented a strong dependency on the local search mechanism.

4.2 Techniques based on Pareto schemes

Carlos Fonseca was apparently the first to propose the idea of using the Pareto dominance relation to handle constraints [12, 19]. His proposal consisted of modifying the definition of Pareto dominance in order to incorporate constraints. It is worth noting that this proposal was really a small component of a multiobjective evolutionary algorithm (MOGA [18]), and was, therefore, mainly used to solve constrained multiobjective optimization problems. Because of this, Fonseca's proposal did not attract much interest from researchers working with constraint-handling techniques for single-objective optimization. Next, we will discuss several other constraint-handling techniques that directly incorporate Pareto-based schemes.

Jiménez et al. [24] proposed an approach that transforms the NLP (and also the constraint satisfaction and goal programming problems) into a MOP by assigning priorities. Regarding the NLP, constraints are assigned a higher priority than the objective function. Then, a multiobjective algorithm based on a pre-selection scheme is applied. This algorithm generates from two parents a set of offspring which will be also mutated to generate another set. The best individual from the first set of offspring (non-mutated) and the best one of the mutated ones, will replace each of the two parents. The idea is to favor the generation of individuals close to their parents and to promote implicit niching. Comparisons among individuals are made by using Pareto dominance. A real-coded GA was used as a search engine with two types of crossover operators (uniform and arithmetic) and two mutation operators (uniform and non-uniform). The results on 11 problems taken from a well-known benchmark [30] were promising. The main drawback of the approach is the evident lack of knowledge about the effect of the parameter "q" related with the pre-selection scheme, which the authors do not discuss in their paper. Also, the authors do not provide any information regarding the number of evaluations performed by the approach.

Coello Coello [5] proposed a ranking procedure based on a counter which was incremented based on the number of individuals in the population which dominated a given solution based on some criteria (feasibility, sum of constraint violation and number of constraints violated). The approach was tested on a set of engineering design problems providing competitive results. An adaptive mechanism was also implemented in order to fine tune the parameters of the approach. Its main drawbacks are the computational cost of the technique and its difficulties to handle equality constraints [29].

Ray et al. [42, 44] 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 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 with respect to other state-of-the-art approaches. Its main drawback is that its implementation is considerably more complex than that of any of the other techniques previously discussed.

Ray extended his work to a simulation of social behavior [1, 43], where a societies-civilization model is proposed. Each society has its leaders which influence their neighbors. Also, the leaders can migrate from one society to another, promoting exploration of new regions of the search space. Constraints are handled by a nondominated sorting mechanism [16] in the constraints space. A leader centric operator is used to generate movements of the neighbors influenced by their leaders. The main drawback of the approach is its high computational cost derived from the nondominated sorting and a clustering technique required to generate the societies. Results reported on some engineering design problems are very competitive. However, to the authors' best knowledge, this technique has not been compared against state-of-the-art approaches adopting the same benchmark [30].

Coello Coello and Mezura-Montes [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 Coello and Mezura-Montes' approach does not require niches (or fitness sharing [14]) to maintain diversity in the population. Instead, it 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 (based on Deb's proposal [15]), but when both solutions are infeasible, a dominance criterion in the constraints space is used to select the best solution. The remaining $1 - S_r$ parents are selected using a purely probabilistic approach. Results indicated that the approach was robust, efficient and effective. However, it was also found that the approach had scalability problems (its performance degraded as the number of decision variables increased).

The use of dominance to select between two infeasible solutions was taken to the differential evolution metaheuristic by Kukkonen and Lampinen [26]. In their approach, when the comparison between a parent vector and its child vector is performed and both of them are infeasible, a dominance criterion is applied. The results on the extended version of the benchmark [26] were very competitive.

Angantyr et al. [2] proposed to assign a fitness value to solutions based on a two-ranking mechanism. The first rank is assigned according to the objective function value (regardless of feasibility). The second rank is assigned by using nondominated sorting [16] in the constraints space. These ranks have adaptive weights when defining the fitness value. The aim is to guide the search to the unconstrained optimum solution if there are many feasible solutions in the current population. If the rate of feasible solutions is low, the search will be biased to the feasible region. The goal is to promote an oscillation of the search between the feasible and infeasible regions of the search space. A typical GA with BLX crossover was used. The main advantage of this approach is that it does not add any extra parameters to the algorithm. However, it presented some problems when solving functions with equality constraints [2].

