Swarm and Evolutionary Computation 1 (2011) 173–194

Contents lists available at SciVerse ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Survey paper

Constraint-handling in nature-inspired numerical optimization: Past, present and future

Efrén Mezura-Montes [a,*], Carlos A. Coello Coello [b]

[a] Laboratorio Nacional de Informática Avanzada (LANIA A.C.), Rébsamen 80, Centro, Xalapa, Veracruz, 91000, Mexico

[b] Centro de Investigación y de Estudios Avanzados del IPN (CINVESTAV-IPN), Departamento de Computación (Evolutionary Computation Group), Av. IPN 2508, Col. San Pedro Zacatenco, México D.F, 07360, Mexico

ARTICLE INFO

Article history:
Received 28 July 2011
Received in revised form
22 October 2011
Accepted 24 October 2011
Available online 3 November 2011

Keywords:
Constraint-handling
Differential evolution
Evolution strategies
Particle swarm optimization
Genetic algorithms
Evolutionary programming

ABSTRACT

In their original versions, nature-inspired search algorithms such as evolutionary algorithms and those based on swarm intelligence, lack a mechanism to deal with the constraints of a numerical optimization problem. Nowadays, however, there exists a considerable amount of research devoted to design techniques for handling constraints within a nature-inspired algorithm. This paper presents an analysis of the most relevant types of constraint-handling techniques that have been adopted with nature-inspired algorithms. From them, the most popular approaches are analyzed in more detail. For each of them, some representative instantiations are further discussed. In the last part of the paper, some of the future trends in the area, which have been only scarcely explored, are briefly discussed and then the conclusions of this paper are presented.

1. Introduction

In spite of the complexity of several real-world optimization problems, their solution by nature-inspired meta-heuristics such as evolutionary algorithms (EAs) [1] and swarm intelligence algorithms (SIAs) [2] is common nowadays. In this paper, these two types of approaches will be generically called *nature-inspired algorithms* (NIAs).

Michalewicz in [3] describes some sources of difficulty found in real-world optimization problems. Besides huge search spaces, noise in the objective function(s) and the complexity of the modeling process, the presence of constraints was pointed out. Constraints may cause the search to keep away the focus on optimization to just seeking a feasible (i.e., valid) solution. On the other hand, in their original versions, NIAs were designed to deal with unconstrained search spaces [4,5]. This was the main motivation to add constraint-handling techniques to NIAs aiming to guide the search to those regions with feasible solutions.

The problem of interest in this paper is the constrained numerical optimization problem (CNOP), which, without loss of generality, can be defined as to:

Find $\vec{x}$ which minimizes

$$f(\vec{x}) \tag{1}$$

subject to

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

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

where $\vec{x} \in \mathbb{R}^n$ is the vector of solutions $\vec{x} = [x_1, x_2, \ldots, x_n]^T$, m is the number of inequality constraints, and p is the number of equality constraints. Each x_k, $k = 1, \ldots, n$ is bounded by lower and upper limits $L_k \leq x_k \leq U_k$ which define the search space $\mathcal{S}$. $\mathcal{F}$ comprises the set of all solutions which satisfy the constraints of the problems and it is called the feasible region. Both, the objective function and the constraints can be linear or nonlinear. To handle equality constraints in NIAs, they are usually transformed into inequality constraints as follows [6]: $|h_j(\vec{x})| - \varepsilon \leq 0$, where ε is the tolerance allowed (a very small value).

In the specialized literature there are comprehensive surveys and edited books which have registered, at different times, the previous and undergoing research on constrained optimization[1] [7–9].

* Corresponding author. Tel.: +52 228 841 6100; fax: +52 228 841 6101.
E-mail addresses: emezura@lania.mx (E. Mezura-Montes), ccoello@cs.cinvestav.mx (C.A. Coello Coello).

[1] The second author maintains the repository on constraint-handling techniques for evolutionary algorithms, which is available at: http://www.cs.cinvestav.mx/~constraint/.

2210-6502/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.swevo.2011.10.001

There are also reviews on specific EAs such as evolution strategies (ES) [10] and particle swarm optimization (PSO) [11] designed to solve CNOPs. In fact, there are reviews focused only on repair methods [12], which are associated with constraint-handling techniques (although repairing solutions is more common in combinatorial optimization than in a numerical optimization [12]).

This document aims to update the review of the state-of-the-art with the most significant contributions on constraint-handling techniques in the recent years, and also with a discussion of representative NIA-based approaches adopting such techniques to solve CNOPs (i.e., the present). This analysis is preceded by a classification of the most representative approaches included in previous reviews of the specialized literature [7,8] (i.e., the past). Based on this literature review, a prospective view with some future trends, which have been only scarcely tackled until now, is presented (i.e., the future). Due to the considerably high number of publications available on this topic, this review is not (and does not attempt to be) exhaustive. However, it presents the current flavor and tendencies in the resolution of CNOPs using NIAs.

The paper is organized as follows: Section 2 revisits the constraint-handling techniques proposed in the early days of this area. In Section 3 a number of novel proposals which have had high-impact in the area are summarized and discussed. After that, Section 4 includes a prospective view of the future paths of research on NIAs to solve CNOPs. Finally, Section 5 draws some conclusions.

2. The early years

In the previous reviews on nature-inspired algorithms for searching in constrained spaces, two classifications were proposed: one by Michalewicz and Schoenauer [7] and another one by Coello Coello [8]. Both taxonomies agreed on penalty functions and hybrid methods as two separated classes. Whereas the first one is kept in this paper (based on the permanent interest that remains on this type of constraint-handling technique), the second class is not considered here because it mainly groups combinations of NIAs with other types search algorithms (something which is beyond the scope of this paper). This new classification for earlier methods is based on constraint-handling mechanisms, whereas the search algorithm employed is discussed as a separate issue. The class of "methods based on preserving feasibility of solutions" by Michalewicz and Schoenauer was merged with Coello Coello's "special representation and operators" and "repair algorithms" into one single class simply called "special operators" (repair methods are seen in this paper as special operators designed to move an infeasible solution to the feasible region of the search space). The "separation of objective function and constraints" class by Coello Coello is kept in this work and it includes those approaches classified by Michalewicz and Schoenauer as "methods which make a clear distinction between feasible and infeasible solutions". This type of techniques, such as penalty functions, are still very popular among researchers and practitioners. Finally, a particular category, called 'decoders" was added because it covers the type of constraint-handling technique whose results were the most competitive in the early years of the field. The simplified taxonomy from the methods included in both surveys [7,8] is the following:

1. Penalty functions
2. Decoders
3. Special Operators
4. Separation of objective function and constraints.

In the following subsections each of them is introduced and discussed.

2.1. Penalty functions

Based on mathematical programming approaches, where a CNOP is transformed into an unconstrained numerical optimization problem, NIAs have adopted penalty functions [13], whose general formula is the following:

$$\phi(\vec{x}) = f(\vec{x}) + p(\vec{x}) \tag{4}$$

where $\phi(\vec{x})$ is the expanded objective function to be optimized, and $p(\vec{x})$ is the penalty value that can be calculated as follows:

$$p(\vec{x}) = \sum_{i=1}^{m} r_i \cdot \max(0, g_i(\vec{x}))^2 + \sum_{j=1}^{p} c_j \cdot |h_j(\vec{x})| \tag{5}$$

where r_i and c_j are positive constants called "penalty factors".

As can be noted, the aim is to decrease the fitness of infeasible solutions in order to favor the selection of feasible solutions. In Eq. (4), the penalty value is added to the fitness of a solution because low values are preferred as expected in a minimization problem (as stated in Section 1). Unlike mathematical programming approaches, where interior and exterior penalty functions are employed, NIAs have mainly focused on the second type of approach, because the normal assumption is that the first generation of an NIA may contain only infeasible solutions.

Even though their implementation is quite simple, penalty functions require a careful fine-tuning of their penalty factors in order to determine the severity of the penalties to be applied, and these values are highly problem-dependent [6].

The most simple penalty function is known as "death-penalty". Under this scheme, infeasible solutions are assigned the worst possible fitness value or are simply eliminated from the optimization process [14,15].

Based on the fact that a "death-penalty" keeps the search from using valuable information from the infeasible solutions, there are penalty-based constraint-handling techniques which can deal with them, but also focus on defining appropriate penalty factors.

There are penalty functions whose penalty factor values (c_i and $c_j, i = 1, \ldots, m$ and $j = 1, \ldots, m$) remain fixed during all the process as those proposed by Kuri and Villegas-Quezada [16], Homaifar et al. [17], Hoffmeister and Sprave [18], and Le Riche et al. [19]. The main drawback of keeping fixed penalty factor values is the generalization of such type of approach, i.e., the values that may be suitable for one problem are normally unsuitable for another one.

The use of time (usually the generation counter in a NIA) as a value that affects the penalty factor has also been also explored, among others, by Joines and Houck [20] Kazarlis and Petridis [21], and Crossley and Williams [22]. Even the cooling factor of the simulated annealing algorithm has been employed to vary the penalty factors as in the approach proposed by Michalewicz and Attia [23], called GENOCOP II. Recalling from a previous comment in this section about the preference for exterior penalty functions in NIAs, in a dynamic penalty function, smaller values (i.e. soft penalties) are defined in the first steps of the optimization process so as to allow the algorithm to sample the search space and find its feasible region. In contrast, larger values (i.e., severe penalties) are adopted in the last part of the search in order to preserve the feasible solutions previously found and to speed up convergence towards the global optimum. The main disadvantage of dynamic penalty functions is that they require parameters for the dynamic tuning of the penalty factors. Furthermore, the schedule established for the penalty factors may be as hard to generalize as it is to generalize penalty factors in a static penalty function.

The information given by the same algorithm has been used to update the penalty factors in those so-called adaptive penalty functions. Hadj-Alouane and Bean [24] used the feasibility of the

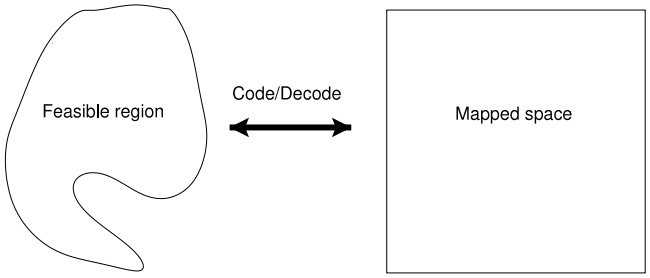

Fig. 1. Graphical view of a general decoder. The search performed by a NIA is carried out in the mapped space.

best solution in a number of generations as a criterion to modify the penalty factor. The fitness of the best feasible solution so far was used to modify the penalty values by Rasheed [25]. Hamda and Schoenauer [26] and later Hamida and Schoenauer [27] opted for a balance between feasible and infeasible solution in the current population as their criterion to modify the penalization values. Barbosa and Lemonge [28] proposed the use of the average of the objective function and the level of violation of each constraint during the evolution as update criteria for the penalty function. On the other hand, more elaborate ways to ask the algorithm to compute convenient penalty factors have been based on co-evolution, as in the approach proposed by Coello [29] and fuzzy logic, as proposed by Wu and Yu [30]. The main drawback of adaptive penalty functions is inherent to their design, because the values that are adapted are based on the current behavior of the algorithm, and it is unknown if the changes made will be indeed useful in later cycles of the optimization process.

Discussion

The earliest constraint-handling techniques based on penalty functions showed a lot of diversity in the way of defining the penalty factors (static, dynamic, adaptive, co-evolved, fuzzy-adapted, etc.). However, it was not clear, at that point of time, which of them could be more competitive. This question is answered in the next section of this paper. In fact, the previously mentioned approaches introduced, most of the time, additional parameters to define the penalty factors. The argument was that the calibration of such parameters was less complicated with respect to the careful fine-tuning required by a traditional penalty factor.

2.2. Decoders

Decoders were one of the most competitive constraint-handling techniques in the early years of this area. They are based on the idea of mapping the feasible region $\mathcal{F}$ of the search space $\mathcal{S}$ onto an easier-to-sample space where a NIA can provide a better performance [31] (see Fig. 1).

The mapping process of a decoder must guarantee that each feasible solution in the search space is included in the decoded space and that a decoded solution corresponds to a feasible solution in the search space. Moreover, the transformation process must be fast and it is highly desirable that small changes in the search space of the original problem cause small changes in the decoded space as well. Koziel and Michalewicz proposed the homomorphous maps (HM), where the feasible region is mapped into an n-dimensional cube [31,32]. HM was the most competitive constraint-handling approach during some years, before the advent of stochastic ranking and other modern approaches discussed in the next section. Kim and Husbands reported another (less known) decoder-based approach based on Riemann mappings [33–35].

Discussion

Although decoders are an interesting constraint-handling technique from a theoretical perspective, their actual implementation is far from trivial and involves a high computational cost which was normally not evaluated when assessing their performance (e.g., when assessing the performance of HM [32], only the actual fitness function evaluations performed by the evolutionary algorithm were considered). These shortcomings have made decoders such as HM a relatively rare approach nowadays.

2.3. Special operators

When solving a CNOP, a special operator is conceived as a way of either preserving the feasibility of a solution [36] or moving within a specific region of interest within the search space, i.e., the boundaries of the feasible region [37]. Michalewicz proposed GENOCOP [36], which only works with linear constraints and requires an initial feasible solution and also a pre-processing process to eliminate equality constraints and some variables of the problem. GENOCOP uses a variation operator which constructs linear combinations of feasible solutions to preserve their feasibility. Michalewicz and Nazhiyath proposed GENOCOP III [38], which uses GENOCOP in one of its two populations which interact each other. The other population stores feasible solutions. Special operators are designed to convert solutions from the first population, which only satisfy linear constraints, into fully feasible solutions. Kowalczyk [39] proposed a constraint-consistent GA where a feasible, or at least partially feasible, initial population is required. Moreover, in this approach, special operators assign values to the decision variables aiming to keep the feasibility of the solution. On the other hand, Schoenauer and Michalewicz in [37,40], after an ad-hoc initialization process, employed special operators for two specific problems to sample the boundaries of their feasible regions. Finally, the same authors in [41], presented an interesting transition from their special operators to a decoder.

Discussion

Some authors have reported highly competitive results when adopting special operators. The main drawback of this sort of approach is their limited applicability. Additionally, most of them require an ad-hoc initialization process or at least one feasible or partially-feasible solution in the initial population, and this may be quite difficult and/or computationally expensive when dealing with highly-constrained optimization problems.

2.4. Separation of objective function and constraints

Unlike the idea of combining the objective function and the values of the constraints into a single value as done in a penalty function (see Eq. (4)), there exist constraint-handling techniques which work with the opposite idea, i.e. keeping those two values apart in the NIA's selection process. Powell and Skolnick in [42] proposed an approach based on Eq. (6).

$$\text{fitness}(\vec{x}) = \begin{cases} f(\vec{x}) & \text{if feasible} \\ 1 + r\left(\sum_{i=1}^{m} g_i(\vec{x}) + \sum_{j=1}^{p} h_j(\vec{x})\right) & \text{otherwise} \end{cases} \tag{6}$$

where a feasible solution has always a better fitness value with respect to that of an infeasible solution, whose fitness is based only on their accumulated constraint violation.

The idea of dividing the search in two phases was explored by Hinterding and Michalewicz in [43]. The first phase aims to find feasible solutions, regardless of the objective function value. After an appropriate number of feasible solutions has been found, the second phase starts with the goal of optimizing the objective function.

The use of lexicographic ordering to satisfy constraints (one at a time) was studied by Schoenauer and Xanthakis in [44]. This approach, called *behavioral memory*, sequentially satisfies the

constraints of a problem. When a certain number of solutions in the population satisfy the first constraint, an attempt to satisfy the second one is made (but the first constraint must continue to be satisfied). All the solutions that violate any of the constraints that had been previously considered for satisfaction are eliminated from the population.

One of the most popular and effective constraint-handling techniques in current use belongs to this category and was originally proposed by Deb [45]. In this approach, a set of three feasibility criteria are added to a binary tournament selection in a GA as follows:

1. When comparing two feasible solutions, the one with the best objective function is chosen.
2. When comparing a feasible and an infeasible solution, the feasible one is chosen.
3. When comparing two infeasible solutions, the one with the lowest sum of constraint violation is chosen.

The sum of constraint violation can be calculated as follows:

$$\phi(\vec{x}) = \sum_{i=1}^{m} \max(0, g_i(\vec{x}))^2 + \sum_{j=1}^{p} |h_j(\vec{x})| \quad (7)$$

where the values of each inequality constraint $g_i(\vec{x}),\ i = 1, 2, \ldots, m$ and also each equality constraint $h_j(\vec{x}),\ j = 1, 2, \ldots, p$ are normalized.

The lack of user-defined parameters is one of its main advantages. However, it may also lead to premature convergence [46].

Based on the idea proposed by Coello [47], where in a multi-population scheme each sub-population tries to satisfy one constraint of a CNOP and another one optimizes the objective function, Liang and Suganthan proposed a dynamic assignment of sub-swarms to constraints [48] in PSO. Furthermore, sequential quadratic programming was used as a local search operator. The approach provided highly competitive results in a set of 24 test problems. The approach was further improved in [49], where only two sub-swarms, one of them with a tolerance for inequality constraints, were used. Each particle, and not a sub-swarm, was dynamically assigned an objective (either the original objective function or a constraint) in such a way that more difficult objectives to optimize (satisfy) were assigned more frequently. The approach was tested on eighteen scalable test problems and the results were highly competitive. However, several parameters values must be defined by the user. Li et al. [50] adopted a similar approach but using DE as a search algorithm. Three DE variants were used in the algorithm and were chosen at random. The results were better in the same set of scalable test problems where improved the PSO-based approach was tested.

