Optimization of Shape, Size, and Topology Design Variables in Trusses with a Genetic Algorithm

Kevin M. Gillman
Brigham Young University - Provo

Follow this and additional works at: https://scholarsarchive.byu.edu/etd
Part of the Civil and Environmental Engineering Commons

BYU ScholarsArchive Citation
https://scholarsarchive.byu.edu/etd/233
OPTIMIZATION OF SHAPE, SIZE, AND TOPOLOGY DESIGN VARIABLES IN TRUSSES WITH A GENETIC ALGORITHM

by

Kevin M. Gillman

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Civil and Environmental Engineering

Brigham Young University

April 2005
ABSTRACT

OPTIMIZATION OF SHAPE, SIZE, AND TOPOLOGY DESIGN VARIABLES IN TRUSSES WITH A GENETIC ALGORITHM

Kevin M. Gillman

Department of Civil and Environmental Engineering

Master of Science

Briggs’ genetic algorithm was extended in the Gillman algorithm to include shape optimization of trusses. Other contributions include value representation, different member linking, alternate genes, automatic k-factor evaluation for buckling, and the option to prevent overlapping members. The purpose of these contributions was to make optimization using a genetic algorithm more accessible to design engineers. The Gillman algorithm was demonstrated in two original examples as well as an example from a published work. The Gillman algorithm was effective in finding lighter designs.
ACKNOWLEDGEMENTS

I would like to thank Dr. Richard Balling for his time and patience with me as I researched for this thesis. It was exciting to work with such an intelligent person. I would also like to thank Dr. Warren Lucas and Dr. Fernando Fonseca for their reviews of my work.
# TABLE OF CONTENTS

Chapter 1: Introduction ........................................................................................................1

Chapter 2: Background on Structural Optimization ............................................................3

Chapter 3: Briggs’ Genetic Algorithm .................................................................................7

Chapter 4: Gillman’s Genetic Algorithm ..........................................................................15

Chapter 5: Briggs’ Analysis ...............................................................................................25

Chapter 6: Gillman’s Analysis ...........................................................................................29

Chapter 7: Pier Cost Example ............................................................................................35

Chapter 8: Bridge Truss Example ......................................................................................51

Chapter 9: Transmission Tower Example ..........................................................................65

Chapter 10: Rajan’s Truss Example ..................................................................................75

Chapter 11: Conclusion ......................................................................................................81

Appendix ............................................................................................................................83

References ..........................................................................................................................87
# LIST OF FIGURES

**Figure 3-1**: Software Organization .....................................................................................8  
**Figure 3-2**: Single-Point Crossover Example ...................................................................12  
**Figure 4-1**: Single-Point Crossover Example ..................................................................17  
**Figure 4-2**: Chromosome for a Candidate Design with Value representation ..............18  
**Figure 4-3**: Ground Structure for the Explanation of Size and Topology Linking ..........20  
**Figure 4-4**: Ground Structure for Explanation of Alternate Topology Genes ..............21  
**Figure 4-5**: Optimal Design if Internal Vertical Members are Present Along with Members 1-2 and 3-4. This is a Howe Topology ..................................................22  
**Figure 4-6**: Optimal Solution for Ground Structure in Figure 4-4. This is a Warren Topology ....................................................................................................................22  
**Figure 6-1**: A Ground Structure to Help Illustrate the Need for k-factor Adjustments .................................................................................................................................29  
**Figure 6-2**: Resulting Structure That Illustrates the Need for k-factor Adjustments .................................................................................................................................30  
**Figure 7-1**: Ground Structure for Pier Cost Example .....................................................36  
**Figure 7-2**: Overall Lightest Design from the 20 Runs of Section 7.4 .............................39  
**Figure 7-3**: First 5 Designs that the Algorithm Found all 20 Times .................................40  
**Figure 7-4**: Second 5 Designs that the Algorithm Found all 20 Times ............................41  
**Figure 7-5**: Third 5 Designs that the Algorithm Found all 20 Times ...............................42  
**Figure 7-6**: Last 2 Designs that the Algorithm Found all 20 Times .................................43  
**Figure 7-7**: Overall Lightest Design from the 20 Runs of Section 7.5 .............................45
Figure 7-8: First 5 Designs that the Algorithm Found all 20 Times.................................46
Figure 7-9: Second 5 Designs that the Algorithm Found all 20 Times .........................47
Figure 7-10: Last 4 Designs that the Algorithm Found all 20 Times .............................48
Figure 7-11: First 5 Best Topologies when the Pier had High Cost...............................49
Figure 7-12: Second 5 Best Topologies when the Pier had a High Cost..........................50
Figure 8-1: Ground Structure for Bridge Truss Example.................................................52
Figure 8-2: Typical Bay for Bridge Truss Example ..........................................................52
Figure 8-3: Lightest Design from the 20 Runs when Overlap Was Allowed ..................57
Figure 8-4: Lightest Design from the 20 Runs when Overlap Was Prevented...............57
Figure 8-5: 10 Most Frequently Found Designs when Overlap is Allowed.....................58
Figure 8-6: Most Frequently Found Designs when Overlap is Prevented ....................59
Figure 8-7: First 10 Out of 15 Lightest Topologies when Overlap is Allowed...............60
Figure 8-8: Last 5 Out of 15 Lightest Topologies when Overlap is Allowed .................61
Figure 8-9: First 5 Out of the Lightest 15 Topologies when Overlap is Prevented .......62
Figure 8-10: Last 10 of the Lightest 15 Topologies when Overlap is Prevented ..........63
Figure 9-1: Ground Structure of the 25-Bar Transmission Tower Space Truss ..........66
Figure 9-2: Lightest Topology for Transmission Tower Example without
            Topology as a Variable .......................................................................................71
Figure 9-3: Lightest Topology for Transmission Tower Example with
            Topology as a Variable .......................................................................................71
Figure 10-1: Ground Structure for Rajan’s Truss Example..............................................75
Figure 10-2: Rajan’s Best Result for Rajan’s Truss Example ........................................80
Figure 10-3: Gillman’s Algorithm Best Result for Rajan’s Truss Example
            when overlap is allowed .....................................................................................80
Figure 10-4: Gillman’s Algorithm Best Result for Rajan’s Truss Example when overlap is prevented ..........................................................80

Figure A: Results of Test to Determine the Best Tournament Size..........................86
LIST OF TABLES

Table 4-1: Comparison of Binary Gene Values to Discrete Design Variable Values ....................................................................................................................16

Table 7-1: Available Discrete Sizes for Examples 1 and 2--AISC Standard Weight Steel Pipe...................................................................................................37

Table 7-2: The Size Variables for Pier Cost Example .............................................38

Table 8-1: Available Sizes for Bridge Truss Example—AISC Standard Weight Steel Pipe ....................................................................................................54

Table 8-2: The Size Variables to which members of the typical bay are linked ..........54

Table 8-3: The Topology Variables to which members of the typical bay are linked ................................................................................................................55

Table 8-4: The Members That Are Mirrored When Alternate Variables Change Value .........................................................................................................56

Table 8-4: Summary of Results From the Runs of Bridge Truss Example ...............57

Table 9-1: Loads for Transmission Tower Example .........................................................67

Table 9-2: Size Variables for Transmission Tower Example. Joints C - E and G - I have variable values that depend on Shape Variables 2 or 3 ...............68

Table 9-3: Shape Variable Limits for Transmission Tower Example .......................68

Table 9-4: Size Variables for Transmission Tower Example ............................................69

Table 9-5: Topology Variables for Transmission Tower Example .............................70

Table 9-6: Comparison of Weights and Iterations Among the Works .........................71

Table 9-7: Joint Coordinates for Best Design When Topology is not a Design Variable ........................................................................................................72
Table 9-8: Joint Coordinates for Best Design When Topology is a Design Variable .................................................................72

Table 9-9: Size Variables of the Best Designs when Topology was a Design Variable and When Topology was not a Design Variable ....................73

Table 10-1: Loads for Rajan’s Truss Example ...............................................................76

Table 10-2: Size Variables and the Members Linked to Them for Rajan’s Truss Example ........................................................................................................77

Table 10-3: Topology Variables and the Members Linked to Them for Rajan’s Truss Example ..................................................................................78

Table 10-4: Summary of Results for Rajan’s Truss Example .............................................79

Table A: Average Best Cost for the Respective Probability Settings..............................85
Chapter 1: Introduction

This thesis is about the optimization of skeletal structures. Skeletal structures are those constructed of long, thin members. There are three types of design variables that can be considered in a skeletal structure. These are size variables, topology variables, and shape variables. A size variable dictates the size of a member, which could be the cross-sectional area, the moment of inertia, etc. Topology variables dictate the presence or absence of members in a structure. Shape variables dictate the joint coordinates of a structure. Size variables are ideally represented with discrete section property values. Topology variables are represented as discrete integer values. However, shape variables are typically represented as continuous, or real number values.

Briggs (2004) developed a genetic algorithm that optimizes size and topology variables in a truss. The algorithm found multiple optimum designs as opposed to a single optimum design. The main contribution of this thesis is to include shape optimization of trusses along with size and topology optimization. Other contributions include value representation, different member linking, alternate genes, automatic k-factor evaluation for buckling, and the option to prevent overlapping members. The purpose of these contributions is to make optimization using a genetic algorithm more accessible to design engineers.
This thesis describes the modifications to Briggs’ genetic algorithm. Chapter 2 provides a background on structural optimization. Briggs’ genetic algorithm is explained in Chapter 3. Chapter 4 describes the changes made to Brigg’s genetic algorithm. Briggs’ analysis is explained in Chapter 5. Changes to Briggs’ analysis are explained in Chapter 6. Chapters 7 through 10 present four examples where Gillman’s algorithm is tested. The examples in Chapters 7 and 8 are original examples, and the examples in chapters 9 and 10 were obtained from the literature.
Chapter 2: Background on Structural Optimization

Structural optimization has attracted many researchers starting with Michell in 1904 (Michell 1904). The classic paper of Michell presents theoretical optimum shapes for statically determinate trusses. Even though it has been studied for more than 100 years, no single method has proved to be entirely efficient and robust for the entire range of structural engineering optimization problems (Rajeev and Krishnamoorthy 1992). Some of the earliest methods involved mostly continuous design variables for size and shape. However, discrete design variables would have to be implemented into optimization to treat the standardized sizes available on the market.

Researchers have modified continuous methods for discrete optimization. Design variables can be rounded to the nearest discrete values if a structure is optimized using a continuous method. Grierson and Lee (1984) use such an approach in their work. Rounded design variable values however may give suboptimal, overly conservative, or infeasible designs (Glover and Sommer 1975). Rounding continuous design variables to discrete values is not effective.

