This is a preprint of the following article, which is available at: http://mdolab.engin.umich.edu
M. Shahabsafa, A. Mohammad-Nezhad, T. Terlaky, L. Zuluaga, S. He, J. R. R. A. Martins, John. Hwang, A novel approach to discrete truss design problems using mixed integer neighborhood search. *Structural and Multidisciplinary Optimization*, (In press).
The original article may differ from this preprint and is available at:
https://link.springer.com/article/10.1007/s00158-018-2099-8.

A novel approach to discrete truss design problems using mixed integer neighborhood searchs

Mohammad Shahabsafa[1], Ali Mohammad-Nezhad[1], Tamás Terlaky[1], Luis Zuluaga[1], Sicheng He[2], Joaquim R. R. A. Martins[2], and John Hwang[3]

[1]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA
[2]University of Michigan, Department of Aerospace Engineering, Ann Arbor, MI 48109
[3]Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA 92093, USA

Abstract

Discrete truss sizing problems are very challenging to solve due to their combinatorial, nonlinear, non-convex nature. Consequently, truss sizing problems become unsolvable as the size of the truss grows. To address this issue, we consider various mathematical formulations for the truss design problem with the objective of minimizing weight, while the cross-sectional areas of the bars take only discrete values. Euler buckling constraints, Hooke's law, and bounds for stress and displacements are also considered. We propose mixed integer linear optimization (MILO) reformulations of the non-convex mixed integer models. The resulting MILO models are not solvable with existing MILO solvers as the size of the problem grows. Our novel methodology provides high quality solutions for large-scale real truss sizing problems, as demonstrated through extensive numerical experiments.

1 Introduction

Truss design problems focus on the optimal selection of geometry, topology and sizing of a truss structure (Bendsøe et al., 1994). Dorn et al. (1964) was the first to apply numerical optimization to the truss design problem. They used the ground structure approach, where 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 of minimizing 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 was 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 are 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, but not considering the kinematic compatibility and the stress-strain relation. He then developed a numerical approach to solve this model (Achtziger, 1999b).

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. The class of discrete truss design problems is motivated by manufacturing constraints in a practical setting. The areas of the truss elements are discrete because the bars are manufactured in fixed sizes. These restrictions also appear in other structural design problems, such as the design of shell element structures when dealing with laminated composites. 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 by 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 formulated and solved a MILO model taking into account the Euler buckling and kinematic stability constraints. 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. The MILO formulations of a truss structure can be generalized to other structures, e.g., Mellaert et al. (2017) develop MILO formulations for frame structures with various engineering constraints.

Several meta-heuristic methods have been used to solve truss design optimization problems. Genetic algorithms are one of the most common meta-heuristics historically used to solve these problems (Rajeev and Krishnamoorthy, 1992; Wu and Chow, 1995; Hajela and Lee, 1995; Kaveh and Kalatjari, 2004). Ant colony optimization (Bland, 2001; Camp and Bichon, 2004a; Kaveh et al., 2008) and particle swarm optimization methods (Li et al., 2009; Zeng and Li, 2012) have also been widely used to solve truss design problems.

Other methods including simulated annealing (Kripka, 2004), artificial bee colony optimization (Sonmez, 2011; Stolpe, 2011), mine blast algorithm (Sadollah et al., 2012, 2015), colliding bodies optimization (Kaveh and Mahdavi, 2014; Kaveh and Ghazaan, 2015)), and harmony search (SeokLee and Geem, 2004) have also been used. Stolpe (2016) provides 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. In the conclusions, he also stated the need for more publicly available benchmarking problems, which we address herein by contributing three new scalable problems.

Mladenović and Hansen (1997) developed the variable neighborhood search (VNS) algorithm to solve discrete optimization problems. In the VNS algorithm, the problem is solved iteratively over the neighborhood structures. At each iteration, local search methods are used to find the optimum in the neighborhood. Several general-purpose and problem-specific variants of the VNS algorithm have been developed (Hansen and Mladenović, 2003; Lazić, 2010; Hanafi, 2016). Svanberg and Werme (2005) used a neighborhood search method to solve the topology optimization problem. However, they consider a limited neighborhood, where "*two different designs are neighbors if they differ in only one single element*". Thus, the complexity of solving the associated subproblems is $\mathcal{O}(n)$, where n is the number of elements in the structure. In another article, Svanberg and Werme (2007) consider a M-neighborhood, where the number of the design variables that can simultaneously change is limited to M at each iteration ($M = 1, 2, 4$). The complexity of the associated subproblem on a M-neighborhood is $\mathcal{O}\left(n^M\right)$ for $M = 1, 2, 4$.

The subproblems of the neighborhood search MILO (NS-MILO) that we propose here are defined over exponentially large neighborhoods, which in turn decreases the

likelihood of getting stuck in a local optimum. As our experiments illustrate, the trade-off between solving more complex subproblems and the time needed to obtain near-optimal solutions for large-scale truss design problems is advantageous. Additionally, we do not solve the subproblems to optimality in the NS-MILO approach, but rather, we stop the solution process of the subproblems as soon as a solution better than the current best solution is found. This is one of the reasons that enables the NS-MILO approach to scale well as the size of the problem increases.

The minimum weight discrete truss design problem considering the Euler buckling constraints, Hooke's law, bounds for the stress and displacements has only been solved for small-scale problems. The main contributions of the work presented herein are the enhancement of the mathematical models for truss design problems, and an efficient solution methodology to approximately solve large-scale discrete truss design problems with more than 12,000 binary variables. We focus on applying our NS-MILO approach to truss sizing problems and do not consider truss topology problems.

The remainder of this 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, and in Section 4, we propose a novel solution methodology that enables us to generate high quality solutions to large-scale discrete truss design problems significantly faster. In Section 5, we demonstrate the efficiency of the proposed method through numerical results, followed by the conclusions 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 that induces the internal force that 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' stresses (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 nonlinear non-convex optimization problem (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, $I = \{1, \ldots, m\}$, and n be the number of degrees of freedom, which is

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

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

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

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 (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. Additionally, let $\boldsymbol{l} \in \mathbb{R}^m$ and $\boldsymbol{\Delta l} \in \mathbb{R}^m$ denote the length of the bars and elongation of the bars, respectively. The elongation of the bars depends on the displacement vector $\boldsymbol{u}$ as follows

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

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

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

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

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

where $\boldsymbol{K_i}$ is positive semidefinite. We may further assume that the truss structure is stable, and hence $\boldsymbol{K}(\boldsymbol{x})$ is positive definite, denoted as $\boldsymbol{K}(\boldsymbol{x}) \succ 0$. As mentioned below, this property is 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 obtained by assembling the bar stiffness matrices as follows

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

Lower and upper bounds are enforced for the bar stresses. Let $\boldsymbol{\sigma^{\min}}$, $\boldsymbol{\sigma^{\max}} \in \mathbb{R}^m$ be the given lower and upper bounds on the bar stress, respectively. The bounds $\boldsymbol{\sigma^{\min}}$ and $\boldsymbol{\sigma^{\max}}$ are actually bounds on the compression and tension of the bars, respectively. Therefore, we have $\boldsymbol{\sigma^{\max}} > \mathbf{0}$ and $\boldsymbol{\sigma^{\min}} < \mathbf{0}$. Lower and upper bounds $\boldsymbol{u^{\min}}$ and $\boldsymbol{u^{\max}}$ are considered on the nodal displacements of the structure as well. Moreover, we consider lower and upper bounds on the cross-sectional areas of the bars, namely, $\boldsymbol{x^{\min}}$, $\boldsymbol{x^{\max}} \in \mathbb{R}^m$. We consider the truss sizing problem, where $x_i^{\min} > 0$ 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 both ends of all the bars are pinned. Then, the Euler buckling stress for bar $i \in I$ with a circular cross section is

$$\sigma_i^E = -\frac{\pi^2 E_i}{4\left(\frac{l_i}{\tau_i}\right)^2}, \tag{7}$$

where τ_i is the radius of bar $i \in I$. If we define $\gamma_i = \pi E_i / 4l_i^2$, then $\sigma_i^E = -\gamma_i x_i$, and the Euler buckling constraints can be written as

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

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

$$q_i + \gamma_i x_i^2 \geq 0, \qquad i \in I. \tag{9}$$

Constraint (8) can be generalized to the cases where the discrete choice of the bar sizes corresponds to similar cross-sectional shapes.

For example, suppose we have a discrete choice set composed of similar rectangles: $h_i/b_i = c$, for $i \in I$, where b_i and h_i denote the width and height for the ith choice, respectively, and c is a constant. Then the Euler buckling constraints are written as

$$\sigma_i + \gamma_i' x_i \geq 0, \quad \gamma_i' = \frac{c\pi^2 E}{12 l_i^2}, \qquad i \in I. \tag{10}$$

Notice that neither Hooke's law nor the Euler buckling constraints (9) are convex. This results in the following non-convex quadratic optimization formulation of the truss design problem:

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

where ρ is the density of the bar material. We assume without loss of generality that all bars have the same density.