Following the same scheme adopted by Hinterding and Michalewicz [43], Venkatraman and Yen [51] solved a CNOP in two-steps: The first one focused only on satisfying the constraints of the problem, i.e.,. a CNOP is transformed into a constraint-satisfaction problem. Once a feasible solution was generated, the second part of the approach started with the goal of optimizing the objective function value. The approach presented premature convergence in some test problems because the approach to the feasible region was biased by the minimization of the sum of constraint violation but ignoring the objective function value.

Liu et al. [52] proposed a separation scheme based on a co-evolutionary approach in which two populations are adopted. The first one optimized the objective function without considering the constraints, while the second population aimed to satisfy the constraints of the problem. Each population could migrate solution to the other. Finally, a Gaussian mutation was applied as a local search mechanism to the best individuals. However, this approach was tested on a small set of test problems and different parameter values were required for solving some of them.

Several authors have adopted multi-objective optimization concepts to solve constrained optimization problems. In this case, the objective functions and the constraints are also handled separately. These approaches are reviewed in [53] and can be divided in the following groups:

1. Techniques which transform a CNOP into a bi-objective problem (the original objective function and the sum of constraint violation as indicated in Eq. (7)).
2. Techniques which transform a CNOP into a multi-objective optimization problem (the original objective function and each constraint are handled, each, as a single objective function).

Multi-objective concepts such as Pareto dominance, Pareto ranking and population-based selection have been adopted by these constraint-handling approaches [53]. Such approaches have been relatively popular in the specialized literature in spite of their shortcomings (mainly related to the lack of bias provided by Pareto ranking when used in a straightforward manner [54], and the difficulties of these approaches to preserve diversity in the population [53]).

Additionally, other mechanisms such as Pareto ranking in different search spaces [55–58], the shrinking of the search space [59] and the use of non-dominated sorting and clustering techniques to generate collaboration among sub-populations [60], have been proposed to tackle the disadvantages of this type of constraint-handling technique.

Discussion

The separation of constraints and the objective function has been found to generate an important diversity loss. Therefore, it is important to design appropriate diversity maintenance mechanisms when designing such approaches. However, this has not prevented the use of these techniques which are among the most popular in the two of the most popular constraint-handling techniques discussed in the following section, are based on the idea of using the objective function and the constraints as different criteria in the selection or replacement mechanism within a NIA to solve CNOPs.

2.5. General comments

In a similar way of a typical search made by a NIA, the first attempts to generate constraint-handling techniques are similar to an exploration phase, in which a variety of approaches are proposed. Issues such as premature convergence by unsuitable bias (Section 2.4), need of a careful fine-tuning of parameters (Section 2.1), non-generalization (Section 2.3), high computational cost and difficult implementations (Section 2.2) are among the main limitations of early constraint-handling techniques. As will be seen in the following section, the second generation of constraint-handling techniques is characterized for having features that overcome most of the previously indicated limitations. Additionally, these modern approaches also share similarities which seems to suggest the possibility of adopting generic schemes in the near future. However, this may have also limited the variety of approaches possible, which is clearly reflected by a much lower number of publications on this topic in recent years.

3. Current constraint-handling techniques

This section presents a set of recent constraint-handling techniques which have had a relatively high impact in the area. Unlike the techniques presented in the previous section, the number of approaches reviewed in this case is lower. This is due to the fact that the differences among approaches are, in this case, more focused on modifications to the elements of the NIA

adopted, and not on the constraint-handling technique itself. From the list presented below the first three and the seventh approach are all new constraint-handling techniques, while the fourth and the fifth are updated versions of constraint-handling techniques previously discussed. The use of multi-objective concepts is now considered as a separate class due to its popularity in recent years. The approaches considered here are:

1. Feasibility rules
2. Stochastic ranking
3. ε-constrained method
4. Novel penalty functions
5. Novel special operators
6. Multi-objective concepts
7. Ensemble of constraint-handling techniques.

Next, we will provide a detailed discussion of each of them.

3.1. Feasibility rules

The three feasibility rules proposed for binary tournaments in [45][2] and detailed in Section 2.4 constitute an example of a constraint-handling technique that was proposed several years ago, but whose impact is still present in the literature. The popularity of this simple constraint-handling scheme lies on its ability to be coupled to a variety of algorithms, without introducing new parameters. Mezura-Montes and Coello Coello [62] emphasized the importance of combining these feasibility rules with other mechanisms (e.g., retaining infeasible solutions which are close to the feasible region) in order to produce a constraint-handling technique that is able to deal with problems having active constraints. However, such approach required a dynamic decreasing mechanism for the tolerance value (ε) for equality constraints.

Mezura-Montes et al. [46,63,64] later extended these feasibility rules to the selection process between target and trial vectors in differential evolution (DE). Although the resulting approaches were easy to implement, premature convergence was observed for some test problems and two additional modifications to the search algorithm (DE in this case) were required: (1) each target vector generated more than one trial vector (a user-defined parameter was added for this sake) [63] and (2) a new DE variant was designed [64]. Lampinen used a similar DE-based approach in [65]. However, the third criterion was based on Pareto dominance in constraints space instead of the sum of constraint violation. Lampinen's approach was later adopted by Kukkonen and Lampinen in their Generalized Differential Evolution (GDE) algorithm [66] which showed promising results in a set of 24 benchmark problems. However, GDE had difficulties when facing more than three constraints. This seems to be the same limitation faced when attempting to use Pareto dominance in problems having four or more objective functions (the so-called *many-objective optimization problems*) [67].

Feasibility rules have also been used for designing parameter control mechanisms in DE-based constrained numerical optimization [68]. However, if self-adaptation mechanisms are incorporated as well (as in [68]), the computational cost of the approach may considerably increase, since more iterations will be required to collect enough information about the search as to make these self-adaptation mechanisms work in a proper manner.

Zielinski and Laur [69] coupled DE with the feasibility rules in a greedy selection scheme between target and trial vectors. Their approach, which is indeed very simple to implement, presented some difficulties in high-dimensionality problems with equality constraints. They also analyzed, in a further work, different termination conditions (e.g., improvement-based criteria, movement-based criteria, distribution-based criteria) for their algorithm [70]. They determined that the last criterion from the list indicated before was the most competitive. Zielinski et al. [71] also used the feasibility rules in a study to adapt two DE parameters (F and CR) when solving CNOPs. They concluded that the adaptation mechanism was not as significant as expected in the performance of the algorithm. Furthermore, Zielinski et al. [72] studied the effect of the tolerance utilized in the equality constraints, where values between $\epsilon = 1 \times 10^{-7}$ and $\epsilon = 1 \times 10^{-15}$ allowed the algorithm, coupled with the feasibility rules, to reach competitive results.

The feasibility rules have also been adopted by DE-based approaches which use self-adaptive mechanisms to choose among their variants, such as SaDE [73]. In this approach, sequential quadratic programming (SQP) is applied during some iterations to a subset of solutions in the population. Although this approach is very competitive, it heavily relies on the use of SQP, which may limit its applicability.

Brest [74] used the feasibility rules in his self-adaptive approach called jDE-2, which also combines different DE variants into a single approach to solve CNOPs. A replacement mechanism to keep diversity in the population was implemented to eliminate those k worst vectors at every l generations with new randomly-generated vectors. This mechanism reflects the premature convergence that the feasibility rules may cause in some test problems despite providing competitive results in others mainly related with inequality constraints.

Landa and Coello [75] also adopted the feasibility rules in an approach in which a cultural DE-based mechanism was developed, with the aim of incorporating knowledge from the problem into the search process when solving CNOPs. This approach adopts a belief space with four types of knowledge generated and stored during the search. Such knowledge is used to speed up convergence. The approach was able to provide competitive results in a set of benchmark problems. However, there are two main shortcomings of the approach: (1) it requires the use of spatial data structures for knowledge handling and (2) it also needs several parameters which must be defined by the user. Furthermore, spacial data structures are not trivial to implement.

Menchaca-Mendez and Coello Coello [76] proposed a hybrid approach which combines DE and the Nelder–Mead method. The authors extended a variant of the Nelder–Mead method called *Low Dimensional Simplex Evolution* [77] and used it to solve CNOPs. The set of feasibility rules are used in this case to deal with the constraints of the problems in both, the DE algorithm and the m-simplex-operator. However, a fourth rule which considers a tie between the sum of the constraint values, is also incorporated. The objective function value is used in this case to break this tie. This approach also adds some concepts from stochastic ranking, which will be discussed later on. The approach was tested in some benchmark problems and the results obtained were highly competitive, while requiring a lower number of fitness function evaluations than state-of-the-art algorithms. However, the approach added a significant set of parameters which must be fine-tuned by the user, such as those related to the m-simplex-operator.

Barkat Ullah [78] reported the use of feasibility rules coupled with a mechanism to force infeasible individuals to move to the feasible region through the application of search space reduction and diversity checking mechanisms designed to avoid premature convergence. This approach, which adopts an evolutionary agent system as its search engine, requires several parameters to be defined by the user and it was not compared against state-of-the-art NIAs to solve CNOPs. However, in the testbed reported by the authors the results were competitive.

[2] Apparently, this sort of scheme was originally proposed in [61] using an evolution strategy as the search engine.

The feasibility rules have been a popular constraint-handling mechanism in PSO-based approaches, too. Zielinski and Laur [79] added feasibility rules into a local best PSO. The approach presented premature convergence in test problems with a high number of equality constraints due to the lack of a diversity maintenance mechanism. In a similar approach, but focused on mixed-variable optimization problems, Sun et al. [80,81] added feasibility rules to a global-best PSO. This approach was tested only in two engineering design problems.

He and Wang [82] used feasibility rules to select the global best (*gbest*) and for updating the personal best (*pbest*) of each particle in a PSO-based approach designed to solve CNOPs. They used simulated annealing (SA) as a local search operator and applied it to the gbest particle at each generation. The approach was tested on a small set of benchmark problems as well as on a set of engineering design problems. The usage of SA improved the PSO performance. However, the main shortcoming of the approach is that it requires several user-defined parameter for both the PSO and the SA algorithms.

Toscano-Pulido and Coello Coello [83] combined feasibility rules with a global-best PSO but required a mutation operator to avoid converging to local optimum solutions. This same problem (premature convergence) was tackled by using two mutation operators by Muñoz-Zavala et al. [84,85]. Additionally, they also tackled this problem employing different topologies in local-best PSO algorithms [86,87]. The evident need of a mutation operator showed that the feasibility rules combined with PSO may cause premature convergence [5].

Cagnina et al. [88] tackled the premature convergence of PSO combined with feasibility rules by using a global–local best PSO. However, the use of a dynamic mutation operator was also required. The results obtained by this approach showed evident signs of stagnation in some test problems. In a further version of this approach [89], a bi-population scheme and a "shake" operator were added. This approach showed a significant improvement in performance with respect to its previous version.

In [11], feasibility rules were used as a constraint-handling mechanism in an empirical study aimed to determine which PSO variant was the most competitive when solving CNOPs. The authors found that the version adopting a constriction factor performed better than the (popular) version that uses inertia weight. Furthermore, local-best was found to be better than global-best PSO.

The use of feasibility rules motivated the definition of the *relative feasibility degree* in [90], which is a measure of constraint violation in pairwise comparisons. The aim of this work was to compare pairs of solutions but with a feasibility value based only on the values of their constraints and on the ratio of the feasible region of a given constraint with respect to the entire feasible region of the search space. The approach was coupled to DE and tested in some benchmark problems. The convergence rate was better with respect to the original feasibility rules but the final results were not considerable better with respect to those obtained by other state-of-art algorithms.

Karaboga and Basturk [91] and Karaboga and Akay [92] changed a greedy selection based only on the objective function values by the use of feasibility rules with the aim of adapting an artificial bee colony algorithm (ABC) to solve CNOPs. The authors also modified the probability assignment for their roulette wheel selection employed to focus the search on the most promising solutions. The approach was tested on a well-known set of 13 test problems and the results obtained were comparable with those obtained by the homomorphous maps [32], stochastic ranking [6] and other approaches based on penalty functions. However, the approach modified one ABC operator adding a new parameter to be fine-tuned by the user.

Mezura-Montes and Cetina-Domíngez extended Karabogas' approach by using feasibility rules as a constraint-handling technique but with an special operator designed to locate solutions close to the best feasible solution [93]. This approach was tested on 13 test problems and the results that they obtained were shown to be better than those reported by Karaboga and Basturk in [91]. However, this approach added extra parameters related to the tolerance used to handle equality constraints. An improved version was proposed in [94], where two operators were improved and a direct-search local operator was added to the algorithm. The approach provided competitive results in a set of eighteen scalable test problems but its main disadvantage was the definition of the schedule to apply the local search method.

Mezura-Montes and Hernández-Ocaña [95] used feasibility rules with the Bacterial Foraging Optimization Algorithm [96], to solve CNOPs. The feasibility rules were used in the greedy selection mechanism within the chemotactic loop, which considers the generation of a new solution (swim) based on the random search direction (tumble). This approach, called *Modified Bacterial Foraging Optimization Algorithm* (MBFOA), considered a swarming mechanism that uses the best solution in the population as an attractor for the other solutions. The approach was used to solve engineering design problems.

Mezura-Montes et al. [97] used feasibility rules as a constraint-handling mechanism in an in-depth empirical study of the use of DE as an optimizer in constrained search spaces. A set of well-known test problems and performance measures were used to analyze the behavior of different DE variants and their sensitivity to two user-defined parameters. From such analysis, the simple combination of two of them (DE/rand/1/bin and DE/best/1/bin) called Differential Evolution Combined Variants (DECV) was proposed by the authors. This approach is able to switch from one variant to the other based on a certain percentage of feasible solutions present in the population. The results obtained in a set of 24 test problems were competitive with respect to state-of-the-art algorithms. However, the performance of this approach strongly depends on the percentage used to perform the switch from one variant to the other and this value is problem-dependent.

Elsayed et al. [98] proposed two multi-operator NIAs to solve CNOPs. A four sub-population scheme is handled by one of two options: (1) a static approach where each sub-population with a fixed size evolves by using a particular crossover and mutation operator and, at some periods of time, the sub-populations migrate the best solutions that they had found to another sub-population, and (2) an adaptive approach in which the size of each subpopulation varies based on the feasibility of the best solution in the population in two contiguous generations. This approach was tested in two versions: with a real-coded GA using four crossover–mutation combinations and also with DE adopting four DE mutation variants, all of them with binomial crossover. The latter version outperformed the former after being extensively tested in 60 benchmark problems. The approach requires the definition of some additional parameters related to the minimum size that a sub-population can have, as well as to the generational interval for migrating solutions among sub-populations.

Elsayed et al. [99] proposed a modified GA where a novel crossover operator called multi-parent crossover and also a randomized operator were added to a real-coded GA to solve CNOPs. The feasibility rules were adopted as the constraint-handling mechanism. The approach was tested on a set of eighteen recently proposed test problems in 10D and 30D showing very competitive results. However, some disadvantages were found in separable test problems with a high dimensionality. The approach provided better results with respect to other approaches based on DE and PSO.

Elsayed et al. [100] compared ten different GA variants to solve CNOPs by using, in all cases, the feasibility rules as the constraint-handling technique. The crossover operators employed were triangular crossover, Simulated binary crossover, parent-centric crossover, simplex crossover, and blend crossover. The mutation operators adopted were non-uniform mutation and polynomial crossover. Statistical tests were applied to the samples of runs to provide confidence on the performances showed. An interesting conclusion of the comparison was that no GA was clearly superior with respect to the others GAs compared. Nonetheless, non-uniform mutation and polynomial mutation provided competitive results in 10D and 30D test problems.

Hamza et al. [101] proposed a DE algorithm to solve CNOPs where the feasibility rules were used as the constraint-handling mechanism and the population was divided in feasible and infeasible vectors. A constraint-consensus operator was applied to infeasible vectors so as to become them feasible. Even the approach showed competitive results in a set of thirteen well-known test problems, the constraint-consensus operator requires gradient calculations, which were made by numerical methods.

In an interesting adaptation of the feasibility rules combined with the idea of focusing first on decreasing the sum of constraint violation [43,51], Tvrdík and Poláková [102] adapted DE to solve CNOPs. If feasible solutions were present in the population, in a single cycle of the algorithm two generations were carried out, the first one based only on the sum of constraint violation and the second one based on the feasibility rules with a simple modification on the third rule, where between two infeasible solutions the one with the lowest sum of constraint violation was preferred if it also had a better value of the objective function. The approach was tested on eighteen scalable test problems in 10D and 30D. Even some competitive results were obtained, premature convergence was generally observed in the approach because the isolated usage of the sum of constraint violation kept the algorithm for sampling, in a more convenient way, the feasible region of the search space.

The feasibility rules were used in a real-coded GA with simulated binary crossover and adaptive polynomial mutation by Saha et al. [103]. Furthermore, a special operator based on gradient information was employed to favor the generation of feasible solutions in presence of equality constraints. Even the results improved by the approach in such test problems, the parameter which mainly controls the special operator required a careful fine-tuning based on the difficulty of the test problem.

The feasibility rules were added by Tseng and Chen to the multiple trajectory search (MTS) algorithm to solve CNOPs [104]. Those rules worked as the criteria to choose solutions in three region searches which allowed MTS to generate new solutions. The approach was able to provide feasible solutions in most of eighteen scalable test problems. However, it presented premature convergence.

Wang et al. [105] implicitly used feasibility rules to rank the particles in a hybrid multi-swarm PSO (HMPSO). They took inspiration from two papers: (1) the way Liang and Suganthan [48] constructed sub-swarms to promote more exploration of the search space and (2) the way Muñoz-Zavala et al. [85] used a differential mutation operator to update the local-best particle. The results agree with those found by Mezura-Montes and Flores-Mendoza, in which local-best PSO performs better than global-best PSO when solving CNOPs. The main shortcoming of the approach relies in its implementation due to the mechanisms added to PSO.

HMPSO was improved by Lui et al. in [106], where the DE mutation operator was extended by using two other operators. The number of evaluations required by the improved approach, which was called PSO-DE, decreased with respect to those required by HMPSO in almost 50%. This approach was validated using some engineering design problems but was not further tested on benchmark problems with higher dimensionalities.