Hernandez et al. [21] proposed an approach named IS-PAES which is based on the Pareto Archived Evolution Strategy (PAES) originally proposed by Knowles and Corne [25]. IS-PAES uses an external memory to store the best set of solutions found. Furthermore, IS-PAES requires a shrinking mechanism to reduce the search space. The multiobjective concept is used in this case as a secondary criterion (Pareto dominance is used only to decide whether or not a new solution is inserted in the external memory). The authors acknowledge that the most important mechanisms of IS-PAES are its shrinking procedure and the information provided by the external memory which is used to decide the shrinking of the search space. Furthermore, despite its good performance as a global optimizer, IS-PAES is an approach far from simple to implement.

Runarsson and Yao [46] presented a comparison of two versions of Pareto ranking applied in the constraints space: (1) considering the objective function value in the ranking process and (2) without considering it. These versions were compared against a typical over-penalized penalty function approach. The authors found in their work that using Pareto ranking leads to bias-free search, and thus concluded that it causes the search to spend most of the time searching in the infeasible region. Therefore, the approach is unable to find feasible solutions (or finds feasible solutions with a poor value of the objective function).

Oyama et al. [39] used a similar approach to the one proposed by Coello Coello and Mezura-Montes [9]. However, the authors proposed to use a set of criteria based on feasibility to rank all the population (instead of using them in a tournament [9]). Moreover, this approach is designed to solve also constrained multiobjective optimization problems. A real-coded GA with BLX crossover was used as the search engine. This technique was used to solve one engineering design problem and also a real-world NLP. No further experiments or comparisons were provided.

5 Remarks

Based on the features found in each of the methods previously discussed, we highlight the following findings:

- The transformation of the NLP into a multiobjective problem with constraints and objective function as separated objectives is a more popular approach than the transformation of the NLP to a bi-objective optimization problem.
- The use of sub-populations has been the least popular, although they may present certain advantages in some particular optimization problems (see for example [8]).
- There seems to be a certain trend towards using mean-centric crossover operators (BLX [2, 39], random-mix [42, 44], SPX [54, 55]) over using parent-centric crossover (uniform design methods [53], leader centric operator [1, 43]) when adopting real-coded GAs. Furthermore, other authors used more than one crossover operator (uniform and arithmetic [24]). This choice seems to contradict the findings about competitive crossover operators that have been reported by other researchers when using other constraint-handling techniques such as GENOCOP and penalty functions [37, 38].
- The use of diversity mechanisms is found in most approaches, which is a clear indication of the loss of diversity experienced when adopting multi-objective optimization schemes for handling constraints [51, 3, 54, 52, 42, 44, 1, 43, 9, 2, 21].
- The use of explicit local search mechanisms is still scarce, despite the evident advantages that such mechanisms may bring into this area [27].
- The difficulty of using Pareto concepts when solving the NLP pointed out by Runarsson and Yao [46] has been confirmed by other researchers like Mezura-Montes and Coello Coello [29]. However, the methods described in this survey provide several alternatives to deal with the inherent shortcoming for the lack of bias provided by Pareto ranking.

6 A Limited 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 based on multiobjective concepts. The techniques selected are the following: COMOGA [51] which tranforms the constrained problem into a bi-objective problem, the use of VEGA proposed by Coello Coello [6] which handles a problem of "$m + p + 1$" objectives with the same number of sub-populations (where m is the number of inequality constraints and p is the number of equality constraints), the NPGA to handle constraints [9] which calculates Pareto Dominance in the constraints space (the number of objectives depends of the number of constraints of the problem) and the approach that uses Pareto ranking (called MOGA by us although it does not exactly follow Fonseca's proposal [18]) [5] where dominance is computed based on separated objectives (number of violated constraints and amount of constraint violation).

These techniques were chosen because all of them mainly modify the parent selection scheme of an EA and do not use additional mechanisms (specialized crossover or mutation operators, external memory, etc.). Therefore, and because of their simplicity they can be included inside a typical EA without further changes. In order to simplify our notation, the last three techniques previously indicated will be called CHVEGA, CHNPGA and CHMOGA, respectively (CH stands for constraint-handling).

To evaluate the performance of the techniques selected, we decided to use a well-known benchmark found in the specialized literature [36] and also three engineering design problems [31]. The Appendix at the end of the chapter includes the details of all the test functions adopted.

To get an estimate of how difficult is to generate feasible points through a purely random process, we computed the ρ metric (as suggested by Michalewicz and Schoenauer [36]) using the following expression:

$$\rho = |F|/|S| \tag{6}$$

where $|S|$ is the number of random solutions generated ($S = 1,000,000$ in our case), and $|F|$ is the number of feasible solutions found (out of the total $|S|$ solutions randomly generated).

