# CONCEPTUAL DESIGN OF LONG-SPAN TRUSSES USING MULTI-STAGE HEURISTICS

A Thesis

by

PRANAB AGARWAL

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2005

Major Subject: Civil Engineering

# CONCEPTUAL DESIGN OF LONG-SPAN TRUSSES USING MULTI-STAGE HEURISTICS

A Thesis

by

PRANAB AGARWAL

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

May 2005

Major Subject: Civil Engineering

# ABSTRACT

Conceptual Design of Long-Span Trusses
Using Multi-Stage Heuristics. (May 2005)
Pranab Agarwal, B.E., Delhi College of Engineering (India)
Chair of Advisory Committee: Dr. Anne M. Raich

A hybrid method that addresses the design and optimization of long-span steel trusses is presented. By utilizing advancements in present day computing and biologically inspired analysis and design, an effort has been made to automate the process of evolving optimal trusses in an unstructured problem domain. Topology, geometry and sizing optimization of trusses are simultaneously addressed using a three stage methodology.

Multi-objective genetic algorithms are used to optimize the member section sizes of truss topologies and geometries. Converting constraints into additional objectives provides a robust algorithm that results in improved convergence to the pareto-optimal set of solutions. In addition, the pareto-curve plotted based on how well the different objectives are satisfied helps in identifying the trade-offs that exist between these objectives, while also providing an efficient way to rank the population of solutions during the search process.

A comparison study between multi-objective genetic algorithms, simulated annealing, and reactive taboo search is conducted to evaluate the efficiency of each method with relation to its overall performance, computational expense, sensitivity to initial parameter settings, and repeatability of finding near-global optimal designs.

The benefit of using a three stage approach, and also implementing the entire model on parallel computers, is the high level of computational efficiency that is obtained for the entire process and the near-optimal solutions obtained. The overall efficiency and effectiveness of this method has been established by comparing the truss design results obtained using this method on bridge and roof truss benchmark problems with truss designs obtained by other researchers. One of the salient features of this research is the large number of optimal trusses that are produced as the final result. The range of designs available provides the user with the flexibility to select the truss design that best matches their design requirements. By supporting human-computer interactions between these stages, the program also incorporates subjective aesthetic criteria, which assist in producing final designs in consonance with the user’s requirements.

## DEDICATION

This thesis is dedicated to my family and friends for their unswerving support and belief in my abilities.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER I

# INTRODUCTION

Computer programs that aid professional engineers and students in performing structural analysis and optimization are prevalent across engineering disciplines and their use has become essential in the practice of engineering. In contrast, computer programs that aid designers in performing conceptual design are scarce even though it is recognized that conceptual design offers the best opportunity for designers to propose efficient and innovative solutions. Synthesizing design alternatives during conceptual design, however, requires expanding the number and diversity of design alternatives examined. Instead of working with a single structural topology and geometry, a wide range of design alternatives with different topology and geometry (structural configurations) must be examined within a single problem domain.

In this research effort, a design system that creates, evaluates, and optimizes truss design alternatives is developed. The system enhances the existing capabilities of designers in designing economical structures by evaluating a broad range of possible truss topologies and geometries. Human designers often must limit the time spent exploring for more economical design alternatives since it can be very costly (Grierson 2001). Currently the design and optimization of complex truss systems using commercially available structural engineering software is limited to sizing optimization of members and is also typically very time consuming. The quality of the designs obtained also depends significantly on the designer's experience. This research effort stems from the latest advancements in computational technology and biologically inspired design and optimization in order to address the need for automating the process of designing truss structural systems.

---

This thesis follows the style and format of the *Journal of Structural Engineering*.

This research is motivated by the need to reduce the overall computational design time, while also being able to produce design alternatives that are beyond the established conventions of the designer. A computational design program can provide the ideal environment for exploring for more efficient designs: "Computers, not yet able to feel embarrassment or peer pressure, are not afraid to try off-the-wall ideas. Ideas are just ideas; the more the merrier" (Benyus 1997). Therefore, one of the objectives of this research is to develop a computational design system that supports a more intensive exploration of the design space. By integrating the concepts of artificial intelligence and biological evolution in optimization creates a system that is robust and computationally efficient. So in much the same way as human beings are a better product of their ancestral monkeys over several hundred generations, the structures that are produced over several generations perform better than their reproducing ancestors.

In many cases, the optimality of the design selected also depends significantly on the subjective preferences of designers, including aesthetic criteria. Since aesthetic criteria are difficult to capture in the form of mathematical equations that can be used as objectives or constraints, another mechanism must be introduced for capturing this design information. The design method developed allows users to select specific designs for further optimization at several stages of the overall optimization process. These interactions allow the user to impose their design preferences regarding the truss topology and geometry during optimization. Exploiting parallel computing for computational efficiency is another salient feature of this work.

# CHAPTER II

# OBJECTIVES AND SCOPE

A hybrid structural design and optimization methodology that combines the strengths of heuristic methods, local search techniques, and parallel computing is developed to evolve optimal truss systems in this research effort. The primary objective is to evolve near-optimal or optimal structural systems. The program developed provides engineers with a tool that is capable of satisfying user-defined design criteria for a specific design space, while also minimizing the computational time required.

The process derives its strength by working in an unstructured problem domain. The program evolves hundreds of stable truss topologies within the bounds specified by the user. These truss topologies then undergo shape optimization using one of three modern heuristic techniques: multi-objective genetic algorithms, simulated annealing and reactive taboo search. An additional mutation of the nodal locations is investigated to further refine the near-optimal truss designs obtained after shape optimization.

This research supports conceptual design by concurrently addressing topology, size and geometry optimization. The three-stage methodology developed in this research helps in expediting the entire process of design and optimization. This work has a comparable computational efficiency than similar research efforts that address these three objectives simultaneously. By providing a wide range of alternatives to the designer, it supersedes the work done by previous researchers. The hierarchy of developing stable truss systems, followed by sizing optimization, with a final fine-tuning of the configuration, provides an effective solution to designing both bridge and roof truss systems.

Although the main focus is placed on using multi-objective genetic algorithms, this research also explores the application of simulated annealing and reactive taboo search for sizing optimization. All three of these methods provide a balance between exploration and exploitation during optimization, and the results obtained are in general, independent of the initial starting point. The main benefit of using these methods over non-linear programming techniques is their efficiency in escaping the local optimal and ability to attain the global optimum with minimal computation time and expense. A comparison study is also conducted to evaluate the efficiency of each method with relation to its overall performance, computational expense, sensitivity to initial parameter settings, and repeatability of finding near-global optimal designs.

To evaluate the performance of the proposed methodology, the truss designs obtained using the hybrid method are compared to designs obtained by other researchers on a benchmark bridge truss design. The truss designs produced using the proposed hybrid method are more optimal than the truss designs reported by other researchers while still providing a comparable level of search efficiency. Currently, a problem definition that may be considered as a benchmark problem in the area of roof truss design does not exist. Therefore, in this research a new roof truss benchmark problem is proposed. The performance of the new hybrid optimization method on evolving near-global optimal designs for this problem is also investigated.

In addition, the implementation of the entire model on a four node parallel computer platform promises to significantly reduce the total computational time required for design. By supporting human-computer interactions at various stages, this program also allows the final design to be produced in consonance with the engineer's design criteria.

# CHAPTER III

# LITERATURE REVIEW

## LAYOUT OPTIMIZATION

Layout optimization of truss systems refers to producing trusses with minimal weight and primarily satisfying the specified stress, slenderness and displacement criteria. According to the literature, it is classified into three main categories:

1. Sizing Optimization: The design variables are the sizing parameters associated with the finite-element model, such as the cross-sectional areas of the truss members (Rajan 1995). These areas might be considered to be continuous or discrete variables. Continuous members are relatively easy to optimize, and extensive work has been done using traditional optimization for continuous search domains. Discrete members refer to distinct shapes (e.g. WF, WT and Channels) that have independent values of area, inertia and radius of gyration relative to other members. Optimization of discrete members is performed well using modern heuristic techniques.
2. Geometry Optimization: The shape of the structure is taken as the design variable. This is primarily done by changing the nodal locations and/ or the loads and displacements associated with particular nodes.
3. Topology Optimization: This refers to the placement of the nodes and members relative to each other. Along with this, topology optimization also governs the number of nodes that actually exist in the structure along with their support conditions.

## HISTORICAL EVOLUTION

Traditional methods of optimization were the focus of research during the late seventies and eighties, and a number of mathematical optimization techniques were developed for truss optimization. Most of the research during this period was focused on fixed structural topology and geometry and only considered sizing of member dimensions to meet the design requirements. Berke and Khot (1987) used the optimality criteria method and Schmit (1981) used a mathematical programming approach for optimization. Queau and Trompette (1980) and Bennett and Botkin (1985) addressed sizing optimization for two and three dimensional structures. Templeman and Yates (1983) suggested a method for discrete optimization using segmental members. A combinatorial optimization approach using branch and bound algorithms was studied by John and Ramakrishnan (1987) for discrete structural truss optimization.

The concept of using the principles of biological evolution for analysis and design can be traced back to the work of Rechenberg (1965), although genetic algorithms by themselves gained prominence through the work of Holland (1975). Goldberg (1989) researched the applicability of genetic algorithms to optimization problems, which started the era of research in this area.

## SIZING, GEOMETRY AND TOPOLOGY OPTIMIZATION

Rajeev and Krishnamoorthy (1992) and Jenkins (1992) were among the first researchers to successfully implement simple genetic algorithms (GA) on structural optimization problems. The former explored the application of GAs for truss sizing optimization, whereas the latter used GA for shape optimization of plane frames. In both research efforts, the fitness of each design was assessed using an objective function and any violations of the constraints was applied by penalizing the assigned fitness.

Other researchers tried to provide the ability to perform limited topology and geometry optimization from different perspectives during this period. Hajela and Lee (1995) used a two step design methodology. After achieving kinematic stability in Stage I, member sizing was carried out in the next step. This method allowed for the removal or addition of structural members during optimization based on a defined ground structure. Fitness sharing was also used as an effective technique to maintain the diversity in the population and curtail the need for increasing population sizes for longer string lengths. Roston and Sturges (1995) advocated the combination of formal grammars with genetic programming, which they referred to as Genetic Design. Their method presented the designer with an array of viable design alternatives. Gage et al. (1995) developed a variable-complexity genetic algorithm procedure for topological design. The topology optimization using genetic algorithms was followed up by a gradient based optimizer to size the members. This research effort also introduced the variable length GA and the use of cut-splice crossover operation. Sizing, geometry and topology optimization were simultaneously addressed by Rajan (1995) using genetic algorithms. A simple GA was used to obtain optimal trusses starting with a defined ground structure. Exception handling was performed to remove unstable structures, structures with no structural deformation, zero force members. The method was able to perform restarts to find better solutions, which increased the efficiency of the overall search process.

## UNSTRUCTURED PROBLEM DOMAIN

Shrestha and Ghaboussi (1998) researched a partially unstructured design domain problem for evolving optimal structural truss designs. The sizing, geometry and topological aspects of design were addressed simultaneously. Each string encoded a fixed number of nodal and member information. For each node, it encoded an on/off switch, the co-ordinates and the support type. For each member, it encoded an on/off switch, the sector priority, member type and connection type. Although this research was capable of developing near-optimal trusses for the 70 meter by 10 meter design problem domain, the main drawback was the considerable computational expense.

Soh and Yang (2000) proposed a fuzzy logic integrated genetic programming approach for structural optimization and design. The loadings and member strengths were represented as linguistic variables instead of discrete values. Hence, this methodology overcame the weakness of crisp sets, and removed the boundary that separated members from non-members. This approach was implemented on a benchmark 10-bar truss problem and the 70 meter by 10 meter truss design problem domain researched in the current research effort.

## REAL-ENCODED GENETIC ALGORITHMS

The concept of basic and non-basic nodes was used by Deb and Gulati (2001) to research the address size, geometry and topology optimization of trusses. Real coded GAs, along with simulated binary crossover and a parameter-based mutation operator, were used to produce optimal trusses and also to reduce the computational time. A similar effort was made by Azid et al. (2002) using real coded GAs for optimizing two dimensional and space trusses. All three aspects of sizing, geometry and topology optimization were addressed. Jenkins (2002) used a decimal encoded GA with the omission of the crossover operator. Instead, an intelligent mutation strategy for adding or subtracting an integer step from the current coded value of the variable being mutated was used.

## HYBRID TECHNIQUES

By the late nineties, a number of hybrid approaches using modified genetic algorithms gained prominence. Rajeev and Krishnamoorthy (1997) proposed a strategy to automatically arrive at appropriate lower bound indices for each design variable along with using a variable length genetic algorithm. The topologies that could not result in optimal solutions died off in early generations, and after a few generations the population was found to be composed of individuals of the same length.

Jenkins (1997) proposed space condensation heuristics and adaptive controls to speed up the process of optimization. String partitioning and string replacement were used as effective controls to improve the population. A record of selections was maintained by the algorithm for those individuals near the maximum population fitness positively, and negatively for those near the minimum fitness. This record was used to recognize clusters, which were used to decrease the population size.

Yeh (1999) proposed a hybrid GA that combined the concept of survival of the fittest with the concept of adaptation. The GA was combined with a fully stressed design (FSD) optimality criterion, which was able to achieve a higher speed of convergence and more stability in comparison to using a GA alone. FSD consists of a iterative process for analysis and a redesign rule. If an analysis shows that a particular member is over-stressed, then the redesign rule increases the size of that member to reduce the stress in that member.

Raich and Ghaboussi (2000) used an implicitly redundant representation (IRR) of the GA string to produce optimal frames. The IRR GA allowed the representation of a variable number of location independent parameters, which overcame the fixed parameter limitations of standard GAs.

Three strategies were discussed and compared by Ryoo and Hajela (2004) for implementing variable length individuals in topology optimization problems. They compared inter-species and intra-species crossover with micro-GAs on a set of benchmark problems.

An object-oriented framework for genetic algorithms was developed by Krishnamoorthy et al. (2002), which consisted of a core GA library consisting of all genetic operators having an interface to a generic function. This was developed to act as a learning aid as well as for solving optimization problems.

## MULTI-OBJECTIVE OPTIMIZATION

During the last few decades, multi-objective optimization has also been discussed by a number of researchers who used different approaches to obtain feasible solutions. Some of these were VEGA (Schaffer 1985), lexicographic ordering (Rao 1984) and weighted sum (Hajela and Lin 1992). The concept of pareto-optimality and addressing the trade-off between various objectives gained prominence very late in the nineties, although the basic concepts were developed as early as 1906 by Pareto.

In 1997, Cheng and Li (1997) used multi-objective genetic algorithms (MOGA) to successfully address the trade-off between weight and strain energy for a 72-bar space truss. Along with selection, crossover and mutation, a niche and a pareto-set filter were used to improve their results.

Coello and Christiansen (1999) used a specialized min-max optimization approach to transform the multi-objective problem into several single objective optimization problems that were easier and faster to solve. Ruy et al. (2001) proposed a hybrid MOGA that generated promising topologies with consideration to the multi-objective environment. Following the selection of topologies, sizing and geometry optimization was carried out. Reynolds and Azarm (2002) used a hybrid MOGA that combined a standard MOGA with heuristics specifically tailored to address the deficiencies in multi-objective design for trusses. An ‘advance pareto’ technique was able to generate more-optimal solutions with respect to all objectives. The ‘search between’ and the ‘extend pareto’ techniques were able to increase the number of unique solutions while improving the distribution along the pareto frontier.

Deb’s book (2001) provides a detailed description of implementing multi-objective optimization using evolutionary algorithms.

## OTHER HEURISTIC TECHNIQUES

While genetic algorithms stand out as a very powerful heuristic for solving optimization problems, many researchers have also tried other heuristic techniques for structural optimization. Most of these also try to simulate the natural environment. Among these were ant-colony optimization (Dorigo et al. 1999, Colorni et al. 1992) and swarm optimization (Kennedy and Eberhart 1995). The former tries to imitate the nature of ants in converging to the best path for traveling by depositing pheromone trails for other ants to follow. The latter was inspired by the behavior of flocks of birds and their communication for evading predators and identifying the sources of food.

Other prominent heuristic techniques are simulated annealing, which bases its methodology on the manufacturing processes of metals, and taboo search, which is an improved probabilistic version of a local hill climber algorithm. Reddy and Cagan (1995) and Shea et al. (1997) have used simulated annealing for topology and shape optimization of trusses. In addition to resizing of members, they used a shape grammar to perform limited topology optimization. Similar work for truss optimization was carried out by Cai and Cheng (1998), and Hasancebi and Oguzhan (2002) using simulated annealing.

Bennage and Dhingra (1995) used taboo search for truss topology optimization. Bland (1995) used it in a discrete-variable optimization algorithm and established results on benchmark problems for planar and space trusses. An improved version called the reactive taboo search was used by Hamza et al. (2003) for optimizing the sizes, geometry and topology of N-shaped roof trusses.

A detailed description of multi-objective genetic algorithms can be found in Chapter VI. In addition, the implementation details of simulated annealing and taboo search are also discussed in Chapter VI.

# CHAPTER IV

# HYBRID DESIGN METHODOLOGY DESCRIPTION

## MULTI-STAGE HEURISTIC APPROACH

A heuristic is defined as a problem solving technique in which the most appropriate solution of several found is selected at successive stages of a program for further refinement. It provides a guidance to reach the most appropriate solution, but in the end is not infallible or fully proven. All the heuristic methods implemented in this research have a probabilistic nature.

In the proposed research, the process of optimizing the size, geometry and topology of roof and bridge trusses in an unstructured problem domain is divided into three stages. The "best" results obtained in each stage are passed onto the next stage in accordance with user's preferences. "Best" may be most-optimal with respect to the stated design objectives or due to being selected by the designer as meeting the unstated aesthetic criteria. The most-optimal solution produced during Stage II and III in each generation is passed onto the following generations for further improvement. Hence, this research can be best explained as a hybrid approach that implements heuristics in several stages to perform a search process.

Stage I develops a broad range of truss topologies in the user-specified design domain. Stage II uses one of three heuristic techniques for sizing optimization. Finally, Stage III is responsible for geometry optimization. Design details including the topology, member properties and loading information can be exported to commercial softwares after Stage II and/or Stage III. Separate data files are written after each stage, which are used by an Open GL viewer to allow the user to review the current set of design solutions. To aid in presenting the designs to the user, the generated topologies are sorted in ascending order of their total height or the total number of members. Another file is used to extract the data required to plot the Pareto-optimal surface that visually shows the trade-off that occurs between the different objectives for truss optimization. A front-end utility was developed to help the user to easily implement these stages. Figure

1 shows the front-end interface developed for the Windows operating environment. Along with controlling the hybrid optimization program, it also helps in viewing the truss designs using Open GL viewer and in exporting the design information to commercial structural analysis programs for further analysis and integration into other structural systems.

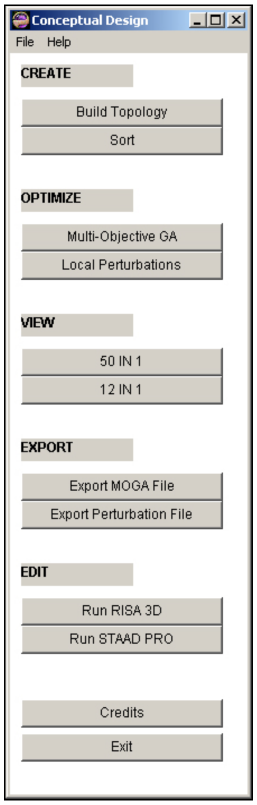

**Fig. 1. Front-end utility for windows operating system**

## IMPLEMENTATION DETAILS

The entire project is written in Visual C++. For implementing the parallel process on a Unix based parallel platform, the Message passing interface library of C++ was used. The front end for the stand alone utility has been made using WIN32 programming. All other modules are DOS based C++ programs.