The use of feasibility rules has been particularly popular in approaches based on artificial immune systems (AISs) as described in [107]. The first attempts to solve CNOPs with an AIS were based on hybrid GA-AIS approaches, in which the constraint-handling technique was the main task performed by the AIS embedded within a GA [108–110]. The AIS was evolved with the aim of making an infeasible solution (the antibody) as similar as possible (at a binary string level) as a feasible solution used as a reference (the antigen). After increasing the number of feasible solutions in the population, the outer GA continued with the optimization process. The main advantage of this technique is its simplicity. In further AIS-based approaches [111,112] in which the clonal selection principle was adopted [113], feasibility rules were incorporated as a way to rank antibodies (i.e., solutions) based on their affinity (objective function values and sum of constraint violation). In another approach based on a T-cell model, in which three types of cells (solutions) are adopted [114], the replacement mechanism uses feasibility rules as the criteria to select the survivors for the next iteration.

Liu et al. [115] proposed the organizational evolutionary algorithm (OEA) to solve numerical optimization problems. When extending this approach to constrained problems, a static penalty function and feasibility rules are compared as constraint-handling techniques. As expected, the static penalty function required specific values for each test problem solved. Although the use of the static penalty function allowed OEA to provide slightly better results than the use of feasibility rules, such results were only comparable with respect to state-of-the-art algorithms used to solve CNOPs.

Sun and Garibaldi [116] proposed a memetic algorithm to solve CNOPs. In this approach, the search engine is an estimation of distribution algorithm (EDA) while the local search operator is based on SQP. Some knowledge, called history, is extracted from the application of the local search and is given to the EDA with the aim of improving its performance. This knowledge consists in the application of a variation operator which uses the location of the best solution found so far to influence the generation of new solutions. Feasibility rules are used as the constraint-handling mechanism in the selection process. In fact, a comparison against a version of this approach but using stochastic ranking as the mechanism to deal with the constraints showed that the feasibility rules were more suitable for this approach. The approach provided competitive results with respect to state-of-the-art algorithms. However, the local search adopted requires gradient information.

Ullah et al. [117,118] adopted feasibility rules in their agent-based memetic algorithm to solve CNOPs. This approach is similar to a GA, and adopts the SBX operator to generate offspring which are subjected to a learning process that lasts up to four life spans. This actually works as a mutation operator whose use is based on a proper selection being made by each individual (agent). The measures used to select an operator are based on the success of each of them to generate competitive offspring. The communication among agents in the population is restricted to the current population. This approach seems to be sensitive to the value of the parameter associated with the communication mechanism. The results obtained were comparable with previously proposed approaches.

Ma and Simon [119] proposed an improved version of the biogeography-based optimization (BBO) algorithm. The idea is to add a migration operator inspired on the blend crossover operator used in real-coded GAs. BBO is inspired on the study of distributions of species over time and space and it adopts two variation operators: migration (or emigration) and mutation. A habitat (solution) has a habitat suitability index, HSI (i.e., the

fitness function). High-HSI solutions have a higher probability to share their features with low-HSI solutions by emigrating features to other habitats. Low-HSI solutions accept a lot of new features from high-HSI solutions by immigration from other habitats. Feasibility rules are used in this approach as the constraint-handling mechanism. The approach was found to be competitive with respect to PSO-based approaches and one GA-based algorithm. However, no further comparisons against state-of-the-art were reported.

Ali and Kajee-Bagdadi [120] compared feasibility rules with respect to the superiority of feasible points proposed by Powell and Skolnick [42] in a DE-based approach, in which a modified version of the pattern search method was used as a local search operator. They also compared their approach with respect to another based on a GA and found the former to be more competitive. The proposed DE-based approach presented a comparable performance with respect to other DE-based algorithms.

Discussion

As should be evident from the previous review, the feasibility rules have been very popular in constrained optimization using NIAs. This is due to their simplicity and flexibility, which makes them very suitable to be coupled to any sort of selection mechanism relatively easily. Their main drawback is that they are prone to cause premature convergence. This is due to the fact that this sort of scheme strongly favors feasible solutions. Thus, if no further mechanisms are adopted to preserve diversity (particularly paying attention to the need to keep infeasible solutions in the population), this approach will significantly increase the selection pressure [62].

Some approaches use special operators [78,93] but their role can be considered as secondary, because feasibility rules provide the main bias during the search. The use of feasibility rules has favored the development of approaches with self-adaptive variation operator selection mechanisms in DE [73,74,98], PSO [88,89] and GAs [98–100].

There are several empirical studies involving different NIAs in which feasibility rules were adopted as the constraint-handling mechanism. There are also studies centered on parameter control of DE when solving CNOPs, either exclusively related to DE parameters [71] or also coupled to parameters required by diversity mechanisms [68]. Furthermore, there is empirical evidence of the importance of defining the tolerance adopted for equality constraints [72], and the definition of termination criteria [70] when using feasibility rules. Additionally, there are empirical studies on PSO variants to solve CNOPs [11,79,83] and also on Differential Evolution variants [121] in which feasibility rules have been adopted.

Feasibility rules have been the most preferred constraint-handling mechanism in recent NIAs such as those based on AISs [107–114], BBO [119], OEA [115], ABC [91–94], BFOA [95], cultural algorithms [75], and hybrid approaches such as DE-PSO [85,105,106] and SA-PSO [82]. Memetic approaches have also adopted feasibility rules as the way to deal with CNOPs in a number of proposals [76,94,101,116–118,120,122].

Finally, there have been recent proposals to improve the traditional feasibility rules [90].

3.2. Stochastic ranking

Stochastic ranking (SR) was originally proposed by Runarsson and Yao [6,54]. SR was designed to deal with the inherent shortcomings of a penalty function (over and under penalization due to unsuitable values for the penalty factors) [6]. In SR, instead of the definition of those factors, a user-defined parameter called P_f controls the criterion used for comparison of infeasible solutions: (1) based on their sum of constraint violation or (2) based only on their objective function value. SR uses a bubble-sort-like process to rank the solutions in the population as shown in Fig. 2.

```
Begin
  For i=1 to N
    For j=1 to P-1
      u=random(0,1)
      If (φ(I_j) = φ(I_{j+1}) = 0) or (u < P_f)
        If (f(I_j) > f(I_{j+1}))
          swap(I_j,I_{j+1})
        Else
          If (φ(I_j) > φ(I_{j+1}))
            swap(I_j,I_{j+1})
    End For
    If (not swap performed)
      break
  End For
End
```

Fig. 2. Stochastic ranking sort algorithm [6]. I is an individual of the population. $\phi(I_j)$ is the sum of constraint violation of individual I_j. $f(I_j)$ is the objective function value of individual I_j.

SR was originally proposed to work with an ES [6] in its replacement mechanism which indeed requires a ranking process. However, it has been used with other NIAs where the replacement mechanism is quite different as in the approach reported by Zhang et al. [123]. In this case, the authors used SR with a DE variant proposed by Mezura-Montes et al. [63], in which more than one trial vector is generated per each target vector. Moreover, the parameter P_f was manipulated by a dynamic parameter control mechanism in order to conveniently decrease it, aiming to favor diversity during the initial generations of the search (infeasible solutions close to the feasible region are maintained) whereas only feasible solutions are kept during the final part of the search. The approach was compared against state-of-the-art algorithms and the results obtained were very competitive while requiring a low number of fitness function evaluations. However, the main disadvantage of this approach is that it requires the definition of the number of trial vectors generated by each target vector.

There are other approaches which combine DE with SR such as the proposal of Liu et al. [124] in which the diversity promoted by SR is exploited in a traditional DE variant. In a further paper, Liu et al. [125] used the concept of directional information related to the choice of the most convenient search direction based on the DE mutation operator and an adaptive population partitioning scheme. This approach was incorporated into a DE-based algorithm with SR, in order to deal with the constraints of a problem. These two algorithms were assessed with a fairly limited set of test problems.

SR has also been combined with ant colony optimization (ACO). Leguizamón and Coello Coello in [126] added SR to an ACO version for dealing with CNOPs. A comparison against traditional penalty functions showed that SR provided better and more robust results in a set of well-known benchmark problems. However, the approach presented some difficulties to find feasible solutions for some test problems.

SR has been further developed by Runarsson and Yao [127] in one of the earliest approaches focused on using fitness approximation for constrained numerical optimization. In this approach k-nearest-neighbors (NN) regression is coupled to SR. The authors found that the simplest NN version (using $k = 1$) provided the most competitive performance in a set of benchmark problems.

In a further paper, Runarsson and Yao [54] improved their ES by adding a differential mutation similar to that used in DE. The authors concluded that a good constraint-handling mechanism needs to be coupled to an appropriate search engine.

Zhang et al. [128] improved SR by giving the best feasible solutions a higher probability of being chosen, in an attempt to determine the search direction in the differential mutation added in [54]. However, the approach performed better with larger populations and more generations, which may be impractical in the solution of real-world problems with a high computational cost.

Mallipeddi et al. [129] proposed a two-population evolutionary programming (EP) approach with an external memory to store solutions based on an Euclidean distance measure that aimed to promote diversity. SR was used as a constraint-handling technique and it was compared with respect to the use of feasibility rules. Although the importance of the diversity mechanism based on the external memory was evident in the solution of some CNOPs, there was no clear evidence of the superiority of SR over the use of feasibility rules in this algorithm.

The simplicity of SR has made it suitable for being used in different domains. Fan et al. [130] solved robust layout synthesis of MEMS components by a DE-SR algorithm in which the vectors are ranked with SR before the DE operators are applied. After that, the population is split into two sets based on SR. The first set contains the vectors with the highest ranks while the second set includes the remaining vectors. The base vector and the vector which determines the search direction are chosen at random from the first set. The other vector is chosen at random from the second set. The number of vectors in each set is determined by a user-defined parameter. Although this approach obtained competitive results in the aforementioned problem, its use has not been extended to other problems to the authors' best knowledge. Huan-Tong et al. [131] used SR with its original search algorithm (an ES) for solving reactive power optimization problems. The authors reported a decrease of about 4% of the initial loss reported for such problems. Fonseca et al. [132] used ACO with SR with the aim of avoiding the careful fine-tuning of the penalty factors required for the solution of discrete structural optimization problems. The results of the proposed approach outperformed those obtained with traditional penalty functions. However, this approach has not been extended to other optimization problems to the authors' best knowledge.

Discussion

SR, in its original version whose search algorithm is an ES [6], has been applied to different domains [131]. Furthermore, in a similar way to the behavior found for the feasibility rules, SR has been combined with DE [123,124,130], although no actual ranking process is considered in the original DE algorithm. Nevertheless, this induced ranking mechanism in DE has motivated the use of special differential mutation variants specifically designed to solve CNOPs [123,124]. Also, SR has been coupled to other NIAs such as EP (in which the replacement mechanism uses a ranking process [129]), and ACO [126,132]. SR has also been improved by its authors by combining ES with the mutation operator adopted in DE [54]. Finally, SR has also been employed in one of the few known efforts to apply fitness approximation techniques to CNOPs [127].

3.3. ε-constrained method

One of the most recent constraint-handling techniques reported in the specialized literature is the ε-constrained method proposed by Takahama and Sakai [133]. This mechanism transforms a CNOP into an unconstrained numerical optimization problem and it has two main components: (1) a relaxation of the limit to consider a solution as feasible, based on its sum of constraint violation, with the aim of using its objective function value as a comparison criterion, and (2) a lexicographical ordering mechanism in which the minimization of the sum of constraint violation precedes the minimization of the objective function of a given problem. The value of ε, satisfying $\varepsilon > 0$, determines the so-called ε-level comparisons between a pair of solutions $\vec{x}_1$ and $\vec{x}_2$ with objective function values $f(\vec{x}_1)$ and $f(\vec{x}_2)$ and sums of constraint violation $\phi(\vec{x}_1)$ and $\phi(\vec{x}_2)$ (which can be calculated as stated in Eq. (7)) as indicated in Eqs. (8) and (9).

$$(f(\vec{x}_1), \phi(\vec{x}_1)) <_\varepsilon (f(\vec{x}_2), \phi(\vec{x}_2)) \Leftrightarrow \begin{cases} f(\vec{x}_1) < f(\vec{x}_2), & \text{if } \phi(\vec{x}_1), \phi(\vec{x}_2) \le \varepsilon \\ f(\vec{x}_1) < f(\vec{x}_2), & \text{if } \phi(\vec{x}_1) = \phi(\vec{x}_2) \\ \phi(\vec{x}_1) < \phi(\vec{x}_2), & \text{otherwise} \end{cases} \quad (8)$$

$$(f(\vec{x}_1), \phi(\vec{x}_1)) \le_\varepsilon (f(\vec{x}_2), \phi(\vec{x}_2)) \Leftrightarrow \begin{cases} f(\vec{x}_1) \le f(\vec{x}_2), & \text{if } \phi(\vec{x}_1), \phi(\vec{x}_2) \le \varepsilon \\ f(\vec{x}_1) \le f(\vec{x}_2), & \text{if } \phi(\vec{x}_1) = \phi(\vec{x}_2) \\ \phi(\vec{x}_1) < \phi(\vec{x}_2), & \text{otherwise.} \end{cases} \quad (9)$$

As can be seen, if both solutions in the pairwise comparison are feasible, slightly infeasible (as determined by the ε value) or even if they have the same sum of constraint violation, they are compared using their objective function values. If both solutions are infeasible, they are compared based on their sum of constraint violation. Therefore, if $\varepsilon = \infty$, the ε-level comparison works by using only the objective function values as the comparison criteria. On the other hand, if $\varepsilon = 0$, then the ε-level comparisons $<_0$ and $\le_0$ are equivalent to a lexicographical ordering in which the minimization of the sum of constraint violation $\phi(\vec{x})$ precedes the minimization of the objective function $f(\vec{x})$, as promoted by the use of feasibility rules [45].

Takahama and Sakai [134] have an earlier approach called the α-constrained method. In this case, the authors perform α-level comparisons which work in a similar way as those of the ε-constrained method. However, unlike the ε value which represents a tolerance related to the sum of constraint violation, the α value is related to the satisfaction level of the constraints for a given solution. Therefore, the condition to consider the objective function as a criterion in a pairwise comparison is based on the aforementioned satisfaction level of both solutions. If both levels are higher than an $0 \le \alpha \le 1$ value, the comparison can be made by using the objective function value, regardless of the full feasibility of the solutions. The main drawback of the α-constrained method with respect to the ε-constrained method is that the first may require user-defined parameters to compute the satisfaction level [134], while the second uses the sum of constraint violation which requires no additional parameters (see Eq. (7)). Nonetheless, in both mechanisms, the careful fine-tuning of α and ε remains as the main shortcoming. The authors have proposed dynamic mechanisms [135,136] which have allowed these two algorithms to provide competitive results.

The α-constrained method was coupled to a GA in [134], while the use of the Nelder–Mead method was reported by the same authors in [135]. The results obtained by using multiple simplexes allowed the approach to obtain competitive results with respect to those found by SR [6]. Wang and Li adopted the α-constrained method in [137], using DE as their search engine, and improved the results reported in [135]. Also, the ε-constrained method was combined with a hybrid PSO–GA algorithm by Takahama et al. in [133]. The approach considered the reproduction for particles as in a GA with the goal to tackle the premature convergence observed in a previous version in which the α-constrained method was coupled only to PSO [138]. The hybrid approach was tested only in one benchmark function and two engineering design problems [133].

A successful attempt to find a more suitable search algorithm for the ε-constrained method was reported in [136], where a DE variant (DE/rand/1/exp) and a gradient-based mutation operator (acting as a local search engine) were employed. This version obtained the best overall results in a competition on constrained

real-parameter optimization in 2006, in which a set of 24 test problems were solved [139]. Gradient-based mutation was applied to newly infeasible generated trial vectors in order to make them feasible. Evidently, the main limitation of this sort of approach is that gradient information must be computed. Also, additional parameters must be fine-tuned by the user in this approach. Finally, it is worth remarking that there seem to be no studies that analyze the role of adopting gradient-based information in an EA used for constrained optimization.

Further improvements have been proposed to the ε-constrained method. In [140], Takahama and Sakai improved the dynamic control for the ε value by using an adaptive approach which allowed a faster decrease in its value if the sum of constraint violation was reduced quickly enough during the search process. This mechanism produced improved results when dealing with CNOPs having equality constraints. However, in this case, there is also an additional user-defined parameter, which is related to the adaptation process. Additionally, the authors did not analyze the performance of this variant in CNOPs that have only inequality constraints.

In [141], Takahama and Sakai improved their approach by adding a decreasing probability on the use of the gradient-based mutation. They also introduced two new mechanisms to deal with boundary constraints: (1) one based on a reflecting back process for variable values lying outside the valid limits when DE mutation was applied, and (2) another one that consisted in assigning the limit value to a variable lying outside a boundary when the gradient-based mutation was computed. With the aforementioned changes, the authors could obtain feasible solutions for one highly difficult problem known as g22 [139]. The main drawback of this improved version was the addition of user-defined parameters for the dynamic mechanism used by the gradient-based mutation operator. A further improved version of the aforementioned algorithm was proposed by Takahama and Sakai in [142], where an archive to store solutions and the ability of a vector to generate more than one trial vector [143] were added. The approach has provided one of the most (if not the most) competitive performance in different sets of test problems. However, the algorithm still depends on the gradient-based mutation to provide such competitive results.

Motivated by its competitive performance, the ε-constrained method has been adopted as a constraint-handling technique in other proposals. This is the case of the jDE algorithm proposed by Brest et al. [144], in which the authors propose to self-adapt the parameters of DE using stochastic values. In a version called ε-jDE [145], the ε-constrained method was one of the improvements proposed, besides the use of additional DE variants and a reduction scheme of the population size. Brest [145] also added a novel way to adapt the ε value, but additional user-defined parameters were introduced. However, the results obtained by ε-jDE were highly competitive in a set of 24 well-known benchmark problems. An improved version called jDEsoco was proposed in [146], where an aging mechanism to replace those solutions stagnated in a local optimum was added. Moreover, only the 60% of the population was compared by the ε-constrained method and the remaining 40% was compared by only using the objective function value. The results were improved but two parameters, the population ratio and an aging probability were added to the algorithm.