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—the first constraint in (P_1)—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

$$
\begin{array}{lll}
\min & \rho \boldsymbol{l}^T \boldsymbol{x} & \\
\text{s.t.} & \boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{f}, & \\
& \dfrac{E_i}{l_i}(\boldsymbol{r_i}^T \boldsymbol{u}) = \sigma_i, & i \in I, \\
& \sigma_i + \gamma_i x_i \geq 0, & i \in I, \\
& \boldsymbol{u^{\min}} \leq \boldsymbol{u} \leq \boldsymbol{u^{\max}}, & \\
& \sigma_i^{\min} \leq \sigma_i \leq \sigma_i^{\max}, & i \in I, \\
& x_i^{\min} \leq x_i \leq x_i^{\max}, & i \in I,
\end{array}
\tag{P_2}
$$

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

$$
\begin{array}{lll}
\min & \rho \boldsymbol{l}^T \boldsymbol{x} & \\
\text{s.t.} & \boldsymbol{R}\boldsymbol{q} = \boldsymbol{f}, & \\
& \sigma_i - \dfrac{E_i}{l_i}\boldsymbol{r_i}^T \boldsymbol{u} = 0, & i \in I, \\
& q_i - \sigma_i x_i = 0, & i \in I, \\
& \sigma_i + \gamma_i x_i \geq 0, & i \in I, \\
& \boldsymbol{u^{\min}} \leq \boldsymbol{u} \leq \boldsymbol{u^{\max}}, & \\
& \sigma_i^{\min} \leq \sigma_i \leq \sigma_i^{\max} & i \in I, \\
& x_i^{\min} \leq x_i \leq x_i^{\max} & i \in I.
\end{array}
\tag{P_3}
$$

This model has m non-convex equalities, each of which has only one bilinear term. As a result, this is simpler than models (P_1) and (P_2) (e.g., Bendsoe and Sigmund (2013) compare models with different decision variables). 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

In problems (P_1), (P_2), and (P_3), we assume 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 standard pre-manufactured bars (Achtziger and Stolpe, 2006, 2007a; Cerveira et al., 2009). Thus, x_i for $i \in I$ takes values from the finite set

$$S_i = \{s_{i1}, s_{i2}, \ldots, s_{ip_i}\}, \tag{11}$$

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 (11), which are referred to as the *basic discrete model* and the *incremental discrete model*.

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

$$\begin{aligned} x_i &= \sum_{k=1}^{p_i} s_{ik} z_{ik}, && i \in I, \\ \sum_{k=1}^{p_i} z_{ik} &= 1, && i \in I,\ k \in P_i, \\ z_{ik} &\in \{0,1\}, && i \in I,\ k \in P_i. \end{aligned} \tag{12}$$

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

$$\begin{aligned} x_i &= s_{i1} + \sum_{k=1}^{p_i-1} \delta_{ik} z_{ik}, && i \in I, \\ z_{ik} &\geq z_{i,k+1}, && i \in I,\ k = 1, \ldots, p_i - 2, \\ z_{ik} &\in \{0,1\} && i \in I,\ k = 1, \ldots, p_i - 1. \end{aligned} \tag{13}$$

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 $x_i \geq s_{ik}$, 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 (12) and (13). This idea can be used to reformulate problems (P_1) and (P_2). We start by substituting the basic discrete model (12) in the constraint $q_i - \sigma_i x_i = 0$ for all $i \in I$. Then we have

$$q_i - \sum_{k=1}^{p_i} s_{ik} z_{ik} \sigma_i = 0,\ i \in I, \tag{14}$$

where 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 by Glover (1975, 1984), we can linearize the constraint (14) by introducing auxiliary variables $\psi_{ik} = \sigma_i z_{ik}$ and by adding the following constraints:

$$\begin{aligned} z_{ik}\sigma_i^{\min} &\leq \psi_{ik} \leq z_{ik}\sigma_i^{\max}, \\ \sigma_i - \sigma_i^{\max}(1 - z_{ik}) &\leq \psi_{ik} \leq \sigma_i - \sigma_i^{\min}(1 - z_{ik}), \end{aligned}$$

for all $i \in I, k = 1, \ldots, p_i$. Then, using the basic discrete model (12), problem (P_3) can be reformulated with 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 $\boldsymbol{\psi} \in \mathbb{R}^{\sum_i p_i}$, to obtain the following MILO problem:

$$
\begin{aligned}
\min \quad & \sum_{i\in I} \rho l_i x_i \\
\text{s.t.} \quad & \boldsymbol{Rq} && = \boldsymbol{f}, && \\
& x_i - \sum_{k=1}^{p_i} s_{ik} z_{ik} && = 0, && i \in I \\
& \sigma_i - \frac{E_i}{l_i} \boldsymbol{r_i}^T \boldsymbol{u} && = 0, && i \in I, \\
& q_i - \sum_{k=1}^{p_i} s_{ik} \psi_{ik} && = 0, && i \in I, \\
& \sigma_i + \gamma_i x_i && \geq 0, && i \in I, \\
& \sigma_i^{\min} && \leq \sigma_i \quad \leq \sigma_i^{\max}, && i \in I, \\
& \boldsymbol{u^{\min}} && \leq \boldsymbol{u} \quad \leq \boldsymbol{u^{\max}}, && \\
& z_{ik}\sigma_i^{\min} && \leq \psi_{ik} \quad \leq z_{ik}\sigma_i^{\max}, && i \in I,\ k \in P_i, \\
& \sigma_i - \sigma_i^{\max}(1 - z_{ik}) && \leq \psi_{ik} \quad \leq \sigma_i - \sigma_i^{\min}(1 - z_{ik}), && i \in I,\ k \in P_i, \\
& \sum_{k=1}^{p_i} z_{ik} && = 1, && i \in I, \\
& z_{ik} && \in \{0, 1\}, && i \in I,\ k \in P_i.
\end{aligned}
\tag{15}
$$

In a similar manner, we can consider the incremental discrete model (13) and reformulate problem (P_3) as the following MILO model:

$$
\begin{aligned}
\min \quad & \sum_{i\in I} \rho l_i x_i \\
\text{s.t.} \quad & \boldsymbol{Rq} && = \boldsymbol{f}, && \\
& x_i - s_{i1} - \sum_{k=1}^{p_i-1} \delta_{ik} z_{ik} && = 0, && i \in I, \\
& \sigma_i - \frac{E_i}{l_i} \boldsymbol{r_i}^T \boldsymbol{u} && = 0, && i \in I, \\
& q_i - s_{i1}\sigma_i - \sum_{k=1}^{p_i-1} \delta_{ik} \psi_{ik} && = 0, && i \in I, \\
& \sigma_i + \gamma_i x_i && \geq 0, && i \in I, \\
& \sigma_i^{\min} && \leq \sigma_i \quad \leq \sigma_i^{\max}, && i \in I, \\
& \boldsymbol{u^{\min}} && \leq \boldsymbol{u} \quad \leq \boldsymbol{u^{\max}}, && \\
& z_{ik}\sigma_i^{\min} && \leq \psi_{ik} \quad \leq z_{ik}\sigma_i^{\max}, && i \in I,\ k \in \overline{P}_i, \\
& \sigma_i - \sigma_i^{\max}(1 - z_{ik}) && \leq \psi_{ik} \quad \leq \sigma_i - \sigma_i^{\min}(1 - z_{ik}), && i \in I,\ k \in \overline{P}_i, \\
& z_{ik} && \geq z_{i,k+1}, && i \in I,\ k \in \overline{\overline{P}}_i, \\
& z_{ik} && \in \{0, 1\}, && i \in I,\ k \in \overline{P}_i,
\end{aligned}
\tag{16}
$$

where $\overline{P}_i = \{1, 2, \ldots, p_i - 1\}$ and $\overline{\overline{P}}_i = \{1, 2, \ldots, p_i - 2\}$. Note 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 MILO formulations for models (15) and (16) in terms of solution time. In particular, these cuts can replace the Euler buckling constraints (8). Next, we derive the cuts for

model (16). The derivation of the cuts for model (15) can be done in a similar fashion.

Considering the discrete model (13), the Euler buckling constraints can be rewritten as

$$\sigma_i + \gamma_i \left(s_{i1} + \sum_{k=1}^{p_i - 1} \delta_{ik} z_{ik} \right) \geq 0, \ \ i \in I. \tag{17}$$

For each bar $i \in I$, the Euler buckling constraint enforces an inequality on z_{ik} for $k = 1, \ldots, p_{i-1}$. We replace the Euler buckling constraint (17) 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 (17) for incremental discrete model (13).

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

Proof. Suppose $x_i = s_{ik}$ for $k = 1 \ldots, p_i - 1$ and $i = 1, \ldots, m$. Then the Euler buckling constraint (17) reduces to

$$\sigma_i + \gamma_i s_{ik} \geq 0. \tag{19}$$

From the incremental discrete model (13) 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 (18) is equivalent to

$$\begin{array}{lll} \sigma_i + \gamma_i s_{ij} & \geq 0, & j = k, \ldots, p_i - 1, \\ \sigma_i - \sigma_i^{\min} & \geq 0. & \end{array} \tag{20}$$

By the inequalities $\sigma_i - \sigma_i^{\min} \geq 0$, the constraints above can be written as

$$\sigma_i \geq \max_{j = k, \ldots, p_i - 2} \{ -\gamma_i s_{ij} \}. \tag{21}$$

This constraint can be written as $\sigma_i \geq -\gamma_i s_{ik}$, which is equivalent to the Euler buckling constraint (19). 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 (18) 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 (17). □

□

In Section 5.3, we demonstrate that replacing the Euler buckling constraints with constraints (18) in truss instances with large discrete sets can, in most cases, decrease the solution time by more than 20% on average.