Problem	**n**	**Type of function**	ρ	**LI**	**NI**	**LE**	**NE**
1	5	quadratic	27.0079%	0	6	0	0
2	2	non linear	0.0057%	0	2	0	0
3	10	quadratic	0.0000%	3	5	0	0
4	7	non linear	0.5199%	0	4	0	0
5	8	linear	0.0020%	3	3	0	0
6	2	quadratic	0.0973%	0	0	0	1
7	4	quadratic	2.6859%	6	1	0	0
8	4	quadratic	39.6762%	3	1	0	0
9	3	quadratic	0.7537%	1	3	0	0

Table 1. Main features of the 9 test problems used. 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.

The different values of ρ besides the main features of each test function are shown in Table 1.

In our comparative study, we used a binary-gray-coded GA with two-point crossover and uniform mutation. Equality constraints were transformed into inequalities using a tolerance value of 0.001 (see [7] 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 %
  - $\epsilon = 0.01$
- **CHVEGA**:
  - Population Size = 200
  - Number of generations = 400
  - Crossover rate = 0.6
  - Mutation rate = 0.05
  - Tournament size= 5
- **CHNPGA**:
  - 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
- **CHMOGA**:
  - 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. A summary of all results is shown in Table 2, where P_i refers to the problem solved ($1 \leq i \leq 9$).

6.1 Discussion of Results

Based on the obtained results, all of them summarized in Table 2, we will focus our discussion on the following topics:

- **Quality**: Which approach provides the "best" result overall (measured by the best result in column 4 on Table 2).
- **Consistency**: Which approach provides the "best" mean and standard deviation values (measured by the mean and standard deviation (Std. Dev.) results in columns 5 and 6, respectively on Table 2).
- **Diversity**: To analyze the rate of feasible solutions of each approach during a single run.

		Statistical Results on 9 test problems.				
P.	**Approach**	**Optimal**	**Best**	**Mean**	**St. Dev.**	F_p
P_1	COMOGA	−30665.539	−30533.057	−30329.563	7.48E+1	0.24%
	CHVEGA		−30647.246	−30628.469	**7.88E+0**	41%
	CHNPGA		**-30661.033**	**-30630.883**	2.05E+1	35%
	CHMOGA		−30649.959	−30568.918	5.35E+1	3.5%
P_2	COMOGA	−6961.814	−6808.696	−5255.105	9.95E+2	0.20%
	CHVEGA		**−6942.747**	**−6762.048**	**1.02E+2**	4.3%
	CHNPGA		−6941.307	−6644.539	3.36E+2	2%
	CHMOGA		−6939.440	−6678.926	1.57E+2	2%
P_3	COMOGA*(8)	24.306	485.579	1567.294	9.24E+2	0.03%
	CHVEGA		28.492	34.558	2.93E+0	15%
	CHNPGA		**26.986**	**31.249**	**2.32E+0**	4.9%
	CHMOGA		29.578	45.589	1.52E+1	1.3%
P_4	COMOGA	680.63	733.00	983.63	1.16E+2	1.1%
	CHVEGA		693.64	739.31	2.51E+1	4.5%
	CHNPGA		**680.95**	**682.34**	**8.36E-1**	24%
	CHMOGA		681.71	692.97	1.09E+1	4.9%
P_5	COMOGA1*(71)	7049.25	10865.43	18924.58	3.85E+3	0.0001%
	CHVEGA*(63)		9842.45	17605.59	3.87E+3	0.005%
	CHNPGA*(29)		8183.30	13716.70	4.80E+3	0.05%
	CHMOGA		**7578.34**	**9504.36**	**1.50E+3**	2%
P_6	COMOGA	0.75	**0.75**	**0.75**	**4.95E-4**	0.029%
	CHVEGA		**0.75**	0.80	2.58E-2	1.1%
	CHNPGA		**0.75**	**0.75**	1.21E-2	2.6%
	CHMOGA		**0.75**	**0.75**	5.95E-4	1.7%
P_7	COMOGA	2.381	2.471158	2.726058	1.20E-1	0.03%
	CHVEGA		2.386833	**2.393504**	**3.80E-3**	35%
	CHNPGA		**2.382860**	2.420906	2.56E-2	20%
	CHMOGA		2.386333	2.504377	9.90E-2	5%
P_8	COMOGA	6059.946	6369.428	7795.412	7.01E+2	0.4%
	CHVEGA		6064.724	6259.964	1.70E+2	43%
	CHNPGA		**6059.926**	**6172.527**	**1.24E+2**	33%
	CHMOGA		6066.967	6629.064	3.85E+2	45%
P_9	COMOGA	0.012681	0.012929	0.014362	8.64E-4	2.11%
	CHVEGA		0.012688	0.012886	2.09E-4	25%
	CHNPGA		0.012683	**0.012752**	**6.20E-5**	10%
	CHMOGA		**0.012680**	0.012960	3.63E-4	4.8%

