Study of the sequential constraint-handling technique for evolutionary optimization with application to structural problems

Motte, Damien; Nordin, Axel; Bjärnemo, Robert

Published in:
Proceedings of the 37th Design Automation Conference - DETC/DAC'11

DOI:
10.1115/DETC2011-47057

2011

Document Version:
Publisher's PDF, also known as Version of record

Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/DTM 2011
August 29-31, 2011, Washington, DC, USA

DETC2011-47057

STUDY OF THE SEQUENTIAL CONSTRAINT-HANDLING TECHNIQUE FOR EVOLUTIONARY OPTIMIZATION WITH APPLICATION TO STRUCTURAL PROBLEMS

Damien Motte[†], Axel Nordin*[†], Robert Bjärnemo
Division of Machine Design
Department of Design Sciences
Lund University
Box 118, 221 00 Lund
Sweden
Email: axel.nordin@mkon.lth.se,
damien.motte@mkon.lth.se,
robert.bjarnemo@mkon.lth.se

ABSTRACT

Engineering design problems are most frequently characterized by constraints that make them hard to solve and time-consuming. When evolutionary algorithms are used to solve these problems, constraints are often handled with the generic weighted sum method or with techniques specific to the problem at hand. Most commonly, all constraints are evaluated at each generation, and it is also necessary to fine-tune different parameters in order to receive good results, which requires in-depth knowledge of the algorithm. The sequential constraint-handling techniques seem to be a promising alternative, because they do not require all constraints to be evaluated at each iteration and they are easy to implement. They nevertheless require the user to determine the ordering in which those constraints shall be evaluated. Therefore two heuristics that allow finding a satisfying constraint sequence have been developed. Two sequential constraint-handling techniques using the heuristics have been tested against the weighted sum technique with the ten-bar structure benchmark. They both performed better than the weighted sum technique and can therefore be easy to implement, and powerful alternatives for solving engineering design problems.

INTRODUCTION

It is now common for complex engineering optimization problems to use evolutionary computational algorithms, such as genetic algorithms (GA, see e.g. [1] for a review). A characteristic of engineering optimization problems is that the evaluation of engineering constraints is time-intensive [2]. For an analysis related to structural or thermodynamic problems, finite element techniques may for instance be required. Despite that specificity, constraint handling is an issue that has been seldom addressed in the engineering design field. Most of the constraint handling techniques (CHT) used by evolutionary algorithms require an evaluation of all constraints at each generation, and it is also necessary to fine-tune different parameters in order to receive good results, which requires in-depth knowledge of the algorithm. Most of the time, the constraints are thus evaluated using the generic weighted sum technique, which requires few adjustments, or using techniques that are specific to the problem at hand.

The sequential constraint-handling technique (SCHT) seems a promising alternative [2] to other CHTs and the weighted sum, because it does not require that all constraints be evaluated at each iteration. However, it too requires the user to give some parameters, especially the ordering in which those constraints shall be evaluated [3].

* Address all correspondence to this author.
† Both authors contributed equally.

In this paper, we propose two heuristics to deal efficiently with the constraint ordering issue, and we compare two different SCHTs with the weighted sum technique. The comparison is performed using the ten-bar truss problem [4]. We show that at least for this kind of application, the SCHT is indeed a promising alternative to the weighted sum technique. Its limitations are also discussed.

In the first part of the paper the different CHTs are reviewed.

CONSTRAINT-HANDLING TECHNIQUES

Although small in comparison with the sum of works on evolutionary computing, the number of publications dedicated to CHTs is increasing at a fast pace. A website gathering studies in that area lists more than 870 references [5]. These techniques can be classified in four or five categories, see [3;6-8] for review (note that other classifications are possible). These different categories are reviewed below and the choice of the SCHT is motivated.

The most common approach to handling constraints is to use methods based on penalty functions (category 1). The concept behind those methods is "to transform a constrained-optimization problem into an unconstrained one by adding (or subtracting) a certain value to/from the objective function based on the amount of constraint violation present in a certain solution" [6]. Those methods are relatively easy to implement, but the penalty factors/values must be determined by the user and is problem-dependent [9, p. 2]. The weighted sum can be seen as one specific penalty technique: the constraints are incorporated in the objective function and the given weights that penalize the fitness value.

The second category groups methods that try to maintain feasibility of the solutions. For example, the genetic algorithm for numerical optimization for constrained problems system (Genocop, [10]) ensures that the next generated population of a feasible population will also be feasible. Another example is the work done by Schoenauer and Michalewicz [11], who noticed that in real-world problems the global solution lies on the boundary of the feasible region, and devised a method to limit search to that boundary. Those methods require a feasible starting point that may be computationally costly to find, or that must be set by the user [6, p. 1259] and/or necessitates the use of problem-specific operators [11, p. 245].

The third category regroups methods based on the search for feasible solutions. One possibility is "repairing" unfeasible individuals (see details in [6, Section 4]), which has been proved an efficient method if the individuals can be easily transformed; this unfortunately is rarely the case in real-world engineering problems. Another technique often mentioned is the superiority of feasible points heuristic, presented in [12] and [13]. The method does not require any user-defined parameters, which makes it a good candidate, but according to [6, pp. 1263-1264], the method seems to have problems maintaining diversity in the population, and specific methods (niching) need to be implemented to avoid stagnation. These methods are time-consuming and introduce a new parameter to be tuned by the user.

The lexicographic, or sequential, method is the one specifically studied in this paper. Coming from the domain of multi-criteria decision making (see e.g. [14, pp. 188-191]), the lexicographic method consists in considering each constraint separately, in a specific order. When the first constraint is fulfilled, the next constraint is considered. When all constraints are fulfilled, the objective function is optimized. The techniques developed are the behavioral memory method (BM) [2;15] and the lexicographic constraint-handling technique (Lexcoht) [16]. They are presented in the next section. Theoretically, this approach presents the advantages of being easy to implement and less time-consuming than the non-sequential CHTs. An SCHT, however, requires from the user a sequencing of all constraints. This aspect is crucial, as the constraints ordering significantly influences the results (in terms of running time and precision [3]). How to choose an optimal sequence has not been considered in the literature. Two heuristics dealing with this issue are proposed in the section "choosing the right sequence" of this paper.

Finally, the multiobjective optimization techniques can also be used. The constraints are transformed into objectives to fulfill. This is also a promising technique for engineering optimization problems. For reviews, see [6;7;9].

Categories 4 and/or 5 regroup hybrid methods or specific methods. Hybrid methods combine techniques from the different categories above and/or with techniques from other domains, such as fuzzy logic [17] or constraint satisfaction problems [18], see [3;6]. They require supplementary knowledge from the user for their implementation; they have therefore not been investigated further.