The flowchart shown in Figure 2 provides the schematic representation of the proposed hybrid optimization method developed in this research implemented on parallel machines. The entire process is implemented on four computer nodes in parallel. Stage I and III are executed on the Master node, while Stage II (which is the most time-consuming step) is implemented simultaneously on three computer nodes. The objective of the parallel implementation is to minimize the total computational time.

The program developed is formulated to cater to human-computer interactions. Among other control features, the user can influence the overall complexity, shape and sections used for designing the truss. In this way the program can design trusses that better match with the user's design requirements. There is an obvious trade-off, however, between the total computational time and better meeting the design requirements.

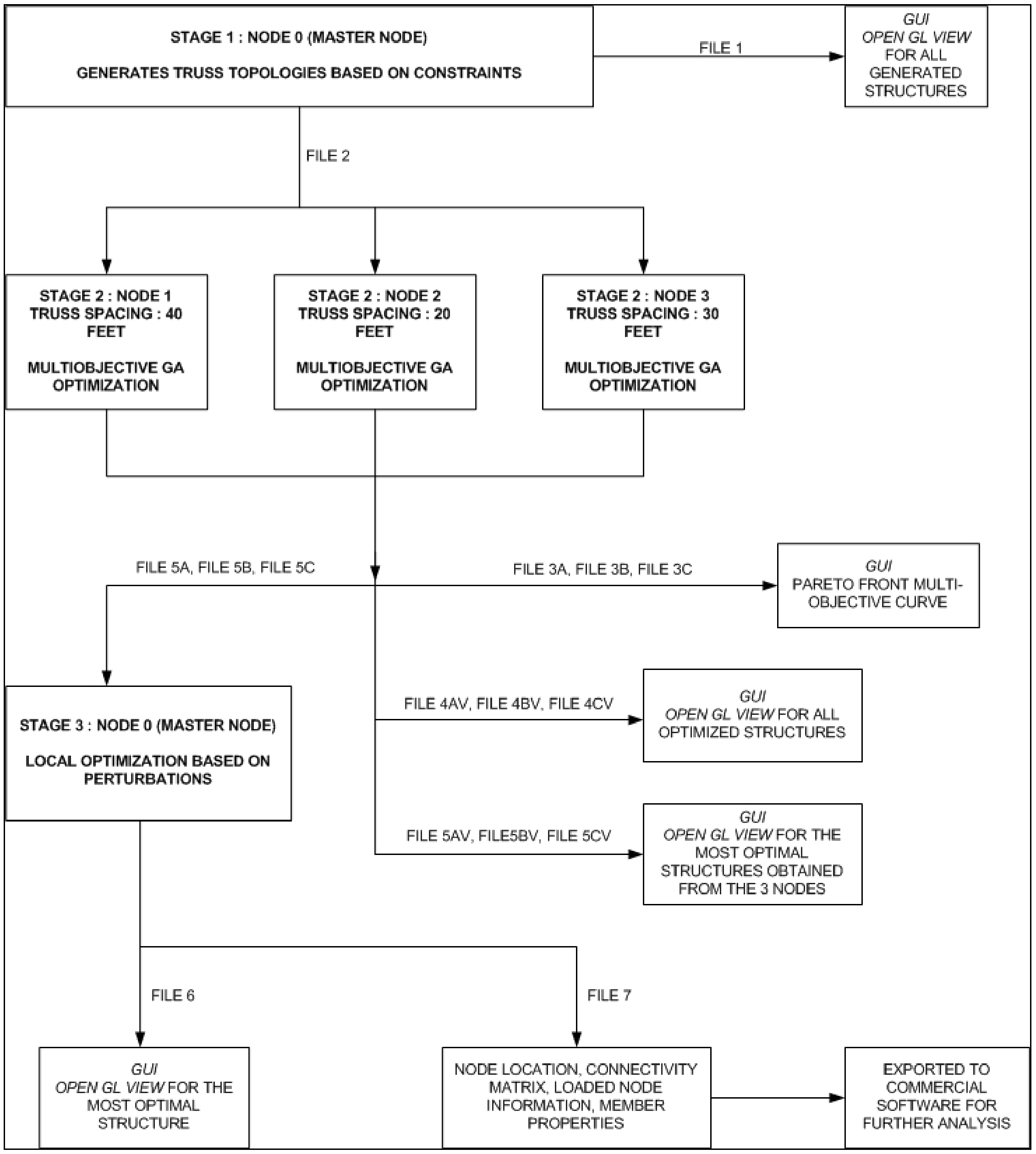
STAGE 1 : NODE 0 (MASTER NODE)
GENERATES TRUSS TOPOLOGIES BASED ON CONSTRAINTS
FILE 1
GUI
OPEN GL VIEW FOR ALL GENERATED STRUCTURES
FILE 2
STAGE 2 : NODE 1 TRUSS SPACING : 40 FEET
MULTIOBJECTIVE GA OPTIMIZATION
STAGE 2 : NODE 2 TRUSS SPACING : 20 FEET
MULTIOBJECTIVE GA OPTIMIZATION
STAGE 2 : NODE 3 TRUSS SPACING : 30 FEET
MULTIOBJECTIVE GA OPTIMIZATION
FILE 5A, FILE 5B, FILE 5C
FILE 3A, FILE 3B, FILE 3C
GUI
PARETO FRONT MULTI-OBJECTIVE CURVE
STAGE 3 : NODE 0 (MASTER NODE)
LOCAL OPTIMIZATION BASED ON PERTURBATIONS
FILE 4AV, FILE 4BV, FILE 4CV
GUI
OPEN GL VIEW FOR ALL OPTIMIZED STRUCTURES
FILE 5AV, FILE5BV, FILE 5CV
GUI
OPEN GL VIEW FOR THE MOST OPTIMAL STRUCTURES OBTAINED FROM THE 3 NODES
FILE 6
FILE 7
GUI
OPEN GL VIEW FOR THE MOST OPTIMAL STRUCTURE
NODE LOCATION, CONNECTIVITY MATRIX, LOADED NODE INFORMATION, MEMBER PROPERTIES
EXPORTED TO COMMERCIAL SOFTWARE FOR FURTHER ANALYSIS

**Fig. 2. Schematic representation of the proposed design optimization method**

# CHAPTER V

# GENERATION OF TOPOLOGY

Both ‘unstructured problem domain’ and ‘conceptual design’ refer to developing structures independent of established conventions in a user-defined design domain. For a two dimensional truss problem, this design domain refers to the extent of the maximum span of the truss and the maximum height possible for its crown. In this research, a third dimension refers to the direction in which the same truss design is duplicated at user-specified distances. This allows the real-time simulation of roof or bridge systems.

Triangles were selected as the building units for the trusses. The advantage of using triangles over other shapes is that their use leads to stable trusses. The selection of nodal points is completely random, however, as opposed to any fixed grids or ground structure used in previous research studies. This allows a comprehensive search of the entire design domain and the potential to suggest new and innovative designs.

Each truss structure is produced symmetrically from the defined supports. The production of the first triangle affects the shape of the entire design. Therefore, a provision is made for the user to influence the shape of this crucial triangle by providing the location of the first node next to the support nodes. The truss generation process continues by superimposing triangles on the sides of already existent triangles until the entire truss structure converges at the mid-point of the design domain. The sides of the triangle on which new triangles are built are chosen to facilitate the closure of the entire structure. The angles of each of the triangles are selected randomly from a set of ten pre-defined angles that range from 15 to 150 degrees.

The entire process of truss topology generation requires numerous constraints in order to produce aesthetically pleasing stable trusses. For example, an area constraint on newly generated triangles is imposed that prohibits them from varying too greatly from the general size of other triangles. A boundary constraint prevents the newly generated triangles from being placed outside the domain specified by the user. Another constraint is used to close the entire truss at the mid point when the triangles converge within the central ten percent of the design space.

A bound on the maximum number of members is provided by the user, which influences the complexity of the truss designs generated. If not specified, the trusses produced may have numerous small triangles and might not meet the users' design requirements.

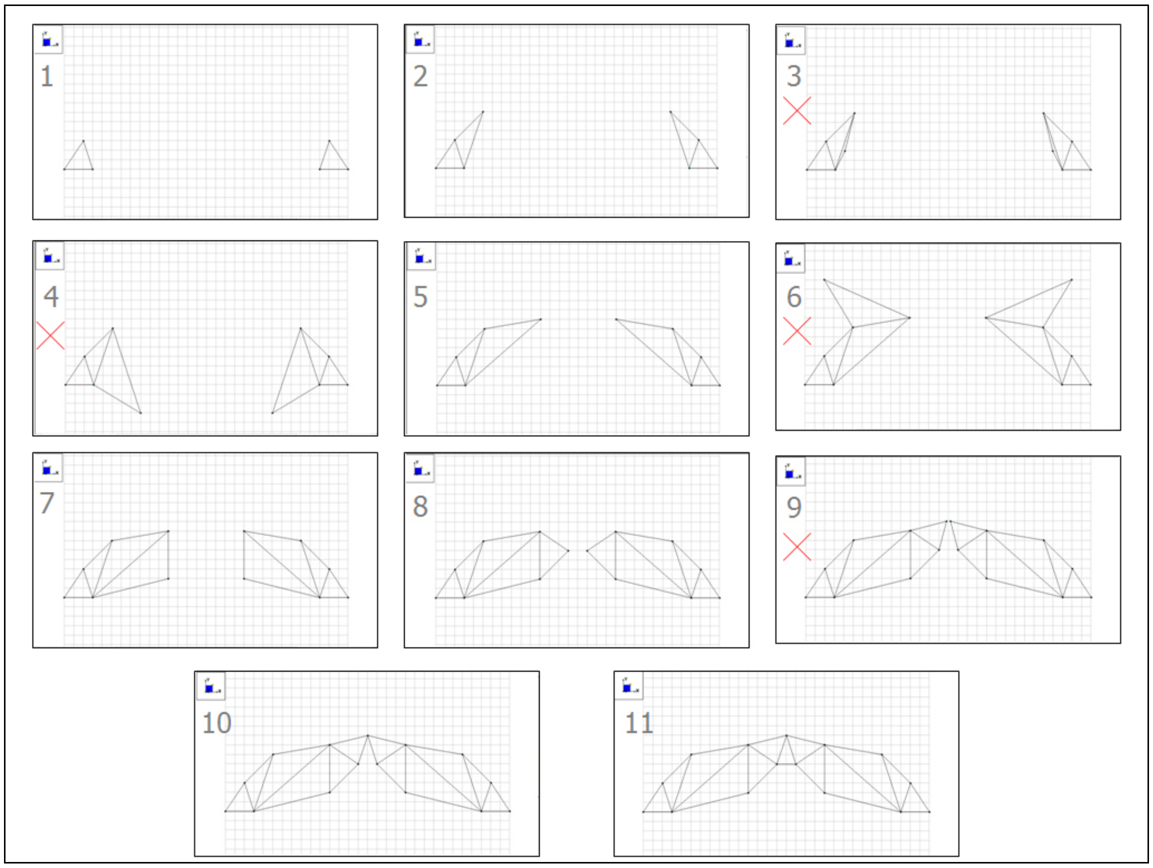

**Fig. 3. Generation of a truss topology shown in viewports 1-11**

The following is a detailed description of the process of evolution of a single truss. Figure 3 presents an example of how a single truss may be generated in this research.

i. Viewport 1: The support locations are specified at the extreme edge of the design domain. The co-ordinates of the first nodes next to each support nodes are specified by the user. Base angles are chosen at random from the set of pre-defined angles, which result in a randomized location of the apex node. It is important to note that in the above figure, the first nodes next to the support nodes have the same co-ordinates in the y-direction as the support nodes themselves. The user is not limited to this restriction. These nodes can be located anywhere in the design domain. The location of the nodes will affect the shape of the trusses generated. For instance, if the first node is specified at (10,10) for a design domain of (60,30), then the trusses produced are more likely to have a parabolic geometry.

ii. Viewport 2: The newly generated triangles are super-imposed on the already existent triangles. The side on which this super-imposition takes place is chosen to provide the final convergence of the entire design. The angles of these triangles are again chosen randomly from the set of pre-defined angles.

iii. Viewport 3: This view gives an example of applying the area constraint used for generating triangles. The area of each triangle super-imposed on the existent triangles must be within fifty to two hundred percent of the area of the existent triangle. Along with this, the new triangle must also satisfy other area ratios in relation to other triangles in the structure. The area constraint helps in maintaining uniformity along the entire truss structure. As shown in this view-port, the third triangle generated at random from each support does not satisfy the area constraints. Hence, this triangle is removed and a new triangle is generated.

iv. Viewport 4: The boundary constraint is imposed to prevent the truss from moving out of the user-specific design domain. In this view-port, the apex of the newly generated triangle is out of the design domain; hence it is automatically removed.

v. Viewport 5: The new triangle generated shown here satisfies all constraints acting on the truss. Therefore, the triangle generated in this step is accepted and the process continues.

vi. Viewport 6: The role of the divergence constraint is evident in this viewport. The sides of the existent triangles on which new triangles are super-imposed are chosen such that the structure converges to the centre of the entire domain. In this figure, the truss diverges instead on converging. This side of the existent triangle is removed from consideration and consequently the generated triangle is also removed.

vii. Viewport 7-8: After the correct side is chosen for advancement of the design, two new triangles are generated from each side that satisfy the specified constraints. The truss eventually converges within the central ten percent of the design space.

viii. Viewport 9-11: A special mechanism is invoked for the closure of the truss when the converging triangles lie within the central ten percent of the design space. As is shown in viewport 10, the generated triangles activate the closure mechanism, in which the triangles automatically converge at the central point (viewport 10). To provide stability, all unconnected members of the truss are joined together to form complete triangles (viewport 11). This may require the addition of one or two more members at this stage.

The entire process of generation of a single truss takes a few milliseconds on a two gigahertz Pentium machine. Overall, the generation of fifty trusses takes around four seconds. The OpenGL viewer can be used to display all the truss topologies and geometries generated in Stage 1.

Figure 4 shows fifty trusses generated for a problem domain with a span of sixty feet and a maximum height of thirty feet. The co-ordinates of the first node next to the support node was specified as (12 ft,0 ft). The maximum number of members, which influences the complexity of the overall truss generated, was specified as thirty.

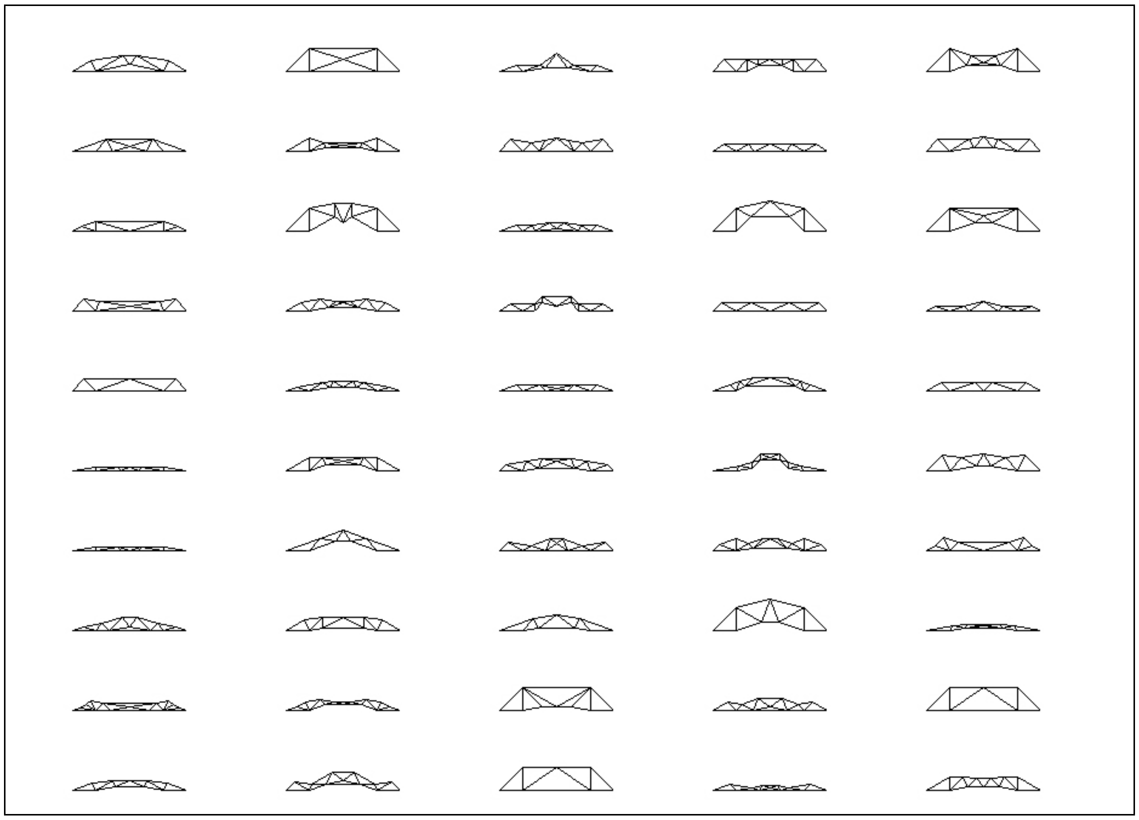

**Fig. 4. Truss topologies generated using Stage I**

By changing the maximum number of members allowed per truss and the co-ordinates of the first node, the user is able to influence the complexity of the entire design. The trusses shown in Figure 4 were developed with maximum allowable number of members as thirty. Figure 5 shows the trusses generated by Stage I with the maximum allowable members as hundred. In addition, the first node next to the supports was changed from (12 ft, 0 ft) to (5 ft, 0 ft). Overall, the trusses generated in the latter case were more complex and contained more members of shorter lengths.

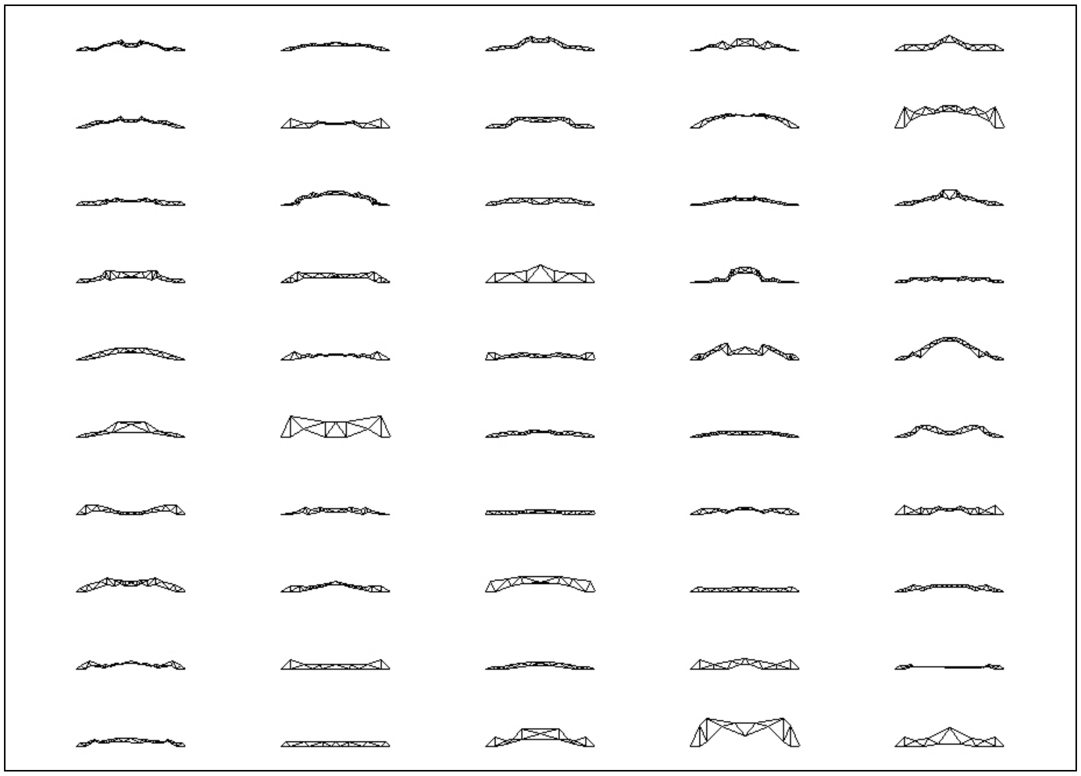

**Fig. 5. Truss topologies generated with a hundred maximum allowable members**

The specification of the first node next to the support node is also crucial in determining the shape of the overall design. The trusses shown in Figure 4 and 5 were developed with a starting point in-line with the support nodes, which led to more flatter geometries. Figure 6 shows an example of trusses generated when the co-ordinates of this node were not taken in-line with the support nodes. In this case, the design domain was considered the same as sixty feet by thirty feet, but the start point was considered to be (12 ft, 12 ft). As is evident from the figure, a specification of such a point led to more parabolic structures.