Later approaches treated the design variables as discrete. The branch and bound method has been used extensively (Land and Doig 1960, Dakin 1965, Garfinkel and Nemhauser 1972, Gupta and Ravindran 1984, Sandgren 1990, Cha and Mayne 1989a,
Cha and Mayne 1989b, Mesquita and Kamat 1987). The problem with the branch and bound method is that it requires solving multiple continuous optimizations as subproblems, which can take a large amount of time (Thanedar and Vanderplaats 1995).

In the early 1960s, some researchers modeled genetic information on computers (Barricelli 1962, Fraser 1962, Martin and Cockerham 1960). Advances in modeling DNA on the computer have led to developing algorithms (genetic algorithms) to optimize systems including structures (Holland 1975, Goldberg 1989a). Genetic algorithms can be effective in optimizing discrete, as well as continuous design variables.

Genetic algorithms have advantages and disadvantages. The following is a list of some advantages:

- Designs that cause the analysis program to crash can be penalized, and the algorithm can proceed
- Discrete sizing is allowed
- A large number of design variables is possible
- Nondifferentiable objectives and constraints can be considered
- Multiple optimum solutions can be determined

One main disadvantage of a genetic algorithm is computation time (Soh and Yang 1996). This does not mean that the method is useless; rather, it means that it may be some time before computers get fast enough to handle some problems.

Several studies about optimizing size variables in 2D or 3D frames with genetic algorithms exist. Some include modifications to the genetic algorithms, while others include modifications to the analysis. The size optimization of frames usually employs discrete design variables. Pezeshk, Camp and Chen (2000) optimized discrete size
variables in 2-D frames with their analysis including AISC-LRFD specifications. Their analysis also considered the P-Δ effect. Foley and Schinler (2003) considered connection behavior in planar frames in their analysis. Their genetic algorithm used homologous and nonhomologous crossover operations (Voss and Foley 1999) where binary genes could migrate from their original location to alternate locations. Toropov, Mahfouz, and Westbrook (1999) used a process that “more accurately evaluated effective buckling length of columns” in their analysis.

Shape variables are usually included when a truss is to be optimized. Most of the devised methods for shape optimization of trusses also include size variable optimization with modifications to the genetic algorithm or the analysis. Soh and Yang (1996) optimized trusses using continuous shape and size variables. A relationship between the cross sectional area and the moment of inertia was implemented to consider buckling. Their contribution involved applying fuzzy set theory to their analysis. Chapter 9 of this thesis compares results of the Soh and Yang genetic algorithm to the work here presented. Galante (1996) also used a genetic algorithm that included shape and size variables. Galante’s algorithm found more realistic designs than Soh and Yang, since the size variables were discrete. The algorithm preserved diversity and included an objective to minimize the number of sizes in a design.

Including topology design variables in an algorithm can make it more robust. Researchers have used topology optimization in conjunction with size or shape optimization. Few have used all three variables together. Rajeev and Krishnamoorthy (1997), Shrestha and Ghaboussi (1998), and Rajan (1995) have used these three types of
design variables. Chapter 10 of this thesis compares results of Rajan’s genetic algorithm to the work here presented.

Topology optimization with a genetic algorithm can be handled in two ways. Goldberg proposed a “messy genetic algorithm” (Goldberg 1989b). The chromosomes representing structures with different topologies would have different lengths with this approach. Another way is used by Grierson and Pak (1993), Hajela and Lee (1994), and Shrestha and Ghaboussi (1998). They used ground structures with binary genes that represented whether or not a member was present. Rajeev and Krishnamoorthy (1997) propose that this type of representation is inefficient since a “designer can visualize possible topologies” and “many members in the preceding representations become redundant.” They use variable sizes of chromosomes in their algorithm that considers topology, but the user defines all the topologies. Using a ground structure approach is not necessarily inefficient as shown by the work of Rajan (1995) and the work presented in this thesis.
Chapter 3: Briggs’ Genetic Algorithm

This chapter briefly explains Briggs’ genetic algorithm. The bulk of Brigg’s genetic algorithm is used as Gillman’s genetic algorithm with the exception of some modifications that will be discussed in Chapter 4. The representation as well as the generations of designs is explained in this chapter followed by the five-step process used in Briggs’ algorithm to create new generations.

The genetic algorithm used in this thesis follows the organization of Figure 3-1. It begins by calling a routine that initializes the analysis application. This routine inputs and saves in the memory all of the values used in the analysis, such as joint coordinates, member connectivity, loads, etc. This routine is only called once. The genetic algorithm repeatedly calls the application analysis for every trial design that it generates. The genetic algorithm provides the design variables to the analysis. The analysis returns the objective and feasibility values to the genetic algorithm.
3.1: Representation

Genetic algorithms use chromosomes to represent designs. Chromosomes are composed of several genes that indicate values of design variables. Briggs used a binary representation, which uses multiple binary genes to represent a single design variable value. For example, three binary genes are required to represent a design variable whose integer values range from 1 to 5.

Briggs used one binary gene to represent each topology variable. If a topology variable has a value of 1, the members linked to that variable are present. If the topology variable has a value of 0, the members linked to that variable are absent.

3.2: Generations

Genetic algorithms work with generations of designs. The designer specifies the generation size N, which is the number of designs in each generation. The genetic algorithm begins with a starting generation of randomly generated designs. This is accomplished by randomly generating the values of the genes of the N chromosomes in

Figure 3-1: Software Organization
the starting generation. From the starting generation, the algorithm creates the second generation, and then the third generation, and so forth until the specified number of generations has been created.

The genetic algorithm goes through a five-step process to create a new generation from a current generation:

1. fitness
2. selection
3. crossover
4. mutation
5. elitism

3.3: Fitness

Every chromosome should be assigned a fitness value. The fitness is a single number indicating the quality of the design represented by the chromosome. To evaluate the fitness, each design must be analyzed to evaluate the objective, \( f(\text{minimized}) \) and constraints \( g_i \leq 0 \) (\( i = 1 \) to 2, where 2 is the number of constraints). Objective and constraint values must be combined into a single fitness value. The feasibility is defined in Equation 3.1.

\[
g = \max(0, g_1, g_2) \quad \text{Eq. 3.1}
\]

Note that the design is infeasible if \( g > 0 \) and feasible if \( g = 0 \).

The structural analysis is implemented to determine if the constraints have been violated. The constraint value for deflection, \( g_1 \), is calculated with Equation 3.2 if the displacement constraint is violated where \( \Delta_i \) is the deflection for joint \( i \) and \( \Delta_{\text{max}(i)} \) is the allowable deflection for joint \( i \).
\[ g_1 = \max\left(\frac{\Delta_i}{\Delta_{\text{allow}(i)}} - 1\right) \quad \text{Eq. 3.2} \]

The constraint value for stress, \( g_2 \), is calculated with Equation 3.3 if the stress constraint is violated where \( \sigma_i \) is the stress in member \( i \) and \( \sigma_{\text{max}(i)} \) is the allowable stress in member \( i \). The stress constraints involve maximum normal stress and buckling stress.

\[ g_2 = \max\left(\frac{\sigma_i}{\sigma_{\text{allow}(i)}} - 1\right) \quad \text{Eq. 3.3} \]

After the feasibility is determined, the objective value is calculated. The objective used is weight. The weight of each design is determined by summing the weight of each member. Each member’s weight is calculated by multiplying its specified unit weight by its length. With the feasibility and objective determined, a fitness value needs to be assigned to each design.

The definition of fitness used is the segregation approach. The segregation approach involves the user-specified non-negative segregation parameter \( G \). The fitness value is calculated with Equation 3.4 where \( f_{\text{max}}^{\text{feas}} \) is the maximum value of \( f \) for all designs where \( g < G\alpha \). The fitness is minimized. For Briggs’ algorithm, \( G = 0 \). This guarantees that the fitness values of feasible designs in the generation are always better (lower) than the fitness values of infeasible designs. After the fitness values with respect to feasibility and objectives are assigned to the design, then the designs are evaluated for more fitness adjustments.

Briggs developed a step to assign large fitness values to duplicate topologies. Duplicate topologies are assigned a fitness value that is essentially infinity (the maximum
real value that the computer handles). This step helps maintain diversity by ensuring that
the whole population does not become occupied by one design. A design's topology
variables indicate whether or not a design is a duplicate topology of another design.

3.4: Selection
Selection is performed after fitness values are assigned. Two designs from the
current generation are chosen as the mother and father designs for the next generation.
Briggs’ algorithm uses tournament selection. A specified tournament size dictates how
many designs will be randomly selected from the current generation. The most fit of this
number becomes the mother design. This is repeated to obtain the father design.

3.5: Crossover
Two children designs are created for the next generation by the crossover process
after selection. First, it must be determined whether or not crossover should occur. The
user specifies a crossover probability. A random number between zero and one is
generated. If the number is less than the crossover probability, crossover is performed.
Otherwise, the mother and father designs become the two children designs, and are
copied without modification to the next generation.

The crossover method used by Briggs is single-point crossover. Figure 3-2 shows
the chromosomes for a mother design and a father design. Each chromosome has ten
binary genes. With single point crossover, a random integer i from 1 to n is generated,
known as the crossover point, where n is the number of genes in the chromosome. The
genes in the mother and father chromosomes after gene i are swapped. In Figure 3-2, i =
7. The first child is identical to the father after the crossover point. The second child is
identical to the father before the crossover point and identical to the mother after the crossover point.

```
crossover point
mother 1 0 0 1 1 0 1 0 0 0
father 1 1 1 0 1 0 0 1 1 1
first child 1 0 0 1 1 0 1 1 1 1
second child 1 1 1 0 1 0 0 0 0 0
```

**Figure 3-2:** Single-Point Crossover Example

### 3.6: Mutation

The next step for creating the new generation is mutation. The user specifies a mutation probability, which is generally much lower than the crossover probability. The mutation process is performed for each gene of the first child design and for each gene of the second child design. The mutation process is very simple. If a randomly generated real number between 0 and 1 is less than the mutation probability then the gene is randomly changed to another value. Otherwise, the gene is left alone. The majority of genes are left alone since the mutation probability is low. Mutation makes it possible to introduce diversity into the population of designs.