THE SEQUENTIAL CONSTRAINT-HANDLING TECHNIQUES

The lexicographic constraint-handling technique (Lexcoht)

Lexcoht [16] can be described as follows:

For each constraint:

- Evaluate the constraint violation
- If the constraint is satisfied: evaluate the next constraint
- If the constraint is not satisfied: stop the evaluation and score the individual according to:

$$p = \frac{m + 1 - a}{c} \tag{1}$$

where p is the individual's score, m is the number of constraints the individual satisfied up until the last constraint evaluated, a is the constraint violation of the last evaluated constraint, and c is the total number of constraints. The constraint violation a is normalized (e.g. $a = \frac{maximal\ allowed\ value}{observed\ value}$), which means p also ranges from 0 to 1.

As a result of Eq. (1), an individual satisfying m constraints is certain to get a higher score than an individual satisfying m–k constraints, $k \in [1, m\text{-}1]$. Lexcoht is presented in Figure 1.

This method requires from the user a linear order of all constraints.

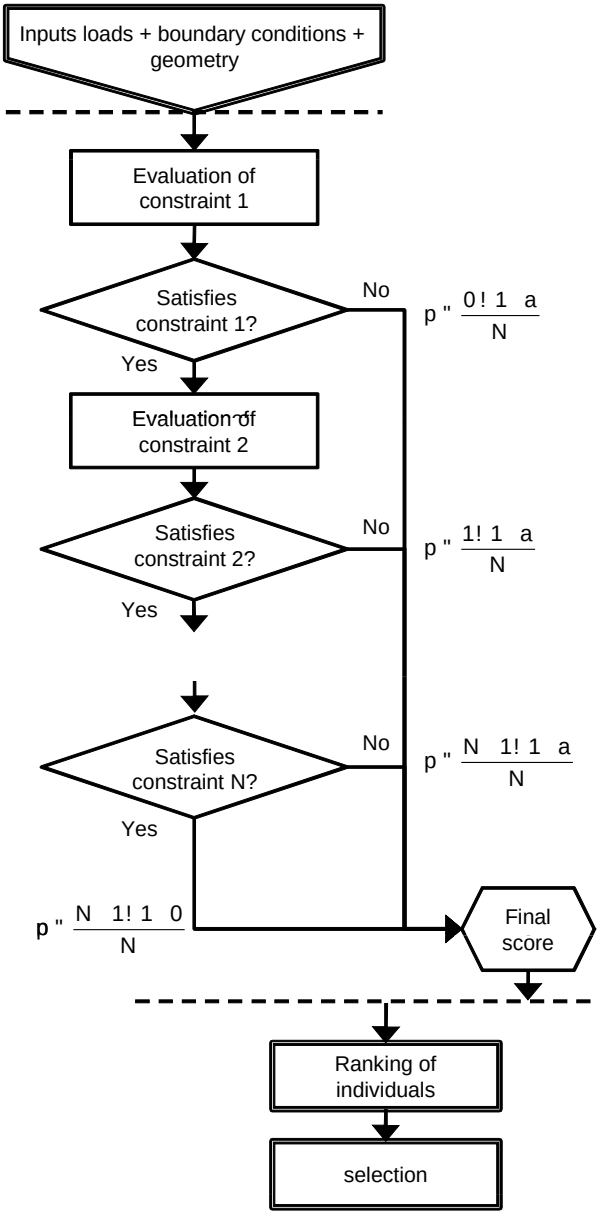

Figure 1. Diagram of the evaluation function of Lexcoht

The behavioral memory (BM) technique

In [2], Schoenauer and Xanthakis describe another sequential approach, the BM technique. It is based on the Behavioral Memory paradigm [19], in which several techniques have been implemented to increase the diversity of the population to avoid premature convergence around certain constraints. The algorithm can be summarized as follows:

A randomly initialized population is optimized in regard to the first constraint. This continues until a certain percentage, the flip-threshold φ, of the population satisfies the constraint. The population is then optimized in regard to the next constraint, until φ percent satisfies the second constraint. Any individual not satisfying the prior constraints is given the score zero. This continues until all constraints have been satisfied. The algorithm is presented in Figure 2.

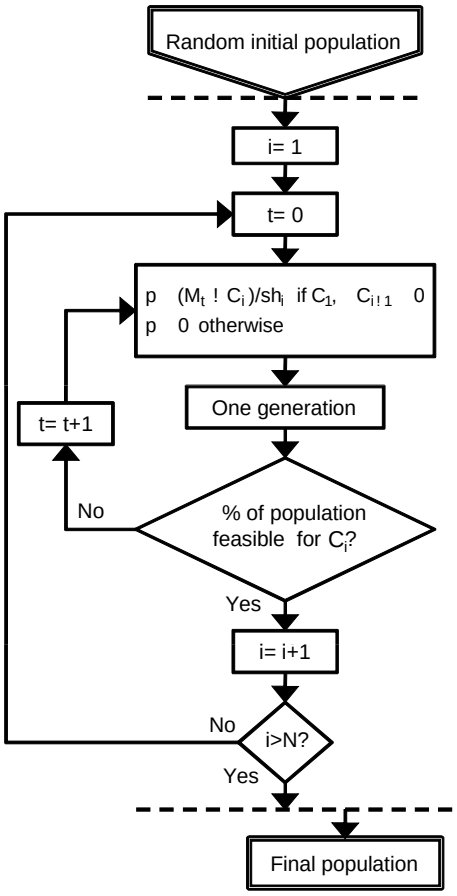

Figure 2. Diagram of the BM technique

To maintain population diversity, a sharing scheme is used as described by [20] and [21]. This method reduces the fitness of individuals that are similar to each other to promote diversity. The user-defined parameter sharing factor, or σ_{sh} is used to decide whether two individuals are similar or not; it is also used to calculate the sharing score sh_i used to penalize individuals that are similar. The score p for each individual can be described as $p = \frac{M_t - C_i}{sh_i}$, where C_i is the constraint violation, and M_t is an arbitrarily large positive number larger than or equal to the largest constraint violation.

Furthermore, a restricted mating scheme as described by [22] is used that promotes mating of similar individuals to

create fitter offspring. The parameter σ_{sh} is also used here to decide if two individuals are similar or not.

This method requires the user to select a linear order of all constraints, and to determine the flip-threshold φ and the sharing factor σ_{sh}. Recommendations for tuning the last two are given in [2]: "the order of magnitude of σ_{sh} can be approximated from below using large φ and increasing σ_{sh} until the required percentage of feasible points cannot be reached anymore. Slightly decreasing σ_{sh} should then allow to find good values for both σ_{sh} and φ."

