Transdisciplinary Engineering Methods for Social Innovation of Industry 4.0
M. Peruzzini et al. (Eds.)

doi:10.3233/978-1-61499-898-3-837

A Design Methodology for a CSP-Based Optimization Approach

Paolo CICCONI[a,1], Miriam NARDELLI[b], Roberto RAFFAELI[a] and Michele GERMANI[b]
[a] *Università degli Studi eCampus*
[b] *Università Politecnica delle Marche*

Abstract. Design optimization is a common practice in industry. Different mathematical algorithms have been developing to support the optimization in engineering design. The integration between optimization methods and simulations is an interesting issue in engineering design. A typical optimization workflow can include simulation steps; however, the Virtual Prototyping analysis is more time-consuming than analytical calculations. The study of Constraints Satisfaction Problems is a mathematical topic which can be applied for solving engineering issues in design. The strength of this approach is the velocity on searching the satisfied solutions. This paper proposes a design methodology which considers the use of CSP models and calculation tools to optimize the sizing of columns and beams in the design of a steel structure. The calculation tools regard analytical models and numerical analysis. While the analytical approach regards the computing of cost and weight, the numerical analysis is used to verify and check the engineering performance in terms of deformation and stress state. A customized application, based on MiniZinc platform, has been developed and proposed to solve the CSP model for a test case steel structure. The CSP problem has been limited to the calculation of analytical constraints such as cost and weight. Finally, the resultant set of solutions has been evaluated using numerical solution to complete the optimization analysis.

Keywords. Optimization, Constraints Satisfaction Problems (CSP), MiniZinc, Virtual Prototyping, Steel Structures

Introduction

Nowadays, different commercial tools and mathematical algorithms are available to support the engineering design in fields such as mechanics, electronics, and civil. However, a lack still exists in the development of a flexible and agile design methods to support the optimization workflow [1]. This paper proposes a methodological approach to prove the use of a CSP-based method in engineering optimization [2]. Constraints Satisfaction Problems (CSP) are mathematical problems where each solution must satisfy constraints and limits. In a CSP model, the definition of constraints can be seen as a formalization of knowledge using mathematical equations.

The CSP approach requires the definition of a mathematical model which reproduces the behavior of the physical system [2]. A software tool is necessary to apply this approach in mechanical design. The model to be applied is parametric;

[1] Corresponding Author, Mail: paolo.cicconi@uniecampus.it

therefore, each variable can change is value in function of a defined variation range. Traditional CSP solvers iterate each configuration of parameters until the defined mathematical constraints are satisfied. However, in mechanical industry, many design problems allow numerous solutions to be applied. For this reason, a lot of research is focused on techniques and methods for design optimization. As described by [3], the multi-objective optimization is an approach for satisfying conflicting criteria. In the context of oil & gas applications, [4] describes an optimization workflow to reduce the weight of big steel structures. This approach considers the integration of FEM simulations into the optimization analysis. Constraints were defined as normative checks to be verified after each structural calculation.
Trabelsi [5] applied a CSP method to support the preliminary design of a linear vehicle suspension system. This research, which was focused on the sizing of the system, proposes a comparison between conventional design methods and shows how the computation time related to CSP models is satisfactory.

Yvars [3] introduced a deterministic CSP approach to solve a pareto bi-criterion optimization. He proposed a research example focused on a bolt coupling [3]. His research used CSP methods to reduce the size and the complexity of the design phase. However, this approach does not consider the integration of the design workflow with other tools such as external configuration software or numerical FEM solvers. The use of FEM solvers increases the calculation time and requires the use of other kinds of optimization algorithms. Yvars shows how designer can test several ways by adding or deleting constraints in a CSP model. Raffaeli [6] also studied CSP problems. His focus was on mechanical systems such as Engineer-To-Order (ETO) products. These complex products require a more complex formalization of knowledge. He studied a CSP approach for a further implementation into a generic configuration system. Generally, design parameters in CSP problems must satisfy a collection of constraints related to standard, customer requirements, marketing requirements, etc.

Lin [7] proposed a TRIZ-based approach to support design when the solutions of optimization methods do not meet the objectives of problems to solve. His approach is focused on the analysis of system contradictions which extracts data from the simulations generated from the optimization loop.