### 3.7: Elitism

The selection, crossover, and mutation processes produce two new children designs for the new generation. These processes are repeated continuously to create
additional children until the number of designs in the new generation reaches the specified generation size. The final step that must be performed on this new generation is elitism. This step is necessary to guarantee that the best designs survive from generation to generation. One may think of elitism as the rite of passage for children designs to qualify as future parents. The new generation is combined with the previous generation to produce a combined generation of $2N$ designs, where $N$ is the generation size. The combined generation is sorted by fitness, and the $N$ most fit designs survive as the next parent generation. Thus, children must compete with their parents to survive to the next generation.
Chapter 4: Gillman’s Genetic Algorithm

Some changes to Briggs’ genetic algorithm were necessary so that shape, size and topology could be considered together in an optimization process. These changes include:

1. The implementation of a value representation with true continuous genes.
2. The implementation of a continuous uniform crossover operator.
3. The implementation of a continuous mutation operator.
4. The implementation of different member linking for size and topology variables.
5. The implementation of alternating topology variables.

4.1: Value Representation

Three problems exist with binary representation. These problems are:

- Bias
- Precision
- Inheritance

The following example illustrates the bias of binary representation. Consider the case of a discrete design variable whose value is an integer ranging from 1 to 5. Three binary genes are needed to represent this design variable. Table 4-1 shows the correspondence between the gene values and integer design variable values. Note that there is bias in the representation. The discrete values 1, 2, and 3 occur twice as often as the discrete values 4 and 5.
Consider the problem with precision. For some optimization problems with shape design variables, precise variable values are necessary. Consider the case of a continuous design variable whose value ranges from 3.000 to 9.000. The number of genes used to represent this design variable in a binary representation will dictate the precision of the representation. For example, if three genes are used, the gene values and continuous design variable correspond in this manner:

<table>
<thead>
<tr>
<th>Gene Values</th>
<th>Design Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>1</td>
</tr>
<tr>
<td>0 0 1</td>
<td>2</td>
</tr>
<tr>
<td>0 1 0</td>
<td>3</td>
</tr>
<tr>
<td>0 1 1</td>
<td>4</td>
</tr>
<tr>
<td>1 0 0</td>
<td>5</td>
</tr>
<tr>
<td>1 0 1</td>
<td>1</td>
</tr>
<tr>
<td>1 1 0</td>
<td>2</td>
</tr>
<tr>
<td>1 1 1</td>
<td>3</td>
</tr>
</tbody>
</table>

Consider the problem with precision. For some optimization problems with shape design variables, precise variable values are necessary. Consider the case of a continuous design variable whose value ranges from 3.000 to 9.000. The number of genes used to represent this design variable in a binary representation will dictate the precision of the representation. For example, if three genes are used, the gene values and continuous design variable correspond in this manner:

<table>
<thead>
<tr>
<th>Gene Values</th>
<th>Design Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>3.000</td>
</tr>
<tr>
<td>0 0 1</td>
<td>3.857</td>
</tr>
<tr>
<td>0 1 0</td>
<td>4.714</td>
</tr>
<tr>
<td>0 1 1</td>
<td>5.571</td>
</tr>
<tr>
<td>1 0 0</td>
<td>6.429</td>
</tr>
<tr>
<td>1 0 1</td>
<td>7.286</td>
</tr>
<tr>
<td>1 1 0</td>
<td>8.143</td>
</tr>
<tr>
<td>1 1 1</td>
<td>9.000</td>
</tr>
</tbody>
</table>
Note that the precision of this representation is \((9.000 - 3.000) / (2^3 - 1) = 0.857\).

Consider a second example where the minimum and maximum values are 0.1 and 50, respectively. A precision of 0.01 is desired. Nine binary genes are required to achieve the desired precision.

![Crossover Example](image)

**Figure 4-1: Single-Point Crossover Example**

Consider the inheritance problem. The goal of crossover is to generate two new children designs that inherit many of the characteristics of the fit parent designs. This goal may not be achieved when the binary representation is used. Suppose the last four genes in the chromosomes in Figure 4-1 represent a single discrete design variable whose value is equal to the base ten value of the last four genes. Binary values of 1000 give a design variable value of 8 for the mother design. Binary values of 0111 give a design variable value of 7 for the father design. Binary values of 1111 give a design variable value of 15 for the first child design. Binary values of 0000 give a design variable value of 0 for the second child. Thus, the parents have values of 8 and 7, while the children
have values that are very different from the parent values of 15 and 0. These problems with binary representation motivated changing to value representation. With value representation, the chromosome consists of the values of the design variables placed side by side. For example, suppose there are 6 discrete design variables whose integer values range from 1 to 5. Suppose there are 4 continuous design variables whose values range from 3.000 to 9.000. A possible chromosome is shown in Figure 4-1.

| 4 | 3 | 1 | 3 | 2 | 5 | 3.572 | 6.594 | 5.893 | 8.157 |

Figure 4-2: Chromosome for a Candidate Design with Value representation

### 4.2: Uniform Crossover Operator

Uniform crossover is used instead of single point crossover. The following process is followed for crossover of discrete values. Consider the discrete parent gene values of 7 and 8 discussed above. A random number between 0 and 1 is generated for this gene. If the random number is greater than 0.5, the first child receives the gene value of the mother (7) and the second child receives the gene value of the father (8). If the random number is less than 0.5, the first child receives the gene value of the father (8) and the second child receives the gene value of the mother (7). When used with value representation, then the problem of inheritance discussed in section 4.1 is solved.

A method for crossover needs to be established for real number values. Simply trading the continuous values between the designs may not provide the desired diversity to achieve optimal designs. The following crossover routine is used in the Gillman algorithm (Deb 2001).
Uniform crossover may be generalized further for a value representation of continuous design variables. As is, uniform crossover requires that children designs receive either the mother value or the father value of each particular design variable. This can be modified so that children designs receive a random value between the mother value and the father value. Thus, for a particular continuous design variable, if $x_1$ is the mother value and $x_2$ is the father value, then the children values $y_1$ and $y_2$ are calculated with Equations 4.1 and 4.2 where $r$ is a random number between 0 and 1. This modification makes it possible for crossover to produce continuous values that are between the parent values.

4.3: Continuous Mutation Operator

As with crossover, a new scheme for mutation needs to be used for the continuous design variables. The method used in the Gillman algorithm is very simple. Consider a continuous gene that may be mutated. A randomly generated real number between 0 and 1 is less than the user specified probability of mutation; therefore this gene will be mutated. A random real number between the minimum and maximum values for that design variable is generated. This value becomes the new design variable value.

4.4: Topology and Size Variable Linking

Briggs linked topology variables and size variables to the same members. The same number of topology variables as size variables exist in the Briggs genetic algorithm. All the members that are linked to a particular size variable are linked to the same
topology variable. It was anticipated that it might be useful to have more topology variables than size variables or more size variables than topology variables. The Gillman algorithm allows for different size linking and topology linking. Therefore, all the members that are linked to a particular size variable do not need to be linked to the same topology variable.

Consider the case where more topology variables than size variables are desired. The ground structure for a truss in Figure 4-3 will be used to illustrate this case. All of the internal members (all the members except the top and bottom chords) are to have the same size. Fifteen topology variables exist if each internal member is linked to its own topology variable. The top chord and bottom chord are not linked to topology variables, so they will always be present. All the members in the top chord are linked to the same size variable. All the members in the bottom chord are linked to the same size variables. Therefore, there are 3 size variables and 15 topology variables. All the members linked to one variable would have the same size, but they also would either be present or absent if Briggs’ method of linking were used. Linking topology and size variables separately allows members linked to the same size variable to be selectively deleted, or set absent. This method also allows members linked to the same topology variable to assume different sizes.

Figure 4-3: Ground Structure for the Explanation of Size and Topology Linking
4.5: Alternate Topology Genes

A method of altering topology genes is proposed in the Gillman algorithm.

Consider the ground structure shown in Figure 4-4. This is the same structure as that in Figure 4-3, but with more bays. Corresponding bay members are linked to the same topology variables. Members 1-2 and 3-4 are connected to the same topology variable in bays A, B, C, and D.

If all of the internal members except the vertical members and members 1-2 and 3-4 are set absent then the resulting Howe design is shown in Figure 4-5. Consider a case where the Warren design in Figure 4-6 is as optimal as the Howe design. The member linking could be changed so that the algorithm can find the Warren design. However, it will not be able to find the Howe design. More topology genes can be introduced to ensure that both of these topologies can result. The members of bays A and D can be linked and the members of bays B and C can be linked. This introduces 15 more topology variables, for a total of 30.
Figure 4-5: Optimal Design if Internal Vertical Members are Present Along with Members 1-2 and 3-4. This is a Howe Topology

Figure 4-6: Optimal Solution for Ground Structure in Figure 4-4. This is a Warren Topology

The Howe and Warren designs are identical besides the fact that bays B and C in the Warren topology is effectively a mirror image of bays B and C in the Howe topology. Alternate genes are introduced to allow for topologies such as in Figure 4-5. Alternate genes allow for the bays to be oriented in the original direction, or oriented as a mirror image of the original direction.

An alternate gene is linked to two sets of members, X and Y. The members in sets X and Y retain their original topology variable values if their alternate gene has a value of 1. The members in set X assume the topology variable values of the members in set Y while the members in set Y assume the topology variable values of the members in set X if their alternate gene value has a value of 0. Notice that the alternate genes work directly
with members rather than topology variables. This is because it may be desirable to alternate one set of members, but not all of the members that share its topology variables, such as the bays in Figure 4-4. The bays in the structure in Figure 4-4 can be oriented in the original direction or as a mirror image of the original direction if the members are appropriately linked to two alternate genes.
Chapter 5: Briggs’ Analysis

Briggs used a standard space frame analysis with a few modifications. He split the analysis into an initialize step and an analysis step as shown Figure 3-1. The initialize step is only called once and the analysis step is called for each child design. The initialize step inputs the structure data and stores it in the computer memory. The genetic algorithm provides the design variable values of a child design to the analysis. The analysis then returns the objective and feasibility values for the child design. The design variable values passed to the analysis include the size variables and the topology variables.

Briggs devised a method to deal with the absent members during the analysis. The members linked to topology variables that have a value of 0 are not really absent. They are just given a negligible stiffness. Before the member stiffness matrix is constructed for an absent member, the modulus of elasticity for that member is divided by a large number specified by the user. This prevents a singular stiffness matrix so that the analysis does not crash. Assuming members have negligible stiffness is a simple way to combat failure in the analysis.

The stiffness matrix is used to solve for the displacements in the structure. Displacements are divided by the allowable displacements for each joint to calculate the displacement constraint values. The displacement constraint value for the structure is the
maximum displacement constraint value of all of the joints (Eq. 3.2).