CHOOSING THE RIGHT SEQUENCE

The choice of the linear ordering of the constraints is important for the SCHTs to obtain good results. One of the difficulties is that the number of sequences increases exponentially with the number of constraints: for c constraints there are also $c!$ possible sequences. In this section we investigate this aspect and discuss a first heuristic to deal with constraint ordering. As mentioned in the review section, the different sequences lead to different running times and precision [3]. Precision here means the aptitude to converge for very sparse feasible space [23]. The convergence issue is dealt with in another on-going study, and the focus of the present work lies on the running time.

We first characterize the constraints in a way that we think is specific to engineering design problems and discuss the behavior of the Lexcoht and BM SCHT under this characterization. We propose two possible heuristics for finding sequences giving satisfying results that will be tested in the subsequent section.

Characterization of the engineering constraints

As Schoenauer and Xanthakis describe it in [2], in engineering optimization problems, "the feasible region is small and quite sparse, and the constraints are available only through some heavy numerical computation." The constraints can also be characterized in terms of how "hard" they are, and how time-consuming they are.

A "hard" constraint can be characterized by the large number of times the constraint is evaluated before an individual is found that fulfills the constraint. The "hardness" of a constraint *using a SCHT* is depending on its place in the sequence. The hardness of a constraint, that is, the size of its solution space, is indeed determined *by the already fulfilled constraints*. Let $\mathcal{F}_\lambda$ be the solution space for constraint λ. Let $\mathcal{S}_{c_{first},...,c_{last}}$ be the remaining search space after the constraints $c_{first},...,c_{last}$ have been fulfilled. The solution space of λ given $c_{first},...,c_{last}$ is $\mathcal{F}_{\lambda|c_{first},...,c_{last}} = \mathcal{F}_\lambda \cap \mathcal{S}_{c_{first},...,c_{last}}$. The solution space of λ given $c'_{first},...,c'_{last}$ is $\mathcal{F}_{\lambda|c'_{first},...,c'_{last}} = \mathcal{F}_\lambda \cap \mathcal{S}_{c'_{first},...,c'_{last}}$. Depending on how $\mathcal{F}_\lambda$ overlaps with the different search spaces, the ratios between the sizes of the solution spaces and the search spaces, $\rho = \dfrac{\left|\mathcal{F}_{\lambda|c_{first},...,c_{last}}\right|}{\left|\mathcal{S}_{c_{first},...,c_{last}}\right|}$ and $\rho' = \dfrac{\left|\mathcal{F}_{\lambda|c'_{first},...,c'_{last}}\right|}{\left|\mathcal{S}_{c'_{first},...,c'_{last}}\right|}$, may differ greatly. A constraint can thus be easy to fulfill in a certain configuration, and hard to fulfill in another.

In the following, t_i is the mean time needed to evaluate a constraint i, n_i is the number of evaluations needed to fulfill constraint i, and n_{ij} is the number of evaluations needed to fulfill constraint i once constraint j is fulfilled ($j \neq i$), etc.

Lexcoht

The total time $T_{1,..,c}$ needed to evaluate c constraints for the sequence $S_{1,..,c}$ using Lexcoht is on average

$$T_{1,..,c} = n_1 t_1 + n_{21} t_2 + ... + n_{c...21} t_c \quad (2)$$

It is necessary to control at each generation whether the individuals evaluated at constraint i still fulfill the prior constraints. Therefore, constraints 1, 2, …, c-1 are evaluated even when they are fulfilled. For any sequence $S_{\sigma(1),..,\sigma(c)}$, we have

$$T_{\sigma(1),..,\sigma(c)} = n_{\sigma(1)} t_{\sigma(1)} + n_{\sigma(2)\sigma(1)} t_{\sigma(2)} + ... + n_{\sigma(c)\sigma(c-1)...\sigma(2)\sigma(1)} t_{\sigma(c)} \quad (3)$$

For ease of notation, the permutation signs (σ) are dropped and Eq. (2) will be used instead of Eq. (3) to represent any sequence permutation.

The sequence to choose is the one that takes the least time. This however depends on the t_is and the different number of evaluations made for each constraint during each sequence. The t_is can be easily determined by simulating the evaluation time of the different constraints and getting an average time. The hardness is much more difficult to determine. First, because the number of ns to determine is equal to $c!c$, which is larger than the $c!$ possible sequences. Second, the hardness in one sequence gives no indication on the hardness in another, as showed earlier. Nevertheless, the first constraint is always evaluated more often than the second, which is evaluated more often than the third, etc. We know from practice that the evaluation times for the different constraints may shift greatly: one problem can involve both the determination of the length of a component (which takes virtually no time) and a structural analysis (requiring time-consuming finite element analysis). That is, the evaluation time ratio $t_1{:}t_2{:}\ldots{:}t_c$ can easily be of the order of magnitude 1 to 10, 1 to 100 or even more. On the other hand, nothing is known about the "hardness ratio" $n_1 : n_{21} : ... : n_{c...21}$ when applied to engineering design problems. If the hardness ratios of the different sequences are similar, that is, of the same order of magnitude, then the influence of the hardness would be negligible in comparison with the evaluation time ratio. A first heuristic would thus be to choose the sequence where the constraints are ordered according to their evaluation times. This heuristic will hereafter be called the evaluation time-based heuristic.

How efficient is this heuristic? For a given problem, we expect the hardness ratio to change heavily depending on

whether the SCHT would be applied for finding one individual or several individuals, and on how long the subsequent optimization would be. If the user wants no more than one solution fulfilling all the constraints, $n_{c...21}$ may well be very small if constraint *c* given *c*-1,.., 1 is easy to fulfill. Second, the particularity of Lexcoht is that once an individual fulfills a constraint the convergence of the population is rapid. Thus the same ratio will decrease rapidly if several solutions are to be found (the increase rate of $n_{c...21}$ would be much higher than n_1). Finally, if the algorithm converges, nearly all individuals will fulfill all constraints after a while. If optimization follows, the ratio becomes $n_1 + K : n_{21} + K : ... : n_{c...21} + K$ where *K* is a constant representing the number of evaluations done by the subsequent optimization. The ratio decreases with *K*. This heuristic can thus guarantee finding very bad sequences, but not finding "very good" sequences Note that a large *K* means that the focus is on optimization rather than constraint handling, and a SCHT is no longer necessary.