The scope of the proposed research is the implementation of a CSP models and solution-searching algorithms to support optimization in engineering design. The aim of the paper is to demonstrate the use of constraints satisfaction problems in a design workflow which also considers the use of numerical simulations. While a CSP model can be solved using a combination of heuristics and combinatorial search methods to be solved in a reasonable time, numerical simulations can take from few seconds to some hours for computing. This paper overcomes this limit introducing a constraints-based approach to optimize cost and weight of a steel structure using analytical calculation tools. A customized application, based on MiniZinc platform, have been developed and proposed to solve the CSP model for a test case steel structure. The test case shows a CSP study applied to steel structures used in oil & gas applications. The developed tool includes the mathematical formulation of design constraints used for the design of steel structures.

The CSP problem has been limited to the calculation of analytical constraints such as cost and weight. Numerical simulations are introduced in a second step. In the proposed test case, a set of satisfied solutions, from the CSP analysis, has been evaluated using simulation tools for structural analysis.

1. State of the Art

Generally, designers use and manage design rules, tables, formulas, and relations during the mechanical design process. A CSP model implements all these knowledge representations as mathematical constraints. Many design problems are analyzed in literature using a CSP approach with design synthesis. A multidisciplinary design optimization advisor system [8] and the implementation of an ICAD generative model [9] was proposed in the design of aircraft applications.

There are numerous methods to solve this kind of problems and many other methods are also studied to improve calculation time and precision in CSP problems. The more common algorithms are: backtracking, branch and bound, and depth first search. They represent three categories of methods to solve CSP problems. Many variations and improving of them have been studied and implemented in literature. Backtracking is a technique that consists to enumerate all possible solutions and prunes all the ones that do not satisfy some constraint [10]. It explores a graph tree remembering all the nodes already analyzed, in that way, if a path must be pruned, it can come back to the node that is not visited jet without the risk to move in a path already explored. The backtracking algorithm has an exponential complexity, so it is not so efficient for not NP-complete problems. Nevertheless, the algorithm integrates some heuristic techniques that allow to decrease complexity.

Branch and bound is a general technique to solve finite optimization problems [11]. This approach starts subdividing the original problem in subproblems, which are easier to solve. Branch and bound algorithms are called as implicit enumeration because they enumerate all the possible solutions and tries all of them, but some will be deleted proving their non-optimality. Depth first search is another technique of tree search, its particularity is that the algorithm can explore nodes far from the root without having visited the nodes of first generation [12]. The search strategy explores the graph reaching the deepest possible node in it. Once all the deepest nodes are analyzed, the algorithm comes back to explore all the previous ones.

A development platform that implements all these algorithms and many others is Gecode, which is an open, free, and fast toolkit used to solve CSP problems. Gecode already contains many features but its strength is in the fact that it is programmable: it supports the implementation of new constraints, strategies, and search methods [13]. The programming language for extensions is C++ based. Moreover, Gecode is efficient in time and memory consuming. It has been used to solve over 50000 test cases with good results. However, Gecode has some limitations: it does not support geometrical constraints, it does not support connections with database, it has also limitations using constraints that involves strings, finally the connection with other software must be implemented with custom libraries.

Münzer [14] used Gecode to implement part of a model to search design solutions derived from a set of specified techniques. His method consists in: define a metamodel of the problem and its constraints, find the interconnections, assign variables using CSP problems (Gecode) and optimize an objective function using simulating annealing. Other tools have been analyzed to solve CSP problems such as MiniZinc [15], FlatZinc, and Choco. MiniZinc is a medium-level constraint modelling platform based on Zinc language [2]. It can be mapped onto existing solvers. Algorithms developed with MiniZinc are compiled in FlatZinc language. FlatZinc is a low-level programming language which can directly interact with solver such as Gecode and others [16] to translate the CSP model into the language required by the defined solver.

2. Method

Figure 1 shows the proposed methodological approach. The CSP model includes a set of variables, values, and constraints. Each variable can take a range of values. Constraints are mathematical functions and limits. They include relationships, variables, and values. Even if the approach is generic, this paper is focused on the design optimization of steel structures.