Table 2. Experimental results using the four approaches with the 9 test problems. The symbol "*" and the number between parenthesis "(n)"mean that only in n runs feasible solutions were found , F_p is the average percentage of feasible solutions found during a single run (with respect to the full population).

Quality of the results

CHNPGA provided the "best" best results in five problems (P1, P3, P4, P7 and P8) and slightly improved the best known solution in one of them (P8). CHVEGA obtained the "best"best result in problem P2 and CHMOGA in problems P5 and P9. All the four approaches reached the best solution in problem P6.

Consistency

CHNPGA provided the most consistent results in four problems (P3, P4, P8 and P9). In problem P1, CHNPGA showed a mean value closer to the optimal solution than that provided by CHVEGA; however, CHVEGA's standard deviation value was smaller than CHNPGA's. We consider that the behavior of CHNPGA was more consistent because of its mean closeness value to the optimal result. CHVEGA presented the "best" mean and standard deviation values in two functions (P2 and P7). Finally, CHMOGA presented the best consistency in problem P5. It is important to note that, for function P5, only CHMOGA consistently found feasible solutions in each single run. The remaining techniques had problems to reach the feasible region in this problem. Again, problem P6 was easily solved with a similar performance by all four approaches.

Diversity

It is quite interesting to analyze the average of feasible solutions that each algorithm maintains in a single run. The fact that the population size is the same for all four algorithms compared gives a more fair point of comparison. For most of the problems, the approach which consistently reached the vicinity of the optimum, was able to handle a rate of feasible solutions above the average rate of the four approaches: CHNPGA in P1, P3 P4, P8 and P9, CHVEGA in P2, and P7, and CHMOGA in P5. As expected, this rate corresponds to the approximate size of the feasible region with respect to the whole search space (reported as ρ, fourth column in Table 1), i.e., high rates on problems with larger feasible regions and low rates on problems with very small feasible regions. See for example test function P1 (27% of the search space is feasible and a 35% rate is maintained by CHNPGA) and P2 (0.0057% of the search space is feasible and a 4.3% rate is maintained by CHVEGA).

Summary of Results

Based on the observations made on each aspect of our small set of experiments and analysis, we now summarize our main findings:

- The Pareto dominance tournament selection promoted by CHNPGA provided the most accurate and consistent results for the set of test problems used in the experiments (P1, P3, P4, P7, P8 and P9), regardless the features of the problem to be solved (type of objective function and constraints, dimensionality, size of the feasible region with respect to the whole search space).
- The population-based mechanism used by CHVEGA was very effective in problems with a low dimensionality, small feasible regions and nonlinear objective function (P2 and P7).
- The Pareto ranking approach based on feasibility used by CHMOGA was very competitive in problems with average dimensionality, linear or quadratic objective function and a very small feasible regions (P5 and P9). In fact, CHMOGA was the only approach that consistently found feasible solutions in problem P5.

- These three multiobjective-based constraint-handling mechanisms (CHNPGA, CHVEGA and CHMOGA) were able to maintain an appropriate rate of feasible solutions (with respect to the size of the feasible region of the problem) as to reach the neighborhood of the optimum.
- COMOGA was competitive only in problem P6, where all approaches were also very competitive. This can be explained by the fact that COMOGA was explicitly designed to solve a specific type of problem rather that being a general constraint-handling technique.

These findings are far from being conclusive, but provide some clues about the behavior of these types of constraint-handling mechanisms.

7 Conclusions

We have presented in this chapter a survey of constraint-handling techniques based on multiobjective optimization concepts. A taxonomy of techniques based on the type of transformation made from the NLP to either a bi-objective (objective function and sum of constraint violation) or a MOP (with the objective function and each constraint considered as separate objectives) has been proposed. We have presented a discussion about the main features of each method (selection criteria, diversity handling mechanism, genetic operators, advantages and disadvantages, and validation). Furthermore, some interesting findings about all methods have been summarized and briefly discussed.