BM

The total time $T_{\sigma(1),..,\sigma(c)}$ needed to evaluate *c* constraints for the sequence $S_{\sigma(1),..,\sigma(c)}$ using the BM technique is on average

$$T_{\sigma(1),..,\sigma(c)} = n^{BM}_{\sigma(1)} t_{\sigma(1)} + n^{BM}_{\sigma(2)\sigma(1)} t_{\sigma(2)} + ... + n^{BM}_{\sigma(c)\sigma(c-1)...\sigma(2)\sigma(1)} t_{\sigma(c)} \quad (4)$$

As above, the permutation signs are dropped in the following, and Eq. (4) is expressed as

$$T_{1,..,c} = n^{BM}_1 t_1 + n^{BM}_{21} t_2 + ... + n^{BM}_{c...21} t_c \quad (5)$$

Equations (4) and (5) have the same form as in Lexcoht, but the number of evaluations for each constraint n^{BM}_1, n^{BM}_{21},…, $n^{BM}_{c...21}$ will be different than the equivalent number of evaluations for Lexcoht. The diversity technique used to avoid premature convergence adds several iterations. Consequently, the ratio $n^{BM}_{\sigma(1)} : n^{BM}_{\sigma(21)} : ... : n^{BM}_{\sigma(c...21)}$ is likely to be inferior to the Lexcoht's hardness ratio, and the evaluation time ratio may be more important. The evaluation time-based heuristic is expected to give better results for the BM technique than for Lexcoht. Otherwise, as for Lexcoht, not much can be said about the hardness of the constraints.

A stochastic sequence choice heuristic

An alternative to the evaluation time-based heuristic is to test several sequences. For the cases of three constraints, there can be at most 6 different sequences. It is possible to test all of them and select the best one. From 4 and more constraints, testing all the sequences becomes intractable. However, it is still possible to test a few constraints sequences and have a high probability of obtaining a good sequence. Let *v* be the proportion of targeted top constraint sequences (for example one wishes to get a constraint from the top 10%) and let *k* be the number of sequences one is willing to test. $v = \frac{w}{c!}$, where *w* (rounded to the nearest integer) is the number of constraints sequences in *v*, and *c*! is the total number of possible sequences. The probability P_v of choosing at least one sequence among the *k* chosen that belongs to the top *v* of all the possible sequences is

$$P_v = 1 - \frac{c!-w}{c!} \cdot \frac{c!-1-w}{c!-1} \cdot .. \cdot \frac{c!-w-k+1}{c!-k+1} \quad (6)$$

When *k* is negligible in front of *c*! and *w*, Eq. (6) becomes

$$P_v = 1-(1-v)^k, \forall c \quad (7)$$

Table 1 presents the different probabilities of getting a constraint sequence in the top *w* sequences as a function of the number of randomly chosen sequences *k* and the number of constraints *c*. With a *k* relatively small, the probability of getting a good sequence in the top 25% is high ($P > 82\%$ for $k = 6$). Moreover, Eq. (7) is already verified with *c*! = 7! or 8! This means that the probability of getting a good sequence remains stable whatever the number of constraints.

Table 1. Table of probabilities that at least one sequence is in the set *v*

c		4	5	6	7	8	∞
c!		24	120	720	5040	40320	∞
v = 5%	*k* = 2	0.163	0.098	0.098	0.098	0.098	0.098
	3	0.239	0.144	0.143	0.143	0.143	0.143
	4	0.312	0.188	0.186	0.186	0.186	0.185
	5	0.380	0.230	0.227	0.226	0.226	0.226
	6	0.446	0.270	0.266	0.265	0.265	0.265
	7	0.507	0.308	0.303	0.302	0.302	0.302
10%	2	0.239	0.191	0.190	0.190	0.190	0.190
	3	0.343	0.273	0.271	0.271	0.271	0.271
	6	0.597	0.476	0.470	0.469	0.469	0.469
	7	0.664	0.531	0.523	0.522	0.522	0.522
25%	2	0.446	0.439	0.438	0.438	0.438	0.438
	3	0.597	0.582	0.579	0.578	0.578	0.578
	6	0.862	0.830	0.823	0.822	0.822	0.822
	7	0.908	0.874	0.868	0.867	0.867	0.867
50%	2	0.761	0.752	0.750	0.750	0.750	0.750
	3	0.891	0.878	0.876	0.875	0.875	0.875
	6	0.993	0.986	0.985	0.984	0.984	0.984
	7	0.998	0.994	0.992	0.992	0.992	0.992

APPLICATION TO THE TEN-BAR TRUSS PROBLEM

In this section we test whether the SCHTs are interesting alternatives to the weighted sum technique when applied to the well-known ten-bar truss benchmark problem [4]. The two sequence selection heuristics that have been proposed are also applied to the ten-bar truss problem.

The modified ten-bar truss problem

The problem is based on a ten-bar truss with geometry, loads and boundary conditions as shown in Figure 3. The ten-bar truss problem has been used as a benchmark in evolutionary computing in several works, for example [15;24-26], and consists of ten bars connected to six nodes, with the external load P applied to nodes n_3 and n_6, and a fixed support at nodes n_1 and n_2.

The objective of the problem is to find a truss structure that supports the load *P* given the boundary conditions and constraints. The original problem has been modified to include additional constraints, presented in the next section. The focus being on comparing different CHTs, *the optimization part is not considered here.*

In order to fulfill the constraints, the constraint-handling algorithm can change the material and profile of each of the ten bars. The cross-sectional profile and material selection for each bar constitutes each individual's genome and are the only design variables. The positions and connections of the nodes are fixed. The possible values of the design variables are presented Table 2.

Table 2. Materials and beam profiles

Material	Elasticity modulus (Pa)	Density (kg/m^3)	Price (€/kg)	Max tensile stress (Pa)	Max compressive stress (Pa)
1	2.00×10^{11}	8000	7.2	5.70×10^{8}	3.50×10^{8}
2	1.10×10^{10}	600	9.5	4.00×10^{7}	1.96×10^{7}
3	1.10×10^{11}	4500	13	1.00×10^{9}	9.70×10^{8}
4	6.90×10^{10}	2700	3.2	3.10×10^{8}	5.30×10^{8}
Profile		A (m^2)		I (m^4)	
1		7.64×10^{-4}		80.14×10^{-8}	
2		1.32×10^{-3}		317.8×10^{-8}	
3		2.01×10^{-3}		869.3×10^{-8}	
4		3.34×10^{-3}		$2{,}770\times10^{-5}$	

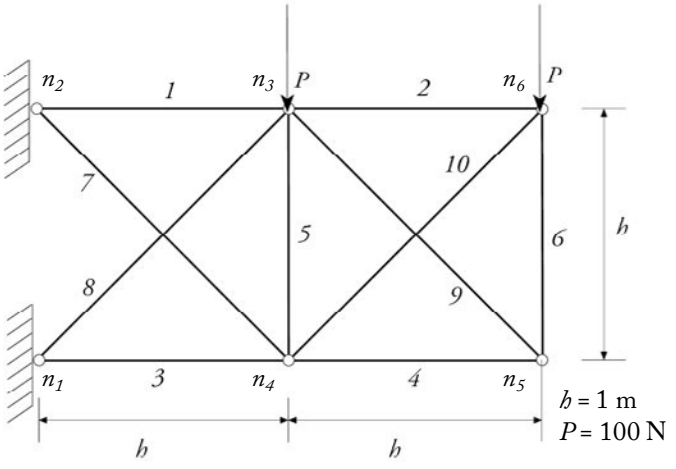

Figure 3. The ten-bar structure

The constraints

Six constraints need to be fulfilled.