The stresses in each member are calculated after the displacements are determined. The stresses are scaled to conclude whether the stress constraints are violated. The stresses considered for a truss are buckling and normal stress. These stresses are not calculated for absent members. The stress constraint value for the structure is the maximum stress constraint value out of all of the members. Equations 5.1-5.4 are used to calculate the stresses.

\[
S_1^i = \frac{\sigma_{\text{Normal}}}{\sigma_{\text{Allow}}} - 1 \quad \text{Eq. 5.1}
\]

\[
S_2^i = \frac{-P_iL_i^2}{\pi^2 EI_i} - 1 \quad \text{Eq. 5.2}
\]

\[
g^i_{\text{stress}} = \max(S_1^i, S_2^i) \quad \text{Eq. 5.3}
\]

\[
g_2 = \max(g^i_{\text{stress}}) \quad \text{Eq. 5.4}
\]

\(S_1^i\) = Constraint value for normal stress for member \(i\)
\(\sigma_{\text{Normal}}^i\) = Calculated normal stress for member \(i\)
\(\sigma_{\text{Allow}}^i\) = User specified maximum normal stress

\(S_2^i\) = Constraint value for buckling for member \(i\)
\(P_i\) = Calculated normal force for member \(i\)
\(L_i\) = length of the member \(i\)
\(E\) = Young’s Modulus of Elasticity
\(I_i\) = User specified moment of inertia of member \(i\)
\(g^i_{\text{stress}}\) = Stress constraint value for member \(i\)
\(g_2\) = Stress constraint value for the design

Another modification Briggs made with his analysis involves members that have a very small stress constraint value. Topology variables linked to members that have stress constraint values below a user specified “zero stress” value are set to 0. Only members with approximately zero stress in them will be considered absent.
After the design’s displacement and stress constraints have been determined, the weight of the structure is calculated. The weight of the structure is the sum of the weights of all present members. The unit weight of a member is associated with the size variable to which that particular member is linked.
Chapter 6: Gillman’s Analysis

Two modifications to Briggs’ analysis are made in the Gillman analysis. These are an automatic evaluation of the buckling k-factor and the preventing overlapping members option.

6.1: Automatic Evaluation of k-factor

The structural analysis in the algorithm uses Euler’s buckling equation (Equation 6.1) to find the critical compressive loads on the members.

\[ P_{cr} = \frac{\pi^2 EI}{(kL)^2} \]  

Eq. 6.1

Figure 6-1: A Ground Structure to Help Illustrate the Need for k-factor Adjustments

A problem revealed itself while testing the algorithm. Consider the ground structure in Figure 6-1. Optimum topologies such as the one in Figure 6-2 could be found when topology is allowed to change. The algorithm needs to be adjusted so it considers
when collinear members’ supporting members are deleted. The top member in Figure 6-2, which will be in compression, really has an effective length composed of the sum of the lengths of all the top members. In other words, the k factors for buckling become 2 instead of 1 for both top members. A routine was devised to correct this problem.

![Figure 6-2: Resulting Structure That Illustrates the Need for k-factor Adjustments](image)

The process of fixing this problem for this thesis only applies to truss structures where the k-factor is usually 1. When ends of members are not pin connected, the effective length factor nomographs can be used in the AISC Manual. Applying the nomograph to analysis used in a genetic algorithm where connection conditions can vary, such as how many members are connected at a joint, will require a more advanced routine.

The routine will change the k factor for members in a truss. It loops through all the present members, $i$ and checks them to determine if they are unsupported at their joints and collinear with other members. Member $i$ needs to be present.
Two conditions need to be met before the joint on either side of member $i$ is considered unsupported.

1. Two present members connected at a joint need to be collinear.

2. All other members at a joint need to be absent.

Both joints associated with member $i$ need to be checked. To check the first joint $k$ of member $i$, the routine finds all other present members that connect at joint $k$. If a member $j$ is found to also connect at joint $k$, then the three sets of joint coordinates associated with members $i$ and $j$ are sent to another routine. This routine concludes whether or not the three coordinates are collinear. Because both members are associated with the same joint, there are only three points that need to be tested.

This collinear test uses the fact that if two vectors are collinear, their cross product will be zero. Members $i$ and $j$ are represented as two vectors. The two vectors are considered collinear if the absolute value of their cross product is less than a specified tolerance. The routine notes with which member $j$ that member $i$ is parallel if the members are considered parallel. The routine notes that joint $k$ is supported if the members are not considered parallel. The routine follows the assumption that any other members parallel to member $i$ at joint $k$ will be supported against buckling if joint $k$ has two nonparallel members connected to it.

The other joint with which member $i$ is attached is checked after the first joint $k$ that member $i$ is connected to has been evaluated, or if any member $j$ is assumed to support joint $k$. The same procedure for examining joint $k$ is followed.

The routine collects information throughout the routine that allows it to adjust the $k$ factors for the lengths. The routine finds sets of members that are collinear. As in
Figure 6-2, there are three members that are collinear and unsupported. These	hree members would compose a set of members for which that this routine searches. The
sum of the lengths of all the members in this set is calculated. This sum is assigned to all
the members in the set. This effectively adjusts the k factor value for buckling
calculations.

6.2: Preventing Overlapping Members Option

Some ground structures include members that overlap, or in other words, they
cross but are not connected. The option of penalizing designs with overlapping members
exists in the Gillman algorithm. A designer does not necessarily desire that the optimal
solution have overlapping members even though many overlapping members are
included in a ground structure. This thesis proposes that any designs with members that
overlap are deemed unfeasible to prevent undesirable designs from surviving to the next
generation. This constraint has proved to produce more aesthetically pleasing designs
with more real world application. It was shown that using this constraint in the truss
bridge example in Chapter 8 yields lighter designs.

Determining if two members overlap is accomplished in the following way. A
routine called from the analysis loops through all pairs of present members. An equation
is formed that is defined by the coordinates of the joints with which each member is
connected. The routine determines if the two lines are in the same plane. If not, then the
members do not overlap. If so, then the routine determines if the two equations define
parallel lines. If not, then the members do not overlap. If so, then the routine determines
the common coordinates for the two equations. The two members intersect if the
coordinates of intersection lie within the boundaries of the members. Two members may
not intersect though the lines that run through their joint coordinates intersect. This is because the point of intersection may lie outside the spans of one or both members. The feasibility is set to infinity if there is a point of intersection within the spans of both members.
Chapter 7: Pier Cost Example

The first example involves a ground structure that includes the cost of placing a pier in the middle of the span of a bridge. The algorithm was run for two cases. The cost of the pier is low for the first run. The cost of the pier is high for the second run.

7.1: Ground Structure

The ground structure for this example is shown in Figure 7-1. Because the structure is considered symmetric, joints A, B, C, E, and F are the same on both sides of member DG. Member DH simulates the cost of placing a pier in the middle of the span. The objective for this example will be cost with the assumption that the weight of the members corresponds to their cost.

The support conditions are as shown on Figure 7-1. A joint is not included where the members overlap (such as members BF and CE). The point loads at joints B, C, and D are 5 kips downward. This load is the dead load of the roadway as well as the live loads the bridge will need to carry. It is assumed that the support members for the roadway (AB, BC, CD) will take the entire axial load for the roadway. That is why some of the topologies that the algorithm will find will have the axial support members for the roadway absent. The roadway is not shown in this ground structure. The only shape variables that can vary correspond to the vertical coordinates of joints E and F.
This example was run 20 times with different random number sequences, as recommended by Davis (Davis 1991). Some general inputs are listed below. For a justification for using the probabilities and tournament size, see Appendix.

- **Generation Size** = 300
- **Number of Generations** = 300
- **Probability of Crossover** = 0.1
- **Probability of Mutation** = 0.1
- **Tournament Size** = 12
- **Members are considered to have zero stress if their stress constraint values are less than 0.001. The stress constraint value has no units.**
- **The deflection limit for this problem is 10 inches in the horizontal or vertical directions for all joints.**
- **Overlapping members are prohibited.**
7.2: Shape Inputs

The two shape variables will be linked to joints E and F. Their vertical coordinates are allowed to vary with a minimum value of 0.1 inches and a maximum value of 50 inches.

7.3: Size and Topology Inputs

The material properties are listed below. The sizes available for this example are the AISC Standard Weight Steel Pipes. Table 7-1 summarizes their properties.

- Modulus of elasticity is 29,000 ksi
- Normal allowable stress is 60 ksi
- Weight density is 0.28 lb/in$^3$

Table 7-1: Available Discrete Sizes for Examples 1 and 2--AISC Standard Weight Steel Pipe

<table>
<thead>
<tr>
<th>Diameter</th>
<th>Area (in$^2$)</th>
<th>I (in$^4$)</th>
<th>Weight (lb/in)</th>
</tr>
</thead>
<tbody>
<tr>
<td>½</td>
<td>0.25</td>
<td>0.0171</td>
<td>0.071</td>
</tr>
<tr>
<td>¾</td>
<td>0.333</td>
<td>0.037</td>
<td>0.094</td>
</tr>
<tr>
<td>1</td>
<td>0.494</td>
<td>0.0873</td>
<td>0.140</td>
</tr>
<tr>
<td>1 ¼</td>
<td>0.669</td>
<td>0.195</td>
<td>0.189</td>
</tr>
<tr>
<td>1 ½</td>
<td>0.799</td>
<td>0.31</td>
<td>0.227</td>
</tr>
<tr>
<td>2</td>
<td>1.07</td>
<td>0.666</td>
<td>0.305</td>
</tr>
<tr>
<td>2 ½</td>
<td>1.7</td>
<td>1.53</td>
<td>0.483</td>
</tr>
<tr>
<td>3</td>
<td>2.23</td>
<td>3.02</td>
<td>0.632</td>
</tr>
<tr>
<td>3 ½</td>
<td>2.68</td>
<td>4.79</td>
<td>0.760</td>
</tr>
<tr>
<td>4</td>
<td>3.17</td>
<td>7.23</td>
<td>0.900</td>
</tr>
<tr>
<td>5</td>
<td>4.3</td>
<td>15.2</td>
<td>1.217</td>
</tr>
<tr>
<td>6</td>
<td>5.58</td>
<td>28.1</td>
<td>1.583</td>
</tr>
<tr>
<td>8</td>
<td>8.4</td>
<td>72.5</td>
<td>2.383</td>
</tr>
<tr>
<td>10</td>
<td>11.9</td>
<td>161</td>
<td>3.375</td>
</tr>
<tr>
<td>12</td>
<td>14.6</td>
<td>279</td>
<td>4.133</td>
</tr>
</tbody>
</table>
This example uses size variable linking. Table 7-2 indicates which members are linked to the 4 size variables. Each member is linked to its own topology variable except members AB, BC, and CD. These members are linked to the same topology variable.