The reason to replace the Euler buckling constraint by p_i constraints is that it helps to decompose the Euler buckling constraints for the binary variables z_{ik}, $k = 1 \ldots, p_i - 1$. As it can be seen from constraint (17), the original Euler buckling constraint is written in terms of all the $p_i - 1$ binary variables corresponding to bar i, while in the reformulation (18), each constraint is in terms of only one binary variable. That is the reason that the reformulation would be more effective if p_i is large.

4 Solution methodologies

We now present a methodology for solving discrete truss design problems (15) and (16). For 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 (Stolpe, 2016). In Table 1, results for the optimization of 3D cantilever trusses (see Section 5.1.4) using GuRoBi 7.0.2 are 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: Solutions of 3D-truss sizing instances obtained directly using GuRoBi.

# bars	# bin. var.	Time(s)	Weight (kg)	Opt. Gap
20	800	247.55	9.75	0.07%
40	1600	86400.02	21.24	25.76%
60	2400	86400.03	24.82	30.03%
80	3200	86400.04	31.22	32.43%
100	4000	86400.05	32.04	34.36%

To address the numerical performance issues highlighted in Table 1, we present a new solution methodology, referred as neighborhood search MILO (NS-MILO). The NS-MILO approach solves subproblems which are defined over the feasible set of the original problem.

In existing neighborhood search algorithms used to solve the truss design problems (see e.g., Mladenović and Hansen (1997); Hansen and Mladenović (2003); Svanberg and Werme (2005, 2007); Lazić (2010); Hanafi (2016)), the subproblems are defined on a small neighborhood of the incumbent solution so that those subproblems are polynomially solvable. This, in turn, may result in small local improvements, since the subproblems explore a very small neighborhood for a better solution. However, the neighborhoods of the subproblems in the NS-MILO approach are significantly larger, such that the number of the feasible solutions of the subproblems grows exponentially as the number of the bars increases. Therefore, the subproblems become NP-hard, i.e., the time needed to solve them to global optimality grows exponentially as the size of the problem grows. This approach enables us to explore a significantly larger neighborhood for a better solution, which decreases the likelihood of getting stuck in a local optimum.

In existing neighborhood search algorithms (see e.g., Mladenović and Hansen (1997); Hansen and Mladenović (2003); Svanberg and Werme (2005, 2007); Lazić (2010); Hanafi (2016)), the simple subproblems are solved to optimality, while in the NS-MILO approach, we do not solve the NP-hard subproblems to global optimality, since proving optimality of the subproblems does not necessarily help us in solving the original problem. Instead, we stop solving the subproblems as soon as a better solution than the current best solution is found, and define the next subproblem in the neighborhood

of the improved solution. Nowadays, MILO solvers, such as GuRoBi(2016), have extremely powerful methods to improve the best integer feasible solution found, which helps to significantly reduce the time needed to improve the integer feasible solution of the subproblems of the NS-MILO approach.

The fact that the NS-MILO approach explores significantly larger neighborhoods and does not prove global optimality for the subproblems makes it possible to provide high-quality solutions to large-scale truss design problems.

If we continued to increase the size of the neighborhood of the MILO subproblems in the NS-MILO approach, and solve the last MILO subproblem—which would be identical to the original problem—to optimality, the NS-MILO approach would provide an optimal solution for the truss design problem.

The NS-MILO 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 a given feasible 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 assignment of the bar cross-sectional areas $\boldsymbol{x} \in \mathbb{R}^n$. Note that the complexity of solving a MILO_k subproblem is $\mathcal{O}(m^k)$. 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:

1. $\text{MILO}_2(\boldsymbol{x})$, for $\boldsymbol{x} \in \mathbb{R}^n$, 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

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

2. $\text{MILO}_3(\boldsymbol{x})$ 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

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

3. $\text{MILO}_5(\boldsymbol{x})$ 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 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 by the finite set (11). We start the NS-MILO approach by obtaining a high-quality feasible solution for the continuous model (P_3) using a nonlinear optimization (NLO) solver. The solution provided by the NLO solver is a KKT point of model (P_3) and is denoted by $\boldsymbol{x^0}$. Then we solve a sequence of MILO$_2$ subproblems. We solve MILO$_2(\alpha \boldsymbol{x^0})$ with $\alpha = 1$ using a MILO solver considering a predefined limit on the solution time. If an integer feasible solution is not found within the time limit, we increase α by 0.05, and again solve MILO$_2(\alpha \boldsymbol{x^0})$. We continue solving MILO$_2(\alpha \boldsymbol{x^0})$ until an integer feasible solution is found.

Next, we generate MILO$_3(\hat{\boldsymbol{x}})$, where $\hat{\boldsymbol{x}}$ is the best integer feasible solution obtained from MILO$_2(\alpha \boldsymbol{x^0})$. Using a MILO solver, we run the MILO solver on 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 running the MILO solver on MILO$_3(\hat{\boldsymbol{x}})$ until the objective function does not improve.

Afterwards, we generate MILO$_5(\hat{\boldsymbol{x}})$, and similarly run the MILO solver on 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 running the MILO solvers on the actual MILO$_5(\hat{\boldsymbol{x}})$ until the objective function does not improve.

Note 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. Note that 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. In this algorithm, 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. In $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ Solve(P) and $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(P), $\hat{\boldsymbol{x}}$ is the solution that is returned and $\hat{\eta}$ is the weight of the solution $\hat{\boldsymbol{x}}$.

Algorithm 1 The NS-MILO approach

$\boldsymbol{x^0}$:= a solution of the continuous model
$\alpha := 1$
repeat
    $\overline{\boldsymbol{x}} := \alpha \boldsymbol{x^0}$
    $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ Solve(MILO$_2(\overline{\boldsymbol{x}})$)
    $\alpha := \alpha + 0.05$
until MILO$_2(\overline{\boldsymbol{x}})$ is feasible
repeat
    $\eta_{\text{curr}} := \hat{\eta}$
    $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(MILO$_3(\hat{\boldsymbol{x}})$)
until $\hat{\eta} = \eta_{\text{curr}}$
repeat
    $\eta_{\text{curr}} := \hat{\eta}$
    $(\hat{\boldsymbol{x}}, \hat{\eta}) :=$ FindSol(MILO$_5(\hat{\boldsymbol{x}})$)
until $\hat{\eta} = \eta_{\text{curr}}$
return $\hat{\boldsymbol{x}}$

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, we solve two well-known instances from the literature: the 10-bar truss and the 72-bar truss problems (Haftka and Gürdal, 2012) to compare the NS-MILO approach with other algorithms used in the literature to solve these instances. In addition, we introduce three different scalable truss design problems: 2D cantilever truss, 3D cantilever truss, and truss models of an airplane wing with the goal to demonstrate how the NS-MILO approach scales as the size of the problem grows. These three new problems are available online [1].

The numerical results are run on a machine with Dual Intel Xeon® CPU E5-2630 @ 2.20 GHz (20 cores) and 64 GB of RAM. We use the NLO solver IPOPT 3.14 (Wächter and Biegler, 2006) to obtain a KKT point of the continuous truss sizing problem. Additionally, GuRoBi 7.0.2 (2016) is used to solve the MILO models. GuRoBi has the capability of using multiple threads in solving a MILO problem with the branch and bound algorithm. The optimality gap threshold 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. We set GuRoBi to use 16 threads when attempting to solve the problems and subproblems in this section.

[1] https://github.com/shahabsafa/truss-data.git

5.1 Truss Problems

Here we solve two classical truss problems and the three scalable truss sizing problems that are used to evaluate the models and the solution methodology.

5.1.1 The 10-bar truss

The 10-bar problem (Haftka and Gürdal, 2012) shown in Figure 1 is frequently used as a benchmark example. The external force P on nodes 2 and 4 is equal to 444,800 N (10^5 lb), and the material properties are listed in Table 2. Additionally, the displacement bound on the y-direction of the nodes 1 and 2 is ± 2.0 in. The discrete set of potential cross-sectional areas is listed in the appendix.

Table 2: Aluminum alloy material properties used for 10-bar and 72-bar problems.

Property	Value
ρ	$0.1\,\mathrm{lbm/in}^3$
E	10^7 psi
σ_Y	25000 psi

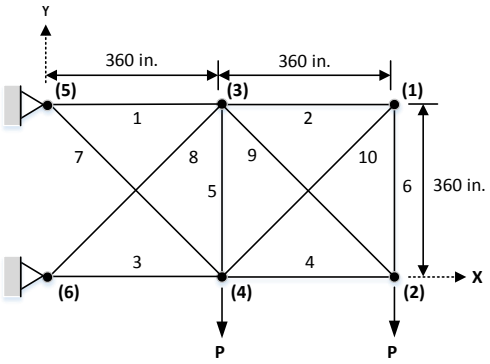


Figure 1: The 10-bar truss.

