Proceedings of the International Association for
Shell and Spatial Structures (IASS) Symposium 2013
„BEYOND THE LIMITS OF MAN"
23-27 September, Wroclaw University of Technology, Poland
J.B. Obrębski and R. Tarczewski (eds.)

Discrete design optimization accounting for practical constraints

Mattias Schevenels[1], Sean McGinn[3], Anke Rolvink[2,3], Jeroen Coenders[2,3]

[1]Department of Architecture, KU Leuven, Belgium, *mattias.schevenels@asro.kuleuven.be*
[2]BEMNext Lab, Delft University of Technology, Netherlands *{a.rolvink, j.l.coenders}@tudelft.nl*
[3]Arup Amsterdam, Netherlands *{sean.mcginn, anke.rolvink, jeroen.coenders}@arup.com*

Summary: This paper presents a heuristic algorithm for discrete design optimization, based on the optimality criteria method. Practical applicability is the first concern; special attention is therefore paid to the implementation of technological constraints. The method is generally applicable, but in order to clarify the idea, it is used in this paper for the optimal design of a Warren type truss composed of steel CHS members with welded joints. For each member, the optimal section is chosen from a given steel catalog. Following Eurocode 3 and the CIDECT design guidelines, restrictions are imposed on the deflections, the member forces, the layout of the joints, and the joint forces. The buildability of the design is ensured by requiring all top chord members and all bottom chord members to have the same outer diameter. The algorithm is successfully applied to a simple test case as well as the example truss design optimization problem. In terms of computational cost, the algorithm is comparable to a continuous optimization scheme based on finite differences, and much faster than an evolutionary algorithm.

Keywords: *structural design optimization, discrete design optimization, optimality criteria method, buildability constraints*

1. INTRODUCTION

The optimal design of structures has been an active research area over the past half century. However, practicing structural engineers seem to hesitate to adopt optimization as a daily design tool [8], even for relatively simple and tedious tasks such as the sizing of the members of a steel structure. One of the reasons for this apparent reluctance is the fact that it is very difficult to take into account the appropriate technological constraints so as to ensure the buildability of the optimized design.

In this paper we present a method to account for technological constraints in design optimization. The method is applicable to any design optimization problem, but in order to explain the idea we apply it here to a simple (but realistic) example problem: the size and shape optimization of a Warren type truss under static loading. The truss is composed of steel Circular Hollow Section (CHS) members. The objective is to minimize the weight of the truss. The usual displacement, member force, and buckling constraints as formulated in part 1-1 of Eurocode 3 [4] are imposed. In addition, the joints must satisfy the (geometrical and mechanical) rules specified in the CIDECT design guide for CHS joints [11], the member sections must be chosen from a given section catalog, all top chord members must have the same external size, and all bottom chord members must have the same external size.

The example problem has the following properties, which are typical for a design optimization problem that takes into account practical constraints: (1) not all constraints are physically motivated and mathematically well-behaved (this is particularly true for the CIDECT design rules), (2) it is a mixed discrete-continuous optimization problem (section diameters and wall thicknesses are discrete, truss dimensions are continuous), and (3) some of the optimization parameters are linked (section diameters and wall thicknesses are linked via a table of available sections, and the diameters of the chord members are required to be identical).

Several optimization methods have been proposed in the literature to handle problems with one or more of these properties. Practically all recent methods are based on evolutionary computing. Evolutionary computing methods are attractive as they are conceptually simple and easy to implement, but unfortunately their convergence is extremely slow, making them less ideal for use in engineering practice. Moreover, algorithms that can handle linked optimization parameters are often restricted to parameters that are linked through a one-to-one relationship. They cannot handle the example optimization problem where a CHS profile with a given diameter can have multiple wall thicknesses. In conclusion, there is still a need for an optimization scheme that is fast, easy to use, and capable of handling all types of practical constraints.

In this paper we present a new optimization scheme for discrete design optimization based on the optimality criteria method [10]. The optimality criteria method is only applicable to continuous optimization problems. We propose an alternative method based on the same principles for problems with discrete optimization parameters that may or may not be linked.

The remainder of this paper is organized as follows. Section 2 introduces a framework to cope with the discrete and linked nature of the optimization parameters. Section 3 describes the actual optimization scheme. In section 4, this scheme is used for a simple test case with two optimization parameters in order to illustrate the principle. Finally, section 5 addresses a realistic design optimization problem – the truss design problem introduced before.

2. OPTIMIZATION FRAMEWORK

Before outlining the actual optimization scheme, we first set up a framework to cope with the discrete nature of the optimization parameters. In the terminology of Arora and Huang [2], we adopt a hybrid single - multiple variable approach, using a single variable (the design entity) to identify the design state, and multiple variables (the parameters) to select the best design update in each iteration of the optimization process. We also propose a scheme to ensure that all links between optimization parameters remain satisfied when performing a design update.