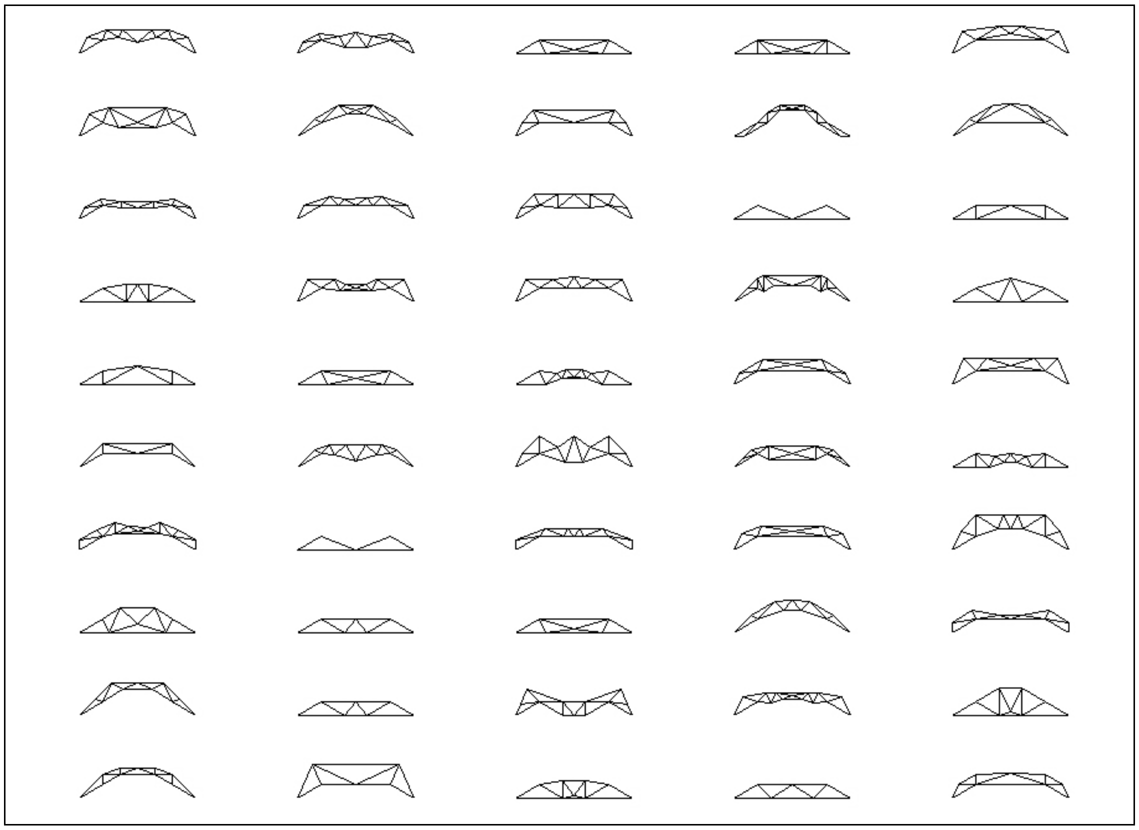
**Fig. 6. Truss topologies generated with a start point of (12 ft, 12ft)**

The trusses generated in Stage I can be sorted in the ascending order of the number of members. There is also a provision of sorting out these members in descending order of the maximum height. A separate module has been developed to cater to sorting. The user decides whether to invoke this module in between Stage I and II. Figure 7 and Figure 8 present the same trusses shown in Figure 4 ordered in terms of the maximum number of members and maximum height.

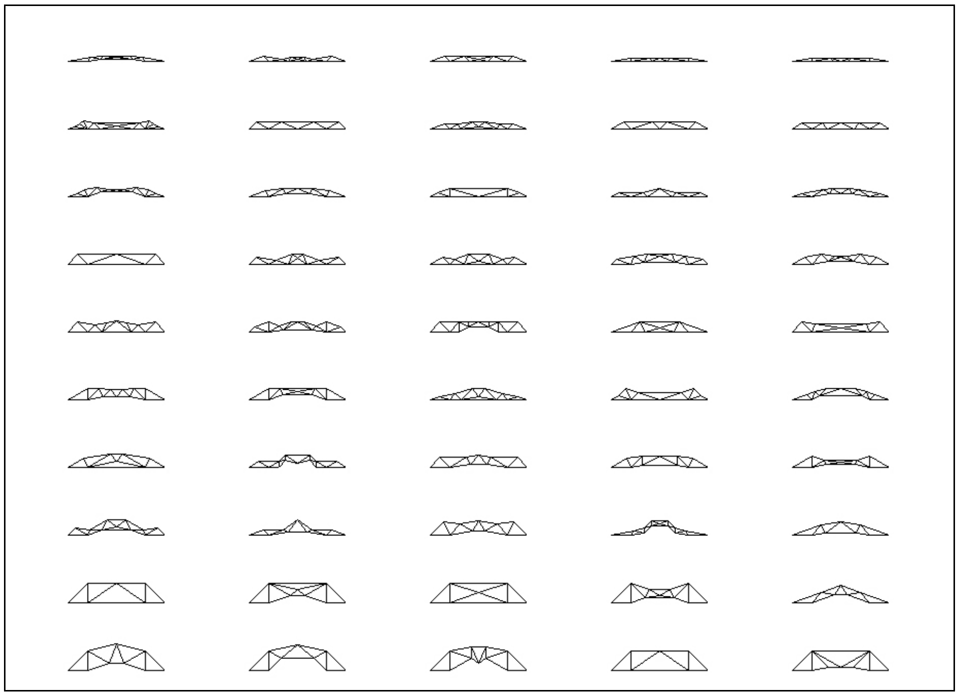

**Fig. 7. Trusses sorted in descending order of their maximum heights**

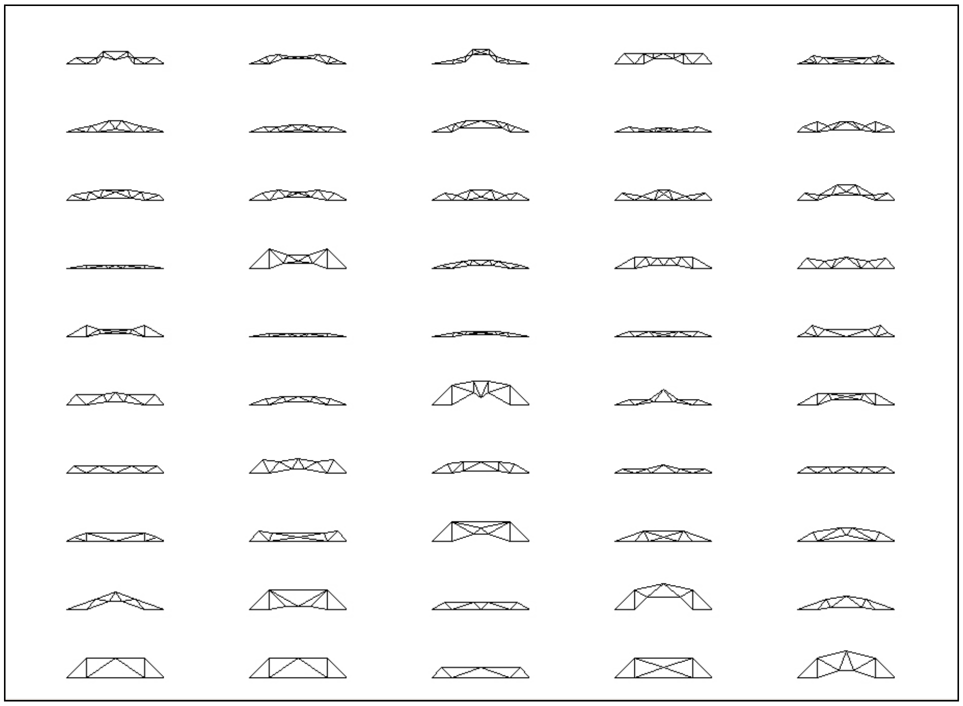

**Fig. 8. Trusses sorted in ascending order of the number of members**

Currently, the program supports the development of a maximum of fifty trusses for the stand-alone utility. A limited set of twelve trusses are generated for trials presented using the parallel application.

# CHAPTER VI

# SIZING OPTIMIZATION USING HEURISTIC TECHNIQUES

After generating a diverse set of stable truss topologies in Stage I, sizing optimization of members is performed in Stage II. The motivation in using heuristic techniques for sizing optimization is to obtain optimal truss designs while reducing the total computational time required. As discussed earlier, a heuristic is defined as a problem solving technique in which the most appropriate solution of several found by alternative methods is selected at successive stages of a program for further refinement based on problem dependent or heuristic knowledge. Although this stage in the research is focused primarily on the application of multi-objective genetic algorithms, it also explores the application of simulated annealing and reactive taboo search for truss sizing optimization problem.

The goal of all three heuristic methods investigated is to escape being trapped in a local optimum in order to have a better chance of obtaining the global optimum. Both simulated annealing and the reactive taboo search methods work with a single solution, which in this research is comprised of the current member sizes for the specified truss layout being optimized. All three methods also depend on probabilistic search operators, although to different extents. In simulated annealing, the specified temperature is reduced at a constant rate over the search process. At each temperature, solutions in the neighborhood of the current solution are investigated in an attempt to find a more optimal solution. If a better solution is found, it is always selected as the next solution. In some cases, a worse solution may be selected depending on the Metropolis criteria defined. The neighborhood of the current solution can be controlled through a step size, which is adapted in this research as the search proceeds depending on the number of acceptances of neighborhood solutions. The reactive taboo search method provides an adaptive feature to taboo search that assists in more efficiently reaching the optimal solutions. The parameters used in reactive taboo search, therefore, are reactive and depend on the current search process. If the solution gets stuck in a local optimum, an escape mechanism is invoked that allows a random point away from the local optimum to be selected and the search continues from that point.

Genetic algorithms, in comparison, work with a population of current solutions. In this research, a multi-objective genetic algorithm (MOGA) formulation is used to convert the stress and deflection constraints into additional objectives. The three objectives are then simultaneously minimized. Selection is performed using each individual's rank in the current population, which is determined by the trade-off surface defined among the three objectives. The three-dimensional surface is defined by the set of Pareto-optimal solutions optimized. Therefore, multiple solutions can exist that are equally optimal. Innovative probabilistic selection and crossover operators are also used to provide a more efficient MOGA search method.

The user is given an option to choose one of five sets of member section sizes to be used during optimization. If a particular set leads to unsatisfactory results, the entire stage can be re-run with a different property set. This provides flexibility in terms of steel section availability. The five sets being used in this research contain sixteen sections each, selected from the AISC LRFD (2001) manual between the sections mentioned in Table 1. For comparing the results for the bridge benchmark problem, a set of thirty-two sections between W14x22 to W14x426 was used.

**Table 1. Section sets available for sizing optimization**

| Set Number | Start Section | End Section |
|---|---|---|
| 1 | W4x13 | W8x24 |
| 2 | W6x15 | W8x27 |
| 3 | W10x17 | W10x112 |
| 4 | W12x14 | W12x190 |
| 5 | W14x22 | W14x211 |

This program currently considers only one LRFD load combination: 1.2 times the dead load combined with 1.6 times the live load. The values for dead and live loads used in the trials presented in this section were specified as follows:

DL: Dead + Mechanical Loads: 30 psf

LL: Live Loads: 40 psf

Total Load: 1.2 DL + 1.6 LL = (1.2 * 30) + (1.6 * 40) = 100 psf (1)

The loads used for comparing the results obtained for the roof and bridge truss benchmark problems investigated in this research may differ according to the problem definition.

The spacing between the individual trusses can be specified by the user. In the parallel model, the optimization for three different spacings is carried out simultaneously. This spacing in the third dimension is multiplied by the area load to obtain the linearly distributed loading carried by each truss. Furthermore, the linearly distributed load is multiplied by the tributary length of members associated with each node to obtain the value of the concentrated nodal reaction at node on the top for roof truss problems. For the bridge trusses, the concentrated loads are applied at the nodes of the bottom chord in accordance with the tributary lengths of the members associated with those nodes.

Calculation of stresses in members and deflections of nodes is performed using a finite element solver, which has been embedded in the program. The input to the FE solver provided consists of the support and nodal locations, connectivity matrix, and the loading information for each truss evaluated using MOGA. The output provided is the nodal displacements and member stresses. These values are used in accordance to AISC LRFD (2001) criteria to determine the optimality of the member for the roof truss problems. These values are checked against the AISC ASD (1992) criteria for the bridge truss benchmark problem. A separate check is done for allowable tensile and compressive stresses, slenderness criteria, and allowable displacement at each node. A summary of these design criteria is given in chapter XI.

The member sizing optimization results obtained from Stage II can directly be exported to commercial structural analysis software packages. Stage II also compiles the design results in the form of a data file that is used in Stage III to perform local geometry optimization.

## MULTI-OBJECTIVE GENETIC ALGORITHMS

Genetic algorithms “combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search” (Goldberg 1989). In essence, it is a computational technique that gains its probabilistic mechanisms from natural evolution and Darwin’s “Theory of the Survival of the Fittest”. GAs provide a balance between efficiency and efficacy in any search domain. Traditional methods of optimization like calculus-based, enumerative or hill-climbing algorithms work best in finding a single optimal solution, and are therefore highly ineffective in discontinuous and multi-modal search domains. GAs, in comparison, rely on a probabilistic directed search of the entire domain, and hence are able to reach global optimum in multi-modal environments. Figure 9 shows the standard fitness landscape for the ‘sum of squares’ function, $\Sigma x_i^2$ which has a single optimal solution. Figure 10 shows the fitness landscape of Schaffer’s F6 function, $0.5 + (\sin^2(\sqrt{(x^2 + y^2)}) – 0.5)/((1 + 0.001(x^2 + y^2))^2)$ that has a large number of local optimal solutions and just one global optimum. The global optimal solution in the case of Schaffer’s F6 function can be more effectively searched using genetic algorithms, in comparison with other traditional mathematical techniques.

After the evolution of genetic algorithms in the 1970’s, a large number of hybrid and advanced GA methods have been proposed by researchers and implemented to solve optimization problems. State-of-art techniques for multi-objective GAs are used in this research. Multi-objective genetic algorithms (MOGA) convert stated constraints into additional objectives, which helps in identifying the trade-offs that exist between conflicting objectives like minimizing weight and deflections.

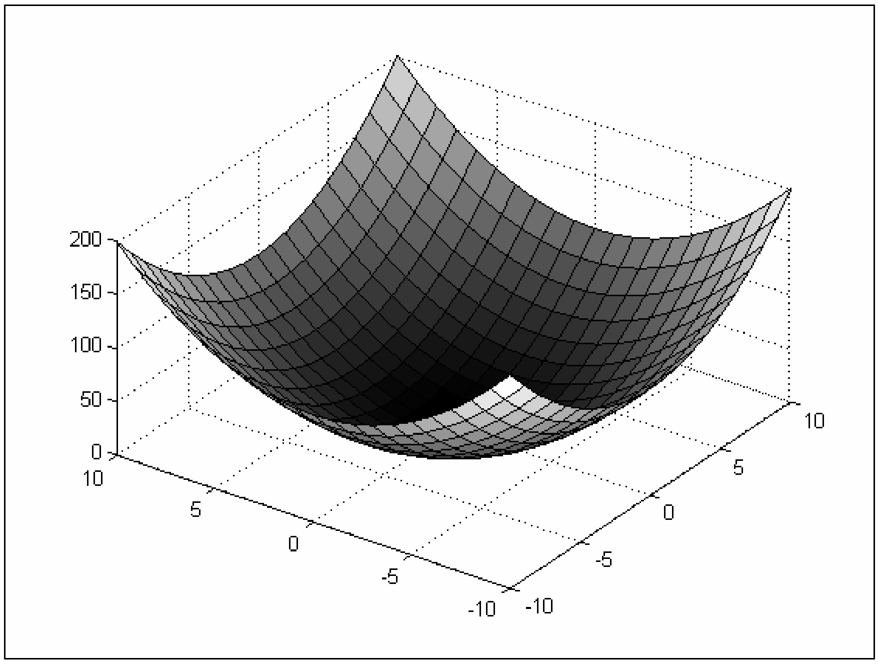

**Fig. 9. ‘Sum of squares’ function with a single global optimum**

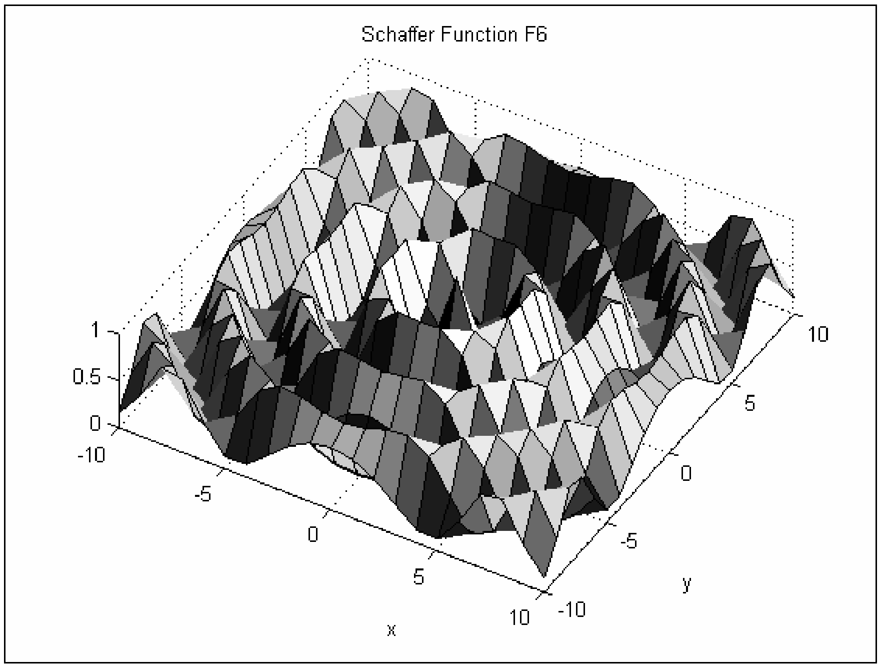

**Fig. 10. Schaffer’s F6 function with multiple local optimum and one global optimum**

Figure 11 presents a schematic representation of the generalized process flow for a simple genetic algorithms.