5.1.2 The 72-bar truss

The 72-bar truss problem (Haftka and Gürdal, 2012) shown in Figure 2 is another common benchmark. The bar material properties are listed in Table 2. Additionally, the displacement bound on the x and y direction of the nodes 1, 2, 3, and 4 is ± 0.25 in. We have two load cases. In the load case one, the external force is only exerted on node 1, with value $f_x = 5000$ lbf, $f_y = 5000$ lbf, and $f_z = -5000$ lbf. In the load case two, the external force is exerted on the z direction of the nodes 1, 2, 3, and 4 with value $f_z = -5000$ lbf. The discrete set S is defined in the appendix

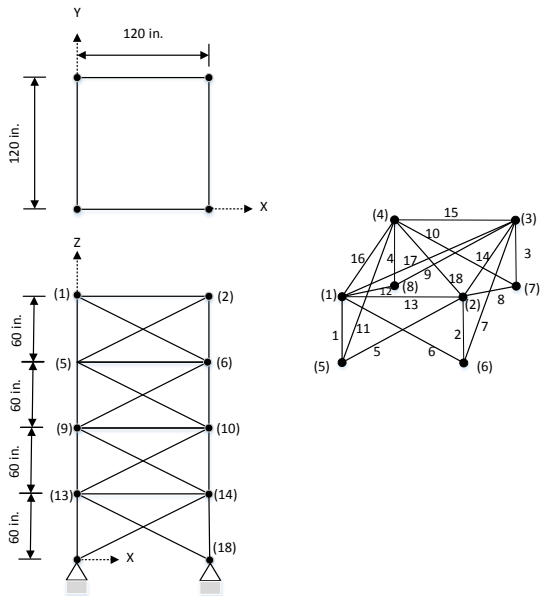


Figure 2: The 72-bar truss.

5.1.3 Scalable 2D cantilever problem

The 2D cantilever problem is made scalable by varying the number of blocks, where each block has five bars. A 2D cantilever instance with 3 blocks is illustrated in Figure 3.

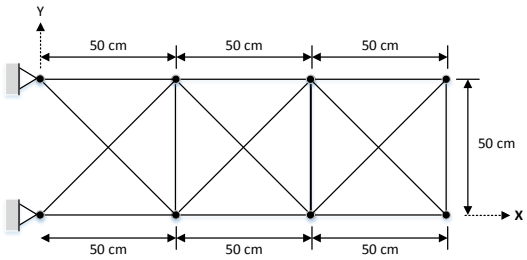


Figure 3: The 2D cantilever problem instance with 3 blocks.

The material properties are listed in Table 3. The yield stress corresponds to the yield strength of an aluminum alloy with a 50% safety margin. The external force is generated randomly at each node from a given interval using a uniform distribution. Let $f_0 = 1.25\times10^5/n_b$ N, where n_b is the number of the blocks of the cantilever problem. For all the top nodes, the y coordinate of the force randomly takes values in the interval $[-f_0, 0]$, and the x coordinate of the force takes value in the interval $[-f_0/10, f_0/10]$. For the top nodes, the y and x force 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. The bars can take 41 different cross-sectional areas, which are listed in the appendix.

Table 3: Aluminum alloy material properties used for the 2D and 3D cantilever problems.

Property	Value
ρ	$2.7\,\mathrm{kg/m^3}$
E	$69\,\mathrm{GPa}$
σ_Y	$172.36\,\mathrm{MPa}$

The displacement bounds are considered for the two nodes of the tip of the cantilever problem in the x and y directions, and are presented in Table 4. Note that the displacement bounds varies with the number of the blocks n_b.

Table 4: Displacement bounds for the 2D cantilever problem.

n_b	Bounds (cm)	n_b	Bounds (cm)	n_b	Bounds (cm)
1	0.1	8	6.4	36	129.6
2	0.4	12	14.4	40	160.0
3	0.9	16	25.6	44	193.6
4	1.6	20	40.0	48	230.4
5	2.5	24	57.6	52	270.4
6	3.6	28	78.4	56	313.6
7	4.9	32	102.4	60	360.0

5.1.4 The 3D cantilever truss problem

The 3D cantilever scalable problem is an extension of the 2D cantilever problem. Now each block is a cube, and all the diagonals of each face are has five bars. Additionally, two of the main diagonals of each cube are connected with bars. This adds up to 20 bars per block. The 3D cantilever instance with 3 blocks is illustrated in Figure 4.

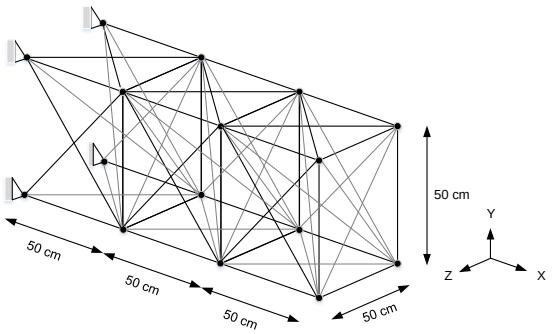


Figure 4: 3D cantilever instance with 3 blocks.

The material properties are listed in Table 3. Similar to the 2D cantilever problem, the force is generated randomly at each node from a given interval using a uniform

distribution. Let $f_0 = 1.2 \times 10^6/\ell$ N, 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 force coordinates randomly take values in the interval $[-f_0/10, f_0/10]$. For the bottom nodes, the y, x, and z force 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, the bars can take 41 different cross-sectional areas in the range $[0.25 - 85]$ cm^2, which are listed in the appendix.

The displacement bounds are considered for the four nodes at the tip of the cantilever instance on the x, y, and z direction, and is presented in Table 5. Similar to the 2D cantilever instances, the displacement bounds varies with the number of the blocks n_b.

Table 5: Displacement bounds of the 3D cantilevers.

n_b	Bounds (cm)	n_b	Bounds (cm)	n_b	Bounds (cm)
1	0.3	6	3.6	11	12.1
2	0.4	7	4.9	12	14.4
3	0.9	8	6.4	13	16.9
4	1.6	9	8.1	14	19.6
5	2.5	10	10.0	15	22.5

5.1.5 Wing truss problem

We now consider a 3D wing modeled with bars. The truss layout is generated based on the undeformed common research model (uCRM) geometry (Brooks et al., 2018), and is shown in Figure 5. For the aerodynamic load, we assume an elliptical distribution in the spanwise direction and a uniform distribution in the chordwise direction. The total load of one wing is set to be one half of the maximum takeoff weight of the uCRM. For simplicity, we also assume that the aerodynamic load is not affect by the structural deformation, i.e., the aeroelastic effect is neglected in this study. The bars can take 40 different cross-sectional areas in the range $[0.25 - 1200]$ cm^2, which are listed in the appendix.

The material properties are listed in Table 6. While it would be possible to consider bounds on the displacements, we do not do that here, because in practical aircraft and wing design, stress and buckling constraints are sufficient to achieve feasible designs from the structural point of view.

Table 6: Aluminum alloy material properties for the wing problem.

Property	Value
ρ	$2.7\,\mathrm{kg/m^3}$
E	$69\,\mathrm{GPa}$
σ_Y	$270\,\mathrm{MPa}$

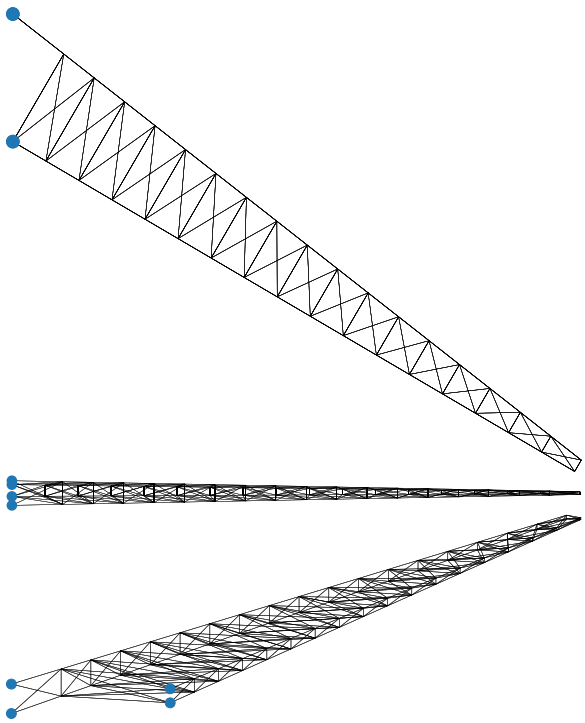


Figure 5: Wing truss problem instance with 315 bars.

5.2 Basic versus 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 6, the objective function improvement of the basic and incremental models is plotted for the 2D cantilever instance with 5 blocks. As we can see, the incremental model stops at $t = 64.11$ seconds, while the basic model stops at $t = 256.64$ s. Additionally, the incremental model finds the optimal solution at $t = 31$ s, while the basic model finds the optimal solution about 7 times slower at $t = 221$ s. Therefore, the incremental model is faster in proving optimality, and it finds the optimal solution significantly faster than the basic model.