2.1. Design entities

The primary element of the optimization framework is the design entity. Each design entity represents one of the unknowns in the optimization problem. Design entities can be scalar variables such as the dimensions of a structure, or more complex variables such as a section profile to be chosen from a steel catalog. We assume that all unknowns are discrete, so that we can construct a table containing all possible values for each design entity. In a practical design optimization problem, continuous variables can always be made discrete using an appropriate discretization step.

In the example problem, the height of the truss is a scalar design entity – possible values are given in table 1. The member sections must be chosen from a catalog of CHS profiles – this catalog is given in table 2. The catalog contains 119 sections defined by the European standard EN 10210-2 [5]. For each section, an index and a name are specified, as well as the outer diameter d, the wall thickness t, the section area A, and the moment of inertia I.

The state of each design entity e is represented by an index i_e. These indices are collected in a vector $\mathbf{i}$ representing the state of the entire design.

Table 1: Possible truss height values.

Index	Height [m]
1	1.8
2	2.0
3	2.2
4	2.4

Table 2: CHS profile catalog.

Index	Name	Outer diameter [mm]	Wall thickness [mm]	Section area [cm^2]	Moment of inertia [cm^4]
1	CHS 21×3.2	21.3	3.2	1.82	0.77
2	CHS 26×3.2	26.9	3.2	2.38	1.70
3	CHS 33×2.6	33.7	2.6	2.54	3.09
4	CHS 33×3.2	33.7	3.2	3.07	3.60
5	CHS 33×4.0	33.7	4.0	3.73	4.19
6	CHS 42×2.6	42.4	2.6	3.25	6.46
7	CHS 42×3.2	42.4	3.2	3.94	7.62
8	CHS 42×4.0	42.4	4.0	4.83	8.99
9	CHS 48×3.2	48.3	3.2	4.53	11.59
10	CHS 48×4.0	48.3	4.0	5.57	13.77
11	CHS 48×5.0	48.3	5.0	6.80	16.15
⋮	⋮	⋮	⋮	⋮	⋮
119	CHS 660×50	660	50	958	448670

2.2. Parameters

In each iteration of the optimization process, we will modify the design state in order to move in the direction of the optimum. To this end, we have to make a well-considered design update: the update must have a beneficial impact on the objective function and/or the critical constraint. Complex constraints make it very difficult to predict the effect of a design update. It is therefore important to consider an appropriate set of candidate states in order to select the best update.

Assessing the influence of increasing or decreasing a single property of a design entity, which would typically be the section area, is not sufficient. As an example, assume that one of the chord members in the truss design problem has been assigned section 5 from table 2, and that the joint between this member and one of the braces is susceptible to chord face failure. In an attempt to improve the joint capacity, we take the section with the next larger area – section 7. However, this section has a smaller wall thickness, which is one of the determining factors for chord face failure. As a consequence, the design update does not have a beneficial effect at all on the critical constraint; on the contrary, the problem becomes worse.

This issue is solved by combining the single variable approach (to identify the state of a design entity) with a multiple variable approach (to determine the set of candidate states for the next design update). To this end, we declare certain properties of the design entities as optimization parameters. These parameters are decreased and increased one after another in order to find all neighboring design states. The states thus obtained constitute the set of candidate states.

The choice of which design entity properties are considered as parameters must be based on their expected effect on the objective function and the constraints. In the case of the example problem, we choose the values of the scalar design entities as optimization parameters, as well as the outer diameter *d* and the wall thickness *t* of the CHS profiles. The section area *A* and the moment of inertia *I* are also relevant for the objective function and some of the constraints, but since they depend directly on the diameter *d* and the thickness *t*, there is no need to consider them as separate optimization parameters.

For each design entity *e*, all possible parameter values are collected in a matrix $\mathbf{P}^e$. For the scalar design entity representing the height of the truss, $\mathbf{P}^e$ is a column vector containing the values in table 1. In design state **i**, the value of the design entity is equal to $P^e_{i_e}$. For the design entities representing a CHS profile, $\mathbf{P}^e$ is a 119 × 2 matrix containing the diameters *d* (in the first column) and the thicknesses *t* (in the second column) specified in table 2. In design state **i**, the diameter *d* and the thickness *t* of the CHS profile represented by design entity *e* are given by $d = P^e_{i_e 1}$ and $t = P^e_{i_e 2}$.

