Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

Article

Extraction of Structural System Designs from Topologies via Morphological Analysis and Artificial Intelligence

Achyuthan Jootoo [1,†*] and David Lattanzi [2]

[1] PhD candidate, Department of Civil, Environmental, and Infrastructure Engineering, George Mason University, Fairfax, VA, USA, 22030; ajootoor@gmu.edu
[2] Assisstant Professor, Department of Civil, Environmental, and Infrastructure Engineering, George Mason University, Fairfax, VA, USA, 22030; dlattanzi@gmu.edu
* Correspondence: ajootoor@gmu.edu; Tel.: +1-571-447-2842
† Current address: Department of Civil, Environmental, and Infrastructure Engineering, George Mason University, Fairfax, VA, USA, 22030

Abstract: Structural system design is the process of giving form to a set of interconnected components subjected to loads and design constraints while navigating a complex design space. While safe designs are relatively easy to develop, optimal designs are not. Modern computational optimization approaches employ population based metaheuristic algorithms to overcome challenges with the system design optimization landscape. However, the choice of the initial population, or ground structure, can have an outsized impact on the resulting optimization. This paper presents a new method of generating such ground structures, using a combination of topology optimization (TO) and a novel system extraction algorithm. Since TO generates monolithic structures, rather than systems, its use for structural system design and optimization has been limited. In this paper, truss systems are extracted from topologies through morphological analysis and artificial intelligence techniques. This algorithm, and its assessment, constitutes the key contributions of this paper. The structural systems obtained are compared with ground truth solutions to evaluate the performance of the algorithms. The generated structures are also compared against benchmark designs from the literature. The results indicate that the presented truss generation algorithm produces structures comparable to those generated through metaheuristic optimization, while mitigating the need for assumptions about initial ground structures.

Keywords: Structural Optimization; System Design; Artificial Intelligence; Morphological operations; Topology Optimization; Structural Design.

1. Introduction

The conventional structural design process centers around the use of design requirements and constraints to choose a single efficient structure from the large decision space of all possible designs, all the while working with time and budget limitations. The structural design space has been shown to be typically comprised of local optima, resulting in non-convex optimizations [1,2], and with solutions often dependent upon specified initial conditions [3]. Compounding the problem, structures are generally composed of a system of components, and so the performance and efficiency of any design is governed by a combination of the system topology and the design of each component as well. Thus, the design space for any set of design criteria can be extremely large.

Modern computational approaches for determining efficient structural designs often use a metaheuristic, such as evolutionary computation, to obtain an optimal solution. Such metaheuristic approaches are typically initialized via a ground structure approach, in which an assumed structural system configuration serves as an initial sample population [4,5]. This paper presents a new method of

generating such ground structures automatically, using a combination of topology optimization (TO) and a novel system extraction algorithm.

1.1. Literature Review

The overall methodology for design optimization in the literature primarily revolves around initializing the optimization problem with initial structural assemblies and iteratively improving them, resulting in an optimal structure. [6] presented a method for discrete optimization of structural assemblies using genetic algorithms. The structure was treated as a collection of nodes interconnected via structural members. [7] used the ground structure approach for truss topology optimization. Similar methods of structural optimization using the ground structure approach have been applied to diverse problems [2–4] using various algorithms [5,8–11]. Further, with improved optimization heuristics and higher computation capabilities, optimization has been performed for complex structures. However, the key drawback to such approaches is that the final optimal design is dependent on the initial structural systems selected for optimization [1,3] and a poorly chosen ground structure may lead to suboptimal designs.

Topology optimization (TO), the process of finding the optimal layout (topology) of a structure, requires fewer assumptions about the design domain compared to metaheuristic approaches. Two of the most established TO approaches are *density based* methods and *hard kill* methods. Density based methods discretize the design domain into a mesh and optimize the design by varying the density of each element of the mesh based on an objective function. Fundamentally, this is a challenging large-scale integer programming problem. In order to simplify the TO problem and express it as a function of continuous design variables, an interpolation function with a penalization mechanism is generally used. Different penalization and interpolation methods lead to different algorithms for TO such as SIMP [12,13], Rational Approximation of Material Properties [14] and SINH [15]. A detailed discussion of the SIMP approach, including implementation issues, can be found in [16]. For a review of the recent developments in SIMP, the reader is referred to [17,18]. Research has also been conducted in using TO with material failure constraints [19] and stress constraints [20]. TO has been widely used in different domains such as fluid flow [21], heat transfer [22] and aerospace design [23,24]. Hard kill methods gradually remove (or add) material from the design based on a heuristic strategy, with Evolutionary Structural Optimization (ESO), developed by [25], as the most well known of such methods. In this research, solid isotropic material with penalization (SIMP) was implemented due to its widespread usage [17] and the chaotic convergence behavior of Evolutionary Structural Optimization (ESO)[26].

1.2. Contribution of this Research

As stated, a critical aspect of metaheuristic design optimization is the choice of ground structure that serves as the initial population for optimization. Selecting such ground structures requires implicit assumptions about the location of those structures in the design landscape. As such, there is a need for methods to generate such structures automatically, without strong assumptions about the design landscape.

Topology optimization presents an avenue for addressing this need. However regardless of the specific method used, topology optimization results in monolithic structures, rather than the systems of engineered components that comprise the majority of structural designs. This is a critical limitation, as it inhibits topology optimization from being used to either generate initial structures for further optimization, or for generating structural designs in many practical scenarios.

Presented in this paper is a new algorithm to generating structural systems, in the context of truss design. This algorithm extracts structural systems from generated topologies, which can either be analyzed directly or serve as the initial ground structures for further optimization. The extraction is accomplished through a combination of morphological analysis and artificial intelligence techniques. This system extraction approach, and its evaluation, constitutes the primary contribution of this work.

In this manuscript, the complete system extraction methodology is first presented. This is followed by a study on the behavior of systems manually extracted from monolithic structural topologies. The performance of three automatic extraction algorithm variants is then evaluated. A sensitivity study of algorithm performance is included as well.

2. Methodology

The system generation process starts with a given design domain and the initial conditions necessary to support topology optimization. A system topology is then generated through topology optimization, based on these inputs (Section 2.1). Nodes and structural elements are extracted from this topology through a combination of morphological analysis and associated computational techniques (Section 2.2), yielding a structural assembly (system). The extracted structural assembly can then be analyzed using FEA to perform structural analysis to obtain the deflections and member stresses. This assembly can also serve as a basis for structural optimization. An overview of the process is provided in Fig. 1.

2.1. Topology Optimization

While any number of topology generation approaches can be used as part of the overall methodology, in this work the SIMP density-method is used due to its consistent performance across a range of application scenarios [26]. [27] first presented this approach for generating optimal topologies in structural design using numerical optimization and homogenization methods. The objective of their optimization was to minimize the work done by applied loads as shown in Eq. 1.