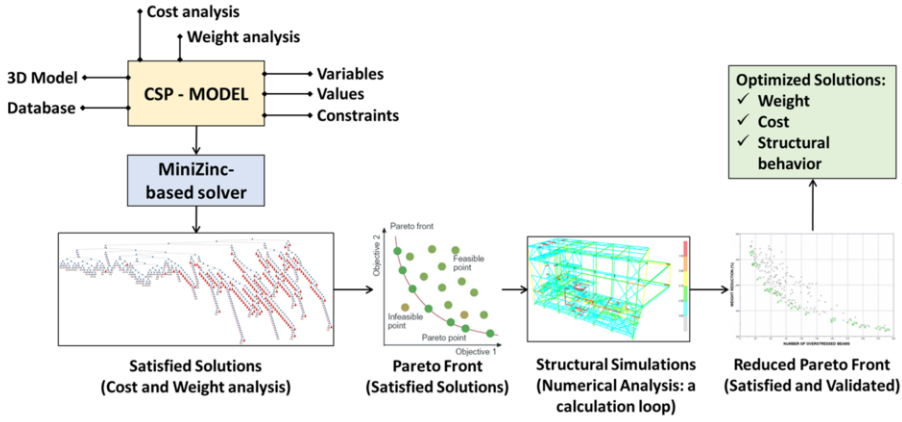

Figure 1. The methodological approach.

The proposed test case considers an analytical calculation to estimate product cost and weight. Therefore, the cost and weight models are included into the CSP model. The CSP model has been developed in MiniZinc, which is Gecode-based programming tool. Additional features, such as cost and weight functions, have been implemented in VB.NET. In particular, a 3D model interaction has been performed with VB.NET and the steel structure's geometry. The geometry has been simplified through a trusses model and represented using the CAD module of SAP2000, which is a FEM software used for the structural analysis of buildings and steel trusses in general. Using the API tools provided by SAP2000 has been possible to perform the connection between the CSP model and the interactive geometry. In fact, during the constraints satisfaction solution, the section dimensions of each beam and column were directly changed into the 3D geometrical model, in order to generate the updated trusses model for the analytical and numerical computing.

As cited before, a prototypical software has been developed to support the CSP solving. An user-interface has been also defined for the management of the project data and design objectives. The applciation core is a MiniZinc.-based solver which implements variables, values, and constraints related to the steel constructions analyzed in this paper. This tool implements a knowledge-base which is formalized in rules, formulas, and constraints realted to the engineering design.

The constraints related to the engineering design of a steel structure are mainly focused on normative checks and conditions such as the structural limits to be applied. Other constraints are related to validation rules based on the technical know-how. These constraints can refer to the formalization of explicit and implicit knowledge. Examples can be validation rules to check the weight balance of assemblies into a steel structure. Another example can be rules to evaluate and limit the number of different types of components such as beams into an assembly.

3. Steel structures

The proposed test case shows the CSP modelling of a big steel structure, called module, which is a 500-ton steel construction used for power applications in oil & gas. An oil & gas plant is often a collection of pre-fabricated modules. Modularization is a common design strategy in several engineering fields [17] such as oil & gas [4] where the products are pre-assembled and shipped. The modularity approach for oil & gas plants reduces the overall cost and the delivery time. A module is the smallest functional unit with its equipment, machines, and steel structure. Therefore, a single module (Figure 2) can contain power generation units, gas compression units, and process equipment for oil & gas applications. The structure of a classic module consists of steel beams used for internal support of machines and equipment.

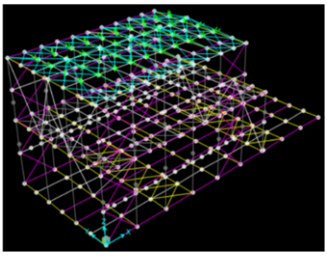

Figure 2. The simplified structural modelling of a steel structure (module) used in oil & gas.

3.1. Analytical constraints

Regarding the design of steel structures with a CSP approach, the collection of the proposed constraints has been analyzed and discussed with an expert team of designers, in collaboration with an Italian enterprise which produces oil & gas solutions. Different analytical constraints are related to the structure's sizing. An important constraint is the ratio between the section of each column and its height, in order to limit the buckling problem. Another constraint is related to the sizing of pipe beams. The ratio diameter per width is a constraint which can be described using a mathematical formulation; the limit value depends on the diameter range. Constraints can be also applied to regulate the weight balance of each group of beams. The steel structure's weight can be also considered as a constraint for the CSP analysis. This paper also introduces a cost constraint into the CSP model, to limit the expensive cost of big steel structure from the early design phase.

