Automation in Construction 39 (2014) 59–69

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Automatic innovative truss design using grammatical evolution

Michael Fenton [a,*], Ciaran McNally [a], Jonathan Byrne [a], Erik Hemberg [b], James McDermott [a], Michael O'Neill [a]

[a] UCD, Ireland
[b] MIT, United States

ARTICLE INFO

Article history:
Accepted 29 November 2013
Available online 4 January 2014

Keywords:
Structural optimization
Genetic programming
Evolutionary computation
Grammatical evolution
Truss design
Computer aided design

ABSTRACT

Truss optimization in the field of Structural Engineering is a growing discipline. The application of Grammatical Evolution, a grammar-based form of Genetic Programming (GP), has shown that it is capable of generating innovative engineering designs. Existing truss optimization methods in GP focus primarily on optimizing global topology. The standard method is to explore the search space while seeking minimum cross-sectional areas for all elements. In doing so, critical knowledge of section geometry and orientation is omitted, leading to inaccurate stress calculations and structures not meeting codes of practice. This can be addressed by constraining the optimisation method to only use standard construction elements.

The aim of this paper is not to find fully optimized solutions, but rather to show that solutions very close to the theoretical optimum can be achieved using real-world elements. This methodology can be applied to any structural engineering design which can be generated by a grammar.

1. Introduction

A major part of engineering design is the process of satisfying hard constraints. In structural engineering, topology optimization is known as the science of "optimal layout theory" [1]. It allows engineers to design highly optimized structures—maximizing material efficiency while minimizing waste and reducing material cost. This allows for structures that are stiff yet lightweight, which can lead to savings in terms of resources and cost [2–4]. Engineering optimization is an important problem as minor savings in weight or cost on a small scale can have larger implications when extrapolated over a larger design or project.

The theory of topology optimization in structural engineering states that it is possible to create a structurally "perfect" design with both optimal shape topology and member sizes by both rearranging the topological layout of the members and by varying the sizes of those individual members [4]. All members in the design should have similar high states of stress at, or close to (but not exceeding) the limits of the material as specified by design codes of practice and manufacturer specifications. This eliminates redundancy, minimizes material usage and creates a more economical design. This is most usually achieved by minimizing the cross-sectional area of each structural member, which consequently minimizes the overall weight of the entire structure.

Genetic Programming (GP) has been shown to be routinely capable of achieving human-competitive performance in a number of real-world scenarios [5–7]. Evidence of an increase in use of GP in industry can be found in the increasing number of patent applications using GP [8]. GP is particularly well suited for engineering tasks for a number of reasons, including its ability to handle multiple conflicting objectives [14,15] and its capacity to optimize both the structure and the contents of that structure in parallel [14]. Since the solution is unknown (due to incomplete information or theory), GP is one method in particular which can uncover its optimal structure/topological form [1,11,16,17]. Sizing optimization is similar in theory to solving a simple linear equation: the form (topology) is known, and the variables (member sizes) are increased/decreased to fit. It is therefore possible to use both GP and linear optimization as a hybrid approach towards topology optimization.

Grammatical Evolution (GE) is a version of GP that uses a formal grammar [9,11,12], allowing the user to easily embed domain knowledge (such as structure boundary conditions, loading conditions, and basic form including span and depth), and to generate output in any language [11]. Both GE and topology optimization represent the cutting edge of both GP and structural engineering fields respectively.

This paper introduces a new method of topology optimization: Dual Optimization in Grammatical Evolution (DO-GE). While existing structural optimization methods in GE [14,16,18] primarily focus on a structural topology scale (optimization of the structural layout), optimization of individual element sizes is also possible [1,19,20]. The combination of both topology and sizing optimization is established [1,16,17,19,29], but the use of both standard construction elements and compliance to design codes of practice in the process is novel. Standard practice is to optimize element sizings by specifying the required cross-sectional area. While this gives theoretically optimized results, the output is of little use to structural engineers as in practice

* Corresponding author.
E-mail address: michaelfenton1@gmail.com (M. Fenton).

0926-5805/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.autcon.2013.11.009

trusses are constructed using structural elements with preset cross section and geometry. This highlights a fundamental weakness in traditional sizing optimization methods: by omitting knowledge of section geometry and orientation, it is not possible to include accurate buckling calculations as a constraint for structural design and thus structures cannot be designed according to standard codes of practice. The approach presented in this paper addresses this deficiency by allowing for any number of standard construction elements to be specified for any elements within a design, leading to code-compliant construction-ready designs which truly represent their evolved form.

The DO-GE approach has a number of advantages over a two-stage approach of optimizing topology and element sizes separately. With a single stage approach, a large number of designs can be assessed in a relatively short space of time, whereas a two-stage approach would be slower and would fail to allow for interactions between structural topology and element sizes parameters. A single-stage approach also allows for real-time analysis of both design variables and structural properties of the individuals as evolution progresses.

Section 2 will begin with a summary of related research in this area, along with a description of the DO-GE method, including our approach to design generation and analysis. Section 3 compares and contrasts recent research methods with the DO-GE method using examples from the literature, and a discussion on the implications of those results is presented in Section 4. Finally, our conclusions and suggestions for future work are presented in Section 5.

2. Evolutionary approaches to structural engineering

The use of computers in structural design has been growing rapidly in recent years. The advent of techniques such as Topology Optimization [1] and Evolutionary Computation (EC) [17] has heralded engineering applications ranging from analog circuit design [5] to the design of structures such as shelters [30] and bridges [14].

2.1. Engineering design approaches

A recent survey of the applications of evolutionary computation in structural engineering design [17] has found the most difficult aspects of the design to be i) appropriate representation of the engineering system itself and ii) finding a suitable evaluation function. Appropriate representation of the engineering system is possible using the relevant design codes of practice [2,3,30,32]. In the case of structural design this entails creating boundary conditions (supports and loading), material limits (usually expressed as stress or strain) and design limits (deflection). The use of the Finite Element method of structural analysis [4] as a fitness function has been proven useful [14,18], and it enables the EC program to assess and evaluate individuals based on the results of a finite element analysis.

Both Murawski et al. [41] and Kicinger et al. [44] successfully used Evolutionary Computation (EC) methods to evolve steel wind bracings for tall structures, based on Grierson and Cameron's SODA method

```
<S> ::= <program>{}<call>

<program> ::= def program(chromosome_b):{<init>{}<constants>{}<define_funcs>{}<make_all>{}<return>{}}

<init> ::= truss_graph = graph.graph(){}

<constants> ::= span = <span>{}depth = <depth>{}r = <r>{}genome = chromosome_b

<define_funcs> ::= <chord>{}<cross_brace>{}

<chord> ::= def chord(r):{top_chord = []{}bottom_chord = []{}for i in range (r+1):{bay_span =
i*span/(2*r){}tnode, bnode = [0,bay_span,depth],
[0,bay_span,0]{}top_chord.append(tnode){}bottom_chord.append(bnode)}return top_chord, bottom_chord}

<cross_brace> ::= def cross_brace(r):{edge_list = []{}top_ids = []{}bottom_ids = []{}chords =
chord(r){}top_chord = chords[0]{}bottom_chord = chords[1]{}<truss_type>for i, edge in
enumerate(edge_list):{truss_graph.add_edge(edge[0], edge[1], material=genome[i])}return edge_list}

<connection_type> ::= <howe>|<pratt>|<howe>|<fully_braced>|<warren>|<modified_warren>|<vierendeel>

<pratt> ::= connect_pratt_truss{}

<howe> ::= connect_howe_truss{}

<fully_braced> ::= connect_fully_braced_truss{}

<warren> ::= connect_warren_truss{}

<modified_warren> ::= connect_modified_warren_truss{}

<vierendeel> ::= connect_vierendeel_truss{}

<r> ::= 2|3|4|...|19|20

<span> ::= 24000

<depth> ::= <span>/10|<span>/11|<span>/12...|<span>/24|<span>/25

<make_all> ::= edge_list = cross_brace(r){}mirror_graph = truss_graph. mirror(truss_graph){}

<return> ::= return mirror_graph{}

<call> ::= program(chromosome_b)
```