In Eq. 1, E_{ijkl} is the elasticity matrix, $a_E(u,v)$ is the energy in bilinear form i.e. the internal virtual work of an elastic body at equilibrium $\mathbf{u}$ and for an arbitrary virtual displacement $\mathbf{v}$, $L(\mathbf{v})$ represents the load linear form of the energy, $\mathbf{f}$ and $\mathbf{t}$ are body and boundary forces, $\epsilon_i j$ represents the linearized strain in each direction. Thus, this formulation minimizes the strain energy, while limiting the displacements as per the imposed design constraints and the admissible displacements (U).

$$\begin{aligned} &\text{Minimize} \quad L(\mathbf{u}) \\ &\quad \text{subject to } a_E(\mathbf{u},\mathbf{v}) = L(\mathbf{v});\ \text{all } \mathbf{v} \in U,\ \text{design constraints} \\ &\qquad \text{where} \\ &\qquad\quad a_E(\mathbf{u},\mathbf{v}) = \int_\Omega E_{ijkl}\epsilon_{kl}(u)\epsilon_{ij}(v)dx, \\ &\qquad\quad \epsilon_{ij}(\mathbf{u}) = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}), \\ &\qquad\quad L(\mathbf{v}) = \int_\Omega \mathbf{f}.\mathbf{v}dx + \int_\Gamma \mathbf{t}.\mathbf{v}ds \end{aligned} \tag{1}$$

The design space of the problem is defined as Ω, the strain as ϵ, and the indicator function χ indicates whether material is present at a point x or not. Hence, for all $\mathbf{x} \in \Omega$, either $\chi(x) = 1$ or $\chi(x) = 0$ i.e. at any point in the domain, either material is present or it is not. This formulation also assumes that the material is linearly elastic. For example, in Fig. 1a, the black rectangle is the design domain, Ω, with the black color throughout the design domain indicating that $\chi(x) = 1$ everywhere. As the design is optimized, the design domain remains the same but the location of material in the domain changes, resulting in Fig. 1b. The elasticity matrix for the material is therefore defined as $E_{ijkl} = \chi(x)E^0_{ijkl}$ at any location $\mathbf{x}$.

This optimization problem can be modeled as a large scale integer programming problem and is generally ill posed and difficult to solve [16]. [12] modified the problem by replacing the indicator function $\chi(x)$ with a density function, $\rho(x)$, that is continuous between 0 and 1, rather than binary. The modified elasticity matrix is $E_{ijkl} = \rho(x)E^0_{ijkl}$. The density function transforms the problem from a large scale integer optimization problem to an easier to handle continuous problem. Since the desired

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

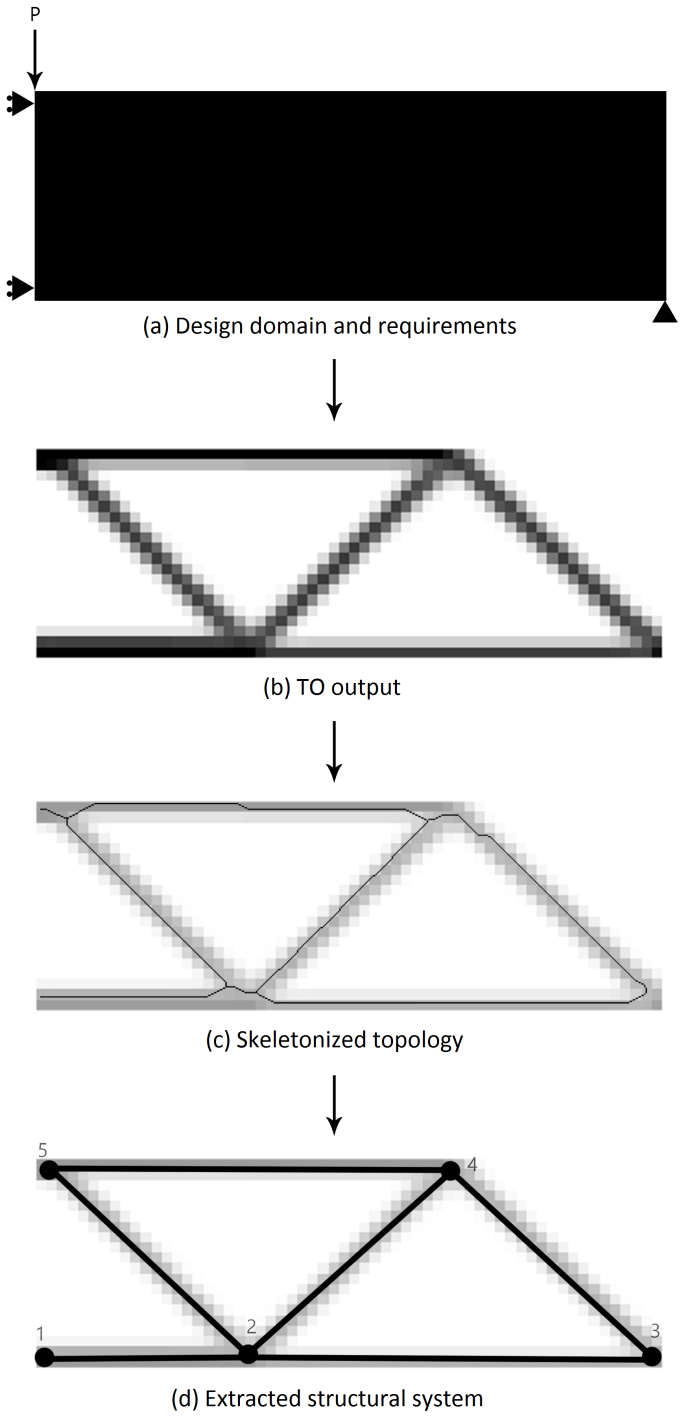


Figure 1. Desired representation of a TO structure for structural optimization

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

density value is either 0 or 1, constraints are imposed via penalization. This penalization supplies the name of this approach: solid isotropic material with penalization (SIMP). The definition of E_{ijkl} is further modified and it is redefined as:

$$E_{ijkl} = \rho(x)^p E^0_{ijkl}, \qquad p > 1 \tag{2}$$

Choosing $p > 1$ makes the densities between 0 and 1 unfavorable since for the same amount of material, it provides a lesser stiffness due to the exponent p. In general, in order to obtain solutions with minimal volumes with intermediate densities, or binary designs, it is recommended to use $p >= 3$. Once the problem has been thus formulated, the density of the material within the design space Ω is optimized. The design at each iteration is analyzed through finite element analysis where each region with a density value corresponds to an element. The value of the objective function is thus computed. Based on the objective function, the topology is iteratively updated to yield a new topology until the termination criteria are satisfied.

Since SIMP treats the topology optimization problem as essentially a material redistribution problem, all resulting topologies are monolithic structures, rather than structural systems of interconnected components. The second part of the presented computational process is designed to extract such systems directly from the topologies.

2.2. Node-Element Extraction from Topologies

After a topology is obtained, the next step is to convert it into a system consisting of nodes (joints) and connecting elements to suit structural engineering purposes. The morphological skeleton of the topology, as well as branch and end points in that skeleton, are first found. The skeleton is then further analyzed using one of the three variants of a node-element identification algorithm (NEI) developed in this study (Fig. 1). The first algorithm, NEI-CL (Section 2.2.2), is based on clustering directional vectors that initiate at a given a node and terminate at discretized locations throughout the topology. The second algorithm, NEI-HLT (Section 2.2.3), uses Hough transform line-finding to identify components [28]. The third algorithm, NEI-TRA (Section 2.2.4), is based on the concept of a structural component as a path between nodes such that there is no other node along this path. Therefore, given the location of nodes, this algorithm traverses from one node to another and identifies the connectivity between nodes.