3.2. Structural Analysis

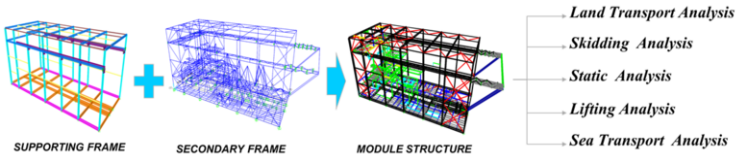

Figure 3. The module's structure and main simulation analyses.

SAP2000® by CSI America is the FEM solver employed in this research to simulate the mechanical behavior of a simplified steel structure. Using the API tools provided

by SAP2000®, an algorithm has been implemented with VB script to update each geometry realted to the CSP model during the otpimization analysis.

The structure of a classic module consists of a supporting frame (or main frame) and a secondary one (Figure 3). The total average self-weight may raise over 500 tons per module. Due to their typical functionality, the modules are subjected to static loads, dynamic loads, land transport loads, sea transport and fatigue loads. In fact, they are prefabricated in a construction site and then shipped by sea to destination for the installation. Therefore, the weight optimization of such steel constructions is a very important issue in the field of the oil & gas industry. In fact, the cost of a steel module is related to the structure's self-weight, the number of beams, the types of beam sections, the number of overstressed beams and the type of construction. Figure 3 describes every design level related to each structural analsyis which the designer has to perform. Simulation tools based on FEM analysis are used in the design of steel structures due to the possibility to investigate the behavior of mechanical structures with virtual prototypes. In general, this research aims to reduce time and cost related to the early design phase of big steel structures.

3.3. Cost modeling

The approach used for the costing of the proposed steel structures is an analytical approach. The analyzed structure is composed by two principle types of beams: strandard beams and pipe beams. The cost of standard beams is parametric, this can be derived from studies on tables of catalogues: more or less 0.75 EUR/kg (Italian market). For pipe beams, the calculation of the cost is a function which depends on the external diameter. This is a non-linear function in relation with the weight of the related beam. This is for the fact that the fabrication of pipes with smaller diameters is more expensive than processing pipes with huger dimeters. This is valid for a diamater which varies from 200 mm to 1000 mm. After this range, the cost function has an increasing trend.

4. Tool development

A prototypical tool has been developed using VB.NET and implementing the API (Application Programming Interface) tool provided by SAP2000. This tool has an inteface which interacts with MiniZinc to run Gecode, which implements the CSP solver. Figure 4 shows the tool developing phases. As cited before, the test case proposes the CSP modelling of a steel structure. The principal task is to find the minimum weight and cost configuration varying the size of every beam. In particular, these variations concern: external diameters and thickness of pipes and the dimensions of the standard beams. The constraints regard the dimensional limits for each diameter and thickness, and the ratio between diameter and thickness. The last but not the least constraint is that the weight must be between 400 and 600 ton.

Firstly, the authors have implemented all the classes necessary to manage the product structure of the model inside the proposed software tool. A class is a structure of data and algorithms typical of the Object-Oriented programming. This software, which implements the highlighted classes, includes a graphical user interface (GUI) and consists of three modules called: *CspModel*, *BeamModule* and *MinizincTest*. The aim of this research is to support the user in the definition of the CSP model, without

programming functions and using the Gecode language. The principal project is the generalization of the model (*CspModel*), which consists in sub-dividing the problem in variables, constants, and constraints. The last project is the customization of the model for this case study (*BeamModule*). It represents a module, which is a group of beams. For this model have been implemented several classes: one to manage them (a class-list) and the others to represent the structure of each beam. For the classes that specialize the beams (pipe, standard and built-up) different geometrical variables have been settled. In the manager class have been implemented all that generic functions like calculate the weight of a group of beams and calculate the cost of a group of beams. Once the problem is formalized, it is send to MiniZinc to be analyzed and solved. The results are gathered in the interface and shown to the user.

