

Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of some Toulouse researchers and makes it freely available over the web where possible.

This is an author's version published in: https://oatao.univ-toulouse.fr/19389

Official URL: http://dx.doi.org/10.1007/978-3-319-67988-4_140

To cite this version :

Barjhoux, Pierre-Jean and Diouane, Youssef and Grihon, Stéphane and Bettebghor, Dimitri and Morlier, Joseph Mixed Variable Structural Optimization: Toward an Efficient Hybrid Algorithm. (2017) In: 12th World Congress on Structural and Multidisciplinary Optimisation, 5 June 2017 - 9 June 2017 (Braunschweig, Germany).

Any correspondence concerning this service should be sent to the repository administrator: tech-oatao@listes-diff.inp-toulouse.fr

12th World Congress on Structural and Multidisciplinary Optimisation
05th - 09th, June 2017, Braunschweig, Germany

Mixed Variable Structural Optimization: toward an Efficient Hybrid Algorithm

Pierre-Jean Barjhoux[1], Youssef Diouane[2], Stéphane Grihon[3], Dimitri Bettebghor[4], Joseph Morlier[5]

[1] IRT Saint Exupery, Toulouse, France, pierre-jean.barjhoux@irt-saintexupery.com
[2] ISAE-SUPAERO, Toulouse, France, youssef.diouane@isae.fr
[3] AIRBUS, Toulouse, France, stephane.grihon@airbus.com
[4] ONERA, Châtillon, France, dimitri.bettebghor@onera.fr
[5] Université de Toulouse, Institut Clément Ader (ICA), CNRS-ISAE SUPAERO-INSA-Mines Albi-UPS, Toulouse, France, joseph.morlier@isae.fr

1. Abstract

Designing a structure implies a selection of optimal concept and sizing, with the aim of minimizing the weight and/or production cost. In general, a structural optimization problem involves both continuous variables (e.g., geometrical variables, …) and categorical ones (e.g., materials, stiffener types, …). Such a problem belongs to the class of mixed-integer nonlinear programming (MINLP) problems. In this paper, we specifically consider a subclass of structural optimization problems where the categorical variables set is non-ordered. To facilitate categorical variables handling, design catalogs are introduced as a generalization of the stacking guide used for composite optimization. From these catalogs, a decomposition of the MINLP problem is proposed, and solved through a branch and bound method. This methodology is tested on a 10 bars truss optimization inspired from an aircraft design problem, consistent with the level of complexity faced in the industry.

2. Keywords: mixed optimization, structural optimization, branch and bound, categorical variable, multi-material.

3. Introduction

In the field of aircraft design, structural optimization involves different kinds of variables. Continuous variables describe the size of aircraft structural parts: in case of thin-sheet stiffened sizing, they represent panel thicknesses and stiffening cross-sectional areas. The set of discrete variables encompasses two kinds of variables. First, variables that are ordered, e.g. a list of cross-sectional areas. Second, technological choices or even computation assumptions (e.g. material choices, buckling margin policy) are depicted by categorical variables.

Many industrial problem formulations natively involve a mixture of continuous and discrete variables. Mixed-variable optimization algorithms have been proposed, such as Genetic Algorithms [1], Differential Evolution [2], Particle Swarm Optimization [3] and Pattern Search [4]. In many cases, discrete variables are considered as ordered through a continuous relaxation approach, for example in [5][6][7]. In other cases, categorical variables are handled through a native mixed-variable optimization approach for both categorical and continuous variables without relaxation [8][9]. Other strategies consist in turning categorical variables into continuous ones, allowing the use of continuous optimization algorithms. For example, continuous indices containing shape and material information are built [10], meta-models are constructed through Multiple Kernel Regression [11].

In industry, most structural optimization software solutions mainly handle continuous variables and ignore ordinal and categorical variables. This is due to the difficulty of managing all variable types simultaneously while maintaining the optimization performance: discrete variables always introduce a combinatorial behavior leading to an exponential number of configurations to be explored. For instance, at Airbus, two structural optimization software were developed to split up the problem. A first one is inspired by classical continuous optimization approaches using gradients. The second one focuses on discrete and categorical variables in a preliminary phase of the design process where trade-offs are evaluated [12]. On this basis, major decisions governing the structural definition are made.

In this article, the work related to mixed variable optimization will be presented in section 4, and an industrial formulation is proposed in section 5. Then, the formulation of a test case is introduced in section 6. The section 7 is dedicated to the description of the proposed methodology, followed by an analysis of numerical results.

4. Related work

To our knowledge, an open problem is to define an efficient and generic algorithm combining both categorical and continuous variables for solving multi-concept structural optimization problems. In industry, attempts have been made to tackle this kind of optimization problems. One can cite the PRESTO approach, developed at Airbus [13], that is a pure discrete approach to manage both sizing and categorical variables. However, sizing variables (e.g., thicknesses, areas) are treated as discrete ones. In this setting, a value of a categorical variable points to a

choice among catalogs for a single element or group of elements.
In the literature, a catalog for an element can be considered as a guide to a given composite panel [14][15][16]. Composite guide approach has been extended to stiffening profiles as illustrated in Figure 1. On top of stacking sequences, the profile dimensions can be varied according to the area. For metal, a skin catalog reduces to a list of thicknesses and a stiffener catalog to a list of profiles.

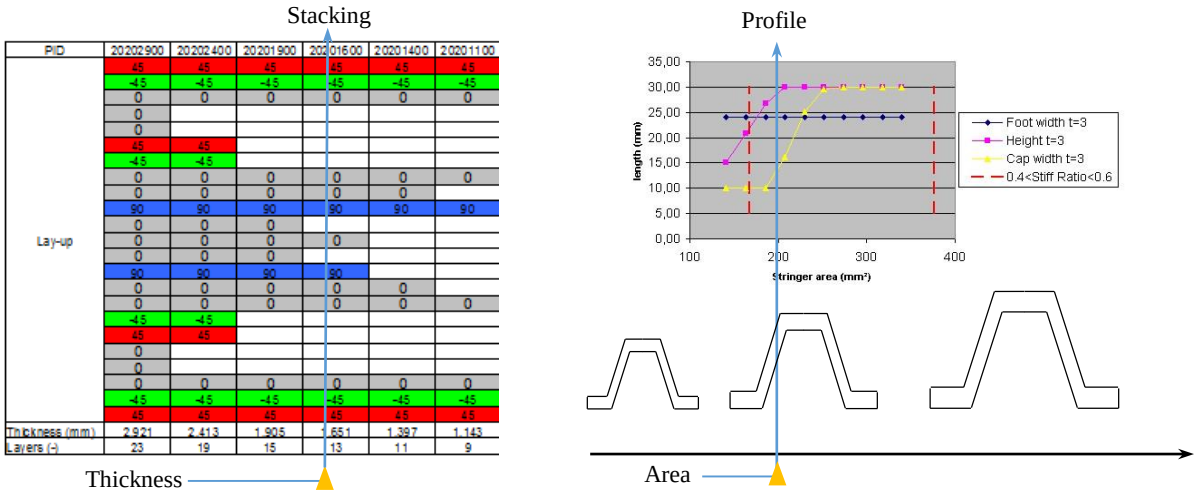


Figure 1: catalogs principle, where the links between the thickness and stacking sequences, area and detailed variables, are depicted.