Zeng et al. [147] employed the ε-constrained method with a ε variation process based on the dynamic decrease mechanism originally proposed in [27]. A crossover operator biased by the barycenter of the parents, plus a uniform mutation were used as variation operators. The approach was tested in 24 test problems and the results were found to be competitive with respect to the ε-constrained DE [136]. An improved version of this approach was proposed by Zhang et al. in [148], where a gradient-based mutation similar to the one proposed by Takahama and Sakai [136] was added. The results obtained by this approach were compared with respect to those obtained by the ε-constrained DE. The approach added parameters related with the variation of the ε value as well as those required by the gradient-based mutation.

Mezura-Montes et al. used the ε-constrained method within the ABC algorithm [121]. Additionally, a dynamic mechanism to decrease the tolerance for equality constraints was considered. The results obtained outperformed those reported by a previous ABC version in which feasibility rules were used as the constraint-handling technique [93]. However, this approach showed premature convergence in some test problems having high dimensionality.

Discussion

The ε-constrained method and its predecessor, the α-constrained method, have been applied to different NIAs such as GAs [134], PSO [138], hybrid PSO–GA [133] and (mainly) to DE [136,137]. Even mathematical programming methods have been used with this approach [135].

The improvements on the ε-constrained method rely on the use of the gradient-based local search operator [141], the way the ε value is fine-tuned [135,136,140], and the usage of an archive to store solutions [142]. On the other hand, there are approaches whose constraint-handling mechanism was originally based on feasibility rules but then switched to the ε-constrained method: for example, the approach reported in [145] which is based on DE, and the approach reported in [121], which is based on ABC.

Finally, the ε-constrained method has been coupled to other NIAs with different variation operators [147], but the need of the gradient-based mutation to provide highly competitive results has been emphasized in them [148].

3.4. Novel penalty functions

In spite of the fact that the two previous types of constraint-handling techniques discussed avoid the use of a penalty function, there are proposals based on such penalty functions which provide very competitive results. Here, we will briefly review the most representative work in this direction.

Xiao et al. [149] used the so-called KS function in a static penalty function to solve CNOPs. However, even when the approach was competitive in some test problems, it was clearly outperformed in others.

Deb and Datta [150] revisited the static penalty function by proposing a method to compute a suitable value for a single penalty factor, assuming the normalization of the constraints. As a first step a bi-objective problem was solved by a multi-objective evolutionary algorithm (MOEA) (e.g., NSGA-II [151]). The first objective was the original objective function while the second was the sum of constraint violation $\phi = 0$. Furthermore, ϕ was restricted by a tolerance value (in a similar way as the ε-constrained method [136] but with a fixed value in this case). The tolerance value was determined by a user-defined parameter based on the number of constraints of the problem. After a certain number of generations (also defined by the user), a cubic curve to approximate the current obtained Pareto front was generated by using four points whose ϕ values were below a small tolerance. The penalty factor was then defined by calculating the corresponding slope at $\phi = 0$. After that, a traditional static penalty function was used to solve the original CNOP by using a local search algorithm (Matlab's **fmincon()** procedure was used by the authors) using the solution with the lowest ϕ value from the population of the MOEA as the starting point for the search. The termination criterion for the local search algorithm was the feasibility of the final solution combined with a small tolerance for the difference between objective function values of the starting point and the

final one. The approach was tested on a set of six benchmark problems in which it obtained competitive results, while requiring a significant lower number of evaluations with respect to those reported by other state-of-the-art NIAs. The approach, however, requires the calibration of the MOEA as well as a tolerance value for the constraint related to the sum of constraint violation. Additionally, it also requires the number of generations to define the interval of use for the local search and, finally, the tolerance for the termination criterion of the local search. It is worth noting that this approach considered only inequality constraints. In [152], Datta and Deb extended their approach to deal with equality constraints, too. The extension consisted in two main changes: (1) the punishment provided by the penalty value obtained by the bi-objective problem was increased if the local search failed to generate a feasible solution and (2) the small tolerance used for choosing the four points employed to approximate the cubic curve was relaxed. Both changes were motivated by the difficulties to generate feasible solutions caused by the presence of equality constraints. The results obtained in eight well-known test problems were highly competitive with respect to two state-of-the-art approaches based on PSO and DE.

Tasgetiren and Suganthan [153] proposed the use of a dynamic penalty function coupled with a multi-population differential evolution algorithm. In this approach, the authors allowed a user-defined number of sub-populations to evolve independently. However, the selection of the solutions to compute the differential mutation could be made by considering all sub-populations. Furthermore, a regrouping process, similar to a recombination operator among best solutions in each sub-population, was carried out after a number of generations, defined by the user. The approach was tested on 24 test problems, and the authors reported a high sensitivity of their approach to the parameters related with the severity of the penalty.

Farmani and Wright [154] proposed a two-parts adaptive penalty function in which no penalty factors need to be defined by the user. The first part increases the fitness of the infeasible solutions with a better value of the objective function with respect to the best solution in the current population. The best solution can be the feasible solution with the best objective function value. However, if no feasible solutions are present in the population, the best solution is the infeasible solution with the lowest sum of constraint violation. This first part of the penalization focuses on promoting diversity in promising regions of the search space, regardless of their feasibility. The second part modifies the fitness values of the worst infeasible solutions (those with the highest sum of constraint violation and a poor objective function value) aiming to make them similar to the fitness of the solution with the worst value of the objective function. The aim is to generate more solutions in the boundaries of the feasible region but with better values of the objective function. In spite of its lack of user-defined penalty factors, the approach was computationally expensive, since it required more than one million evaluations to provide competitive results in a set of 11 test problems.

Puzzi and Carpinteri [155] explored a dynamic penalty function based on multiplications instead of summations in a GA-based approach. However, this approach performed well in problems having only inequality constraints.

Tessema and Yen [156] used the number of feasible solutions in the current population to determine the penalty value assigned to infeasible solutions in a two-penalty based approach. This parameterless penalty function allows, based on the feasibility of solutions in the population, to favor slightly infeasible solutions having a good objective function value, as promoted in [62]. This is done in the selection process by assigning such solutions a higher fitness value. The approach obtained competitive results in 22 test problems. However, the number of evaluations required was higher (500,000) than that required by other state-of-the-art approaches (they require around 250,000 evaluations). Furthermore, three mutation operators (which require three mutation probabilities defined by the user) are required to maintain the explorative capabilities of the approach.

Mani and Patvardhan [157] explored the use of an adaptive penalty function in a two-population-GA-like-based approach in which the first population evolves by using a parameter-free adaptive penalty function based on the objective function and the constraint violation of the best solution available so far in the population. The other population evolves based on feasibility rules. Then, both populations exchange their best solutions plus an additional percentage of randomly chosen solutions. The approach was tested on a set of test problems. However, the approach required parameters related to the migration process as well as the variation operators, as well as a local search mechanism based on gradient information.

In an analogous way as Coello [29] used co-evolution to optimize penalty factors to solve CNOPs by using two-nested GAs, He et al. [158] used two PSO algorithms instead. Their approach was used to solve a set of engineering design problems and the results were encouraging. However, as in the approach using GAs, this one requires the definition of parameter values for the two PSO algorithms.

Wu [159] proposed an AIS which combines the metaphor of clonal selection with idiotypic network theories. To deal with CNOPs, an adaptive penalty function was defined to assign its affinity to each antibody. Different operators based on the clonal selection principle, affinity maturation and the bone marrow operator were applied to generate new solutions. The approach was tested on four benchmark nonlinear programming problems and four generalized polynomial programming (GPP) problems.

Discussion

In spite of their well-documented shortcomings, penalty functions are still being used and improved in the specialized literature, as has been made evident in the previously discussed approaches. The open question stated in Section 2.1 about which penalty function is the most competitive might find an answer here, because the most frequent are approaches based on adaptive penalty functions [154,156–159], in which (interestingly enough) a GA is used, in most cases, as the search engine [154,156–158]. Dynamic penalty functions, which adopt the current generation number to control the decrement of the penalty factor are still popular [153,155]. Finally, even traditional static penalty functions have been scarcely proposed as new constraint-handling methods [149]. Additionally, there are other approaches in which more elaborate methods are used to define the appropriate penalty factors and which have obtained highly competitive results (see for example [150,152]).

3.5. Novel specialized operators

Leguizamón and Coello Coello [160] proposed a boundary operator based on conducting a binary search between a feasible and an infeasible solution. Furthermore, three strategies to select which constraint (if more than one is present in a CNOP) is analyzed. The search algorithm was an ACO variant for continuous search spaces. The approach provided highly competitive results, mostly, as expected, in problems having active constraints. However, it was outperformed in others. The main disadvantage of the approach is the need of an additional constraint-handling technique (a penalty function was used in this case) to deal with solutions which are on the boundary of the constraint treated but violate other constraints. Furthermore, no other search algorithms which are more popular in the solution of CNOPs (e.g., DE, ES) have been coupled to this proposed boundary operator.

Huang et al. [161] proposed a boundary operator in a two-population approach. The first population evolves by using DE as the search engine, based only on the objective function value (regardless of feasibility). The second population stores only feasible solutions and the boundary operator uses solutions from both populations to generate new solutions, through the application of the bisection method in the boundaries of the feasible region. Furthermore, the Nelder–Mead simplex method was used as a local search operator applied to the best feasible solutions. Unlike Leguizamón and Coello's proposal, this approach does not require an additional constraint-handling technique, but a feasible solutions is needed at the beginning of the process. The approach was tested only in a few problems having only inequality constraints and it required different parameter values for each test problem, showing some sensitivity to them.

Wanner et al. [162] proposed the Constraint Quadratic Approximation (CQA), which is a special operator designed to restrict an evolutionary algorithm (a GA in this case) to sample solutions inside an object with the same dimensions of the feasible region of the search space. This is achieved by a second-order approximation of the objective function and an equality constraint, which is updated at each generation. A subset of solutions from the population was used to build the quadratic approximations. The operator was applied based on a number of generations defined by the user. Moreover, a static penalty function was used to guide the GA search and the equality constraint was transformed into two inequality constraints by using a small ϵ tolerance. The approach was tested on a small set of problems but it could only deal with one quadratic equality constraint. With the aim of solving CNOPs with more than one equality constraint, Peconick et al. [163] proposed the Constraint Quadratic Approximation for Multiple Equality Constraints (CQA-MEC). This was achieved by an iterative projection algorithm which is able to find points satisfying the approximated quadratic constraints with a low computational overhead. However, CQA-MEC still requires the static penalty function to work as its predecessor (CQA). Araujo et al. [164] extended the previous approaches to deal with multiple inequality constraints by using an special operator in which the locally convex inequality constraints are approximated by quadratic functions, while the locally non-convex inequality constraints are approximated by linear functions. The dependence of the static penalty function remains in this last approach.

Ullah et al. [165] proposed an agent-based memetic algorithm to solve CNOPs, in which the authors adopt a special local operator for equality constraints, which is one of five life span learning processes. After a selection process in which pairs of agents (i.e., solutions) are chosen based on their fitness and location in the search space, the SBX operator is applied. Thereafter, the special operator for equality constraints is applied to some individuals in the population as follows: the satisfaction of a randomly chosen equality constraint is verified for a given solution. If it is not satisfied, a decision variable, also chosen at random, is updated with the aim to satisfy it. If the constraint is indeed satisfied, two other variables are satisfied in such a way that the constraint is still satisfied (i.e., the constraint is sampled). This special operator is only applied during the early stages of the search because it reduces the diversity in the population. The four other learning processes were taken from a previous version discussed in an earlier paper [118]. These processes are applied based on their success. The approach was tested on a set of benchmark problems with equality constraints and the results were promising. However, the approach requires additional parameters to be defined by the user (e.g., the number of generations during which the operator must be applied, the number of decision variables to be updated in the equality constraint). In fact, the authors do not provide any guidelines regarding the way in which these parameters must be tuned.

Lu and Chen [166] proposed an approach called self-adaptive velocity particle swarm optimization (SAVPSO) to solve CNOPs. This approach relies on an analysis based on three elements: (1) the position of the feasible region with respect to the whole search space, (2) the connectivity and the shape of the feasible region, and (3) the ratio of the feasible region with respect to the search space. As a result of this analysis, the velocity update formula was modified in such a way that each particle has the ability to self-adjust its velocity according to the aforementioned features of the feasible region. The fitness of a solution is assigned based on its feasibility: feasible solutions are evaluated by their objective function value, while infeasible solutions are evaluated by their sum of constraint violation. The approach was tested on a set of 13 benchmark problems. The approach, however, showed some sensitivity to some of its parameters.

Spadoni and Stefanini [167] transformed a CNOP into an unconstrained search problem by sampling feasible directions instead of solutions of a CNOP. Thereafter, three special operators, related to feasible directions for box constraints, linear inequality constraints, and quadratic inequality constraints, are utilized to generate new solutions by using DE as the search engine. The main contribution of the approach is that it transforms a CNOP into an unconstrained search problem without using a penalty function. However, it cannot deal with nonlinear (either equality or inequality) constraints.

Wu et al. [168] and Li and Li [169] modified variation operators in NIAs in such a way that the recombination of feasible and infeasible solutions led to the generation of more feasible solutions. An adaptive mechanism to maintain infeasible solutions was added to the approach. This latter version was specifically based on DE's variation operators [169].

Discussion

The recent proposals based on the use of special operators that have been revised here emphasize the current focus on generating proposals which are easier to generalize. Such is the case of the definition of boundary operators in ACO [160] and DE [161]. Moreover, there are operators to approximate equality [162,163,165] and inequality [164] constraints. Finally, there have also been attempts to modify variation operators of some NIAs aiming to tailor them to a certain (desirable) behavior when solving CNOPs [166,168,169].

3.6. Use of multi-objective concepts

In spite of the fact that empirical evidence has suggested that multi-objective concepts are not well-suited to solve CNOPs, there are highly competitive constraint-handling techniques based on such concepts.

Motivated by the idea of keeping suitable infeasible solutions [27,43,62,170] Ray et al. [171] proposed the Infeasibility Driven Evolutionary Algorithm (IDEA) whose replacement process requires the definition of a proportion of infeasible solutions to remain in the population for the next generation. IDEA works in a similar way as NSGA-II [151]. Nonetheless, an additional objective, besides the original objective function, is added. This objective consists on the constraint violation measure, whose value is computed as follows: each individual in the population has a rank for each constraint of the CNOP being solved and each rank value depends on the constraint violation value for such solution (lower values are ranked higher because they represent a smaller violation for a constraint). If a solution satisfies the constraint, a zero rank is assigned to it. After each solution is ranked for each constraint, the violation measure is computed as the sum of ranks per solution. After the offspring are generated, the union of parents and offspring is split in two sets, one with the feasible solutions and the other with the infeasible ones. Non-dominated sorting is used to rank both

sets separately and, based on the proportion of desired feasible solutions, they are chosen first from the infeasible set and later on, the best ranked feasible solutions are chosen. IDEA is able to work with CNOPs and also with constrained numerical multi-objective optimization problems (CNMOPs). However, its performance has been more competitive when solving CNMOPs. The usage of local search, sequential quadratic programming in this case, was added to IDEA in the so-called Infeasibility Empowered Memetic Algorithm (IMEA) [172]. The approach was tested in eighteen scalable test problems and its performance improved with respect to the original IDEA when solving CNOPs. However, the local search algorithm adopted requires gradient calculation.

Reynoso-Meza et al. [173] proposed the spherical-pruning multi-objective optimization differential evolution (sp-MODE) to solve CNOPs, which were transformed into three-objective optimization problems, where the first objective was the original objective function, the second objective was the sum of constraint violation for inequality constraints and the third objective was the sum of constraint violation for equality constraints. An external archive was used to store non-dominated solutions. The sphere-pruning operator aims to find the best trade-off between feasibility and the optimization of the objective function. The approach required the definition for some parameter values depending of the number of constraints. However, the sphere-pruning operator might be an interesting operator to be applied in some parts of the search.

Wang et al. [174] proposed the use of Pareto dominance in a Hybrid Constrained EA (HCOEA) to solve a CNOP which was transformed into a bi-objective optimization problem. In this case, the first objective is the original objective function while the second one is the sum of constraint violation. A global search carried out by an EA is coupled to a local search operator based on a population division scheme and on the use of the SPX operator. In both cases, Pareto dominance is the criterion adopted to select solutions. The approach was tested in 13 benchmark problems and the results were found to be competitive with respect to four state-of-the-art algorithms. However, the approach requires the definition of two crossover probabilities (one for the global search and another for the local search) as well as the number of subsets in which the population will be divided. HCOEA showed some sensitivity to this last parameter.

Wang et al. [175] proposed an steady state EA to solve a CNOP which was also transformed into a bi-objective problem. At each generation, a set of offspring solutions are generated by applying orthogonal crossover to a randomly chosen set of solutions in the current population. After that, the non-dominated solutions obtained from the set of offspring are chosen. If there are no feasible offspring, two randomly chosen solutions from the set of parents will be replaced by the offspring which dominate them. Alternative, solutions can also be chosen if they have a lower sum of constraint violation. Furthermore, the individual with the lowest sum of constraint violation will replace the worst parent in the population. If there are feasible offspring, based on a user-defined probability, two randomly chosen parents will be replaced by two offspring which dominate them. Otherwise, the worst parent, based on feasibility rules [45], will be replaced by one offspring. After the steady state replacement, all solutions are affected by an improved version of the BGA mutation operator [176] based on a user-defined probability. The approach was tested in a set of 11 test problems and showed competitive results in some of them, but premature convergence was observed in others. Furthermore, some additional parameters need to be defined by the user.