Each bar is subject to

$$\sigma_i^{min} \le \sigma_i \le \sigma_i^{max} \tag{8}$$

and

$$\sigma_i \ge \sigma_i^{cr} \tag{9}$$

where σ_i is the stress in bar *i*, $\sigma_i^{min} \le 0$ is the stress limit in compression for the bar profile and material, and $\sigma_i^{max} \ge 0$ is the stress limit in tension, and $\sigma_i^{cr} \le 0$ is the stress limit for buckling, and is calculated as $\sigma_i^{cr} = \dfrac{\pi^2 EI}{(KL)^2 A}$.

Each node is subject to

$$u_i \le u_{max} \tag{10}$$

where u_i is the node displacement, and u_{max} is the maximum displacement.

The entire truss structure is subject to

$$p \le p_{max} \tag{11}$$

$$n \le n_{max} \tag{12}$$

$$w \le w_{max} \tag{13}$$

where *p*, *n*, *w* are the price, number of different combinations of member profiles and materials in the structure, and weight of the structure, and p_{max}, n_{max}, w_{max} are the maximum allowed price, number of different combinations of member profiles and materials, and weight, respectively. For demonstration purposes the price of the truss structure is used as a constraint, although in most applications it would rather be an objective. The stress constraints are dependent on the profile and material used in the member; the rest of the constraints are set as follows: u_{max} = 0.001 m, p_{max} = 55 €, n_{max} = 3, and w_{max} = 6 kg.

Preliminary runs where the constraints were individually studied showed that the number of generations needed to find a feasible point was 1:1:28.3:52.3:1:1 for the stress, buckling, weight, price, number of elements, and displacement constraints respectively.

Evaluation times for each constraint have been set at 10:10:1:1:1:10 time units (t.u.). They are thought to reflect a typical engineering design problem: groups of constraints have an evaluation time of the same order of magnitude, and differ among each other by one or several orders of magnitude. This difference was not set too high in order not to mask the possible hardness effect. In reality, the evaluation of the stress, buckling, and displacement constraints rely on a single finite element-analysis, which means the time-consuming analysis only needs to be run once to evaluate all three constraints. However, in this example they are still treated as independent constraints. The handling of groups of constraints dependent on one evaluation is discussed in the conclusion.

Objectives of the study

The first objective of this study is to compare the SCHTs against the weighted sum technique. The second objective is to investigate the efficiency of the proposed selection heuristics. For sake of simplicity, this section begins with the second objective.

With the 6 constraints of the problem, 720 sequences are possible.If the stochastic sequence choice heuristic is correct, large differences in the constraint satisfaction time should be observed among a small set of randomly chosen sequences. 7 sequences are chosen randomly among the 720 sequences. *Hypothesis 1a:* for Lexcoht, there is one group of sequences performing significantly better than another group of sequences. *Hypothesis 1b:* for the BM technique, there is a group of sequences performing significantly better than another group of sequences.

Using the evaluation-time based heuristic, the sequences in which the constraints whose evaluation times are the largest are positioned last should perform better than the sequences in which the constraints whose evaluation times are the largest are positioned first. Two sequences with constraints with large evaluation times positioned last and one sequence with constraints with large evaluation times positioned first were chosen. *Hypotheses 2a and b:* At least one of the two sequences with constraints with large evaluation times positioned last should perform better than the sequence with constraints with large evaluation times positioned first a) for Lexcoht and b) for the BM technique.

Moreover, at least one of these two sequences should perform well in comparison with the 7 sequences chosen at random using the stochastic sequence choice heuristic (*Hypotheses 3a and 3b*). Finally, it is expected that the evaluation time-based heuristic will perform better for the BM technique than for Lexcoht (*Hypothesis 4*).

Several elements intervene in the comparison of the SCHTs against the weighted sum technique. The weights of the weighted sum can play a role, and thus three sets of weights are used (see next section). Comparing the Lexcoht and BM techniques against the weighted sum technique amounts to comparing their best sequences for each heuristic against the three weighted sums. It is expected that the Lexcoht and BM techniques should perform better than the weighted sum technique variants (*Hypothesis 5*). The relative performance of the Lexcoht and BM techniques is also tested. We thus test the hypothesis that the Lexcoht and BM techniques will get different running times for the same sequences vs. the null hypothesis that they will not (*Hypothesis 6*). This experimental setup allows us also to investigate whether there are differences among the different weighted sum technique variants (*Hypothesis 7*).

Finally, determining the weights of the different constraints is a recurring problem when using GAs [27, p. vi]. Because three weighted sum variants are used in this investigation, it is possible to study whether the assignment of different weights plays a role. The constraint sequences used for the study of the BM and Lexcoht techniques are also used for this test: the weights are put in the same order for each sequence. It is expected that different weighting schemes will lead to differences in performance (*Hypothesis 8*).

Experimental setup

The factors (or independent variables) of the experiment, are the methods (5 in total: Lexcoht, BM and 3 weighted sum technique variants) and the sequences (10 in total: 7 randomly chosen sequences, 2 sequences with constraints with large evaluation times positioned last and 1 sequence with constraints with large evaluation times positioned first), forming a 5×10 factorial design. The dependent variable is the time taken by a population to fulfill the constraints of the modified ten-bar truss problem. For each level of the methods and sequences factors, the problem is run 200 times, i.e. 10,000 times in total. For each run, a new population is created. When there is no convergence, the run is discarded.

The parameters for the BM techniques are set according to the recommendations from [2], and yield good results with $\varphi = 0.6$ and $\sigma_{sh} = 0.05$.

The three different weighting schemes are as follow. In the first configuration, all weights are set at 1 (unweighted sum technique, or UWS). The second is a linearly weighted scheme (WS1), where the constraint i in the sequence is given the weight $(c+1-i)\cdot 10$. The third scheme is an exponentially weighted sum (WS2), where the constraint i in the sequence is given the weight 10^{c-i}.

Each individual is represented by the design variables described earlier, in total 20 variables per individual with 4 possible values per variable (one bar profile and one type of material can be chosen for each of the 10 bars independently from one another). Using binary chromosome encoding, the chromosome can thus be represented by 2 bits per variable resulting in a chromosome with 40 bits.

The genetic algorithm used is the standard MATLAB implementation, with bit string as population type, rank as scaling method, stochastic uniform as selection method, Gaussian as mutation function, single point as crossover function, elite count 2, and crossover fraction 0.8. The genetic algorithm is run with a population of 150 individuals, during a maximum of 1500 generations.