In Table 7, the 2D and 3D cantilever trusses that are solved to global optimality in 24 hours with either the basic or incremental model are reported. As the incremental model is faster than the basic model, we use the incremental model through the rest of the paper.

The intuition behind the better performance of the incremental discrete model is

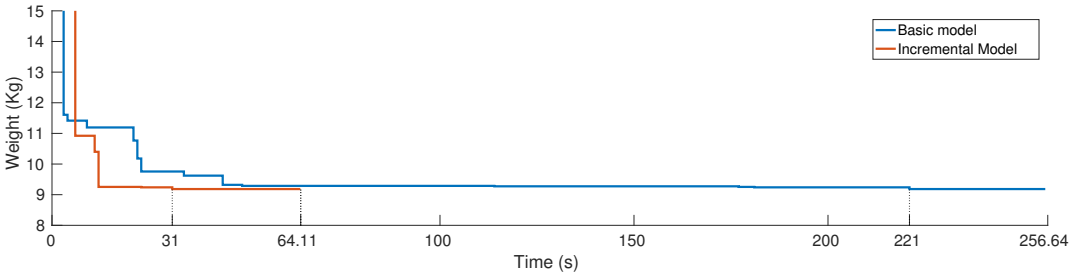


Figure 6: Comparison of the basic model and the incremental model for the 2D cantilever problem instance with 5 blocks.

that 1) In the incremental discrete model, fixing one of the binary variables to either zero or one, automatically fixes the value of a high number of binary variables thanks to inequality constraints (13); 2) This same set of inequality constraints (13) involving the binary variables can be effectively used by MILO solvers to generate extra cuts that help in the solution of the MILO model.

Table 7: Solution times (s) and weights (kg) for the basic and incremental models.

	n_b	m	Basic model			Incremental model		
			bin. var	weight	time	bin var	weight	time
2D	1	5	205	2.55	0.93	200	2.55	0.89
	2	10	410	5.68	27.92	400	5.68	4.33
	3	15	615	6.87	240.42	600	6.87	24.17
	4	20	820	9.64	3883.97	800	9.64	174.54
	5	25	1025	9.18	256.64	1000	9.18	64.11
	6	30	1230	13.76	86400.00	1200	13.76	9891.51
3D	1	20	820	9.75	37275.40	800	9.75	247.55

5.3 Strengthening the Euler buckling constraints

Next, we numerically examine the effect of replacing the Euler buckling constraint introduced in (8) with constraints (18) introduced in Theorem 1.

Table 8: Impact of introducing the buckling constraints (18) on the solution time (s).

	n_b	m	Euler Const. (8)		Constr. (18)		$\frac{t_2}{t_1}$
			weight (kg)	t_1	weight (kg)	t_2	
2D	1	5	2.55	0.89	2.55	0.56	0.63
	2	10	5.68	4.33	5.68	3.16	0.73
	3	15	6.87	24.17	6.87	21.36	0.88
	4	20	9.64	174.54	9.64	190.51	1.09
	5	25	9.18	64.11	9.18	110.07	1.71
	6	30	13.76	9891.51	13.76	8891.16	0.90
3D	1	20	9.75	247.55	9.75	201.47	0.81

The impact of replacing the Euler buckling constraints (8) with constraints (18) is demonstrated in Table 8, where t_1, and t_2 denote the solution time of the model with constraints (8), and the solution time of the model with constraints (18), respectively. As we can see in Table 8, replacing the Euler buckling constraints (8) with constraints (18), reduces the solution time in 5 out of 7 instances. Specifically, for the largest 2D cantilever truss and the 3D cantilever truss with 20 bars, the solution time decreases by 10% and 19% respectively.

Replacing the Euler buckling constraints (8) with constraints (18) 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 directly with 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, 1500, and 2500 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 6000 seconds respectively. That means that our experiments compare the quality of the solution of full-MILO approach after 24 hrs (if an optimal solution was not obtained earlier) with the NS-MILO approach solution with the time settings mentioned above. To evaluate the NS-MILO approach, we solve the well-known 10-bar and 72-bar instances (Haftka and Gürdal, 2012) and compare the solution of the NS-MILO with that of other approaches in Tables 9, 10, 11, and 12. Additionally, we solve the 2D and 3D cantilever instances with 20 to 300 bars, and the wing instances with 81 to 315 bars to see how NS-MILO scales as the size of the problem grows. Results of the 2D and 3D cantilever instances are presented in Tables 13 and 14 respectively, and results of the wing instances are presented in Table 15. In Tables 13, 14, and 15, 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 13 and 14 n_b is the number of blocks of the cantilever instances, and in Table 15, w_c, and t_c denote the weight of the solution of the continuous design problem, and the time to obtain that solution, respectively.

5.4.1 10-bar truss

As we can see in Table 9, the solution of the continuous model matches that of Haftka and Gürdal (2012), and the solution of the discrete model is identical to that of Cai and Thierauf (1993), Mahfouz (1999), Camp and Bichon (2004b), Camp (2007), Barbosa et al. (2008), Sonmez (2011), and Camp and Farshchin (2014). The full-MILO approach has solved the problem to global optimality, thus the solution of the full-MILO approach

is the global optimal solution. As the solution of the NS-MILO is equal to that of the full-MILO, we can conclude that the solution of the NS-MILO is the global optimal solution of the problem for the 10-bar instance without the Euler buckling constraints.

Table 9: Cross-sectional areas (in^2) and the weights (lbm) for the 10-bar truss problem solutions without Euler buckling constraints.

Vars.	Continuous		Discrete	
	Haftka and Gürdal (2012)	Model (P_3)	Full-MILO	NS-MILO
1	30.52	30.52	33.50	33.50
2	0.10	0.10	1.62	1.62
3	23.20	23.20	22.90	22.90
4	15.22	15.22	14.20	14.20
5	0.10	0.10	1.62	1.62
6	0.55	0.55	1.62	1.62
7	7.46	7.46	7.97	7.97
8	21.04	21.04	22.90	22.90
9	21.53	21.53	22.00	22.00
10	0.10	0.10	1.62	1.62
W	5060.85	5060.60	5490.74	5490.74

Table 10: Cross-sectional areas (in^2) and the weight (lbm) of the solution of the 10-bar truss with Euler buckling constraints.

Vars.	Continuous		Discrete	
	Petrovic et al. (2017)	Model (P_3)	Full-MILO	NS-MILO
1	11.56	17.68	18.80	18.80
2	8.17	0.10	1.62	1.62
3	65.90	57.09	52.50	52.50
4	24.38	40.62	42.50	42.50
5	0.11	0.10	1.62	1.62
6	9.46	0.10	1.80	1.62
7	26.27	7.42	4.80	4.97
8	41.50	69.16	80.00	80.00
9	4.31	12.55	14.20	14.20
10	54.64	0.10	1.62	1.62
W	10492.80	8707.56	9400.97	9403.15

In Table 10 the solution of the continuous and discrete truss sizing problem with Euler buckling constraints is presented for the 10-bar instance. Petrovic et al. (2017)

stated that "*there is no research found which gives buckling constrained results for 10 bar trusses* ". However, they considered only the continuous truss design problem with Euler buckling constraints for the 10-bar instance. Although there are some articles that consider buckling constraints (Wu and Chow, 1995; Rajeev and Krishnamoorthy, 1997; Rahami et al., 2008; Ho-Huu et al., 2016), they do not consider the buckling constraints in the form (8) or (9). In fact, the solutions provided by these authors do not satisfy the buckling constraints in the form (8) or (9). The full-MILO approach finds the global optimal solution in this small instance. Thus, to check the NS-MILO approach, it is enough to compare the solution of the NS-MILO with that of the full-MILO approach. As we can see in Table 10, the weight of the solution of the NS-MILO approach is within 0.1% of that of the global optimal solution obtained from the full-MILO approach.

5.4.2 72-bar truss

A solution to the 72-bar instance that satisfies Euler buckling constraints has not been reported in the literature. Thus to benchmark the NS-MILO approach for this instance, we show in Table 11 results of the 72-bar instance without Euler buckling constraints. As we can see, the weight of the solution of the continuous model (P_3) matches that of the solution by Haftka and Gürdal (2012) with 0.1% precision. Additionally, the solution of the discrete model using the NS-MILO approach has the same weight as that by Kaveh and Ghazaan (2015), Sadollah et al. (2015), and Ho-Huu et al. (2016). Note that we let the full-MILO approach run for 120 hrs for the 72-bar instance without Euler buckling constraints, and the optimality gap is still 32.61% when the full-MILO approach stops. Furthermore, the solution of the 72-bar instance with Euler buckling constraints is presented in Table 12.

Table 11: Cross-sectional areas (in^2) and the weights (lbm) for the 72-bar truss problem solutions without Euler buckling constraints.