Fig. 1. A sample truss grammar.

```
def program(chromosome_b):
    truss_graph = graph.graph()
    span = 24000
    depth = span/19
    r = 9
    genome = chromosome_b
    def chord(r):
        for i in range (r+1):
            bay_span = i*span/(2*r)
            tnode, bnode = [0,bay_span,depth], [0,bay_span,0]
            top_chord.append(tnode)
            bottom_chord.append(bnode)
        return top_chord, bottom_chord
    def cross_brace(r):
        chords = chord(r)
        top_chord = chords[0]
        bottom_chord = chords[1]
        connect_vierendeel_truss()
        for i, edge in enumerate(edge_list):
            truss_graph.add_edge(edge[0], edge[1], material=genome[i])
        return edge_list
    edge_list = cross_brace(r)
    mirror_graph = truss_graph.mirror(truss_graph)
    return mirror_graph
program(chromosome_b)
```

Fig. 2. A sample derived truss program.

[43], while Kicinger et al. [42] incorporated multi-objective optimization in minimizing both structure self weight and displacement. Other well established optimization methods include Artificial Neural Networks (ANN) (with model induction capabilities) [33] and Particle Swarm Optimization (PSO) methods. However, the format of ANNs present difficulties when encoding solutions, and their outputs can be difficult to understand, thus they are not as flexible as the GP approach in the structural engineering field. PSOs, on the other hand, have proven particularly efficient in solving sizing optimization problems. Li et al. [34] proposed a heuristic PSO, Kaveh and Talatahari [35] combined a heuristic PSO with an ant colony strategy for an efficient hybridized approach, while Luh and Li [28] used PSO for full topological optimization of truss structures.

2.2. Truss Topology Optimization in Design

The field of Topology Optimization in Design (TOD) has seen rapid expansion in the last few years with improvements in computational power and efficiency [17]. Traditional TOD methods take a bit-array approach where material is added to or removed from a solid mass to obtain the most optimal topological arrangement of the material and void for its particular application [1]. The field can be broken down into two parts: Continuum and Discrete TOD.

Continuum TOD is similar in principle to the finite element method of structural analysis [4] in that the system is assumed to be continuous and as such can be discretized into smaller elements which, when optimized, can be extrapolated to account for the overall design [21,22]. Popular methods include the Principle Stress Line [23], Evolutionary Structural Optimization (ESO) [21,22,24] and Bidirectional Evolutionary Structural Optimization (BESO) [20,21] methods, which have proved highly effective in the area of architectural design [24] among others. While this approach has been repeatedly proven to be computationally and structurally more efficient than other forms of truss design [1,24], it is implicitly cost-ineffective to manufacture as non-standard elements, forms, and construction methods are required. A system that instead uses standard construction elements has the potential to be applied to a far wider array of applications in the real world environment as existing fabrication technologies and construction practices can still be used, requiring no bespoke industries.

The discrete TOD approach is much closer to traditional beam-truss design in that it looks at the design of the complete structure, with particular focus on element connectivity. This method lends itself especially well to truss design, where appropriate connectivity of the members is paramount. GA's have been used extensively in discrete TOD to evolve trusses, and numerous approaches have been identified [16,25–27]. Existing methods include the representation of the truss as a combination of triangles [16], and the topological bit approach [25,26]. More recent methods have successfully used combinations of Evolutionary Algorithms and approximate gradients [29] in truss topology optimization.

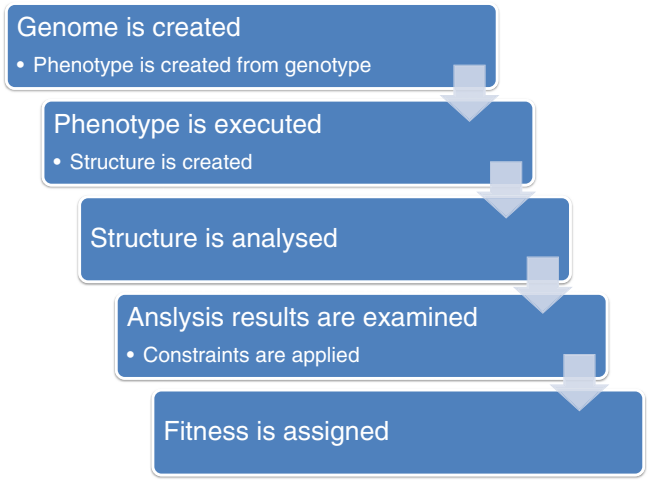

Fig. 3. Flowchart of the evolutionary process.

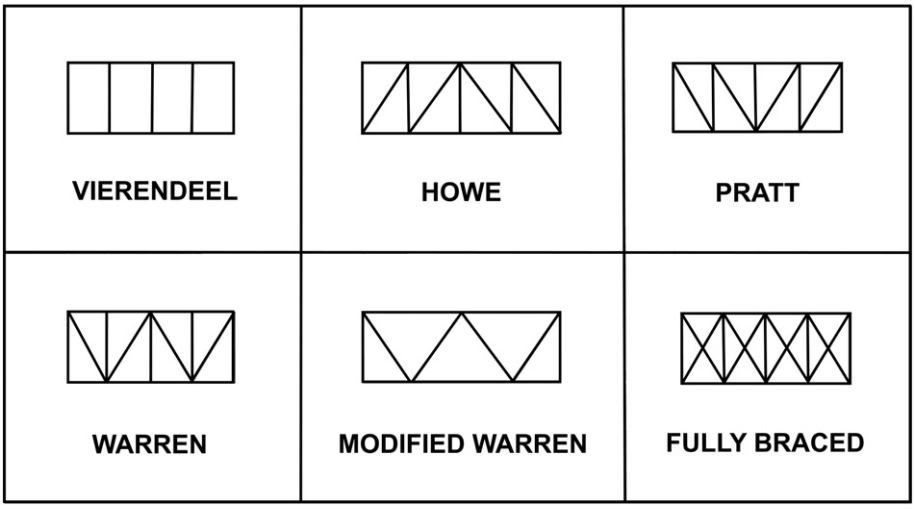

Fig. 4. Basic truss designs.

2.3. The GE method and its comparative advantages

GE is a grammar-based form of GP in which the grammar provides a representation in which one can encode the structure of the solution [10,11,13]. A grammar (Fig. 1) defines a derivation via a series of rules: non terminals on the left, and a number of production choices on the right. Production choices can be terminals (i.e. no further choices can be made), non-terminals (leading to more production choices), or combinations of both. It is also possible for grammars to be recursive by setting a production choice equal to its own non-terminal, however the current study is scope-limited to match the characteristics of other methods, leading to a non-recursive grammar.