2.2.1. Topology skeletonization

The first step in identifying nodes and elements is to convert the generated topology into a binarized and discretized representation, analogous to a black and white image. The binarization is performed using Otsu's method [29]. The *skeleton* of the topology is then determined via morphological thinning of the binarized topology [30]. The branch points and end points of the skeleton are detected automatically by analyzing the 8-connectivity of the skeleton, yielding the potential nodes of the structure. However, it is important to note that this method of identifying nodes does not necessarily yield all of the correct nodes in some cases because of approximations in the skeletonization process. Evidence of this effect will be shown in later analyses.

2.2.2. NEI-CL:Directional clustering algorithm variant

For the NEI-CL algorithm variant, the direction vectors from each node to each pixel in the skeleton, within a specified distance from the node, are calculated. These vectors are clustered together for each node using k-means clustering [31], with each cluster of vectors corresponding to a potential structural element. Any two nodes with a clustered set of direction vectors oriented towards each other are matched and identified as nodes and endpoints of a potential element. Structural elements are then identified along each cluster's direction vector, and the endpoints of these elements are labeled as nodes.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008


```
start
    Morphologically thin image to create skeleton
    Define skeleton branch points and end points as potential nodes
    For each potential node:
        find distances and orientations to all pixels within a range
        cluster orientations together
    Match nodes which orient towards each other
    Define elements as connections between matched nodes
    For each potential node, find elements based on their orientations
    Find elements by connecting nodes with material along the straight line joining them
    Remove similar and duplicate nodes, elements
    Remove nodes with no connectivity
end
```

Figure 2. Pseudo code for the NEI-CL algorithm

```
start
    Morphologically thin image to create skeleton
    Use Hough transform to find lines in the image
    Merge and extend nearby lines which are in similar directions
    Cluster nearby endpoints of the lines to form nodes
end
```

Figure 3. Pseudo code for the NEI-HLT algorithm

This process occasionally results in multiple, and closely spaced, nodes at the ends of identified elements. As a post-processing step, the set of nodes is clustered, again via the k-nearest neighbors algorithm, to remove closely spaced and potentially duplicate nodes. Potential elements are also evaluated for duplicates. Finally, all nodes without any connected elements are removed, yielding the final set of nodes and elements. The pseudo code for this algorithm is shown in Fig. 2.

2.2.3. NEI-HLT: Hough line transform based algorithm variant

As with the NEI-CL algorithm, the topology is first converted into a morphological skeleton, and branch and endpoints are determined. The Hough transform [28,30], a well-established method for finding line segments in images, is then applied. Given the assumption that all structural elements will connect between nodes in a straight line, lines found through the Hough transform are labeled as candidate structural elements. Duplicate endpoints of lines are merged using k-means clustering and line segments with similar directionality and endpoints are joined to form a single element. The pseudo code for this algorithm is shown in Fig. 3.

2.2.4. NEI-TRA: Node to node traversal algorithm variant

The NEI-TRA algorithm variant iteratively traverses each segment initiating from each potential node of the skeleton and stores the coordinates of each pixel in the traversal. The traversal, starting from a node, proceeds by repeatedly moving from one pixel to another pixel in its 8 point neighborhood. Once another identified node is reached and the traversal is complete, the goodness of fit of a line connecting the two nodes is evaluated against the traversed pixels. The goodness of fit is evaluated by using the coefficient of correlation and checking if all the traversed pixels fall within a specified bandwidth from the line. If the fit is good, then an element is assigned between the two nodes. In Fig. 5a, the pixels from traversal along the segments 1, 2 and 3 have a good fit with the straight lines connecting the nodes. If the fit is not good (Fig. 5a, segment 4), then the set of pixels is divided into subsegments of the skeleton of a specified length (Fig. 5b) and the average direction vector is computed for each subsegment. The direction vectors are then clustered so that subsegments with similar direction vectors together are part of the same cluster. Subsegments in each cluster constitute an element. Fig. 5c shows a sample in which the subsegments have been clustered to yield elements. If the endpoints of this element are not already present in the potential nodes, they are added as additional nodes. Lastly, similar and duplicate nodes and elements are removed through k-means clustering to obtain the final set of nodes and elements. The pseudo code for the NEI-TRA algorithm is shown in Fig. 4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008


```
start
    Morphologically thin image to create skeleton
    Define skeleton branch points and end points as potential nodes
    For each node in potential node
        For each branch for the current node
            Traverse along the selected branch till another node is reached
            Store each pixel that was part of this travsersal
            Check if the line connecting the two nodes is a good fit for the pixels
            If it is not a good fit
                Subdivide the pixels into segments of the skeleton
                Compute the average direction vector of each segment
                Cluster the direction vectors to identify segments in each cluster
                All segments in each cluster together constitute an individual element
                If endpoint of elements does not occur in potential nodes, add new node
            Else if it is a good fit
                Add element connecting the two nodes
    Remove similar and duplicate nodes and elements
end
```

Figure 4. Pseudo code for the NEI-TRA algorithm

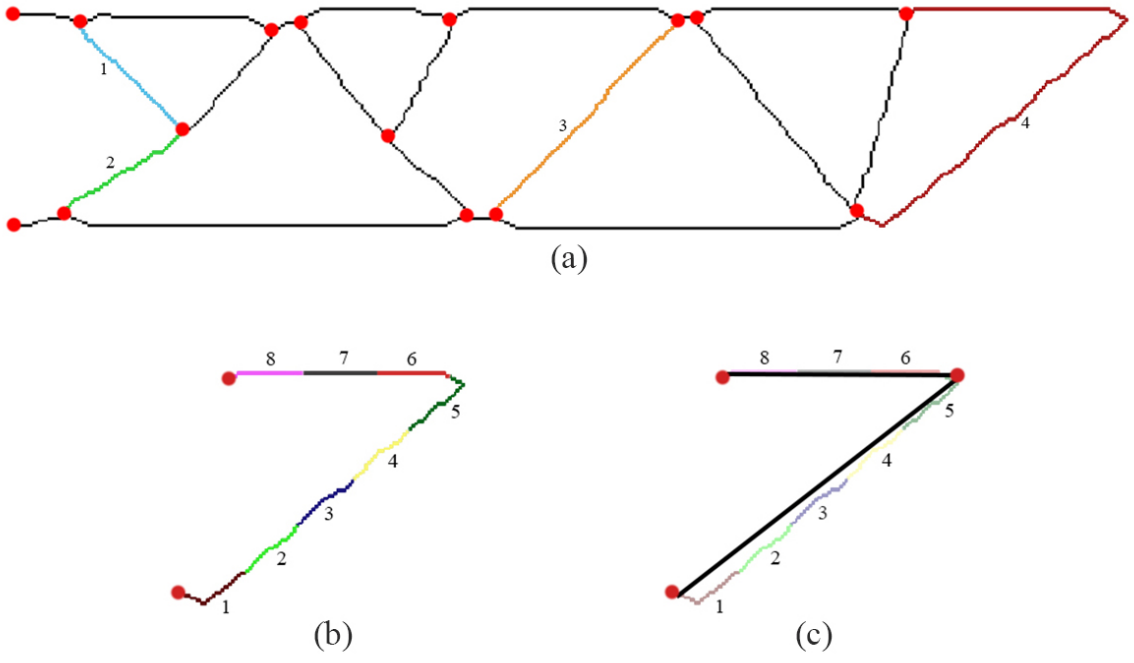


Figure 5. Cluster based element identification in NEI-TRA algorithm. (a) Segments connecting branch points (b) Subsegments for identifying elements from the segment (c) Elements identified after clustering

3. Experiments and Results

This section presents and discusses the results of experiments designed to evaluate the suitability of TO for generating structural systems. First, truss systems manually extracted from topologies were compared against benchmark optimization solutions in the literature, in order to better understand the nature of systems extracted from topologies. The three automatic system extraction variants were then comparatively analyzed. A parameter sensitivity study of the NEI-TRA algorithm variant was then performed, as this variant produced the most consistent results.