In the final part of the chapter, we included a small comparative experiment of four representative approaches. The aim of this study was to provide some basic guidelines of their use to those interested in adopting these techniques. In this study, emphasis was placed on relating each type of constraint-handling scheme to the type of problem being solved.

Based precisely on these preliminary results, we foresee several potential paths for future research in this area: (1) more intensive use of explicit local search mechanisms coupled to constraint-handling techniques, (2) in-depth studies of the influence of the genetic operators used in these types of methods, (3) novel and more effective proposals of diversity maintenance mechanisms, (4) the combination of multiobjective concepts (Pareto methods with population-based techniques) into one single constraint-handling approach.

Acknowledgments

The first author acknowledges support from CONACyT through project number 52048-Y. The second author acknowledges support from CONACyT through project number 42435-Y.

Appendix A

The details of each test function used in our experiments are presented below:

1. **P1**:
Minimize: $f(\mathbf{X}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$

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

2. **P2**
Minimize: $f(\mathbf{X}) = (x_1 - 10)^3 + (x_2 - 20)^3$

subject to:
$g_1(\mathbf{X}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$
$g_2(\mathbf{X}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$

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

3. **P3**
Minimize: $f(\mathbf{X}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$

subject to:
$g_1(\mathbf{X}) = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0$
$g_2(\mathbf{X}) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0$
$g_3(\mathbf{X}) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0$
$g_4(\mathbf{X}) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0$
$g_5(\mathbf{X}) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0$

$g_6(\mathbf{X}) = x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0$
$g_7(\mathbf{X}) = 0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30 \leq 0$
$g_8(\mathbf{X}) = -3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10} \leq 0$

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

4. **P4**
Minimize: $f(\mathbf{X}) = (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 + 3(x_4-11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$

subject to:
$g_1(\mathbf{X}) = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0$
$g_2(\mathbf{X}) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0$
$g_3(\mathbf{X}) = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0$
$g_4(\mathbf{X}) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0$

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

5. **P5**
Minimize: $f(\mathbf{X}) = x_1 + x_2 + x_3$

subject to:
$g_1(\mathbf{X}) = -1 + 0.0025(x_4 + x_6) \leq 0$
$g_2(\mathbf{X}) = -1 + 0.0025(x_5 + x_7 - x_4) \leq 0$
$g_3(\mathbf{X}) = -1 + 0.01(x_8 - x_5) \leq 0$
$g_4(\mathbf{X}) = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \leq 0$
$g_5(\mathbf{X}) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0$
$g_6(\mathbf{X}) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0$
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: $\mathbf{X}^* = (579.19, 1360.13, 5109.92, 182.0174, 295.5985, 217.9799, 286.40, 395.5979)$, where $f(\mathbf{X}^*) = 7049.248$. g_1, g_2 and g_3 are active.

6. **P6**
Minimize: $f(\mathbf{X}) = x_1^2 + (x_2-1)^2$

subject to:
$h(\mathbf{X}) = x_2 - x_1^2 = 0$

where: $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$. The optimum solution is $\mathbf{X}^* =$

$(\pm 1/\sqrt{2}, 1/2)$ where $f(\mathbf{X}^*) = 0.75$.

7. **P7: Design of a welded beam** A welded beam is designed for minimum cost subject to constraints on shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c), end deflection of the beam (δ), and side constraints. There are four design variables as shown in Figure 1: h (x_1), l (x_2), t (x_3) and b (x_4). The problem can be stated as follows:
Minimize: $f(\mathbf{X}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$

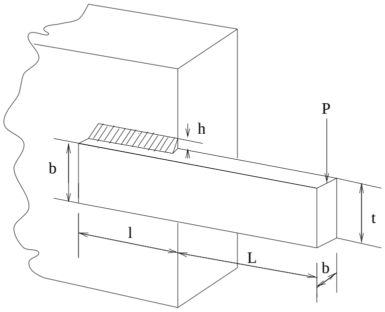

Fig. 1. Welded beam.