Results

All the experiment runs converged. The exploratory data analysis revealed that the distributions of the convergence time for each combination were markedly positively skewed. The standard deviations were found proportional to the means, thus a logarithmic transformation was applied to the data [28, pp. 319-321]. The log-transformed populations were mostly normally distributed; the Jarque-Bera test for normality [29] failed to show a significant deviation from a normal distribution for most of the combinations. With the largest variance ratio being 1:11.8, the heteroscedasticity was much higher than the limit on

heterogeneity of variance for which the analysis of variance is still robust (see [28, p. 317]). In the case of equal sample sizes, the procedure is found to be too liberal. Therefore, the Box procedure [28, p. 317;30, p. 891] was applied.

The 5x10 factorial analysis of variance reveals that there is a significant method effect, a significant sequence effect and a significant interaction between methods and the sequences; see Table 3, upper level. The simple effect analysis of the sequence factor for each level of the method factor reveals that the different sequences significantly affect the sequential methods, but not the unweighted and weighted methods; see Table 3, lower level. Thus hypothesis 8 has to be rejected.

Table 3. Overall analysis of variance and simple effects of of the Sequences factor

Overall analysis				
Source	df	Sum of Squares	Mean Squares	*F*
Methods	4	3150.22	229.71	329.89*
Sequences	9	2067.43	787.56	1131.00*
Methods × Sequences	36	3271.48	90.87	130.50*
Error	9950	6928.52	0.70	
Total	9999	15417.65		
Simple effects of Sequences				
Lexcoht	9	3053.30	339.26	487.20*
BM	9	2270.30	252.26	362.27*
UWS	9	6.30	0.70	1.01
WS1	9	4.83	0.54	0.77
WS2	9	4.18	0.47	0.67
Error	9950	6928.52	0.70	

* $p < .001$, $F_{Box,\alpha = .001}(1, 200 - 1) = 11.16$.

For Hypotheses 1 to 7, post hoc multiple comparisons using the Tukey test were performed. As for the analysis of variance, the Tukey test, tend to be liberal for large ratios [31]. However, a study by Game and Howell [32] shows that the this test is still robust for a ratio as large as 1:13, which is higher than the largest variance ratio of this experiment (1:11.8). Figure 4 presents the log-transformed means for each method and sequence. In the following, the reported means are the antiloged means; cf. [28, p. 319].

Concerning the stochastic sequence choice heuristic, the worst randomly chosen Lexcoht sequence (M = 19,718 t.u.) was significantly different from the best Lexcoht sequence (M = 957 t.u., Tukey $p < .001$), and the randomly chosen BM sequence (M = 59,415 t.u.) was significantly different from the best BM sequence (M = 5,491 t.u., Tukey $p < .001$). This confirms hypotheses 1a and 1b.

Concerning the evaluation time-based heuristic, the best sequence of constraints with large evaluation times positioned last (M_{last} = 705 t.u.) was significantly better than the sequence of constraints with large evaluation times positioned first (M_{first} = 19,229 t.u., Tukey $p < .001$) for Lexcoht. The same result was obtained for the BM method (M_{last} = 2,981 t.u., M_{first} = 58,105 t.u., Tukey $p < .001$). This confirms hypotheses 2a and 2b.

The sequences chosen through the evaluation time-based heuristic performed well in comparison with the 7 sequences chosen at random using the stochastic sequence choice heuristic. The best sequence selected using the stochastic sequence choice heuristic was not better than the best evaluation time-based sequence (M = 957 t.u. vs. M_{last} = *705* t.u., Tukey $p = .15$) for Lexcoht. The best evaluation time-based sequence for BM was even better than the best randomly chosen sequence (M = 5,491 t.u. vs. M_{last} = 2,981 t.u., Tukey $p < .001$). This confirms hypotheses 3 and 4.

The best Lexcoht sequences for each heuristic were better than the UWS (M_{UWS} = 19,051 t.u.), WS1 (M_{WS1} = 19,026 t.u.), and WS2 (M_{WS2} = 17,413 t.u., Tukey ps < .001). Likewise, the best BM sequences for each heuristic were better than the UWS, WS1, and WS2 (Tukey ps < .001). This confirms hypothesis 5. The best Lexcoht sequences for each heuristic performed better than the best BM sequences for each heuristic (Tukey ps < .001, hypothesis 6). The three weighted sum technique variants did not differ significantly from one another, rejecting hypothesis 7. The worst Lexcoht sequence did not differ at all from the weighted sum technique variants (Tukey ps ≈ 1.0), but the worst BM sequence did (Tukey ps < .001).

Empirical results for the hardness ratios

The hardness ratios for the Lexcoht and BM techniques are presented Table 4.

Table 4. Hardness ratios for the ten-bar truss problem

Seq	Lexcoht $n_1 : n_{21} : \ldots : n_{6\ldots21}$	BM $n_1^{BM} : n_{21}^{BM} : \ldots : n_{6\ldots21}^{BM}$
1	831:831:831:109:109:109	2454:2450:2439:90:50:31
2	925:925:104:1:1:1	2214:2161:274:65:33:15
3	872:872:872:872:80:1	2441:2434:2419:2404:173:15
4	881:881:114:114:114:114	2321:2299:151:114:93:71
5	828:75:75:75:1:1	1831:260:223:217:44:15
6	764:764:75:75:1:1	2276:2234:202:178:43:15
7	830:642:642:88:88:1	2320:2289:2260:168:137:15
8	877:877:877:877:110:1	2391:2385:2370:2355:168:15
9	815:79:1:1:1:1	1939:319:78:52:32:15
10	865:120:1:1:1:1	1820:277:83:52:34:15

Discussion

In this application, both the Lexcoht and the BM techniques performed significantly better than the weighted sum technique. The difference is huge: $M_{Lexcoht,\ best}$ = 705 t.u. and $M_{BM,\ best}$ = 2,981 t.u., vs. M_{UWS} = 19,051 t.u., M_{WS1} = 19,026 t.u. and M_{WS2} = 17,413 t.u. This also confirms that the SCHTs are relevant alternatives for engineering design problems.