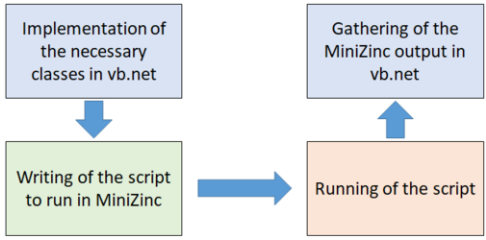

Figure 4. Developing phases from VB.NET to MiniZinc.

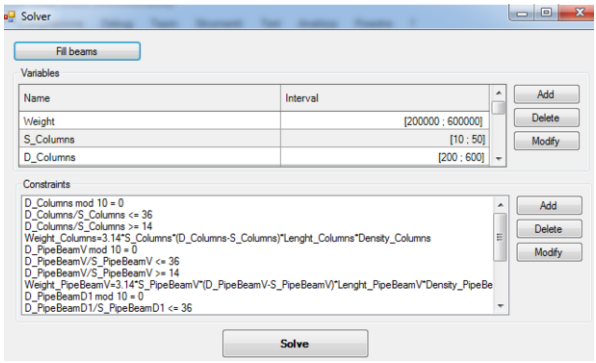

Figure 5. The settings interface and first optimization results.

The computational model of a CSP problem is divided in variables, constants, and constraints. Figure 5 shows constraints definitions and ranges of variables. The Solve command runs a simplified optimization using MiniZinc. The method proposes a software application which is based on Gecode toolkit. In this approach, while Gecode runs in the background, an interface shows and manages the CSP problem. The most difficult phase is the definition of the CSP problem because design routines are often based on experience. In fact, the formalization of the technical knowledge means to translate something that often is only in the mind of the designer in a standard understandable problem, made of variables and constraints. While explicit knowledge is easy to translate in a mathematical constraint, implicit and tacit knowledge are very difficult to be elicited and formalized.

The information sent to Minizinc runs in the background. The process elaborets the following data. For each group three constants are defined, as shown in Figure 6: the total length of the group, the count of the group and the density of the group material.

```
float: Lenght_Columns= 306000;
float: Count_Columns= 18;
float: Density_Columns= 7.73E-06;
```

Figure 6. Constants of the MiniZinc problem.

The variables are the external diameter and the thickness for groups which contain pipe beams (Figure 7) and the area related to the designation for groups which contains standard beams (Figure 8). Figure 9 shows the domains of the variables where MiniZinc will search the best solution. Finally, Figure 10 shows the declaration of constraints in MiniZinc language.

```
var S_domain: S_Columns;
var D_domain: D_Columns;
```

Figure 7. Variables for pipe beams.

```
var hA_domain: Area_BeamD0MBT500;
```

Figure 8. Variables for standard beams.

```
set of int: S_domain= 10..50;
set of int: D_domain= {200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430,
440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600};
set of int: hA_domain= {2124, 2534, 3142, 3877, 4525, 5383, 6434, 7684, 8682, 9726, 11250, 12440, 13350, 14280, 15900, 17800, 19750, 21180,
```

Figure 9. Domains of variables.

```
constraint D_PipeBeamD2/S_PipeBeamD2 <= 36;
constraint D_PipeBeamD2/S_PipeBeamD2 >= 14;
constraint 3.14*S_Columns*(D_Columns-S_Columns)*Lenght_Columns*Density_Columns+3.14*S_PipeBeamV*(D_PipeBeamV-
S_PipeBeamV)*Lenght_PipeBeamV*Density_PipeBeamV
+Area_BeamD0MBT500*Lenght_BeamD0MBT500*Density_BeamD0MBT500+Area_BeamD0MBL400*Lenght_BeamD0MBL400*Density_BeamD0MBL400+Area_BeamD1MBT400*Lenght_B
eamD1MBT400*Density_BeamD1MBT400+3.14*S_PipeBeamD1*(D_PipeBeamD1-
S_PipeBeamD1)*Lenght_PipeBeamD1*Density_PipeBeamD1+Area_BeamD0SB300*Lenght_BeamD0SB300*Density_BeamD0SB300+Area_CrossBeam*Lenght_CrossBeam*Densit
y_CrossBeam+Area_CrossBeamDeck*Lenght_CrossBeamDeck*Density_CrossBeamDeck
+Area_BeamD2MBL500*Lenght_BeamD2MBL500*Density_BeamD2MBL500+Area_BeamD2MBT500*Lenght_BeamD2MBT500*Density_BeamD2MBT500+Area_BeamI*Lenght_BeamI*De
nsity_BeamI+Area_Supports*Lenght_Supports*Density_Supports+3.14*S_PipeBeamD2*(D_PipeBeamD2-S_PipeBeamD2)*Lenght_PipeBeamD2*Density_PipeBeamD2<=
600000;
```

Figure 10. Constraints in MiniZinc language.