Table 7-2: The Size Variables for Pier Cost Example

<table>
<thead>
<tr>
<th>Size Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AB, BC, CD</td>
</tr>
<tr>
<td>2</td>
<td>AE, EF, FG</td>
</tr>
<tr>
<td>3</td>
<td>BE, CF, BF, CE, CG, DF</td>
</tr>
<tr>
<td>4</td>
<td>DG, DH</td>
</tr>
</tbody>
</table>

7.4: Results When the Pier has a Low Cost

Figure 7-2 shows the lightest design that the algorithm determined out of all the runs. This topology was found 20 out of the 20 runs. All but 1 of these designs from the 20 weighed from 55.7 lb to 56.0 lb (with 10 designs weighing 55.7 lb). For 19 out of the 20 runs, this was the best design. The one that was not the best of its run weighed 90.6 lb.

The size variable areas for the lightest design are $A_2 = 0.333 \text{ in}^2$, $A_3 = 0.25 \text{ in}^2$, and $A_4 = 0.799 \text{ in}^2$ (No present members are linked to the first size variable). The shape variables for the lightest design are $E_y = 0.122 \text{ in.}$ and $F_y = 16.774 \text{ in.}$ Seventeen other designs appeared in 20 out of the 20 runs. These are shown in Figures 7-3 through 7-6 along with their lowest and average weights.
Figure 7-2: Overall Lightest Design from the 20 Runs of Section 7.4

It was determined that equilibrium will be satisfied for the suspension topology in Figure 7-2 if \( y_F = \frac{y_G}{3} + y_E \), where \( y_F, y_G, \) and \( y_E \) are the vertical coordinates for joints F, G, and E respectively. For a suspension topology where Joint E has a vertical coordinate of 5, Joint F needs to have a vertical coordinate of exactly \( \frac{21}{3} \) inches for equilibrium to be satisfied.

The difference in \( y_F \) in the suspension designs that the algorithm found and the \( y_F \) required to satisfy equilibrium is unacceptable in a linear analysis. Analysis of the designs using the joint coordinates from the genetic algorithm yields large displacements in the horizontal and vertical directions for joints B, C, E, and F. The vertical displacements for these four joints are on the order of 63,000 inches. The horizontal displacements at these 4 joints range between 259 inches and 41,681 inches. A maximum displacement of 0.921 inches occurs in the vertical direction at joint B if the calculated equilibrium value with respect to \( y_E \) (16.789 inches) of the lightest design is analyzed. The algorithm reports a
Figure 7-3: First 5 Designs that the Algorithm Found all 20 Times
Figure 7-4: Second 5 Designs that the Algorithm Found all 20 Times

Average Weight = 88.6 lb
Lightest Weight = 85.9 lb

Average Weight = 93.4 lb
Lightest Weight = 93.4 lb

Average Weight = 98.3 lb
Lightest Weight = 98.0 lb

Average Weight = 129.3 lb
Lightest Weight = 98.2 lb

Average Weight = 126.4 lb
Lightest Weight = 105.1 lb
**Figure 7-5:** Third 5 Designs that the Algorithm Found all 20 Times

- **Design 11:**
  - Average Weight = 120.2 lb
  - Lightest Weight = 105.6 lb

- **Design 12:**
  - Average Weight = 119.6 lb
  - Lightest Weight = 109.2 lb

- **Design 13:**
  - Average Weight = 117.3 lb
  - Lightest Weight = 114.9 lb

- **Design 14:**
  - Average Weight = 124.3 lb
  - Lightest Weight = 115.2 lb

- **Design 15:**
  - Average Weight = 123.0 lb
  - Lightest Weight = 117.6 lb
Figure 7-6: Last 2 Designs that the Algorithm Found all 20 Times

maximum displacement from its analysis of 0.901 inches. These deflection values are nearly in agreement.

The fact that the designs that the algorithm found have extremely large deflections needs to be resolved. The reason that this happens has to do with zero stress members. Consider the case where the topology in Figure 7-2 is found by the algorithm with the exception that member DF is present. If joint F is off of the equilibrium value with respect to joint E by $10^{-5}$ inches, then member DF will bear a normal force of 0.014 lb. The design’s maximum displacement is 1.19 inches with this member in place. The design’s maximum displacement without this member in place is 71.2 inches. This member, or another that may have little stress, was present before the analysis. After finding the stresses in the members, this member was considered absent since the normal
stress was below the inputted zero stress. It is up to the judgment of the engineer using this algorithm to recognize when this is happening.

The algorithm also found a cable stay topology in 11 out of the 20 runs with an average weight of 103.5 lb, a maximum weight of 195.0 lb, and a minimum weight 90.58 lb. Seven of the cable stay topologies weighed 90.58 lb, the optimal weight for this topology. This topology is different from design #11 in Figure 7-5. Design #11 has extra members that are connected to joint E. The cables stay topology discussed here has no members attached to joint E. The size variables for the best cable stay design have values of $A_1 = 0.494 \text{ in}^2$, $A_2 = A_3 = 0.25 \text{ in}^2$, and $A_4 = 0.799 \text{ in}^2$. Because no members connect at joint E, its shape value does not matter. Joint F has a vertical coordinate value of 24.9987 inches for the best cables stay design. A linear analysis of this cable stay bridge yields a maximum displacement of 1,320 inches. A linear analysis when $y_E$ is exactly 25 inches yields a maximum deflection of 0.541 inches.

**7.5: Results When the Pier has a High Cost**

Pier member DH was assigned large unit cost values with the same cross sectional properties. The unit cost for these members was assigned to be ten times their unit weight. There are no units for the weight objective in this section since the pier member no longer correlates directly to its weight.

The algorithm was run 20 times with all the same inputs except for the greater unit cost member DH. An arch topology was optimal instead of the suspension topology as previously obtained. The algorithm determined the arch topology in 9 out of the 20 runs (Figure 7-7). This topology was not determined at all in Section 7.4. Eight of the 9 were the optimal cost of 86.4, however one was 146.2. Four runs yielded
suspension topologies as the best design. The algorithm can result in suspension topologies more frequently than arch topologies even though the arch topology may be optimal. This is because the arch topology needs to have two supporting members while the suspension topology needs only one supporting member to avoid excess deflections when the coordinates of joints E and F do not exactly satisfy equilibrium. Fourteen other designs showed up in 20 out of the 20 runs. These are shown in Figures 7-8 through 7-10 along with their lowest and average weights.

The topologies that the algorithm found when the pier cost was low were the best designs. This was not what happened when the pier cost was high. The best designs that the algorithm found are shown in Figures 7-11 and 7-12. The best topologies have similar topologies in that their pier members are deleted as well as their axial support members for the roadway. The topology costs for these several designs are so close, they could be considered a tie when it comes to the cost objective.

Figure 7-7: Overall Lightest Design from the 20 Runs of Section 7.5
Figure 7-8: First 5 Designs that the Algorithm Found all 20 Times

1. Average Cost = 142.3
   Best Cost = 130.7

2. Average Cost = 170.6
   Best Cost = 150.8

3. Average Cost = 163.1
   Best Cost = 151.5

4. Average Cost = 167.9
   Best Cost = 152.0

5. Average Cost = 216.0
   Best Cost = 187.9
Figure 7-9: Second 5 Designs that the Algorithm Found all 20 Times

- **Figure 6:** Average Cost = 221.2  
  Best Cost = 194.3
- **Figure 7:** Average Cost = 254.7  
  Best Cost = 195.4
- **Figure 8:** Average Cost = 207.6  
  Best Cost = 196.5
- **Figure 9:** Average Cost = 214.6  
  Best Cost = 197.4
- **Figure 10:** Average Cost = 206.4  
  Best Cost = 200.0

47
Figure 7-10: Last 4 Designs that the Algorithm Found all 20 Times

The algorithm also found cable stay topologies with the settings in this section. It was found 8 times. The average cost was 247.2. Only 2 of the cable stay topologies were at the optimal cost.
Average Cost = 93.1
Best Cost = 86.4
Found 9 Times

Figure 7-11: First 5 Best Topologies when the Pier had High Cost
Figure 7-12: Second 5 Best Topologies when the Pier had a High Cost
Chapter 8: Bridge Truss Example

The example in this chapter is a 2 dimensional bridge truss. Size, shape, and topology variables are used in this example, along with alternate genes. This example illustrates the usefulness of preventing overlapping members.

8.1: Ground Structure

The ground structure for the Bridge Truss Example is found in Figure 8-1. This ground structure consists of six bays whose members are all linked similarly to the topology variables. The typical bay is shown in Figure 8-2 (Members CF and FI for bay i are members AD and DG for bay i + 1). This structure is considered symmetric, so joints and members on either side of members 6-11 and 11-16 are considered the same. The width of the bays L for this example is 50 inches. The vertical coordinate H of joints 12 through 16 is discussed in 8.2: Shape Inputs.
This example was run 20 times with different random number sequences twice. The first time, overlapping members are allowed. The second time, overlapping members are prohibited. The lightest design from all 20 runs is shown. Some general inputs are listed below.

- Generation Size = 200
- Number of Generations = 200
- Probability of Crossover = 0.1
- Probability of Mutation = 0.1
- Tournament Size = 12
Members are considered to have zero stress if their stress constraint values are less than 3E-5. The stress constraint value has no units.

The deflection limit for this problem is 1 inch in the vertical direction for all joints on the bottom of the truss (2 through 6).

Downward point loads with a magnitude of 5 kips are at joints 2, 4, and 6.

8.2: Shape Inputs
There is only one shape variable used in this example. All of the vertical coordinates of the top joints (12 through 16) are linked to this variable. The vertical coordinate ranges from 1 inch to 20 inches. Joints 7 through 11 are also linked to the one shape variable. However, instead of assuming the shape variable value, the algorithm was fixed so that joints 7 through 11 assume half of the shape variable value.

8.3: Size Inputs
The sizes used in this example are the same as those used in Example 1. Their section properties can be found in Table 8-1. Table 8-2 describes which members are linked to the 4 size variables. The first size variable is linked to all of the internal members. The second is linked to all of the vertical members that border the bays. The third is linked to the bottom members and the fourth to the top members.

