Industrial and
Systems Engineering

A Novel Approach to Discrete Truss Design Problems Using Mixed Integer Neighborhood Search

Mohammad Shahabsafa, Ali Mohammad-Nezhad, Tamás Terlaky, and Luis Zuluaga

Dept. of Industrial and Systems Engineering
Lehigh University, Bethlehm, PA, USA

Sicheng He, John T. Hwang, and Joaquim R. R. A. Martins

Dept. of Aerospace Engineering
University of Michigan, Ann Arbor, MI, USA

ISE Technical Report 17T-015

LEHIGH
UNIVERSITY.

A Novel Approach to Discrete Truss Design Problems Using Mixed Integer Neighborhood Search

Mohammad Shahabsafa[1], Ali Mohammad-Nezhad[1], Tamás Terlaky[1], Luis Zuluaga[1], Sicheng He[2], John T. Hwang[2], and Joaquim R. R. A. Martins[2]

[1] Dept. of Industrial and Systems Engineering, Lehigh University, Bethlehm, PA, USA ,
[2] Dept. of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA

October 25, 2017

Abstract

Discrete truss sizing problems are very challenging to solve due to their combinatorial, non-linear, non-convex nature. Consequently, truss sizing problems become unsolvable as the size of the truss grows. In this paper, we focus on modeling and efficiently solving discrete truss sizing problems. We consider various mathematical formulations for the truss design problem with the objective to minimize the truss weight. The non-convex Euler buckling constraints and Hooke's law are also considered. We deal with the truss design problems where the cross-sectional areas of the bars take only discrete values. We propose novel Mixed Integer Linear Optimization (MILO) reformulations of the non-convex models. The resulting MILO models are not solvable with existing MILO solvers as the size of the problem grows. We propose a novel solution methodology to solve the MILO models, and present numerical experiments to demonstrate the power of the novel computational methodology.

1 Introduction

The truss design problem is concerned with the optimal selection of geometry, topology and sizing of a structural system (see, e.g., the review paper by Bendsøe et al. (1994)). Dorn et al. (1964) first used numerical optimization for the truss design problem. They used the ground structure approach, in which the optimal structure is a subset of a set of bars defined prior to solving the problem. They considered the single-load minimum weight truss design problem. Achtziger et al. (1992) considered the truss design problem and developed linear and quadratic optimization models using displacement variables only, with the goal to minimize compliance. Bendsøe and Ben-Tal (1993) considered the problem of minimizing the compliance for a given volume of the material in a truss, where the mathematical model is formulated in terms of the nodal displacements and bar cross-sectional areas. They developed a steepest descent algorithm to solve the problem.

Consider a truss design problem with the objective to minimize the total weight of the structure, and assume that the cross-sectional areas of the bars are continuous decision variables. If we only impose Hooke's law, force balance equations, and bounds on the stress as the constraints, then all the bars will be fully stressed in the optimal solution, and the model can be reformulated as a linear optimization problem. However, adding the Euler buckling constraints makes the problem non-convex and thus, harder to solve. Achtziger (1999a) proposed an optimization model for the minimum weight truss design problem taking into account yield stress and Euler buckling constraints. However, he did not consider the kinematic compatibility and the stress-strain relation in the model. Then Achtziger (1999b) developed a numerical approach to solve the model that was suggested by Achtziger (1999a).

One of the frequent restrictions in practice is that the cross-sectional areas of the bars cannot take an arbitrary value; instead they only take values from a predefined finite set. Achtziger and Stolpe (2007a,b,c) considered the minimum compliance truss design problem with bounds on the volume of the truss, where the cross-sectional areas of the bars only take values from a discrete set. They proposed a Mixed Integer Nonlinear Optimization model and used a branch and bound algorithm to find the global optimum of the problem. They solved continuous relaxations of the problem to obtain lower bounds for the optimal objective value. However, they did not consider the bounds on the stress and Euler buckling constraints in the design problem. Stolpe (2007) considered the minimum compliance problem with constraints on the displacements and total volume of the structure. He proposed Mixed Integer Linear Optimization (MILO) and Mixed Integer Quadratic Optimization reformulations using the techniques presented by Petersen (1971) and Glover (1975, 1984). Rasmussen and Stolpe (2008) used a branch-and-cut approach to solve the MILO formulation of the minimum weight truss design problem, taking into account the stress and displacement constraints. However, they did not consider the buckling constraints in the model. They solved a 2D L-shaped truss problem with 54 bars and 108 binary variables, and a 3D cantilever truss with 40 bars and 160 binary variables to global optimality. Mela (2014) investigated the minimum weight truss design problem, where he assumed that the cross-sectional areas of the bars are discrete. He proposed a MILO model taking into account the Euler buckling and kinematic stability constraints and used a MILO solver to solve the problem. Mela (2014) solved a 2D truss tower with 209 bars, 110 of which were *overlapping members*. Additionally, he solved a 2D L-shaped truss with 160 bars, 23 of which were overlapping. Stolpe (2004) suggested a mixed integer non-convex mathematical model for the minimum weight truss design problem with displacements, stress, and cross-sectional areas as variables, considering bounds on stress and cross-sectional areas, as well as Euler buckling constraints. Then, he used a branch and bound framework to obtain the global optimum of the truss problem instances with a maximum of 25 bars. Stolpe (2015) has recently published a review on truss design problems with discrete cross-sectional areas. He presented various models and different methods, including global optimization methods, heuristics and meta-heuristics to solve discrete truss design problems. To the best of our knowledge the minimum weight discrete truss design problem considering Hooke's law, bounds on the stress, and Euler buckling constraints have only been solved for small-scale instances. The main contribution of the paper is to enhance the mathematical models of the problem, and to propose an efficient solution methodology to approximately solve large-scale (with more than 12,000 binary variables) discrete truss design problems.

The rest of the paper is organized as follows. In Section 2, we review various truss design problem models assuming that the cross-sectional areas of the bars are continuous. Then we propose two alternative MILO models for the design problem, where the cross-sectional areas of the bars are discrete. In Section 3, we propose a set of cuts to strengthen the MILO model of the discrete design problem. In Section 4, we propose a novel solution methodology which enables us to solve the discrete truss design problems significantly faster. In Section 5, we provide numerical results to demonstrate the efficiency of the proposed solution methodology. Our conclusions are presented in Section 6.

2 Preliminary models

In this section, we first assume that the cross-sectional areas of the bars are continuous, and propose a variety of non-convex models for the truss design problem. Then, we refine the models to consider the case where the cross-sectional areas of the bars are discrete. We propose two alternative MILO models for the discrete truss design problem, which are used in Section 4 to solve the problem.

In a truss structure, some nodes are fixed at a point, while the others are free. The external force on the nodes results in a deformation which induces the internal force and so balances the external force on the free nodes. We consider a design problem where the topology is fixed and the total weight of the truss is minimized. The decision variables are the cross-sectional areas (design variables), as well as internal forces, nodal displacements, and the bars' stress (state variables).

2.1 Continuous cross-sectional areas

In this section, we assume that the cross-sectional areas of the bars are continuous decision variables. In general, the truss design problem can be formulated as a non-linear non-convex optimization problem (see, e.g., Stolpe (2004)). Next, we present three equivalent mathematical models for the continuous truss design problem.

Let m be the number of bars in the truss, $I = \{1, \dots, m\}$, and n be the degree of freedom, which is defined as the number of free nodes $\times$ dimension, i.e.,

$$n = (\text{\# of nodes} - \text{\# of fixed nodes}) \times \text{dim. of the space}.$$

The vector $x \in \mathbb{R}^m$ denotes the cross-sectional areas of the bars, and $q \in \mathbb{R}^m$ is the vector of the internal forces on the bars. The stress in bar $i \in I$ is defined as

$$\sigma_i = \begin{cases} \dfrac{q_i}{x_i} & \text{if } x_i > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The vector of external forces exerted on the nodes is denoted by $\boldsymbol{f} \in \mathbb{R}^n$. The equilibrium between the internal forces and the external forces applied to the free nodes is maintained as a result of the *force balance* equation (see, e.g., De Klerk et al. (1995))

$$\boldsymbol{Rq} = \boldsymbol{f}, \tag{2}$$

where $\boldsymbol{R} \in \mathbb{R}^{n\times m}$ is the topology matrix associated with the design problem. The i^{th} column of $\boldsymbol{R}$, denoted by $\boldsymbol{r_i}$, is the vector representing the topology of the i^{th} bar in the truss for all $i \in I$.