In essence, GE takes a grammar in Backus-Naur Form (BNF) [10] beginning from the start symbol and maps a variable-length integer array of genes (a chromosome) to a phenotype (an executable program, Fig. 2) by expanding non terminals from left to right in the derivation string [9]. The expansion of a non-terminal is determined by the value of the current gene modulo the number of production choices in the current rule. The mapping terminates when there are no non-terminals left.

GE's advantages over a regular GA lie not only in the variable length nature of the genotype integer array, but also in the power of the mapping process itself. The use of a formal grammar allows complex programs to be derived with ease, and one can effectively embed all manner of useful domain knowledge in the grammar constraining the form of the generated solution [9,12]. Genetic operators including tournament selection, replacement, one-point crossover, and per-gene mutation are similar or identical to those of a standard GA.

When comparing the use of grammars [9] in structural optimization to pre-existing methods such as ANNs or PSO, grammars present a much more advantageous approach as the output from the search process is human-readable and can be readily altered to suit any application, which allows for easy and quick analysis of results across a variety of platforms. It is also possible to add numerous constraints and bias about the structure into the grammar itself [9,11,14,30], all of which reduce the search space to a more manageable size. Integration with existing methods and systems is also one of our goals. Our implementation has the highly useful option of allowing the user to save designs as different file types such as those compatible with AutoDesk's .dxf extension, which allows for further manipulation of designs in an engineering design environment.

Comparison with traditional TOD approaches (both Continuum and Discreet fields) yields further advantages. One drawback with both TOD approaches is that substantial computational effort is required for large trusses [26]. The approach described in this paper has the significant advantage that the physical dimensions of the design do not have a bearing on the computational effort required to evolve the design. In contrast to previous GA methods, the node locations in the design search space are not fixed, but have a high degree of variability. This combination of evolution of element size, connectivity, and node location allows for far more complex designs to be evolved.

2.4. The DO-GE method

DO-GE differs from regular GE by utilizing two separate chromosomes simultaneously. DO-GE creates two separate integer array chromosomes: the A-chromosome, which governs the topological form of the structure, and the B-chromosome, which assigns material section sizes to each individual edge in the individual. Chromosome A operates in the normal GE fashion, controlling the derivation of the grammar which details the layout of the trusses (as explained in Section 2.3 above). Chromosome B is passed in as an argument to the derived program (as shown in Fig. 2), which itself creates a graph object through its execution. Each graph edge is assigned a material id from a corresponding gene in Chromosome B (passed in as an argument to the derived program). The list of edges is arranged in the order in which the edges are generated, meaning that if the edge order changes, the solution changes. This is not a concern, however, as the creation of edges is explicitly controlled. While Chromosome A is fixed in length, a variable-length chromosome is required for Chromosome B due to the variable nature of the number of edges in each individual as the truss type changes. Structural analysis of individuals is then carried out using the free open-source finite element modeling program SLFFEA [37] (Fig. 3). Previous work [14,18] has shown this method to be both reliable and fast in analyzing any engineering structure produced by a grammar.

Fig. 5. Sample truss design evolved using DO-GE. Experimental variables are: span: 24 m, style = Vierendeel, height = span/19, r = 9.

Table 1
Material properties.

	10 bar truss		17 bar truss	
	Tata steel sections	Aluminium solid sections	Tata steel sections	Steel solid sections
Name	Tata	Mat_10	Tata	Mat_17
Section type	CHS	CHS	CHS	CHS
Section sizes	157 standard sizes; diam. from 0.838 to 20 in.	350 sections; CSA from 0.1 to 35 in^2, increments 0.1 in^2	157 standard sizes; diam. from 0.838 to 20 in.	350 sections; CSA from 0.1 to 35 in^2, increments 0.1 in^2
Young's modulus (ksi)	30,458	10,000	30,458	30,000
Density (lb/in^3)	0.285	0.100	0.285	0.268
Max tensile stress (ksi)	24.66 (t < 1.6 in.); 22.48 (t > 1.6 in.)	25	24.66 (t < 1.6 in.); 22.48 (t > 1.6 in.)	20
Max comp stress (ksi)	Manufacturers limits	25	Manufacturers limits	20

The regular genetic search operators used by GE – mutation and crossover – are modified to accommodate the use of two separate chromosomes. First, randomized pairs of parents are selected from the parent population using tournament selection. For each pair, either Chromosome A or Chromosome B is chosen, and then crossover is performed on that chromosome of both parents, creating a pair of children (the other chromosome is copied from each parent unchanged). Once the child population is fully created, mutation is applied to either Chromosome A or Chromosome B for each individual child (choice-independent of the crossover stage), creating the next parent population.

DO-GE has the added capability of being able to detect redundant members in its truss designs. If a particular solution contains a member with zero stresses (i.e. all member stresses xx, xy, zx, and moments xx, yy, zz are zero), that individual is then re-analyzed without the presence of that particular edge. If the fitness is improved upon (i.e. if all constraints are still within their limits, and the overall structure self-weight is lowered), then that member is omitted from the solution. It must be noted that at present this is not an evolutionary feature; it doesn't create a change in the chromosome, rather it is a measure that is performed within the fitness function. This is an example of a post-processing measure, any number of which can be run as part of the fitness function after the DO-GE mapping has completed.

DO-GE encourages population diversity by removing individuals from the child population which either match parent individuals or other children. A number of checks are performed to ensure that only exact duplicates are removed from the child population:

i. The fitness is checked
ii. Both Chromosomes are checked
iii. The phenotype is checked (locations of all nodes and edges).

This multi-level checking process catches all possible duplicates, including any individuals that may occur due to many-to-one mapping. This feature is not possible in this study, but possible with certain recursive grammars whereby different chromosomes will generate the same phenotype.

2.4.1. Truss topology optimization

A truss grammar was built to design a variety of simple steel trusses based on existing designs. Design constants were a span of 24 m and a Universally Distributed Load (UDL) of 107.9 kips (480 kN). The grammar uses six basic truss designs commonly used in structural engineering practice, as shown in Fig. 4. These designs are represented as the production choices of the non-terminal <connection_type> in the grammar (Fig. 1).

There are also additional rules which allow the grammar to further adapt the "macro" design of the truss. These include the rule <r>, describing the number of bays in each truss (number of times a design feature is repeated, ranging from 4 to 40). Symmetry may be exploited and a range of 2 to 20 employed. The rule <depth> is used to set the depth of the truss, which can vary between span/10 and span/25.

A simplified truss grammar example is presented in Fig. 1. It shows the basic operators and moderators and their relation to one another within the grammar itself. The grammar works by defining the top and bottom chords of the truss, and then connecting them in accordance with the variable type of truss selected by the grammar; the function "cross_brace()" differs for each truss type. Three non-terminals are defined by Chromosome A: <r>, the range or number of bays of the truss; <depth>, the depth/height of the truss; and <connection_type>, the design of the truss. Each time the grammar generates a new individual, production choices are applied to each non-terminal. When the derived program is created, these production choices are set and an executable program is created which defines the structure. Chromosome B is then passed in as an argument to the derived program (it is not used during the process of grammatical derivation), allowing individual section sizes to be applied to different members in the structure.

A simplified derived program example is presented in Fig. 2 (derived from the grammar presented in Fig. 1). A truss is created using the "cross_brace" function and the nodes are then connected to corresponding top and bottom chord nodes using whichever non-terminal connection type the grammar has selected; in the presented instance creating the "box" style truss known as the "Vierendeel" design. A graphic representation of the output of this program can be seen in Fig. 5.