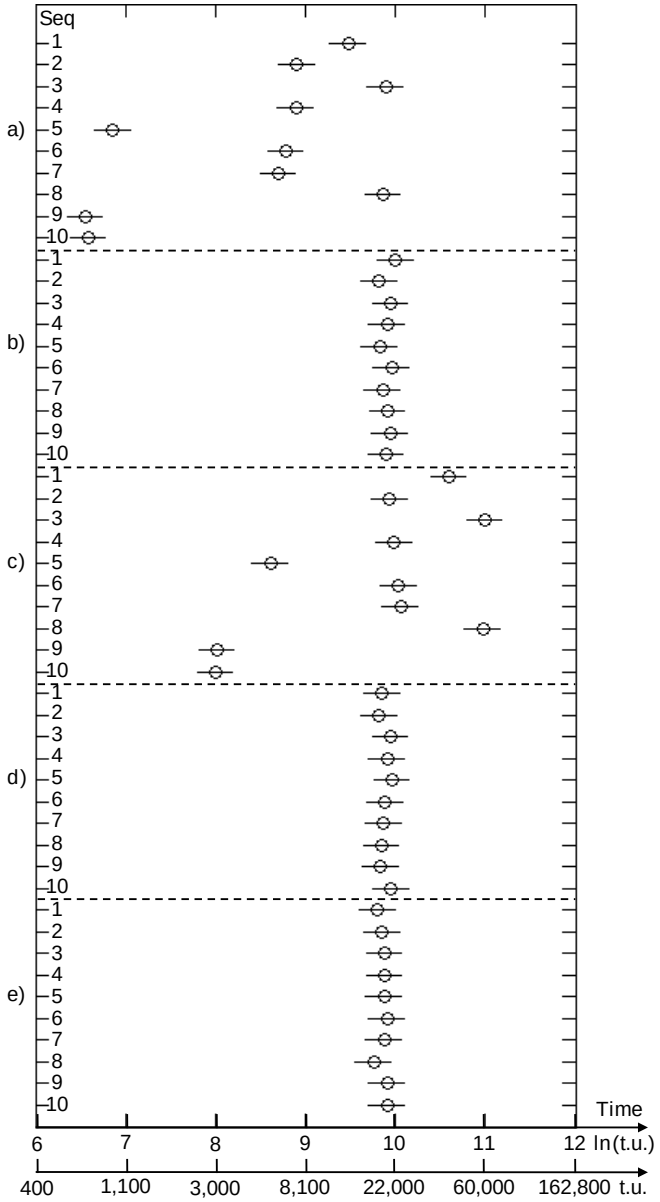

Figure 4. Representation of the log-transformed means and their comparisons intervals (99.9%) for the a) Lexcoht, b) UWS, c) BM, d) WS1, and e) WS2 techniques.

One of the main issues related to the use of SCHTs is the selection of a satisfying constraint sequence. The two heuristics presented in this paper have both permitted the selection of a good sequence for the ten-bar truss benchmark application. This is a strong indication that the SCHTs can be used in practice. Using Lexcoht, the hardness ratios can vary greatly among the sequences. For example, the ratios of sequences 1 and 4 are quite different from those of the other sequences. This implies that the evaluation time-based heuristic is less likely to be adequate for Lexcoht. In this particular application, the two sequences selected by the evaluation-time based heuristic did not perform better than the stochastically selected best sequence. The hardness ratios vary less dramatically for the BM technique, making the evaluation time-based heuristic more powerful. This is confirmed by the empirical results where the sequences selected with this heuristic performed significantly better than the best sequence selected by the other heuristic. This makes the BM technique very interesting if one does not want to test 6 or 7 sequences before selecting a good one.

The evaluation times ratio (10:10:1:1:1:10) proved to be an interesting choice. It was clearly less than the hardness ratios (Table 4) but still considerably influences the time taken for the different sequences to fulfill the constraints. This also speaks for the relevance of the evaluation time-based heuristic.

The weighted sum technique variants did not perform significantly better from one to the other. Surprisingly, the weights put on the different constraints did not affect the constraint satisfaction time. This may be an artifact of this specific problem and would require further investigation.

What this experiment also shows is that it can be detrimental to select only one sequence and solve the problem at hand with it. The worst BM sequence was significantly worse than the weighted sum technique. Lexcoht did not perform worse than the weighted sum technique with the chosen sequences, but it cannot be excluded that it wouldn't with some of the 710 remaining combinations. In this particular problem, all the simulations have converged towards a solution. In other problems, this may not be the case.

CONCLUSION

In this paper, the issue of CHT for engineering design problems has been emphasized and the SCHTs presented as an alternative to the weighted sum technique regarding genericity and computation time. The efficiency of the SCHTs depends on the choice of a "good sequence". For that purpose, two heuristics have been proposed and tested. They have proved to perform well for the ten-bar truss problem. Moreover, both the Lexcoht and BM techniques have performed better than the weighted sum technique.

In terms of usability, Lexcoht has a slight advantage over the BM technique because it does not require the determination of a flip-threshold or sharing factor value. A sharing scheme for maintaining diversity has not so far proven necessary for Lexcoht. This leaves another parameter out of the hands of the user. Nevertheless, this may be useful in other applications. The Lexcoht and BM techniques have behaved quite similarly for the ten-bar truss problem, that is, the sequence running times have the same ordering for both methods. This requires further investigation.

The two heuristics do not require any advanced knowledge from the user: either choose the sequence where all constraints are ordered according to their evaluation time, or choose 6 or 7 sequences randomly and test them. These simulations may take time and might be a limitation of the SCHTs if the user's problem has only to be solved once. This is, however, not unlike the

tuning of a penalty function or the determination of the weights for a weighted sum. Recommendations for minimizing the sequence selection testing time of the stochastic heuristic need to be devised.

This paper has dealt with the constraint satisfaction time issue. As mentioned in the CHT review section, there can be a problem of "precision" by using SCHTs, i.e. the difficulty for the SCHTs to converge for some sequences {Michalewicz, 1996 MICHALEWICZ1996A /id} (there was no such problem in this application). This is being investigated in a subsequent study, not published yet.

In computationally-intensive simulations, it is quite common to have some constraints evaluated simultaneously, for example constraints on stress, displacement or buckling in the case of finite element analysis. For such linked constraints, an SCHT cannot be used as described in this paper; instead, a possibility could be to treat these linked constraints as a grouped constraint evaluated at the same time using techniques such as the weighted sum; further research is needed in this area. With SCHTs, the constraint satisfaction and optimization parts of a structural problem are considered separately. Only the constraint satisfaction part has been investigated here. It is possible to transform constraints into objectives to fulfill and use multiobjective optimization techniques instead of the SCHTs; see [6;7;9]. The relative benefits of the multiobjective optimization techniques and the SCHTs require further analysis.

ACKNOWLEDGMENT

This work was supported by the Swedish Governmental Agency for Innovation Systems, VINNOVA, through the Production Strategies and Models for Product Development program (project 2009-04057). We also thank one of the reviewers for pointing out an important limitation of the presented techniques.

REFERENCES

[1] Coello Coello, C. A., Lamont, G. B. and Van Veldhuizen, D. A., 2010, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd Edition, Springer, New York, NY.

[2] Schoenauer, M. and Xanthakis, S., 1993, "Constrained GA optimization", *5th International Conference on Genetic Algorithms - ICGA'93*, Urbana, IL, July 17-21, 1993, pp. 573-580.