Indeed, for PRESTO, the catalogs allow for mapping detailed variables values with a given couple of thickness and area values. This means that a choice of thickness is equivalent to a choice of a stacking, and a choice of area corresponds to a choice of scaled profile (including stacking). Furthermore, a change in stacking sequence (symmetric, balanced, oriented, iso) leads to a change of catalog choice. Using the same approach, information on material, stiffening profile or even buckling margin policy choices have been added to catalogs. Indeed, results of ISAMI, the reference strength analysis package of Airbus used for sizing and airframes certification, have been stored into the PRESTO database. So in addition to the sizing variables link mentioned above, a catalog contains a table of Reserve Factors (RF) corresponding to predefined internal forces. This RF aims at measuring the strength of an element: a RF greater than 1 illustrates that the element does hold the given load.

For PRESTO, the sizing is performed separately for each element by finding, through the catalog database, the skin and stiffener parameters (i.e.; super-stiffener) that minimize the weight with respect to internal loads (IL). Considering that the catalog set describes a finite set of choices, this local (spatially speaking, in the structural model) optimization is straightforward and consists of a simple loop that carries out a numerical sorting to find the weight minimum feasible super-stiffener as illustrated in Figure *2*.

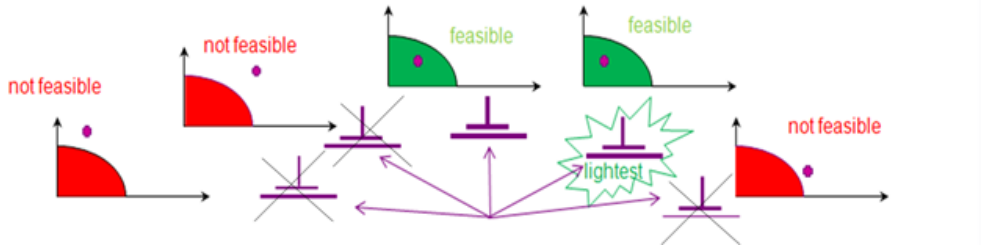


Figure 2: sizing engine of PRESTO

Since the choice of catalog is performed separately for each element, it does not take into account overall effects of optimal internal forces redistribution, or rigidity constraints. Both sizing and categorical variables are concerned by this drawback. The main objective of this work consists in improving the existing industrial PRESTO approach by setting up an efficient and performing methodology that tackles directly both categorical and sizing variables (as continuous variables) using a new formulation of the problem.

In order to implement a generic approach as far as possible, no distinction is made between material and stiffening principle selection. To that end, a particular attention has been paid to avoid methods that are focused

on specific categorical variables (e.g. dedicated to material, stacking sequences). Our methodology relies on PRESTO catalogs to ensure a unified approach handling all the categorical variables.

5. Industrial problem formulation

Let us name $\boldsymbol{t}$ the thicknesses and $\boldsymbol{A}$ the areas where $(\boldsymbol{t}, \boldsymbol{A}) \in \mathbb{R}^{2N}$, N being the number of elements in the structural model. The set Γ is an enumerated set which allows to list choices $\boldsymbol{c}$, for example the material or the stiffener type of an element. If m is the number of catalogs, we can note $\Gamma = [\![1, m]\!]$.

All these variables are vectors for which each component is associated to an element. In our industrial case, a finite element model involves about N=100 elements leading to 200 continuous variables, and 10 up to 100 categorical choices per element. The categorical variable can take a value among m^N (number of categorical choices per element at the power of the number of elements) values of the set Γ^N. This high combinatory dimension demonstrates the need for a methodology to solve efficiently such problems.

Within the framework of stiffened panels, as key concept of aeronautics structures, the optimization problem to solve is:

$$\begin{aligned} \min_{\substack{(\boldsymbol{t},\boldsymbol{A}) \in \mathbb{R}^{2N} \\ \boldsymbol{c} \in \Gamma^N}} \quad & W(\boldsymbol{c}, \boldsymbol{t}, \boldsymbol{A}) \\ \text{Subjected to:} \quad & \boldsymbol{RF}(\boldsymbol{c}, \boldsymbol{t}, \boldsymbol{A}, IL(\boldsymbol{c}, \boldsymbol{t}, \boldsymbol{A})) \geq 1 \\ & G(\boldsymbol{c}, \boldsymbol{t}, \boldsymbol{A}) \leq 0 \\ & \underline{t}(\boldsymbol{c}) \leq \boldsymbol{t} \leq \overline{t}(\boldsymbol{c}) \\ & \underline{\alpha}(\boldsymbol{c}) \leq \frac{\boldsymbol{A}}{b\boldsymbol{t}} \leq \overline{\alpha}(\boldsymbol{c}) \end{aligned} \tag{1}$$

In this formulation, the constraints on $\boldsymbol{RF}$ ensure the structural strength element per element. G represents any constraint, especially stiffness constraint that depends on all thicknesses and areas through the stiffness matrix (e.g. displacement). Bounds on sizing variables, in particular on the stiffening ratio α (of areas over thicknesses), prevent over-stiffened stiffeners supporting thin skin, or the reverse (design rule). Note that both continuous variables (i.e.; thickness $\boldsymbol{t}$, area $\boldsymbol{A}$) and categorical variables (i.e.; $\boldsymbol{c}$, hiding material or stiffening principle choices) have a significant role in this problem. The categorical variables affect the weight W, internal loads IL, rigidity constraints G, reserve factors $\boldsymbol{RF}$, bounds on thicknesses and stiffening ratio. On the other hand, continuous variables affect the weight, internal loads, reserve factors and rigidity constraints. It is worth noting that a change in a categorical variable, thickness or area will modify IL distribution along the structure. Since the stresses and then $\boldsymbol{RF}$ require the value of IL, each component of $\boldsymbol{RF}$ vector depends on the whole structure description. Internal loads IL are computed using a finite element model (FEM).

6. Test case formulation

Before tackling the targeted industrial case that deals with aeronautics box-section structures, a preliminary step has consisted in the implementation of a test-case that can be handled within short execution-times. This makes possible the exploration of new algorithms and allows for identifying promising leads.

To that end, the well-known 10 bars truss optimization test-case, introduced as a continuous optimization problem in [17][18][19], has been adapted to match the needs of the mixed optimization problem.