3.1. Preliminary Analysis: Extracting Structural Systems from Topologies

In this study, the design constraints for two established problems from the structural optimization literature were used as a basis for TO. Structural systems were then manually extracted from the topologies, and the performance of these systems was compared against benchmark optimized solutions from the literature. This study was performed in order to validate the initial idea of generating structural systems using TO. The design problems shown in Fig. 6a and 7a were used as input for TO, and extracted systems were compared with the results published by [4,10,32–34]. It is important to clarify that the solutions in these previous works were obtained by starting from an initial structural system, and iteratively optimized by adding members, removing members and, in some cases, modifying node locations. These variations to the structure were introduced as per optimization heuristics such as a genetic algorithm, particle swarm optimization, or differential evolution.

For the first design domain, "Problem 1", a design space 18.288 meters (720 inches) wide and 9.144 meters (360 inches) high with two loads, each P = -444.82 kN (-100 kips), was specified. A design space 31.75 meters (1250 inches) wide and 6.35 meters (250 inches) high with 5 point loads, each P = -88.96 kN (-20 kips), acting on it was specified for "Problem 2". Loads and boundary conditions were applied as per Fig. 6a and 7a. Topologies were generated using a modified version of Sigmund's 99 line Matlab code for SIMP [35]. Each structure was converted to a truss system by manually identifying nodes and elements, and adding extra members if needed for stability. The results obtained were then imported into [36] for evaluation. The designs reported in the literature were also evaluated in Risa 2D.

SIMP requires a volume fraction parameter, a parameter that determines the fraction of the volume of the design domain that will contain material. Several volume fractions were tested and it was observed that higher volume fractions increased the weight of the structure and reduced the deflections and member stresses, as is to be expected. For both problems, a volume fraction was eventually selected that generated design weights comparable to prior efforts. Since the objective function of SIMP topology optimization is evaluated via finite element analysis, parameters for discretization of the design domain such as the number of elements needs to be specified. The impact of the fineness of the discretization has been well documented in the TO literature [16], and is not studied here. The mesh resolutions chosen here were 60x30 for Problem 1 and 175x35 for Problem 2 because changing the discretization around these resolutions the did not impact the TO result.

3.1.1. Metrics

As the the volume fraction was iteratively changed to match previous design weights, the structural weight was not useful as a comparative metric. Instead, the average stress, maximum stress, and maximum deflection were used for comparisons.

3.1.2. Results

For Problem 1, the volume ratio was specified as 0.2. With regards to the compared studies, [32] and [4] only varied the cross-sectional area of the initial structure but did not add or remove members. [10] presented a solution with no removal of members and another solution in which removal of structural members was permitted. A comparison of the weight, deflection, maximum stress and average stress is shown in Table 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

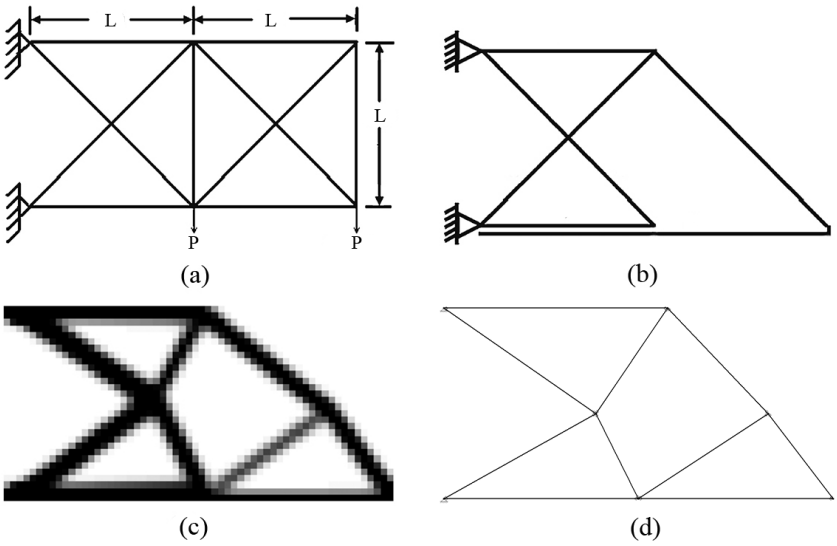


Figure 6. (a)Initial structure for prior solutions to Problem 1 [10], (b) optimal solution from [10], and (c) TO solution, and (d) TO solution converted to a truss

Table 1. Comparative results for Problem 1

Metrics	Perez and Behdinan (2007)	Xu et al. (2009)	Wu and Tseng (2010) *	Wu and Tseng (2010)**	TO System
Weight (kgs)	2278.94	2303.45	2295.38	2145.80	2323.07
Deflection (cm)	5.11	5.08	5.05	5.08	5.21
Maximum Stress (MPa)	172.51	140.65	172.37	128.93	57.98
Average Stress (MPa)	56.54	49.64	56.54	74.26	49.78

* Results without member removal
** Results with member removal

The structure optimized by [10], which permitted member removal, was 5.8% lighter than the other structures, which all had very similar weights. The TO-based truss was the heaviest, and had slightly higher deflections. The TO-based results reflect a maximum stress of 58.0 MPa and an average stress of 49.8 MPa, which is a notably more even stress distribution than the other structures. It is assumed that, given additional optimization, the TO-based system would converge to a solution similar to those of prior efforts.

The TO solution to Problem 2 was compared with the designs published in [33,34]. In both of these studies, an initial structure was assumed for optimization. The node locations could be modified, but member removal was not permissible. The area of cross-section of the members could also be modified to optimize the structure. For the topology optimization, the volume ratio was set to 0.2.

TO resulted in structures that were unstable when directly converted to trusses, a problem not encountered in Problem 1. Hence, the structure was first evaluated as a frame with complete joint fixity. Members were then manually added to this frame for stability, and it was converted to a truss with pin supports (Fig. 7d). A comparison of the results is shown in Table 2.