2.3. Links

Due to the existence of links, it is not always possible to find a neighboring state by simply changing a single parameter. As an example, assume that one of the CHS profiles has section 8 from table 2. If we want to increase the wall thickness *t* of this profile, we have to modify its outer diameter *d* as well. Similarly, if we want to modify the outer diameter of one of the top chord members of the example truss, we have to modify the other top chord member sections accordingly.

The neighboring design state in the direction of a certain parameter is therefore defined as the closest state to the current design state for which (1) all links are satisfied and (2) the target parameter moves in the desired direction. The distance between the current design state **i** and a candidate neighboring design state **j** is measured by means of the function *d* (**i**, **j**):

$$d(\mathbf{i},\mathbf{j}) = \sqrt{\sum_{e=1}^{M}\sum_{k=1}^{N_e}\left(\frac{P^e_{j_e k} - P^e_{i_e k}}{\hat{P}^e_{i_e k}}\right)^2} \tag{1}$$

where *M* is the number of design entities and N_e is the number of parameters associated with design entity *e*. Different parameters may have different dimensions or magnitudes; all terms in the distance function (1) are therefore divided by a reference value $\hat{P}^e_{ik}$ in order to make them dimensionless. In the examples considered in the present paper, the reference value $\hat{P}^e_{ik}$ is chosen as follows: for parameters that can only assume strictly positive values (such as the dimensions of a section), the current value P^e_{ik} is used; for parameters that may assume zero or negative values, the reference value is computed as the distance between the minimum and the maximum value:

$$\hat{P}^e_{ik} = \begin{cases} P^e_{ik} & \text{if } \mathrm{P}^e_{jk} > 0\ \forall \mathrm{j} \\ \max\limits_j P^e_{jk} - \min\limits_j P^e_{jk} & \text{otherwise} \end{cases} \tag{2}$$

Finding the design state that minimizes the distance function *d* (**i**, **j**) and simultaneously complies with the aforementioned constraints (to satisfy the links and to move in the desired direction) is a combinatorial problem that has the same dimensions as the original discrete design optimization problem. Solving this problem by means of enumeration would involve a very high computational effort. In order to reduce the dimensions of the problem, only the design entities that have an actual (direct or indirect) link with the target design entity are considered. The other design entities do not need to be updated; they simply preserve their current state. A branch-and-bound-like algorithm is then used to find the neighboring state: all terms in the distance function (1) are positive; as a consequence, we can terminate the summation operation as soon as the value corresponding to the best candidate state yet is exceeded. In this way, large numbers of combinations can be discarded, and the computational cost is somewhat reduced. As long as the number of linked parameters remains limited, this approach has proven to be viable.

3. OPTIMIZATION SCHEME

The optimality criteria method is an algorithm for continuous design optimization that has been developed around 1970 (see e.g. reference [9]). It has been conceived as an efficient alternative to mathematical programming methods, allowing for the solution of optimization problems with much larger numbers of parameters. The method is based on the formulation of the optimality conditions for the problem at hand and the derivation of an iterative updating scheme from these conditions. Originally, the iterative updating scheme was derived for each individual type of problem, based on a physical interpretation of the optimality conditions. Two decades later, Venkayya [10] presented a generalization of the method, making it directly applicable to any type of design optimization problem. For a detailed description of the generalized optimality criteria method, the reader is referred to the book of Haftka and Gürdal [6].

The optimality criteria method is not directly applicable to discrete design optimization problems. An alternative algorithm has therefore been developed, based on the same principles as the optimality criteria method, but within the framework set up in section 2.

3.1. Optimality criteria method for discrete problems

The discrete design optimization problem is formulated as follows:

$$\begin{aligned} \min_{\mathbf{i}} : \quad & f(\mathbf{i}) \\ \text{subject to :} \quad & g(\mathbf{i}) \le 1 \\ & i \in \mathcal{I}_\mathrm{a} \end{aligned} \tag{3}$$

where $\mathbf{i}$ is the design state as defined in section 4, $f(\mathbf{i})$ is the objective function, $g(\mathbf{i}) \le 1$ is the constraint, and $\mathcal{I}_\mathrm{a}$ is a set collecting all admissible design states, i.e. the design states for which the parameter links are satisfied.

This optimization problem is solved in an iterative way. Given a certain design state $\mathbf{i}^{\mathrm{old}}$, all candidate new design states $\mathbf{j}$ in the neighborhood of state $\mathbf{i}^{\mathrm{old}}$ are identified as explained in subsections 2.2 and 2.3. These states are collected in a set $\mathcal{I}_\mathrm{c}$. For each candidate state $\mathbf{j} \in \mathcal{I}_\mathrm{c}$, the objective and constraint functions $f(\mathbf{j})$ and $g(\mathbf{j})$ are evaluated and the differences $\Delta f(\mathbf{j}) = f(\mathbf{j}) - f(\mathbf{i}_{\mathrm{old}})$ and $\Delta g(\mathbf{j}) = g(\mathbf{j}) - g(\mathbf{i}_{\mathrm{old}})$ are calculated.