Vars.	Continuous		Discrete							
	Haftka and Gürdal (2012)	Model	Wu and Chow (1995)	Kaveh and Talatahari (2009)	Sadollah et al. (2012)	Kaveh and Ghazaan (2015)	Sadollah et al. (2015)	Ho-Huu et al. (2016)	Full-MILO	NS-MIL
1-4	0.157	0.156	0.196	0.196	1.800	0.196	0.196	0.196	0.196	0.1
5-12	0.536	0.546	0.602	0.563	0.602	0.563	0.563	0.563	0.563	0.5
13-16	0.410	0.410	0.307	0.442	0.111	0.391	0.391	0.391	0.442	0.3
17-18	0.569	0.570	0.766	0.563	0.111	0.563	0.563	0.563	0.602	0.5
19-22	0.507	0.524	0.391	0.563	1.266	0.563	0.563	0.563	0.785	0.5
23-30	0.520	0.517	0.391	0.563	0.563	0.563	0.563	0.563	0.563	0.5
31-34	0.100	0.100	0.141	0.111	0.111	0.111	0.111	0.111	0.111	0.1
35-36	0.100	0.100	0.111	0.250	0.111	0.111	0.111	0.111	0.111	0.1
37-40	1.280	1.268	1.800	1.228	0.442	1.228	1.228	1.228	1.000	1.2
41-48	0.515	0.512	0.602	0.563	0.442	0.442	0.442	0.563	0.563	0.5
49-52	0.100	0.100	0.141	0.111	0.111	0.111	0.111	0.111	0.111	0.1
53-54	0.100	0.100	0.307	0.111	0.111	0.111	0.111	0.111	0.111	0.1
55-58	1.897	1.886	1.563	1.800	0.196	1.990	1.990	1.990	1.990	1.9
59-66	0.516	0.512	0.766	0.442	0.563	0.563	0.563	0.442	0.442	0.4
67-70	0.100	0.100	0.141	0.141	0.442	0.111	0.111	0.111	0.111	0.1
71-72	0.100	0.100	0.111	0.111	0.602	0.111	0.111	0.111	0.111	0.1
W (lbm)	379.66	379.61	427.20	393.38	390.73	389.33	389.33	389.33	392.96	389.

Table 12: Cross-sectional areas (in^2) and the weights (lbm) for the 72-bar truss problem solutions with Euler buckling constraints.

Vars.	Continuous	Discrete	
		Full-MILO	NS-MILO
1–4	1.470	1.457	1.457
5–12	2.283	2.380	2.380
13–16	1.649	1.990	1.620
17–18	2.774	2.630	2.880
19–22	1.498	1.563	1.563
23–30	1.776	1.800	1.800
31–34	0.100	0.196	0.196
35–36	0.330	0.785	0.442
37–40	1.518	1.563	1.620
41–48	1.933	1.990	1.990

Table 13: Weights (kg) and the solution times (s) for the 2D cantilever problem instances using the NS-MILO approach.

n_b	m	Full-MILO			NS-MILO			t_f/t_n
		w_f	t_f	gap(%)	n_s	w_n	t_n	
4	20	9.64	155.44	0.00	7	9.64	0.56	277.57
8	40	21.42	86400.00	3.13	5	21.36	8.83	9784.82
12	60	33.52	86400.16	6.54	8	33.53	541.09	159.67
16	80	49.51	86400.21	10.25	6	49.39	287.82	159.68
20	100	68.85	86400.28	12.72	11	68.88	2390.69	36.14
24	120	80.93	86400.15	13.20	10	80.92	2543.93	33.96
28	140	110.78	86400.14	13.49	13	110.83	3035.35	28.46
32	160	152.81	86400.07	13.64	14	152.78	5287.82	16.34
36	180	179.96	86400.14	13.98	15	179.81	2712.94	31.85
40	200	219.12	86400.17	13.31	8	219.25	1691.19	51.09
44	220	274.88	86400.17	13.54	11	275.51	4273.62	20.22
48	240	324.42	86400.19	13.56	18	324.88	5027.54	17.18
52	260	382.51	86400.14	12.97	15	382.77	6414.20	13.47
56	280	437.11	86400.22	12.95	15	437.20	5741.47	15.05
60	300	452.46	86400.11	12.63	20	452.70	6588.58	13.11

5.4.3 2D and 3D cantilever trusses

Among the 2D and 3D cantilever problem instances, only the 2D and 3D instances with 20 bars are solved to proven optimality within the 24 hr time limit of the full-MILO approach as it can be seen in Tables 13 and 14. 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 NS-MILO approach has found the global optimal solution.

From the results shown in Tables 13 and 14, 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 truss instances, the difference between the solution of the full-MILO and NS-MILO approach is less than 0.1%. 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.

In all the 15 3D cantilever trusses, the 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. For instance, the weight of the solution of the full-MILO approach for the 3D instances with 300 bars is 69% more than that of the solution of the NS-MILO approach.

5.4.4 Wing truss problem

None of the discrete problems of the wing trusses were solved to optimality in 24 hrs using the full-MILO approach, as it can be seen in Table 15. We let the wing instances run longer than 24 hrs, however, for the wing instance with 81 bars, after 48 hrs, more

Table 14: Weights (kg) and the solution times (s) for the 3D cantilever problem instances using the NS-MILO approach.

n_b	m	Full-MILO			NS-MILO			w_f/w_n
		w_f	t_f	gap(%)	n_s	w_n	t_n	
1	20	9.75	247.55	0.07	4	9.75	0.51	1.00
2	40	21.24	86400.02	25.76	10	21.18	1559.91	1.00
3	60	24.82	86400.02	30.03	7	24.75	2542.68	1.00
4	80	31.22	86400.03	32.43	5	31.18	4500.35	1.00
5	100	32.04	86400.04	34.36	15	31.43	4637.38	1.02
6	120	59.34	86400.05	36.52	30	59.22	4300.05	1.00
7	140	85.50	86400.06	42.52	5	78.46	4593.84	1.09
8	160	99.64	86400.03	42.21	15	89.55	5668.90	1.11
9	180	123.99	86400.05	40.95	14	113.33	5026.34	1.09
10	200	158.74	86400.06	49.10	8	122.31	4518.66	1.30
11	220	227.43	86400.03	58.14	8	144.21	3939.17	1.58
12	240	255.32	86400.04	55.61	7	172.02	4981.69	1.48
13	260	244.69	86400.12	50.02	16	183.73	6321.59	1.33
14	280	375.48	86400.07	60.04	7	222.61	5254.40	1.69
15	300	408.42	86400.05	58.73	30	246.14	10337.08	1.66

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 in Table 15, we can see that the best solution obtained by 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 12%-291% 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 continuous model solution is listed in Table 15. We also solved the wing instances with the software package SNOPT (Gill et al., 2005), which converged to the same solutions for the continuous model as those of 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. 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, most of the time, less than 12% for the wing instances.

Table 15: Weight (kg) and the solution time (s) for the wing problem instances using the NS-MILO approach.

# 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	19166.90	86400.02	16454.83	8.78	42	17147.00	230.06	375.55	1.04	1.12
99	16361.55	86400.02	13208.30	11.18	41	14106.27	189.40	456.18	1.07	1.16
117	14732.28	86400.02	11797.77	33.70	59	12994.64	2398.43	36.02	1.10	1.13
135	14791.77	86400.02	10845.89	22.99	28	11941.59	1794.42	48.15	1.10	1.23
153	16384.05	86400.03	10145.42	274.17	64	11090.73	7518.98	11.49	1.09	1.48
171	15045.04	86400.01	9508.32	92.78	49	10426.03	6570.13	13.15	1.10	1.44
207	20527.61	86400.12	8656.48	166.32	81	9657.39	11249.02	7.68	1.12	2.13
225	21615.53	86400.04	8320.28	332.60	54	9270.02	9350.30	9.24	1.11	2.33
243	18696.90	86400.13	8170.89	246.15	23	9127.25	6705.11	12.89	1.12	2.05
261	25324.62	86400.27	7923.76	298.48	99	8708.22	26799.14	3.22	1.10	2.91
279	16821.70	86400.05	7833.36	874.01	68	8756.50	25899.99	3.34	1.12	1.92
297	23051.34	86400.03	7699.78	887.11	67	8470.54	31538.54	2.74	1.10	2.72
315	21016.83	86400.24	7590.09	1100.64	62	10555.56	20431.61	4.23	1.39	1.99

The objective function improvement of the full-MILO and NS-MILO approach for the wing instance with 315 bars is plotted in Figure 7. 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 double the weight of the NS-MILO approach solution. On the other hand, the NS-MILO approach took 1.82 hrs to find a feasible solution. The reason is that the first 25 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 1.82 hrs. The weight of the initial solution of the NS-MILO approach is 210,953.22 kg. The objective function improved quickly and the NS-MILO stopped at 5.67 hrs with a solution that has a weight of 10,555.56 kg. The weight of the solution of the NS-MILO approach is 50% lower than that of the full-MILO approach, and is obtained in 23.6% of the time it takes for full-MILO.