Let $\boldsymbol{u} \in \mathbb{R}^n$ denote the displacement vector of the nodes, and $\boldsymbol{\delta} \in \mathbb{R}^m$ denote the elongation of the bars which depends on the displacement vector $\boldsymbol{u}$ as follows

$$\boldsymbol{\delta} = \boldsymbol{R}^T\boldsymbol{u}. \tag{3}$$

Let l_i and E_i, for $i \in I$, denote the length and Young's modulus of bar i, respectively. We can restate the internal force q_i as a function of cross-sectional area x_i and elongation δ_i. This relationship is governed by Hooke's law

$$q_i = E_i \frac{\delta_i}{l_i} x_i, \tag{4}$$

for all $i \in I$. Let matrix $\boldsymbol{K_i} \in \mathbb{R}^{n\times n}$, for $i \in I$, be the contribution of the bar $i \in I$ to the global stiffness matrix, defined as

$$\boldsymbol{K_i} = \frac{E_i}{l_i}\boldsymbol{r_i}\boldsymbol{r_i}^T.$$

Note that by definition, stiffness matrices are positive semidefinite. We may assume that the truss structure is stable, hence $\boldsymbol{K}(\boldsymbol{x})$ is positive definite, denoted as $\boldsymbol{K}(\boldsymbol{x}) \succ 0$. As mentioned below, this property will be enforced by considering positive lower bounds on the bars' cross-sectional areas. From Equations (2), (3), and (4), we can derive the following relationship

$$\boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{f}, \tag{5}$$

where $\boldsymbol{K}(\boldsymbol{x}) \in \mathbb{R}^{n\times n}$ is the stiffness matrix of the truss, defined as

$$\boldsymbol{K}(\boldsymbol{x}) = \sum_{i=1}^{m} x_i \boldsymbol{K_i}. \tag{6}$$

Lower and upper bounds are considered for the stress of the bars. Let $\boldsymbol{\sigma}^{\mathbf{min}},\ \boldsymbol{\sigma}^{\mathbf{max}} \in \mathbb{R}^m$ be the given lower and upper bounds on the bar's stress, respectively. The bounds $\boldsymbol{\sigma}^{\mathbf{min}}$ and $\boldsymbol{\sigma}^{\mathbf{max}}$ are actually bounds on the compression and tension of the bars, respectively. Therefore, we have $\boldsymbol{\sigma}^{\mathbf{max}} > \mathbf{0}$ and $\boldsymbol{\sigma}^{\mathbf{min}} < \mathbf{0}$. Moreover, we consider lower and upper bounds on the cross-sectional areas of the bars, namely, $\boldsymbol{x}^{\mathbf{min}},\ \boldsymbol{x}^{\mathbf{max}} \in \mathbb{R}^m$. We also consider the truss sizing problem, in which $x_i^{\min} > 0$ for all $i \in I$.

Furthermore, we consider the Euler buckling constraints, which are defined as

$$\sigma_i \geq \sigma_i^E, \qquad i \in I,$$

where σ_i^E is the Euler buckling stress for bar $i \in I$. We assume that the cross-sectional areas of the bars are circular, and both ends of all the bars are pinned. Then the Euler buckling stress of bar $i \in I$ is given by

$$\sigma_i^E = -\frac{\pi^2 E_i}{\left(\frac{l_i}{\tau_i}\right)^2},$$

where τ_i is the radius of bar $i \in I$. Let $\gamma_i = \pi E_i / l_i^2$. Then we have $\sigma_i^E = -\gamma_i x_i$, and the Euler buckling constraints can be written as

$$\sigma_i + \gamma_i x_i \geq 0, \qquad i \in I. \tag{7}$$

Using equation (1), we can also write the Euler buckling constraints as

$$q_i + \gamma_i x_i^2 \geq 0, \qquad i \in I. \tag{8}$$

Notice that neither Hooke's law nor the Euler buckling constraints (8) are convex. This gives rise to the following non-convex quadratic optimization formulation of the truss design problem.

$$\begin{array}{llll}
\min & \boldsymbol{l}^T \boldsymbol{x} & & \\
\text{s.t.} & \boldsymbol{R}\boldsymbol{q} & = \boldsymbol{f}, & \\
& q_i - \frac{E_i}{l_i}(\boldsymbol{r_i}^T \boldsymbol{u}) x_i & = 0, & i \in I, \\
& q_i + \gamma_i x_i^2 & \geq 0, & i \in I, \\
& \sigma_i^{\min} x_i \leq \quad q_i & \leq \sigma_i^{\max} x_i, & i \in I, \\
& x_i^{\min} \leq \quad\ x_i & \leq x_i^{\max}, & i \in I.
\end{array} \tag{P$_1$}$$

Throughout, we assume without loss of generality that all bars have the same density. Thus minimizing the total volume of the bars used in the structure is equivalent to minimizing the total weight of the structure. Model (P$_1$) has m non-convex equalities, and m non-convex inequalities. Each of the m non-convex equalities, has n bilinear terms. However, the Hooke's law constraint $q_i - \frac{E_i}{l_i}(\boldsymbol{r_i}^T \boldsymbol{u}) x_i = 0$, $i \in I$, can be used to derive a new formulation in terms of the cross-sectional areas $\boldsymbol{x} \in \mathbb{R}^m$ and the nodal displacements $\boldsymbol{u} \in \mathbb{R}^n$ as follows:

$$\begin{array}{llll}
\min & \boldsymbol{l}^T \boldsymbol{x} & & \\
\text{s.t.} & \boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} & = \boldsymbol{f}, & \\
& \dfrac{E_i}{l_i}(\boldsymbol{r_i}^T \boldsymbol{u}) & = \sigma_i, & i \in I, \\
& \sigma_i + \gamma_i x_i & \geq 0, & i \in I, \\
& \sigma_i^{\min} \leq \sigma_i & \leq \sigma_i^{\max}, & i \in I, \\
& x_i^{\min} \leq x_i & \leq x_i^{\max}, & i \in I,
\end{array} \tag{P$_2$}$$

where all the nonlinearity of the problem is encapsulated in n non-convex equalities, that is the $\boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{f}$ constraints, which include mn bilinear terms. However, we can decrease the number of bilinear terms by adding the internal forces $\boldsymbol{q} \in \mathbb{R}^m$ and equation (1) back to the model. By doing so, problem (P$_1$) can be reformulated as follows:

$$
\begin{array}{llll}
\min & \boldsymbol{l}^T\boldsymbol{x} & & \\
\text{s.t.} & \boldsymbol{Rq} & = \boldsymbol{f}, & \\
& \sigma_i - \dfrac{E_i}{l_i}\boldsymbol{r_i}^T\boldsymbol{u} & = 0, & i \in I, \\
& q_i - \sigma_i x_i & = 0, & i \in I, \\
& \sigma_i + \gamma_i x_i & \geq 0, & i \in I, \\
& \sigma_i^{\min} \leq \quad \sigma_i & \leq \sigma_i^{\max} & i \in I, \\
& x_i^{\min} \leq \quad x_i & \leq x_i^{\max} & i \in I.
\end{array}
\tag{P$_3$}
$$

Model (P$_3$) has m non-convex equalities, each of which has only one bilinear term. As a result model (P$_3$) is simpler than models (P$_1$) and (P$_2$). We utilize model (P$_3$) in Section 2.2 to derive the mathematical optimization model for the truss where the cross-sectional areas are discrete.

2.2 Discrete cross-sectional area

It is assumed in problems (P$_1$), (P$_2$), and (P$_3$) that the cross-sectional areas are continuous decision variables. In reality, however, the cross-sectional areas are frequently chosen from a discrete set, corresponding to sizes in which bars are pre-produced (see e.g., Achtziger and Stolpe (2006, 2007a), Cerveira et al. (2009)). Thus, x_i for $i \in I$ takes values from the finite set

$$S_i = \{s_{i1}, s_{i2}, \ldots, s_{ip_i}\}, \tag{9}$$

where $0 < s_{i1} < s_{i2} < \ldots < s_{ip_i}$. Let $\delta_{ik} := s_{i,k+1} - s_{ik}$, and $P_i = \{1, \ldots, p_i\}$. We propose two discrete modeling approaches for the discrete set (9) which are referred to as *basic discrete model* and *incremental discrete model.*

Basic discrete model: The cross-sectional area of bar i is defined as

$$
\begin{array}{ll}
x_i = \sum\limits_{k=1}^{p_i} s_{ik} z_{ik}, & i \in I, \\
\sum\limits_{k=1}^{p_i} z_{ik} = 1, & i \in I,\ k \in P_i, \\
z_{ik} \in \{0, 1\}, & i \in I,\ k \in P_i.
\end{array}
\tag{10}
$$

Incremental discrete model: The cross-sectional area of bar i is defined as

$$
\begin{array}{ll}
x_i = s_{i1} + \sum\limits_{k=1}^{p_i-1} \delta_{ik} z_{ik}, & i \in I, \\
z_{ik} \geq z_{i,k+1}, & i \in I,\ k = 1, \ldots, p_i - 1, \\
z_{ik} \in \{0, 1\} & i \in I,\ k \in P_i.
\end{array}
\tag{11}
$$

In the basic model, binary variables represent the choice from the discrete set of the cross-sectional areas, while in the incremental model, binary variables represent the increments in the cross-sectional areas of the bars. If $z_{ik} = 1$ in the basic discrete model, then $x_i = s_{ik}$, and $z_{ij} = 0$ for $j \neq k$. However, if $z_{ik} = 1$ in the incremental discrete model, then $x_i > s_{ik}$, and $z_{ij} = 1$ for $j < k$. In Section 5.2, we compare the basic and incremental models and see that the incremental model is a better modeling approach when solving the discrete truss design problem.