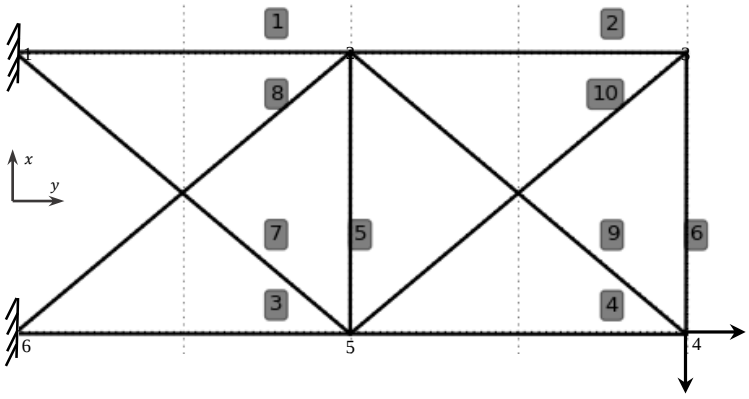


Figure 3: 10 bars truss structure working in tension/compression, with 6 nodes

The continuous optimization part of the problem remains unchanged, while categorical variables have been introduced. As described in the whole problem in Eq.(1), a choice of categorical value corresponds to a choice of material or stiffening principle.
Thus, the problem becomes:

$$\begin{aligned} \min_{\substack{\boldsymbol{A} \in \mathbb{R}^{10} \\ \boldsymbol{c} \in I^{10}}} \quad & W(\boldsymbol{c}, \boldsymbol{A}) \\ \text{Subjected to:} \quad & \boldsymbol{RF_i}\big(c_i, A_i,\ IL(\boldsymbol{c}, \boldsymbol{A})\big) \geq 1, \quad \forall i \in [\![1,\ 10]\!] \\ & G(\boldsymbol{c}, \boldsymbol{A}) \leq 0 \\ & \underline{\boldsymbol{A}}(\boldsymbol{c}) \leq \boldsymbol{A} \leq \overline{\boldsymbol{A}}(\boldsymbol{c}) \end{aligned} \tag{2}$$

Many dummy variables are hidden by categorical variables. For the sake of clarity, Table 1 gives the dependencies between them.

Table 1: variables description and links with the functions of the optimization problem

	c							
	Shape	*Material*				*Cont. variables*		
		ρ	E	σ_{allow}	ν	A	x	*IL*
Weight		•				•		
Constraints (compr/tension)				•				•
Constraints (loc/buckling)	•		•		•	•	•	•
Constraints	•		•	•	•	•	•	•
Internal Loads (IL)			•			•		

As in the whole problem in Eq.(1), the i^{th} component of the vector $\boldsymbol{c}$ corresponds to a choice of material and shape for the i^{th} bar. For each section, and regarding cross-section sizing variables, catalogs establish the link between areas and detailed variables (geometrical description of the cross sections). The PRESTO approach is replicated here, except that the sizing variables (areas, detailed variables) are considered as continuous.

Based on the hypothesis that for each bar, the vector of detailed variables $\boldsymbol{x}_i$ of the i^{th} bar varies linearly with respect to the detailed variables $\boldsymbol{x}_0$ of the reference profile:

$$\boldsymbol{x}_i = \lambda_i\, \boldsymbol{x}_0, \qquad i \in [\![1,\ 10]\!] \tag{3}$$

The corresponding cross-sectional area can then be written as:

$$A_i = {\lambda_i}^2\, A_0, \qquad i \in [\![1,\ 10]\!] \tag{4}$$

From (3) and (4) can be deduced an expression of λ_i, the scaling ratio of the profiles:

$$\lambda_i = \sqrt{A_i / A_0}\,, \qquad i \in [\![1,\ 10]\!] \tag{5}$$

Thus, the simple mathematical function detailed below can be used compute the detailed variables with respect to areas A_i coming from the optimizer, and the reference profile described by A_0 and $\boldsymbol{x}_0$:

$$\boldsymbol{x}_i(A_i) = \sqrt{A_i / A_0}\ \boldsymbol{x}_0, \qquad i \in [\![1,\ 10]\!] \tag{6}$$

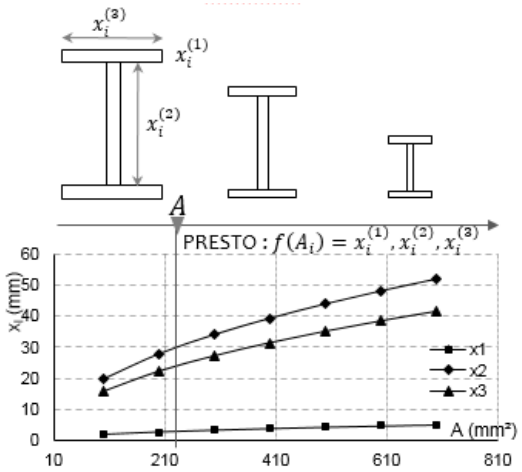


Figure 4: continuous profile scaling

These detailed variables are indeed useful for the computation of $\boldsymbol{RF_i}$ components that are related to local buckling.

Constraints $\boldsymbol{RF_i}$ are vector-valued functions of dimension 4. Indeed, for each bar i, two constraints compare the stress σ_i coming from FEM computation to the allowable stress in tension and compression (material property), two other constraints compare the stress from FEM to the local and Euler buckling allowable stress. The expressions of these constraints is as follows:

$$\boldsymbol{RF}_i\big(c_i, A_i,\ IL(\boldsymbol{c},\boldsymbol{A})\big) = \begin{bmatrix} \left(\sigma_i^{allow.compr}(c_i) - \sigma_i(\boldsymbol{c},\boldsymbol{A})\right) \Big/ \sigma_i^{allow.compr}(c_i) \leq 0 \\ \left(\sigma_i(\boldsymbol{c},\boldsymbol{A}) - \sigma_i^{allow.tension}(c_i)\right) \Big/ \sigma_i^{allow.tension}(c_i) \leq 0 \\ \left(\sigma_i^{Euler.buckl}(c_i, A_i) - \sigma_i(\boldsymbol{c},\boldsymbol{A})\right) \Big/ \sigma_i^{Euler.buckl}(c_i, A_i) \leq 0 \\ \left(\sigma_i^{Local.buckl}(c_i, A_i) - \sigma_i(\boldsymbol{c},\boldsymbol{A})\right) \Big/ \sigma_i^{Local.buckl}(c_i, A_i) \leq 0 \end{bmatrix},\ i \in [\![1,\ 10]\!] \tag{7}$$

with:

$$\sigma_i^{Local.buckl}(c_i, A_i) = \frac{4\pi^2 E_{c_i}}{12(1-{\nu_{c_i}}^2)} \left(\frac{x_i^{(1)}(A_i)}{x_i^{(2)}(A_i)}\right)^2 \tag{8}$$

$$\sigma_i^{Euler.buckl}(c_i, A_i) = \frac{\pi^2 E_{c_i} I_{c_i}(\boldsymbol{x_i}(A_i))}{A_i\, L^2} \tag{9}$$

In (7), $\sigma_i^{allow.tension}$ and $\sigma_i^{allow.compr}$ are the allowable values in terms of tension and compression respectively. They are directly linked to the choice of material, whereas $\sigma_i^{Local.buckl}$ and $\sigma_i^{Euler.buckl}$ depend on both shape (through detailed variables $x_i^{(1)}(A_i)$, $x_i^{(2)}(A_i)$ and quadratic moment $I_{c_i}(\boldsymbol{x_i}(A_i))$) and material choices. The structural constraints computed by FEM are noted σ_i. All allowable constraints value are defined as positive.