- Modulus of elasticity is 29,000 ksi.
- The allowable normal stress is 60 ksi.
- The weight density of the members is 0.28 lb/in$^3$. 
Table 8-1: Available Sizes for Bridge Trus
Example—AISC Standard Weight Steel
Pipe

<table>
<thead>
<tr>
<th>Diameter</th>
<th>Area (in$^2$)</th>
<th>I (in$^4$)</th>
<th>Weight (lb/in)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1/2$</td>
<td>0.25</td>
<td>0.0171</td>
<td>0.071</td>
</tr>
<tr>
<td>$3/4$</td>
<td>0.333</td>
<td>0.037</td>
<td>0.094</td>
</tr>
<tr>
<td>1</td>
<td>0.494</td>
<td>0.0873</td>
<td>0.140</td>
</tr>
<tr>
<td>1 $1/4$</td>
<td>0.669</td>
<td>0.195</td>
<td>0.189</td>
</tr>
<tr>
<td>1 $1/2$</td>
<td>0.799</td>
<td>0.31</td>
<td>0.227</td>
</tr>
<tr>
<td>2</td>
<td>1.07</td>
<td>0.666</td>
<td>0.305</td>
</tr>
<tr>
<td>2 $1/2$</td>
<td>1.7</td>
<td>1.53</td>
<td>0.483</td>
</tr>
<tr>
<td>3</td>
<td>2.23</td>
<td>3.02</td>
<td>0.632</td>
</tr>
<tr>
<td>3 $1/2$</td>
<td>2.68</td>
<td>4.79</td>
<td>0.760</td>
</tr>
<tr>
<td>4</td>
<td>3.17</td>
<td>7.23</td>
<td>0.900</td>
</tr>
<tr>
<td>5</td>
<td>4.3</td>
<td>15.2</td>
<td>1.217</td>
</tr>
<tr>
<td>6</td>
<td>5.58</td>
<td>28.1</td>
<td>1.583</td>
</tr>
<tr>
<td>8</td>
<td>8.4</td>
<td>72.5</td>
<td>2.383</td>
</tr>
<tr>
<td>10</td>
<td>11.9</td>
<td>161</td>
<td>3.375</td>
</tr>
<tr>
<td>12</td>
<td>14.6</td>
<td>279</td>
<td>4.133</td>
</tr>
</tbody>
</table>

Table 8-2: The Size Variables to which members of the typical bay are linked

<table>
<thead>
<tr>
<th>Size Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AH, AE, EI, AF, BG, BE, EH, BI, CD, CE, EG, CH, FG, DI</td>
</tr>
<tr>
<td>2</td>
<td>AD, DG</td>
</tr>
<tr>
<td>3</td>
<td>AB, BC</td>
</tr>
<tr>
<td>4</td>
<td>GH, HI</td>
</tr>
</tbody>
</table>

8.4: Topology Inputs

Most of the bay members are linked to topology variables independently. Three of the topology variables are linked to two members, such as topology variable 1, which is linked to members AB and BC (see Figure 8-2). Topology variables 1 and 2 cannot be set to 0. This means that the top and bottom members cannot be considered absent. Allowing
these members to be absent will waste time since it is highly unlikely that a final design will not include these members. Table 8-3 summarizes which members are linked to the 17 topology variables. Table 8-4 summarizes which members will alternate when an alternate gene (see Chapter 4 Section 3) is set to 0.

Table 8-3: The Topology Variables to which members of the typical bay are linked

<table>
<thead>
<tr>
<th>Topology Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AB, BC</td>
</tr>
<tr>
<td>2</td>
<td>GH, HI</td>
</tr>
<tr>
<td>3</td>
<td>AH</td>
</tr>
<tr>
<td>4</td>
<td>AE</td>
</tr>
<tr>
<td>5</td>
<td>EI</td>
</tr>
<tr>
<td>6</td>
<td>AF</td>
</tr>
<tr>
<td>7</td>
<td>BG</td>
</tr>
<tr>
<td>8</td>
<td>BE</td>
</tr>
<tr>
<td>9</td>
<td>EH</td>
</tr>
<tr>
<td>10</td>
<td>BI</td>
</tr>
<tr>
<td>11</td>
<td>CD</td>
</tr>
<tr>
<td>12</td>
<td>CE</td>
</tr>
<tr>
<td>13</td>
<td>EG</td>
</tr>
<tr>
<td>14</td>
<td>CH</td>
</tr>
<tr>
<td>15</td>
<td>FG</td>
</tr>
<tr>
<td>16</td>
<td>DI</td>
</tr>
<tr>
<td>17</td>
<td>AD, DG</td>
</tr>
</tbody>
</table>
Table 8-4: The Members That Are Mirrored When Alternate Variables Change Value

<table>
<thead>
<tr>
<th>Member</th>
<th>Alternate of Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE</td>
<td>CE</td>
</tr>
<tr>
<td>AF</td>
<td>CD</td>
</tr>
<tr>
<td>AH</td>
<td>CH</td>
</tr>
<tr>
<td>BG</td>
<td>BI</td>
</tr>
<tr>
<td>DI</td>
<td>FG</td>
</tr>
<tr>
<td>EG</td>
<td>EI</td>
</tr>
</tbody>
</table>

8.5: Results

The genetic algorithm was run 20 times for each case, when overlap was allowed and when overlap was prevented. Table 8-5 shows the design variable values of the best designs from each case. Most designs were between 19 inches and 20 inches tall.

The algorithm found a similar best design whether or not overlap has been prohibited (see Figures 8-4 and 8-5). However, this best design was not a very common result for the case where overlap was allowed. Nonduplicated variations of this design result 2 times out of the 20 runs when overlap was allowed. Nonduplicated variations of this design result 33 times out of the 20 runs when overlap is prohibited. There were more than 20 variations because the bays can be oriented in different directions. There are 4 possible variations of this design. Optimal designs are found more consistently when overlap is prevented. The algorithm found lighter designs for each run, on average, when overlap was prevented. The average lightest design from each run averaged 234.0 lb when overlap was prevented and 252.6 lb when overlap was allowed. The designs that the algorithm found most frequently for the case where overlap was allowed are shown in Figures 8-5 and 8-6. The alternate genes did not play a big role in finding the designs in
Figures 8-5 and 8-6. However, they did play a role in the lightest weight designs shown in Figures 8-3 and 8-4. Common topologies are labeled in Figures 8-5 through 8-10.

Table 8-4: Summary of Results From the Runs of Bridge Truss Example

<table>
<thead>
<tr>
<th></th>
<th>Allow Overlap</th>
<th>Prohibit Overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size Values of Overall Best Designs (in²)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Shape Values of Overall Best Designs (in)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>19.14</td>
<td>19.14</td>
</tr>
<tr>
<td>Objective Values--Weights (lb)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall Lightest</td>
<td>232.8</td>
<td>232.8</td>
</tr>
<tr>
<td>Average Lightest from Each Run</td>
<td>252.6</td>
<td>234.0</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>17.8</td>
<td>12.0</td>
</tr>
</tbody>
</table>

Figure 8-3: Lightest Design from the 20 Runs when Overlap Was Allowed

Figure 8-4: Lightest Design from the 20 Runs when Overlap Was Prevented
Figure 8-5: 10 Most Frequently Found Designs when Overlap is Allowed
Average Weight = 305.8 lb  
Lightest Weight = 246.3 lb  
Found 20 Times

Average Weight = 259.1 lb  
Lightest Weight = 252.1 lb  
Found 20 Times--Long Truss

Average Weight = 290.9 lb  
Lightest Weight = 280.9 lb  
Found 20 Times

Average Weight = 255.2 lb  
Lightest Weight = 246.1 lb  
Found 19 Times

Average Weight = 292.1 lb  
Lightest Weight = 281.0 lb  
Found 19 Times

Average Weight = 276.0 lb  
Lightest Weight = 256.0 lb  
Found 15 Times

Average Weight = 511.8 lb  
Lightest Weight = 342.7 lb  
Found 14 Times

Average Weight = 457.2 lb  
Lightest Weight = 285.0 lb  
Found 13 Times

Average Weight = 476.9 lb  
Lightest Weight = 298.0 lb  
Found 13 Times

Average Weight = 252.2 lb  
Lightest Weight = 242.0 lb  
Found 12 Times--Howe Truss

**Figure 8-6: Most Frequently Found Designs when Overlap is Prevented**
Figure 8-7: First 10 Out of 15 Lightest Topologies when Overlap is Allowed
The algorithm found some light topologies as shown in Figures 8-7 and 8-8 when overlap was allowed. The light topologies were not found very frequently. It could be that the algorithm found so many topologies that the best were ties when it came to the weight objective, just as with the pier cost example.

Some familiar topologies were found when overlap was allowed, such as topology 10 in Figure 8-7 and topologies 11, 12, and 14 in Figure 8-8. These were only found 1 or 2 times out of 20 runs. This is not very impressive. A designer should not have to run the algorithm 20 times to find decent results.

The algorithm found some light topologies as shown in Figures 8-9 and 8-10 when overlap was prevented. Many of the lightest topologies were only found in about
half of the 20 runs. These are not the best results, but they are much better than when overlap was allowed.

The alternate genes played more of a role in the lighter designs than the designs that were found more frequently. Topologies 1, 6, 8, and 10 in Figure 8-7 were made possible by alternate genes. Topology 2 in Figure 8-9 and topologies 6, 8, and 12 in Figure 8-10 were made possible by alternate genes. Common topologies such as Howe, Pratt, and a Long Truss were found. They are labeled in the figures.

Figure 8-9: First 5 Out of the Lightest 15 Topologies when Overlap is Prevented
Average Weight = 271.2 lb
Lightest Weight = 247.5 lb
Found 11 Times

Average Weight = 261.3 lb
Lightest Weight = 242.0 lb
Found 12 Times

Average Weight = 265.1 lb
Lightest Weight = 242.9 lb
Found 9 Times

Average Weight = 252.4 lb
Lightest Weight = 242.9 lb
Found 9 Times

Average Weight = 263.4 lb
Lightest Weight = 242.9 lb
Found 9 Times

Average Weight = 258.9 lb
Lightest Weight = 243.9 lb
Found 14 Times--Howe Truss

Average Weight = 271.1 lb
Lightest Weight = 243.5 lb
Found 11 Times--Pratt Truss

Average Weight = 273.7 lb
Lightest Weight = 244.1 lb
Found 2 Times--Warren Truss

Average Weight = 255.2 lb
Lightest Weight = 246.1 lb
Found 19 Times

Average Weight = 305.8 lb
Lightest Weight = 246.3 lb
Found 20 Times

Average Weight = 271.2 lb
Lightest Weight = 247.5 lb
Found 11 Times

Figure 8-10: Last 10 of the Lightest 15 Topologies when Overlap is Prevented
Chapter 9: Transmission Tower Example

The results from the algorithm of this thesis are compared to results of other optimization algorithms. Soh and Yang (1994, 1996), and Vanderplaats (1975) optimized a 25 bar transmission tower space truss. The same tower was analyzed using the Gillman algorithm. The Gillman algorithm did not find lighter designs when topology was not a design variable, but it found lighter designs when topology was a design variable.