If the current design state $\mathbf{i}^{\mathrm{old}}$ is infeasible, i.e. if $g(\mathbf{i}_{\mathrm{old}}) > 1$, the next state $\mathbf{i}^{\mathrm{new}}$ is determined by means of the following updating rule:

$$\begin{aligned} \mathbf{i}^{\mathrm{new}} = \arg\max_{\mathbf{j}} : \quad & \frac{\Delta f(\mathbf{j})}{\Delta g(\mathbf{j}) - \varepsilon} \\ \text{subject to :} \quad & \Delta g(\mathbf{j}) \le 0 \\ & \text{if } \Delta g(\mathbf{j}) = 0 \text{ then } \Delta f(\mathbf{j}) < 0 \\ & \mathbf{j} \in \mathcal{I}_\mathrm{c} \end{aligned} \tag{4}$$

where ε is a very small positive number introduced in order to ensure that the denominator does not become zero.

If the current design state $\mathbf{i}^{\mathrm{old}}$ is feasible, i.e. if $g(\mathbf{i}_{\mathrm{old}}) \le 1$, the following updating rule is used:

$$\begin{aligned} \mathbf{i}^{\mathrm{new}} = \arg\max_{\mathbf{j}} : \quad & \frac{\Delta g(\mathbf{j})}{\Delta f(\mathbf{j}) - \varepsilon} \\ \text{subject to :} \quad & \Delta f(\mathbf{j}) \le 0 \\ & \text{if } \Delta f(\mathbf{j}) = 0 \text{ then } \Delta g(\mathbf{j}) < 0 \\ & \mathbf{j} \in \mathcal{I}_\mathrm{c} \end{aligned} \tag{5}$$

It is possible that none of the neighboring design states satisfies the restrictions specified by the updating rule; in such cases, the optimization scheme terminates. It is also possible that the updating rule has multiple solutions; if this occurs, the next design state $\mathbf{i}^{\mathrm{new}}$ is selected arbitrarily among these solutions.

The selection of the next design state according to equations (4) and (5) is visualized in figure 1. It can be explained as follows: if the current design state is infeasible, the next state is chosen so that the constraint

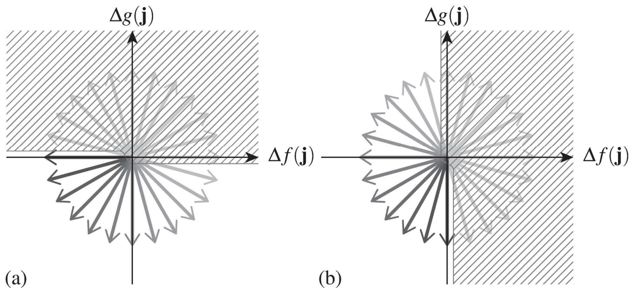


Fig. 1. These graphs illustrate how the next design state is chosen (a) if the current state is infeasible and (b) if the current state is feasible. For each candidate state, the impact on the objective function and the constraint function can be represented by a vector $\{\Delta f(\mathbf{j}); \Delta g(\mathbf{j})\}$. A state for which the vector $\{\Delta f(\mathbf{j}); \Delta g(\mathbf{j})\}$ points in the direction of a dark-colored arrow is preferred over a state for which it points in the direction of a light-colored arrow. States corresponding to vectors $\{\Delta f(\mathbf{j}); \Delta g(\mathbf{j})\}$ located in the hatched area are not allowed.

function decreases maximally while the objective function increases minimally. If the current design state is feasible, the next state is chosen so that the objective function decreases maximally while the constraint function increases minimally. In some rare cases, it might be possible to change the design such that both the constraint function and the objective function decrease; if this occurs, these changes are obviously highly preferable. Conversely, design changes for which both functions increase might also be possible; these changes will never be selected.