Next, we explain the derivation of the MILO formulation for problem (P$_3$) considering the discrete models (10) and (11). The idea can be used to reformulate problems (P$_1$) and (P$_2$) analogously. We start by substituting the basic discrete model (10) in the constraint $q_i - \sigma_i x_i = 0$ for all $i \in I$. Then we have

$$q_i - \sum_{k=1}^{p_i} s_{ik} z_{ik} \sigma_i = 0,\ i \in I. \tag{12}$$

Here, for all i and k we have the multiplication of a binary variable and a bounded continuous variable, i.e., $\sigma_i z_{ik}$. Using the idea introduced by Petersen (1971) and Glover (1975, 1984), we can linearize constraint

(12) by introducing auxiliary variables $\psi_{ik} = \sigma_i z_{ik}$ and adding the following constraints:

$$\begin{array}{rcl} z_{ik}\sigma_i^{\min} & \le \psi_{ik} \le & z_{ik}\sigma_i^{\max}, \\ \sigma_i - \sigma_i^{\max}(1-z_{ik}) & \le \psi_{ik} \le & \sigma_i - \sigma_i^{\min}(1-z_{ik}), \end{array}$$

for all $i \in I, k = 1, \dots, p_i$. Then using the basic discrete model (10), problem (P$_3$) can be reformulated as the following MILO problem with the decision variables $\boldsymbol{x} \in \mathbb{R}^m$, $\boldsymbol{z} \in \mathbb{R}^{\sum_i p_i}$, $\boldsymbol{u} \in \mathbb{R}^n$, $\boldsymbol{q} \in \mathbb{R}^m$, $\boldsymbol{\sigma} \in \mathbb{R}^m$, and $\boldsymbol{\psi} \in \mathbb{R}^{\sum_i p_i}$.

$$\begin{array}{llll} \min & \sum\limits_{i\in I} l_i x_i & & \\ \text{s.t.} & \boldsymbol{Rq} & = \boldsymbol{f}, & \\ & x_i - \sum\limits_{k=1}^{p_i} s_{ik} z_{ik} & = 0, & i \in I \\ & \sigma_i - \dfrac{E_i}{l_i}\boldsymbol{r_i}^T\boldsymbol{u} & = 0, & i \in I, \\ & q_i - \sum\limits_{k=1}^{p_i} s_{ik}\psi_{ik} & = 0, & i \in I, \\ & \sigma_i + \gamma_i x_i & \ge 0, & i \in I, \\ & \sigma_i^{\min} \le \sigma_i & \le \sigma_i^{\max}, & i \in I, \\ & z_{ik}\sigma_i^{\min} \le \psi_{ik} & \le z_{ik}\sigma_i^{\max}, & i \in I,\ k \in P_i, \\ & \sigma_i - \sigma_i^{\max}(1-z_{ik}) \le \psi_{ik} & \le \sigma_i - \sigma_i^{\min}(1-z_{ik}), & i \in I,\ k \in P_i, \\ & \sum\limits_{k=1}^{p_i} z_{ik} & = 1, & i \in I, \\ & z_{ik} & \in \{0,1\}, & i \in I,\ k \in P_i. \end{array} \tag{13}$$

In a similar manner, we can consider the incremental discrete model (11) and reformulate problem (P$_3$) as the following MILO model.

$$\begin{array}{llll} \min & \sum\limits_{i\in I} l_i x_i & & \\ \text{s.t.} & \boldsymbol{Rq} & = \boldsymbol{f}, & \\ & x_i - s_{i1} - \sum\limits_{k=1}^{p_i-1} \delta_{ik} z_{ik} & = 0, & i \in I, \\ & \sigma_i - \dfrac{E_i}{l_i}\boldsymbol{r_i}^T\boldsymbol{u} & = 0, & i \in I, \\ & q_i - s_{i1}\sigma_i - \sum\limits_{k=1}^{p_i-1} \delta_{ik}\psi_{ik} & = 0, & i \in I, \\ & \sigma_i + \gamma_i x_i & \ge 0, & i \in I, \\ & \sigma_i^{\min} \le \sigma_i & \le \sigma_i^{\max}, & i \in I, \\ & z_{ik}\sigma_i^{\min} \le \psi_{ik} & \le z_{ik}\sigma_i^{\max}, & i \in I,\ k \in \overline{P}_i, \\ & \sigma_i - \sigma_i^{\max}(1-z_{ik}) \le \psi_{ik} & \le \sigma_i - \sigma_i^{\min}(1-z_{ik}), & i \in I,\ k \in \overline{P}_i, \\ & z_{ik} & \ge z_{i,k+1}, & i \in I,\ k \in \overline{\overline{P}}_i, \\ & z_{ik} & \in \{0,1\}, & i \in I,\ k \in \overline{P}_i, \end{array} \tag{14}$$

where $\overline{P}_i = \{1, 2, \dots, p_i - 1\}$ and $\overline{\overline{P}}_i = \{1, 2, \dots, p_i - 2\}$. Notice that the incremental model has one less binary variable for each bar.

3 Reformulating the MILO models

In this section, we propose a set of cuts that can be used to obtain better (in terms of solution time) MILO formulations for (13) and (14). In particular, these cuts can replace the Euler buckling constraints defined in (7). Next, we derive the cuts for model (14). The derivation of the cuts for model (13) can be done in a similar fashion.

Considering the discrete model (11), the Euler buckling constraint can be rewritten as

$$\sigma_i + \gamma_i \left(s_{i1} + \sum_{k=1}^{p_i-1} \delta_{ik} z_{ik} \right) \geq 0,\ i \in I. \tag{15}$$

For each bar $i \in I$, the Euler buckling constraint enforces an inequality on z_{ik} for $k = 1, \ldots, p_i$. We replace the Euler buckling constraint (15) by p_i constraints each of which is formulated using only one binary variable.

Theorem 1. *The following set of constraints is equivalent to the Euler buckling constraint* (15) *for incremental discrete model* (11)*.*

$$\begin{array}{lll} \sigma_i + \gamma_i s_{ij}(1 - z_{ij}) - \sigma_i^{\min} z_{ij} & \geq 0, & j = 1, \ldots, p_i - 1, \\ \sigma_i + \gamma_i s_{i,p_i} z_{i,p_i-1} - \sigma_i^{\min}(1 - z_{i,p_i-1}) & \geq 0. & \end{array} \tag{16}$$

Proof. Suppose $x_i = s_{ik}$ for $k = 1 \ldots, p_i - 1$ and $i = 1, \ldots, m$. Then the Euler buckling constraint (15) reduces to

$$\sigma_i + \gamma_i s_{ik} \geq 0. \tag{17}$$

From (11) we know that $z_{i1} = \ldots = z_{i,k-1} = 1$, and $z_{ik} = z_{i,k+1} = \ldots = z_{i,p_i-1} = 0$. Additionally, the set of constraints (16) is equivalent to

$$\begin{array}{lll} \sigma_i + \gamma_i s_{ij} & \geq 0, & j = k, \ldots, p_i - 1, \\ \sigma_i - \sigma_i^{\min} & \geq 0. & \end{array} \tag{18}$$

By the inequalities $\sigma_i - \sigma_i^{\min} \geq 0$, the constraints in (18) can be equivalently written as

$$\sigma_i \geq \max_{j=k,\ldots,p_i-2} \{-\gamma_i s_{ij}\}. \tag{19}$$

Constraint (19) can be written as $\sigma_i \geq -\gamma_i s_{ik}$, which is equivalent to the Euler buckling constraint (17).

Now, suppose that $x_i = s_{ip_i}$ for $i = 1, \ldots, m$. In this case, $z_{i1} = \ldots = z_{i,p_i-1} = 1$, and the set of constraints (16) is equivalent to

$$\begin{array}{ll} \sigma_i - \sigma_i^{\min} & \geq 0, \\ \sigma_i + \gamma_i s_{i,p_i} & \geq 0, \end{array}$$

which is equivalent to the Euler buckling constraint (15). □

□

In Section 5.3, we demonstrate that replacing the Euler buckling constraints with constraints (16) in truss instances with large discrete sets can, in most cases, decrease the solution time by more than 20% in average.