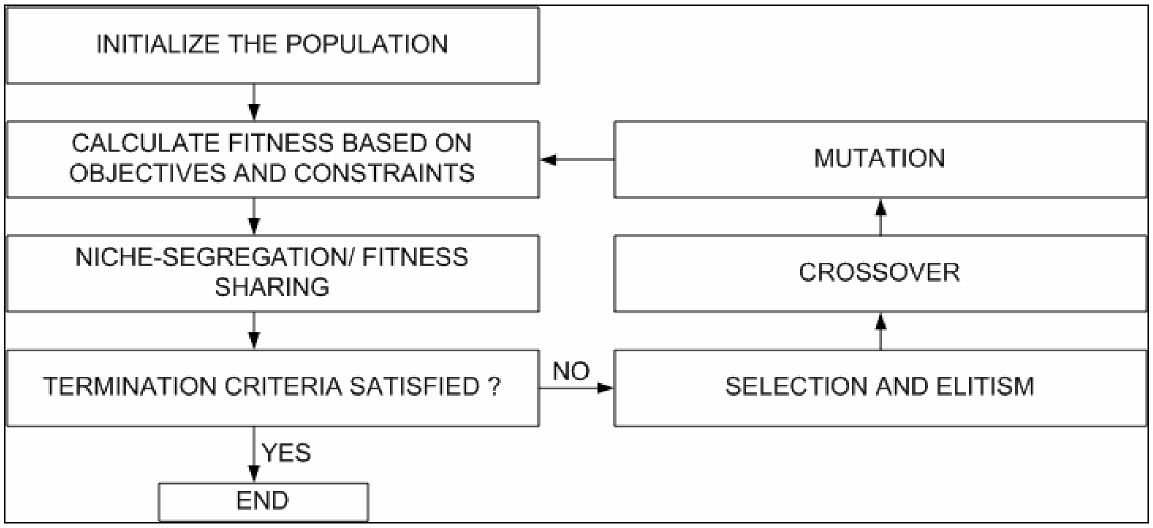
INITIALIZE THE POPULATION
CALCULATE FITNESS BASED ON OBJECTIVES AND CONSTRAINTS
NICHE-SEGREGATION/ FITNESS SHARING
TERMINATION CRITERIA SATISFIED ?
YES
END
NO
SELECTION AND ELITISM
CROSSOVER
MUTATION

**Fig. 11. Schematic representation of a simple genetic algorithm**

Genetic algorithms work with a population of individuals. For the sizing optimization problem addressed in Stage II of this research, each individual encodes a set of member sections associated with the members in the truss being optimized. These member sections are encoded as binary variables in order to improve the search efficiency of the genetic algorithm. The initial population is generated by randomly assigning 0/1 values to the binary bits of each individual. Table 2 shows an example of the binary representation mapping of one of the section sets used in this research.

**Table 2. Binary and decimal representation of member sections in the third set**

| Member Section | Notation | |
|---|---|---|
| | Decimal | Binary |
| W10x17 | 0 | 0 0 0 0 |
| W10x19 | 1 | 0 0 0 1 |
| W10x22 | 2 | 0 0 1 0 |
| W10x26 | 3 | 0 0 1 1 |
| W10x30 | 4 | 0 1 0 0 |
| W10x33 | 5 | 0 1 0 1 |
| W10x39 | 6 | 0 1 1 0 |
| W10x45 | 7 | 0 1 1 1 |
| W10x49 | 8 | 1 0 0 0 |
| W10x54 | 9 | 1 0 0 1 |
| W10x60 | 10 | 1 0 1 0 |
| W10x68 | 11 | 1 0 1 1 |
| W10x77 | 12 | 1 1 0 0 |
| W10x88 | 13 | 1 1 0 1 |
| W10x100 | 14 | 1 1 1 0 |
| W10x112 | 15 | 1 1 1 1 |

A fitness value can be assigned to each individual in the population. An individual's fitness plays a role in determining whether an individual is selected for reproduction to produce the individuals in the next generation. For simple genetic algorithms (SGA), the fitness is calculated using a composite function of the objective function and any violated constraints, which are imposed as penalties to the fitness value. In structural optimization, often minimizing the total weight is considered as the objective function and excessive displacements and stresses are applied as penalties for the sizing optimization problem in trusses.

Fitness can be formulated as a composite function of the objective function f(x) and the penalty constraints P( ). A few examples of these formulations are shown in Table 3.

**Table 3. Composite fitness functions used in previous research efforts**

| Reference | Fitness | Comments |
|---|---|---|
| Rajan (1995) | f(x) + P(constraints) | |
| Yeh (1999) | f(x)( 1 + K*P(constraints) ) | K: Penalty factor |
| Shrestha and Ghaboussi (1998) | $f(x)\prod(P(constraints))^{\alpha(constraints)}$ | α: Exponential penalty weight factor |
| Raich and Ghaboussi (2000) | f(x) x P(constraints) | |
| Jenkins (2002) | $C_{max}$–Mass–P(constraints) | Cmax: Constant sized to prevent -ve fitness |

In a simple genetic algorithm, often the fitness function becomes highly sensitive to its formulation and has to be optimized in order to produce good results.

One of the major advantages of using MOGA over SGA is its independence to the fitness function on the priority factors used to penalize for constraint violations. In MOGA, the selection of parents undergoing reproduction is done on the basis of their individual rank, which in turn is determined from the trade-off surface obtained for the stated objectives. This surface, commonly known as the Pareto surface, is a three-dimensional curve in this research since there are three objectives stated that related to weight, stress and deflection. Rank 1 is assigned to non- dominated individuals in the population. Consequently, these individuals are eliminated, and the next set of non-dominated individuals is assigned a Rank of 2 and so on (Srinivas and Deb 1994). The Eschenauer criteria (1990) is used to evaluate the non – dominated points. It is stated as follows for the minimization problem as:

A feasible solution x* is a non-dominated (Pareto) optimum if and only if there exists no feasible vector x such that:

$$f_i(x) <= f_i(x^*) \quad \text{for all } i \in \{1, 2, \ldots. N\} \tag{2}$$

$$f_i(x) < f_i(x^*) \quad \text{for at least one } i \in \{1, 2, \ldots. N\} \tag{3}$$

Clustering might take place while performing the MOGA search process over a number of generations. Clustering is defined as the coupling together of a number of solutions in one region of the Pareto-surface, instead of spreading out evenly over the Pareto surface, as is ideally desired. This might lead to early convergence of the population and hinder the trade-off capability of the MOGA. Niche-segregation or fitness sharing is implemented to remove this effect. This is carried out by degrading the fitness of each design in proportion to the number of designs located in its neighborhood (Hajela and Lee 1995). This form of clustering was not noticed in this research. Therefore, the clustering techniques were not implemented.

Tournament selection is a common technique for selecting parents in genetic algorithms. For the process of selection, a tournament of individuals is randomly picked up from the population, and the fittest individual in this tournament set is chosen as one parent. The other parent is chosen in a similar way from another tournament. In this research, instead of using the conventional method of tournament selection, a new criterion is used for selecting parents. One of the parents is randomly selected from the population, and the other one is selected with a rank lower than the first parent. This modified selection assists in moving the Pareto surface towards an improved solution, while also maintaining the diversity in the population. Another reason for not choosing standard tournament selection is to reduce the extent of sensitivity to the initially specified tournament size parameter. Figure 12 shows an example of the ranked individuals obtained using the above selection process at an intermediate stage in the research program. The individuals having a rank of 1 constitute the Pareto-optimal surface.

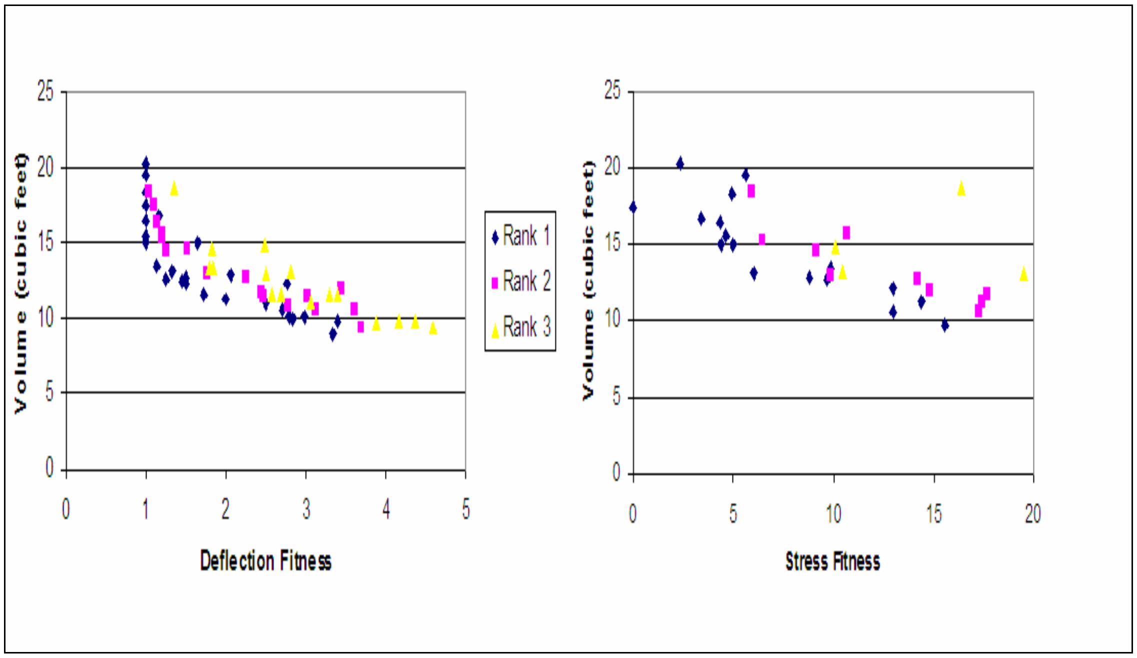

**Fig. 12. Ranking of individuals in MOGA**

Elitism strategy was adopted to help keep the fittest members of the population found to date in the search. A few members are chosen from the Pareto-optimal surface (having a rank 1) and passed onto the next generation without reproduction. This helps in keeping a portion of the best-fit members always in the population, which prevents the solution from diverging heavily from optimal results due to destroying the fittest individual during crossover and mutation or not selecting it into the next generation.

After selecting the parents undergoing reproduction and elitism, cross-over and mutation are carried out. Both of these operators are carried out with some preset probability of occurrence. The probabilities of occurrence of crossover and mutation used in this research are 0.8 and 0.01. Figure 13 shows how single-point crossover operation is performed. Two parents are selected from the current population and undergo crossover. The crossover point is selected at random for each set of parents. The children created replace their parents in the next population.

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parent 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| Parent 2 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Child 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Child 2 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |

**Fig. 13. Single-point crossover operation**

Multi-point cross-over is implemented in this research. A random point is selected for each encoded member section of the truss, and each of these member sections undergoes a change during each cross-over operation. In general, it was observed in this research that multi-point crossover provides better convergence to the global optimal solution. Figure 14 shows the implementation of multi-point crossover as is used in this research. Each member section is encoded by 4 bits similar to the binary mapping previously discussed.

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Parent 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| Parent 2 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Child 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Child 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |

**Fig. 14. Multi-point crossover operation**

Mutation is performed to help maintain diversity in the population. Mutation also helps in visiting unexplored regions of the design space. By randomly flipping the encoded bits, as shown in figure 15, mutation helps in producing solutions that are not yet a part of the population.

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Before Mutation | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| After Mutation | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |

**Fig. 15. Mutation operation**

The flowchart shown in figure 16 is the exact representation of the MOGA process as implemented in this research. If there is no improvement in the solution after a particular number of generations, then the process is automatically terminated for that particular truss geometry and topology, and same procedure is carried out for the next truss.

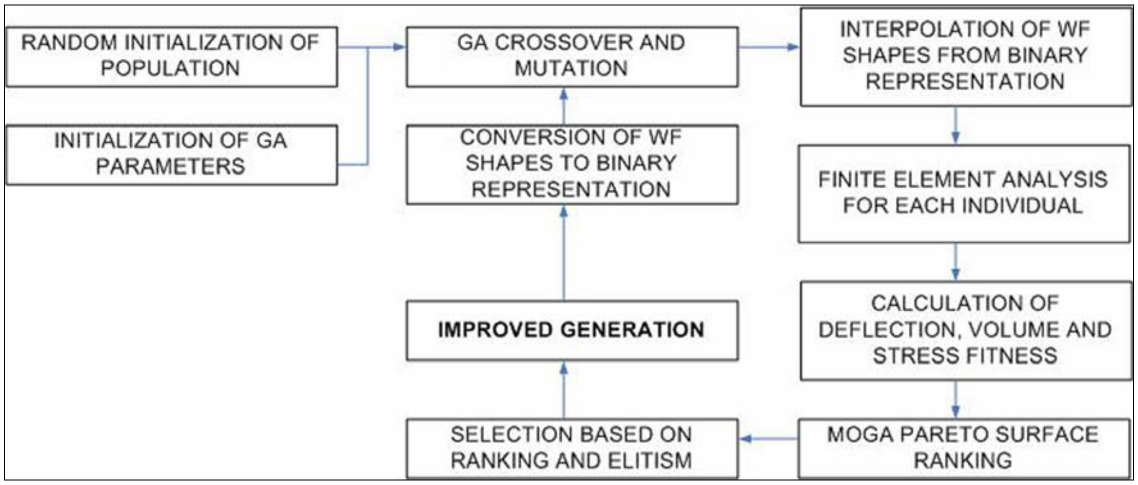
RANDOM INITIALIZATION OF POPULATION
INITIALIZATION OF GA PARAMETERS
GA CROSSOVER AND MUTATION
CONVERSION OF WF SHAPES TO BINARY REPRESENTATION
IMPROVED GENERATION
SELECTION BASED ON RANKING AND ELITISM
INTERPOLATION OF WF SHAPES FROM BINARY REPRESENTATION
FINITE ELEMENT ANALYSIS FOR EACH INDIVIDUAL
CALCULATION OF DEFLECTION, VOLUME AND STRESS FITNESS
MOGA PARETO SURFACE RANKING

**Fig. 16. Process flow for MOGA used in this research**

## SIMULATED ANNEALING

Simulated annealing, as the name suggests, is analogous to the annealing of metals that is performed to produce better materials. The concept of simulated annealing stems from the work carried out by Metropolis et al. (1953) for predicting the equilibrium state of atoms at a particular temperature state. A small displacement is provided to each atom and the corresponding energy $\Delta E$ is computed. If the change in energy is less than or equal to zero, then the displacement is accepted and the configuration with the displaced step is taken as the initial point for the next step. In case $\Delta E$ exceeds zero, then a probabilistic acceptance rate is calculated as $\exp(-\Delta E/K_bT)$. If the probability is greater than one, then the displacement is accepted. Otherwise the original state is used as the initial step. By repeating this process, Metropolis et al. were able to represent the thermal motion of the atoms in contact with a heat surface at temperature T.

The same concept was used by Kirkpatrick et al. (1983) in developing the heuristic method termed simulated annealing. They considered that “The simulated annealing process consists of first melting the system being optimized at a high effective temperature, then lowering the temperature by slow stages until the system freezes and no further changes occur.”

In this research, the simulated annealing (SA) process begins by randomly generating a single string of member sections assigned symmetrically to a given truss layout. A set of SA parameters including the initial temperature (T), rate of cooling (RT), and number of cycles to be performed at each temperature (NCYC) are initialized. Each value is picked randomly from a range of pre-defined values.

The step size, which determines the distance of a new solution from the previous solution, is initially set to one-sixth of the total design variable range of values used to define the search space. In this research, the design space consists of sixteen member sections ranging from W4x13 to W8x24. The fitness, f, of the current solution is obtained by multiplying the scaled stress and deflection penalties by the total volume of the truss.

Figure 17 shows the process flow for simulated annealing. The outer loop that imposes the temperature level runs until the initial temperature (T) becomes zero. Therefore it is dependent on the rate of cooling (RT) specified. In each cycle, the current temperature is reduced by the rate of cooling.

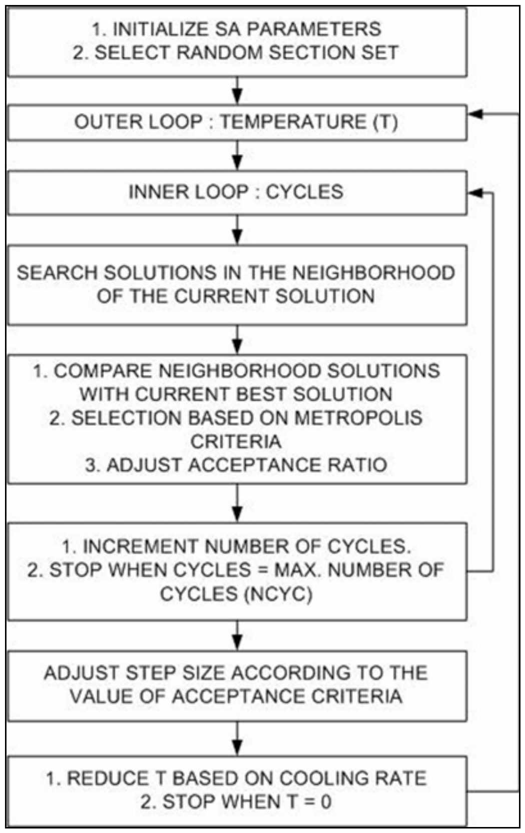
1. INITIALIZE SA PARAMETERS
2. SELECT RANDOM SECTION SET
OUTER LOOP : TEMPERATURE (T)
INNER LOOP : CYCLES
SEARCH SOLUTIONS IN THE NEIGHBORHOOD OF THE CURRENT SOLUTION
1. COMPARE NEIGHBORHOOD SOLUTIONS WITH CURRENT BEST SOLUTION
2. SELECTION BASED ON METROPOLIS CRITERIA
3. ADJUST ACCEPTANCE RATIO
1. INCREMENT NUMBER OF CYCLES.
2. STOP WHEN CYCLES = MAX. NUMBER OF CYCLES (NCYC)
ADJUST STEP SIZE ACCORDING TO THE VALUE OF ACCEPTANCE CRITERIA
1. REDUCE T BASED ON COOLING RATE
2. STOP WHEN T = 0

**Fig. 17. Schematic representation of simulated annealing**

The inner loop, which performs the local neighborhood search, runs for the number of cycles (NCYC) specified. During each cycle within the inner loop, a complete neighborhood evaluation is conducted. A better solution is always accepted as the most optimal solution. In some cases, a worse solution might also be selected depending on Metropolis's criteria. The change of energy term in the Metropolis criteria is replaced by the difference between the fitness of the fittest solution and fitness of the current solution in the current iteration. When T is high, the probability of acceptance is high; and when T reduces, the probability also reduces. Therefore, the probability of selecting a worse solution is high at the beginning of the run, and reduces as the optimal solution is approached. This solution could be a local optimum.

Probability of Selection = $\exp\{ (f - fs) / T^{-}_{i} \}$ (4)

where f : Fitness of the current most optimal solution

fs: Fitness of the current individual

$T_i$ : Current Temperature

If a solution is not accepted, then the acceptance ratio for that particular cycle is changed depending on the current number of cycles.

$(AR)_i = (AR)_{i-1} - 1/(NCYC)$ (5)

| | | |
|---|---|---|
| where | (AR) | : Acceptance Ratio |
| | i | : Current Cycle Number |
| | NCYC | : Total number of cycles to be performed at each temperature. |

After each cycle of examining neighborhood solutions, the step size is adjusted in accordance with the final value of the acceptance ratio. This is done in accordance with the criteria used by Corana et al.(1987) which is given as follows:

If (AR) > 6.0 then $(ST)_n = (ST)_{n-1} * (1.0 + SF*((AR) - 0.6)/0.4)$ (6)

If (AR) < 0.4 then $(ST)_n = (ST)_{n-1} / (1.0 + SF*(0.4 - (AR))/0.4)$ (7)

| | | |
|---|---|---|
| where | (AR) | : Acceptance Ratio |
| | (ST) | : Step Size |
| | n | : Current cycle number |

If the step size exceeds the upper range limit of the design space, then it is reset to the upper limit value. The acceptance ratio is reset to one before proceeding with the new temperature. The new temperature is obtained by multiplying the current temperature by the cooling rate. The entire SA process is continued until the final temperature reaches zero. The most optimal solution obtained over all the cycles at all temperatures is as the final solution obtained using the SA.