2.4.2. Material selection and evolution in GE

In order to reduce the initial size of the search space (without compromising later evolution) and to accelerate the initial search process, the Chromosome B of each individual in the first population is seeded with uniform genes throughout. This means that each individual in the first generation has a single material applied across all edges, resulting in a far larger number of initial "fit" individuals upon which further evolution is based.

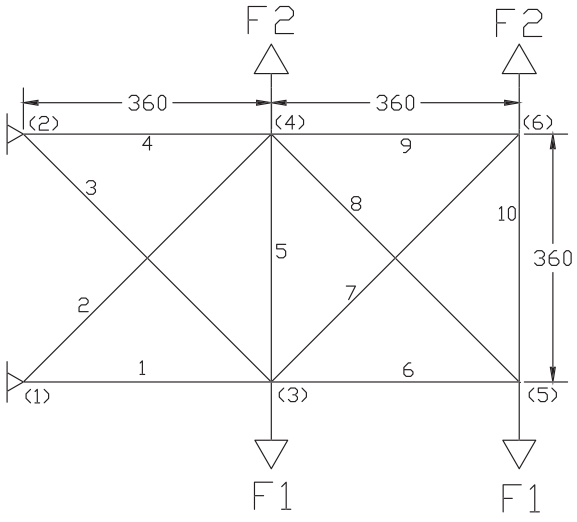

Fig. 6. 10-bar truss problem.

Table 2
10-bar truss problem: Evolved minimum cross-sectional areas for load case 1 (in^2).

Element	Li et al. [30]	Kaveh & Talatahari [31]	GE		DO-GE	
			Using Tata CHS sections	Aluminium solid sections	Using Tata CHS sections	Aluminium solid sections
1	30.704	30.307	8.944	30.5	8.680	29.5
2	0.1	0.1	0.237	0.2	Redundant	Redundant
3	23.167	23.434	7.766	23.8	7.766	23.6
4	15.183	15.505	6.510	17.4	5.828	16.8
5	0.1	0.1	0.237	0.1	Redundant	Redundant
6	0.551	0.5241	0.237	0.2	Redundant	Redundant
7	7.46	7.4365	4.588	7.7	4.588	6.1
8	20.978	21.079	7.766	23.1	7.766	21
9	21.508	21.229	6.526	21.9	7.239	22.8
10	0.1	0.1	1.533	0.1	Redundant	Redundant
Weight (lb)	5060.92	5056.56	5390.27	5287	5102.05	5056.88

Depending on the number of materials supplied for possible selection (n), DO-GE assigns each gene in the chromosome an integer value ranging between 0 and (n-1). At present, element sizes are taken from Tata Steel charts for S355 Hot-Finished Circular Hollow Sections (CHS) [36]. There are 157 variations of steel members on the list with diameters ranging from 0.838 in (21.3 mm) to 20 in (508 mm). In addition the wall thickness can also vary and any of the 157 possible steel sections can be applied to any element in the truss. The chromosome assigns an integer value from 0 to 156 (representing material ids 1 to 157) to each individual element in the structure, which corresponds to the index of the material on the list. Required material properties include the cross-sectional area, mass per meter, second moments of area I_x and I_y, and section thickness. It is possible to include any number of material sections in the materials list, so long as the five necessary variables are provided. The program automatically increases or decreases the range of the chromosome variables based on the length of materials list provided.

2.4.3. Fitness function constraints

A number of constraints within the fitness function are placed on the individuals to ensure only appropriate designs are included in the population. These constraints include limits on deflection, stress (tensile & compressive) and Euler buckling loads for compression members. While these constraints are related to the fitness function, they can be used to modify the selection pressure for each individual. Only a single objective (the self weight of the structure) is passed through as the final fitness value (this method is used by the vast majority of texts cited in this paper and is generally accepted as the standard optimization goal in structural design). The use of a multi-objective optimizer in this particular instance is unnecessary, as any parameters other than weight that might need to be optimized are merely constraints that must be imposed on the design (unlike [42] where minimizing horizontal structural displacement is a design priority). For example, there is no quantifiable engineering benefit from imposing a vertical deflection limit of 10 mm when building design codes might allow a deflection of 50 mm. In the search for minimum structure weight, it is desirable for all constraints to be at their limits in order to find the lightest possible structure. If a constraint is not at its limit, an improvement can be made.

Multiple objective optimization capability is possible with DO-GE, and indeed previous iterations of the program [14] have successfully implemented the NSGA-II algorithm [15] in handling up to three conflicting objectives. However, in this instance only a single optimization objective was deemed necessary, as once all constraints have been satisfied, there is little or no performance improvement in trying to further minimize them.

In the case of deflection, a limit of (span/250) is imposed over the entire structure. The positions of each node before and after loading are recorded, and if the total deviation of any node after loading is greater than the limit of (span/250) then that structure is considered to have failed in deflection [2–4] and a default fitness of 1,000,000 is applied.

Tensile stress limits are based on the relevant design codes [2,31,32]. A maximum tensile limit of 24.656 ksi (170 MPa) is applied to all members with a thickness of less than or equal to 1.575 in (40 mm), while a limit of 22.481 ksi (155 MPa) is applied to all members with a thickness of greater than 1.575 in (40 mm). If any element fails in tension, a default fitness is applied.

Compressive limits on the material are based on the material manufacturer's specifications [32,36]. Compressive resistance limits are given for effective lengths of individual members, and are a function of element size, geometry and end fixing conditions (i.e. fixed-fixed, fixed-pinned, pinned-pinned); these limits are applied to the appropriate elements in the structure and as with the two previous constraints, if any element fails in compression a default fitness is applied. Likewise, the use of Euler buckling limits for elements in compression ensure

Table 3
10-bar truss problem: evolved minimum cross-sectional areas for load case 2 (in^2).

Element	Li et al. [30]	Kaveh & Talatahari [31]	GE		DO-GE	
			Using Tata CHS sections	Aluminium solid sections	Using Tata CHS sections	Aluminium solid sections
1	23.353	23.194	7.239	24.7	8.184	23.40
2	0.100	0.100	0.578	0.1	Redundant	Redundant
3	25.502	24.585	7.766	29.0	8.184	24.80
4	14.250	14.221	7.285	15.4	5.084	15.00
5	0.100	0.100	0.831	0.3	Redundant	Redundant
6	1.972	1.969	1.721	2.0	1.659	2.00
7	12.363	12.489	5.828	11.1	7.239	9.50
8	12.894	12.925	6.510	14.1	4.433	14.30
9	20.356	20.952	7.239	20.7	6.510	20.60
10	0.101	0.101	0.237	0.1	0.237	0.10
Weight (lb)	4677.3	4675.8	5452.7	4919.5	5016.3	4612.8