4 Solution methodologies

In this section, we present a methodology for solving discrete truss design problems (13) and (14). For the purpose of ease of presentation in what follows, we refer to these problems as the *original problems*, with *full discrete sets*.

By way of experimentation, it turns out that MILO solvers are not able to solve to optimality even small-sized 3D instances of the truss design problem (see also Stolpe (2015)). In Table 1, the solution time of optimizing instances of 3D truss design problems, using GuRoBi 7.0.2, is presented to demonstrate that they are indeed hard to solve. Note that the relative optimality gap threshold is 0.1%, and the size of the discrete set of cross-sectional areas for all the bars is 41. The solver is terminated after 24 hours of CPU time if it is not solved to optimality by then.

Table 1: Solving 3D-truss design problem instances directly using GuRoBi. For each bar 41 different cross-sectional areas are assumed.

# bars	# bin. var.	Time(s)	Weight	Opt. Gap
20	820	22.90	5259.01	0.00%
40	1640	13232.60	8609.02	0.00%
60	2460	86400.00	11554.46	4.73%
80	3280	86400.00	15188.46	6.25%
100	4100	86400.00	16232.28	10.55%

In light of Table 1, we present a new solution methodology, referred as Neighborhood Search MILO (NS-MILO). The methodology is based on sequentially solving MILO subproblems where the feasible set of each subproblem is a subset of the feasible set of the original MILO problem. The set of the discrete values of each bar at each subproblem is a subset of the full discrete set of that bar. In fact, the discrete values are chosen from the neighborhood of the given solution. Hence, each subproblem is easier to solve, due to its reduced size, than the original problem. The MILO subproblems are denoted by $\text{MILO}_k(\boldsymbol{x})$. Specifically, $\text{MILO}_k(\boldsymbol{x})$ is the MILO formulation of a subproblem of the discrete truss design problem, where the cardinality of the discrete set for each bar is at most k, and the discrete set is generated from the solution $\boldsymbol{x}$. Let $\hat{\mathcal{S}}_i$ be the discrete set of bar $i \in I$ in subproblem $\text{MILO}_k(\boldsymbol{x})$. We solve three different kinds of MILO subproblems:

- $\text{MILO}_2(\boldsymbol{x})$, for $x \in \mathbb{R}^n$, is the MILO formulation of the discrete design problem, where the size of the discrete set of cross-sectional areas for each bar is equal to two. The set $\hat{\mathcal{S}}_i$ for a $\text{MILO}_2(\boldsymbol{x})$ is defined by

$$\hat{\mathcal{S}}_i := \begin{cases} \{s_{i1}, s_{i2}\}, & \text{if } \ x_i < s_{i1}, \\ \{s_{ik}, s_{i,k+1}\}, & \text{if } \ s_{ik} \le x_i < s_{i,k+1}, \\ \{s_{i,p_i-1}, s_{ip_i}\}, & \text{if } \ x_i \ge s_{ip_i}. \end{cases}$$

- $\text{MILO}_3(\boldsymbol{x})$ for $x_i \in \mathcal{S}_i,\ i \in I$, is the MILO formulation of the discrete design problem where the size of the discrete set for each bar is at most three. Suppose that $x_i = s_{ik}$. Then the set $\hat{\mathcal{S}}_i$ for a $\text{MILO}_2(\boldsymbol{x})$ is defined by

$$\hat{\mathcal{S}}_i := \begin{cases} \{s_{i1}, s_{i2}\}, & \text{if } \ k = 1, \\ \{s_{i,k-1}, s_{i,k}, s_{i,k+1}\}, & \text{if } \ 2 \le k \le p_i - 1, \\ \{s_{i,p_i-1}, s_{i,p_i}\}, & \text{if } \ k = p_i. \end{cases}$$

- $\text{MILO}_5(\boldsymbol{x})$ for $x_i \in \mathcal{S}_i,\ i \in I$, is the MILO formulation of the discrete design problem where the size of the discrete set for each bar is at most five. Suppose that $x_i = s_{ik}$. Then the set $\hat{\mathcal{S}}_i$ for a $\text{MILO}_5(\boldsymbol{x})$ is defined as follows

$$\hat{\mathcal{S}}_i = \begin{cases} \{s_{i1}, s_{i2}, s_{i3}\}, & \text{if } \ k = 1, \\ \{s_{i1}, s_{i2}, s_{i3}, s_{i4}\}, & \text{if } \ k = 2, \\ \{s_{i,k-2}, s_{i,k-1}, s_{i,k}, s_{i,k+1}, s_{i,k+2}\}, & \text{if } \ 3 \le k \le p_i - 2, \\ \{s_{i,p_i-2}, s_{i,p_i-1}, s_{i,p_i}, s_{i,p_i+1}\}, & \text{if } \ k = p_i - 1, \\ \{s_{i,p_i-2}, s_{i,p_i-1}, s_{i,p_i}\}, & \text{if } \ k = p_i. \end{cases}$$

Suppose that the original discrete set of the bar $i \in I$ is defined in (9). We start the NS-MILO approach by solving the continuous model (P_3) using the nonlinear solver IPOPT 3.14 (Wächter and Biegler 2006). The local optimal solution of the model (P_3) is denoted by $\boldsymbol{x^0}$. Then we solve a sequence of MILO_2 subproblems. Let $\alpha = 1$. We solve $\text{MILO}_2(\alpha\boldsymbol{x^0})$ by a MILO solver considering a predefined limit on the solution time. If an integer feasible solution is not found within the time limit, then we increase α by 0.05, and again solve

MILO$_2(\alpha\boldsymbol{x}^{\mathbf{0}})$. We continue solving MILO$_2(\alpha\boldsymbol{x}^{\mathbf{0}})$ until an integer feasible solution is found. Let $\hat{\boldsymbol{x}}$ be the best integer feasible solution obtained for MILO$_2(\alpha\boldsymbol{x}^{\mathbf{0}})$.

Next, we generate MILO$_3(\hat{\boldsymbol{x}})$. Using a MILO solver, we solve MILO$_3(\hat{\boldsymbol{x}})$ until a better solution than the current best solution is found. The improved solution to MILO$_3(\hat{\boldsymbol{x}})$ is assigned to $\hat{\boldsymbol{x}}$. As soon as a better solution is found, we stop the solver, and use the improved solution $\hat{\boldsymbol{x}}$ to generate the next MILO$_3$ subproblem. We continue solving MILO$_3(\hat{\boldsymbol{x}})$ until the objective function does not improve.

Afterwards, we solve MILO$_5(\hat{\boldsymbol{x}})$, and similarly solve MILO$_5(\hat{\boldsymbol{x}})$ until a better solution than the current best solution is found. We stop the solver as soon as the better solution is found, and use the improved solution to generate the next MILO$_5$ subproblem. We continue solving MILO$_5(\hat{\boldsymbol{x}})$ until the objective function does not improve.

Notice that MILO$_3$ and MILO$_5$ subproblems are not solved to optimality, since MILO solvers spend a significant portion of time improving the lower bound of the objective value and proving the optimality of the best integer solution obtained. However, proving that a solution is optimal for the subproblem does not help in solving the original discrete problem. Therefore, we stop the solver as soon as a better feasible solution is found, and use that solution to generate the next subproblem.

The approach, described above, to generate and solve MILO subproblems sequentially is in fact a moving-neighborhood search, where we search for better integer feasible solutions in the neighborhood of the best solution found so far. This neighborhood search approach to solve the truss design problem is summarized in Algorithm 1. Notice that in Algorithm 1, Solve(P) returns the optimal solution of subproblem P, while FindSol(P) returns a better solution as soon as it finds one, or returns the solution that was previously found. If FindSol(P) returns the previously found solution, it indicates that either that solution is optimal for the subproblem, or the time limit of solving subproblem P is reached. Also, note in $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ Solve(P) and $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(P), that $\hat{\boldsymbol{x}}$ is the solution that is returned and $\hat{\eta}$ is the weight of the solution $\hat{\boldsymbol{x}}$.

Algorithm 1 The NS-MILO approach

```
1:  $\boldsymbol{x}^{\mathbf{0}}$ := local optimal solution of the continuous model
2:  $\alpha := 1$
3:  **repeat**
4:      $\bar{\boldsymbol{x}} := \alpha\boldsymbol{x}^{\mathbf{0}}$
5:      $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ Solve(MILO$_2(\bar{\boldsymbol{x}})$)
6:      $\alpha := \alpha + 0.05$
7:  **until** MILO$_2(\bar{\boldsymbol{x}})$ is feasible
8:  **repeat**
9:      $\eta_{\text{curr}} := \hat{\eta}$
10:     $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(MILO$_3(\hat{\boldsymbol{x}})$)
11: **until** $\hat{\eta} = \eta_{\text{curr}}$
12: **repeat**
13:     $\eta_{\text{curr}} := \hat{\eta}$
14:     $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(MILO$_5(\hat{\boldsymbol{x}})$)
15: **until** $\hat{\eta} = \eta_{\text{curr}}$
16: **return** $\hat{\boldsymbol{x}}$
```

It is worth mentioning that solving MILO$_k$ subproblems can be done for bigger values of k ($k = 7, 9, \ldots$). However, in our experiments, considering these subproblems did not help to significantly improve the solution, when one considers the time that was spent on solving those larger neighborhood subproblems.