Table 2 shows that the structures optimized in the literature are lighter than the TO designed structures by about 2% for the frame structure and 6.7% for the truss structure. However, the TO structure has slightly lower maximum stresses, lesser average stress and significantly lower deflections. The deflection of the TO truss structure is less than half the deflection of the structures optimized by

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

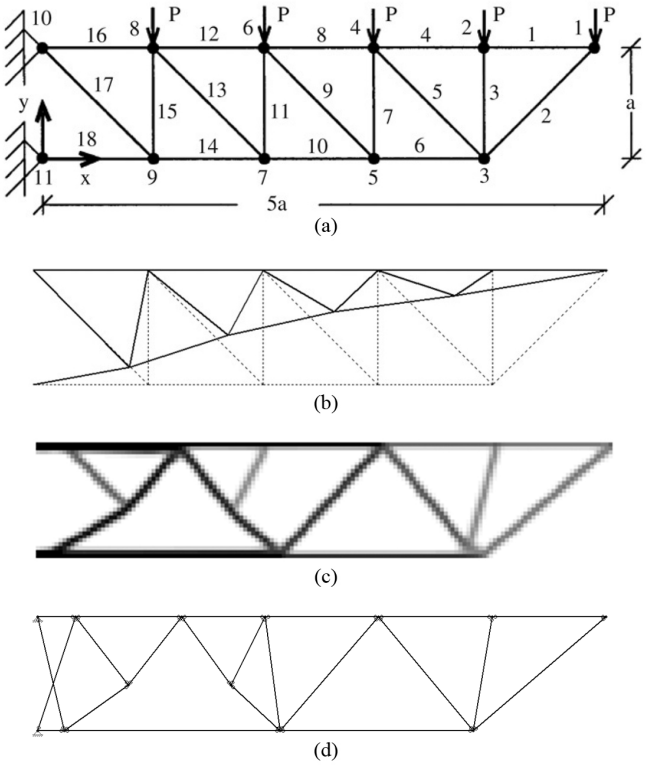


Figure 7. (a)Initial structure for prior solutions to Problem 2, (b) optimal solution from [34], (c) TO solution, and (d) TO solution converted to truss. Note the additional members needed for stability

[33] and [34]. Also, the variation in the member stresses is 13.6% lesser for the TO truss structure and 33.6% lesser for the TO frame structure, which indicates a more evenly distributed load in the TO structure. This is similar to the results of Problem 1.

What the results of this analysis illustrate is that systems extracted from structural topologies can perform comparably to systems optimized assuming an initial structural configuration. This is particularly true given that the extracted systems could be further optimized using any number of established techniques. Therefore, generating a topology and then extracting a system from it can serve as a full replacement for an initial estimate of a structural configuration. In such cases, the specified volume ratio becomes the controlling design parameter. It is also important to recognize that, as shown by Problem 2, topology optimization results can require some post-processing in order to generate stable results.

3.2. Extraction Algorithm Analysis

In the preliminary analysis, systems were manually extracted from topologies. The three automatic extraction algorithms designed to address this issue are analyzed in this section. Four diverse topologies (TO1, TO2, TO3, TO4) were generated and used for testing (Fig. 8). These topologies have variations in member thickness, number of members, and general topological complexities that illustrate both the capabilities and limitations of the extraction process. A ground truth extraction solution was developed based on manual labeling of nodes and elements in the TO structure. After evaluating the three variants, a parameter sensitivity analysis was conducted for NEI-TRA, as it produced the most consistent results.

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

Table 2. Comparative results for Problem 2

Metrics	Kaveh and Kalatjari (2004)	Rahami et al. (2008)	TO System (Frame)	TO System (Truss)*
Weight (kgs)	2062.89	2055.09	2098.73	2192.09
Deflection (cm)	46.33	51.18	28.37	22.94
Maximum Stress (MPa)	137.41	137.90	123.14	135.21
Average Stress (MPa)	75.01	79.57	59.43	62.88

* Truss obtained after adding members

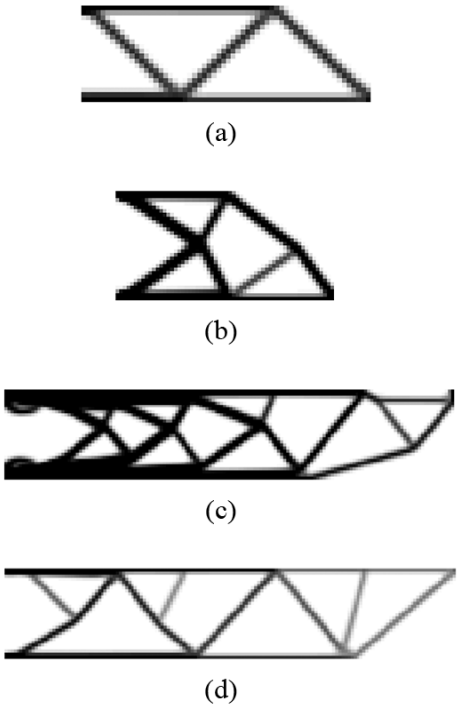


Figure 8. TO structures used as inputs for algorithm to identify nodes and elements

3.2.1. Metrics

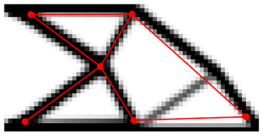
Fundamentally, the performance of each algorithm was evaluated based on how accurately it identified nodes and elements. Since the nodes in a TO structure are not visually distinct as a single point but as a region, identifying the exact ground truth node location is subjective. In order to capture this subjectivity, and to determine the most likely node location, the labeling of the nodes was performed by twenty individuals. The marked locations were then averaged to yield the ground truth. A standard deviation indicating the variation in individual marking of node locations was also computed (termed as "node location variance"). This node location variance quantified the subjectivity of the node identification in the ground truth itself and, as will be shown, correlated with the ability of the extraction algorithms to properly segment the structures. The node location variance is illustrated as green circles (radius is proportional to the variance) in Fig. 9d, 10d, 11d and 12d.

3.2.2. Results

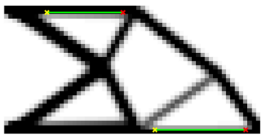
The results for TO1 are shown in Fig. 9. A visual examination clearly shows that the traversal based algorithm (NEI-TRA) outperforms the other two algorithms. As shown in Table 3 the cluster based algorithm (NEI-CL) (Fig. 9a) was unable to identify three of the elements. The hough line

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

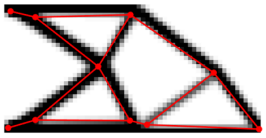
Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008



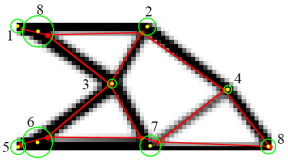
(a) NEI-CL



(b) NEI-HLT



(c) NEI-TRA



(d) NEI-TRA compared with ground truth

Figure 9. Node and element identification results for TO1

Table 3. Number of nodes and elements identified by different algorithms

Structure	Number of Nodes Identified				Number of Elements Identified			
	GT	NEI-CL	NEI-HLT	NEI-TRA	GT	NEI-CL	NEI-HLT	NEI-TRA
TO1	7	6	4	10	10	7	2	13
TO2	6	4	2	7	7	4	1	8
TO3	16	5	7	19	26	5	5	29
TO4	13	6	4	13	19	6	2	19

GT: ground truth result

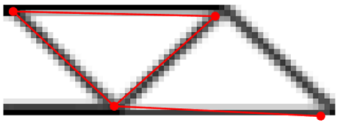
transform based algorithm (NEI-HLT) only identified two elements out of a total of ten elements (Fig. 9b). The reason for this poor performance is that the hough line transform uses the skeleton to identify straight segments. Approximations in the skeleton in the form of curves result in the hough transform being unable to detect lines. Hence, in skeletons obtained from poorly defined or curved topologies, the performance of the hough line transform is poor. With regards to NEI-TRA (Fig. 9c), there is a slight oversegmentation of one element and it thus identifies 13 elements instead of 10.