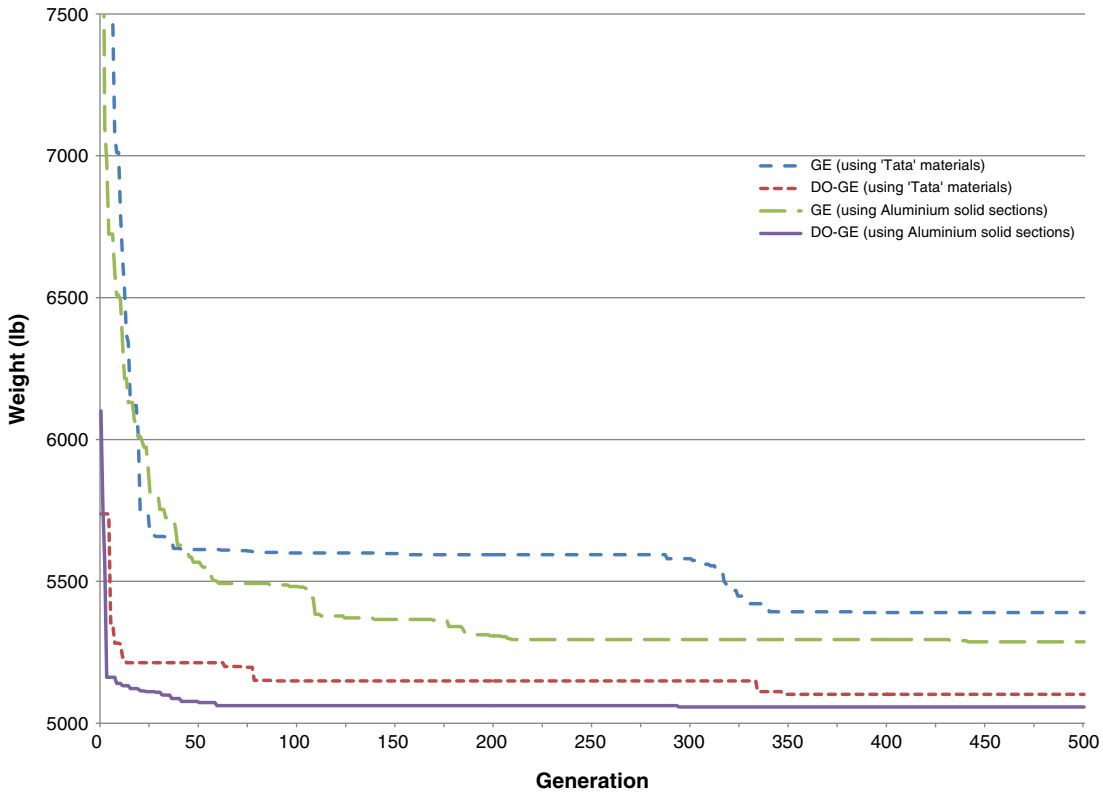

Fig. 7. 10-bar truss optimization: load case 1.

that designs featuring elements prone to buckling will be penalized. The application of these constraints results in high-displacement, over-stressed designs being removed from the population.

3. Numerical examples

The aim of this paper is not to find fully optimized solutions for absolute minimum required cross-sectional areas in standard structures; the literature contains numerous examples of efficient processes for achieving this. Instead, the focus of this paper is to show that real-world solutions very close to the theoretical optimum can be achieved using standard construction elements. Benchmarking tests are completed for common truss optimization problems against popular or recently published methods to demonstrate the effectiveness of using GE as a sizing optimizer in this manner. These problems are then further developed to demonstrate the full dual optimization capabilities of DO-GE, whereby the possibility of removing redundant members is explored.

Two commonly used problems are analyzed from selected papers: a 10-bar cantilever truss and a 17-bar cantilever truss [28,34,35,38]. In these papers imperial units were used to describe the problem; for the sake of comparison they are also used in this paper. All experiments were run with two different sets of materials, the properties of which are described in Table 1. Deflection was universally limited to 2 in. in all cases. After extensive testing to find the most suitable parameters, experimental evolutionary variables were set at:

Population Size: 500
Generations: 500
Mutation: 1%
Crossover: 75%
Generational Replacement

3.1. Sizing optimization: 10-bar cantilevered truss

Luh and Lin [28], Li et al. [34], and Kaveh and Talatahari [35] proposed solutions for a 10-bar planar truss sizing optimization problem shown in Fig. 6.

Two load cases were tested:

1. F1 = 100 kips
   F2 = 0
2. F1 = 150 kips
   F2 = 50 kips

Tables of recent research solutions, including overall structure weight and individual member cross-sectional areas, are presented in Tables 1 and 2 for load cases 1 and 2 respectively. It should be noted that the research results presented by Li et al. and Kaveh and Talatahari use aluminium as the design material; the research presented in this paper utilizes standard steel sections. Using the Tata material set, the best weights using GE are 5390.3 lb for load case 1 and 5453.7 lb for load case 2. These evolved weights are quite a bit higher than the best achieved weights from previous works, with load case 1 being 333.7 lb heavier than that of Kaveh and Talatahari and load case 2 being 776.9 lb heavier than the previous best. However when the GE algorithm makes use of the aluminium solid sections set of materials, minimum weights were

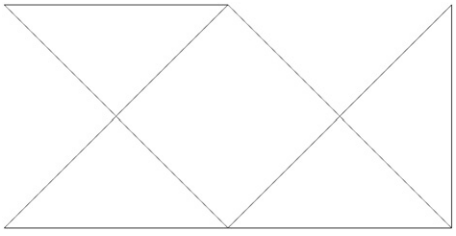

Fig. 8. 10-bar truss problem: evolved minimum topology-stable structure.

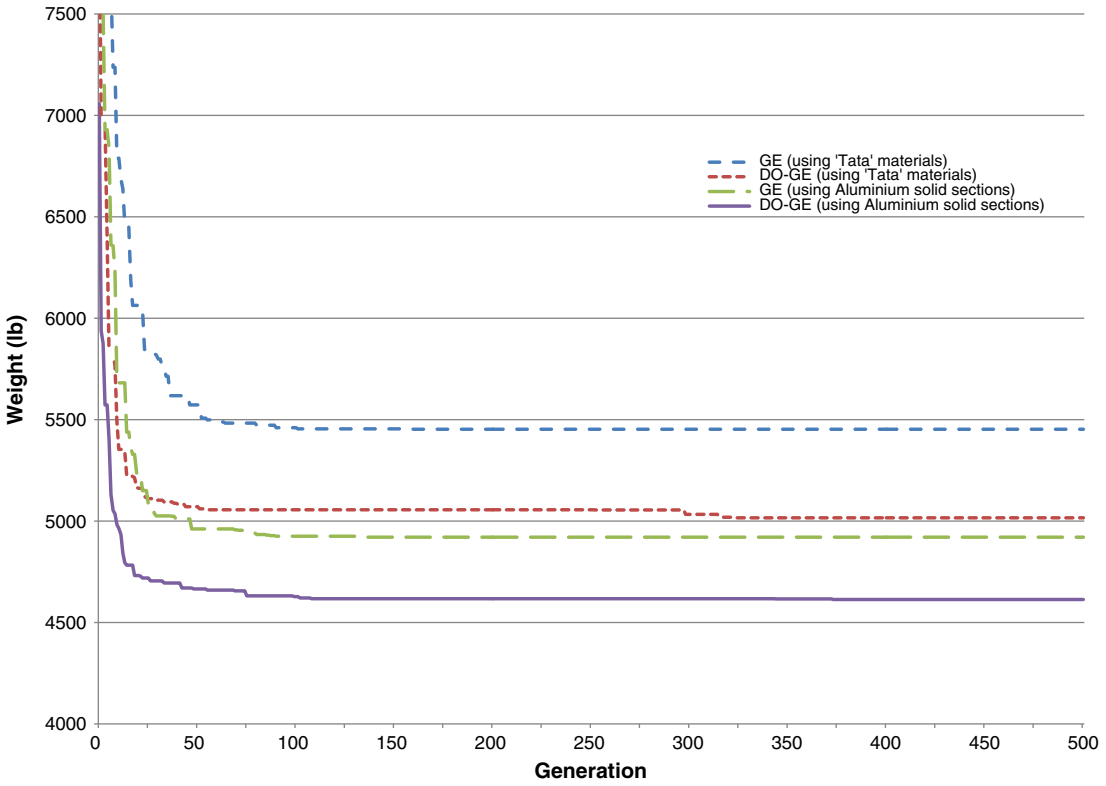

Fig. 9. 10-bar truss optimization: load case 2.

achieved of 5287 lb for load case 1 and 4919.5 lb for load case 2. These are far closer to the optimum, and exceeding the results of Kaveh and Talatahari by 45.5 lb for load case 1 and 243.75 lb for load case 2. The implications of these results are discussed in Section 4.