5 Numerical results

In this section, three different classes of truss design problems are considered: 2D cantilevered truss problems, 3D cantilevered truss problems, and truss models of an airplane wing. All three cases are available online for benchmark usage. The numerical results are run on a machine with Intel Xeon® CPU E5-2630 @ 2.20

GHz and 64 GB of RAM. GuRoBi 7.0.2 2016 is used to solve the MILO models and the optimality gap is set to 0.1%, i.e., when the gap between the best found solution and the lower bound is less than 0.1%, GuRoBi stops and returns the best solution found up to that point. GuRoBi is set to use 16 threads in solving all the problems and subproblems in this section.

5.1 The instances

In this section, we define the three different classes of problems that are used to evaluate the models and the solution methodology. We assume that all the bars in the test instances are made of aluminium. The Young's modulus for all the bars is equal to 69 GPa, and the yield stress σ_Y is 2.7MPa, thus, $\sigma_i^{\min} = -2.7$MPa, and $\sigma_i^{\max} = 2.7$MPa for $i \in I$.

5.1.1 The 2D cantilever instances

The 2D cantilever instances are the simplest instances we consider. Each block is a square with its diagonals. Thus each block has five bars. The length of each block is 50 cm. The 2D cantilever instance with 3 blocks is illustrated in Figure 1.

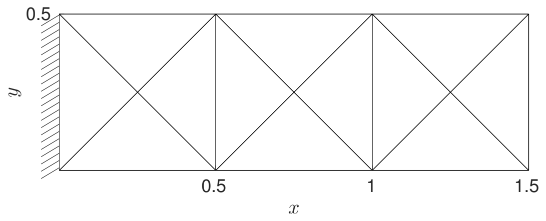


Figure 1: 2D cantilever instance with 3 blocks.

The external force is generated randomly at each node from a given interval using a uniform distribution. Let $f_0 = 1.25 \times 10^5/\ell$, where ℓ is the number of the blocks of the cantilever instance. For all the bottom nodes, the y coordinate of the force randomly takes values in the interval $[-f_0, 0]$, and the x coordinate takes value in the interval $[-f_0/10, f_0/10]$. For the top nodes, the y and x coordinates take value in the intervals $[-f_0/10, 0]$ and $[-f_0/100, f_0/100]$, respectively. Hence, the dominant coordinate of the force at each node is the y direction with a negative sign. The average of the force on the bottom nodes is 10 times bigger than that of the top nodes. Additionally, bars can take 41 different cross-sectional areas in the range $[0.25 - 85]$ cm^2.

5.1.2 The 3D cantilever instances

The 3D cantilever instance is an extension of the 2D cantilever instance. Each block is a cube, and all the diagonals of each face are considered in the structure. Additionally, two of the main diagonals of each cube are considered. Each block has 20 bars. The 3D cantilever instance with 3 blocks is illustrated in Figure 2.

Similar to the 2D cantilever instance, the force is generated randomly at each node from a given interval using uniform distribution. Let $f_0 = 1.2 \times 10^6/\ell$, where ℓ is the number of the blocks of the cantilever instance. For all the bottom nodes, the y coordinate of the force randomly takes values in the interval $[-f_0, 0]$, and the x and z coordinates randomly take values in the interval $[-f_0/10, f_0/10]$. For the bottom nodes, the y, x, and z coordinates take values in the intervals $[-f_0/10, 0]$, $[-f_0/100, f_0/100]$, and $[-f_0/100, f_0/100]$ respectively. Hence, the dominant coordinate of the force at each node is the y direction with a negative sign. The average of the force on the bottom nodes is 10 times bigger than that of the top nodes. Additionally, bars can take 41 different cross-sectional areas in the range $[0.25 - 85]$ cm^2.

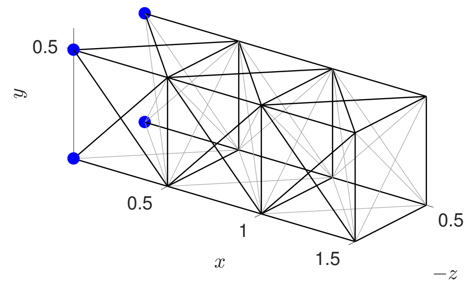


Figure 2: 3D cantilever instance with 3 blocks.

5.1.3 The Wing instances

We consider a 3D wing-truss problem. The truss layout is generated based on the undeformed common research model (UCRM) geometry (Kenway et al. 2014). For the aerodynamic load, we assume an elliptical distribution in the spanwise direction and a uniform distribution in chordwise direction. The total load of one wing is set to be one half of the maximum takeoff weight of the common research model (CRM). For simplicity, we also assumed that the aerodynamic load will not be affected by the structural deformation, i.e. the aeroelastic effect is negelected in this study. Additionally, bars can take 40 different cross-sectional areas in the range $[0.25 - 1000]$ cm^2.

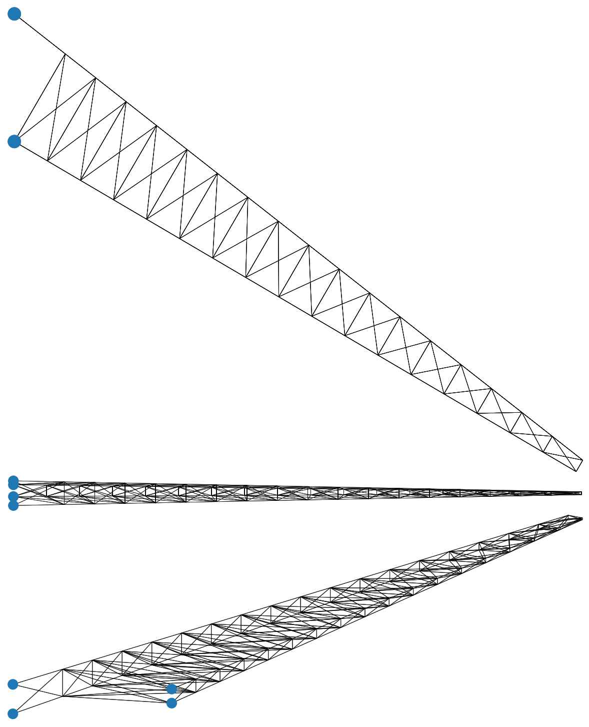


Figure 3: The wing instance with 315 bars.

5.2 Basic versus incremental model

In Table 2, 2D cantilever instances are solved with the basic and incremental model. Although the number of binary variables of the incremental model is not significantly less than that of the basic model, the incremental model is solved significantly faster than the basic model. In Figure 4, the objective function improvement of the basic and incremental models is plotted for the 2D cantilever instance with 10 blocks. As we can see, the incremental model stops at $t = 0.71$ hour, while the basic model stops at $t = 2.62$ hours. Additionally, the incremental model finds the optimal solution at $t = 0.20$ hr, while the basic model finds the optimal solution about 10 times slower at $t = 2.05$ hrs. Therefore, the incremental model is faster in proving optimality, and it finds the optimal solution significantly faster than the basic model. As a result, we use the incremental model through the rest of the paper.

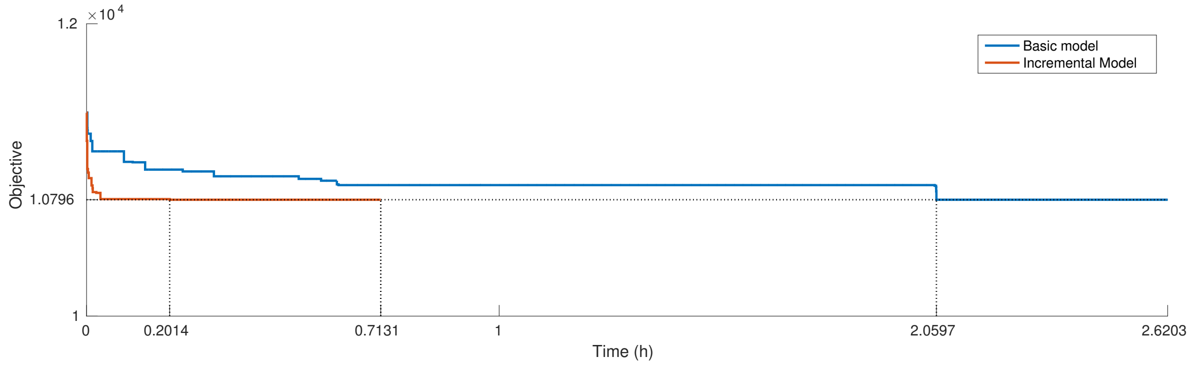


Figure 4: Comparison of the basic model and the incremental model for the 2D cantilever instance with 10 blocks.