Multiple nodes were identified at some joints since the analysis of the morphological skeleton (Fig. 9e) occasionally yielded closely spaced branch points. It is worth noting that 40% of the manually grount truth sets indicated two nodes as well, similar to what the algorithm indicated. For the nodes with more node location variance in the ground truth (1, 6, 8 and 9), the algorithm results were a little farther from the ground truth. The NEI-TRA algorithm detected all the elements of the structure.

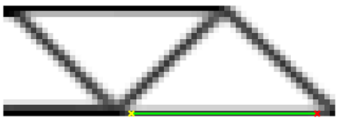
The performance of the three algorithms for TO2 is shown in Fig. 10 and Table 3. The NEI-CL algorithm detected all but two elements and identified a majority of the nodes in the structure. The NEI-HLT algorithm on the other hand detected only one element again due to approximations in

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

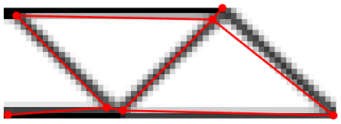
Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008



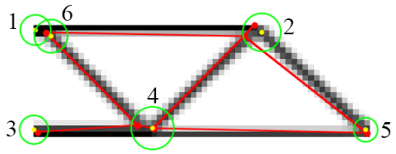
(a) NEI-CL



(b) NEI-HLT



(c) NEI-TRA



(d) NEI-TRA compared with ground truth

Figure 10. Node and element identification results for TO2

skeletonization. The NEI-TRA algorithm identified all the elements and nodes. However, due to the skeleton's structure, additional nodes were detected as seen with TO1.

A comparison with the ground truth (Fig. 10d) showed that the performance of the NEI-TRA algorithm was superior. It identified the nodes accurately in all cases. Multiple nodes along with an extra element were detected at two joints in the structure due to the skeleton having multiple branch points in the vicinity of that joint. The algorithm correctly detected only one node in the top left corner whereas 50% of the ground truth labels had two nodes at this joint.

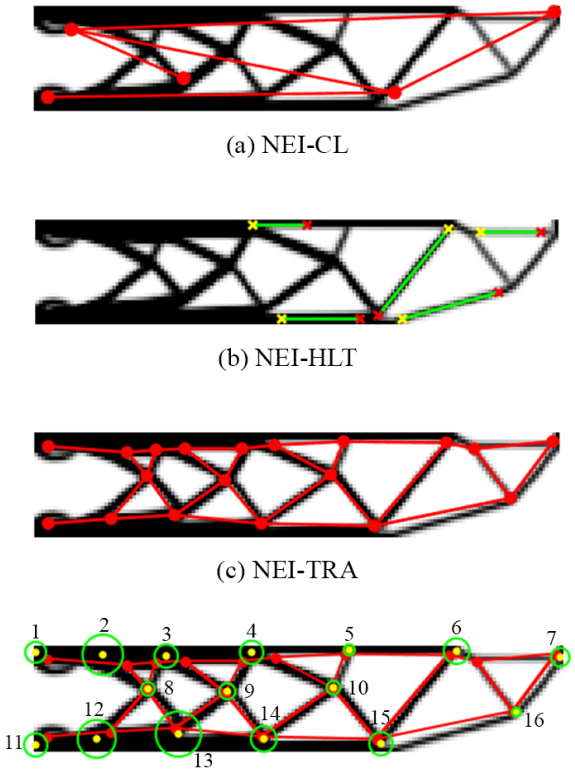
The nodes and elements identified by each of the algorithms for TO3 are shown in Fig. 11. The NEI-TRA algorithm outperforms the other two variants in this case as well. The NEI-CL algorithm performed relatively poorly due to the increased amount of material in this structure (as a result of a higher volume ratio). The algorithm erroneously identified elements which do not exist as a result. The NEI-HLT algorithm also performed poorly, with only a few elements being detected due to approximations in the skeleton. The NEI-TRA algorithm provided the most reasonable system again with a slight oversegmentation.

Comparing with the ground truth, the NEI-TRA algorithm again shows multiple nodes at some joints (Fig. 11d) due to the skeleton containing multiple branch points in those locations. Nodes with more variance in the ground truth labeling, such as nodes 2 and 12, had more distance between the ground truth node and algorithm identified node (see Table 4). In contrast, node 13 is accurately identified although the variance in the ground truth was high for this node. It is worth noting that noticeable errors were obtained at the leftmost nodes of the structure (nodes 1 and 11) since the skeleton endpoint is not exactly at the actual structure support location.

Fig. 12 shows the nodes and elements identified by the three algorithms for TO4 structure. The results show that the NEI-TRA algorithm was again the best performer. The NEI-HLT algorithm showed the poorest performance due to skeleton approximations. The NEI-CL algorithm detected

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008



(a) NEI-CL

(b) NEI-HLT

(c) NEI-TRA

(d) NEI-TRA compared with ground truth

Figure 11. Node and element identification results for TO3

Table 4. Evaluating the NEI-TRA algorithm performance against the ground truth node coordinates

Node #	Ground truth coordinates	Distance from ground truth	Variance in ground truth
1	(74.8,165)	12.1	2.68
2	(130,167)	22.2	5.46
3	(182,168)	7.28	3.13
4	(252,165)	8.97	2.88
7	(505,169)	3.61	2.41
10	(319,194)	0.83	1.81
11	(74.8,241)	11.7	3.07
12	(125,236)	13.1	5.01
13	(192,232)	4.6	5.96

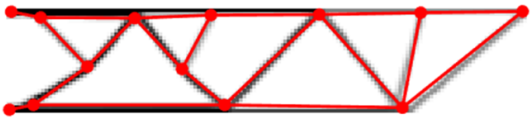
Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

Table 5. Geometric accuracy of the NEI-TRA algorithm

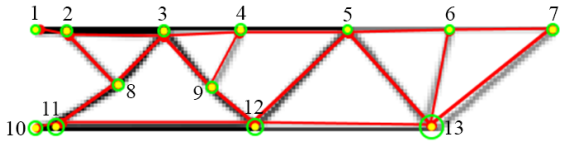
Structure	Mean coordinate distance from ground truth	Standard deviation in ground truth
TO1	9.64	4.82
TO2	9.93	6.85
TO3	6.78	3.00
TO4	2.59	1.51

(a) NEI-CL

(b) NEI-HLT



(c) NEI-TRA



(d) NEI-TRA compared with ground truth

Figure 12. Node and element identification results for TO4

about half of the elements in the structure. The nodes detected were however very close to the actual node locations from the benchmark solutions.

Since this topology's elements have less thickness, the skeleton of the structure is a very good representation of the actual structure. As a result, the NEI-TRA algorithm detects nodes and elements very accurately. Quantitatively, it can be seen that the mean error in node location for TO4 was much lower than the node location error for all other structures tested (Table 5). Hence, the solution obtained using the traversal based algorithm is very close to the ground truth solution (Fig. 12d).