9.1: Ground Structure

The ground structure is shown in Figure 9-1. The buckling constraints used by Soh and Yang and Vanderplaats involved a relationship with the cross sectional area. The moment of inertia was set so that the buckling limits would be the same as that set by Soh and Yang because the Gillman algorithm uses Euler’s equation. The moment of inertia used in the Gillman algorithm is given in Equation 9-1.

\[
I = \frac{100.01A^2}{8\pi} \quad \text{Eq. 9.1}
\]
This example was run 20 times consecutively twice. Topology was not a design variable the first time since the other results did not consider topology as a variable. The second time, topology was a design variable to find even lighter designs. Some general inputs are listed below. For a justification for using the probabilities and tournament size, see Appendix.

- Generation Size = 300
- Number of Generations = 300
- Probability of Crossover = 0.1
- Probability of Mutation = 0.1
- Tournament Size = 12
• Members are considered to have zero stress if their stress constraint values are less than 0.001. The stress constraint value has no units.

• There is no deflection limit for this problem.

• Overlap is not prevented.

• Loading conditions are shown in Table 9-1

<table>
<thead>
<tr>
<th>Table 9-1: Loads for Transmission Tower Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Case</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

9.2: Shape Inputs

All joint coordinates are allowed to vary except for joints 1 and 2. Some of the shape variables are dependent on other shape variables as shown in Table 9-2. For example, the x-coordinate value of joint D equals 200 inches minus the value of shape variable 8. This ensures symmetry around the tower. Table 9-3 shows the minimum and maximum values for each shape variable.
Table 9-2: Size Variables for Transmission Tower Example. Joints C - E and G - I have variable values that depend on Shape Variables 2 or 3

<table>
<thead>
<tr>
<th>Joint</th>
<th>Shape Variables</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x-direction</td>
<td>y-direction</td>
<td>z-direction</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>200 - value(10)</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>200 - value(8)</td>
<td>200 - value(10)</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>200 - value(8)</td>
<td>10</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>8</td>
<td>10</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>2</td>
<td>200 - value(9)</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>200 - value(2)</td>
<td>200 - value(9)</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>200 - value(2)</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>J</td>
<td>2</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Table 9-3: Shape Variable Limits for Transmission Tower Example

<table>
<thead>
<tr>
<th>Shape Variable</th>
<th>Minimum Value (inches)</th>
<th>Maximum Value (inches)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>200</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>137.5</td>
<td>137.5</td>
</tr>
<tr>
<td>7</td>
<td>62.5</td>
<td>62.5</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>
9.3: Size Inputs
Table 9-4 shows which members are linked to what size variable. Soh and Yang (1994, 1996) and Vanderplaats (1975) used continuous cross sectional areas ranging from 0.01 to 1.0 in$^2$. This thesis will approximate the continuous size variables with 21 discrete size variables. The first of the size variables for this thesis will have a cross sectional area of 0.01 in$^2$. The next size variable will have a cross sectional area of 0.05 in$^2$. The rest of the variables will have cross sectional areas that increment 0.05 in$^2$. The following is a list of material properties.

- Modulus of elasticity is 10,000 ksi.
- The allowable normal stress is 400 ksi.
- The weight density of the material is 0.1 lb/in$^3$.

<table>
<thead>
<tr>
<th>Size Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AB</td>
</tr>
<tr>
<td>2</td>
<td>AC, BD, BE, AF</td>
</tr>
<tr>
<td>3</td>
<td>AD, BC, BF, AE</td>
</tr>
<tr>
<td>4</td>
<td>CF, DE</td>
</tr>
<tr>
<td>5</td>
<td>CD, EF</td>
</tr>
<tr>
<td>6</td>
<td>FG, CJ, EH, DI</td>
</tr>
<tr>
<td>7</td>
<td>CH, DG, FI, EJ</td>
</tr>
<tr>
<td>8</td>
<td>CG, DH, EI, FJ</td>
</tr>
</tbody>
</table>

9.4: Topology Inputs
For the case where topology is a design variable, Table 9-5 shows the linking of members to topology variables. The topology variable linking is used only to ensure the
same topology on opposing sides of the tower. For example, the right and left hand sides of the tower would have the same topology, and the front and back sides of the tower would have the same topology (from any frame of reference). The front or back side members are not linked to any left or right side members.

Table 9-5: Topology Variables for Transmission Tower Example

<table>
<thead>
<tr>
<th>Topology Variables</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AB</td>
</tr>
<tr>
<td>2</td>
<td>AC, BE</td>
</tr>
<tr>
<td>3</td>
<td>BD, AF</td>
</tr>
<tr>
<td>4</td>
<td>AD, BF</td>
</tr>
<tr>
<td>5</td>
<td>BC, AE</td>
</tr>
<tr>
<td>6</td>
<td>CF, DE</td>
</tr>
<tr>
<td>7</td>
<td>CD, EF</td>
</tr>
<tr>
<td>8</td>
<td>FG, EH</td>
</tr>
<tr>
<td>9</td>
<td>CJ, DI</td>
</tr>
<tr>
<td>10</td>
<td>CH, FI</td>
</tr>
<tr>
<td>11</td>
<td>DG, EJ</td>
</tr>
<tr>
<td>12</td>
<td>CG, EI</td>
</tr>
<tr>
<td>13</td>
<td>DH, FJ</td>
</tr>
</tbody>
</table>

9.5: Results

Table 9-6 shows the results of this example. The number of iterations it took to reach the designs of this thesis is larger than the other designs. The Gillman algorithm did not find designs with weights that were very close to that of Soh and Yang and Vanderplaats. This could be due to the fact that discrete sizing was used. The case for which topology was a design variable does not present a valid comparison because Soh and Yang (1994, 1996), and Vanderplaats (1975) did not use topology design variables.
Table 9-6: Comparison of Weights and Iterations Among the Works

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Overall Lightest</td>
<td>133.5</td>
<td>137.2</td>
<td>132.3</td>
</tr>
<tr>
<td></td>
<td>Average Lightest</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td># Iterations</td>
<td>171</td>
<td>47</td>
<td>38</td>
</tr>
</tbody>
</table>

Figures 9-2 and 9-3 show the best resulting topologies from both cases. Ten members were absent from the result where topology was a design variable (members BD, AF, CF, DE, FG, CJ, EH, DI, CG, EI). This design had a maximum deflection of 69.7 inches. This is not a problem for this example because deflection was not limited. Maximum displacement for the lightest design when topology was not a design variable is 1.7 inches. Tables 9-7 through 9-9 give more information on the best designs.

Figure 9-2: Lightest Topology for Transmission Tower Example without Topology as a Variable

Figure 9-3: Lightest Topology for Transmission Tower Example with Topology as a Variable
Table 9-7: Joint Coordinates for Lightest Design When Topology is not a Design Variable

<table>
<thead>
<tr>
<th>Joint</th>
<th>Joint Coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x-direction</td>
</tr>
<tr>
<td>A</td>
<td>137.5</td>
</tr>
<tr>
<td>B</td>
<td>62.5</td>
</tr>
<tr>
<td>C</td>
<td>71.596</td>
</tr>
<tr>
<td>D</td>
<td>128.404</td>
</tr>
<tr>
<td>E</td>
<td>128.404</td>
</tr>
<tr>
<td>F</td>
<td>71.596</td>
</tr>
<tr>
<td>G</td>
<td>84.588</td>
</tr>
<tr>
<td>H</td>
<td>115.412</td>
</tr>
<tr>
<td>I</td>
<td>115.412</td>
</tr>
<tr>
<td>J</td>
<td>84.588</td>
</tr>
</tbody>
</table>

Table 9-8: Joint Coordinates for Lightest Design When Topology is a Design Variable

<table>
<thead>
<tr>
<th>Joint</th>
<th>Joint Coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x-direction</td>
</tr>
<tr>
<td>A</td>
<td>137.5</td>
</tr>
<tr>
<td>B</td>
<td>62.5</td>
</tr>
<tr>
<td>C</td>
<td>69.986</td>
</tr>
<tr>
<td>D</td>
<td>130.014</td>
</tr>
<tr>
<td>E</td>
<td>130.014</td>
</tr>
<tr>
<td>F</td>
<td>69.986</td>
</tr>
<tr>
<td>G</td>
<td>75.203</td>
</tr>
<tr>
<td>H</td>
<td>124.797</td>
</tr>
<tr>
<td>I</td>
<td>124.797</td>
</tr>
<tr>
<td>J</td>
<td>75.203</td>
</tr>
</tbody>
</table>
Table 9-9: Size Variables of the Lightest Designs from when Topology was a Design Variable and When Topology was not a Design Variable

<table>
<thead>
<tr>
<th>Size Variable</th>
<th>Areas When Topology Was not a Design Variable</th>
<th>Areas When Topology Was a Design Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>2</td>
<td>0.55</td>
<td>0.7</td>
</tr>
<tr>
<td>3</td>
<td>0.9</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>0.15</td>
<td>--</td>
</tr>
<tr>
<td>5</td>
<td>0.25</td>
<td>0.15</td>
</tr>
<tr>
<td>6</td>
<td>0.3</td>
<td>--</td>
</tr>
<tr>
<td>7</td>
<td>0.75</td>
<td>0.55</td>
</tr>
<tr>
<td>8</td>
<td>0.75</td>
<td>0.7</td>
</tr>
</tbody>
</table>
Chapter 10: Rajan’s Truss Example

This example will involve shape, size and topology design variables. Rajan optimized a 14-node truss with shape, size and topology as the design variables (Rajan 1995). This chapter compares the results obtained from Gillman’s algorithm to that of Rajan’s algorithm. The results indicate that Gillman’s algorithm provides lighter designs.