3.2. Dual sizing and topology optimization: 10-bar cantilevered truss

For the second part of the experiment, the 10-bar cantilevered truss problem was solved allowing access to the full dual optimization capabilities (simultaneous topology and sizing) of the GE program. The results of the evolutionary runs, shown in Tables 2 and 3, were found to be much closer to the true optimum for both load cases.

With load case 1, the program evolved structures with the minimum required materials. This resulted in the removal of elements 2, 5, 6 and 10 from the structural topology, leaving a lightweight, rigid truss which matches the best minimum weight achieved by Kaveh and Talatahari [35]. With the use of the Tata material set, the best achieved solution was 5102.1 lb, 288.2 lb lighter than the best solution obtained by GE using pure sizing optimization. A graph of the evolutionary runs for load case 1 comparing evolution of the structure using the Tata and aluminium solid sections materials sets using both GE and DO-GE is presented in Fig. 7.

For load case 2, the most optimal topology would involve the removal of elements 2, 5 and 10. However, the removal of elements 2 and 10 together would create a mechanism (a kinematically unstable structure), whereby element 6 would be able to rotate freely around node 2. In order to avoid this, an allowance has been made within the program to ensure that only dynamically stable configurations (i.e. no mechanisms) can be evolved; each element requires at least one connection per node (Fig. 8).

Using the Tata material set, the best achieved solution was 5016.3 lb, 436.4 lb lighter than the best single optimization solution by DO-GE. With the use of aluminium solid sections, the best achieved solution was actually better than that achieved by Kaveh and Talatahari [35], at 4612.8 lb (Table 3). A graph of the evolutionary runs for load case 2 comparing evolution of materials 1 and 2 using both GE and DO-GE is presented in Fig. 9.

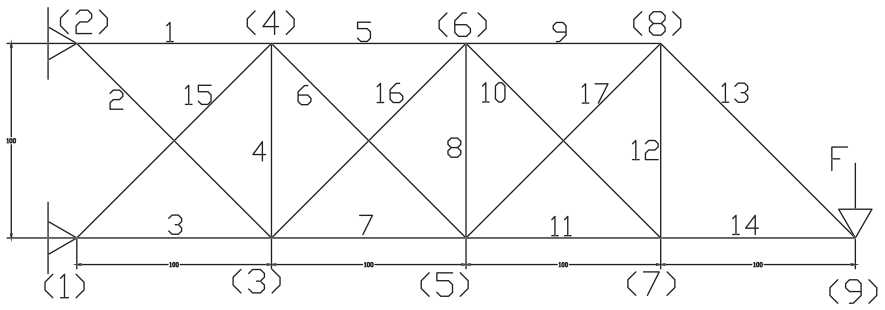

Fig. 10. 17-bar truss problem.

Table 4
17-bar truss problem: Evolved minimum cross-sectional areas (in^2).

Element	Khot [35]	Adeli [36]	Li et al. [30]	GE		DO-GE	
				Using Tata CHS sections	Steel solid sections	Using Tata CHS sections	Steel solid sections
1	15.930	16.029	15.896	12.276	16.0	13.826	16.0
2	0.100	0.107	0.103	2.387	0.1	Redundant	Redundant
3	12.070	12.183	12.092	15.392	12.2	12.276	12.2
4	0.100	0.110	0.100	0.394	0.1	Redundant	Redundant
5	8.067	8.417	8.063	8.944	8.1	7.766	8.1
6	5.562	5.715	5.591	2.403	5.7	5.828	5.7
7	11.933	11.331	11.915	9.486	12.1	12.276	12.1
8	0.100	0.105	0.100	0.578	0.1	Redundant	Redundant
9	7.945	7.301	7.965	5.828	8.0	7.766	8.0
10	0.100	0.115	0.100	2.527	0.1	Redundant	Redundant
11	4.055	4.046	4.076	7.301	4.0	4.433	4.1
12	0.100	0.101	0.100	1.736	0.1	Redundant	Redundant
13	5.657	5.611	5.670	5.208	5.6	5.208	5.7
14	4.000	4.046	3.998	4.588	4.4	4.464	4.1
15	5.558	5.152	5.548	2.651	5.7	5.208	5.7
16	0.100	0.107	0.103	3.317	0.1	Redundant	Redundant
17	5.579	5.286	5.537	3.658	5.7	5.208	5.7
Weight (lb)	2581.9	2594.4	2581.9	2774.5	2605.7	2642.1	2595.4

3.3. Sizing optimization: 17-bar cantilevered truss

Li et al. [34], Lee and Geem [38], Khot and Berke [39] and Adeli and Kumar [40] proposed solutions to the 17-bar planar truss problem as shown in Fig. 10. Nodes 1 and 2 were pinned, while a single vertical point load of 100 kips was set at node 9. It should be noted that in this case, all of the prior research utilized steel as the design material.

As with the 10-bar cantilevered truss problem above, the experiments were again run with two different sets of materials: firstly with the Tata material [36] and secondly with steel solid sections as described in Section 4.

Table 4 lists best evolved solutions from DO-GE and compares them with those found by previous works [34,38–40], including individual member cross-sectional areas and overall structure weight. Using the Tata materials, the best achieved solution was 2774.5 lb. With the use of steel solid sections, the best achieved solution was 2605.7 lb, only 23.74 lb off the best solution achieved by Li et al. [34].

3.4. Dual sizing and topology optimization: 17-bar cantilevered truss

As with the 10-bar truss discussed earlier, the second part of the experiment involved addressing the 17-bar optimization problem with the full topology search functions of the DO-GE program included in the evolutionary run. As with the 10-bar cantilevered truss problem, the best solution found by the full DO-GE program was significantly better than with pure sizing optimization when both topology optimization and sizing optimization were run in tandem, with weight reductions of up to 4.77% coming from the program choosing a "modified Warren" type structure which removed elements 2, 4, 8, 10, 12 and 16 (Fig. 11).

Using the Tata materials, the best achieved solution was 2642.1 lb, 132.4 lb lighter than the best single optimization solution by DO-GE. With the use of the steel solid sections material set, the best achieved solution was only 10.3 lb off that achieved by Li et al. [34], at 2595.4 lb (Table 4). A graph of the evolutionary runs comparing evolution of Tata materials and steel solid sections using both GE and DO-GE is presented in Fig. 12.

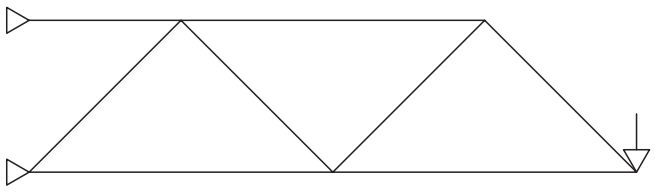

Fig. 11. 17-bar truss problem: evolved minimum topology.