## REACTIVE TABOO SEARCH

Tabu search was first suggested by Glover (1977) as an optimization tool for non-linear problems. According to Bland and Dawson (1991), Tabu search starts at some initially feasible point and uses a local hill-climber technique to search for optimal solutions. A neighborhood of the current solution is defined and the most optimal point within this neighborhood is chosen as the next feasible solution. The advantage of using Tabu search over a local hill-climber is its special feature of allowability. Solutions accepted for a particular set of previous moves are prohibited from re-visits. This prevents the cycling of solution around the same point. Although, simple Tabu search might provide an effective mechanism for solving some 'noisy' functions, it suffers from major drawbacks. Its main demerit is that each point in the solution space must be accessible from every other point within a finite number of moves. As this is not possible for many design problems, the process may never reach the global optimal solution.

Reactive taboo search (RTS) is an advanced implementation of Tabu search, and in general, is more efficient in finding a global optimum. It provides an effective mechanism to escape the local optimum. By using a random mutation strategy, it more efficiently searches the design domain. Therefore it not required for each point to be accessible from every other point in the design space. The parameters used in RTS are reactive, or dependent, on the process of optimization. This reduces the sensitivity of the method to the initial parameter settings.

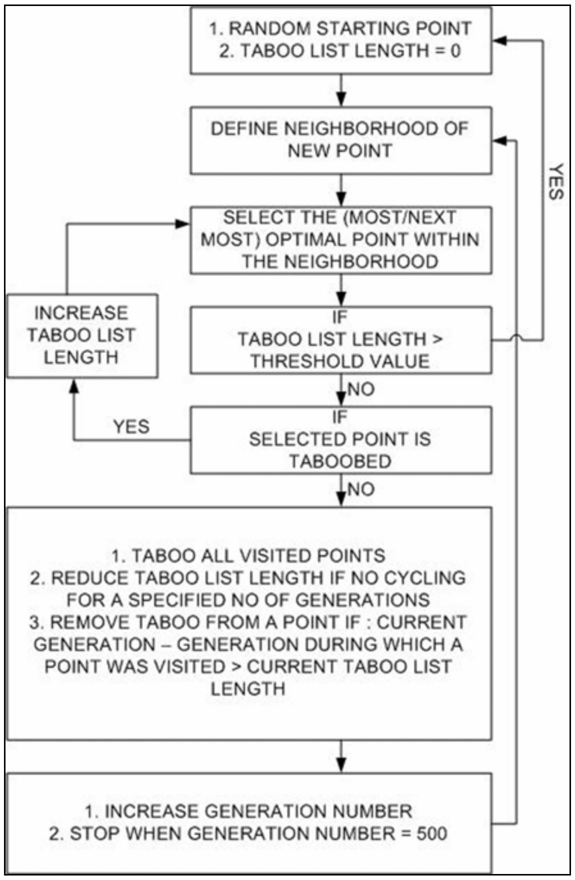
1. RANDOM STARTING POINT
2. TABOO LIST LENGTH = 0
DEFINE NEIGHBORHOOD OF NEW POINT
SELECT THE (MOST/NEXT MOST) OPTIMAL POINT WITHIN THE NEIGHBORHOOD
IF TABOO LIST LENGTH > THRESHOLD VALUE
YES
NO
INCREASE TABOO LIST LENGTH
YES
IF SELECTED POINT IS TABOOBED
NO
1. TABOO ALL VISITED POINTS
2. REDUCE TABOO LIST LENGTH IF NO CYCLING FOR A SPECIFIED NO OF GENERATIONS
3. REMOVE TABOO FROM A POINT IF : CURRENT GENERATION – GENERATION DURING WHICH A POINT WAS VISITED > CURRENT TABOO LIST LENGTH
1. INCREASE GENERATION NUMBER
2. STOP WHEN GENERATION NUMBER = 500

**Fig. 18. Schematic representation of reactive taboo search**

Figure 18 shows the process flow for RTS. The process begins by evaluating a random solution point and its neighborhood. The number of solutions in the neighborhood is calculated as follows for symmetric trusses

Odd number of members : $2 * (N/2 + 1)$ (8)

Even number of members : $2 * (N/2)$ (9)

Where N is the total number of members in a truss.

The factor of two is introduced due to the fact that the neighborhood consists of solutions having member sections both above and below each current section.

Finite element analysis is followed by fitness evaluation for each solution in the neighborhood. The fitness of each individual is obtained by multiplying the scaled stress and deflection penalties by the total volume of the truss. A local hill climber technique is implemented until a local optimal is found. Simultaneously, all visited points are tabooed or prohibited from revisits. The generation during which a solution is visited and its objective function evaluation is stored in the memory. This helps in preventing re-evaluation of the same set of member sections, and in turn helps in reducing the total computational expense.

If a solution is re-visited, the taboo list length is increased. When the local hill climber reaches a local minimum, it will try to re-visit tabooed solutions. This increases the taboo list length. At some point, the taboo list length reaches the specified threshold value. The maximum taboo list length is considered to be less than the size of the neighborhood to prevent the entire neighborhood from being tabooed. The threshold value in this research is considered in accordance with Hamza et al. (2003) to be 50 % of the maximum taboo list length. A quick escape mechanism is triggered when the current taboo list length exceeds the threshold value. It is achieved by randomly mutating the member section set of the current solution. The intent is to escape the local pool and to explore other parts of the search space. The list length is decreased if there is no cycling over a specified number of iterations. The taboo is removed dynamically from previously visited solutions according to the following criteria

Remove Taboo if:
current generation – generation during which the point was visited > current list length

The entire procedure is run for 500 generations. The maximum number of iterations for a structure with a neighborhood size of A is given by

Max number of iterations = (500 * A) – (revisited solutions) (10)

As is evident from the explanation of this methodology, all parameters are dependent on the search process. The taboo list length is dynamically changed as the process continues, and is responsible for triggering the quick escape mechanism, if necessary. Therefore, the main advantage of this method is that its self propagating and independent of any influences of initial parameter settings. The reactive or adaptive nature of this search makes this algorithm more efficient than the simple tabu search.

# CHAPTER VII

# COMPARISON STUDY BETWEEN THE THREE HEURISTIC METHODS

A two-dimensional roof truss is used as a benchmark problem in order to compare the three heuristic methods previously discussed: MOGA, SA and RTS. The problem domain defines a span of 15.24 m (50 ft) and the crown height as 3.34 m (10.95 ft). The roof load imposed on the truss is 58,375.36 N/m (4000 plf). The concentrated nodal loads are calculated by multiplying the distributed load by the tributary length associated with each member framing into a node. The material properties are those of steel (E = 201 GPa, $f_y$ = 344.736 MPa, and $\rho$ = 7851.03 Kg/m$^3$). The AISC LRFD design specifications (2001) are followed: the allowable tension force is $0.9f_yA_g$; the allowable slenderness ratio is 300 for tension members and 200 for compression members; the allowable joint displacement is limited to 25.4 mm (1 inch); and the allowable compression force for each member is determined from buckling considerations, using the relevant parts of the AISC code. The members sections are selected from a set of 16 standard steel sections ranging between a W4x13 to a W8x24 from the AISC LRFD (2001) Manual. Figure 19 shows the near-optimal truss topology considered, along with the calculated nodal loads carried by the truss.

The performance of the MOGA, SA and RTS are compared on the basis of their overall ability to reach optimal solutions and their average computational expense. A study is also conducted to evaluate the repeatability of near-optimal solutions and the sensitivity of each method to the initial parameter settings.

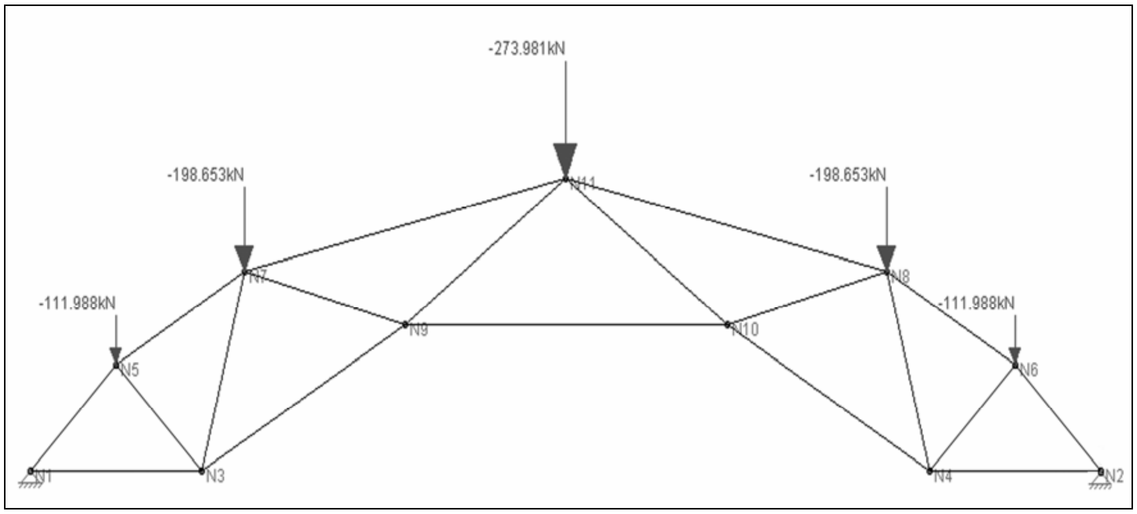

**Fig. 19. Truss topology and nodal forces considered for comparison of results**

## ABILITY TO OBTAIN NEAR-OPTIMAL TRUSS DESIGNS

After running several trials in order to optimize the initial method parameters, thirty trial runs were conducted using each heuristic method to determine their effectiveness in reaching near-optimal solutions. The most optimal solutions (i.e., the solutions with the least volume that also simultaneously satisfy the stress and deflection constraints) obtained in each trial for the roof truss benchmark problem are plotted in Figure 20.

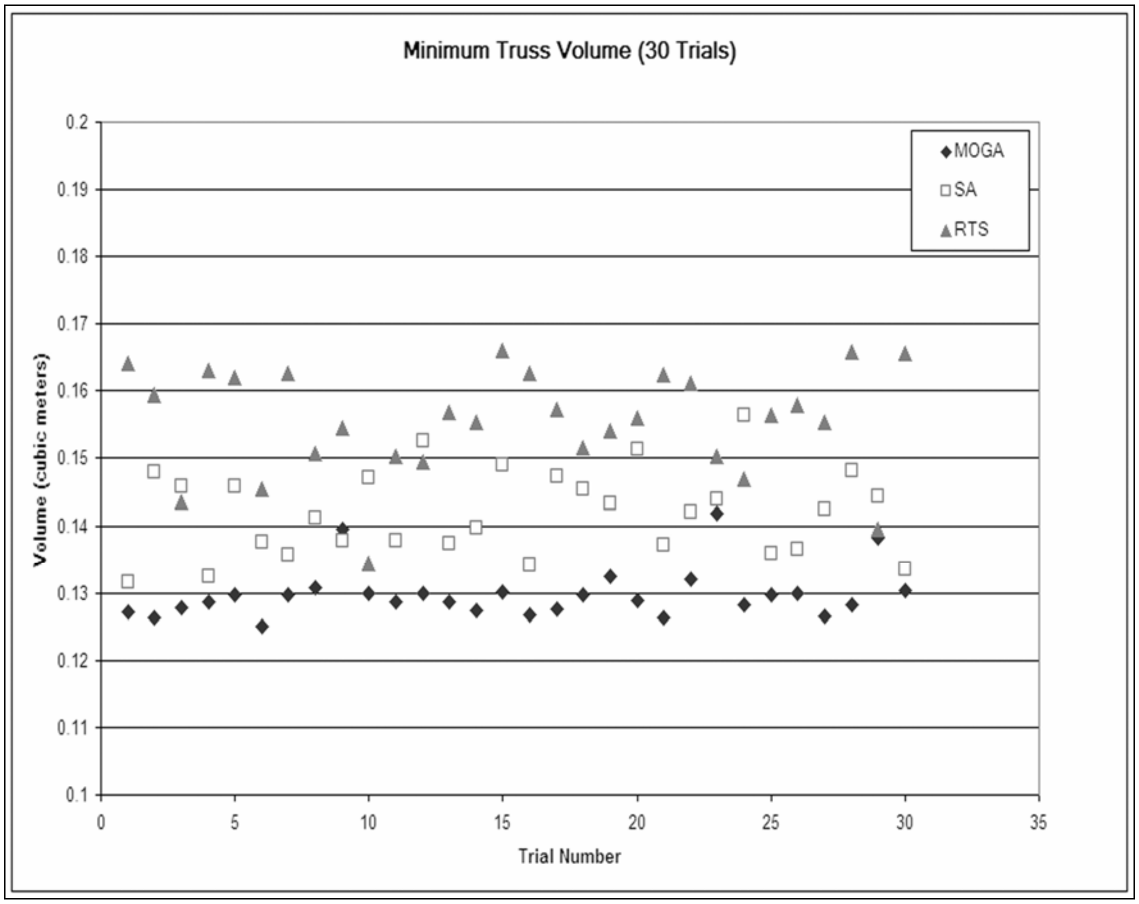

**Fig. 20. Comparison study: Ability to obtain near-optimal truss designs**

It was observed that on average the MOGA performed better than both the SA and RTS. In addition, SA was more consistent in reaching near-optimal solution than RTS. Figure 21 shows the mean and standard deviation obtained over thirty trials for each method. The MOGA clearly out-performs the SA and RTS methods in reaching more optimal truss designs. In addition, the standard deviation of the MOGA results were also much less than the standard deviation of the SA and RTS.

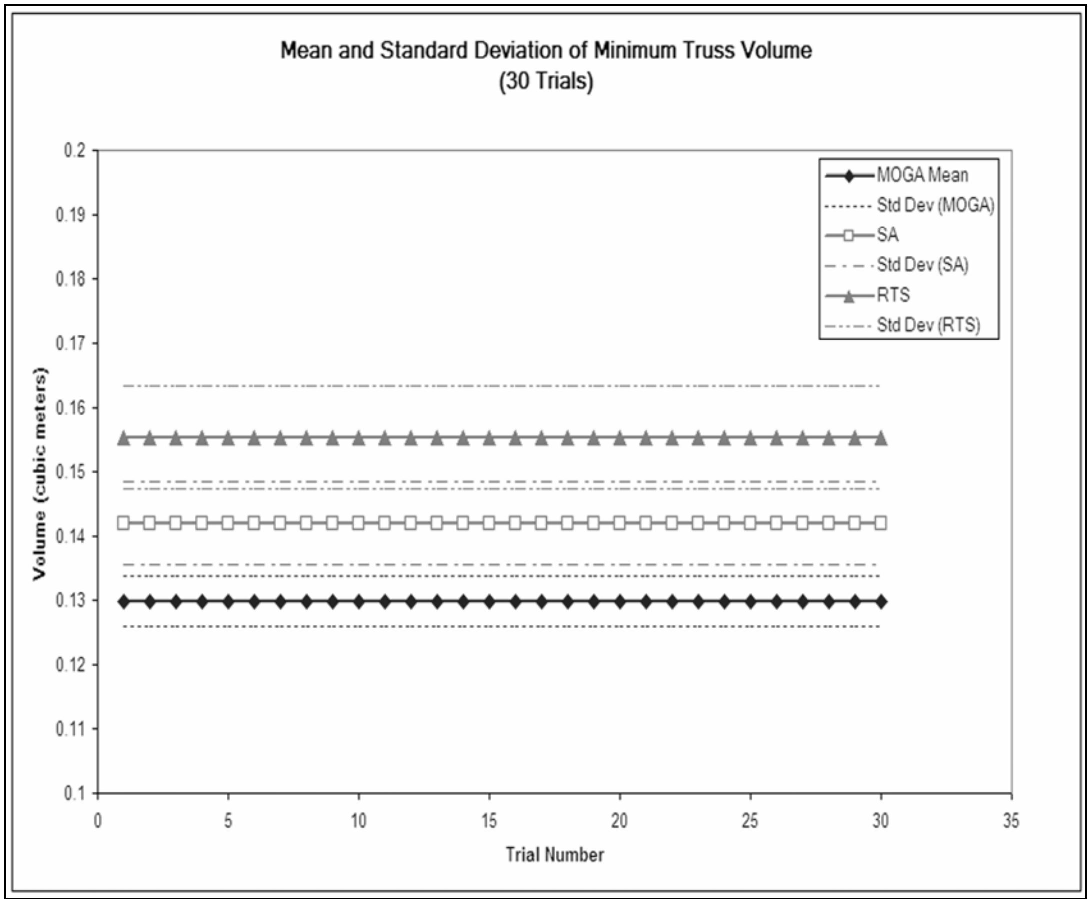

**Fig. 21. Comparison study: Mean and standard deviation of near-optimal truss designs**

Figure 22 presents the member section sizes selected for the most optimal solution obtained by each of the three methods. The volumes of the most optimal solution found over the thirty trials performed were 0.125, 0.132, 0.134 cubic meters for the MOGA, SA and RTS methods, respectively.

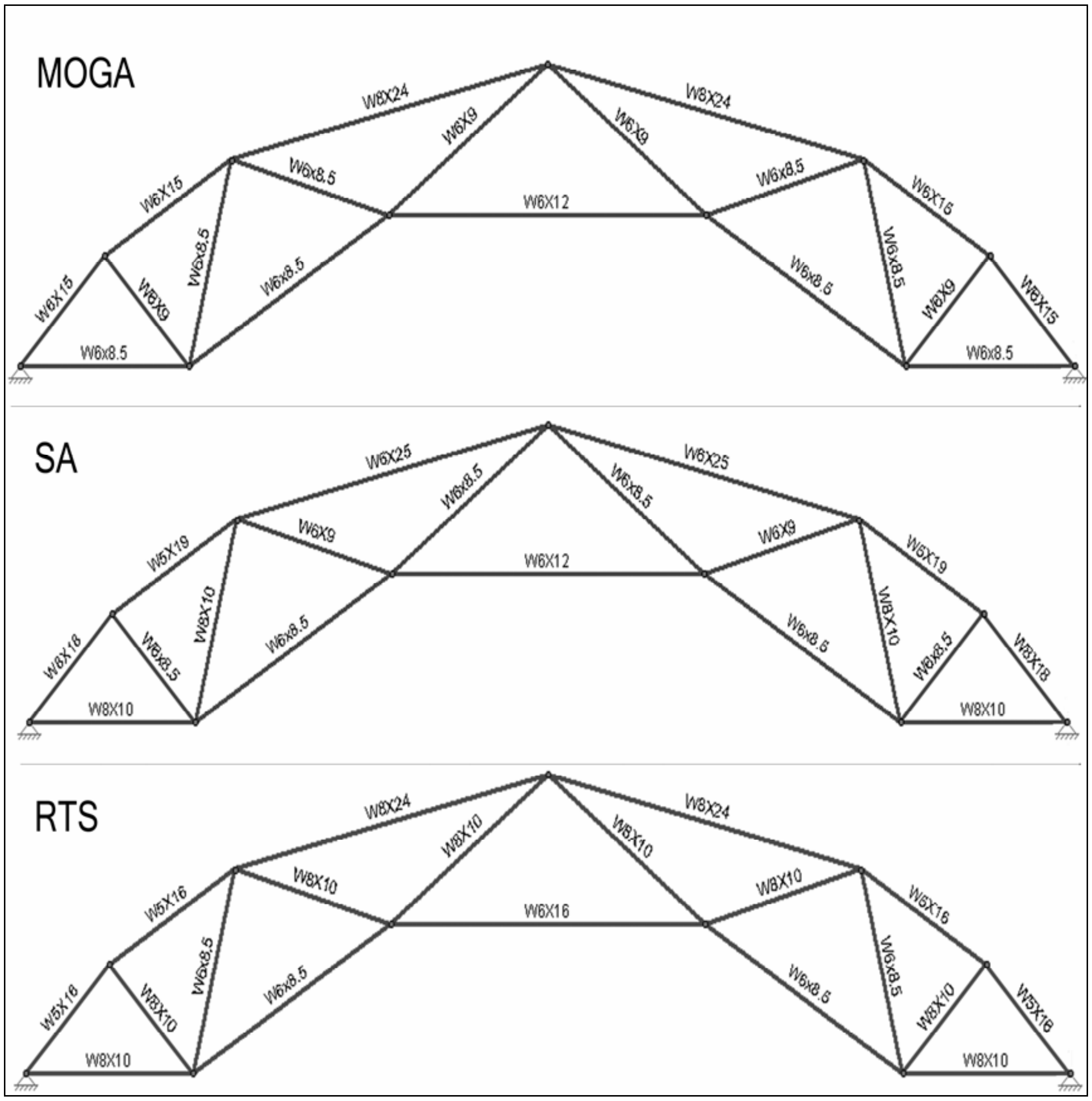

**Fig. 22. Comparison study: Near-optimal truss designs obtained by each heuristic method**

## COMPUTATIONAL EXPENSE

The overall computational expense of implementing each method is calculated as the number of objective function evaluations required to obtain an optimal solution for each trial. Thirty trials were carried out using each method. The average number of iterations per trial and their standard deviation in reaching optimal solutions (Volume ≤ 0.1416 cubic meter) were calculated.