Iterative application of this procedure leads to a path in the parameter space that can generally be subdivided into two parts. In the first part of the optimization path, the design will move towards the boundary of the feasible domain, either from the feasible or from the infeasible side (depending on the starting point). This part is similar to the relative difference quotient algorithm for discrete optimization proposed by Chai and Sun [3]. In the classical optimality criteria method (for continuous optimization problems), this part does not exist, as a scaling operation ensures that the constraint is active throughout the entire optimization run. In the second part of the optimization path, the design will stay close to the boundary of the feasible domain and gradually move in the direction of the optimum. This part is similar to the classical optimality criteria method, in the sense that the design is improved by reducing the investment in the least cost-effective parameters in order to increase the investment in the most cost-effective parameters. The main difference with the continuous scheme is that only a single parameter (or a single family of linked parameters) is changed in each optimization step, resulting in an optimization path that continuously crosses the boundary of the feasible domain instead of exactly following it. Eventually, all parameters become equally cost-effective, which means that the optimality criteria are satisfied – the optimum is reached. At this point it is no longer possible to improve the design by saving on one parameter in order to invest in another. As a consequence, the algorithm will enter a loop. Once a loop has been detected, or once none of the neighboring design states satisfies the restrictions specified by the updating rule, the algorithm is terminated, and the optimum is identified as the design on the optimization path for which the lowest objective function value is reached while the constraint is satisfied.

3.2. Multiple constraints

The optimization scheme outlined in the previous subsection is only usable for optimization problems with a single constraint. Practical design optimization problems are usually characterized by a large number of constraints. In this subsection, a method is proposed to combine a large number of constraints of the type $g_k \le 1$ into a single constraint $g \le 1$.

The most obvious method of combining constraint functions is to take the maximum:

$$g = \max_k g_k \tag{6}$$

However, this approach would cause the optimization scheme to fail, as can be shown by means of the following simple example. Consider the structure shown in figure 2: two vertical cable segments with lengths l_1 and l_2 and sections A_1 and A_2 carry a vertical point load F. The aim is to determine the optimal cable sections A_1 and A_2. The objective function is defined as follows:

$$f = l_1 A_1 + l_2 A_2 \tag{7}$$

Two stress constraints are formulated:

$$g_1 = \frac{F / A_1}{f_y} \le 1 \tag{8}$$

$$g_2 = \frac{F / A_2}{f_y} \le 1 \tag{9}$$

where f_y is the yield strength of the material. Both constraint functions g_1 and g_2 are combined into a single constraint function g according to equation (6).

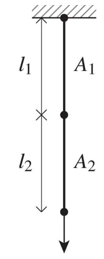


Fig. 3. Vertical cable consisting of two segments with lengths l_1 and l_2 and sections A_1 and A_2 carrying a vertical point load F.

Suppose that the design is in the infeasible domain because both A_1 and A_2 are too small. Iterative application of the updating rule given by equation (4) will first cause the smallest section to increase until both values A_1 and A_2 are equal. At that point, neither A_1 nor A_2 will increase further, as this does not affect the combined constraint function g, while it leads to an increase of the objective function f. Such modifications are not allowed by the updating rule. As a consequence, the optimization scheme is stuck.

Now suppose that the design is in the feasible domain: both A_1 and A_2 are larger than necessary. Iterative application of the updating rule given by equation (5) will cause the largest section to decrease until A_1 and A_2 are equal. At that point, either A_1 or A_2 will be arbitrarily selected for further reduction, as both lead to the same decrease of the objective function f and increase of the combined constraint function g. In the next step, the other section will be reduced, so that A_1 and A_2 become equal again. These two operations will be repeated until the infeasible domain is reached.

From this simple example, we can conclude that combining constraint functions by taking the maximum value is a suitable approach as long as all constraints are satisfied. If one or more constraints are not satisfied, it may cause the optimization scheme to fail.

An alternative approach to combine multiple constraint functions has been proposed by Kreisselmeier and Steinhauser [7]:

$$g = \frac{1}{\rho} \log\left(\sum_k \exp\left(\rho g_k\right) \right) \tag{10}$$

The Kreisselmeier-Steinhauser (KS) function is a smooth approximation of the maximum operator. The parameter $\rho > 0$ determines the accuracy and the smoothness of the approximation – for small values of ρ, the approximation is very inaccurate but smooth; for high values of ρ, the accuracy increases and the smoothness reduces; for ρ tending to infinity, the KS function becomes equal to the maximum operator.

The KS function is frequently used in continuous optimization: by virtue of its smoothness, it allows for the use of a gradient-based optimization scheme. In the frame of the discrete optimization scheme proposed in this paper, the smoothness of the function is less important. What matters here is that the KS function does not only depend on the constraint that is most strongly violated; all constraints are taken into account. As a consequence, the optimization scheme will no longer get stuck in the situation outlined above: when the KS function is used, increasing A_1 or A_2 will have a beneficial impact on the combined constraint function g; hence, this will be allowed by the updating rule. The KS function will always yield an overestimation of the maximum operator – using this function in order to combine constraints of the type $g_k \le 1$ is therefore a conservative approach. However, it may lead to a design that is not truly optimal. In order to avoid this, we modify the KS function by using a variable smoothness parameter ρ:

$$\rho = \frac{1}{\max\left(\max_k g_k, 1 \right) - 1} \tag{11}$$

As the design approaches the feasible domain, the parameter ρ will approach infinity, and the modified KS function will become identical to the maximum operator.