4. Discussion

While the best solution for each load case found using DO-GE was consistently heavier than the best solutions from previous works, this is due to both the use of different materials (aluminium was used in the case of the 10-bar truss) and a more restricted range of available materials. Traditional optimization methods [1,16,17,19,34,35,38] calculate the minimum required cross-sectional area, while the DO-GE method presented here matches the closest commercially available element (based on a predefined list) to that minimum. Moreover, DO-GE's ability to detect unnecessary members allows it to further improve on standard GE methods.

When the results are analyzed in more detail, some interesting points become apparent. These are directly related to the application of the method as a structural design tool and highlight some important short-comings of the more traditional optimization methods.

For the literature examples presented in Section 3, compressive and tensile stress limits were identical and were set at 25 ksi in the case of the 10-bar truss and 50 ksi for the 17-bar truss. In structural design practice this is far from the case, as many section-dependent factors govern the material stress limits, including relevant design codes of practice [2,3,30,32] and manufacturer's specifications [32]. This discrepancy is particularly relevant in the case of axial compression. Both standard codes of practice [30,32] and manufacturer specified compression resistance limits [36] are considerably more conservative as they take into account various factors of safety. These are not used in traditional optimizers. It must also be stressed that these variable stress limits apply regardless of the material used, as compressive stress limits are a function of the length of the member, its cross-sectional area, and its thickness.

In the current standard for structural steel design [32], methodologies are available to calculate the allowable maximum resistance of axial compression members; this is based on the gross cross-sectional area and the compressive strength of the material (which itself is a function of the section geometry). DO-GE addresses these issues by building dictionaries of section data for each structural member, including section geometry and properties, compressive and tensile stress limits (based on section geometry, design codes of practice, and manufacturer specifications), and permissible Euler buckling limits.

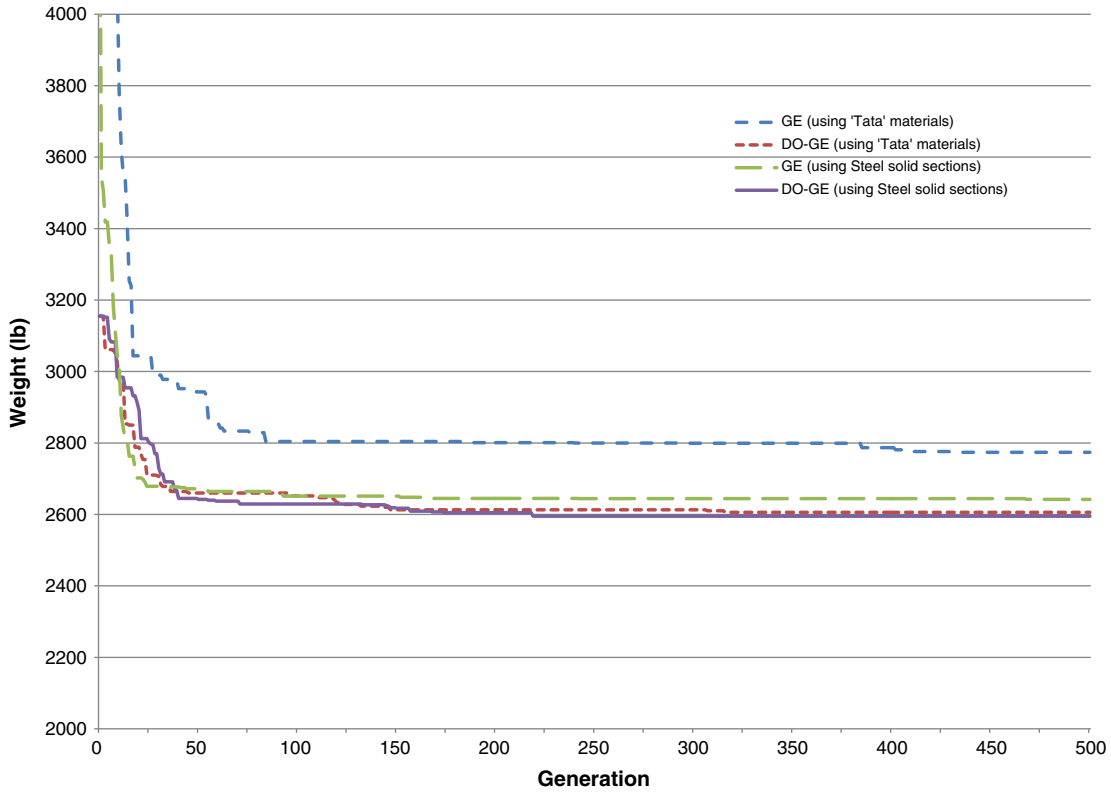

Fig. 12. 17-bar truss optimization.

A summary of the maximum permissible stress for each element in compression is given in Table 5.

Inspection of the data shows that there is considerable variation in the maximum permissible compressive stresses, with values ranging from 3.4 to 49.4 ksi. When one considers the approach used by traditional optimization methods (to take a fixed constant value for this property), the implications of this approach with respect to compressive resistance and buckling are very clear. It also highlights a very significant shortcoming of traditional optimization methods.

For illustrative purposes, all experiments were run a second time with a wider array of materials representative of those of other works (solid aluminium and steel material sets, as described in Table 1). Evolved solutions were found to be much closer to the previous best solution reported in the literature, using the standard cross sections available commercially. Comparisons between element cross-sectional areas evolved using DO-GE and those of previous methods (Tables 2–4) confirm that DO-GE is fully capable of matching other optimization methods. More importantly however, these comparisons also highlight the ever-widening gap between idealized optimization and real-world achievable optimization.

Table 5
Summary of compression elements in sample problems.

Element no.	Length (in)	Second moment of area (in^4)	Max allowable stress (ksi)
10-bar truss (load case 1)			
3	360	153.040	29.810
4	360	72.075	18.505
8	509.12	153.040	19.420
10	509.12	2.883	3.416
10-bar truss (load case 2)			
2	360	0.235	0.758
3	360	153.040	29.810
4	360	101.146	22.552
8	509.12	72.075	18.505
17-bar truss			
3	100	749.583	45.112
4	100	0.074	3.785
7	100	44.927	45.016
11	100	101.146	49.362
12	100	4.132	35.991
14	100	31.713	47.272
15	141.42	9.442	32.392
16	141.42	7.520	22.902
17	141.42	12.757	32.011

5. Conclusions and future work

Although truss optimization methods exist, this study finds that the most popular methods are not fully appropriate for everyday use in the construction industry as they focus purely on optimizing minimum element cross-sectional area, neglecting crucial section properties and material specifications. Furthermore, the widespread use of identical tensile and compressive stress limits on the material and the lack of use of design codes and standards of practice in such optimization methods give a false impression of both the efficiency of the algorithm and the achieved results.

The GE-based truss optimization approach presented in this paper can be considered more suitable for everyday use in that:

- the representation of the structure is easily encoded
- the results are human-readable and easily analyzed
- the limits of the amounts of constraints that can be applied have yet to be explored
- redundant truss elements can be identified and removed
- the approach can be transposed to many different platforms
- it uses existing predefined structural materials
- its outputs are compatible with existing truss fabrication technologies.

Most importantly, our approach only creates designs which conform to standard design codes of practice, and hence all fit individuals can be considered viable for construction.

Future work will concentrate on exploring the recursive capabilities of GE in structure generation, which would theoretically vastly increase the representation space of the program. The use of variable mutation and crossover rates in [41] also warrants investigation as it could yield additional performance benefits.