Table 2: Comparison of the basic and incremental model.

# of blocks	Model	Bin. vars.	Weight	Run time
1	basic	185	591.05	0.39
	inc.	180	591.05	0.39
2	basic	370	1689.94	1.89
	inc.	360	1689.94	0.88
3	basic	555	2443.50	9.61
	inc.	540	2443.50	3.46
4	basic	740	3563.84	43.96
	inc.	720	3563.84	4.31
5	basic	925	3581.52	79.64
	inc.	900	3581.52	14.65
6	basic	1100	5480.78	480.98
	inc.	1080	5480.78	77.97
7	basic	1295	6657.19	797.98
	inc.	1260	6657.19	284.60
8	basic	1480	8699.13	1101.00
	inc.	1440	8699.13	692.10
9	basic	1665	9759.12	4004.81
	inc.	1620	9759.12	1096.82
10	basic	1850	10796.25	9433.64
	inc.	1620	10796.25	2567.24

5.3 Strengthening the Euler buckling constraints

Next, we numerically examine the effect of replacing the Euler buckling constraint introduced in (7) with constraints (16) introduced in Theorem 1.

Table 3: The impact of using constraints (16) to solve the 2D and 3D cantilever instances.

	n_b	m	Euler Const. (7)		Const.(16)		$\frac{\lvert\Delta t\rvert}{t_1}(\%)$
			weight	t_1	weight	t_2	
2D	5	25	3581.52	11.99	3581.52	8.84	26.3%
	6	30	5480.79	65.91	5480.79	62.02	5.9%
	7	35	6657.20	241.69	6657.20	230.25	4.7%
	8	40	8693.96	675.46	8693.96	379.25	43.9%
	9	45	9759.13	957.57	9759.13	960.11	-0.3%
	10	50	10796.26	2132.01	10796.26	1550.21	27.3%
	11	55	11904.10	6538.10	11904.10	5420.38	17.1%
	12	60	14419.26	9659.90	14419.26	14130.12	-46.3%
	13	65	13914.09	13557.87	13914.09	19060.53	-40.6%
	14	70	17625.48	76272.13	17622.45	44822.33	41.2%
	15	75	17294.42	112967.00	17294.42	24274.89	78.5%
3D	1	20	5259.01	13.34	5259.01	13.36	-0.1%
	2	40	8609.03	16438.45	8693.96	8305.35	49.5%

The impact of replacing the Euler buckling constraints (7) with constraints (16) is demonstrated in Table 3, where n_b, m, t_1, and t_2 denote the number of blocks, the number of bars, the solution time of the model with constraints (7), and the solution time of the model with constraints (16), respectively. As we can see in Table 3, replacing the Euler buckling constraints (7) with constraints (16), reduces the solution time in most instances. Specifically, for the largest 2D and 3D cantilever instances, the solution time decreases to at least half by replacing the Euler buckling constraints with constraints (16).

It is worth mentioning that replacing the Euler buckling constraints (7) with constraints (16) is useful for the truss design problems where we have a large discrete set for the cross-sectional areas. However, if the size of the discrete set is less than 5, the replacement does not help.

5.4 The NS-MILO approach

In this section, we compare the NS-MILO approach to solve discrete truss design problems with simply solving the original MILO problem by GuRoBi. We refer to the latter as the *full-MILO approach*. In all the experiments, the incremental model is used. The maximum solution time of the full-MILO approach is set to 24 hrs for all the instances that are solved in this section. The solution time limits for MILO$_2$, MILO$_3$, and MILO$_5$ subproblems are 300, 1800, and 1800 seconds respectively, except for the wing instances with more than 250 bars, where the solution time limits for MILO$_2$, MILO$_3$, and MILO$_5$ subproblems are 300, 3000, and 3000 seconds respectively. That means that our experiments compare the quality of the solution of full-MILO approach after 24 hrs (if optimal solution was not obtained earlier) with the solution of NS-MILO approach with time parameters mentioned above. We solve the 2D and 3D cantilever instances with 20 to 300 bars, and the wing instances with 81 to 315 bars. Results of the 2D and 3D cantilever instances are presented in Tables 4 and 5 respectively, and results of the wing instances are presented in Table 6. In Tables 4, 5,and 6, m, w_f, and t_f denote the number of bars, the weight of the solution of the full-MILO approach, and the time to obtain the solution of the full-MILO approach, respectively. Parameters n_s, w_n, and t_n denote the number of MILO subproblems, the weight of the solution, and the solution time of the NS-MILO approach, respectively. Additionally, in Tables 4 and 5 n_b is the number of blocks of the cantilever instances, and in Table 6, w_c, and t_c denote the weight of the solution of the continuous design problem, and the time to obtain that solution, respectively.

Table 4: The NS-MILO approach to solve the 2D Cantilever instances.

n_b	m	Full-MILO w_f	t_f	NS-MILO n_s	w_n	t_n	t_f/t_n
4	20	3139.21	7.0	4	3167.25	0.54	12.9
8	40	7540.41	389.4	7	7550.76	4.09	95.0
12	60	12342.86	9348.9	11	12342.86	9.60	973.7
16	80	18672.45	86400.0	7	18672.45	52.66	1661.5
20	100	26530.97	86400.0	11	26547.76	634.09	136.2
24	120	31626.83	86400.0	11	31626.83	1483.85	58.2
28	140	43511.60	86400.0	22	43521.96	163.32	530.0
32	160	60025.45	86400.0	20	60005.62	827.00	104.4
36	180	71253.63	86400.0	18	71292.02	2765.51	31.2
40	200	86625.17	86400.0	12	86720.88	1863.15	46.4
44	220	109194.93	86400.0	22	109197.96	2243.28	38.5
48	240	129893.61	86400.0	16	129816.46	2186.59	39.5
52	260	153665.14	86400.0	25	153656.04	4510.74	19.1
56	280	176077.06	86400.0	14	176311.53	3308.12	26.1
60	300	183910.42	86400.0	21	183932.39	3928.48	21.9

Among the 2D and 3D cantilever test instances, only the 2D instances with 20, 40 and 60 bars and the 3D instances with 20 and 40 bars are solved to proven optimality within the 24 hrs time limit of the full-MILO approach. Comparing the solutions obtained from the full-MILO approach for the instances that are solved to optimality with those of the NS-MILO approach indicates that the weights of the NS-MILO approach solutions are within 1% of the weights of the proven optimal solutions for these instances. From Tables 4 and 5, it is clear that the NS-MILO approach to solve the 2D and 3D cantilever instances is significantly faster (at least 10 times) than the full-MILO approach. In all the 2D wing instances, the difference between the solution of the full-MILO and NS-MILO approach is less than 0.25%. Therefore, we can say that the NS-MILO approach is able to find equally good solutions significantly faster for the 2D cantilever instances than the full-MILO approach.

Table 5: The NS-MILO approach to solve the 3D Cantilever instances.

n_b	m	Full-MILO w_f	t_f	NS-MILO n_s	w_n	t_n	w_f/w_n
1	20	5259.01	10.0	11	5259.01	1.71	1.000
2	40	8609.02	9304.1	8	8625.41	14.13	0.998
3	60	11554.46	86400.0	7	11592.81	2257.57	0.996
4	80	15188.46	86400.0	5	15188.46	444.39	1.000
5	100	16232.28	86400.0	13	16125.32	4409.26	1.006
6	120	31358.60	86400.0	11	31554.63	3772.27	0.993
7	140	42692.98	86400.0	4	42698.43	3920.88	0.999
8	160	49845.96	86400.0	16	49674.36	6469.37	1.003
9	180	68537.73	86400.0	13	63853.31	5289.92	1.073
10	200	98180.91	86400.0	17	71491.84	4581.25	1.373
11	220	117389.48	86400.0	11	85518.64	4433.90	1.372
12	240	137170.32	86400.0	22	103205.22	6999.67	1.329
13	260	158313.12	86400.0	23	110973.26	7367.39	1.426
14	280	453539.55	86400.0	17	148290.60	7948.12	3.058
15	300	475922.95	86400.0	19	184215.60	6565.33	2.583

Table 6: The NS-MILO approach to solve the wing instances.