4. TEST CASE

In this section, the discrete optimization scheme is illustrated for a simple test case. The aim is to design a v-shaped cable structure as shown in figure 3 that carries a vertical point load F = 20 kN. The structure spans a distance $2a$ = 12 m.

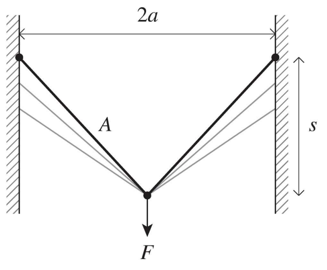


Fig. 2. Cable structure with span $2a$, sag s, and section A carrying a vertical point load F.

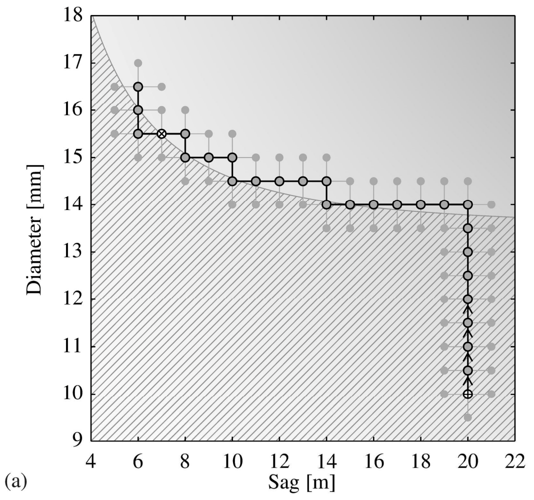
Two design entities are considered: the sag of the cable and the cable section. Possible values for both design entities are given in tables 3 and 4. The cable sag s and the section diameter d are used as optimization parameters.

Table 3: Possible values for the cable sag s.

Index	Sag [m]
1	0
2	1
3	2
⋮	⋮
31	30

Table 4: Possible cable sections.

Index	Diameter [mm]	Section area [mm^2]
1	1.0	0.79
2	1.5	1.77
3	2.0	3.14
⋮	⋮	⋮
99	50.0	1963.50



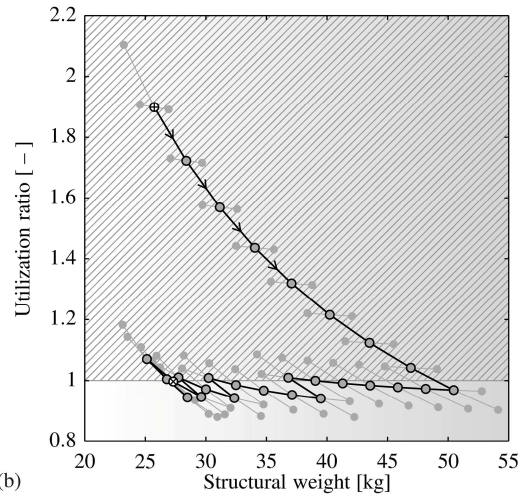


Fig. 4. (a) Sag and diameter and (b) utilization ratio and structural weight for the designs evaluated in the optimization of the v-shaped cable structure. The dot marked by a plus sign corresponds to the initial design, the dots encircled in black represent the designs that are selected in the optimization, and the dot marked by an x-sign corresponds to the optimized design. The shading reflects the objective function; lighter is better. The hatched area is the infeasible domain.

The aim is to minimize the structural weight; the objective function $f(\mathbf{i})$ is therefore defined as:

$$f(\mathbf{i}) = 2\rho A\sqrt{a^2 + s^2} \tag{12}$$

where A is the section area and $\rho = 7850$ kg/m^3 is the density of the material. A single stress constraint is formulated:

$$g(\mathbf{i}) = \frac{F\sqrt{1+\dfrac{a^2}{s^2}}}{2Af_y} \tag{13}$$

where $f_y = 700$ N/mm^2 is the yield strength of the material. The constraint function $g(\mathbf{i})$ represents the ratio of the actual stress to the maximum allowable stress and can therefore be considered as a utilization ratio.

Figure 4 shows a graphical representation of the optimization path in terms of the parameters s and d and in terms of the objective and constraint functions f and g. In the initial design state, the sag s is 20 m and the cable diameter d is 10 mm. In the optimized state, the sag s is 7 m and the diameter d is 15.5 mm. The corresponding structural weight f is 27.31 kg and the utilization ratio g is 0.997. It can be verified by enumeration that this is indeed the optimum.