3.2.3. Summary

This analysis showed that the NEI-TRA algorithm, based on depth first search traversal and k-means clustering, clearly demonstrated superior performance compared to the NEI-CL and NEI-HLT algorithms. NEI-HLT performs poorly due to approximations in skeletonization. NEI-CL erroneously identifies direction vector clusters as elements leading to falsely identified elements. This erroneous identification of clusters is a classic problem of separating signal (clusters that are actual elements) from the noise (clusters that are not elements). The NEI-TRA algorithm demonstrated a better performance

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

since its conceptual assumptions that nodes are branch points in skeleton and that nodes are connected via lines in the skeleton, do not have a signal and noise separation problem as severe as the other two algorithms. Upon identifying the NEI-TRA algorithm as the best approach, a sensitivity analysis was performed, which is presented in the next section.

3.3. Sensitivity Analysis

The NEI-TRA algorithm has five controlling parameters. A bandwidth parameter, "α", is used to check if the line connecting two nodes is a good fit for the pixels (Fig. 4, line 8). Three additional parameters are used for clustering nodes in order to eliminate similar nodes (Fig. 4, line 17), and a second bandwidth parameter "β" is used for removing redundant elements and breaking up long elements into smaller elements if the required criteria are met (Fig. 4, line 17). The effect of each parameter is explored in greater depth in this section.

3.3.1. Impact of bandwidth parameter α

6 different values of the bandwidth parameter α, from 2 pixels to 12 pixels, were tested. In the results shown, an α value of 4 was used. It was observed that the impact of this parameter on the resulting system is not significant since the main purpose of this parameter is to make the algorithm faster. Once the pixels from the traversal are determined, there are two ways of processing it. If the line connecting the end point nodes is a good fit, which is true if no pixel is more than α pixels away from a line, then an element connecting the nodes is added (Fig. 4, line 15-16). If the fit is not good, then a clustering approach is used (Fig. 4, line 9-14). If α is too small, then the clustering approach is used more frequently resulting in a slower runtime. Further, the clustering approach can determine different endpoints from the existing nodes and hence create additional nodes. However, after removing similar nodes, the final nodes and elements obtained did not change significantly as seen for TO3 (Fig. 13c).

3.3.2. Impact of the three clustering parameters

In order to remove redundant nodes, clustering is used (Fig. 4, line 17). Three parameters together control this clustering. 15 combinations of these parameters, with settings varying from 5 pixels to 30 pixels, were evaluated. Small values for these parameters cluster only the nodes which are very close to each other whereas larger values can cluster nodes which are farther apart spatially. This can be seen in Fig. 13f where higher values of these parameters cluster nearby nodes. For TO3, the impact was most noticeable since nodes are relatively close to each other. Hence, higher values of this parameter ended up clustering non redundant nodes which should not be clustered into one node. The best results were obtained with the three parameters having values of 10, 15 and 7.5.

3.3.3. Impact of bandwidth parameter β

Redundant elements are removed and long elements are broken into appropriate smaller elements using the control parameter β. For example, the top elements in Fig. 13h, which connect the leftmost top node to the rightmost top node are formed by breaking a long element that connects the leftmost and rightmost nodes into multiple smaller elements with intermediate nodes. 10 β values ranging from 5 pixels to 50 pixels were tested for this sensitivity analysis. If the value of the parameter was too large (more than 40 pixels), then the element was erroneously segmented as shown in Fig. 13h. Similarly erroneous segmentations were seen in the other topologies as well. If the parameter was too small (10 pixels or less), then some redundancy in elements persisted as illustrated for TO1 in Fig. 13g. A β value of 20 resulted in the most consistent element identification.

4. Conclusions and Future Work

This paper presents a new method of generating structural systems using a combination of topology optimization and an automated algorithm for extracting systems from a topology. The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 January 2018 doi:10.20944/preprints201801.0252.v1

Peer-reviewed version available at Designs 2018, 2, 8; doi:10.3390/designs2010008

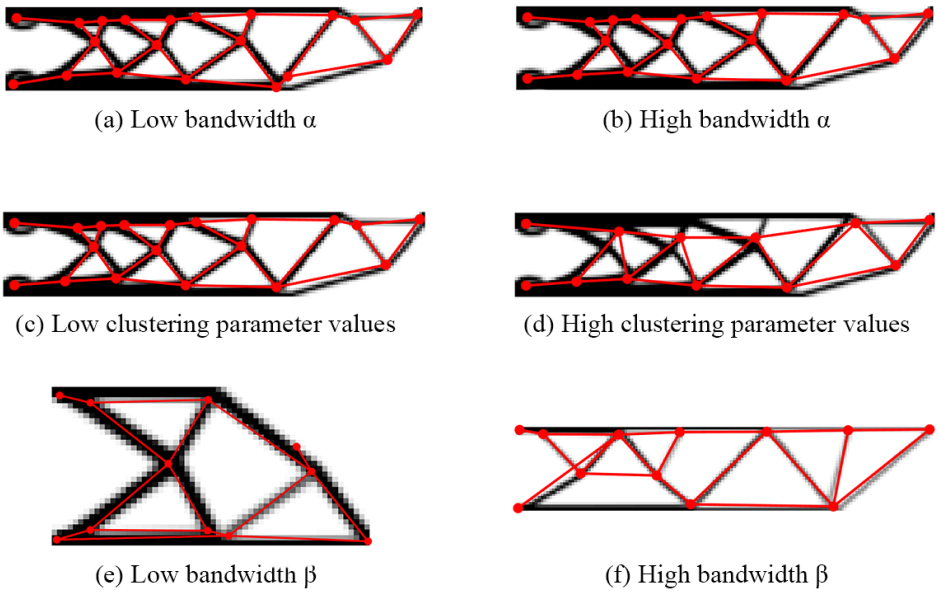


Figure 13. Impact of different parameters on the performance of the NEI-TRA algorithm

benefits of this approach are that, compared to other approaches to structural optimization, it does not require an assumed structural system for initiation. Instead, broader constraints such as loadings, geometric boundaries, and volume ratios are required instead.

From the preliminary TO system study, several conclusions can be drawn. One of the foremost observations is that structural systems derived manually through TO are efficient and comparable to optimal structures from the literature in most cases. These systems were comparable with respect to deflections. The TO-based systems resulted in more balanced member stresses compared to the benchmark solutions, which had members with very high and very low stresses. In some cases, however, the system extracted from the topology was not stable for a truss configuration, requiring manual post-processing.

Once the viability of using TO for deriving systems was verified, the three extraction algorithms were tested. The NEI-TRA algorithm demonstrated the most consistent performance and accurately identified the most nodes and elements across different scenarios. A key aspect of the extraction process presented here is the analysis of the morphological skeleton to identify nodes and their connectivity. Hence, the algorithm performs well in scenarios where the skeleton of the structure is a close representation of the actual structure. In situations where this does not hold, the performance of the algorithm deteriorates. This was clearly seen in cases where the skeletonization process was modified to yield a skeleton unrepresentative of the actual structure. The algorithm itself is dependent on several parameters. The best performance was seen when the values of the bandwidth α was 4, β was 20 and the three clustering parameter values were between 7.5 and 15. The values of these parameters will change depending on the scale of the input for the algorithms. Higher resolution inputs to the algorithm will need larger parameter values and vice versa.