# of bars	Full-MILO		Continuous model		NS-MILO			t_f/t_n	w_n/w_c	w_f/w_n
	w_f	t_f	w_c	t_c	n_s	w_n	t_n			
81	47983.76	86400.00	41017.74	27.98	25	42806.29	601.38	143.67	1.04	1.12
99	39625.54	86400.00	35437.73	17.14	51	37811.28	3392.55	25.47	1.07	1.05
117	40366.35	86400.00	31345.93	21.43	66	33763.94	4333.97	19.94	1.08	1.20
135	42747.63	86400.00	28901.67	28.96	21	32039.01	3223.35	26.80	1.11	1.33
153	39508.10	86400.00	27778.12	52.01	58	29857.79	7795.80	11.08	1.07	1.32
171	48580.45	86400.00	26804.88	95.78	61	28515.16	5982.08	14.44	1.06	1.70
207	60023.34	86400.00	25649.90	183.52	58	27827.38	7429.31	11.63	1.08	2.16
225	67995.80	86400.00	25154.75	149.62	49	27068.66	8162.62	10.58	1.08	2.51
243	35968.02	86400.00	24842.02	303.68	76	26634.80	13374.70	6.46	1.07	1.35
261	38129.39	86400.00	24636.71	401.85	74	26534.96	22896.89	3.77	1.08	1.44
279	82028.13	86400.00	24434.24	738.01	53	26507.51	15919.55	5.43	1.08	3.09
297	107335.74	86400.00	24322.06	750.06	57	26332.40	22040.78	3.92	1.08	4.08
315	45902.92	86400.00	24190.90	773.18	63	26591.64	28263.34	3.06	1.10	1.73

In 11 instances out of 15 of the 3D cantilevers, the total weight of the solution obtained from the NS-MILO approach is equal to, or lower than the weight of the solution obtained from the full-MILO approach. In the 4 instances, where the weight of the solution obtained from the NS-MILO approach is not less than that of the full-MILO approach, the difference is less than 1%. For the 3D cantilever instances with more than 150 bars, the solution of the NS-MILO approach is better than the solution provided by the full-MILO approach and the the ratio w_f/w_n increases as the size of the problem grows. For instance, the solution of the NS-MILO approach for the 3D instances with 280 and 300 bars is more than 2.5 times better than the solution of the full-MILO approach.

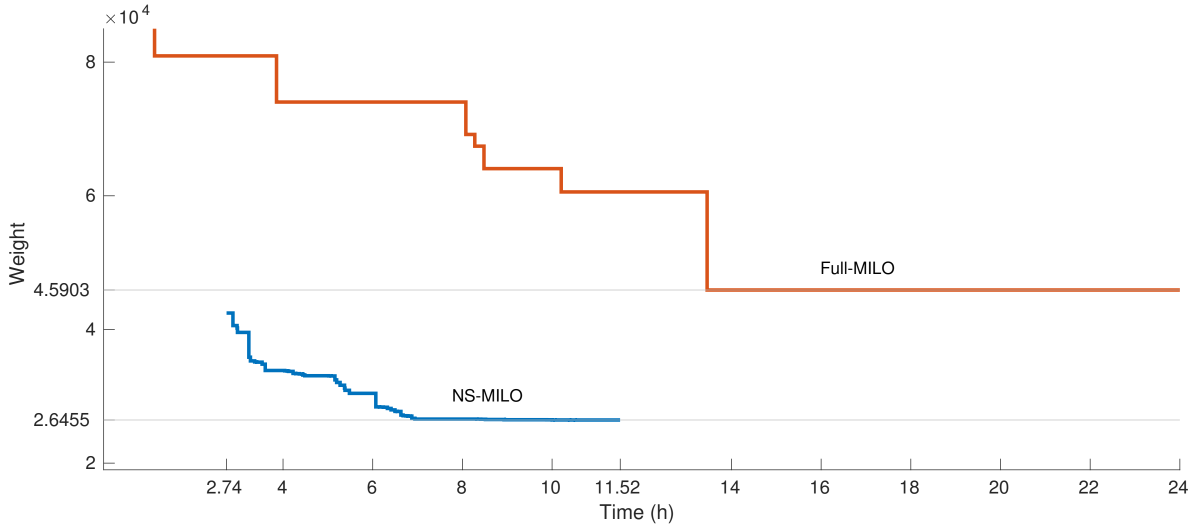


Figure 5: Comparison of Full-MILO and NS-MILO approach for the wing instance with 315 bars.

None of the wing instances were solved to optimality in 24 hrs using the full-MILO approach. We let the wing instances run longer than 24 hrs, however, for the wing instance with 207 bars, after 48 hrs, more than 64 GB of memory was used by the solver to store the nodes of the branch and bound tree, which renders the full-MILO approach inefficient for times beyond 48 hrs. This is because once the memory limit is reached, MILO solvers use the hard drive to store the branch and bound tree information, which results in an extremely slow process for the solver, due to the time spent on writing and accessing the hard drive.

Comparing the best solution of the full-MILO approach with the NS-MILO approach for the wing instances, we can see that the solution of the full-MILO approach is far from being optimal. Even though we stopped the full-MILO approach after one day, still the NS-MILO approach is significantly faster than the full-MILO approach. Note that the weight of the best solution obtained from the full-MILO approach is 5%-408% more than that of the NS-MILO approach for the wing instances.

As mentioned earlier, IPOPT is used to solve the continuous truss design problem. The weight of the solution of the continuous model is given in Table 6. We also solved the wing instances with the software package SNOPT (Gill et al. 2005), which gave the same solutions for the continuous model as IPOPT. We may entertain the idea that the solutions are the global optimal solutions of the continuous model, since IPOPT and SNOPT use the interior point method and sequential quadratic programming, respectively to solve the continuous model and have resulted in the same solution for all instances. If the solution of the continuous model is the global optimum of the problem then its weight provides a lower bound for the optimal solution of the discrete truss design problem. The gap between the weight of the continuous model solution and the weight of the NS-MILO solution is at most 10% for all the wing instances, thus the optimality gap for the NS-MILO solution is less than 10%.

In Figure 5, the objective function improvement of the full-MILO and NS-MILO approach for the wing instance with 315 bars is plotted. The weight of the initial solution of the full-MILO approach is 558,730, which is not a reasonable solution. The Full-MILO approach is plotted in Figure 5, only after a reasonable solution is obtained. As we can see, the full-MILO approach improves the objective function slowly, and after 24 hrs the weight of the best solution found is about 70% more than that of the NS-MILO approach. We can see that the NS-MILO approach took 2.74 hrs to find a feasible solution. The reason is that the primary MILO_2 subproblems were not able to find a feasible solution in the time limit of the MILO_2 subproblems, and the first feasible solution to a MILO_2 subproblem was found at $t = 2.74$ hrs. The first feasible solution of the NS-MILO approach is better than the solution of the full-MILO approach after 24 hrs.

In Figure 6, the solution times of the NS-MILO approach for 2D cantilever, 3D cantilever, and wing instances are plotted. As we can see in this figure, the solution time of the 3D cantilevers is more than that of the 2D cantilevers, and the solution time of the wing instances is significantly more than that of the 3D cantilevers. This reflects the increasing complexity of the three test sets. Additionally, the solution time does not increase exponentially as the size of the problem increases. It is worth mentioning that the total solution time of the NS-MILO approach is dependent on the time limit of the MILO_k subproblems, and if we increase the time limit on solving each subproblem, the total solution time of the NS-MILO approach may increase. However, as we can see in Tables 4, 5, and 6, the number of subproblems increases only moderately as the problem size increases.