Wang et al. [177], in their adaptive trade-off model (ATM) evolution strategy (ATMES), divided the search in three phases based on the feasibility of solutions in the population: (1) only infeasible solutions, (2) feasible and infeasible solutions, and (3) only feasible solutions. Owing to the fact that the CNOP was transformed into a bi-objective problem, the selection in the first phase was based on Pareto dominance. From the Pareto front obtained, the solutions were ranked in ascending order based on the sum of constraint violation and the first half was chosen to survive for the next generation and was deleted from the set. The process was repeated until the desirable number of solutions was achieved. The second phase was biased by a fitness value which is adapted based on the percentage of feasible solutions in the population. The last stage was biased only by the objective function value. The approach provided competitive results in 13 test problems. However, ATMES required some parameters related to the tolerance for equality constraints and the stepsize employed by the ES used as the search engine. This same ATM was coupled by Wang et al. [178] with a NIA in which the offspring generation was as follows: An offspring was generated by one of two variation operator: (1) simplex crossover or (2) one of two mutations (uniform mutation or improved BGA mutation). The approach, besides being tested on a set of 13 benchmark problems, was used to solve some engineering design problems. The results obtained by the authors were found to be very competitive, but some cases of premature convergence were reported. Another improvement to the ATM, which is based on a shrinking mechanism proposed by Hernández-Aguirre et al. [59], was proposed by Wang et al. [179]. This approach, called Accelerated ATM (AATM), outperformed both the original ATM and the approach proposed by Hernández-Aguirre et al. However, additional parameters (which are required by the shrinking mechanism) were introduced by the authors. The ATM was coupled with DE in a recent approach [180], showing an improvement in the results with respect to previous versions of the same algorithm. Liu et al. [115] used the ATM in an EA but with two main differences: (1) good point set crossover was used to generate offspring and (2) feasibility rules were the criteria to select solutions in the second stage of the ATM (at which there are feasible and infeasible solutions in the current population). The approach was tested in some benchmark problems. However, the performance of the proposed crossover operator was not found to be clearly better with respect to the previous version of this approach reported by Wang et al. (in which an ES is adopted as the search engine) [177].

Gong and Cai [181] used Pareto dominance in the many-objective space defined by the constraints of a problem as a constraint-handling mechanism in a DE-based approach. An orthogonal process was employed for both, generating the initial population and for applying crossover. Furthermore, the ϵ-dominance concept [182] was adopted to update an external archive in which the non-dominated solutions found during the search were stored. Orthogonal crossover was applied after DE generated the offspring population. In fact, an intermediate child population was designed to store the offspring which were non-dominated with respect to their parents. The aim is to perform a non-dominance checking on the union of the parent population and the offspring population as a replacement mechanism at the end of each generation of the algorithm. Although the approach provided competitive results in a set of 13 test problems, the contribution of each of the additional mechanisms adopted is not clear. Additionally, no information is provided regarding the fine-tuning required for the parameters required by this approach.

Li et al. [183] solved a CNOP which was also transformed into a bi-objective optimization problem by using a PSO algorithm in which Pareto dominance was used as a criterion in the pbest update process and in the selection of the local-best leaders in a neighborhood. In case of ties, the sum of constraint violation worked as a tie-breaker. A mutation operator was also added to keep the approach from converging prematurely. Additionally, a small tolerance was used to consider as feasible to solutions that

were slightly infeasible (this is similar to the ε-constrained method but with a fixed value instead of a dynamic one). The approach was tested only on three engineering design problems.

Venter and Haftka [184] also transformed a CNOP into a bi-objective optimization problem and used PSO as their search engine. However, the leader selection was based most of the time on the sum of constraint violation, while the rest of the time the criterion was one of the three following choices: (1) the original objective function, (2) the crowding distance or (3) Pareto dominance. The approach was tested on several benchmark problems and some engineering design problems.

Wang et al. [185] used a hybrid selection mechanism based on Pareto dominance and tournament selection into a Adaptive Bacterial Foraging Algorithm (ABFA) to solve CNOPs. The approach uses the so-called good nodes set method to initialize the population, to perform crossover and to spread similar individuals throughout the search space. The approach was tested in a set of benchmark problems and in some engineering design problems, providing competitive results in both cases.

Discussion

Based on the taxonomy proposed by Mezura-Montes and Coello Coello in [53] and described in Section 2.4, both CNOP's transformations (both into a bi-objective problem and into a general multi-objective problem, i.e., having more than two objectives), have been popular in the literature. However, in recent years, and perhaps motivated by the current challenges in many-objective optimization [186], the use of transformation of a CNOP into a bi-objective optimization problem has been preferred [115,174,175,177–180,183–185], with respect to considering each constraint as an independent objective [49,50,181]. Furthermore, in most cases, the use of multi-objective concepts is complemented with other mechanisms (e.g., feasibility rules [175], ranking based on the sum of constraint violation [115], ϵ-dominance [181]). This aims to produce approaches that are competitive with respect to state-of-the-art algorithms.

On the other hand, from the type of multi-objective concepts adopted, both Pareto ranking [115,177–180] and Pareto dominance [173–175,181,183–185] have been popular in recent approaches. Finally, the use of non-dominated sorting coupled to a diversity mechanism which keeps good infeasible solutions has also been reported [171], although this sort of scheme has been less popular in the literature.

3.7. Ensemble of constraint-handling techniques

Motivated by the no free lunch theorems [187], Mallipeddi and Suganthan [188] proposed an ensemble of four constraint techniques: feasibility rules, stochastic ranking, a self-adaptive penalty function and the ε-constrained method in a four sub-population scheme designed to solve CNOPs. Two versions were presented, one based on a modified version of a classical evolutionary programming (EP) algorithm [129], and another based on DE which was initially tested on eighteen scalable test problems in [189] and further tested in [188]. Each constraint-handling technique was used to evolve an specific sub-population. Unlike typical island models in parallel NIAs, in which sub-populations evolve separately, in the ECHT there is a close communication among them since these sub-populations share all of their offspring, i.e., an individual disregarded by its sub-population may survive in another population. Based on their experiments, the authors concluded that the ECHT performs better than any of the constraint-handling techniques that it contains. The approach was extensively tested on a set of 37 test problems. The results obtained by both ECHT versions were highly competitive. However, the main drawback of the approach is the calibration required (from the user) for each of the constraint-handling techniques adopted.

Elsayed et al. [122] proposed a DE-based algorithm where the combination of four DE-mutations, two DE recombinations and two constraint-handling techniques (feasibility rules and ε-constrained method) generated sixteen variants which were assigned to each individual in a single-population algorithm. The rate of usage for each variant was based on its improvement ability measured by its ability to improve solutions. Moreover, a local search algorithm was applied to a random solution at some generations. The approach was tested on a set of 18 test problems with 10 and 30 dimensions, showing a very competitive performance. Its main drawback, as in the previous approach mentioned in this section, is the number of parameters to be tuned by the user.

A similar idea was presented in a combination of two DE variants and a variable neighborhood search with three constraint-handling techniques (feasibility rules, ε-constrained method, and an adaptive penalty function) by Tasgetiren et al. [190]. The approach was tested on eighteen test scalable test problems in 10D and 30D. No comparisons nor discussion were included in the paper but it is clear that the constraint-handling mechanisms (with the exception of the feasibility rules) must be fine-tuned by the user.

Discussion

The ECHT opens a new paradigm in constraint-handling techniques: the design of mechanisms that allows the combination of approaches that can be seen as complementary (in terms of the way in which they operate). Furthermore, the study of different search algorithms (EP and DE in this case) also generates a new research trend in which the aim is to understand better the behavior of different NIAs when coupled to constraint-handling techniques (some studies in this direction already exist, but they have focused only on one particular type of constraint-handling technique [11,97]). However, as the combination of several techniques considerably enhances the capabilities of an approach, it is also required to define parameter values for each of these techniques in such a way that a proper balance is maintained. In other words, parameter control [191] becomes an important issue when designing ensemble approaches. It is also worth noting the similarities between ensemble approaches and hyper-heuristics as discussed later in this paper.

A summary of the main features of each constraint-handling technique is provided in Table 1.

3.8. The importance of the search algorithm

Based on the ever increasing number of NIAs that have been developed to solve optimization problems in the last few years (e.g., PSO [5], DE [192], BFOA [96], ABC [193], ACO in its versions for numerical search spaces [194], AIS [107]), plus the more traditional approaches (i.e., GAs, EE, and EP [1]), how to decide which algorithm to use for a particular application has become an important issue. From the novel and popular constraint-handling techniques previously discussed and the different search algorithms employed, a few trends can be inferred, as briefly discussed next.

The review presented in this paper suggests a preference for DE over other NIAs [46,50,64–66,69,70,72,76,90,97,98,101,102,120,122–125,128,130,136,140–142,144–146,153,161,169,173,180,181,188–190]. Regarding DE-based approaches to solve CNOPs, most of them use feasibility rules as their constraint-handling mechanism [46,64–66,69,70,72,76,90,97,98,101,102,120,144,161,190].

SR [123–125,128,130], and the ε-constrained method [136,140–142,145,146] have both been coupled to DE. Some recent multi-objective-based constraint-handling techniques such as the one that considers each constraint as an objective [50,181] and the ATM [180] have both been coupled to DE as well. In this,

Table 1
Summary of main features of the recent constraint-handling techniques: *FR*: feasibility rules, *SR*: stochastic ranking, *ε-CM*: ε-constrained method, *NPF*: novel penalty functions, *NSO*: novel special operators, *MOC*: multi-objective concepts, *ECHT*: ensemble of constraint-handling techniques.

Technique	Core concept	Pros	Cons
FR	Three criteria for pairwise selection	Simple to add into a NIA no extra parameters.	May cause premature convergence.
SR	Ranking process	Easy to implement	Not all NIAs have ordering in their processes one extra parameter
ε-CM	Transforms a CNOP into an unconstrained problem	Very competitive performance	Extra parameters required. Local search for high performance
NPF	Focus on adaptive and dynamic approaches	Well-known transformation process.	Some of them add extra parameters.
NSO	Focus on boundary operators and equality constraints	Tendency to design "easy to generalize" operators	Still limited usage
MOC	Focused on bi-objective transformation of a CNOP	Both, Pareto ranking and dominance still popular	May require an additional constraint-handling technique
ECHT	Combination of two or more constraint-handling techniques	Very competitive performance	Requires the definition of several parameter values

it is particularly interesting to note that the ATM was originally proposed with an ES [177]. However, the authors have reported better results with the DE-based version [180]. Finally, adaptive penalty functions [153] and the most recent version of the ECHT [188,189], use both DE as their search engine.

GAs have remained as a popular choice of several researchers and practitioners [98–100,103,115,134,147–150,152,154,156,157,162,163,165,171,174,175,178]. From the GA-based approaches revised in this paper, most of them are based on penalty functions [149,150,152,154,156,157] followed by those based on multi-objective optimization concepts [115,171,174,175,178]. Some others are based on the ε-constrained method [98,134,147] and special operators [162,163,165]. It is worth noticing, however, that some GA-based approaches [98,178] have also been implemented using DE instead. In such cases, the authors reported better results when using DE [98,180]. However, highly competitive results were found with a GA in [99], where a multiparent crossover seems to contribute to such performance.

PSO has been popular to solve CNOPs as well [11,49,79–89,105,106,133,138,158,166,183]. The use of feasibility rules to handle constraints has been the most popular choice in PSO-based approaches [11,79–89,105,106]. Additionally, there are a few approaches based on the ε-constrained method [133,138], on penalty functions [158], on the use of special operators [166] and on multi-objective optimization concepts [49,183].

ES have been chosen less often than any of the previous NIAs to solve CNOPs [6,27,54,59,61,63,127,131,177] and they are mainly associated with SR [6,54,127,131]. However, ES has also been coupled with feasibility rules [59,61,62], with the ATM [177] and with adaptive penalty functions [27].

Although ES and EP share characteristics in their original versions, when dealing with CNOPs, the latter is clearly less popular than the former. There is, for example, an approach based on EP which adopts SR [129] and another one that was coupled to the ECHT [188]. However, this last approach was slightly outperformed by the ECHT that uses DE instead [188].

In spite of the fact that ACO versions for continuous optimization are relatively rare, some of them have been used to solve CNOPs adopting, for example, SR [126,132] and special operators [160].

Regarding the most recent NIAs based on cooperative behaviors of simple living organisms such as ABC and BFOA, the most preferred constraint-handling mechanism has been the feasibility rules for ABC [91–94] and BFOA [95]. There is also an attempt of using the ε-constrained method coupled to ABC [121] and one of using Pareto dominance in BFOA [185].

Regarding AIS-based approaches for CNOPs, although in their early days it was common to adopt measures of similarity at bit-string level as their constraint-handling mechanism [108–110], today there is a trend to use feasibility rules [111–114]. There are also attempts to use adaptive penalty functions with AIS-based approach [159].

Finally, there are other NIAs, such as the OEA, in which static penalty functions have been considered, although the use of feasibility rules has provided a more stable performance [115].

It can also be observed from some of the most recent approaches that the combination of global and local search engines (i.e., memetic algorithms), has also become popular for solving CNOPs. There are, for example, local search engines based on gradient information which have been coupled to NIAs for solving CNOPs [48–50,73,94,103,116,122,136,140–142,148,157,172]. Additionally, there are also approaches in which direct search methods (i.e., not requiring derivatives) such as Nelder–Mead [76,135,161] or pattern search (also known as Hooke–Jeeves) [120] have been adopted. Furthermore there are proposals which combine different local search operators, some of which are based on gradient information, while others are based on direct search methods [117,118,165]. Variation operators such as Gaussian mutation [52,122] and simplex crossover [174] have been used as local search operators, too. In fact, other meta-heuristics such as SA has been used [82] to search locally in a CNOP. Even simple local search algorithms like Matlab's **fmincon()** function have been adopted [150,152].

The use of special operators to deal with equality constraints can also be seen as some form of local search operators, as suggested in [162–164].

On the other hand, there are hybrid approaches which combine two meta-heuristics (e.g., the PSO-DE hybrid [85] and the AIS-GA [195]).

4. Future trends

From the literature review reported in this paper, it is clear that there currently exist several well-established constraint-handling techniques available for solving CNOPs. However, there still are several research topics which have attracted little attention from researchers. A list of such topics which constitute, from the authors' perspective, promising paths for future research, is provided next.

4.1. Constraint-handling for multi-objective optimization

As stated by Yen [196] and by Ray et al. [171], the development of constraint-handling techniques for multi-objective optimization problems has received relatively little attention in the specialized literature. This may be due to the fact that most researchers assume that any constraint-handling technique developed for single-objective optimization can be easily coupled to a multi-objective algorithm. For example, death-penalties [197], adaptive penalty functions [198,199], use of constraints as objectives [200], stochastic ranking [201], feasibility rules [151,202,203]

(with a modification in the first rule, in which two feasible solutions are compared, since in this case non-dominance is checked instead of comparing objective function values), and special operators based on gradient information [204] have all been coupled to multi-objective evolutionary algorithms.

There are, however, constraint-handling techniques that were specifically designed for multi-objective problems. Such techniques mostly rely on the use of Pareto dominance or Pareto ranking. For example, Ray et al. [55] proposed the use of Pareto ranking in objective function space, constraints space, or both. Young [205] proposed the use of a combined value obtained from blending rankings from the objective and constraint spaces. Isaacs et al. [170] and Ray et al. [171] proposed the use of non-dominated sorting for feasible and infeasible solutions, separately, and Singh et al. [206] proposed the use of non-dominance checking combined with a special operator to favor constraints satisfaction in simulated annealing. Finally, Qu and Suganthan [207] reported the usage of the ensemble of three constraint-handling techniques (superiority of feasible points, ε-constrained method and an adaptive penalty function) to solve a set of well-known constrained multi-objective optimization problems.

There are other topics within this area that remain almost unexplored. For example, performance measures that are appropriate for constraint-handling techniques used in multi-objective optimization, diversity mechanisms specially designed for constrained search spaces, boundary operators for multi-objective problems, among others.

An interesting topic that has attracted a lot of attention in the evolutionary multi-objective optimization literature is the so-called many-objective optimization, in which many (i.e., four or more) objectives are considered. Many-objective optimization is strongly related to constraint-handling techniques, particularly when defining constraints as objectives. In this regard, there is some work from Saxena and Deb [208] in which the use of dimensionality reduction approaches has been explored to evaluate the importance of the constraints of a CNOP. The approach was later extended to constrained multi-objective optimization problems in [209]. Additionally, Saxena et al. [210] have explored the solution of constrained many-objective optimization problems, but there is practically no work done in this area yet.

4.2. Constraint approximation

In spite of the fact that fitness approximation methods have been extensively applied to unconstrained optimization problems [211,212], their use in CNOPs remains scarce. There are a few papers that report the use of approximation methods for equality and inequality constraints [162–164] as well as the use of fitness inheritance to solve CNOPs [213]. However, the use of approximation techniques for dealing with CNOPs is still relatively rare, opening the possibility for new developments that could be very valuable for real-world applications (when dealing with expensive constrained objective functions).

4.3. Dynamic constraints

Nowadays, there is a considerable amount of research devoted to deal with dynamic optimization problems. However, most of this work is focused on unconstrained search spaces [214] and very few efforts have focused on constrained problems. For example, Nguyen and Yao [215] adopted a penalty function and a special operator to transform infeasible solutions into feasible ones. An interesting conclusion in this preliminary study was the convenience of using special operators instead of keeping or maintaining diversity in the population, as normally done in unconstrained dynamic optimization problems. Singh et al. [216] used their IDEA algorithm (based on multi-objective concepts to handle constraints) as a sub-evolve mechanism within a framework to solve dynamic CNOPs (DCNOPs). Clearly, more work in this direction is still required. Furthermore, more test problems and appropriate performance measures are required in this area [215].