In Figure 8, 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 cantilever problem for the instances with more than 170 bars, and the solution time of the wing instances is significantly more than that of the 3D cantilever instances. This reflects the increasing complexity of the three test sets. Additionally, we can see in Figure 8 and Tables 13, 14, and 15 that the number of subproblems and the overall solution time increases only moderately as the problem size increases.

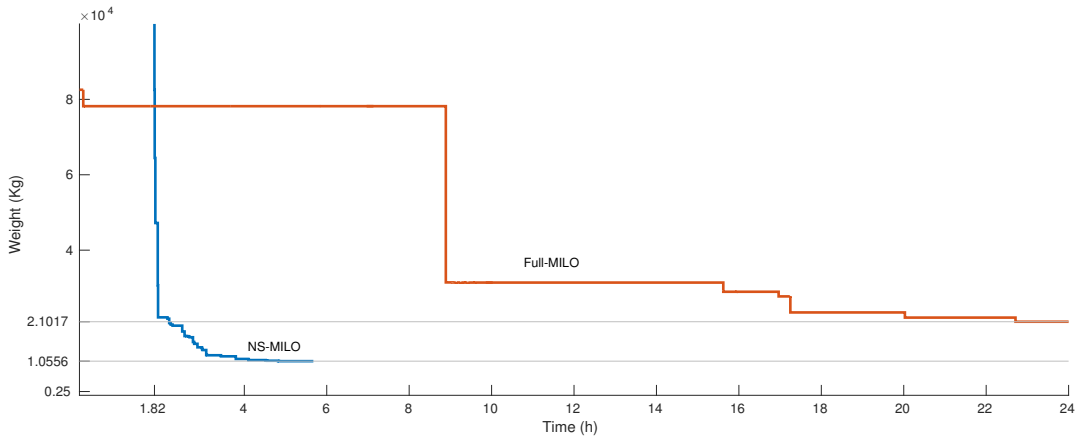


Figure 7: Comparison of the convergence of the full-MILO and NS-MILO approaches for the wing problem instance with 315 bars.

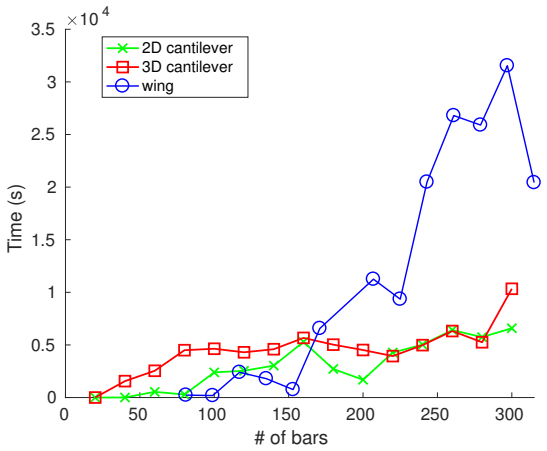
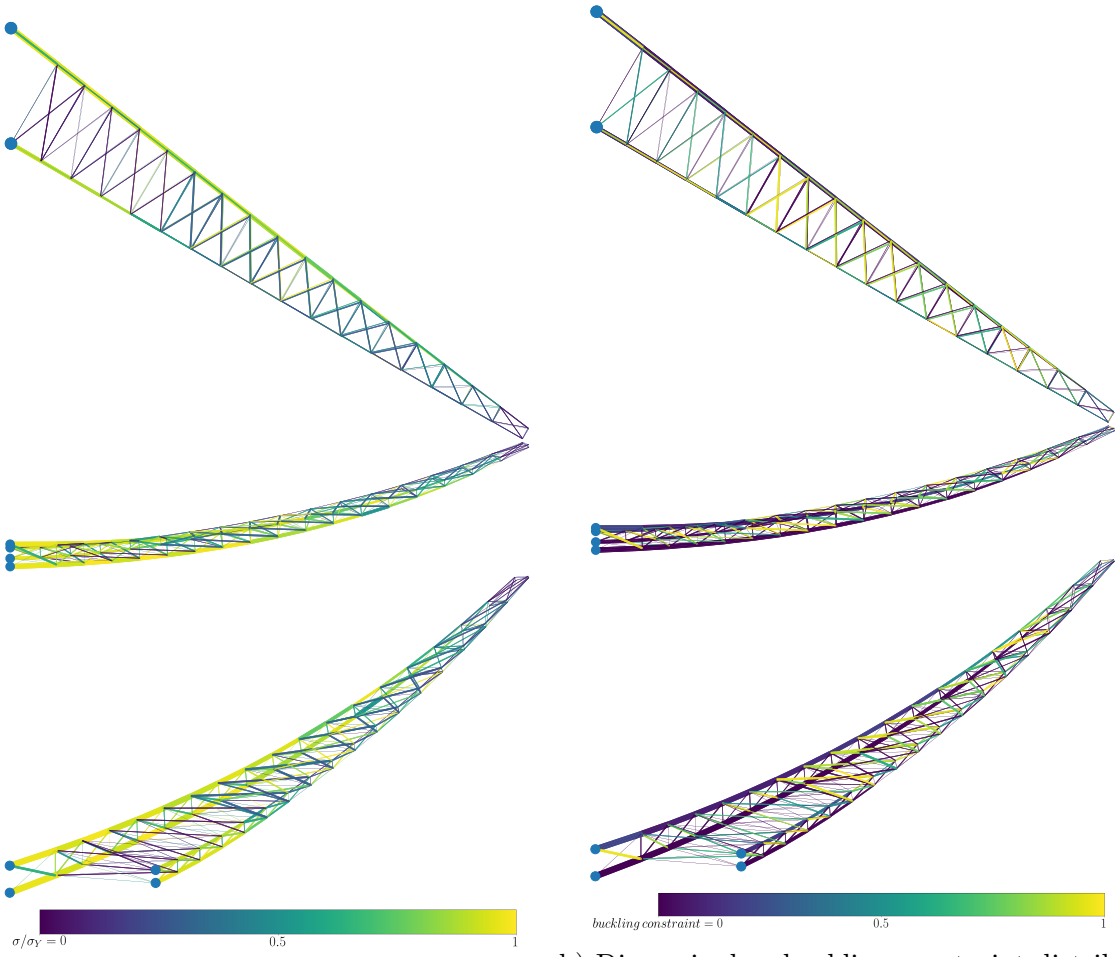


Figure 8: Solution time for the NS-MILO approach versus the number of bars in the truss.

To obtain a better physical intuition, 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 the from NS-MILO approach in Figure 9. The stress constraints are more active for the horizontal bars close to the root of the wing (the fixed end). This is because those bars are mainly responsible for taking the large bending moment around the root. As for the buckling constraints, they are more critical for the upper surface bars because these bars are under compression.

6 Conclusions

We presented various optimization models for the continuous and discrete truss design problems that include the Euler buckling constraints, Hooke's law, and bounds for



a) Dimensionless stress distribution.

b) Dimensionless buckling constraint distribution.

Figure 9: Stress and buckling constraint distribution for the wing problem instance with 315 bars.

stress and displacements are also considered. We also proposed the NS-MILO methodology to generate high quality solutions for those problems, in which a sequence of MILO subproblems in a moving neighborhood search framework are solved. The new methodology enables us to produce high quality solutions for previously unsolvable truss design problems. Numerical results indicate that the NS-MILO approach is significantly faster than simply using a MILO solver to solve the original truss design problems. Additionally, for all the generated wing problem instances, the best solution found is significantly better than the solution obtained from attempting to solve directly the original problems. The weight of the solution of the wing problem instances using the NS-MILO approach is up to 66% less than that of the full-MILO approach and is obtained 2.7—456 times faster.

Acknowledgement

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

Appendix

The discrete set set of the cross-sectional areas for the various problems are listed below.

10-bar truss

$$S = \{1.62,\ 1.8,\ 1.99,\ 2.13,\ 2.38,\ 2.62,\ 2.63,\ 2.88,\ 2.93,\ 3.09,\ 3.13,\ 3.38,\ 3.47,\ 3.55,\ 3.63,\ 3.84,\ 3.87,\ 3.88,\ 4.18,\ 4.22,\ 4.49,\ 4.59,\ 4.80,\ 4.97,\ 5.12,\ 5.74,\ 7.22,\ 7.97,\ 11.50, 13.50,\ 13.90, 14.20,\ 15.50,\ 16.00, 16.90,\ 18.80,\ 19.90,\ 22.00, 22.90,\ 26.50,\ 30.00,\ 33.5,\ 35,\ 37.5,\ 40,\ 42.5,\ 45,\ 47.5,\ 50,\ 52.5,\ 55,\ 57.5,\ 60,\ 62.5,\ 65,\ 67.5,\ 70,\ 72.5,\ 75,\ 77.5,\ 80,\ 82.5,\ 85,\ 87.5,\ 90\}(\text{in}^2).$$