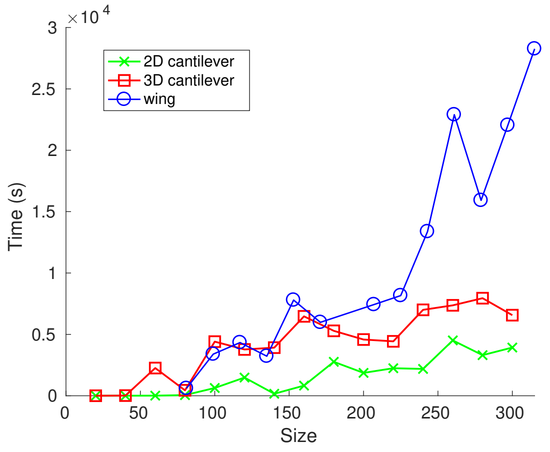
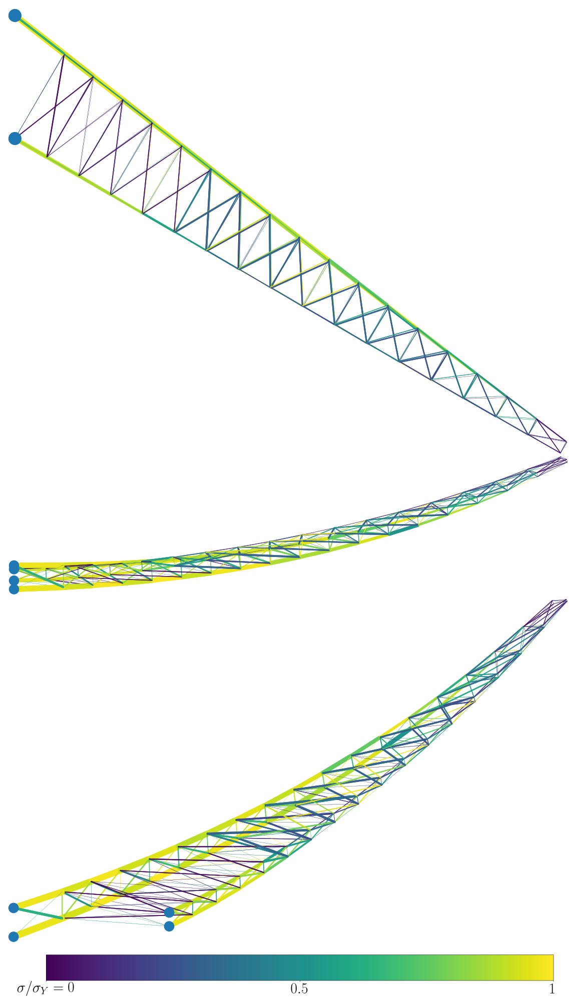
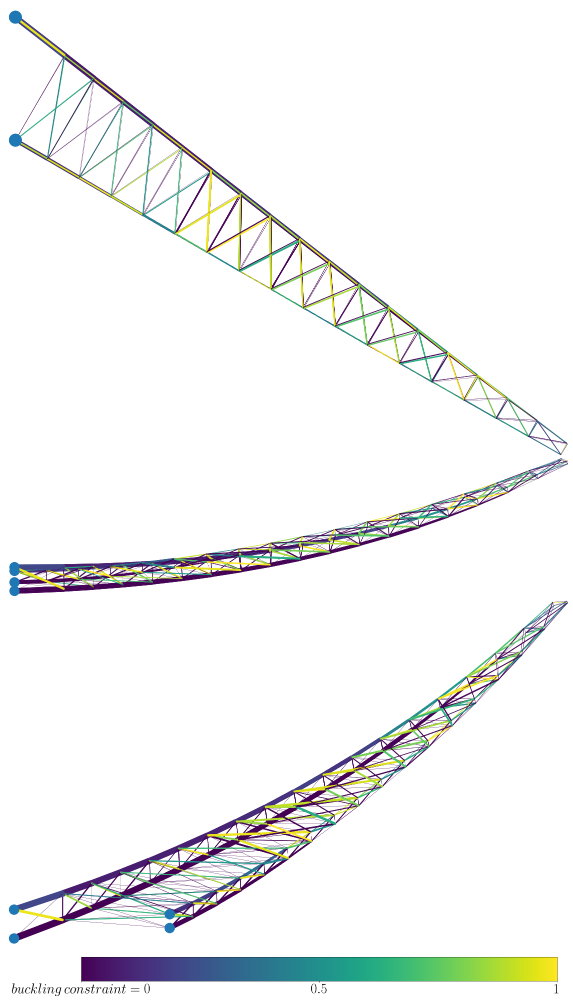


Figure 6: Solution time of NS-MILO vs. size of the truss.

H



a) Dimensionless stress distribution.



b) Dimensionless buckling constraint distribution.

Figure 7: Stress and buckling constraint distribution for the wing instance with 315 bars.

To get a better physical intuition, in Figure 7, we plot the dimensionless stress, $\sigma_i/|\sigma_Y|$, and dimensionless buckling constraints, $\max(-\sigma_i/\gamma_i x_i, 0)$ (stress in bars under tension is substituted with zero for simplicity) for the solution of the wing instance with 315 bars obtained from NS-MILO approach. For the stress constraints, they are more active for the horizontal bars close to the *root*. That is because those bars are mainly responsible to take the large bending moment around the root. As for the buckling constraints, they are more critical for the upper "*skin*" bars. The reason is that those bars take less load, so they are generally thinner and are under compression.

6 Conclusions

We presented novel MILO models for the discrete truss design problems that include both the Euler buckling and the Hooke's law constraints. We also proposed the novel NS-MILO methodology to solve those problems, where a sequence of MILO subproblems in a moving neighborhood search framework are solved. The new methodology enables us to solve previously unsolvable truss design problems. Numerical results indicate that the NS-MILO approach to solve discrete truss design problems is significantly faster than simply using a MILO solver to solve the original truss design problems. Additionally, for all the generated wing instances, the best solution found is significantly better than the solution obtained from attempting to solve directly the original problems.

Acknowledgement

Research is supported by Air Force Office of Scientific Research Grant #FA9550-15-1-0222.

References

Achtziger, W. (1999a). Local stability of trusses in the context of topology optimization part I: Exact modelling. *Structural Optimization*, 17(4):235–246.

Achtziger, W. (1999b). Local stability of trusses in the context of topology optimization part II: A numerical approach. *Structural Optimization*, 17(4):247–258.

Achtziger, W., Bendsøe, M. P., Ben-Tal, A., and Zowe, J. (1992). Equivalent displacement based formulations for maximum strength truss topology design. *IMPACT of Computing in Science and Engineering*, 4(4):315 – 345.

Achtziger, W. and Stolpe, M. (2006). Truss topology optimization with discrete design variables—guaranteed global optimality and benchmark examples. *Structural and Multidisciplinary Optimization*, 34(1):1–20.

Achtziger, W. and Stolpe, M. (2007a). Global optimization of truss topology with discrete bar areas—part I: theory of relaxed problems. *Computational Optimization and Applications*, 40(2):247–280.

Achtziger, W. and Stolpe, M. (2007b). Global optimization of truss topology with discrete bar areas—partII: Implementation and numerical results. *Computational Optimization and Applications*, 44(2):315–341.

Achtziger, W. and Stolpe, M. (2007c). Truss topology optimization with discrete design variables—guaranteed global optimality and benchmark examples. *Structural and Multidisciplinary Optimization*, 34(1):1–20.

Bendsøe, M. P. and Ben-Tal, A. (1993). Truss topology optimization by a displacements based optimality criterion approach. In Rozvany, G., editor, *Optimization of Large Structural Systems*, volume 231 of *NATO ASI Series*, pages 139–155. Springer Netherlands.

Bendsøe, M. P., Ben-Tal, A., and Zowe, J. (1994). Optimization methods for truss geometry and topology design. *Structural Optimization*, 7(3):141–159.

Cerveira, A., Agra, A., Bastos, F., and Gromicho, J. (2009). New branch and bound approaches for truss topology design with discrete areas. In *Proceedings of the American Conference on Applied Mathematics. Recent Advances in Applied Mathematics*, pages 228–233.

De Klerk, E., Roos, C., and Terlaky, T. (1995). *Semi-definite problems in truss topology optimization.* Delft University of Technology, Faculty of Technical Mathematics and Informatics, Report 95-128.

Dorn, W. S., Gomory, R. E., and Greenberg, H. J. (1964). Automatic design of optimal structures. *Journal de Mecanique*, 3:25–52.

Gill, P. E., Murray, W., and Saunders, M. A. (2005). Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131.

Glover, F. (1975). Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460.

Glover, F. (1984). An improved mip formulation for products of discrete and continuous variables. *Journal of Information and Optimization Sciences*, 5(1):69–71.

Gurobi Optimization, I. (2016). Gurobi optimizer reference manual.

Kenway, G. K. W., Kennedy, G. J., and Martins, J. R. R. A. (2014). Aerostructural optimization of the Common Research Model configuration. In *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Atlanta, GA. AIAA 2014-3274.

Mela, K. (2014). Resolving issues with member buckling in truss topology optimization using a mixed variable approach. *Structural and Multidisciplinary Optimization*, 50(6):1037–1049.

Petersen, C. C. (1971). A note on transforming the product of variables to linear form in linear programs. *Working Paper, Purdue University,.*

Rasmussen, M. and Stolpe, M. (2008). Global optimization of discrete truss topology design problems using a parallel cut-and-branch method. *Computers & Structures*, 86(13):1527 – 1538. Structural Optimization.

Stolpe, M. (2004). Global optimization of minimum weight truss topology problems with stress, displacement, and local buckling constraints using branch-and-bound. *International Journal for Numerical Methods in Engineering*, 61(8):1270–1309.

Stolpe, M. (2007). On the reformulation of topology optimization problems as linear or convex quadratic mixed 0–1programs. *Optimization and Engineering*, 8(2):163–192.

Stolpe, M. (2015). Truss optimization with discrete design variables: a critical review. *Structural and Multidisciplinary Optimization*, 3:1–26.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57.