4.4. Hyper-heuristics

Hyper-heuristics, are approaches in which several metaheuristics act at different stages during the search, as determined by a centralized control mechanism [217]. Hyper-heuristics have been popular in combinatorial optimization, but their use in constrained problems is almost unexistent. The authors are aware of only one work in this direction, which remains unpublished (see [218]). It is evidently expected that hyper-heuristics become more popular in this area in the years to come. In fact, hyper-heuristics share clear similarities with the ensemble of constraint-handling techniques [122,188,190], and with approaches that adopt different variation operators in the same NIA [98]. However, no specific hyper-heuristic mechanisms were defined in any of those approaches.

4.5. Theory

Theoretical studies on constrained optimization using NIAs are also very scarce. There is some work on runtime analysis in constrained search spaces with EAs [219] and also in the usefulness of infeasible solutions in the search process [220]. Other theoretical studies have focused on some ES variants, such as the $(1 + 1)$-ES [221] and more recently the $(1, \lambda)$-ES [222], in both cases focused on the solution of a simple CNOP. However, more research in this area is required. It would be interesting, for example, to analyze convergence and fitness landscapes in the presence of constraints.

5. Conclusions

This paper has presented a literature review of techniques to adapt NIAs to the solution of CNOPs. Based on an updated and relatively brief taxonomy of approaches that has been introduced in previous surveys on this topic, the most representative constraint-handling techniques currently available were briefly discussed and analyzed. From the review presented here, the following conclusions can be drawn:

1. Although in the early days of this area, a wide variety of approaches were developed, today, there is clear trend to adopt and further develop a small set of approaches which have been found to be competitive in a variety of problems. Thus, we can say that exploitation is now being favored as opposed to the exploration that characterized the early days of this field.
2. Feasibility rules is the most popular constraint-handling technique in current use. This is due to its simplicity and effectiveness and in spite of the fact that it is prone (when it is not properly coupled to a NIA) to cause premature convergence. Overall, this approach has been found to provide a stable behavior for solving constrained optimization problems using NIAs, as reflected by a number of studies in that direction. This approach has also been commonly adopted when studying other topics related to constrained optimization (e.g., design of variation operators, parameter control, termination conditions, etc.), also because of its simplicity.
3. DE is the most popular search engine found in constrained optimization. This is due to its good and consistent performance and to its simplicity (normally, very few changes are required to make it work well in a wide variety of CNOPs).

4. SR remains as a highly competitive and relatively simple constraint-handling technique, and has been coupled to several NIAs, from which DE is the most popular choice and PSO is the less popular.
5. Regarding penalty functions, they still remain popular (particularly in engineering optimization). The most preferred approaches are those incorporating adaptive mechanisms to define the penalty factor values, and the most popular search engine is the GA. In this regard, the development of specialized variation operators that are specifically designed to deal with constraints may be not only interesting, but can also be a good alternative to reduce the computational cost typically required by most adaptation mechanisms.
6. Memetic algorithms for solving CNOPs are becoming more popular in the literature, even surpassing other hybrid approaches (i.e., those combining features of two or more different NIAs). In this regard, the development of new local search strategies that are specifically designed for constrained search spaces is a promising research topic.
7. Some special operators have been designed to be easier to generalize, making them more popular in the literature.
8. Several research topics need to be tackled or revisited. The design of constraint-handling techniques for dynamic and/or multi-objective problems, the use of fitness approximation and surrogate methods, the use of hyper-heuristics, the study of fitness landscapes in the presence of constraints, the development of theoretical foundations for the approaches in current use are only some of the topics that deserve more research in the years to come.

To finish, it is fair to state that constrained numerical optimization using NIAs is still an active area of research that offers lots of opportunities to both newcomers and established researchers. As such, this area is expected to continue growing in the following years not only in terms of number of publications but also in terms of the depth of the work being undertaken.

Acknowledgments

The first author acknowledges the support from CONACyT project no. 79809. The second author acknowledges the support from CONACyT project no. 103570.

References

[1] A. Eiben, J.E. Smith, Introduction to Evolutionary Computing, in: Natural Computing Series, Springer Verlag, 2003.
[2] J. Kennedy, R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann, UK, 2001.
[3] Z. Michalewicz, D.B. Fogel, How to Solve it: Modern Heuristics, 2nd ed., Springer, Germany, 2004.
[4] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, 1996.
[5] A.P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley & Sons, 2005.
[6] T.P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation 4 (2000) 284–294.
[7] Z. Michalewicz, M. Schoenauer, Evolutionary algorithms for constrained parameter optimization problems, Evolutionary Computation 4 (1996) 1–32.
[8] C.A.C. 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 (2002) 1245–1287.
[9] E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence, vol.198, Springer-Verlag, 2009.
[10] O. Kramer, A review of constraint-handling techniques for evolution strategies, Applied Computational Intelligence and Soft Computing (ISSN: 1666-6038) 2010 (2010) doi:10.1155/2010/185063. Article ID 185063, 11 pages.
[11] E. Mezura-Montes, J.I. Flores-Mendoza, Improved particle swarm optimization in constrained numerical search spaces, in: R. Chiong (Ed.), Nature-Inspired Algorithms for Optimization, in: Studies in Computational Intelligence Series, vol. 193, Springer-Verlag, ISBN: 978-3-540-72963-1, 2009, pp. 299–332.
[12] S. Salcedo-Sanz, A survey of repair methods used as constraint handling techniques in evolutionary algorithms, Computer Science Review 3 (2009) 175–192.
[13] A.E. Smith, D.W. Coit, Constraint handling techniques—penalty functions, in: T. Bäck, D.B. Fogel, Z. Michalewicz (Eds.), Handbook of Evolutionary Computation, Oxford University Press and Institute of Physics Publishing, 1997, C5.2-1–C5.2-6.
[14] T. Bäck, F. Hoffmeister, H.-P. Schwefel, A survey of evolution strategies, in: R.K. Belew, L.B. Booker (Eds.), Proceedings of the Fourth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, California, 1991, pp. 2–9.
[15] H.-P. Schwefel, Numerical Optimization of Computer Models, John Wiley & Sons, Great Britain, 1981.
[16] A. Kuri-Morales, C.V. Quezada, A universal eclectic genetic algorithm for constrained optimization, in: Proceedings 6th European Congress on Intelligent Techniques & Soft Computing, EUFIT'98, Verlag Mainz, Aachen, Germany, 1998, pp. 518–522.
[17] A. Homaifar, S.H.Y. Lai, X. Qi, Constrained optimization via genetic algorithms, Simulation 62 (1994) 242–254.
[18] F. Hoffmeister, J. Sprave, Problem-independent handling of constraints by use of metric penalty functions, in: L.J. Fogel, P.J. Angeline, T. Bäck (Eds.), Proceedings of the Fifth Annual Conference on Evolutionary Programming, EP'96, The MIT Press, San Diego, California, 1996, pp. 289–294.
[19] R.G.L. Riche, C. Knopf-Lenoir, R.T. Haftka, A segregated genetic algorithm for constrained structural optimization, in: L.J. Eshelman (Ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, ICGA-95, University of Pittsburgh, Morgan Kaufmann Publishers, San Mateo, California, 1995, pp. 558–565.
[20] J. Joines, C. Houck, On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs, in: D. Fogel (Ed.), Proceedings of the first IEEE Conference on Evolutionary Computation, IEEE Press, Orlando, Florida, 1994, pp. 579–584.
[21] S. Kazarlis, V. Petridis, Varying fitness functions in genetic algorithms: studying the rate of increase of the dynamic penalty terms, in: A.E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (Eds.), Proceedings of the 5th Parallel Problem Solving from Nature, PPSN V, Amsterdan, The Netherlands, in: Lecture Notes in Computer Science, vol. 1498, Springer-Verlag, Heidelberg, Germany, 1998, pp. 211–220.
[22] W.A. Crossley, E.A. Williams, A study of adaptive penalty functions for constrained genetic algorithm based optimization, in: AIAA 35th Aerospace Sciences Meeting and Exhibit, AIAA Paper 97-0083, Reno, Nevada.
[23] Z. Michalewicz, N.F. Attia, Evolutionary optimization of constrained problems, in: Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific, 1994, pp. 98–108.
[24] A.B. Hadj-Alouane, J.C. Bean, A genetic algorithm for the multiple-choice integer program, Operations Research 45 (1997) 92–101.
[25] K. Rasheed, An adaptive penalty approach for constrained genetic-algorithm optimization, in: J.R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D.B. Fogel, M.H. Garzon, D.E. Goldberg, H. Iba, R.L. Riolo (Eds.), Proceedings of the Third Annual Genetic Programming Conference, Morgan Kaufmann Publishers, San Francisco, California, 1998, pp. 584–590.
[26] H. Hamda, M. Schoenauer, Adaptive techniques for evolutionary topological optimum design, in: I. Parmee (Ed.), Proceedings of the Fourth International Conference on Adaptive Computing in Design and Manufacture, ACDM'2000, Springer-Verlag, University of Plymouth, Devon, UK, 2000, pp. 123–136.
[27] S.B. Hamida, M. Schoenauer, ASCHEA: new results using adaptive segregational constraint handling, in: Proceedings of the Congress on Evolutionary Computation 2002, CEC'2002, IEEE Service Center, Piscataway, New Jersey, vol. 1, 2002, pp. 884–889.
[28] H.J. Barbosa, A.C. Lemonge, An adaptive penalty scheme in genetic algorithms for constrained optimization problems, in: W. Langdon, et al. (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'2002, Morgan Kaufmann Publishers, San Francisco, California, 2002, pp. 287–294.
[29] C.A. Coello Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Computers in Industry 41 (2000) 113–127.
[30] B. Wu, X. Yu, Fuzzy penalty function approach for constrained function optimization with evolutionary algorithms, in: Proceedings of the 8th International Conference on Neural Information Processing, Fudan University Press, Shanghai, China, 2001, pp. 299–304.
[31] S. Koziel, Z. Michalewicz, A decoder-based evolutionary algorithm for constrained parameter optimization problems, in: T. Bäck, A.E. Eiben, M. Schoenauer, H.-P. Schwefel (Eds.), Proceedings of the 5th Parallel Problem Solving from Nature, PPSN V, Amsterdan, The Netherlands, in: Lecture Notes in Computer Science, vol. 1498, Springer-Verlag, Heidelberg, Germany, 1998, pp. 231–240.
[32] S. Koziel, Z. Michalewicz, Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization, Evolutionary Computation 7 (1999) 19–44.
[33] D.G. Kim, P. Husbands, Riemann mapping constraint handling method for genetic algorithms, Technical Report CSRP 469, COGS, University of Sussex, UK, 1997.
[34] D.G. Kim, P. Husbands, Landscape changes and the performance of mapping based constraint handling methods, in: A.E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (Eds.), Proceedings of the 5th Parallel Problem Solving from Nature, PPSN V, Amsterdan, The Netherlands, in: Lecture Notes in Computer Science, vol. 1498, Springer-Verlag, Heidelberg, Germany, 1998, pp. 221–230.

[35] D.G. Kim, P. Husbands, Mapping based constraint handling for evolutionary search: thurston's circle packing and grid generation, in: I. Parmee (Ed.), The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation, Springer-Verlag, Plymouth, United Kingdom, 1998, pp. 161–173.
[36] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, third ed., Springer-Verlag, 1996.
[37] M. Schoenauer, Z. Michalewicz, Evolutionary computation at the edge of feasibility, in: H.-M. Voigt, W. Ebeling, I. Rechenberg, H.-P. Schwefel (Eds.), Proceedings of the Fourth Conference on Parallel Problem Solving from Nature, PPSN IV, Berlin, Germany, Springer-Verlag, Heidelberg, Germany, 1996, pp. 245–254.
[38] Z. Michalewicz, G. Nazhiyath, Genocop III: a co-evolutionary algorithm for numerical optimization with nonlinear constraints, in: D.B. Fogel (Ed.), Proceedings of the Second IEEE International Conference on Evolutionary Computation, IEEE Press, Piscataway, New Jersey, 1995, pp. 647–651.
[39] R. Kowalczyk, Constraint consistent genetic algorithms, in: Proceedings of the 1997 IEEE Conference on Evolutionary Computation, IEEE, Indianapolis, USA, 1997, pp. 343–348.
[40] M. Schoenauer, Z. Michalewicz, Boundary operators for constrained parameter optimization problems, in: T. Bäck (Ed.), Proceedings of the Seventh International Conference on Genetic Algorithms, ICGA-97, Morgan Kaufmann, San Francisco, California, 1997, pp. 322–329.
[41] M. Schoenauer, Z. Michalewicz, Sphere operators and their applicability for constrained optimization problems, in: V. Porto, N. Saravanan, D. Waagen, A. Eiben (Eds.), Proceedings of the 7th International Conference on Evolutionary Programming, EP98, San Diego, California, USA, in: Lecture Notes in Computer Science, vol. 1447, Springer-Verlag, Heidelberg, Germany, 1998, pp. 241–250.
[42] D. Powell, M.M. Skolnick, Using genetic algorithms in engineering design optimization with non-linear constraints, in: S. Forrest (Ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, ICGA-93, University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, San Mateo, California, 1993, pp. 424–431.
[43] R. Hinterding, Z. Michalewicz, Your brains and my beauty: parent matching for constrained optimization, in: Proceedings of the 5th International Conference on Evolutionary Computation, Anchorage, Alaska, pp. 810–815.
[44] M. Schoenauer, S. Xanthakis, Constrained GA optimization, in: S. Forrest (Ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, ICGA-93, University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers, San Mateo, California, 1993, pp. 573–580.
[45] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2000) 311–338.
[46] E. Mezura-Montes, C.A. Coello Coello, E.I. Tun-Morales, Simple feasibility rules and differential evolution for constrained optimization, in: R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (Eds.), Proceedings of the 3rd Mexican International Conference on Artificial Intelligence, MICAI'2004, in: Lecture Notes in Artificial Intelligence, vol. 2972, Springer Verlag, Heidelberg, Germany, 2004, pp. 707–716.
[47] C.A. Coello Coello, Treating constraints as objectives for single-objective evolutionary optimization, Engineering Optimization 32 (2000) 275–308.
[48] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer with a novel constrain-handling mechanism, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 316–323.
[49] J.J. Liang, S. Zhigang, L. Zhihui, Coevolutionary comprehensive learning particle swarm optimizer, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1505–1512.
[50] Z. Li, J. Liang, X. He, Z. Shang, Differential evolution with dynamic constraint-handling mechanism, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1899–1906.
[51] S. Venkatraman, G.G. Yen, A generic framework for constrained optimization using genetic algorithms, IEEE Transactions on Evolutionary Computation 9 (2005).
[52] B. Liu, H. Ma, X. Zhang, B. Liu, Y. Zhou, A memetic co-evolutionary differential evolution algorithm for constrained optimization, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 2996–3002.
[53] E. Mezura-Montes, C.A. Coello Coello, Constrained optimization via multiobjective evolutionary algorithms, in: D.C. Joshua Knowles, K. Deb (Eds.), Multiobjective Problems Solving from Nature: From Concepts to Applications, in: Natural Computing Series, vol. 2008, Springer-Verlag, ISBN: 978-3-540-72963-1, 2008, pp. 53–76.
[54] T.P. Runarsson, X. Yao, Search biases in constrained evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews 35 (2005) 233–243.
[55] T. Ray, T. Kang, S.K. Chye, An evolutionary algorithm for constrained optimization, in: D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, H.-G. Beyer (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'2000, Morgan Kaufmann, San Francisco, California, 2000, pp. 771–777.
[56] T. Ray, K. Liew, A swarm with an effective information sharing mechanism for unconstrained and constrained single objective optimization problems, in: Proceedings of the Congress on Evolutionary Computation 2001, CEC'2001, IEEE Service Center, Piscataway, New Jersey, vol. 1, 2001, pp. 75–80.
[57] S. Akhtar, K. Tai, T. Ray, A socio-behavioural simulation model for engineering design optimization, Engineering Optimization 34 (2002) 341–354.
[58] A. Angantyr, J. Andersson, J.-O. Aidanpaa, Constrained optimization based on a multiobjective evolutionary algorithms, in: Proceedings of the Congress on Evolutionary Computation 2003, CEC'2003, Canberra, Australia, IEEE Service Center, Piscataway, New Jersey, vol. 3, 2003, pp. 1560–1567.
[59] A. Hernández-Aguirre, S. Botello-Rionda, C.A. Coello Coello, G. Lizárraga-Lizárraga, E. Mezura-Montes, Handling constraints using multiobjective optimization concepts, International Journal for Numerical Methods in Engineering 59 (2004) 1989–2017.
[60] T. Ray, K. Liew, Society and civilization: an optimization algorithm based on the simulation of social behavior, IEEE Transactions on Evolutionary Computation 7 (2003) 386–396.
[61] A.I. Oyman, K. Deb, H.-G. Beyer, An alternative constraint handling method for evolution strategies, in: Proceedings of the Congress on Evolutionary Computation 1999, CEC'99, IEEE Service Center, Piscataway, New Jersey, vol. 1, 1999, pp. 612–619.
[62] E. Mezura-Montes, C.A. Coello Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Transactions on Evolutionary Computation 9 (2005) 1–17.
[63] E. Mezura-Montes, J. Velázquez-Reyes, C.A.C. Coello, Promising infeasibility and multiple offspring incorporated to differential evolution for constrained optimization, in: H.-G. Beyer, U.-M. O'Reilly, D. Arnold, W. Banzhaf, C. Blum, E. Bonabeau, E. Cantú-Paz, D. Dasgupta, K. Deb, J. Foster, E. de Jong, H. Lipson, X. Llora, S. Mancoridis, M. Pelikan, G. Raidl, T. Soule, A. Tyrrell, J.-P. Watson, E. Zitzler (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'2005, Volume 1, Washington, DC, USA, ACM Press, New York, ISBN: 1-59593-010-8, 2005, pp. 225–232.
[64] E. Mezura-Montes, J. Velázquez-Reyes, C.A.C. Coello, Modified differential evolution for constrained optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 332–339.
[65] J. Lampinen, A constraint handling approach for the differential evolution algorithm, in: Proceedings of the Congress on Evolutionary Computation 2002, CEC'2002, IEEE Service Center, Piscataway, New Jersey, vol. 2, 2002, pp. 1468–1473.
[66] S. Kukkonen, J. Lampinen, Constrained real-parameter optimization with generalized differential evolution, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 911–918.
[67] A.L. Jaimes, C.A. Coello Coello, H. Aguirre, K. Tanaka, Adaptive objective space partitioning using conflict information for many-objective optimization, in: R. Takahashi, et al. (Eds.), 2011 Evolutionary Multicriterion Optimization Conference, EMO'2011, in: LNCS, vol. 6576, Springer-Verlag, Heidelberg, Germany, 2011, pp. 151–165.
[68] E. Mezura-Montes, A.G. Palomeque-Ortiz, Parameter control in differential evolution for constrained optimization, in: 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Tronheim, Norway, 2009, pp. 1375–1382.
[69] K. Zielinski, R. Laur, Constrained single-objective optimization using differential evolution, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 927–934.
[70] K. Zielinski, R. Laur, Stopping criteria for differential evolution in constrained single-objective optimization, in: U.K. Chakraborty (Ed.), Advances in Differential Evolution, Springer, Berlin, ISBN: 978-3-540-68827-3, 2008, pp. 111–138.
[71] K. Zielinski, X. Wang, R. Laur, Comparison of adaptive approaches for differential evolution, in: G. Rudolph, T. Jansen, S. Lucas, C. Poloni, N. Beume (Eds.), Parallel Problem Solving from Nature, PPSN X, in: Lecture Notes in Computer Science, vol. 5199, Springer, Dortmund, Germany, 2008, pp. 641–650.
[72] K. Zielinski, S.P. Vudathu, R. Laur, Influence of different deviations allowed for equality constraints on particle swarm optimization and differential evolution, in: N. Krasnogor, G. Nicosia, M. Pavone, D. Pelta (Eds.), Nature Inspired Cooperative Strategies for Optimization, Springer, Berlin, ISBN: 978-3-540-78986-4, 2008, pp. 249–259.
[73] V.L. Huang, A.K. Qin, P.N. Suganthan, Self-adaptative differential evolution algorithm for constrained real-parameter optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 324–331.
[74] J. Brest, V. Zumer, M.S. Maucec, Self-adaptative differential evolution algorithm in constrained real-parameter optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 919–926.
[75] R. Landa Becerra, C.A. Coello Coello, Cultured differential evolution for constrained optimization, Computer Methods in Applied Mechanics and Engineering 195 (2006) 4303–4322.
[76] A. Menchaca-Méndez, C.A. Coello Coello, A new proposal to hybridize the nelder-mead method to a differential evolution algorithm for constrained optimization, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 2598–2605.
[77] C. Luo, B. Yu, Low dimensional simplex evolution—a hybrid heuristic for global optimization, in: Proceedings of the Eight ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD'2007, IEEE Computer Society, Washington, DC, USA, 2007, pp. 470–474.