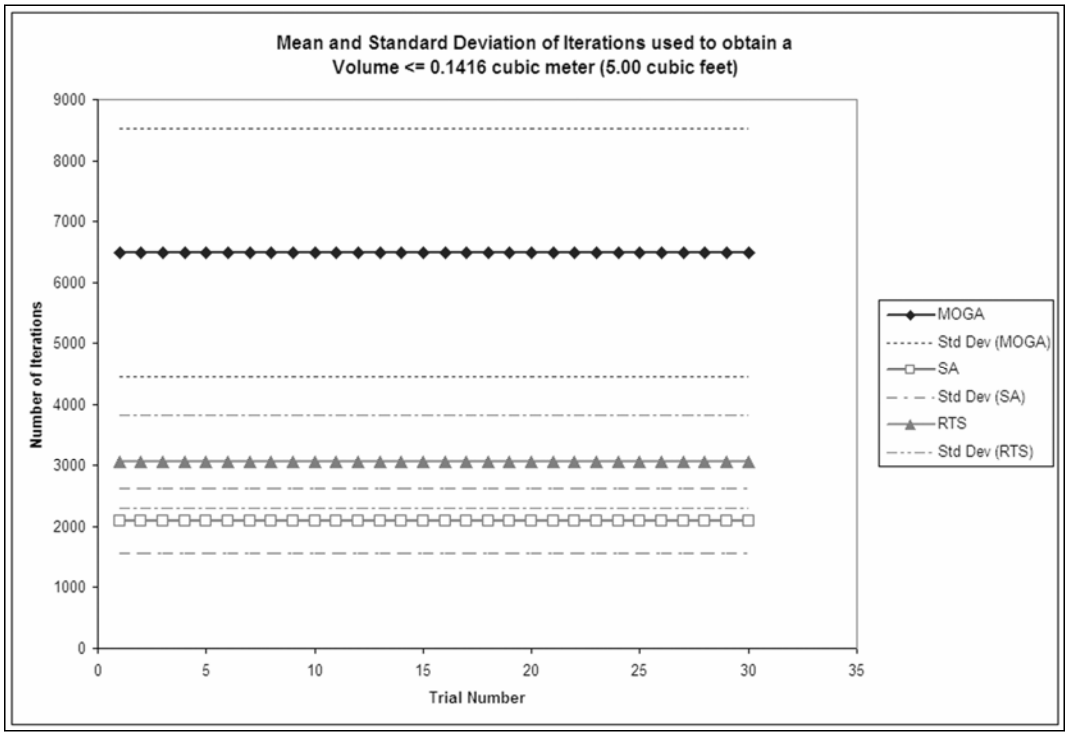

**Fig. 23. Comparison study: Mean iterations and their deviation from the mean**

All thirty MOGA trials resulted in volumes less than 0.1416 cubic meters. Only fourteen of the SA trials and two of the RTS trials achieved the stated performance level. Figure 23 shows the average number of iterations and their deviation from the mean required by each method. It was observed that MOGA required more computational time than either of the other two methods. A trade-off exists, however, between the computational effort required and the quality of results obtained. The MOGA in general, obtains better convergence towards the global optimum, but requires more time. The SA was very efficient with consideration to the total computational expense. Out of the three methods, the SA was able to reach a less optimal solution in the least number of iterations. RTS has a special feature of preserving a list of previously evaluated solutions, which helps it reduce the total computational expense. Both the SA and RTS had less computational expense, but were not able to obtain lighter trusses than the MOGA.

## REPEATABILITY OF SOLUTIONS

Each method was also evaluated to determine the repeatability of obtaining optimal solutions. For the above mentioned truss, the optimal solution was arbitrarily chosen to be less than 0.1416 cu m (5 cu ft). These solutions also satisfied the given stress and deflection criteria.

Figure 24 shows the percentage optimal solutions found be each method over the thirty trials. The MOGA was able to find an optimal solution in all trials. The SA in comparison had a repeatability level of fourteen of thirty trials. The RTS was the least efficient. Only two optimal results were obtained in thirty trials.

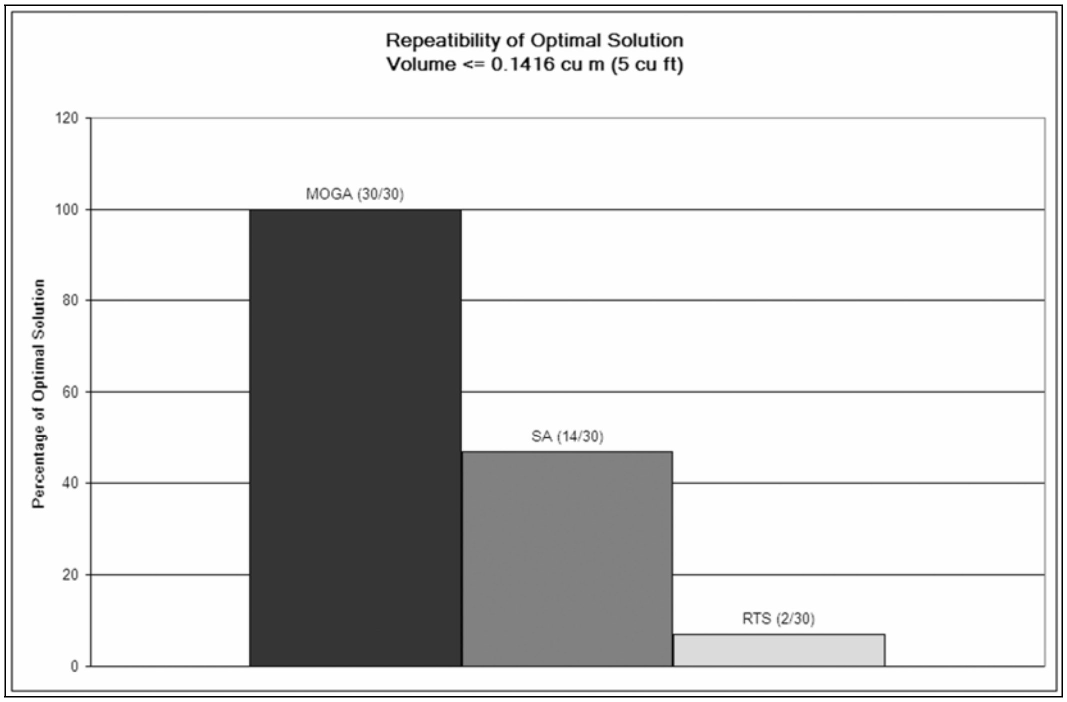

**Fig. 24. Comparison study: Repeatability of optimum results**

## SENSITIVITY TO PARAMETER SETTINGS

In order to determine the robustness of each method, a test for parameter sensitivity was carried out by varying the initial parameters of the MOGA and SA. The RTS, unlike the conventional taboo search, is less dependent on the initial parameter settings. The only RTS parameter that has an affect on performance is the threshold value, which is used to perform a quick escape from the local optimal. Therefore, the sensitivity study was not conducted for the RTS method. The rate of crossover and mutation were varied for MOGA, while the initial temperature, rate of cooling, and the number of cycles to be performed at each temperature were varied for SA.

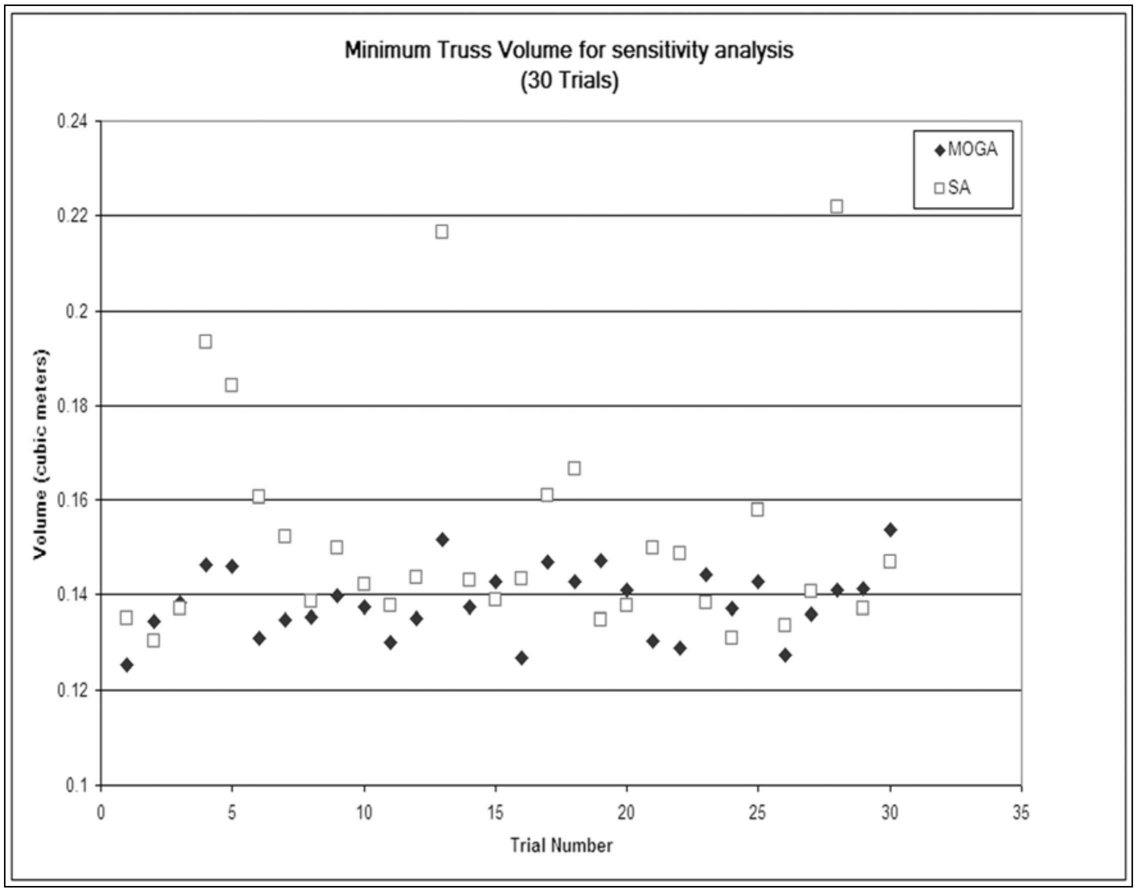

**Fig. 25. Comparison study: Sensitivity to parameter settings trial results considering minimum volume**

Figure 25 shows the most optimal solutions obtained over the thirty trials using different sets of initial parameters. Figure 26 shows the mean value of the most optimal solutions over the thirty trials. The results indicate that the MOGA is more robust with respect to initial parameter settings than the SA. This is evident by the lower value of the mean and standard deviation of the solutions obtained by the MOGA in comparison to those obtained by the SA.

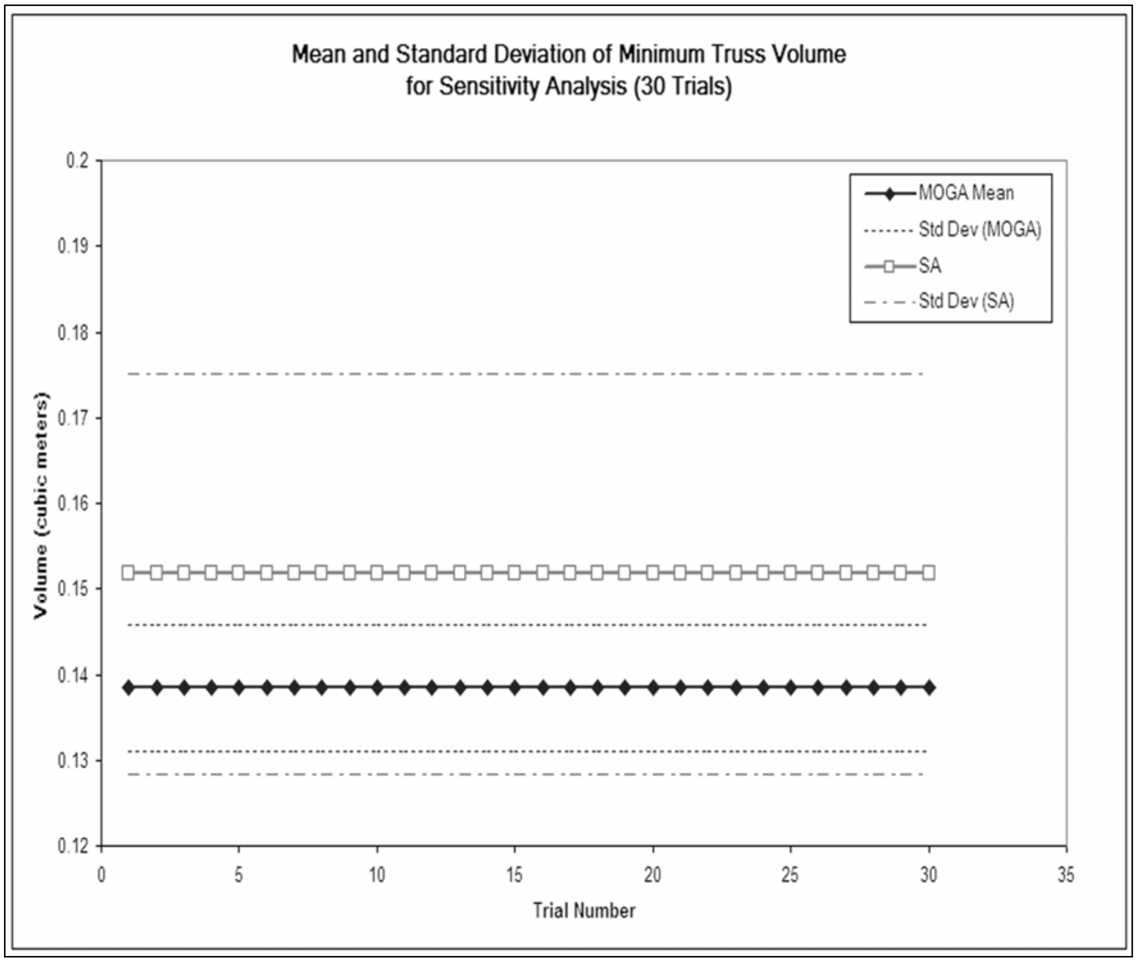

**Fig. 26. Comparison study: Sensitivity to parameter settings trial results for mean and standard deviation of volume**

As is evident from the presented results, the MOGA and SA methods provide an effective mechanism for producing near-optimal solutions for two-dimensional truss problems. A single trial run, on average, using either method takes less than ten seconds to find an optimal solution. Among the three heuristic methods under consideration, the MOGA provides the best performance. Although it required a higher computational expense, the MOGA is the most efficient in nearing the global optimal and showed hundred percent repeatability of optimal solutions for the thirty trials performed. The MOGA is also more robust to the initial parameter settings, in comparison with SA.

The SA requires the least computational time to reach an optimal solution. The main demerit of using SA for truss optimization is its sensitivity to the initial parameter settings. Once these parameters are optimized, the SA provides good results. The SA performance suffers from a lower repeatability in obtaining optimal solutions. The SA could be useful if good solutions are desired quickly and finding the near-optimal is not required.

The RTS provides the least efficient method among the three methods investigated in this research. RTS is relatively inefficient in producing a solution in close proximity to the optimum, while also having a significantly reduced performance in repeatability of near-optimal solutions. The strength of RTS, however, lies in a lower computational expense and being independent of the initial parameter settings. Once the parameters for the RTS method are set, almost no changes or addendums for fine tuning are required to produce better results.

All the methods discussed in this research are simple to implement. The computer implementation of these methodologies is very quick and is highly efficient, especially when compared to their benefits. Once the framework is completed and the parameters tuned, these methods can be applied to a wide range of engineering problems. The MOGA and SA methods especially promise to be highly effective in solving complex design problems.

# CHAPTER VIII

# GEOMETRY OPTIMIZATION USING LOCAL PERTURBATIONS

After optimizing the member sections of a truss using one heuristic techniques, a final attempt is made to optimize the geometry by mutating the nodal locations. In some cases, this additional step in the overall design process is able to improve the optimality solution significantly. This is because although GAs are good at global search, often they might not converge to the exact optimum. Often GAs require a local search method that starts with the GA solution and fine-tunes it. The objective of minimizing the total weight of the structure subjected to stress and deflection constraints is stated, which is the same as the objective of Stage II. Structures that fail to meet the allowable stress and deflection limits are eliminated from consideration. Stage II exports the optimal designs to the geometry optimization program provided by Stage III. Any of the trusses investigated previously can undergo geometry optimization in Stage III.

The local perturbations are imposed by shifting the nodal locations in the defined truss layout in the vertical direction. Each set of symmetric nodes is shifted by one foot in the upward and downward direction. Therefore, for a truss with ‘n’ symmetric nodes (total ‘2n’ nodes excluding the support nodes) implementing the local perturbation will produce a population of ‘2n’ trusses. The entire population of trusses is analyzed using the finite element solver. The fittest individual in this population, i.e., the truss with the least weight that satisfies the allowable stress and deflection criteria, is chosen as the new solution for the next iteration.

The nodal locations of the selected solution are then mutated in a similar manner to create a new population of ‘2n’ trusses. The fittest individual is then selected for the next iteration. Figure 27 describes how the geometry of a particular parent truss is mutated to produce a population of trusses for the next iteration. The process of selecting a solution based on fitness and creating a new generation is carried out, until there is no further improvement in solution. A boundary constraint acts on the generated trusses. It prevents the trusses that cross the design domain from being considered for analysis. The final solution obtained is required to satisfy the stress and deflection criteria.