The optimization path in figure 4 clearly consists of two parts; a first part where the design moves in the direction of the feasible domain, and a second part where it follows the boundary of the feasible domain until it reaches the optimum.

Figure 4b clearly visualizes the updating mechanism: if the design infeasible, we increase the investment in the most cost-effective parameter: the steepest step downwards is selected. If the design is feasible, we reduce the investment in the least cost-effective parameter: the flattest step leftwards is selected. Eventually, all steps become parallel. At this point, all parameters are equally cost-effective – the optimality criteria are satisfied; the optimum is reached.

5. TRUSS DESIGN PROBLEM

In this section, the example optimization problem described in the introduction is addressed. The aim is to minimize the self weight of a Warren type truss consisting of CHS members with welded joints. The layout of the truss as well as the loads and the boundary conditions are shown in figure 5. The truss must be designed in accordance with the displacement, member force, and buckling constraints specified in part 1-1 of Eurocode 3 [4]. In addition, the rules in the CIDECT design guide for CHS joints must be followed [11]. The truss dimensions, steel grade, and load level are inspired by one of the worked-out design examples in the CIDECT guide: the span is l = 36 m and the height at mid-span is h_1 = 2.4 m. S355 steel is used for the chords and S275 steel for the braces. The loading consists of the purlin loads F and the self weight of the truss. The unfactored value of the purlin loads is F = 68 kN; the factored value is F = 100 kN. The safety factor for the self weight is 1.35. Factored load values are used for the analysis of the ultimate limit state (to check the capacities of members and joints), unfactored values for the serviceability limit state (to check the deflections).

The optimization problem is formulated in terms of 15 design entities: the height h_2 of the truss at end-span, 3 top chord sections (we aim for a symmetric structure), 4 bottom chord sections, and 7 brace sections. The height h_2 must take one of the values in table 1; the sections must be chosen from the catalog in table 2. In order to ensure that the optimized design is buildable, the top chord members must all have the same outer

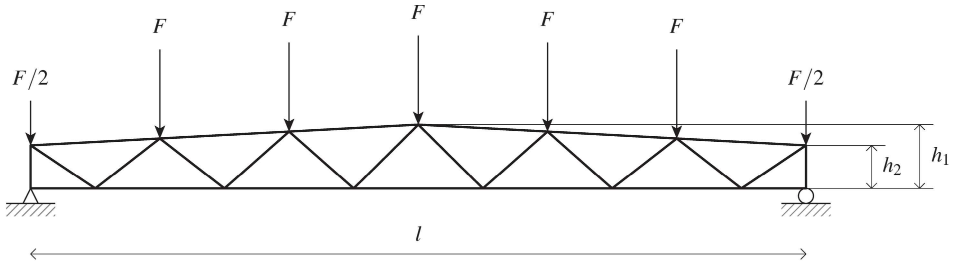


Fig. 5. Simply supported Warren type truss with a span l and a height varying from h_1 at mid-span to h_2 at the ends, subjected to 7 vertical point loads F and F/2.

diameter, and the bottom chord members must all have the same outer diameter.

First, the initial design state is chosen. In an attempt to choose realistic values, the height h_2 at end-span is set to 2 m, and CHS 193×5.0 profiles are selected for all members. The corresponding structural weight f is 2797 kg, and the overall utilization ratio g is 6.403.

Next, the discrete optimality criteria based algorithm is used to optimize the design. Figure 6 shows the structural weight and the utilization ratio for all designs evaluated in the optimization process. After 65 iterations, a loop is detected, and the optimization scheme is terminated. At this point, a total number of 2815 designs has been evaluated. This is comparable with the number of design evaluations required for continuous design optimization problems where the gradients of the objective function and the constraints are computed with finite differences [1]. It is at least two orders of magnitude lower than the number of design evaluations required for similar optimization problems solved with an evolutionary algorithm [1].

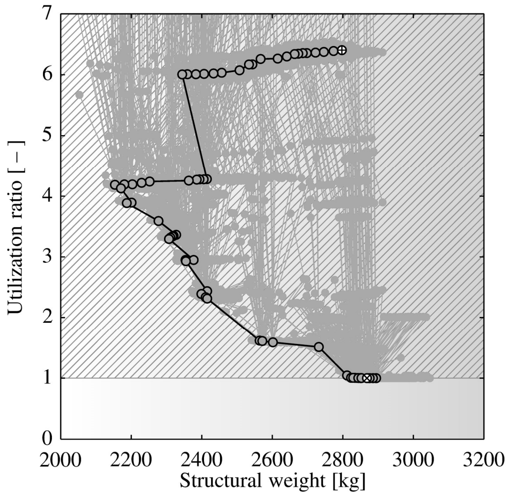


Fig. 6. Utilization ratio versus structural weight for the designs evaluated in the optimization of a Warren type truss.