Subject to:
$g_1(\mathbf{X}) = \tau(\mathbf{X}) - \tau_{max} \leq 0$
$g_2(\mathbf{X}) = \sigma(\mathbf{X}) - \sigma_{max} \leq 0$
$g_3(\mathbf{X}) = x_1 - x_4 \leq 0$
$g_4(\mathbf{X}) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0$
$g_5(\mathbf{X}) = 0.125 - x_1 \leq 0$
$g_6(\mathbf{X}) = \delta(\mathbf{X}) - \delta_{max} \leq 0$
$g_7(\mathbf{X}) = P - P_c(\mathbf{X}) \leq 0$

where $\tau(\mathbf{X}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$ $\quad \tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J}$,
$M = P\left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2}$
$J = 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\} \quad \sigma(\mathbf{X}) = \frac{6PL}{x_4x_3^2}, \delta(\mathbf{X}) = \frac{4PL^3}{Ex_3^3x_4}$
$P_c(\mathbf{X}) = \frac{4.013\sqrt{\frac{EGx_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$
$P = 6000\ lb, \quad L = 14\ in, \quad E = 30 \times 10^6\ psi, \quad G = 12 \times 10^6\ psi$
$\tau_{max} = 13{,}600\ psi, \quad \sigma_{max} = 30{,}000\ psi, \quad \delta_{max} = 0.25\ in$ where $0.1 \leq x_1 \leq 2.0$, $0.1 \leq x_2 \leq 10.0$, $0.1 \leq x_3 \leq 10.0$ y $0.1 \leq x_4 \leq 2.0$.

8. **P8: Design of a pressure vessel**
   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.

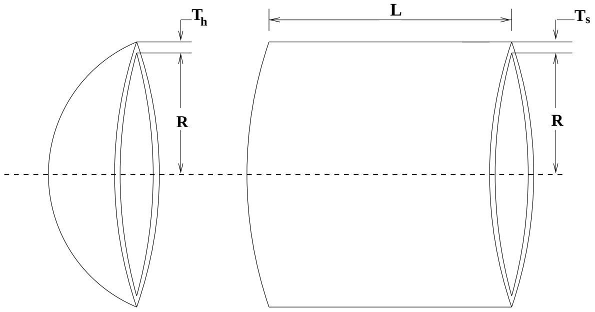

Fig. 2. Pressure vessel.

The problem can be stated as follows:
Minimize : $f(\mathbf{X}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

Subject to :
$g_1(\mathbf{X}) = -x_1 + 0.0193x_3 \leq 0$
$g_2(\mathbf{X}) = -x_2 + 0.00954x_3 \leq 0$
$g_3(\mathbf{X}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0$
$g_4(\mathbf{X}) = x_4 - 240 \leq 0$

where $1 \leq x_1 \leq 99$, $1 \leq x_2 \leq 99$, $10 \leq x_3 \leq 200$ y $10 \leq x_4 \leq 200$.

9. **P9: Minimization of the weight of a tension/compression spring**
   This problem consists of minimizing the weight of a tension/compression spring (see Figure 3) subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables. The design variables are: the mean coil diameter D (x_2), the wire diameter d (x_1) and the number of active coils N (x_3). Formally, the problem can be expressed as:

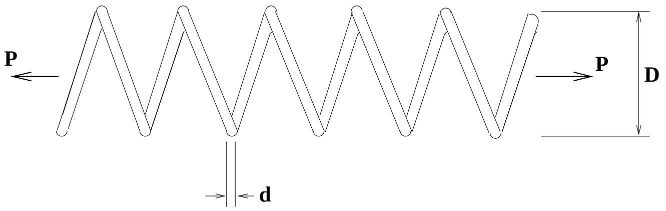

Fig. 3. Tension-compression spring.

Minimize: $(N+2)Dd^2$

Subject to:
$g_1(\mathbf{X}) = 1 - \frac{D^3N}{71785d^4} \leq 0$
$g_2(\mathbf{X}) = \frac{4D^2-dD}{12566(Dd^3-d^4)} + \frac{1}{5108d^2} - 1 \leq 0$
$g_3(\mathbf{X}) = 1 - \frac{140.45d}{D^2N} \leq 0$
$g_4(\mathbf{X}) = \frac{D+d}{1.5} - 1 \leq 0$

where $0.05 \leq x_1 \leq 2$, $0.25 \leq x_2 \leq 1.3$ y $2 \leq x_3 \leq 15$.

References

1. Shamim Akhtar, Kang Tai, and Tapabrata Ray. A Socio-Behavioural Simulation Model for Engineering Design Optimization. *Engineering Optimization*, 34(4):341–354, 2002.
2. Anders Angantyr, Johan Andersson, and Jan-Olov Aidanpaa. Constrained Optimization based on a Multiobjective Evolutionary Algorithms. In *Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003)*, volume 3, pages 1560–1567, Piscataway, New Jersey, December 2003. Canberra, Australia, IEEE Service Center.
3. 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.
4. C. W. Carroll. The created response surface technique for optimizing nonlinear restrained systems. *Operations Research*, 9:169–184, 1961.
5. Carlos A. Coello Coello. Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering and Environmental Systems*, 17:319–346, 2000.
6. Carlos A. Coello Coello. Treating Constraints as Objectives for Single-Objective Evolutionary Optimization. *Engineering Optimization*, 32(3):275–308, 2000.
7. 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.