[3] Michalewicz, Z. and Schoenauer, M., 1996, "Evolutionary algorithms for constrained parameter optimization problems", *Evolutionary Computation*, **4**(1), pp. 1-33.

[4] Haug, E. J. and Arora, J. S., 1979, *Applied Optimal Design - Mechanical and Structural Systems*, Wiley, New York, NY.

[5] Coello Coello, C. A., 2010, *List of References on Constraint-Handling Techniques used with Evolutionary Algorithms*, <http://www.cs.cinvestav.mx/~constraint/>, last accessed: November 16, 2010.

[6] Coello Coello, C. A., 2002, "Theoretical and numerical constraint-handling techniques used with evolutionary A survey of the state of the art", *Computer Methods in Applied Mechanics and Engineering*, **191**(11-12), pp. 1245-1287.

[7] Mezura-Montes, E., 2004, *Alternative Techniques to Handle Constraints in Evolutionary Optimization*, PhD Thesis, Computer Science Section, Electrical Engineering Department, CINVESTAV-IPN, Mexico.

[8] Yeniay, Ö., 2005, "Penalty function methods for constrained optimization with genetic algorithms", *Mathematical and Computational Applications*, **10**(1), pp. 45-56.

[9] Mezura-Montes, E. and Coello Coello, C. A., 2006, *A Survey of Constraint-Handling Techniques Based on Evolutionary Multiobjective Optimization*, EVOCINV-04-2006, Technical Report, Evolutionary Computation Group, Departamento de Computación, CINVESTAV-IPN, Mexico.

[10] Michalewicz, Z. and Janikow, C. Z., 1991, "Handling constraints in genetic algorithms", *4th International Conference on Genetic Algorithms - ICGA'91*, San Diego, CA, July, 1991, pp. 151-157.

[11] Schoenauer, M. and Michalewicz, Z., 1996, "Evolutionary computation at the edge of feasibility", *4th Parallel Problem Solving from Nature Conference - PPSN IV*, Volume 1141 of Lecture Notes in Computer Science, Berlin, September 22-26, 1996, pp. 245-254.

[12] Powell, D. J. and Skolnick, M. M., 1993, "Using Genetic Algorithms in Engineering Design Optimization with Non-Linear Constraints", *5th International Conference on Genetic Algorithms - ICGA'93*, Urbana-Champaign, IL, July 17-21, 1993, pp. 424-431.

[13] Deb, K., 2000, "An efficient constraint handling method for genetic algorithms", *Computer Methods in Applied Mechanics and Engineering*, **186**(2-4), pp. 311-338.

[14] Bouyssou, D., Marchant, T., Pirlot, M., Tsoukiàs, A. and Vincke, P., 2006, *Evaluation and Decision Models with Multiple Criteria - Stepping Stones for the Analyst*, Springer, New York, NY.

[15] Michalewicz, Z., Dasgupta, D., Le Riche, R. G. and Schoenauer, M., 1996, "Evolutionary algorithms for constrained engineering problems", *Computers & Industrial Engineering*, **30**(4), pp. 851-871.

[16] Nordin, A., Hopf, A., Motte, D., Bjärnemo, R. and Eckhardt, C.-C., 2011, "Using genetic algorithms and Voronoi diagrams in product design", *Journal of Computing and Information Science in Engineering*, Accepted.

[17] Van Le, T., 20-5-1996, "A fuzzy evolutionary approach to constrained optimisation problems", *3rd IEEE International Conference on Evolutionary Computation - ICEC'96*, Nagoya, May 20-22, 1996, pp. 274-278.

[18] Paredis, J., 1994, "Co-evolutionary constraint satisfaction", *3rd Parallel Problem Solving from Nature Conference - PPSN III*, volume 866 of Lecture Notes in Computer Science, Jerusalem, October 9-14, 1994, pp. 46-55.

[19] de Garis, H., 1990, "Genetic programming: Building artificial nervous systems with genetically programmed neural network modules", *7th International Conference on Machine Learning*, Austin, TX, June 21-23, 1990, pp. 132-139.

[20] Holland, J. H., 1975, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology,*

Control and Artificial Intelligence, University of Michigan Press, Ann Arbor, MI.

[21] Goldberg, D. E. and Richardson, J., 1987, "Genetic algorithms with sharing for multimodal function optimization", *2nd International Conference on Genetic Algorithms - ICGA'87*, Cambridge, MA, July 28-31, 1987, pp. 41-49, Mahwah, NJ, USA.

[22] Deb, K. and Goldberg, D. E., 1989, "An investigation of niche and species formation in genetic function optimization", *3rd International Conference on Genetic Algorithms - ICGA'89*, Fairfax,VA, June 4-7, 1989, pp. 42-50.

[23] Michalewicz, Z., 1995, "Genetic algorithms, numerical optimization, and constraints", *6th International Conference on Genetic Algorithms - ICGA'95*, Pittsburgh, PA, July, 1995, pp. 151-158.

[24] Giger, M. and Ermanni, P., 2006, "Evolutionary truss topology optimization using a graph-based parameterization concept", *Structural and Multidisciplinary Optimization*, **32**(4), pp. 313-326.

[25] Guo, X., Cheng, G. and Yamazaki, K., 2001, "A new approach for the solution of singular optima in truss topology optimization with stress and local buckling constraints", *Structural and Multidisciplinary Optimization*, **22**(5), pp. 364-373.

[26] Stolpe, M. and Svanberg, K., 18-3-2003, "A note on stress-constrained truss topology optimization", *Structural and Multidisciplinary Optimization*, **25**(1), pp. 62-64.

[27] Hutchison, D., Branke, J., Deb, K., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Miettinen, K., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Slowinski, R., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y. and Weikum, G., 2008, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Springer, Berlin.

[28] Howell, D. C., 2007, *Statistical Methods for Psychology*, 6th Edition, Thomson Wadsworth, Belmont, CA.

[29] Jarque, C. M. and Bera, A. K., 1987, "A test for normality of observations and regression residuals", *International Statistical Review*, **55**(2), pp. 163-172.

[30] Geisser, S. and Greenhouse, S. W., 1958, "An extension of Box's results on the use of the *F* distribution in multivariate analysis", *The Annals of Mathematical Statistics*, **29**(3), pp. 885-891.

[31] Ramseyer, G. C. and Tcheng, T.-K., 1973, "The robustness of the Studentized range statistic to violations of the normality and homogeneity of variance assumptions", *American Educational Research Journal*, **10**(3), pp. 235-240.

[32] Games, P. A. and Howell, J. F., 1976, "Pairwise multiple comparison procedures with unequal *n*'s and/or variances: A Monte Carlo study", *Journal of Educational Statistics*, **1**(2), pp. 113-125.