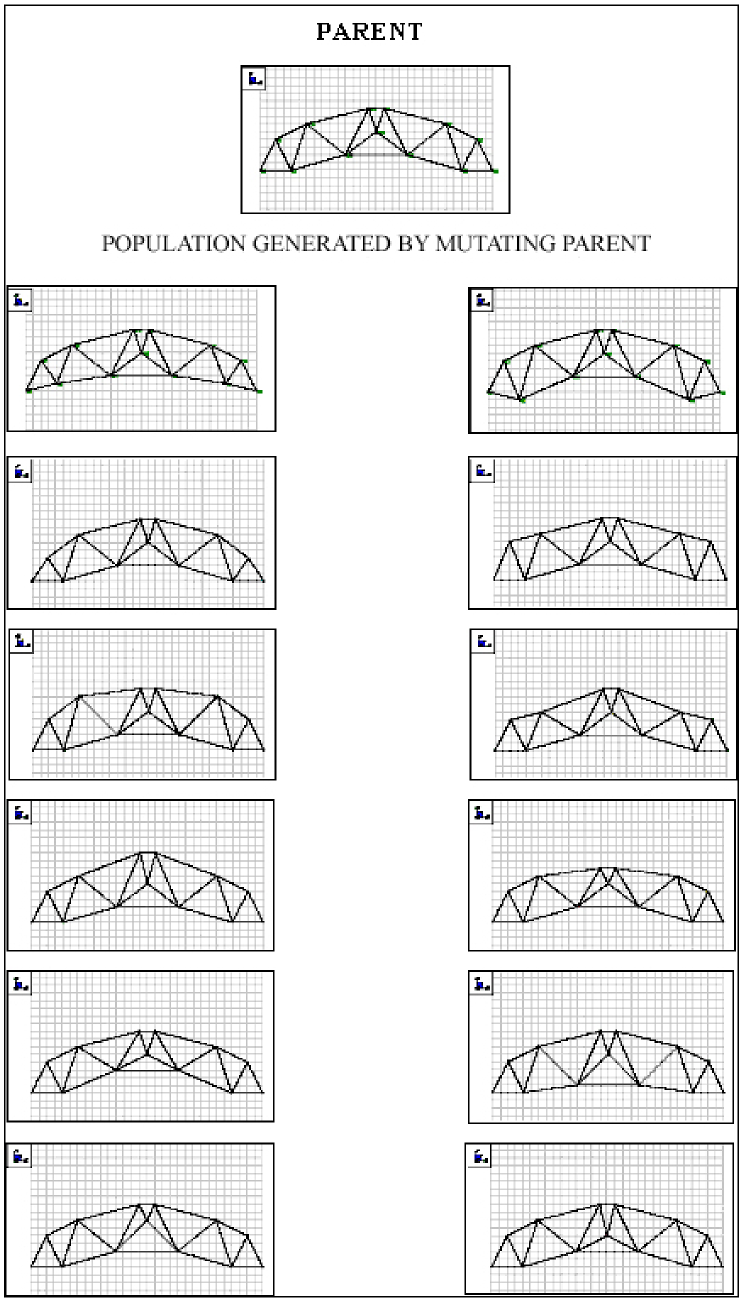

**Fig. 27. Generation of new population in Stage III using local perturbations of nodal locations**

Figure 28 represents the overall local optimization process that takes place over a several iterations. The figure shows that there is an improvement in the optimality of the design over the three iterations shown. As the overall weight of the truss decreases, there is also a tendency of the truss to move towards a parabolic geometry. This is an influence of the support conditions and loading applied for this problem domain. The final truss design produced through the final optimization procedure is lighter than the truss imported from Stage II. The overall computational time required to perform the geometry optimization in Stage III is negligible, since each iteration requires only a few milliseconds.

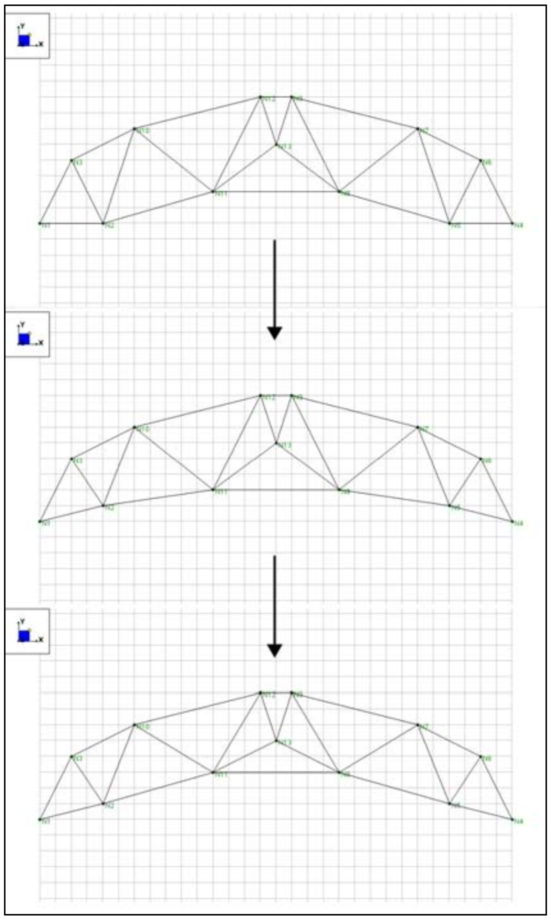

**Fig. 28. Improvement of truss designs over three iterations of local geometry optimization**

It is also important to note that Stage III might not always increase the optimality of the results obtained from Stage II. This is because the final results of Stage II might already be the best geometric configurations for the corresponding section sets.

The final truss design can be exported to commercially available structural analysis software packages like RISA-3D and STAAD PRO. Complete truss details, which include the nodal locations, member connectivity matrix, loads, and support locations, are stored in a data file in a format compatible with the software. This information will assist the engineer in integrating the optimized truss designs obtained by the hybrid optimization method into the overall structural system for further design and analysis.

# CHAPTER IX

# STAND ALONE AND PARALLEL IMPLEMENTATION

The hybrid optimization method developed in this research has been implemented on both a single machine and a parallel architecture. The objective of carrying out this research on a single machine is to assist in facilitating human-computer interactions during the design process. This interaction is desired in order to produce a final truss design that matches the user's design requirements. This program allows the user to have control over a variety of problem domain parameters that have an influence on the final design obtained. In addition, there is an option for the user to re-run entire stages if they are not satisfied with the final truss designs obtained. A front-end user interface was created to help the user in switching between the three different stages in the hybrid method. The entire optimization process in this implementation has been compiled and executed using WIN32 and DOS-SHELL programming in Microsoft Visual C++.

The parallel implementation is focused on minimizing the total computational required to optimize truss designs for a three-dimensional system. The program automatically considers three different truss spacings in the third dimension and suggests the most optimal spacing among the three to the user. This information helps the user determine spacing interval of trusses to select along with the sizing and geometry optimization of the truss. The parallel implementation is performed on a four-node parallel platform (Beowulf Cluster) in the Civil Engineering Department at the Texas A&M University. The 'Message Passing Interface' library available in C++ was used to communicate between the four nodes. The compilation and execution in this case was done on the UNIX platform (Agarwal and Raich 2004).

The following sections present the results obtained from trials performed using the stand alone and parallel implementations of the program.

## STAND ALONE IMPLEMENTATION

For an example stand alone program trial, a set of hundred trusses were generated in Stage I. The design domain specified for the trusses was a span of 18.288 meters (60 ft) and a maximum height of 9.144 meters (30 feet). Figure 29 shows the wide range of truss topologies that are produced using method developed in this research, which is discussed in Chapter V.

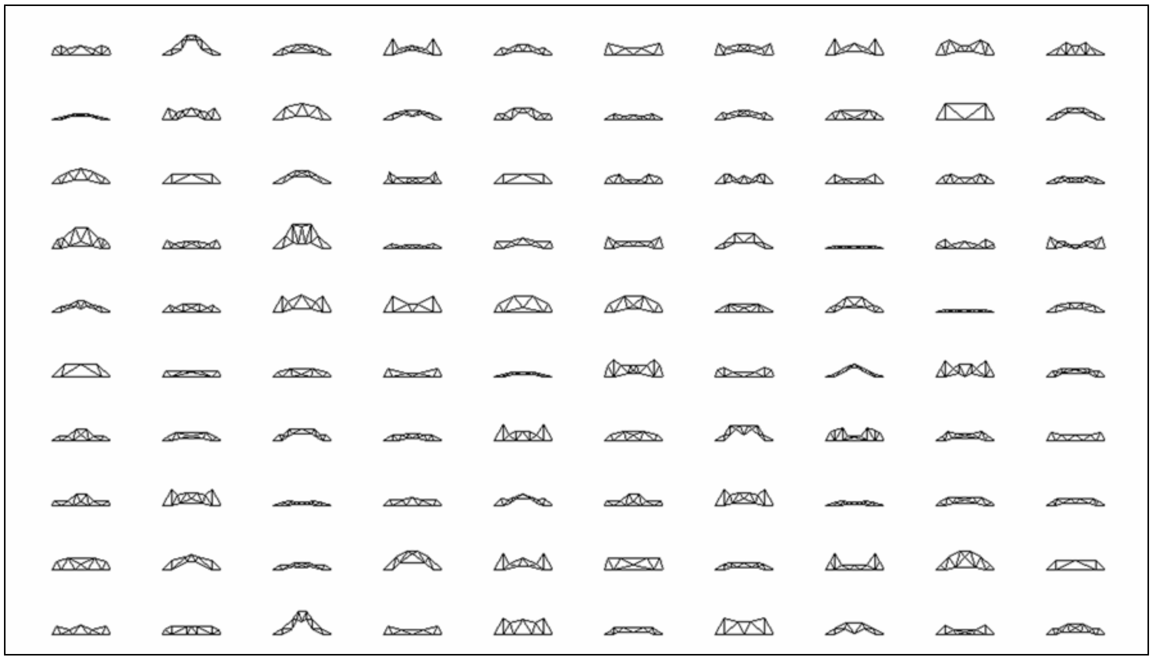

**Fig. 29. Stand alone implementation: Results of Stage I topology generation**

The user is given the option to select a subset of desired topologies from all the generated trusses. The ability to control the selection process helps in directing the search process towards meeting the user's design requirements. The population of hundred trusses is reduced to twelve before processing Stage II using the controlled selection process. One goal of this program is to expedite the overall process of design and to neglect those trusses that do not satisfy user's requirements, which include subjective criteria such as aesthetic preferences.

In Stage II, the multi-objective genetic algorithm was able to perform size optimization on ten out of the twelve trusses selected. It is important to note that not all of the trusses passed onto Stage II may be optimized. This is because a feasible truss design may not exist for a selected truss topology for the selected section set. If however a heavier section set is selected, all trusses may be optimized.

The optimized designs for the ten trusses are shown in Figure 30, which also identifies their corresponding volumes. Structure 8 with a volume of 0.3018 cubic meters is the most optimal design obtained in Stage II. The design obtained will depend on the trusses selected at the beginning of Stage II. In addition the designs depend on the member section set selected.

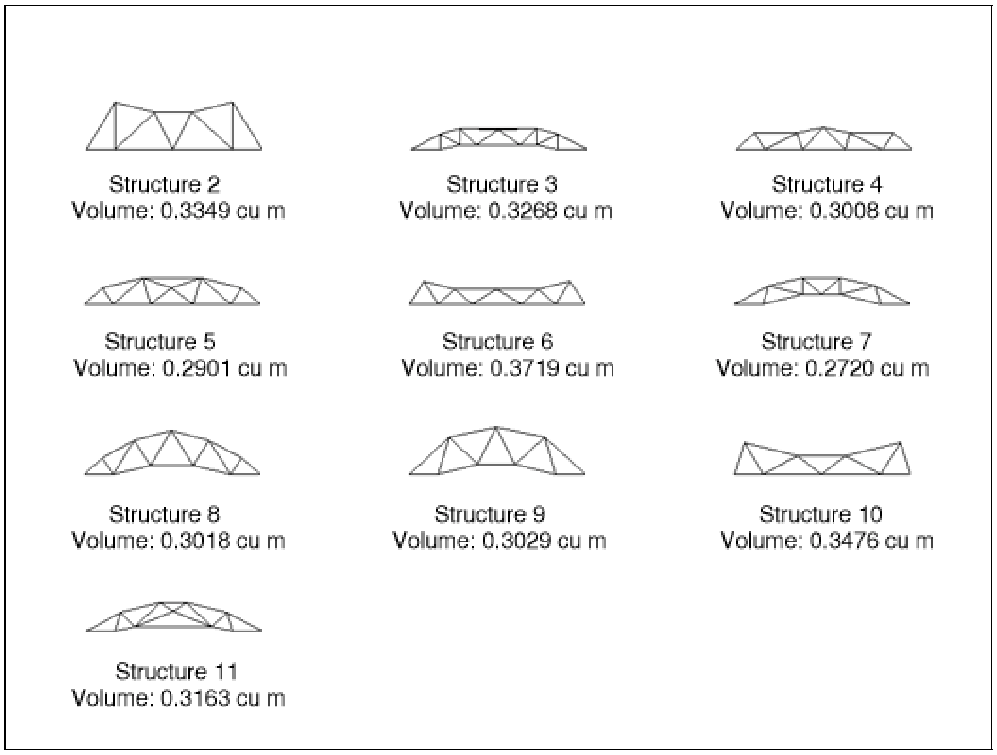

**Fig. 30. Stand alone implementation: Results of sizing optimization using MOGA performed in Stage II**

The design results obtained in Stage II are saved in data files that are automatically used to perform the geometry optimization in Stage III. By using the local perturbation method discussed in chapter VIII, the volume of the final truss design is further reduced by 16.3 %. The height of the truss is reduced from 4.901 m to 4.292 meters. The final truss design obtained satisfies the stress, slenderness and displacement criteria. Figure 31 shows the transformation in nodal geometry obtained after Stage III is completed.

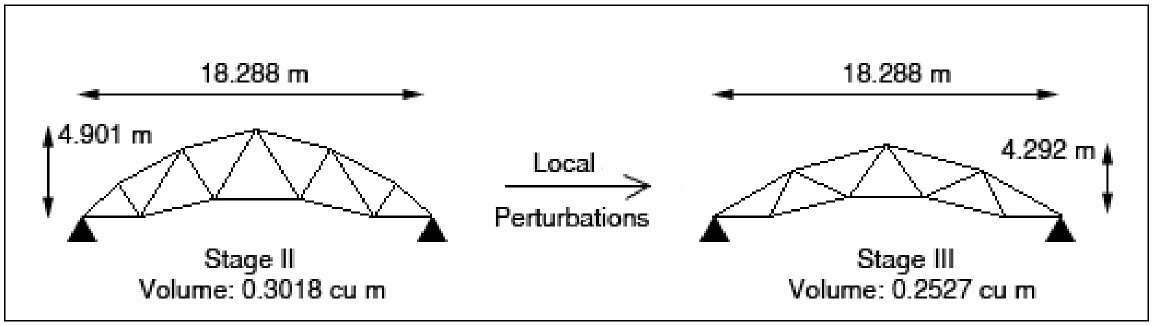

**Fig. 31. Stand alone implementation: Results of geometry optimization using nodal perturbations performed in Stage III**

## PARALLEL IMPLEMENTATION

The parallel implementation of the three stages takes place simultaneously on four computer nodes. Once the problem domain information is provided, the process only stops after executing all three stages. Therefore, no other human-computer interaction is supported between the stages. The parallel process executes on one Master node and three other nodes. Stage I and III are processed on the Master node, while Stage II (which requires the most computational effort) is processed simultaneously on the other three nodes.

A set of twelve trusses is generated during Stage I that satisfy the design domain limits pre-specified by the user. Figure 32 presents the trial results obtained for Stage I in the parallel implementation.

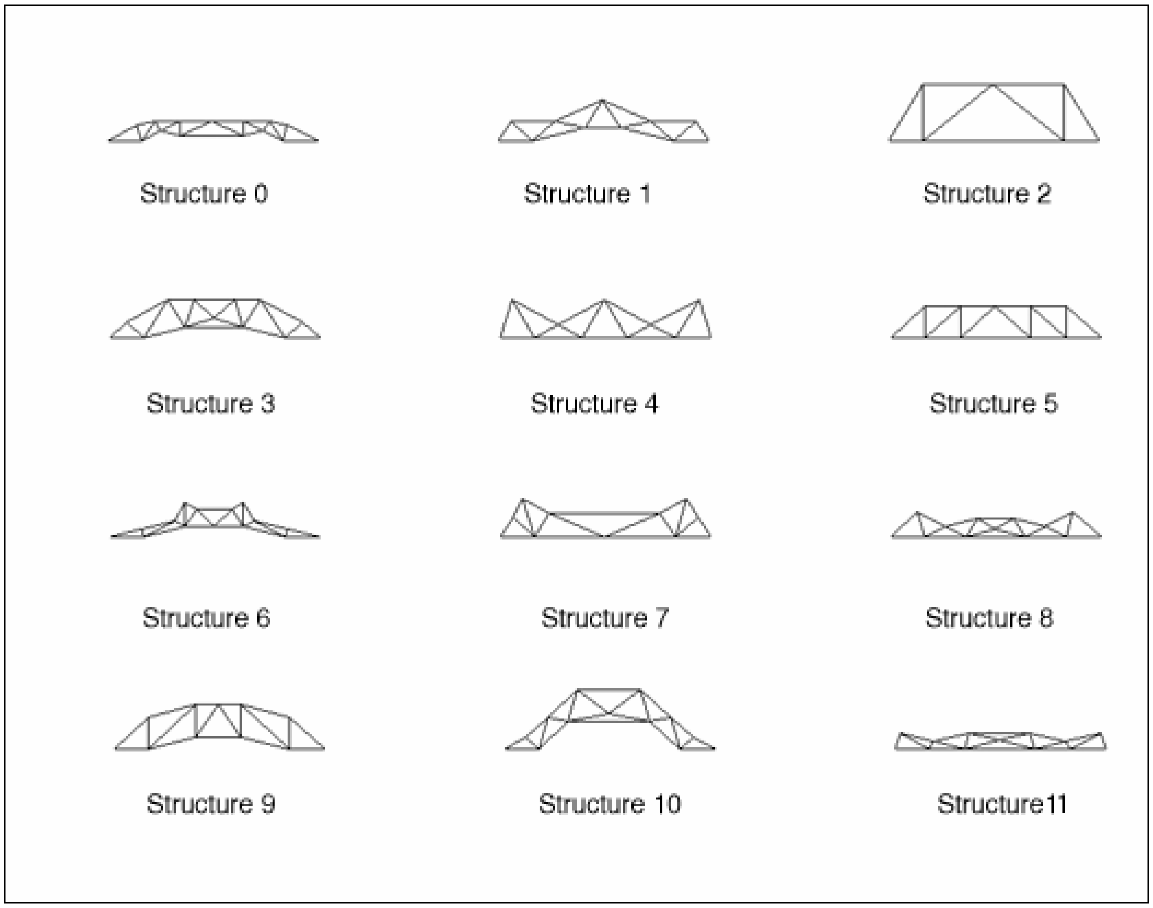

**Fig. 32. Parallel implementation: Results of topology generation after Stage I**

In the parallel implementation of Stage II, the sizing optimization of the trusses is enhanced by considering the most optimal spacing of the trusses in the third dimension. The program assists in predicting an optimal spacing by running the optimization procedure in parallel for three possible spacings. The program optimizes the section sets individually for each spacing and then selects the truss design that provides the minimum volume. As an example, consider a problem in which the total distance to span is 73.152 meters (240 feet) in the third dimension. Seven trusses would be required if spaced at 12.192 m (40 feet), nine trusses are required if spaced at 9.144 m (20 feet) and thirteen trusses are required if spaced at 6.096 m (20 feet). The loading applied to these trusses is calculated according to tributary area, which considers truss spacing, and the total volume is calculated for the entire truss system. Figure 33 shows the near-optimal truss configurations obtained considering the three spacings specified after Stage II.