8. 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.
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 in Design and Manufacture (ACDM'2002)*, volume 5, pages 273–284, University of Exeter, Devon, UK, April 2002. Springer-Verlag.
10. Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, June 2002.
11. R. Courant. Variational Methods for the Solution of Problems of Equilibrium and Vibrations. *Bulletin of the American Mathematical Society*, 49:1–23, 1943.
12. Carlos Manuel Mira de Fonseca. *Multiobjective Genetic Algorithms with Applications to Control Engineering Problems*. PhD thesis, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK, September 1995.
13. Hugo de Garis. Genetic Programming: Building Artificial Nervous Systems using Genetically Programmed Neural Networks Models. In R. Porter and B. Mooney, editors, *Proceedings of the 7th International Conference on Machine Learning*, pages 132–139. Morgan Kaufmann Publishers, 1990.
14. Kalyanmoy Deb. Genetic algorithms in multimodal function optimization. Master's thesis, University of Alabama, Alabama, USA, 1989.
15. Kalyanmoy Deb. An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2/4):311–338, 2000.
16. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
17. A. V. Fiacco and G. P. McCormick. Extensions of SUMT for nonlinear programming: equality constraints and extrapolation. *Management Science*, 12(11):816–828, 1968.
18. Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generali zation. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. Morgan Kauffman Publishers.
19. Carlos M. Fonseca and Peter J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 28(1):26–37, 1998.
20. David Goldberg. *The Design of Innovation*. Kluwer Academic Publishers, New York, June 2002. ISBN 1-4020-7098-5.
21. Arturo Hernández-Aguirre, Salvador Botello-Rionda, Carlos A. Coello Coello, Giovanni Lizárraga-Lizárraga, and Efrén Mezura-Montes. Handling Constraints using Multiobjective Optimization Concepts. *International Journal for Numerical Methods in Engineering*, 59(15):1989–2017, April 2004.

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 *Primer Congreso Español de Algoritmos Evolutivos y Bioinspirados (AEB'02)*, pages 460–465. Mérida España, 2002.
25. Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.
26. Saku Kukkonen and Jouni Lampinen. Constrained Real-Parameter Optimization with Generalized Differential Evolution. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 911–918, Vancouver, Canada, July 2006. IEEE Press.
27. J.J. Liang and P.N. Suganthan. Dynamic Multi-Swarm Particle Swarm Optimizer with a Novel Constraint-Handling Mechanism. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 316–323, Vancouver, Canada, July 2006. IEEE Press.
28. Efrén Mezura-Montes and Carloa A. Coello Coello. Adding a Diversity Mechanism to a Simple Evolution Strategy to Solve Constrained Optimization Problems. In *Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003)*, volume 1, pages 6–13, Piscataway, New Jersey, December 2003. Canberra, Australia, IEEE Service Center.
29. Efrén Mezura-Montes and Carlos A. Coello Coello. Multiobjective-Based Concepts to Handle Constraints in Evolutionary Algorithms. In Edgar Chávez, Jesús Favela, Marcelo Mejía, and Alberto Oliart, editors, *Proceedings of the Fourth Mexican International Conference on Computer Science (ENC'2003)*, pages 192–199, Los Alamitos, CA, September 2003. Apizaco, Tlaxcala, México, IEEE Computer Society.
30. Efrén Mezura-Montes and Carlos A. Coello Coello. A Simple Multimembered Evolution Strategy to Solve Constrained Optimization Problems. *IEEE Transactions on Evolutionary Computation*, 9(1):1–17, February 2005.
31. Efrén Mezura-Montes, Carlos A. Coello Coello, and Ricardo Landa-Becerra. Engineering Optimization Using a Simple Evolutionary Algorithm. In *Proceedings of the Fiftheenth International Conference on Tools with Artificial Intelligence (ICTAI'2003)*, pages 149–156, Los Alamitos, CA, November 2003. Sacramento, California, IEEE Computer Society.
32. Zbigniew Michalewicz, Kalyanmoy Deb, Martin Schmidt, and Thomas Stidsen. Test-Case Generator for Nonlinear Continuous Parameter Optimization Techniques. *IEEE Transactions on Evolutionary Computation*, 4(3):197–215, September 2000.
33. Zbigniew Michalewicz and David B. Fogel. *How to Solve It: Modern Heuristics*. Springer, Germany, 2nd edition, 2004.