5. Results and discussions

As cited before, the proposed test case regards the CSP optimization of a steel structure used for oil & gas applications (Figure 2). The optimization objectives concern weight and cost minimization. Figure 11 shows the resulting solutions optimized using the proposed CSP tool. In particular, Figure 11 reports the first 30 solutions ordered by an objective function. This function weights each contribute related to the two analyzed objectives: cost and weight. The resulting list has been evaluated using a FEM analysis, in order to evaluate the structural behavior of each configuration. SAP2000 has been used as FEM tool to simualte the structural behavior for each solution analyzed in the table highlighted in Figure 11. The results of the structural analysis are highlighted in Figure 12, which shows each solution comparated by the number of overstressed beams and weight. Figure 12 also shows how the weight reduction can increase the number of overstressed beams in the structural analysis. The loading conditions of the structural analysis are provided by standardized and international normatives. The proposed steel structure is focused on the North America area; therefore, loading conditions are related to the ASCE (American Society of Civil Engineers) standards. Generally, design loads considered for steel structures are a combination of dead loads (the total weight of the structure), live loads (variable loads related to the use of the structure and their accessories), wind load (the action of the wind in each direction), and seismic load (the simulation of the dynamic load related to a possible earthquake).

S;D (Input)				Area (Output)										Output	
COL	PIPEV	PIPED1	PIPED2	D0MBT500	D0MBL400	D1MBT400	D0SB300	CROSSBEAM	BEAMDECK	D2MBL500	D2MBT500	BEAMI	SUPPORTS	Weight	Cost
10;200	10;200	10;200	10;200	HE700A	HE280A	HE700A	HE600A	HE500A	HE450A	HE100A	HE100A	HE100A	HE100A	398004	278309
10;200	10;200	10;200	10;200	HE600A	HE550A	HE550A	HE650A	HE550A	HE450A	HE120A	HE100A	HE100A	HE100A	398502	278309
10;200	10;200	10;200	10;200	HE600A	HE450A	HE500A	HE450A	HE600A	HE700A	HE120A	HE100A	HE100A	HE100A	398801	278309
10;200	10;200	10;200	10;200	HE550A	HE600A	HE500A	HE700A	HE500A	HE700A	HE100A	HE100A	HE100A	HE100A	399504	278309
10;200	10;200	10;200	10;200	HE700A	HE450A	HE550A	HE550A	HE500A	HE650A	HE100A	HE100A	HE100A	HE100A	400001	278309
10;200	10;200	10;200	10;200	HE700A	HE650A	HE700A	HE700A	HE450A	HE260A	HE100A	HE100A	HE100A	HE100A	400016	278319
10;200	10;200	10;200	10;200	HE650A	HE650A	HE600A	HE700A	HE450A	HE500A	HE100A	HE100A	HE100A	HE100A	400017	278320
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE650A	HE360A	HE550A	HE100A	HE100A	HE100A	HE100A	400021	278322
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE600A	HE340A	HE700A	HE100A	HE100A	HE100A	HE100A	400058	278346
10;200	10;200	10;200	10;200	HE700A	HE650A	HE700A	HE700A	HE340A	HE650A	HE100A	HE100A	HE100A	HE100A	400497	278627
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE700A	HE340A	HE700A	HE100A	HE100A	HE100A	HE100A	400571	278674
10;200	10;200	10;200	10;200	HE650A	HE700A	HE700A	HE700A	HE340A	HE700A	HE100A	HE100A	HE100A	HE100A	400965	278926
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE700A	HE320A	HE700A	HE100A	HE100A	HE100A	HE100A	400977	278934
10;200	10;200	10;200	10;200	HE700A	HE650A	HE600A	HE500A	HE500A	HE450A	HE100A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE600A	HE400A	HE600A	HE700A	HE600A	HE340A	HE100A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE650A	HE600A	HE600A	HE400A	HE700A	HE700A	HE100A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE650A	HE140A	HE500A	HE550A	HE700A	HE500A	HE140A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE700A	HE650A	HE400A	HE650A	HE600A	HE300A	HE200A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE600A	HE280A	HE450A	HE340A	HE600A	HE450A	HE340A	HE100A	HE100A	HE100A	401000	278949
10;200	10;200	10;200	10;200	HE500A	HE340A	HE260A	HE450A	HE650A	HE400A	HE500A	HE100A	HE100A	HE100A	401050	278949
10;200	10;200	10;200	10;200	HE360A	HE260A	HE360A	HE450A	HE500A	HE400A	HE700A	HE100A	HE100A	HE100A	401065	278949
10;200	10;200	10;200	10;200	HE700A	HE600A	HE700A	HE500A	HE400A	HE650A	HE100A	HE100A	HE100A	HE100A	401082	278950
10;200	10;200	10;200	10;200	HE700A	HE260A	HE700A	HE600A	HE450A	HE700A	HE100A	HE100A	HE100A	HE100A	401100	278950
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE550A	HE400A	HE500A	HE100A	HE100A	HE100A	HE100A	401150	278952
10;200	10;200	10;200	10;200	HE650A	HE500A	HE700A	HE700A	HE400A	HE600A	HE100A	HE100A	HE100A	HE100A	401180	278959
10;200	10;200	10;200	10;200	HE700A	HE600A	HE700A	HE700A	HE360A	HE600A	HE100A	HE100A	HE100A	HE100A	401200	278960
10;200	10;200	10;200	10;200	HE700A	HE650A	HE700A	HE700A	HE360A	HE700A	HE100A	HE100A	HE100A	HE100A	401310	279147
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE700A	HE340A	HE700A	HE100A	HE100A	HE100A	HE100A	401335	279163
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE650A	HE340A	HE700A	HE100A	HE100A	HE100A	HE100A	402268	279760
10;200	10;200	10;200	10;200	HE700A	HE700A	HE700A	HE700A	HE340A	HE650A	HE100A	HE100A	HE100A	HE100A	402550	279943