One persistent issue with the NEI-TRA algorithm was the multi-node problem wherein multiple nodes were detected instead of a single node. Further research is required to remedy this issue. Another avenue for future work is to use additional structural optimization to further improve the designs generated through the topology extraction process.

Author Contributions: Achyuthan Jootoo and David Lattanzi conceived and designed the experiments. Achyuthan Jootoo developed the algorithms and programs, performed the experiments, and analyzed the data. Achyuthan Jootoo wrote the paper with reviews, significant inputs and edits by David Lattanzi.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

TO: Topology Optimization
EA: Evolutionary Algorithms
NEI-CL: Node element identification via Clustering
NEI-HLT: Node element identification via Hough Line Transform
NEI-TRA: Node element identification via node to node Traversal

1. Rajan, S.D. Sizing, Shape, and Topology Design Optimization of Trusses Using Genetic Algorithm. *Journal of Structural Engineering* **1995**, *121*, 1480–1487.
2. Ho-Huu, V.; Vo-Duy, T.; Luu-Van, T.; Le-Anh, L.; Nguyen-Thoi, T. Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme. *Automation in Construction* **2016**, *68*, 81–94.
3. Kicinger, R.; Arciszewski, T.; DeJong, K. Evolutionary Design of Steel Structures in Tall Buildings. *Journal of Computing in Civil Engineering* **2005**, *19*, 223–238. Similar to the Murawski (2000) paper; experiments and results; not much theory;.
4. Xu, T.; Zuo, W.; Xu, T.; Song, G.; Li, R. An adaptive reanalysis method for genetic algorithm with application to fast truss optimization. *Acta Mech Sin* **2009**, *26*, 225–234.
5. Tejani, G.G.; Savsani, V.J.; Bureerat, S.; Patel, V.K. Topology and Size Optimization of Trusses with Static and Dynamic Bounds by Modified Symbiotic Organisms Search. *Journal of Computing in Civil Engineering* **2017**, *32*, 04017085.
6. Rajeev, S.; Krishnamoorthy, C. Discrete Optimization of Structures Using Genetic Algorithms. *J. Struct. Eng.* **1992**, *118*, 1233–1250. Insights into the GA setup and working.
7. Hajela, P.; Lee, E. Genetic algorithms in truss topological optimization. *International Journal of Solids and Structures* **1995**, *32*, 3341–3357.
8. Gandomi, A.H.; Yang, X.S.; Alavi, A.H. Mixed variable structural optimization using Firefly Algorithm. *Computers & Structures* **2011**, *89*, 2325–2336.
9. Hasancebi, O.; Azad, S.K. Adaptive dimensional search: A new metaheuristic algorithm for discrete truss sizing optimization. *Computers & Structures* **2015**, *154*, 1–16.
10. Wu, C.Y.; Tseng, K.Y. Truss structure optimization using adaptive multi-population differential evolution. *Struct Multidisc Optim* **2010**, *42*, 575–590.
11. Goncalves, M.S.; Lopez, R.H.; Miguel, L.F. Search group algorithm: A new metaheuristic method for the optimization of truss structures. *Computers and Structures* **2015**, *153*, 165–184.
12. Bendsoe, M.P. Optimal shape design as a material distribution problem. *Struct Multidisc Optim* **1989**, *1(4)*, 193–202.
13. Zhou, M.; Rozvany, G.I.N. The COC algorithm, Part II: Topological, geometrical and generalized shape optimization. *Computer Methods in Applied Mechanics and Engineering* **1991**, *89*, 309–336.
14. Stolpe, M.; Svanberg, K. An alternative interpolation scheme for minimum compliance topology optimization. *Struct Multidisc Optim* **2001**, *22*, 116–124.
15. Bruns, T.E. A reevaluation of the SIMP method with filtering and an alternative formulation for solid–void topology optimization. *Struct Multidisc Optim* **2005**, *30*, 428–436.
16. Bendsoe, M.P.; Sigmund, O. *Topology optimization: theory, methods, and applications*; Springer Science & Business Media, 2013.
17. Deaton, J.D.; Grandhi, R.V. A survey of structural and multidisciplinary continuum topology optimization: post 2000. *Struct Multidisc Optim* **2013**, *49*, 1–38.
18. Sigmund, O.; Maute, K. Topology optimization approaches. *Struct Multidisc Optim* **2013**, *48*, 1031–1055.
19. Pereira, J.T.; Fancello, E.A.; Barcellos, C.S. Topology optimization of continuum structures with material failure constraints. *Struct Multidisc Optim* **2004**, *26*, 50–66.
20. Bruggi, M.; Duysinx, P. Topology optimization for minimum weight with compliance and stress constraints. *Struct Multidisc Optim* **2012**, *46*, 369–384.

21. Kreissl, S.; Pingen, G.; Maute, K. Topology optimization for unsteady flow. *Int. J. Numer. Meth. Engng.* **2011**, *87*, 1229–1253.
22. Zhou, S.; Li, Q. Computational design of multi-phase microstructural materials for extremal conductivity. *Computational Materials Science* **2008**, *43*, 549–564.
23. Wang, Q.; Lu, Z.; Zhou, C. New Topology Optimization Method for Wing Leading-Edge Ribs. *Journal of Aircraft* **2011**, *48*, 1741–1748.
24. Zhu, J.H.; Zhang, W.H.; Xia, L. Topology Optimization in Aircraft and Aerospace Structures Design. *Archives of Computational Methods in Engineering* **2016**, *23*, 595–622.
25. Xie, Y.M.; Steven, G.P. A simple evolutionary procedure for structural optimization. *Computers & structures* **1993**, *49*, 885–896.
26. Rozvany, G.I.N. A critical review of established methods of structural topology optimization. *Struct Multidisc Optim* **2008**, *37*, 217–237.
27. Bendsoe, M.P.; Kikuchi, N. Generating optimal topologies in structural design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering* **1988**, *71*, 197–224.
28. Duda, R.O.; Hart, P.E. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Commun. ACM* **1972**, *15*, 11–15.
29. Otsu, N. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics* **1979**, *9*, 62–66.
30. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*; Pearson, 2007.
31. Witten, I.; Frank, E.; Hall, M. *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed.; Morgan Kaufmann: Burlington, MA, 2011.
32. Perez, R.E.; Behdinan, K. Particle swarm approach for structural design optimization. *Computers & Structures* **2007**, *85*, 1579–1588.
33. Kaveh, A.; Kalatjari, V. Size/geometry optimization of trusses by the force method and genetic algorithm. *Z. angew. Math. Mech.* **2004**, *84*, 347–357.
34. Rahami, H.; Kaveh, A.; Gholipour, Y. Sizing, geometry and topology optimization of trusses via force method and genetic algorithm. *Engineering Structures* **2008**, *30*, 2360–2369.
35. Sigmund, O. A 99 line topology optimization code written in Matlab. *Struct Multidisc Optim* **2001**, *21*, 120–127.
36. Risa2D. Risa-2D Educational, 2017.