[78] A.S.S.M. Barkat Ullah, R. Sarker, D. Cornforth, Search space reduction technique for constrained optimization with tiny feasible space, in: 2008 Genetic and Evolutionary Computation Conference, GECCO'2008, ACM Press, Atlanta, USA, ISBN: 978-1-60558-131-6, 2008, pp. 881–888.
[79] K. Zielinski, R. Laur, Constrained single-objective optimization using particle swarm optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 1550–1557.
[80] C.-L. Sun, J.-C. Zeng, J.-S. Pan, A particle swarm optimization with feasibility-based rules for mixed-variable optimization problems, in: J.-S. Pan, J. Liu, A. Abraham (Eds.), Ninth International Conference on Hybrid Intelligent Systems, 2009, HIS'09, IEEE Computer Society Press, Shenyang, China, 2009, pp. 543–547.
[81] C.-L. Sun, J.-C. Zeng, J.-S. Pan, An improved particle swarm optimization with feasibility-based rules for mixed-variable optimization problems, in: Fourth International Conference on Innovative Computing, Information and Control, 2009, IEEE Computer Society Press, 2009, pp. 897–903.
[82] Q. He, L. Wang, A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization, Applied Mathematics and Computation 186 (2007) 1407–1422.
[83] G. Toscano-Pulido, C.A. Coello Coello, A constraint-handling mechanism for particle swarm optimization, in: Proceedings of the Congress on Evolutionary Computation 2004, CEC'2004, Portland, Oregon, USA, IEEE Service Center, Piscataway, New Jersey, vol. 2, 2004, pp. 1396–1403.
[84] A.-E. Muñoz-Zavala, A.H. Aguirre, E.R.V. Diharce, Constrained optimization via particle evolutionary swarm optimization algorithm (PESO), in: H.-G. Beyer, et al. (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'2005, Volume 1, Washington DC, USA, ACM Press, New York, ISBN: 1-59593-010-8, 2005, pp. 209–216.
[85] A.E. Muñoz-Zavala, A. Hernández-Aguirre, E.R. Villa-Diharce, S. Botello-Rionda, PESO+ for constrained optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 935–942.
[86] A. Muñoz Zavala, A. Hernández Aguirre, E. Villa Diharce, Robust PSO-based constrained optimization by perturbing the particle's memory, in: F.T. Chan, M.K. Tiwari (Eds.), Swarm Intelligence. Focus on Ant and Particle Swarm Optimization, I-Tech Education and Publising, Croatia, 2007, pp. 57–76.
[87] A.E. Muñoz Zavala, A. Hernández Aguirre, E.R. Villa Diharce, S. Botello Rionda, Constrained optimization with an improved particle swarm optimization algorithm, International Journal of Intelligent Computing and Cybernetics 1 (2008) 425–453.
[88] L.C. Cagnina, S.C. Esquivel, C.A.C. Coello, A particle swarm optimizer for constrained numerical optimization, in: T.P. Runarsson, H.-G. Beyer, E. Burke, J.J. Merelo-Guervós, L.D. Whitley, X. Yao (Eds.), Parallel Problem Solving from Nature, PPSN IX, 9th International Conference, Reykjavik, Iceland, in: Lecture Notes in Computer Science, vol. 4193, Springer, Reykjavik, Iceland, 2006, pp. 910–919.
[89] L. Cagnina, S. Esquivel, C.A. Coello Coello, A bi-population PSO with a shake-mechanism for solving constrained numerical optimization, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 670–676.
[90] C. gang Cui, Y. jun Li, T. jun Wu, A relative feasibility degree based approach for constrained optimization problems, Journal of Zhejiang University-Science C-Computers & Electronics 11 (2010) 249–260.
[91] D. Karaboga, B. Basturk, Artificial bee colony(ABC) optimization algorithm for solving constrained optimization problems, in: P. Melin, O. Castillo, L.T. Aguilar, J. Kacptrzyk, W. Pedrycz (Eds.), Foundations of Fuzzy Logic and Soft Computing, 12th International Fuzzy Systems Association, World Congress, IFSA 2007, in: Lecture Notes in Artificial Intelligence, vol. 4529, Springer, Cancun, Mexico, 2007, pp. 789–798.
[92] D. Karaboga, B. Akay, A modified artificial bee colony (ABC) algorithm for constrained optimization problems, Applied Soft Computing 11 (2011) 3021–3031.
[93] E. Mezura-Montes, O. Cetina-Domínguez, Exploring promising regions of the search space with the scout bee in the artificial bee colony for constrained optimization, in: C.H. Dagli, et al. (Eds.), Proceedings of the Artificial Neural Networks in Enginnering Conference, ANNIE'2009, in: Intelligent Engineering Systems Through Artificial Neural Networks, vol. 19, ASME Press, St. Louis, MO, USA, 2009, pp. 253–260.
[94] E. Mezura-Montes, R.E. Velez-Koeppel, Elitist artificial bee colony for constrained real-parameter optimization, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 2068–2075.
[95] E. Mezura-Montes, B. Hernández-Ocaña, Modified bacterial foraging optimization for engineering design, in: C.H. Dagli, et al. (Eds.), Proceedings of the Artificial Neural Networks in Enginnering Conference, ANNIE'2009, in: Intelligent Engineering Systems Through Artificial Neural Networks, vol. 19, ASME Press, St. Louis, MO, USA, 2009, pp. 357–364.
[96] K. Passino, Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Systems Magazine 22 (2002) 52–67.
[97] E. Mezura-Montes, M.E. Miranda-Varela, R. del Carmen Gómez-Ramón, Differential evolution in constrained numerical optimization. An empirical study, Information Sciences 180 (2010) 4223–4262.
[98] S.M. Elsayed, R.A. Sarker, D.L. Essam, Multi-operator based evolutionary algorithms for solving constrained optimization problems, Computers & Operations Research 38 (2011) 1877–1896.
[99] S. Elsayed, R. Sarker, D. Essam, Ga with a new multi-parent crossover for constrained optimization, in: 2011 IEEE Congress on Evolutionary Computation, CEC'2011, IEEE Press, New Orleans, USA, 2011, pp. 857–864.
[100] S. Elsayed, R. Sarker, D. Essam, A comparative study of different variants of genetic algorithms for constrained optimization, in: 2010 International Conference on Simulated Evolution and Learning, SEAL'2010, in: LNCS, vol. 6457, Springer, Kanpur, India, 2010, pp. 177–186.
[101] N. Hamza, S. Elsayed, D. Essam, R. Sarker, Differential evolution combined with constraint consensus for constrained optimization, in: 2011 IEEE Congress on Evolutionary Computation, CEC'2011, IEEE Press, New Orleans, USA, 2011, pp. 865–871.
[102] J. Tvrdík, R. Poláková, Competitive differential evolution for constrained problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1632–1639.
[103] A. Saha, R. Datta, K. Deb, Hybrid gradient projection based genetic algorithms for constrained optimization, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 2851–2858.
[104] L.-Y. Tseng, C. Chen, Multiple trajectory search for single objective constrained real-parameter optimization problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 3433–3439.
[105] Y. Wang, Z. Cai, A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems, Frontiers of Computer Science in China 3 (2009) 38–52.
[106] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Applied Soft Computing 10 (2010) 629–640.
[107] N. Cruz-Cortés, Handling constraints in global optimization using artificial immune systems, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence Series, vol. 198, Springer-Verlag, ISBN: 978-3-642-00618-0, 2009, pp. 237–262.
[108] P. Hajela, J. Lee, Constrained genetic search via schema adaptation. An immune network solution, Structural Optimization 12 (1996) 11–15.
[109] C.A.C. Coello, N.C. Cortés, A parallel implementation of an artificial immune system to handle constraints in genetic algorithms: preliminary results, in: Proceedings of the Congress on Evolutionary Computation 2002, CEC'2002, IEEE Service Center, Piscataway, New Jersey, vol. 1, 2002, pp. 819–824.
[110] H. Bernardino, H. Barbosa, A. Lemong, A hybrid genetic algorithm for constrained optimization problems in mechanical engineering, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 646–653.
[111] N. Cruz-Cortés, D. Trejo-Pérez, C.A.C. Coello, Handling constraints in global optimization using an artificial immune system, in: C. Jacob, M.L. Pilat, P.J. Bentley, J. Timmis (Eds.), Artificial Immune Systems. 4th International Conference, ICARIS 2005, in: Lecture Notes in Computer Science, vol. 3627, Springer, Banff, Canada, 2005, pp. 234–247.
[112] V.S. Aragón, S.C. Esquivel, C.A. Coello Coello, Artificial immune system for solving constrained optimization problems, Revista Iberoamericana de Inteligencia Artificial 11 (2007) 55–66.
[113] L. Nunes de Castro, F.V. Zuben, Learning and optimization using the clonal selection principle, IEEE Transactions on Evolutionary Computation 6 (2002) 239–251.
[114] V.S. Aragón, S.C. Esquivel, C.A. Coello Coello, A novel model of artificial immune system for solving constrained optimization problems with dynamic tolerance factor, in: A. Gelbukh, Ángel Fernando Kuri Morales (Eds.), Proceedings of the Mexican International Conference on Artificial Intelligence, MICAI'2007, in: Lecture Notes in Artificial Intelligence, vol. 4827, Springer, Aguascalientes, México, 2007, pp. 19–29.
[115] J. Liu, W. Zhong, L. Hao, An organizational evolutionary algorithm for numerical optimization, IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics) 37 (2007) 1052–1064.
[116] J. Sun, J.M. Garibaldi, A novel memetic algorithm for constrained optimization, in: 2010 IEEE Congress on Evolutionary Computation, CEC'2010, IEEE Press, Barcelona, Spain, 2010, pp. 549–556.
[117] A.S.S.M. Barkat Ullah, R. Sarker, D. Cornforth, C. Lokan, An agent-based memetic algorithm (AMA) for solving constrainted optimization problems, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 999–1006.
[118] A.S.S.M. Barkat Ullah, R. Sarker, D. Cornforth, C. Lokan, Ama: a new approach for solving constrained real-valued optimization problems, Soft Computing 13 (2008) 741–762.
[119] H. Ma, D. Simon, Blended biogeography-based optimization for constrained optimization, Engineering Applications of Artificial Intelligence 24 (2011) 517–525.
[120] M.M. Ali, Z. Kajee-Bagdadi, A local exploration-based differential evolution algorithm for constrained global optimization, Applied Mathematics and Computation 208 (2009) 31–48.
[121] E. Mezura-Montes, M. Damian-Araoz, O. Cetina-Domínguez, Smart flight and dynamic tolerances in the artificial bee colony for constrained optimization, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 4118–4125.
[122] S. Elsayed, R. Sarker, D. Essam, Integrated strategies differential evolution algorithm with a local search for constrained optimization, in: 2011 IEEE Congress on Evolutionary Computation, CEC'2011, IEEE Press, New Orleans, USA, 2011, pp. 2618–2625.