In this simplified test-case, the catalog is made of two aluminum alloys with different physical properties: one has a higher young modulus, the other higher allowable values. This is typically the case of a metallic wing, where the extrados working under compression requires a higher young modulus, whereas the intrados part needs higher fatigue allowable summarizing other properties than stiffness.

7. Methodology

As a first approach it was necessary to build a baseline methodology. For that purpose, we chose the branch and bound algorithm, whose guiding principle is mainly due to [20], [21]. Many authors improved this method afterwards, for example in the field of structural optimization [22].

7.1. Generic branch and bound process

Based on an enumerative divide-and-conquer technique, this generic approach searches for the best solution into the whole space of solutions for a given problem. The algorithm repeatedly splits the original design space into several design spaces, this is called "branching". A diagram of the branches can be seen as a "tree", while the subspaces and their associated sub-problem are illustrated by "nodes" of this tree. A sub-problem is then generated for each subspace. For each sub-problem, a lower bound (if the original objective is to get a minimum) of a feasible solution is computed. This corresponds to the "bound" step. Relying on this evaluation, all active sub-problems that meet the following two criteria are deleted. First, a design sub-space that does not contain allowable values has no reason to be further explored. Secondly, the subspace may not contain a solution better than the best known solution (called upper bound). Since an exploration of this subspace would not lead to the optimal solution, this part of the design space is discarded.

Here are the main steps of this algorithm:

0. *Initialization* : let be P_0 the initial problem, $\bar{b}(P)$ the upper bound of the full problem P and equal to the best known solution of P (or $+\infty$ if no best solution known), Q the queue containing active nodes and equal to $\{P_0\}$
1. *Branch*: select the best problem ${P_i}^{(j)}$ according to a given strategy among all active nodes from Q, and delete it from Q. Split the design space of problem ${P_i}^{(j)}$ into m subspaces. Build all associated sub-problems ${P_i}^{(j)}$ and add them to Q, with a priority following a given strategy.
2. *Bound*: compute lower bound $\underline{b}\left({P_{i+1}}^{(j)}\right)$ of ${P_{i+1}}^{(j)}$ through continuous relaxation of ${P_{i+1}}^{(j)}$ noted $\widehat{{P_{i+1}}^{(j)}}$. If the problem is infeasible, $\underline{b}\left({P_{i+1}}^{(j)}\right) \coloneqq -\infty$.
    a. If $\underline{b}({P_{i+1}}^{(j)}) > \bar{b}(P)$, ${P_{i+1}}^{(j)}$ node is removed from Q
    b. If ${P_{i+1}}^{(j)}$ not feasible, ${P_{i+1}}^{(j)}$ node is removed from Q
    c. If $\underline{b}\left({P_{i+1}}^{(j)}\right) \leq \bar{b}(P)$ and $\underline{b}\left({P_{i+1}}^{(j)}\right)$ is an integral feasible solution of P, update $\bar{b}(P)$: $\bar{b}(P) \coloneqq \underline{b}\left({P_{i+1}}^{(j)}\right)$. If Q is not empty, go to step 1,

The index j of ${P_i}^{(j)}$ reminds that at each branching step, we have to manage m sub-problems, each of them being associated with a choice among m catalogs (index $j \in [\![1, m]\!]$). ${P_i}^{(j)}$ is an instance of a problem P_i where the catalog j has been chosen for the current bar i. For simplification purposes, ${P_i}^{(j)}$ and P_i will be used interchangeably later in this article.

7.2. The developed methodology

In the context of structural optimization, at each level of the tree, the categorical variable corresponding to a bar is chosen. Thus at level i of the tree, $i-1$ categorical variables have been fixed by the previous i categorical variables associated to the first i bars are fixed. The branch and bound process generates, at each branching step, as many subspaces as the number of available catalog choices (i.e.; m). This formulation is inspired from the generalized Benders [23] decomposition, that allows to treat independently complicating variables (in our case, categorical variables).

The problem formulation in Eq.(2) at a level i of the branch and bound tree is the following:

$$
\begin{aligned}
\text{(Problem } P_i\text{)} \qquad & \min && W(\boldsymbol{c}, \boldsymbol{A}) = \sum_{k=1}^{i} W_k(\bar{\bar{c}}_k, A_k) + \sum_{k=i+1}^{n} W_k(c_k, A_k) \\
& \boldsymbol{A} \in \mathbb{R}^{10} && \\
& \boldsymbol{c} \in [\![1,\ m]\!]^{10-i} && \\
& \text{Subjected to:} && \boldsymbol{RF_k}\left(\bar{\bar{c}}_k, A_k,\ IL(\bar{\bar{\boldsymbol{c}}}, \boldsymbol{c}, \boldsymbol{A})\right) \geq 1, \quad \forall k \in [\![1,\ i]\!] \\
& && \boldsymbol{RF_k}\left(c_k, A_k,\ IL(\bar{\bar{\boldsymbol{c}}}, \boldsymbol{c}, \boldsymbol{A})\right) \geq 1, \quad \forall k \in [\![i+1,\ 10]\!] \\
& && G(\bar{\bar{\boldsymbol{c}}}, \boldsymbol{c}, \boldsymbol{A}) \leq 0 \\
& && \underline{\boldsymbol{A}}(\bar{\bar{\boldsymbol{c}}}) \leq \boldsymbol{A} \leq \overline{\boldsymbol{A}}(\bar{\bar{\boldsymbol{c}}}) \\
& && \underline{\boldsymbol{A}}(\boldsymbol{c}) \leq \boldsymbol{A} \leq \overline{\boldsymbol{A}}(\boldsymbol{c})
\end{aligned}
\tag{10}
$$

All fixed variables from upper levels are noted with double over-bar accent: for example, $\overline{\overline{\boldsymbol{c}}}$ is the i-dimensional vector of fixed categorical values. As in the 10 bars truss problem description, $\boldsymbol{A}$ is the vector of continuous design variable describing the areas, invariably of dimension 10 during the branch and bound process: the areas of all the bars are design variables regardless of the position in the tree. A breakdown of the objective function shows that two terms are involved. The first one gives the weight of the truss made up of the i first bars. It only depends on the continuous variables: the areas. The second corresponds to the weight of the remaining bars of the truss: from $i+1$ to 10. This last term depends on two variables: the categorical variables that have not been fixed already, and the areas corresponding to the same undetermined bars from $i+1$ to 10. Since the evaluation of internal loads depends on all the categorical variables and areas, both fixed and undetermined categorical variable computation are required ($\overline{\overline{\boldsymbol{c}}}$ and $\boldsymbol{c}$ respectively) in addition to all the areas.