Figure 11. The result of the CSP optimization which minimizes the module's weight and cost for the parameters analyzed.

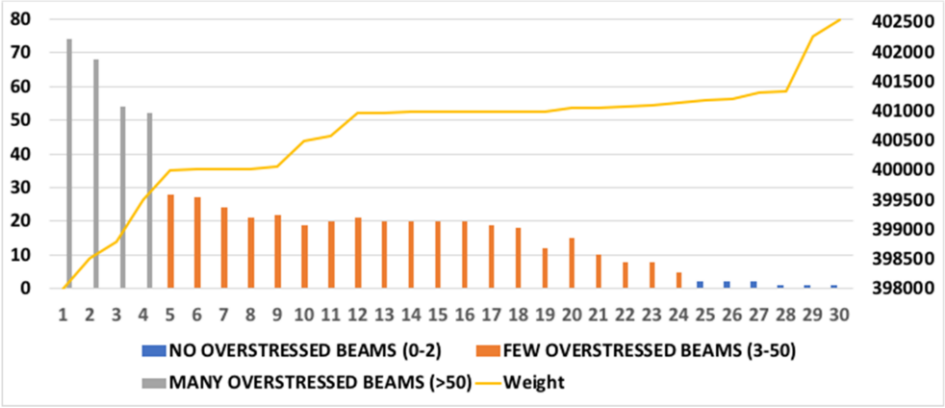

Figure 12. The comparison in terms of weight (Y-axis, on the right) and number of overstressed beams (Y-axis on the left) for each solution analyzed (X-axis) with the previous CSP analysis.

The results show that the optimal solution is a compromise between cost, weight, and structural performance. Each analyzed solution satisfies the normative checks even if there are some overstressed beams. The overstressed beams are items where the stress ratio exceeds the limit in simulations. They are not accepted in the present size. However, every overstressed beam can be replaced with a reinforced profile, in order to avoid structural default in the operation phase. The cost of each beam replacing has not been yet analyzed in this paper. It could be considered as a future work. One of the optimized solution can be considered the 9-index configuration (Figure 11), which provides a weight of 400058 kg with 22 overstressed beams to be substituted.