Acknowledgments

This research is based upon works supported by the Graduate Research Education Programme in Sustainable Development, jointly funded by IRCSET and IRCHSS, and based upon works supported by the Science Foundation Ireland under Grant no. 08/IN.1/I1868 and 08/RFP/CMS1115.

References

[1] G.I.N. Rozvany, A critical review of established methods of structural topology optimization, Struct. Multidiscip. Optim. 37 (2009) 217–237.
[2] F. Cobb, Structural Engineer's Pocket Book, Second edition Butterworth-Heinemann, 2009.
[3] E. Glaylord, C. Glaylord, Structural Engineering Handbook, McGraw-Hill, 1979.
[4] A. Ghali, A.M. Neville, T.G. Brown, Structural Analysis, a Unified Classical & Matrix Approach, 6th ed. Spon Press, New York, 2009.
[5] J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu, G. Lanza, Genetic Programming IV. Routine Human-Competitive Machine Intelligence, Kluwer Academic Publishers, Boston, MA, 2003.
[6] J.R. Koza, Human-competitive results produced by genetic programming, Genet. Program Evolvable Mach. 11 (3/4) (2010) 251–284.
[7] W. Banzhaf, J.R. Koza, C. Ryan, L. Spector, C. Jacob, Genetic programming, IEEE Intell. Syst. Appl. 15 (3) (May/Jun 2000) 74–84.
[8] M. O'Neill, A. Brabazon, Recent patents on genetic programming, Recent Patents Comput. Sci. 2 (1) (2009) 43–49.
[9] M. O'Neill,, C. Ryan, Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language, Kluwer Academic Publishers, 2003.
[10] M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. 5 (4) (August 2001).
[11] R.I. McKay, N. Xuan Hoai, P.A. Whigham, Y. Shan, M. O'Neill, Grammar-based genetic programming: a survey, Genet. Program Evolvable Mach. 11 (3/4) (2010) 365–396.
[12] I. Dempsey, M. O'Neill, A. Brabazon, Foundations in Grammatical Evolution for Dynamic Environments, Springer, 2009.
[13] J. Hugosson, E. Hemberg, A. Brabazon, M. O'Neill,, Genotype representations in grammatical evolution, Appl. Soft Comput. 10 (2010) 36–43.
[14] J. Byrne, M. Fenton, E. Hemberg, J. McDermott, M. O'Neill,, E. Shotton, C. McNally, Combining Structural Analysis and Multi-Objective Criteria for Evolutionary Architectural Design, Applications of Evolutionary Computation, LNCS, 6625, Springer, 2011.
[15] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[16] H. Kawamura, H. Ohmori, N. Kito, Truss topology optimization by a modified genetic algorithm, Struct. Multidiscip. Optim. 23 (2002) 467–472.
[17] R. Kicinger, T. Arciszewski, K.A. De Jong, Evolutionary computation and structural design: a survey of the state-of-the-art, Comput. Struct. 83 (23–24) (2005) 1943–1978.
[18] M. Fenton, Analysis of Timber Structures Created Using a G.E-based Architectural Design Tool, (Master's thesis) University College Dublin, Ireland, 2010.
[19] G.I.N. Rozvany, Exact analytical solutions for some popular benchmark problems in topology optimization, Struct. Optim. 15 (1998) 42–48.
[20] Z.H. Zuo, Y.M. Xie, X. Huang, Combining genetic algorithms with BESO for topology optimization, Struct. Multidiscip. Optim. 38 (2009) 511–523.
[21] X. Huang, Y.M. Xie, A new look at ESO and BESO optimization methods, Struct. Multidiscip. Optim. 35 (2008) 89–92.
[22] M. And Zhou, G.I.N. Rozvany, On the validity of ESO type methods in topology optimization, Struct. Multidiscip. Optim. 21 (2001) 80–83.
[23] Y. Li, Y. Chen, Beam structure optimization for additive manufacturing based on principal stress lines, Proceedings of Solid Freeform Fabrication Symposium, Austin, Texas, August 8–11, 2010.
[24] X. And Huang, Y.M. Xie, A further review of ESO type methods for topology optimization, Struct. Multidiscip. Optim. 41 (2010) 671–683.
[25] P. Hajela, E. Lee, Genetic algorithms in truss topological optimization, Int. J. Solids Struct. 32 (22) (1995) 3341–3357.
[26] M. Ohsaki, Genetic algorithm for topology optimization of trusses, Comput. Struct. 57 (2) (1995) 219–225.
[27] S.D. Rajan, Sizing, shape and topology design optimization of trusses using genetic algorithm, J. Struct. Eng. 121 (1995) 1480–1487.
[28] G.C. Luh, C.Y. Lin, Optimal design of truss-structures using particle swarm optimization, Comput. Struct. 89 (2011) 2221–2232.
[29] N. Pholdee, S. Bureerat, Performance enhancement of multi-objective evolutionary optimisers for truss design using an approximate gradient, Comput. Struct. 106–107 (2012) 115–124.
[30] M. O'Neill, J. McDermott, J.M. Swafford, J. Byrne, E. Hemberg, A. Brabazon, E. Shotton, C. McNally, M. Hemberg, Evolutionary design using grammatical evolution and shape grammars: designing a shelter, Int. J. Des. Eng. 3 (1) (2010) 4–24.
[31] British Standards Institution, BS 449–2, The Structural Use Of Steel In Building, BSI, London, 1969.
[32] British Standards Institution, BS 5950–1, Structural Use Of Steelwork In Building, BSI, London, 2000.
[33] B. Yegnanarayana, Artificial Neural Networks, Prentice-Hall of India, New Delhi, 2006.
[34] L.J. Li, Z.B. Huang, F. Liu, Q.H. Wu, A heuristic particle swarm optimizer for optimization of pin connected structures, Comput. Struct. 85 (2007) 340–349.
[35] A. Kaveh, S. Talatahari, Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures, Comput. Struct. 87 (5–6) (March 2009) 267–283.
[36] Tata Steel Sections Interactive "Blue Book", available online at http://www.tatasteelconstruction.com/en/design_guidance/the_blue_book/.
[37] San Le's Free Finite Element Analysis, http://slffea.sourceforge.net/.
[38] K.S. Lee, Z.W. Geem, A new structural optimization method based on the harmony search algorithm, Comput. Struct. 82 (2004) 781–798.
[39] N.S. Khot, L. Berke, Structural optimization using optimality criteria methods, in: E. Atrek, R.H. Gallagher, K.M. Ragsdell, O.C. Zienkiewicz (Eds.), New Directions in Optimum Structural Design, John Wiley, New York, 1984.
[40] H. Adeli, S. Kumar, Distributed genetic algorithm for structural optimization, J. Aerosp. Eng. ASCE 8 (3) (1995) 156–163.
[41] K. Murawski, T. Arciszewski, K. De Jong, Evolutionary computation in structural design, Engineering with Computers, 16, Springer-Verlag Limited, London, 2000. 275–286.
[42] R. Kicinger, S. Obayashi, T. Arciszewski, Evolutionary multiobjective optimization of steel structural systems in tall buildings, in: S. Obayashi (Ed.), EMO 2007, LNCS, 4403, Springer-Verlag, Berlin Heidelberg, 2007, pp. 604–618.
[43] D.E. Grierson, G.E. Cameron, Microcomputer-based optimisation of steel structures in professional practice, Microcomput. Civ. Eng. 4 (2) (1989) 289–296.
[44] R. Kicinger, T. Arciszewski, K. De Jong, Evolutionary design of steel structures in tall buildings, J. Comput. Civ. Eng. 19 (3) (July 2005) 223–238.