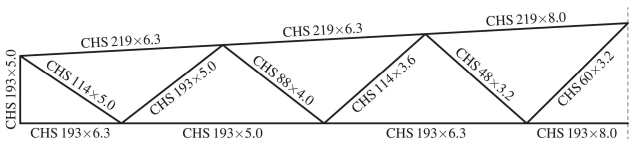


Fig. 7. Optimized member sections obtained with the optimality criteria based algorithm.

In the optimized state, the height h_2 is equal to 2 m. The section profiles for all members are shown in figure 7. The top chord members all have the same outer diameter, as requested. Their wall thickness increases towards mid-span – where the bending moment in the equivalent beam reaches a maximum. The bottom chord members have a smaller diameter than the top chord members – they do not need buckling resistance. The wall thickness also increases towards mid-span, but the leftmost member has a larger thickness than its neighbor – this is explained by the increased risk for chord face failure of the leftmost bottom joint due to a strong lateral compression of the chord caused by the reaction force. The brace members have varying diameters. The braces in compression have a larger section than the braces in tension due to the risk of buckling. Sections increase towards end-span, where the shear force in the equivalent beam reaches a maximum. The structural weight f of the optimized design is equal to 2868 kg. The overall utilization ratio g is exactly equal to 1 due to the fact that the diameter of some of the braces is equal to the diameter of the bottom chord – as specified by the CIDECT rules, this is the largest allowable brace diameter.

6. CONCLUSION

This paper presents a new algorithm for discrete design optimization under buildability constraints. The algorithm is based on the optimality criteria method. It proceeds as follows: given a certain design state, all neighboring design states that satisfy the links between optimization parameters are identified. For each of these neighboring states, the change in cost and the change in performance with respect to the current state are evaluated. If the current state is infeasible, the next state is chosen so that performance increases maximally while cost increases minimally. If the current state is feasible, the next state is chosen so that cost decreases maximally while performance decreases minimally. This operation is applied iteratively until it gets trapped in a loop; at this point, the optimum is found. The new algorithm is successfully applied to a simple test case, as well as a realistic optimization problem involving the design of a Warren type truss composed of steel CHS members with welded joints. In terms of numerical efficiency (the number of design evaluations), the new algorithm is comparable with continuous optimization schemes based on finite differences and much more efficient than evolutionary algorithms.

ACKNOWLEDGEMENTS

This paper is the product of a research stay of the first author at the Arup office in Amsterdam. The research stay has been financially supported by the Research Foundation – Flanders. The first author is a member of OPTEC – the KU Leuven Optimization in Engineering Center (KU Leuven-BOF PFV/10/002).

REFERENCES

[1] Optimization of shell and truss structures based on size and shape parameterization. PhD thesis, Department of Civil Engineering, KU Leuven, 2012.

[2] J.S. Arora and M.W. Huang. Discrete structural optimization with commercially available sections. Structural Engineering/Earthquake Engineering, 13(2):105–122, 1996.

[3] S. Chai and H.C. Sun. A relative difference quotient algorithm for discrete optimization. Structural Optimization, 12(1):46–56, 1996.

[4] European Committee for Standardization. Eurocode 3: Design of steel structures - Part 1-1: General rules and rules for buildings, 2005.

[5] European Committee for Standardization. European standard EN 10210-2: Hot finished structural hollow sections of non-alloy and fine grain steels - Part 2: Tolerances, dimensions and sectional properties, 2006.

[6] R.T. Haftka and Z. Gürdal. Elements of structural optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, 3rd edition, 1992.

[7] G. Kreisselmeier and R. Steinhauser. Application of vector performance optimization to a robust-control loop design for a fighter aircraft. International Journal of Control, 37(2):251–284, 1983.

[8] G.N. Vanderplaats. Structural optimization for statics, dynamics and beyond. Journal of the Brazilian Society of Mechanical Sciences and Engineering, 28(3):316–322, 2006.

[9] V.B. Venkayya. Design of optimum structures. Computers & Structures, 1(1):265–309, 1971.

[10] V.B. Venkayya. Optimality criteria: a basis for multidisciplinary design optimization. Computational Mechanics, 5(1):1–21, 1989.

[11] J. Wardenier, Y. Kurobane, J.A. Packer, D. Dutta, and N. Yeomans. Design guide for circular hollow section (CHS) joints under predominantly static loading. CIDECT and Verlag TÜV Rheinland GmbH, 1992.