Although branching is performed on categorical variables associated to the i first bars, there is still a dependency to the categorical variables linked to the remaining bars. Even if the categorical variable dimension of the generated subspaces decreases when branches depth increases, the computation of the associated sub-problems still requires to handle the categorical variable. Indeed, computation of constraints $\boldsymbol{RF_k}$, even for $k \in [\![1,\ i]\!]$, requires categorical choices of the whole structure through internal load computation. One major advantage of the separation and evaluation process is that no exact solution of the sub-problems is required. Actually, only a lower bound of the optimum has to be provided. Since the convergence speed of the branch and bound algorithm is directly driven by its accuracy, a specific attention has to be paid to the lower bound computation process.

Now that the formulation of a sub-problem has been described, the computation of a lower bound $\underline{b}(P_i)$ of such problem can be detailed. A lower bound of an optimization problem could, for example, be provided by removing one or several constraints from the problem. In the context of the branch and bound algorithm, it would consist in adding progressively the constraints, from one level of the tree to another, that can be computed when one more categorical variable is fixed. This would ensure to get a lower bound of the problem, while avoiding the complexity linked to categorical variables. However in the case of problem P_i in Eq.(10), each $\boldsymbol{RF_k}$ computation depends on internal loads, and thus to the whole vector of categorical variables (including the undetermined components). In this paper, two different optimization problems formulation that allow to compute a lower bound are detailed.

7.2.1. lower bound computation problem #1

The optimization constraints linked to buckling have been removed from the optimization problem, as a first approach. Indeed, the removal of the buckling $\boldsymbol{RF_k}$ makes the optimization problem independent from cross-sectional shape choices. On the other hand, whereas the internal loads computation requires categorical variables, two properties of the material information are still needed: the Young modulus, and the allowable constraint in tension and compression. Taking the highest allowable constraint value in Eq.(12) leads to a relaxed problem, where the resulting optimum weight is a lower bound of the sub-problem. Concerning the objective function, only density is required through material component of the categorical choices. Also, applying the lowest density of the available materials to the undetermined bars ensures then to get a lower bound of the truss weight. Finally, the lower bound process comes to a weight optimization with respect to all areas and the young modulus of the remaining bars. This optimization problem is the following one:

(Problem $\widehat{P_l}$)

$$
\begin{aligned}
\min_{\substack{\boldsymbol{A} \in \mathbb{R}^{10} \\ \boldsymbol{E_c} \in \mathbb{R}^{10-i}}} \quad & W(\boldsymbol{E_c}, \boldsymbol{A}) = \sum_{k=1}^{i} W_k(\rho_{\overline{\overline{c_k}}}, A_k) + \sum_{k=i+1}^{n} W_k(\rho_{min}, A_k) \qquad (11) \\
\text{Subjected to:} \quad & \boldsymbol{RF_k}\left(E_{\overline{\overline{c_k}}}, A_k, IL(\boldsymbol{E}_{\overline{\overline{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \geq 1, \quad \forall k \in [\![1,\ i]\!] \\
& \boldsymbol{RF_k}\left(E_{c_k}, A_k, IL(\boldsymbol{E}_{\overline{\overline{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \geq 1, \quad \forall k \in [\![i+1,\ 10]\!] \\
& G(\boldsymbol{E}_{\overline{\overline{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A}) \leq 0 \\
& \underline{A_k}(\overline{\overline{c_k}}) \leq A_k \leq \overline{A_k}(\overline{\overline{c_k}}), \ \forall k \in [\![1,\ i]\!] \\
& \underline{A_{min}} \leq A_k \leq \overline{A_{max}}, \ \forall k \in [\![i+1,\ 10]\!] \\
& \underline{E_{min}} \leq E_{c_k} \leq \overline{E_{max}}, \ \forall k \in [\![i+1,\ 10]\!]
\end{aligned}
$$

$\underline{A_{min}}$ and $\overline{A_{max}}$ are the lowest lower bound and maximum upper bound on area of all catalogs respectively: $\underline{A_{min}} = \min_{\forall k}(\underline{A_k}(\overline{\overline{c_k}}))$, $\overline{A_{max}} = \max_{\forall k}(\overline{A_k}(\overline{\overline{c_k}}))$. The bounds on Young modulus $\underline{E_{min}}$ and $\overline{E_{max}}$ are defined similarly: $\underline{E_{min}} = \min_{\forall k}(\underline{E_{c_k}}(\overline{\overline{c_k}}))$, $\overline{E_{max}} = \max_{\forall k}(\overline{E_{c_k}}(\overline{\overline{c_k}}))$. Since the buckling constraints have been removed

from the problem P_i in Eq.(10), the constraints $\boldsymbol{RF_k}$ are 2 dimensional: only constraints on allowable in tension and compression are included. Thus, in the context of Eq.(11), Eq.(7) becomes, for all bars that are currently not defined:

$$\boldsymbol{RF}_k(IL(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})) = \begin{bmatrix} \left(\sigma_{max}^{allow.compr} - \sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \Big/ \sigma_{max}^{allow.compr} \leq 0 \\ \left(\sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A}) - \sigma_{max}^{allow.tension}\right) \Big/ \sigma_{max}^{allow.tension} \leq 0 \end{bmatrix}, \quad \forall k \in [\![i+1,\ 10]\!] \tag{12}$$

where $\sigma_{max}^{allow.compr}$ and $\sigma_{max}^{allow.tension}$ are the highest allowable constraints available into the set of catalogs.

7.2.2. lower bound computation problem #2

In this second approach, the buckling constraints are not removed from the optimization problem: as in 7.2.1 with the constraints on allowable stresses, the material values and shape are chosen so that the limit stress for buckling described in (8) and (9) is maximized. Thus, the limit stress for buckling becomes, for all bars that are not defined:

$$\sigma_{max}^{Local.buckl}(A_k) = \frac{4\pi^2 E_{max}}{12(1-{\nu_{max}}^2)} \left(\frac{x_k^{(1)}(A_k)}{x_k^{(2)}(A_k)}\right)^2 \tag{13}$$

$$\sigma_{max}^{Euler.buckl}(A_k) = \frac{\pi^2 E_{max} I_{max}(\boldsymbol{x_k}(A_k))}{A_k\, L^2} \tag{14}$$

with $I_{max}(\boldsymbol{x_k}(A_k))$ the function that returns the highest quadratic moment values, for a given shape (the highest function plotted in Figure 4 corresponds to a 'I' shape).
The expression of $\boldsymbol{RF_k}$ constraints are then given by:

$$\boldsymbol{RF}_k\big(IL(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\big) = \begin{bmatrix} \left(\sigma_{max}^{allow.compr} - \sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \Big/ \sigma_{max}^{allow.compr} \leq 0 \\ \left(\sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A}) - \sigma_{max}^{allow.tension}\right) \Big/ \sigma_{max}^{allow.tension} \leq 0 \\ \left(\sigma_{max}^{Euler.buckl}(A_k) - \sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \Big/ \sigma_{max}^{Euler.buckl}(A_k) \leq 0 \\ \left(\sigma_{max}^{Local.buckl}(A_k) - \sigma_k(\boldsymbol{E}_{\bar{\bar{\boldsymbol{c}}}}, \boldsymbol{E_c}, \boldsymbol{A})\right) \Big/ \sigma_{max}^{Local.buckl}(A_k) \leq 0 \end{bmatrix} \tag{15}$$

The overall formulation of the optimization problem (11) remains unchanged, but there is a better approximation of the constraints because buckling is now taken into account (15).