6. Conclusions

This paper proposes a CSP approach to support the design optimization of steel structure. The design methodology considers the use of CSP models and calculation tools to optimize the sizing of beams in the design of steel structures. A Windows-based application has been developed to implement and solve the relative CSP model. The analyzed design methodology shows how it is possible to integrate analytical and numerical solvers using a CSP-based optimization for mechanical and civil engineering problems. The results show a time reduction in design optimization phases using a CSP solver before the employment of numerical simulations. The introduction of the CSP analysis reduce the simulations of not-valid configurations because it implements a knowledge-based within mathematical constraints.

References

[1] A. Albers, M. Spadinger, M. Serf, S. Reichert, S. Heldmaier, M. Schulz and N. Bursac, Coupling of Computer-Aided Methods: Supporting Product Developer During Embodiment Synthesis, *Advances in Structural and Multidisciplinary Optimization*, 2017, pp. 536–548.

[2] T. Guns, A. Dries, G. Tack, S. Nijssen and L. D. Raedt, The MiningZinc Framework for Constraint-Based Itemset Mining, 2013 IEEE 13th International Conference on Data Mining Workshops, 2013, DOI: 10.1109/ICDMW.2013.38.

[3] P.A. Yvars, Pareto Bi-Criterion Optimization For System Sizing: A Deterministic And Constraint Based Approach, *International Conference On Engineering Design*, ICED11, 2011.

[4] P. Cicconi, M. Germani, S. Bondi, A. Zuliani and E. Cagnacci, A Design Methodology to Support the Optimization of Steel Structures, *Procedia CIRP*, Vol. 50, 2016, pp. 58–64.

[5] H. Trabelsi, P.-A. Yvars, J. Louati and M. Haddar, Interval computation and constraint propagation for the optimal design of a compression spring for a linear vehicle suspension system, *Mechanism and Machine Theory*, Vol. 84, 2015, pp. 67–89.

[6] R. Raffaeli, A. Savoretti and M. Germani, Design knowledge formalization to shorten the time to generate offers for Engineer To Order products, *Advances on Mechanics, Design Engineering and Manufacturing*, 2016, pp. 1107–1114.

[7] L. Lin, I. Rasovska, R. De Guio and S. Dubois, Optimization Methods for Inventive Design, *TRIZ – The Theory of Inventive Problem Solving*, 2017, pp. 151–185.

[8] J. Sobieszczanski-Sobieski, A. Morris and M. Van Tooren, *Multidisciplinary design optimization supported by knowledge based engineering*, Wiley, Hoboken, 2015.

[9] G. La Rocca, L. Krakers and M. van Tooren, Development of an ICAD Generative Model for Blended Wing-Body Aircraft Design, *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2002.

[10] V. Kumar and Y.-J. Lin, A framework for intelligent backtracking in logic programs, *Foundations of Software Technology and Theoretical Computer Science*, 1986, pp. 108–123.

[11] D. R. Morrison, S. H. Jacobson, J. J. Sauppe and E. C. Sewell, Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning, *Discrete Optimization*, Vol. 19, 2016, pp. 79–102.

[12] K. Mehlhorn and P. Sanders, *Algorithms and Data Structures: The Basic Toolbox*, Springer, Berlin, 2008.

[13] C. Schulte and G. Tack, Views and Iterators for Generic Constraint Implementations, *Recent Advances in Constraints*, 2006, pp. 118–132.

[14] C. Münzer, Constraint-Based Methods for Automated Computational Design Synthesis of Solution Spaces, PhD thesis, ETH Zürich, 2015.

[15] N. Nethercote, P. J. Stuckey, R. Becket, S. Brand, G. J. Duck and G. Tack, "MiniZinc: Towards a Standard CP Modelling Language, *Lecture Notes in Computer Science*, 2007, pp. 529–543.

[16] T. Urli, et al., A General Local Search Solver for FlatZinc, *Metaheuristics International Conference, Marocco*, 2015.

[17] R. Raffaeli, P. Cicconi, M. Mengoni and M. Germani, Modular Product Configuration: An Automatic Tool for Eliciting Design Knowledge From Parametric CAD Models, *Volume 1: 36th Design Automation Conference, Parts A and B*, 2010, DETC2010-28242, pp. 207-218.