72-bar truss

$$S = \{\ 0.111,\ 0.141,\ 0.196,\ 0.250,\ 0.307,\ 0.391,\ 0.442,\ 0.563,\ 0.602,\ 0.766, .785,\ .994,\ 1,\ 1.228,\ 1.266,\ 1.457,\ 1.563,\ 1.62,\ 1.8,\ 1.99,\ 2.13,\ 2.38,\ 2.62,\ 2.63,\ 2.88,\ 2.93,\ 3.09,\ 3.13,\ 3.38,\ 3.47,\ 3.55,\ 3.63,\ 3.84,\ 3.87,\ 3.88,\ 4.18,\ 4.22,\ 4.49,\ 4.59,\ 4.8,\ 4.97,\ 5.12,\ 5.74,\ 7.22,\ 7.97,\ 8.53,\ 9.3,\ 10.85,\ 11.5,\ 13.5,\ 13.9,\ 14.2,\ 15.5,\ 16,\ 16.9,\ 18.8, 19.9,\ 22,\ 22.9, 24.5,\ 26.5, 28,\ 30, 33.5\ \}(\text{in}^2).$$

2D cantilever truss problem

$$S = \{\ 0.25,\ 0.5,\ 0.75,\ 1,\ 2,\ 3,\ 4,\ 6,\ 7,\ 8,\ 9,\ 10,\ 12,\ 14,\ 16,\ 18,\ 20,\ 22,\ 24,\ 26,\ 28,\ 30,\ 32,\ 34,\ 36,\ 38,\ 40,\ 42.5,\ 45,\ 47.5,\ 50,\ 52.5,\ 55,\ 57.5,\ 60,\ 62.5,\ 65,\ 70,\ 75, 80,\ 85\}(\text{cm}^2).$$

3D cantilever truss problem

$$S = \{\ .25,\ .5,\ .75,\ 1,\ 2,\ 3,\ 4,\ 6,\ 7,\ 8,\ 9,\ 10,\ 12,\ 14,\ 16,\ 18,\ 20,\ 22,\ 24,\ 26,\ 28,\ 30,\ 32,\ 34,\ 36,\ 38,\ 40,\ 42.5,\ 45,\ 47.5,\ 50,\ 52.5,\ 55,\ 57.5,\ 60,\ 62.5,\ 65,\ 70,\ 75, 80,\ 85\}(\text{cm}^2).$$

Wing truss problem

$$S = \{\ .25,\ 1,\ 5,\ 10,\ 15,\ 20,\ 25,\ 30,\ 35,\ 40,\ 45,\ 50,\ 55,\ 60,\ 65,\ 70,\ 75,\ 80,\ 85,\ 90,\ 95,\ 100,\ 150,\ 200,\ 250,\ 300,\ 350,\ 400,\ 450,\ 500,\ 550,\ 600,\ 650,\ 700,\ 750,\ 800,\ 850,\ 900,\ 950,\ 1000,\ 1050,\ 1100,\ 1150,\ 1200\}(\mathrm{cm}^2).$$

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—part II: 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.

Barbosa, H. J., Lemonge, A. C., and Borges, C. C. (2008). A genetic algorithm encoding for cardinality constraints and automatic variable linking in structural optimization. *Engineering Structures*, 30(12):3708 – 3723.

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.

Bendsoe, M. P. and Sigmund, O. (2013). *Topology optimization: theory, methods, and applications*. Springer Science & Business Media.

Bland, J. A. (2001). Optimal structural design by ant colony optimization. *Engineering Optimization*, 33(4):425–443.

Brooks, T. R., Kenway, G. K. W., and Martins, J. R. R. A. (2018). uCRM: An aerostructural model for the study of flexible transonic aircraft wings. *AIAA Journal*. (In press).

Cai, J. and Thierauf, G. (1993). Discrete optimization of structures using an improved penalty function method. *Engineering Optimization*, 21(4):293–306.

Camp, C. and Farshchin, M. (2014). Design of space trusses using modified teaching–learning based optimization. *Engineering Structures*, 62-63:87 – 97.

Camp, C. V. (2007). Design of space trusses using big bang–big crunch optimization. *Journal of Structural Engineering*, 133(7):999–1008.

Camp, C. V. and Bichon, B. J. (2004a). Design of space trusses using ant colony optimization. *Journal of Structural Engineering*, 130(5):741–751.

Camp, C. V. and Bichon, B. J. (2004b). Design of space trusses using ant colony optimization. *Journal of Structural Engineering*, 130(5):741–751.

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.

Haftka, R. T. and Gürdal, Z. (2012). *Elements of structural optimization*, volume 11. Springer Science & Business Media.

Hajela, P. and Lee, E. (1995). Genetic algorithms in truss topological optimization. *International Journal of Solids and Structures*, 32(22):3341 – 3357.

Hanafi, S. (2016). New variable neighbourhood search based 0-1 MIP heuristics. *Yugoslav Journal of Operations Research*, 25(3).

Hansen, P. and Mladenović, N. (2003). *Variable Neighborhood Search*, pages 145–184. Springer US, Boston, MA.

Ho-Huu, V., Nguyen-Thoi, T., Vo-Duy, T., and Nguyen-Trang, T. (2016). An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. *Computers and Structures*, 165:59 – 75.

Kaveh, A., Azar, B. F., and Talatahari, S. (2008). Ant colony optimization for design of space trusses. *International Journal of Space Structures*, 23(3):167–181.

Kaveh, A. and Ghazaan, M. I. (2015). A comparative study of CBO and ECBO for optimal design of skeletal structures. *Computers & Structures*, 153:137 – 147.

Kaveh, A. and Kalatjari, V. (2004). Size/geometry optimization of trusses by the force method and genetic algorithm. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 84(5):347–357.

Kaveh, A. and Mahdavi, V. (2014). Colliding bodies optimization method for optimum discrete design of truss structures. *Computers & Structures*, 139:43 – 53.

Kaveh, A. and Talatahari, S. (2009). A particle swarm ant colony optimization for truss structures with discrete variables. *Journal of Constructional Steel Research*, 65(8):1558 – 1568.

Kripka, M. (2004). Discrete optimization of trusses by simulated annealing. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 26:170–173.

Lazić, J. (2010). *New variants of variable neighbourhood search for 0-1 mixed integer programming and clustering*. PhD thesis, Brunel University, School of Information Systems, Computing and Mathematics.

Li, L., Huang, Z., and Liu, F. (2009). A heuristic particle swarm optimization method for truss structures with discrete variables. *Computers & Structures*, 87(7):435 – 443.

Mahfouz, S. Y. (1999). *Design optimization of structural steelwork*. PhD thesis, University of Bradford, United Kingdom.

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.

Mellaert, R. V., Mela, K., Tiainen, T., Heinisuo, M., Lombaert, G., and Schevenels, M. (2017). Mixed-integer linear programming approach for global discrete sizing optimization of frame structures. *Structural and Multidisciplinary Optimization*, 57(2):579–593.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097 – 1100.

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

Petrovic, N., Kostic, N., and Marjanovic, N. (2017). Design of space trusses using big bang–big crunch optimization. 2:98–103.

Rahami, H., Kaveh, A., and Gholipour, Y. (2008). Sizing, geometry and topology optimization of trusses via force method and genetic algorithm. *Engineering Structures*, 30(9):2360 – 2369.

Rajeev, S. and Krishnamoorthy, C. S. (1992). Discrete optimization of structures using genetic algorithms. *Journal of Structural Engineering*, 118(5):1233–1250.

Rajeev, S. and Krishnamoorthy, C. S. (1997). Genetic algorithms-based methodologies for design optimization of trusses. *Journal of Structural Engineering*, 123(3):350–358.

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.

Sadollah, A., Bahreininejad, A., Eskandar, H., and Hamdi, M. (2012). Mine blast algorithm for optimization of truss structures with discrete variables. *Computers & Structures*, 102-103:49 – 63.

Sadollah, A., Eskandar, H., Bahreininejad, A., and Kim, J. H. (2015). Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures. *Computers & Structures*, 149:1 – 16.

SeokLee, K. and Geem, Z. W. (2004). A new structural optimization method based on the harmony search algorithm. *Computers & Structures*, 82(9–10):781–798.

Sonmez, M. (2011). Discrete optimum design of truss structures using artificial bee colony algorithm. *Structural and Multidisciplinary Optimization*, 43(1):85–97.

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 0–1 programs. *Optimization and Engineering*, 8(2):163–192.

Stolpe, M. (2011). To bee or not to bee—comments on "discrete optimum design of truss structures using artificial bee colony algorithm". *Structural and Multidisciplinary Optimization*, 44(5):707–711.

Stolpe, M. (2016). Truss optimization with discrete design variables: a critical review. *Structural and Multidisciplinary Optimization*, 53(2):349–374.

Svanberg, K. and Werme, M. (2005). A hierarchical neighbourhood search method for topology optimization. *Structural and Multidisciplinary Optimization*, 29(5):325–340.

Svanberg, K. and Werme, M. (2007). Sequential integer programming methods for stress constrained topology optimization. *Structural and Multidisciplinary Optimization*, 34(4):277–299.

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.

Wu, S.-J. and Chow, P.-T. (1995). Steady-state genetic algorithms for discrete optimization of trusses. *Computers and Structures*, 56(6):979 – 991.

Zeng, S. and Li, L. a. (2012). Particle swarm-group search algorithm and its application to spatial structural design with discrete variables. *International Journal of Optimization in Civil Engineering*, 2(4).