7.3. Numerical examples and discussion

As explained previously, the combinatorial complexity is directly linked to the number of available catalogs. This is why, to start evaluating our methodology, first runs have been launched with only 2 catalogs based on two different materials and an identical cross-section shape *I*. The first material named 'Material 0' has higher density and allowable values than the second one 'Material 1', which is stiffer. A load of 10^5 N has been applied downward on the 4th node of the truss. In order to highlight the changes in optimal structural concepts, several bounds have been imposed on vertical displacement of the 4th node. The sub-problems $\widehat{P_i}$ are solved using the Method of Moving Asymptotes (MMA) [24] algorithm available in NLOPT library.

The implemented node selection methodology (or queue data structure) during branching step consists in a mix of two strategies: depth-first search and best-first search. Indeed, the branching is performed on the node having the lowest lower bound, with a higher priority assigned to the nodes of the current generation (i.e.; current level in the tree). The advantage of this method is that the process returns a choice for the whole truss as quickly as possible. It provides a solution that is possibly better than the best current upper bound. In this case, the upper bound is updated with this solution, and all the active nodes that have a higher lower bound than this new upper bound can

be deleted from the queue. In this way, a number of nodes are not explored further, leading to an improvement of the whole process convergence.
Table 2 shows the results obtained, with the following convention: a '0' in position *k* of categorical variable vector ***c*** corresponds to a choice of 'Material 0' for the bar *k*, a '1' corresponds to a choice of 'Material 1'.

Table 2: optimal weight and categorical values w.r.t. bounds on displacements

cases	**Bounds on displacements** (mm)	**Optimal Weight** W*(kg)	**Optimal Categorical variable *c****	**Lower bound #1 Combinatory explored** (%)	**Lower bound #2 Combinatory explored** (%)
1	displ < -17.	14.592	0011100111	153%	2.9%
2	displ < -18.	14.053	1011100111	197%	11.3%
3	displ < -19.0	13.722	0001100111	200%	16.2%
4	displ < -19.96	13.633	0001000101	200%	11.1%
5	displ < -25.	13.633	0001000101	200%	11.3%

A comparison of the results with those obtained with a factorial design of experiments shows that for these cases, the branch and bound approach returns the global solutions w.r.t. displacements bound changes. In addition, it can be seen that the optimal weight increases with the displacement bound: the more the structure is constrained, the higher is the weight. We can note that if the whole tree was to be explored, it would require $\sum_{k=1}^{10} 2^k$ computations. This is twice more than the 2^{10} optimizations required to solve the problem using a full combinatorial approach.

The results in Table 2 shows that the first approach is not efficient: all the tree volume (twice the combinatorial search cost) is explored in cases 3, 4 and 5. This means that the branch and bound needs more optimizations compared to those required by a full combinatorial approach. This is because our minoring problems $\widehat{P}_l$ in Eq.(11) are not close enough to the original problem in Eq.(2). However, the results obtained with the second approach shows that the number of optimizations performed during the branch and bound run is equal to 16.2% of the optimizations required by a combinatorial approach. This means that despite the fact that the whole design space has been considered (thanks to branch and bound theory) during the optimization process, a lot of nodes have been pruned and thus a lot of optimizations were not required to find the optimal solution.

The computation history of case 3 is depicted on Figure *5* and Figure *6*. On Figure *5*, the lower bound estimation of the current node is presented in blue, the best upper bound in red. An iteration is defined by a branching and bounding step. We can note that the upper bound is not updated until the end of the computation: the branching route obtained by our strategy leads to the first upper bound after 10 optimizations, that appears to be the optimal solution. In Figure 6 we can follow for example the evolution of the number of pruned nodes (line in yellow) during the optimization. We can see for example a gap between the iterations 50 and 60: a node placed at the top of the tree has been deleted because the lower bound was higher than the upper bound. Thus, the remaining nodes (line in blue) decreases significantly: all the nodes behind the deleted one do not need to be explored.

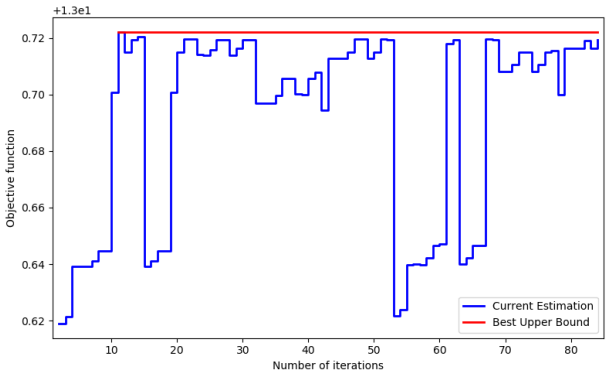


Figure 5: Lower bound w.r.t. iterations in case 3.

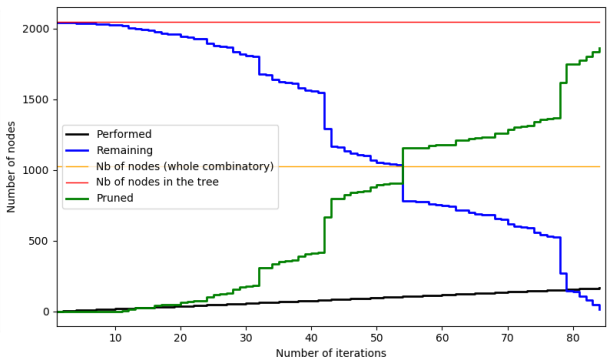


Figure 6: Number of nodes computed, remaining, and pruned w.r.t iterations in case 3.

Figure 7 shows the truss with stress values, optimal areas, and reserve factors at the optimum for the case 3. This illustrates the trades that are made by the optimizer between the materials and the areas in this case. First, it can be noted that the solution is non-uniform in terms of categorical variables: each of the two available materials have been chosen at least once to build the optimal truss. Compared to case 4, the optimal material of bars 5 and 9 changed to the first material ('Material 0'), with higher density and allowable values.