34. Zbigniew Michalewicz and Cezary Z. Janikow. Handling Constraints in Genetic Algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA-91)*, pages 151–157, San Mateo, California, 1991. University of California, San Diego, Morgan Kaufmann Publishers.
35. Zbigniew Michalewicz and G. Nazhiyath. Genocop III: A co-evolutionary algorithm for numerical optimization with nonlinear constraints. In David B. Fogel, editor, *Proceedings of the Second IEEE International Conference on Evolutionary Computation*, pages 647–651, Piscataway, New Jersey, 1995. IEEE Press.
36. Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.
37. D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas. Crossover Operator Effect in Functions Optimization with Constraints. In J. J. Merelo-Guervós and et al., editors, *Proceedings of the 7th Parallel Problem Solving from Nature (PPSN VII)*, pages 184–193, Heidelberg, Germany, September 2002. Granada, Spain, Springer-Verlag. Lecture Notes in Computer Science Vol. 2439.
38. Domingo Ortiz-Boyer, Rafael Del-Castillo-Gomariz, Nicolas Garcia-Pedrajas, and Cesar Hervas-Martinez. Crossover effect over penalty methods in function optimization with constraints. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 2, pages 1127–1134, Edinburgh, Scotland, September 2005. IEEE Service Center.
39. Akira Oyama, Koji Shimoyama, and Kozo Fujii. New Constraint-Handling Method for Multi-Objective Multi-Constraint Evolutionary Optimization and Its Application to Space Plane Design. In R. Schilling, W. Haase, J. Periaux, H. Baier, and G. Bugeda, editors, *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems (EUROGEN 2005)*, Munich, Germany, 2005.
40. 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.
41. David Powell and Michael M. Skolnick. Using genetic algorithms in engineering design optimization with non-linear constraints. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)*, pages 424–431, San Mateo, California, July 1993. University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers.
42. 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.
43. Tapabrata Ray and K.M. Liew. Society and Civilization: An Optimization Algorithm Based on the Simulation of Social Behavior. *IEEE Transactions on Evolutionary Computation*, 7(4):386–396, August 2003.
44. Tapabrata Ray and Kang Tai. An Evolutionary Algorithm with a Multilevel Pairing Strategy for Single and Multiobjective Optimization. *Foundations of Computing and Decision Sciences*, 26:75–98, 2001.
45. Colin B. Reeves. *Modern Heuristic Techniques for Combinatorial Problems.* John Wiley & Sons, Great Britain, 1993.

46. Thomas Philip Runarsson and Xin Yao. Search biases in constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics Part C—Applications and Reviews*, 35(2):233–243, May 2005.
47. 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.
48. Marc Schoenauer and Zbigniew Michalewicz. Evolutionary Computation at the Edge of Feasibility. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the Fourth Conference on Parallel Problem Solving from Nature (PPSN IV)*, pages 245–254, Heidelberg, Germany, September 1996. Berlin, Germany, Springer-Verlag.
49. Marc Schoenauer and Spyros Xanthakis. Constrained GA Optimization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)*, pages 573–580, San Mateo, California, July 1993. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.
50. 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.
51. Patrick D. Surry and Nicholas J. Radcliffe. The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms. *Control and Cybernetics*, 26(3):391–412, 1997.
52. Sangameswar Venkatraman and Gary G. Yen. A Generic Framework for Constrained Optimization Using Genetic Algorithms. *IEEE Transactions on Evolutionary Computation*, 9(4), August 2005.
53. Yuping Wang, Dalian Liu, and Yiu-Ming Cheung. Preference bi-objective evolutionary algorithm for constrained optimization. In Yue Hao et al., editor, *Computational Intelligence and Security. International Conference, CIS 2005*, volume 3801, pages 184–191, Xi'an, China, December 2005. Springer-Verlag. Lecture Notes in Artificial Intelligence.
54. Wang Yong and Cai Zixing. A Constrained Optimization Evolutionary Algorithm Based on Multiobjective Optimization Techniques. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 2, pages 1081–1087, Edinburgh, Scotland, September 2005. IEEE Service Center.
55. Yuren Zhou, Yuanxiang Li, Jun He, and Lishan Kang. Multi-objective and MGG Evolutionary Algorithm for Constrained Optimization. In *Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003)*, volume 1, pages 1–5, Piscataway, New Jersey, December 2003. Canberra, Australia, IEEE Service Center.
56. E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.