[123] M. Zhang, W. Luo, X. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, Information Sciences 178 (2008) 3043–3074.
[124] R. Liu, Y. Li, W. Zhang, L. Jiao, Stochastic ranking based differential evolution algorithm for constrained optimization problem, in: 2009 ACM SIGEVO Summit on Genetic and Evolutionary Computation, GEC'2009, ACM Press, Shanghai, China, ISBN: 978-1-60558-326-6, 2009, pp. 887–890.
[125] J. Liu, Z. Fan, E.D. Goodman, SRaDE: an adaptive differential evolution based on stochastic ranking, in: 2009 Genetic and Evolutionary Computation Conference, GECCO'2009, ACM Press, Montreal, Canada, ISBN: 978-1-60558-325-9, 2009, pp. 1871–1872.
[126] G. Leguizamón, C.A. Coello Coello, A boundary search based ACO algorithm coupled with stochastic ranking, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 165–172.
[127] T.P. Runarsson, Constrained evolutionary optimization by approximate ranking and surrogate models, in: X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tiňo, A. Kabán, H.-P. Schwefel (Eds.), Proceedings of 8th Parallel Problem Solving from Nature, PPSN VIII, Birmingham, UK, in: Lecture Notes in Computer Science, vol. 3242, Springer-Verlag, Heidelberg, Germany, 2004, pp. 401–410.
[128] M. Zhang, H. Geng, W. Luo, L. Huang, X. Wang, A novel search biases selection strategy for constrained evolutionary optimization, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 6736–6741.
[129] R. Mallipeddi, P. Suganthan, B. Qu, Diversity enhanced adaptive evolutionary programming for solving single objective constrained problems, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 2106–2113.
[130] Z. Fan, J. Liu, T. Sorensen, P. Wang, Improved differential evolution based on stochastic ranking for robust layout synthesis of mems components, IEEE Transactions on Industrial Electronics 56 (2009) 937–948.
[131] G. Huan-Tong, S. Qing-Xi, J. Feng, S. Yi-Jie, An evolution strategy with stochastic ranking for solving reactive power optimization, in: 2009 2nd International Conference on Power Electronics and Intelligent Transportation System, PEITS'2009, IEEE Press, Shenzhen, ISBN: 978-1-4244-4544-8, 2009, pp. 14–17.
[132] L. Fonseca, P. Capriles, H. Barbosa, A. Lemonge, A stochastic rank-based ant system for discrete structural optimization, in: Proceedings of the 2007 IEEE Swarm Intelligence Symposium, SIS 2007, IEEE Press, Honolulu, Hawaii, USA, 2007, pp. 68–75.
[133] T. Takahama, S. Sakai, N. Iwane, Constrained optimization by the epsilon constrained hybrid algorithm of particle swarm optimization and genetic algorithm, in: AI 2005: Advances in Artificial Intelligence, in: Lecture Notes in Artificial Intelligence, vol. 3809, Springer-Verlag, 2005, pp. 389–400.
[134] T. Takahama, S. Sakai, Constrained optimization by α constrained genetic algorithm (α GA), Systems and Computers in Japan 35 (2004) 11–22.
[135] T. Takahama, S. Sakai, Constrained optimization by applying the α constrained method to the nonlinear simplex method with mutations, IEEE Transactions on Evolutionary Computation 9 (2005) 437–451.
[136] T. Takahama, S. Sakai, Constrained optimization by the ϵ constrained differential evolution with gradient-based mutation and feasible elites, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 308–315.
[137] L. Wang, L. po Li, An effective differential evolution with level comparison for constrained engineering design, Structural and Multidisciplinary Optimization 41 (2010) 947–963.
[138] T. Takahama, S. Sakai, Constrained optimization by constrained particle swarm optimizer with—level control, in: Proceedings of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology, WSTST05, Muroran, Japan, pp. 1019–1029.
[139] J.J. Liang, T. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C.A. Coello Coello, K. Deb, Problem definitions and evaluation criteria for the CEC 2006, special session on constrained real-parameter optimization, Technical Report, Nanyang Technological University, Singapore, December, 2005. Available at: http://www.lania.mx/~emezura.
[140] T. Takahama, S. Sakai, Constrained optimization by ϵ constrained differential evolution with dynamic ϵ-level control, in: U.K. Chakraborty (Ed.), Advances in Differential Evolution, Springer, Berlin, ISBN: 978-3-540-68827-3, 2008, pp. 139–154.
[141] T. Takahama, S. Sakai, Solving difficult constrained optimization problems by the ϵ constrained differential evolution with gradient-based mutation, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence Series, vol. 198, Springer-Verlag, ISBN: 978-3-642-00618-0, 2009, pp. 51–72.
[142] T. Takahama, S. Sakai, Constrained optimization by the ε-constrained differential evolution with an archive and gradient-based mutation, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1680–1688.
[143] E. Mezura-Montes, C.A. Coello Coello, J. Velázquez-Reyes, L. Muñoz-Dávila, Multiple trial vectors in differential evolution for engineering design, Engineering Optimization 39 (2007) 567–589.
[144] J. Brest, V. Žumer, M.S. Maučecc, Control parameters in self-adaptive differential evolution, in: B. Filipič J. Šilc (Eds.), Bioinspired Optimization Methods and their Applications, 2006, Jožef Stefan Institute, Ljubljana, Slovenia, pp. 35–44.
[145] J. Brest, Constrained real-parameter optimization with ϵ-self-adaptive differential evolution, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence Series, vol. 198, Springer-Verlag, ISBN: 978-3-642-00618-0, 2009, pp. 73–93.
[146] J. Brest, B. Bošković, V. Žumer, An improved self-adaptive differential evolution algorithm in single objective constrained real-parameter optimization, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1073–1078.
[147] S. Zeng, H. Shi, H. Li, G. Chen, L. Ding, L. Kang, A lower-dimensional-search evolutionary algorithm and its application in constrained optimization problem, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 1255–1260.
[148] Q. Zhang, S. Zeng, R. Wang, H. Shi, G. Chen, L. Ding, L. Kang, Constrained optimization by the evolutionary algorithm with lower dimensional crossover and gradient-based mutation, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 273–279.
[149] J. Xiao, J. Xu, Z. Shao, C. Jiang, L. Pan, A genetic algorithm for solving multi-constrained function optimization problems based on KS function, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 4497–4501.
[150] K. Deb, R. Datta, A fast and accurate solution of constrained optimization problems using a hybrid bi-objective and penalty function approach, in: 2010 IEEE Congress on Evolutionary Computation, CEC'2010, IEEE Press, Barcelona, Spain, 2010, pp. 165–172.
[151] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA–II, IEEE Transactions on Evolutionary Computation 6 (2002) 182–197.
[152] R. Datta, K. Deb, A bi-objective based hybrid evolutionary-classical algorithm for handling equality constraints, in: R. Takahashi, et al. (Eds.), 2011 Evolutionary Multicriterion Optimization Conference, EMO'2011, in: LNCS, vol. 6576, Springer-Verlag, Heidelberg, Germany, 2011, pp. 313–327.
[153] M.F. Tasgetiren, P.N. Suganthan, A multi-populated differential evolution algorithm for solving constrained optimization problem, in: 2006 IEEE Congress on Evolutionary Computation, CEC'2006, IEEE, Vancouver, BC, Canada, 2006, pp. 340–354.
[154] R. Farmani, J.A. Wright, Self-adaptive fitness formulation for constrained optimization, IEEE Transactions on Evolutionary Computation 7 (2003) 445–455.
[155] S. Puzzi, A. Carpinteri, A double-multiplicative dynamic penalty approach for constrained evolutionary optimization, Structural and Multidisciplinary Optimization 35 (2008) 431–445.
[156] B. Tessema, G.G. Yen, An adaptive penalty formulation for constrained evolutionary optimization, IEEE Transactions on Systems, Man & Cybernetics, Part A (Systems & Humans) 39 (2009).
[157] A. Mani, C. Patvardhan, A novel hybrid constraint-handling technique for evolutionary optimization, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 2577–2583.
[158] Q. He, L. Wang, F.-Z. Huang, Nonlinear constrained optimization by enhanced co-evolutionary PSO, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 83–89.
[159] J.-Y. Wu, Solving constrained global optimization via artificial immune system, International Journal on Artificial Intelligence Tools 20 (2011) 1–27.
[160] G. Leguizamón, C.A.C. Coello, Boundary search for constrained numerical optimization problems with an algorithm inspired on the ant colony metaphor, IEEE Transactions on Evolutionary Computation 13 (2009) 350–368.
[161] F.-Z. Huang, L. Wang, Q. He, A hybrid differential evolution with double populations for constrained optimization, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 18–25.
[162] E.F. Wanner, F.G. Guimars, R.H. Takahashi, R.R. Saldanha, P.J. Fleming, Constraint quadratic approximation operator for treating equality constraints with genetic algorithms, in: 2005 IEEE Congress on Evolutionary Computation, CEC'2005, vol. 3, IEEE Press, Edinburgh, Scotland, 2005, pp. 2255–2262.
[163] G. Peconick, E.F. Wanner, R.H.C. Takahashi, Projection-based local search operator for multiple equality constraints within genetic algorithms, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 3043–3049.
[164] M.C. Araujo, E.F. Wanner, F.G.G. Aes, R.H.C. Takahashi, Constrained optimization based on quadratic approximations in genetic algorithms, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Computation, in: Studies in Computational Intelligence, vol. 198, Springer, Berlin, ISBN: 978-3-642-00618-0, 2009, pp. 193–217.
[165] A.S.B. Ullah, R. Sarker, C. Lokan, An agent-based memetic algorithm (AMA) for nonlinear optimization with equality constraints, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 70–77.
[166] H. Lu, W. Chen, Self-adaptive velocity particle swarm optimization for solving constrained optimization problems, Journal of Global Optimization 41 (2008) 427–445.
[167] M. Spadoni, L. Stefanini, Handling box, linear and quadratic-convex constraints for boundary optimization with differential evolution algorithms, in: Proceedings of the Ninth International Conference on Intelligence Systems Design and Applications, ISDA'2009, IEEE Computer Society, 2009, pp. 7–12.

[168] Y. Wu, Y. Li, X. Xu, A novel component-based model and ranking strategy in constrained evolutionary optimization, in: R. Huang, Q. Yang, J. Pei, J. Gama, X. Meng, X. Li (Eds.), Advanced Data Mining and Applications, 5th International Conference, ADMA'2009, in: Lecture Notes in Computer Science, vol. 5678, Springer, Beijing, China, 2009, pp. 362–373.
[169] S. Li, Y. Li, A new self-adaption differential evolution algorithm based component model, in: Advances in Computation and Intelligence, in: Lecture Notes in Computer Science, vol. 6362, Springer, 2010, pp. 54–63.
[170] A. Isaacs, T. Ray, W. Smith, Blessings of maintaining infeasible solutions for constrained multi-objective optimization problems, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 2785–2792.
[171] T. Ray, H.K. Singh, A. Isaacs, W. Smith, Infeasibility driven evolutionary algorithm for constrained optimization, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence Series, vol. 198, Springer-Verlag, ISBN: 978-3-642-00618-0, 2009, pp. 145–165.
[172] H.K. Singh, T. Ray, W. Smith, Performance of infeasibility empowered memetic algorithm for CEC 2010 constrained optimization problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 3770–3777.
[173] G. Reynoso-Meza, X. Blasco, J. Sanchis, M. Martínez, Multiobjective optimization algorithm for solving constrained single objective problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 3418–3424.
[174] Y. Wang, Z. Cai, G. Guo, Y. Zhou, Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems, IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics) 37 (2007) 560–575.
[175] Y. Wang, H. Liu, Z. Cai, Y. Zhou, An orthogonal design based constrained evolutionary optimization algorithm, Engineering Optimization 39 (2007) 715–736.
[176] H. Mühlenbein, D. Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I: continuous parameter optimization, Evolutionary Computation 1 (1993) 25–49.
[177] Y. Wang, Z. Cai, Y. Zhou, W. Zeng, An adaptive tradeoff model for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation 12 (2008) 80–92.
[178] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Structural and Multidisciplinary Optimization 37 (2009) 395–413.
[179] Y. Wang, Z. Cai, Y. Zhou, Accelerating adaptive trade-off model using shrinking space technique for constrained evolutionary optimization, International Journal for Numerical Methods in Engineering 77 (2009) 1501–1534.
[180] Y. Wang, Z. Cai, Hybrid differential evolution and adaptive trade-off model to solve constrained optimization problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 2846–2850.
[181] W. Gong, Z. Cai, A multiobjective differential evolution algorithm for constrained optimization, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 181–188.
[182] M. Laumanns, L. Thiele, K. Deb, E. Zitzler, Combining convergence and diversity in evolutionary multi-objective optimization, Evolutionary Computation 10 (2002) 263–282.
[183] L.D. Li, X. Li, X. Yu, A multi-objective constraint-handling method with PSO algorithm for constrained engineering optimization problems, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 1528–1535.
[184] G. Venter, R.T. Haftka, Constrained particle searm optimization using a bi-objective formulation, Structural and Multidisciplinary Optimization 40 (2010) 65–76.
[185] Q. Wang, X.-Z. Gao, C. Wang, An adaptive bacterial foraging algorithm for constrained optimization, International Journal of Innovative Computing Information and Control 6 (2010) 3585–3593.
[186] Manuel López-Ibáñez, Thomas Stützle, Alternative fitness assignment methods for many-objective optimization problems, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO'2010, ACM Press, Portland, Oregon, USA, 2010, pp. 71–78.
[187] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1997) 67–82.
[188] R. Mallipeddi, P.N. Suganthan, Ensemble of constraint handling techniques, IEEE Transactions on Evolutionary Computation 14 (2010) 561–579.
[189] R. Mallipeddi, P. Suganthan, Differential evolution with ensemble of constraint handling techniques for solving CEC 2010 benchmark problems, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 1907–1914.
[190] M. Tasgetiren, P. Suganthan, Q. Pan, R. Mallipeddi, S. Sarman, An ensemble of differential evolution algorithms for constrained function optimization, in: 2010 Congress on Evolutionary Computation, CEC'2010, IEEE Service Center, Barcelona, Spain, 2010, pp. 967–975.
[191] F.G. Lobo, C.F. Lima, Z. Michalewicz (Eds.), Parameter Setting in Evolutionary Algorithms, Springer, Berlin, Germany, 2007.
[192] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, in: Natural Computing Series, Springer-Verlag, 2005.
[193] D. Karaboga, B. Basturk, Powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Journal of Global Optimization 39 (2007) 459–471.
[194] G. Bilchev, I.C. Parmee, The ant colony metaphor for searching continuous design spaces, in: T.C. Fogarty (Ed.), Evolutionary Computing. AISB Workshop. Selected Papers, in: Lecture Notes in Computer Science, vol. 993, Springer-Verlag, Sheffield, UK, 1995, pp. 25–39.
[195] H.S. Bernardino, H.J.C. Barbosa, A.C.C. Lemonge, L.G. Fonseca, A new hybrid AIS-GA for constrained optimization problems in mechanical engineering, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 1455–1462.
[196] G.G. Yen, An adaptive penalty function for handling constraint in multi-objective evolutionary optimization, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Optimization, in: Studies in Computational Intelligence Series, vol. 198, Springer-Verlag, ISBN: 978-3-642-00618-0, 2009, pp. 121–143.
[197] C.A. Coello Coello, A.D. Christiansen, MOSES: a multiobjective optimization tool for engineering design, Engineering Optimization 31 (1999) 337–368.
[198] D. Chafekar, J. Xuan, K. Rasheed, Constrained multi-objective optimization using steady state genetic algorithms, in: E. Cantú-Paz, et al. (Eds.), Genetic and Evolutionary Computation—GECCO 2003. Proceedings, in: Lecture Notes in Computer Science, vol. 2723, Springer, 2003, pp. 813–824. Part I.
[199] Y.G. Woldesenbet, G.G. Yen, B.G. Tessema, Constraint handling in multiobjective evolutionary optimization, IEEE Transactions on Evolutionary Computation 13 (2009) 514–525.
[200] C.M. Fonseca, P.J. Fleming, Multiobjective optimization and multiple constraint handling with evolutionary algorithms I: a unified formulation, Technical Report 564, University of Sheffield, Sheffield, UK, 1995.
[201] H. Geng, M. Zhang, L. Huang, X. Wang, Infeasible elitists and stochastic ranking selection in constrained evolutionary multi-objective optimization, in: T.-D. Wang, X. Li, S.-H. Chen, X. Wang, H. Abbass, H. Iba, G. Chen, X. Yao (Eds.), Simulated Evolution and Learning, 6th International Conference, SEAL 2006, in: Lecture Notes in Computer Science, vol. 4247, Springer, Hefei, China, 2006, pp. 336–344.
[202] A. Oyama, K. Shimoyama, K. Fujii, New constraint-handling method for multi-objective and multi-constraint evolutionary optimization, Transactions of the Japan Society for Aeronautical and Space Sciences 50 (2007) 56–62.
[203] F. Jiménez, A.F. Gómez-Skarmeta, G. Sánchez, K. Deb, An evolutionary algorithm for constrained multi-objective optimization, in: Proceedings of the Congress on Evolutionary Computation 2002, CEC'2002, IEEE Service Center, Piscataway, New Jersey, vol. 2, 2002, pp. 1133–1138.
[204] K. Harada, J. Sakuma, I. Ono, S. Kobayashi, Constraint-handling method for multi-objective function optimization: Pareto descent repair operator, in: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007, in: Lecture Notes in Computer Science, vol. 4403, Springer, Matshushima, Japan, 2007, pp. 156–170.
[205] N. Young, Blended ranking to cross infeasible regions in constrained multi-objective problems, in: Proceedings of the 2005 International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, CIMCA-IAWTIC'05, IEEE Press, 2005, pp. 191–196.
[206] H.K. Singh, T. Ray, W. Smith, C-PSA: constrained Pareto simulated annealing for constrained multi-objective optimization, Information Sciences 180 (2010) 2499–2513.
[207] B. Qu, P. Suganthan, Constrained multi-objective optimization algorithm with ensemble of constraint-handling methods, Engineering Optimization 43 (2011) 403–416.
[208] D.K. Saxena, K. Deb, Trading on infeasibility by exploiting constraint's criticality through multi-objectivization: a system design perspective, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 919–926.
[209] D.K. Saxena, K. Deb, Dimensionality reduction of objectives and constraints in multi-objective optimization problems: a system design perspective, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 3203–3210.
[210] D.K. Saxena, T. Ray, K. Deb, A. Tiwari, Constrained many-objective optimization: a way forward, in: 2009 IEEE Congress on Evolutionary Computation, CEC'2009, IEEE Press, Trondheim, Norway, 2009, pp. 545–552.
[211] Y. Jin, A comprehensive survey of fitness approximation in evolutionary computation, Soft Computing—A Fusion of Foundations, Methodologies and Applications 9 (2005) 3–12.
[212] E.F. Wanner, F.G. Guimaraes, R.H.C. Takahashi, P.J. Flemming, Local search with quadratic approximation in genetic algorithms for expensive optimization problems, in: 2007 IEEE Congress on Evolutionary Computation, CEC'2007, IEEE Press, Singapore, 2007, pp. 677–683.
[213] E. Mezura-Montes, L.M. noz Dávila, C.A. Coello Coello, A preliminary study of fitness inheritance in evolutionary constrained optimization, in: N. Krasnogor, G. Nicosia, M. Pavone, D. Pelta (Eds.), Nature Inspired Cooperative Strategies for Optimization, Springer, Berlin, ISBN: 978-3-540-78986-4, 2008, pp. 1–14.
[214] S. Yang, Y.-S. Ong, Y. Jin, Evolutionary Computation in Dynamic and Uncertain Environments Evolutionary Computation in Dynamic and Uncertain Environments, Springer, 2007.
[215] T.T. Nguyen, X. Yao, Benchmarking and solving dynamic constrained problems, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 690–697.

[216] H. Kumar-Singh, A. Isaacs, T. Thanh-Nguyen, T. Ray, X. Yao, Performance of infeasibility driven evolutionary algorithm (idea) on constrained dynamic single objective optimization problems, in: IEEE 2009 Congress on Evolutionary Computation, CEC'2009, IEEE Service Center, Trondheim, Norway, 2009, pp. 3127–3134.
[217] E.K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, J.R. Woodward, A classification of hyper-heuristic approaches, in: M. Gendreau, J.-Y. Potvin (Eds.), Handbook of Metaheuristics, in: International Series in Operations Research & Management Science, Springer, Heidelberg, Germany, 2010, pp. 449–468.
[218] J.C. Villela-Tinoco, Control and selection mechanism of an hyperheuristic based on differential evolution for optimization in constrained search spaces, Master's Thesis, Dept. of Computer Science, CINVESTAV-IPN, Mexico City, 2001 (in Spanish).
[219] Y. Zhou, J. He, A runtime analysis of evolutionary algorithms for constrained optimization problems, IEEE Transactions on Evolutionary Computation 11 (2007) 608–619.
[220] Y. Yu, Z.-H. Zhou, On the usefulness of infeasible solutions in evolutionary search: a theoretical study, in: 2008 Congress on Evolutionary Computation, CEC'2008, IEEE Service Center, Hong Kong, 2008, pp. 835–840.
[221] D.V. Arnold, D. Brauer, On the behaviour of the $(1 + 1)$-ES for a simple constrained problem, in: G. Rudolph, T. Jansen, S. Lucas, C. Poloni, N. Beume (Eds.), Parallel Problem Solving from Nature, PPSN X, in: Lecture Notes in Computer Science, vol. 5199, Springer, Dortmund, Germany, 2008, pp. 1–10.
[222] D. Arnold, On the behaviour of the $(1, \lambda)$-ES for a simple constrained problem, in: Proceedings of the 2011 ACM/SIGEVO Foundations of Genetic Algorithms, FOGA'2011, ACM Press, 2011, pp. 15–24.