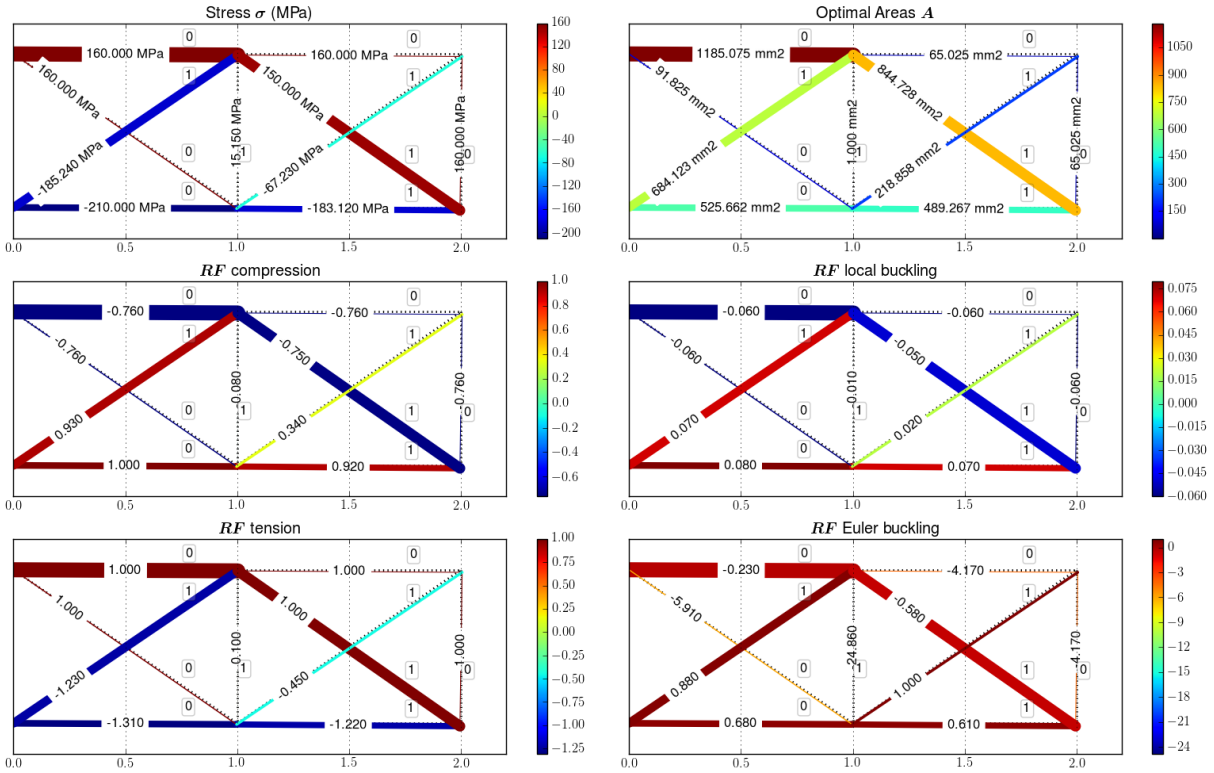


Figure 7: results of case 3, with **c***= '0001100111'

The computation case 3 has been launched again, at the exception that two stiffening principles have been added to the two material choices: one U shape and one I shape. Since a categorical value corresponds to a choice of material and shape, the combination raises the number of catalogs from 2 to 4: $m = 4$. This means that 4^{10} optimizations could solve the problem, and the total volume of the tree counts $\sum_{k=1}^{10} 4^k$ nodes. As it has proven its efficiency, the second methodology to compute the lower bound. The optimal solution is $\boldsymbol{c}^*$= *'2220023000'*, the Figure 8 shows the convergence history. 7040 continuous optimizations have been performed to solve this problem, corresponding to less than 0.7% of the optimizations required by a combinatorial approach. In Figure 8, the bar diagram shows that a few nodes at level 3, 4 and 5 of the tree have been removed during the computation. This is very interesting in terms of algorithm efficiency, because the design space associated to these node has been eliminated, including all depending nodes. It can be seen that most of the deleted nodes are located between the 6th and 10th level of the tree.

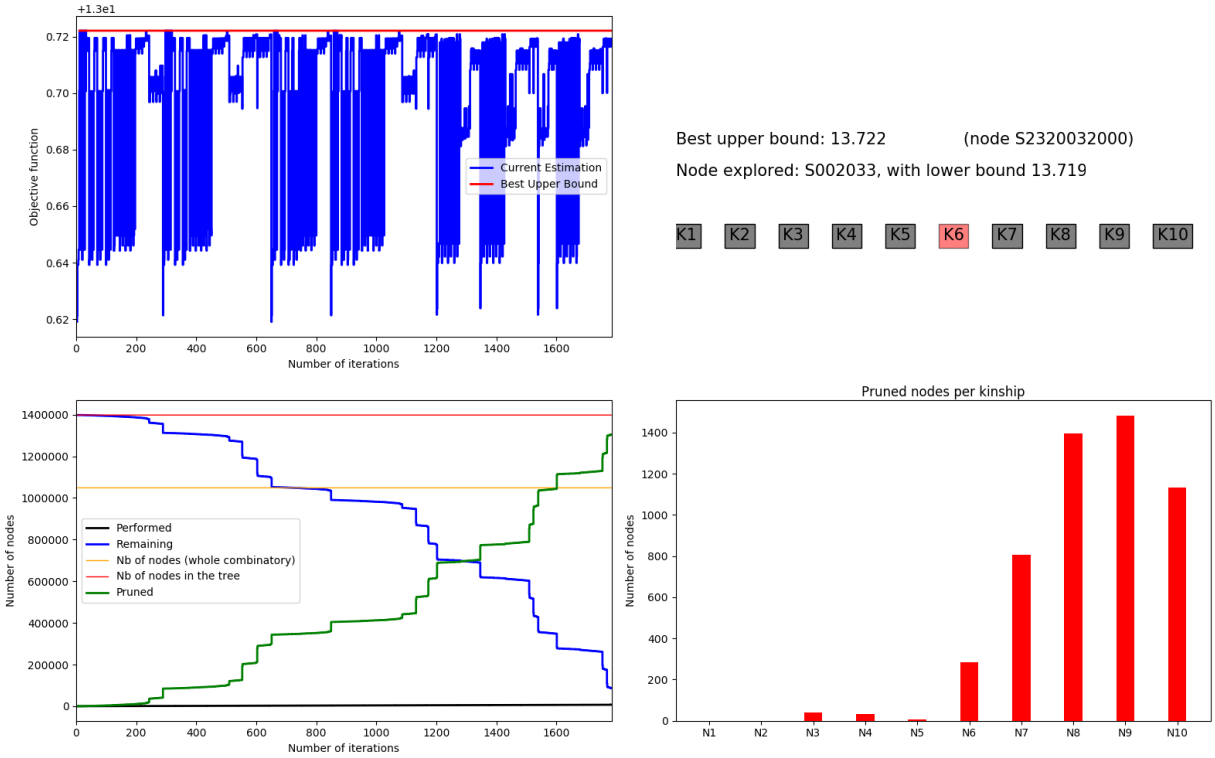


Figure 8: Computation history of case 3 with 4 catalogs

The number of explored nodes, exposed in Table 2, demonstrates that the number of optimizations during the branch and bound process using the first approach exceeds the number of optimizations of a full design of experiment on categorical variables (because the whole tree counts twice more optimizations than the 2^{10} optimizations of the whole combinatorial problem). Indeed, as explained before, the branch and bound algorithm efficiency relies on the pruning rate of the tree. Nodes are discarded because either the solution is infeasible, or the lower bound exceeds the best upper bound. In our cases, independently of the quality of categorical choices, the optimizer can balance with the areas values. Thus, considering the available material quality and the bounds of the continuous optimization subspaces, there is no unfeasible solution. The pruning relies on lower bounds and upper bound comparison too. In this sense, the second approach offers a better approximation of the constraints, and thus a better lower bound. This allows to significantly decrease the number of optimizations: less than 16.2% of the number of optimizations required by a combinatorial approach are now performed to find a global solution of the mixed problem. In the case of 4 catalogs, only 0.7% of the optimizations are required, but it still needs 7040 optimizations. Knowing that an industrial finite element of a pylon counts around 100 elements, with 10 up to 100 choices available per element, this methodology based on the branch and bound approach remains costly.