10.1: Ground Structure

Figure 10-1: Ground Structure for Rajan’s Truss Example

Figure 10-1 shows the ground structure for Rajan’s Truss Example. This example was run 20 times consecutively twice. For the first case, overlapping members was allowed. For the second case, overlapping members was prevented. Some general inputs are listed below. For a justification for using the probabilities and tournament size, see Appendix.
• Generation Size = 200
• Number of Generations = 200
• Probability of Crossover = 0.1
• Probability of Mutation = 0.1
• Tournament Size = 12
• Members are considered to have zero stress if their stress constraint values are less than 1E-5. The stress constraint value has no units.
• Deflection is limited to 0.01 m in the vertical direction for joints C, E, G, I, and K.
• Loading conditions are shown in Table 10-1

Table 10-1: Loads for Rajan’s Truss

<table>
<thead>
<tr>
<th>Load Case</th>
<th>Joint #</th>
<th>Direction</th>
<th>Load (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>y</td>
<td>-3E+06</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>y</td>
<td>-3E+06</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>y</td>
<td>-3E+06</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>y</td>
<td>-3E+06</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>y</td>
<td>-3E+06</td>
</tr>
</tbody>
</table>

10.2: Shape Inputs

There are 4 shape variables. Vertical coordinates for the top joints (B, D, F, H, J, L, N) can vary between 2 m and 8 m. The structure is symmetric about middle member GH. Therefore joints B and N will have the same vertical coordinate, joints D and L will have the same vertical coordinate, and joints F and J will have the same vertical coordinate. Joint H is allowed to vary as well.

10.3: Size Inputs

Rajan used four binary genes to represent continuous size variables. Therefore, the available cross-sectional areas are (0.01, 0.016, 0.022, 0.028, 0.034, 0.04, 0.046,
Rajan used an allowable stress constraint for compression instead of using an equation to constrain buckling. The following list summarizes material properties. Table 10-2 shows which members are linked to the size variables.

- Modulus of elasticity is 2.1E11 Pa.
- The allowable compressive stress is 1.04E8 Pa.
- The allowable tensile stress is 1.3E8 Pa.
- The maximum displacement in the vertical direction is 0.01 m.

### Table 10-2: Size Variables and the Members Linked to Them for Rajan’s Truss Example

<table>
<thead>
<tr>
<th>Size Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BD, DF, FH, HJ, JL, LN</td>
</tr>
<tr>
<td>2</td>
<td>AC, CE, EG, GI, IK, KM</td>
</tr>
<tr>
<td>3</td>
<td>BC, AD, DE, CF, FG, EH, HI, GJ, JK, IL, LM, KN</td>
</tr>
<tr>
<td>4</td>
<td>AB, CD, EF, GH, IJ, KL, MN, AF, CH, BE, DG, AH, GL, IN, JM, HK, HM</td>
</tr>
</tbody>
</table>

### 10.4: Topology Inputs

The structure is symmetric with respect to topology variables as well. Table 10-3 indicates which topology variables the members are linked.
### Table 10-3: Topology Variables and the Members Linked to Them for Rajan’s Truss Example

<table>
<thead>
<tr>
<th>Topology Variable</th>
<th>Linked Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AC, CE, EG, GI, IK, KM</td>
</tr>
<tr>
<td>2</td>
<td>AB, MN</td>
</tr>
<tr>
<td>3</td>
<td>BD, LN</td>
</tr>
<tr>
<td>4</td>
<td>BC, KN</td>
</tr>
<tr>
<td>5</td>
<td>AD, LM</td>
</tr>
<tr>
<td>6</td>
<td>CD, KL</td>
</tr>
<tr>
<td>7</td>
<td>DF, JL</td>
</tr>
<tr>
<td>8</td>
<td>DE, IL</td>
</tr>
<tr>
<td>9</td>
<td>CF, JK</td>
</tr>
<tr>
<td>10</td>
<td>EF, IJ</td>
</tr>
<tr>
<td>11</td>
<td>FH, HJ</td>
</tr>
<tr>
<td>12</td>
<td>FG, GJ</td>
</tr>
<tr>
<td>13</td>
<td>EH, IJ</td>
</tr>
<tr>
<td>14</td>
<td>GH</td>
</tr>
<tr>
<td>15</td>
<td>DG, GL</td>
</tr>
<tr>
<td>16</td>
<td>BE, IN</td>
</tr>
<tr>
<td>17</td>
<td>AF, JM</td>
</tr>
<tr>
<td>18</td>
<td>CH, HK</td>
</tr>
<tr>
<td>19</td>
<td>AH, HM</td>
</tr>
</tbody>
</table>

### 10.5: Results

The Gillman algorithm took 241 generations to find the best volume when allowing overlap. It took 247 generations to find the best volume when overlap was prevented. Rajan’s results took 96 generations with a generation size of 70. Figure 10-2 shows Rajan’s best design. There are great savings in volume even though this algorithm took more iterations. There was a 12% savings in volume when overlapping members was allowed. There was a 7.16% savings in volume when overlapping members was prevented.
Table 10-4 summarizes the results for this example. The designs that resulted when overlap was prevented have slightly more volume than the resulting designs when overlap was allowed. However, 12 out of the 20 runs yielded best designs that had lower volume than Rajan’s best design. The advantage to preventing overlap in this example is that more buildable designs were found. Figures 10-3 and 10-4 present the best designs using Gillman’s algorithm when overlap is and is not allowed, respectively. A more aesthetically pleasing design was obtained when overlap was not allowed.

Table 10-4: Summary of Results for Rajan’s Truss Example

<table>
<thead>
<tr>
<th>Variable</th>
<th>Rajan</th>
<th>Present Work</th>
<th>Present Work</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Allow Overlap</td>
<td>Prohibit Overlap</td>
</tr>
<tr>
<td>Size Variables of Best Design (m²)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.053</td>
<td>0.052</td>
<td>0.058</td>
</tr>
<tr>
<td>2</td>
<td>0.016</td>
<td>0.016</td>
<td>0.022</td>
</tr>
<tr>
<td>3</td>
<td>0.022</td>
<td>0.028</td>
<td>0.04</td>
</tr>
<tr>
<td>4</td>
<td>0.034</td>
<td>0.016</td>
<td>0.034</td>
</tr>
<tr>
<td>Shape Variables of Best Design (m)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2.33</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td>4.31</td>
<td>5.21</td>
<td>5.08</td>
</tr>
<tr>
<td>3</td>
<td>6.68</td>
<td>6.50</td>
<td>6.66</td>
</tr>
<tr>
<td>4</td>
<td>7.66</td>
<td>6.85</td>
<td>7.09</td>
</tr>
<tr>
<td>Objective Values (m³)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Volume</td>
<td>5.10</td>
<td>4.61</td>
<td>4.70</td>
</tr>
<tr>
<td>Average Best Volume</td>
<td>--</td>
<td>4.99</td>
<td>5.84</td>
</tr>
</tbody>
</table>
Figure 10-2: Rajan’s Best Result for Rajan’s Truss Example

Figure 10-3: Gillman’s Algorithm Best Result for Rajan’s Truss Example when overlap is allowed

Figure 10-4: Gillman’s Algorithm Best Result for Rajan’s Truss Example when overlap is prevented
Chapter 11: Conclusion

The Gillman algorithm simultaneously optimized shape, size and topology variables in trusses. The algorithm was successfully executed in several examples. Three important conclusions were verified.

First, the Gillman algorithm found traditionally recognized topologies. Suspension and cable stay topologies were found in the pier cost example when the pier was set to a low cost. Arch, suspension, and cable stay topologies were found in the pier cost example when the pier was set to a high cost. Recognizable trusses were found in the bridge truss example such as Howe, Pratt, Warren, and Long Truss topologies.

Second, constraining designs for aesthetic purposes was successfully executed. More aesthetically pleasing designs were found in the bridge truss example and Rajan’s truss example when topologies with overlapping members were prevented. It may be possible to employ other constraints in genetic algorithms for aesthetic purposes.

Third, Gillman’s algorithm found lighter designs than algorithms from literature. Gillman’s algorithm did not find lighter designs than Soh and Yang (1994, 1996) and Vanderplaats (1975) with size and shape variables. Gillman’s algorithm found much lighter designs when topology variables were used along with the shape and size variables, although Soh and Yang and Vanderplaats did not use topology variables. This demonstrates the effectiveness of topology variables. Gillman’s algorithm found lighter
designs than Rajan’s (1995) algorithm. Rajan’s algorithm also simultaneously
implemented shape, size, and topology variables. Rajan used binary representation as
opposed to value representation, which is used in Gillman’s algorithm.
Appendix
Appendix

The settings of the algorithm can affect the designs that result. Using the right settings is crucial for the algorithm to find the optimum designs. A short study was conducted to determine some good settings to use. Two tests were conducted. One test found the best settings of probability of crossover and probability of mutation. The other test found the best tournament size. A ground structure using shape, size, and topology variables was optimized with these settings. For each setting, the algorithm was run 30 times. Table A shows the results for the first test. This test showed that the best setting was a probability of crossover of 0.1 and a probability of mutation of 0.1 for this example. Figure A shows the results for the second test. For the second test, probability of crossover and probability of mutation were set to 0.1. The Figure shows that the best tournament size for this particular structure is 12.

<table>
<thead>
<tr>
<th>Probability of Mutation</th>
<th>Probability of Crossover</th>
<th>0.01</th>
<th>0.1</th>
<th>0.3</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td></td>
<td>104.1</td>
<td>71.9</td>
<td>68.0</td>
<td>66.2</td>
<td>69.4</td>
<td>73.0</td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td>64.8</td>
<td>60.2</td>
<td>62.8</td>
<td>63.6</td>
<td>64.6</td>
<td>67.0</td>
</tr>
<tr>
<td>0.3</td>
<td></td>
<td>61.9</td>
<td>61.9</td>
<td>63.9</td>
<td>66.2</td>
<td>67.8</td>
<td>68.1</td>
</tr>
<tr>
<td>0.6</td>
<td></td>
<td>68.7</td>
<td>68.8</td>
<td>68.6</td>
<td>69.1</td>
<td>68.5</td>
<td>68.7</td>
</tr>
<tr>
<td>0.8</td>
<td></td>
<td>68.9</td>
<td>69.8</td>
<td>69.1</td>
<td>68.9</td>
<td>68.7</td>
<td>68.7</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>87.7</td>
<td>86.6</td>
<td>86.8</td>
<td>85.7</td>
<td>85.1</td>
<td>86.0</td>
</tr>
</tbody>
</table>
This does not mean that these are the best settings for any ground structure. Conducting a study for any ground structure would be too timely. This study gives a good idea about which settings may produce good results. It was assumed that these settings would be good for the examples in this thesis.

![Graph](image)

**Figure A**: Results of Test to Determine the Best Tournament Size
References