The efficiency could be improved for example by choosing the branching order of the categorical variables, or adapting the numerical parameters (initial point, tolerances, …) of the continuous optimization during the branch and bound process. One promising approach consists in replacing categorical variables by continuous ones belonging to [0,1], using the work done for multi-material and topology optimization [25]. This would keep the consistency between all the optimal physical features, and thus increase the lower bound accuracy.

8. Conclusion

Relying on Airbus experience and previous work on this topic, a formulation of a mixed optimization problem that tackles both continuous and non-ordered categorical variables has been proposed in this article. Specific attention has been paid to keep this formulation as generic as possible, without exhibiting any physical feature directly. A toy case involving a simplified physical model, that illustrates the industrial optimization problem complexity, has been presented. A hybrid approach based on branch and bound generic formulation, and taking advantage of the generalized Benders decomposition, has been implemented. This method allows a trade-off between categorical and continuous variables that leads to the global optimum. However, numerical examples have highlighted the limitations of this approach. The performance in terms of computation time prevents from scaling up the toy problem for industrial purpose. Several improvements are under investigation, mainly based on a better handling of categorical variables transformation into continuous ones.

9. Acknowledgements

This work is part of the MDA-MDO project of the French Institute of Technology IRT Saint Exupery. We wish to acknowledge the PIA framework (CGI, ANR) and the project industrial members for their support, financial backing and/or own knowledge: Airbus, Airbus Group Innovations, SOGETI High Tech, Altran Technologies, CERFACS. We would like to thank Charlie Vanaret for all his advices, specifically for the branch and bound formulation.

10. References

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-We. 1989.

[2] R. Storn and K. Price, Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces, *Journal of Global Optimization*, 11(4), 341–359, 1997.

[3] J. Kennedy, R. C. Eberhart, and Y. Shi, Swarm Intelligence, *Scholarpedia*, 2(9), 1462, 2001.

[4] V. Torczon, On the convergence of pattern search algorithms, *SIAM Journal on Optimization*, 7, 1–25, 1997.

[5] J. A. Lampinen and I. Zelinka, Mixed Integer-Discrete-Continuous Optimization, by Differential Evolution, Part 1: the optimization method, *Proceedings of MENDEL'99, 5th International Mendel Conference on Soft Computing*, 71–76, 1999.

[6] C.-X. Guo, J.-S. Hu, B. Ye, and Y.-J. Cao, Swarm intelligence for mixed-variable design optimization., *Journal of Zhejiang University. Science*, 5(7), 851–60, 2004.

[7] G. G. Dimopoulos, Mixed-variable engineering optimization based on evolutionary and social metaphors, *Computer Methods in Applied Mechanics and Engineering*, 196(4–6), 803–817, 2007.

[8] K. Deb and M. Goyal, A Flexible Optimization Procedure for Mechanical Component Design Based on Genetic Adaptive Search, *Journal of Mechanical Design*, 120(2), 162, 1998.

[9] C. Audet and J. E. Dennis, Pattern Search Algorithms for Mixed Variable Programming, *SIAM Journal on Optimization*, 11(3), 573–594, 2001.

[10] S. Rakshit and G. K. Ananthasuresh, Simultaneous material selection and geometry design of statically determinate trusses using continuous optimization, *Structural and Multidisciplinary Optimization*, 35(1), 55–68, 2008.

[11] M. Herrera, A. Guglielmetti, M. Xiao, and R. Filomeno Coelho, Metamodel-assisted optimization based on multiple kernel regression for mixed variables, *Structural and Multidisciplinary Optimization*, 49(6), 979–991, 2014.

[12] S. Grihon, Structure Sizing Optimization Capabilities at AIRBUS, *WCSMO12*, submitted, 2017.

[13] S. Grihon, PRESTO: A rapid sizing tool for airframe conceptual design studies, *LMS European Aeronautical Conference*, 2012.

[14] D. B. Adams, L. T. Watson, Z. Gürdal, and C. M. Anderson-Cook, Genetic algorithm optimization and blending of composite laminates by locally reducing laminate thickness, *Advances in Engineering Software*, 35(1), 35–43, 2004.

[15] A. Carpentier, L. Michel, S. Grihon, and J.-J. Barrau, Optimization methodology of composite panels, *12th European Conference on Composite Materials (ECCM 12)*, 1–8, 2006.

[16] A. Carpentier, J.-J. Barrau, L. Michel, and S. Grihon, Buckling optimisation of composite panels via lay-up tables, *III European Conference on Computational Mechanics*, Springer Netherlands, Dordrecht, 226–226, 2006.

[17] A. Merval, L'Optimisation Multiniveaux D'une Structure, *PhD Thesis*, 2008.

[18] A. Merval, M. Samuelides, and S. Grihon, Multilevel optimization with local mass minimization, *2nd European conference for aerospace sciences - EUCASS 2007*, 2007.

[19] M. Samuelides, D. Bettebghor, S. Grihon, A. Merval, and J. Morlier, Modèles réduits en optimisation multiniveau de structures aéronautiques, *9e Colloque national en calcul des structures*, 2009.

[20] A. H. Land and A. G. Doig, An Automatic Method of Solving Discrete Programming Problems, *Econometrica*, 28(3), 497, 1960.

[21] R. J. Dakin, A tree-search algorithm for mixed integer programming problems, *The Computer Journal*, 8(3), 250–255, 1965.

[22] C. H. Tseng, L. W. Wang, and S. F. Ling, Enhancing Branch-and-Bound Method for Structural Optimization, *Journal of Structural Engineering*, 121(5), 831–837, 1995.

[23] A. M. Geoffrion, Generalized Benders decomposition, *Journal of Optimization Theory and Applications*, 10(4), 237–260, 1972.

[24] K. Svanberg, A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations, *SIAM Journal on Optimization*, 12(2), 555–573, 2002.

[25] J. Stegmann and E. Lund, Discrete material optimization of general composite shell structures, *International Journal for Numerical Methods in Engineering*, 62(14), 2009–2027, 2005.