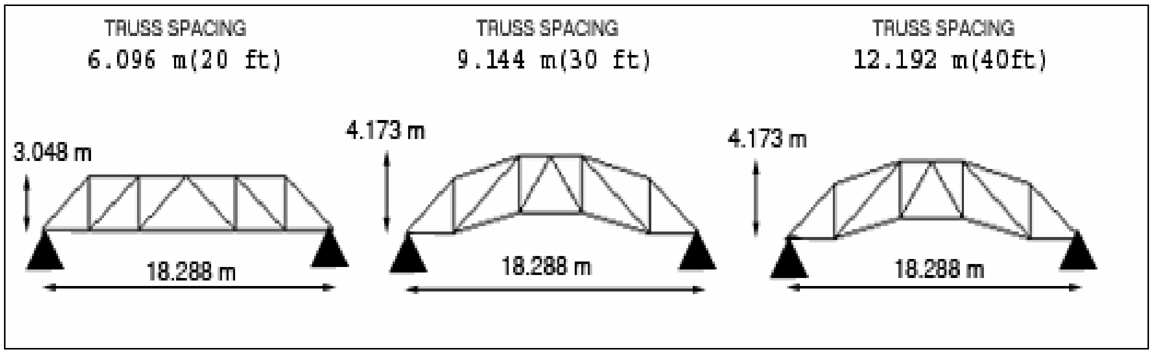

**Fig. 33. Parallel implementation: Results of simultaneous sizing optimization on three nodes in Stage II**

In the MOGA, the volume of the truss is reduced over a number of generations, until no further improvement is found. Figure 34 provides a graph obtained from this specific Stage II trial for the trusses shown in Figure 32 to show how the volume is reduced. A spacing of 12.192 meter (40 feet) is considered. As shown in the figure, the population converges in less than ten generations for each truss optimized.

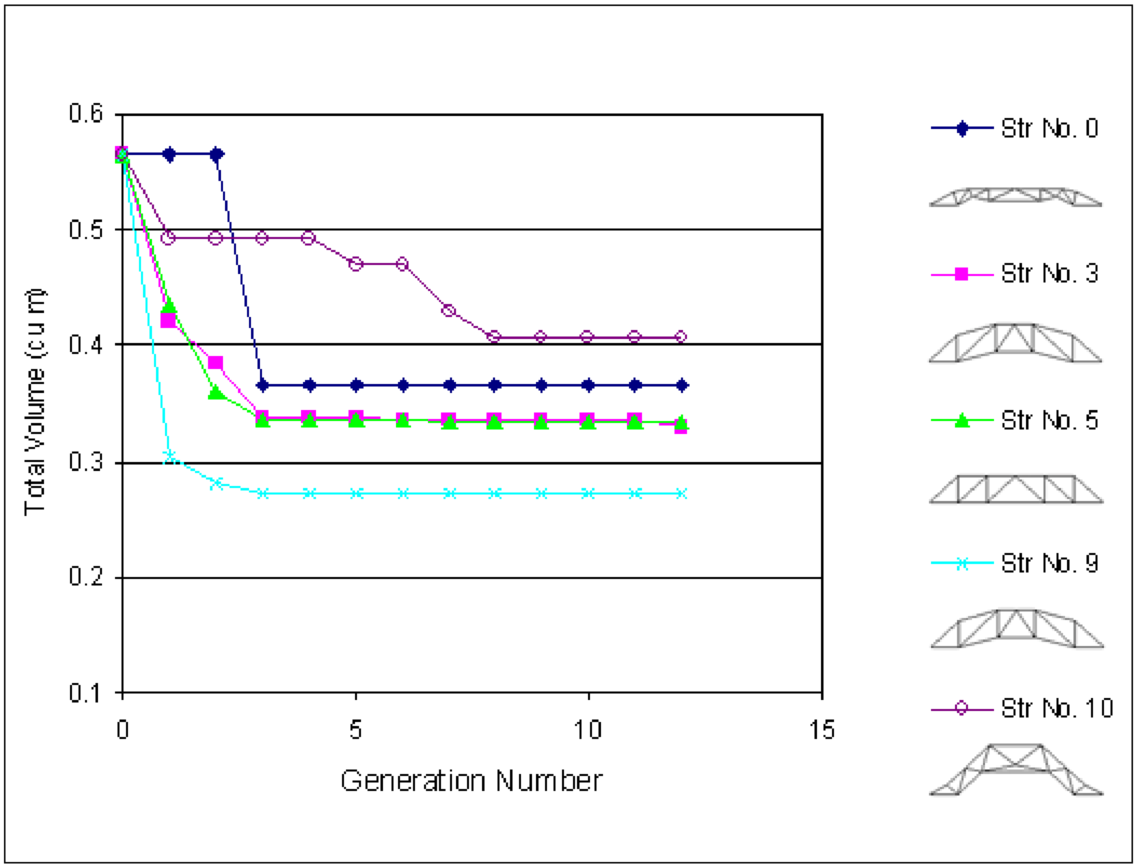

**Fig. 34. Comparison of reduction in volume over the number of generations for each truss design in Stage II**

Considering the third dimension spacing of the trusses over the length of 73.152 meter, the truss designs obtained for a spacing of 12.192 meter (40 feet) resulted in the lightest truss system. These trusses were selected with the corresponding spacing and passed back to the master node. Stage III was implemented and the final truss design as shown in figure 35 was the result.

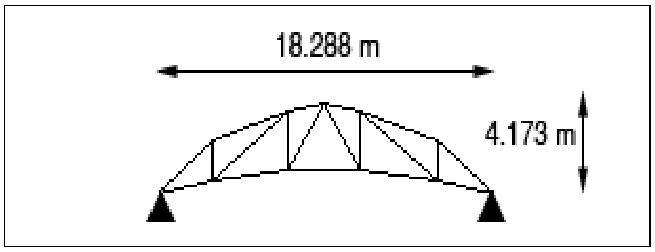

**Fig. 35. Parallel implementation: Results of geometry optimization on the master node in Stage III**

# CHAPTER X

# HUMAN-COMPUTER INTERACTION

As is discussed in the previous chapter, human-computer interactions play a vital role in being able to develop of a final design in consonance with the user's design requirements. Especially since the user's design preferences are often subjective in nature. Aesthetic design criteria are one example of a possible subjective preference. Specifying the dimensions of the problem domain is the first step in meeting design requirements. The overall truss complexity can be influenced by specifying the maximum number of truss members. Specifying a large number of members may lead to smaller member lengths, and more complicated trusses with more members; while a smaller number of members would lead to larger member lengths and fewer nodes. This trade-off can be simplified by looking at connection costs and aesthetic design requirements.

Another interaction that is supported is specifying the location of the first node in from the support. By selecting this nodal coordinate, the user can influence the shape of the trusses generated. Specifying a y-coordinate that lies above the support co-ordinate may lead to more parabolic designs. Selecting a y-coordinate that is in line with the support might lead to trusses that have a flat bottom chord (as shown in figures 4 and 6 in Chapter V). The former case might be more useful when designing roof trusses, while the latter case is more applicable for bridge trusses. Table 4 provides a list of the interactions available to the user during Stage I.

**Table 4. Human-computer interactions for Stage I**

| Interaction | Determines or Influences |
|---|---|
| Design space (X and Y coordinates) | Rectangular search space for truss |
| Maximum number of members | Complexity of truss |
| Coordinates of node next to supports | Shape of truss |

Stage II allows the user to select a subset of all the topologies generated during Stage I. Currently the user can select a set of twelve trusses for optimization in Stage II. The selection of the subset helps in reducing the overall computational time required, while also serving to capture the user's design preferences. The twelve trusses can be chosen by the user after looking at all the generated topologies. The user is also given an option to select the specific section set to be used during sizing optimization. The sets limit the range of wide flange sections examined. The set is picked from five default section set currently available in the program. This allows the user to consider the issue of availability of material. The section set selected may also affect the overall weight of the structure. The user is also allowed to define an inter-truss spacing at this stage. Since the loads applied at the nodes are calculated by multiply this spacing by the tributary member length, this spacing determines the load that must be carried by the truss. Table 5 presents a list of the interactions available to the user during Stage II.

**Table 5. Human-computer interactions for Stage II**

| Interaction | Determines |
|---|---|
| Select trusses from Stage I | Structures in consonance with design requirements |
| Member section set | Overall weight of structure and design requirements |
| Inter-truss spacing | Imposed loads on the truss |

Geometry optimization can be carried out on any truss that was optimized in Stage II. The user is given an option to select a specific truss to undergo geometry optimization. They are also given an option to re-run this stage with any other truss.

# CHAPTER XI

# COMPARISON OF RESULTS ON BENCHMARK PROBLEMS

## DISCUSSION OF BENCHMARK BRIDGE TRUSS PROBLEM

A comparison study was conducted in order to evaluate the performance of the computational method developed in this research. Results obtained for a bridge truss problem using the methodology developed in this research were compared with the results obtained by Shrestha and Ghaboussi (1998) using a simple genetic algorithm and Yang and Soh (2000) using a fuzzy logic integrated genetic programming methodology. The design domain and total loading specified in all three studies were the same. Figure 36 provides a detailed view of this design domain. The location of the nodal loads was fixed in the two previous studies. In this research, the location of nodal loads is not fixed. Instead it depends on the topology and geometry of each truss generated. In this way, a broader range of truss designs is explored. The improved performance obtained in this research, therefore, stems from working with a more flexible design domain formulation that allows a broader range of design alternatives to be explored. By considering truss spacing as an additional design variable, a three-dimensional unstructured problem domain can also be investigated.

A set of thirty-two member sections was designated that ranged in size between W14x22 to W14x426 from the AISC ASD Manual (AISC, 1992). Steel material is specified (E = 201 GPa, fy = 248.8 MPa, and $\rho$ = 7851.03 Kg/m$^3$). Since the AISC ASD design specifications were used to obtain the previous research results, they are also used in this research. Therefore, the allowable tensile stress is 0.6fy; the allowable compressive stress is determined based on buckling considerations, the allowable slenderness ratios are 300 for tension members and 200 for compression members; the member length was constrained between 5 m and 35 m; and the maximum vertical nodal displacements were 1/1000 (i.e., 70 mm) of the total span.

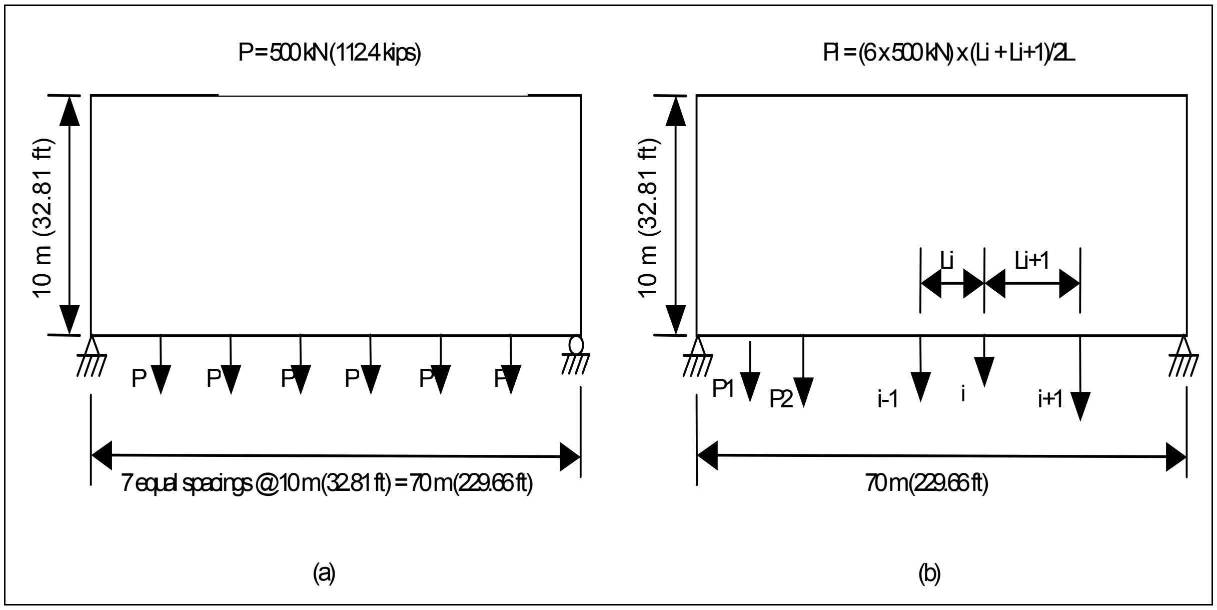

**Fig. 36. Comparison of the unstructured problem domain for the benchmark bridge design problem using (a) Fixed nodal loading and (b) Variable nodal loading**

## DISCUSSION OF BENCHMARK BRIDGE TRUSS RESULTS

After generating a large variety of truss topologies and geometries in Stage I, several of these trusses were passed onto Stage II for sizing optimization. Figure 37 shows the trusses optimized using Stage II for a single trial. Each of these trusses was optimized by the MOGA heuristic method. All the trusses shown satisfy the stress, slenderness and displacement criteria. Therefore, their relative efficiencies is given by their total weight.

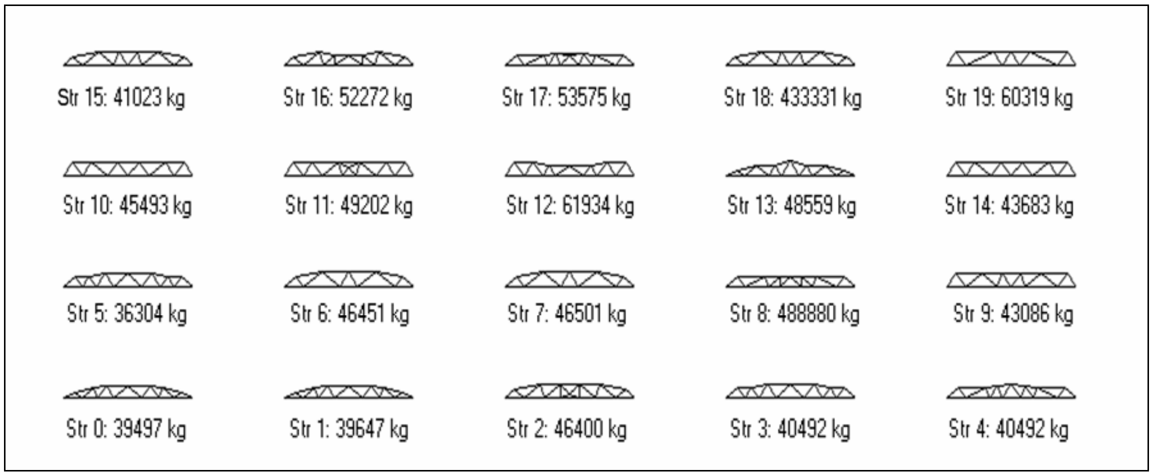

**Fig. 37. Optimized weight of bridge truss designs obtained using the proposed MOGA method**

One of the salient features of this research is the large number of design alternatives that are available at the end of Stage II. Unlike previous studies, where the user is provided with only one optimal truss as the final result, this research allows the user to compare design alternatives with an option to select among them. e.g. Structure 3 and 4 have the same total weight, but have different topologies and geometries. In this trial for example, the user can select any of the twenty structures shown in figure 37 as the final design.

The most optimal truss generated in the trial presented is Structure 5 with a weight of 36,304 Kg. Figure 38 and Table 6 provide the details concerning the truss topology and nodal loads. As is evident from the figure, the nodal loads are calculated in proportion with the tributary lengths associated with each node. The Figure also provides the weight in pounds per foot for each W14 section used in the truss.

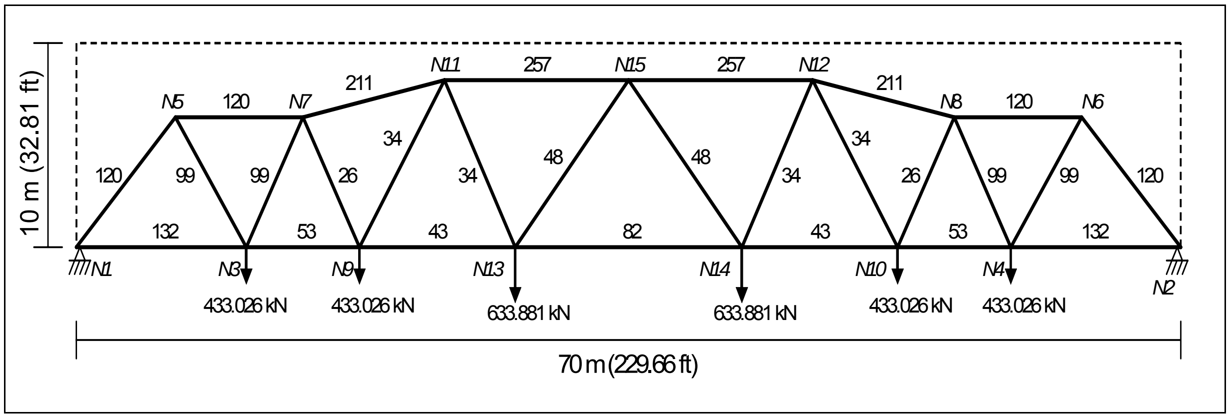

**Fig. 38. Details of a near-optimal design for the bridge truss problem domain**

**Table 6. Nodal coordinates for the optimal bridge truss design after Stage II**

| Node | X(m) | Y(m) | Node | X(m) | Y(m) | Node | X(m) | Y(m) |
|---|---|---|---|---|---|---|---|---|
| N1 | 0 | 0 | N6 | 63.66 | 6.34 | N11 | 22.32 | 8.66 |
| N2 | 70 | 0 | N7 | 13.66 | 6.34 | N12 | 47.68 | 8.66 |
| N3 | 10 | 0 | N8 | 56.34 | 6.34 | N13 | 27.32 | 0 |
| N4 | 60 | 0 | N9 | 17.32 | 0 | N14 | 42.68 | 0 |
| N5 | 6.34 | 6.34 | N10 | 52.68 | 0 | N15 | 35.00 | 8.66 |

It is difficult to directly compare the truss designs obtained using the proposed method with those obtained by other researchers due to the varying locations of the loaded nodes along the bottom span and the different boundary conditions imposed. Improved designs are obtained in this research because this proposed method works with a more flexible problem domain formulation that considers a broader range of design alternatives that have different topology and geometry, especially in the locations of the loaded nodes along the bottom chord. The optimal truss produced in the trial presented using the MOGA was 39.82 % lighter than Shrestha and Ghaboussi (1998) and 19.62 % lighter than Yang and Soh (2000). These results were verified by exporting the design information to RISA-3D and re-running a complete analysis.

An optimal comparison of the number of iterations is difficult to make between the three research methods. This is because the number of iterations in the present work depends directly on the number of trusses selected for optimization during Stage II. The number of iterations can be calculated as follows:

# of iterations = # of Selected Structures * Population Size * # of Generations (11)

In the trial presented, the number of structures was twenty; therefore the maximum iterations possible were 200,000 (population size of 100 and maximum generations of 100). However, almost all structures converged much earlier than the maximum number of generations, with some converging in less than 10 generations. The total number of iterations in general is less than half of the maximum value calculated above. If only one structure is passed onto Stage II, unlike the twenty structures passed in this trial, the number of iterations could be further reduced to a maximum of 10,000 iterations. Soh and Yang obtained optimal results in 100,000 iterations and Shrestha and Ghaboussi took almost ten times this number to obtain their optimal results. Table 7 provides a summary of the computational results.

**Table 7. Comparison of computational results for truss optimization**

| Parameter | Shrestha and Ghaboussi (1998) | Soh and Yang (2000) | Raich and Agarwal (2005) |
|---|---|---|---|
| Weight (Kg) | 60,329 | 45,163 | 36,304 |
| Population Size | 100 | 2,000 | 100 |
| Generations | 9,754 | 50 | < 100 |
| Truss Layouts | 1 | 1 | 1 - 20 |
| Total Evaluations | 975,400 | 100,000 | 1,000 - 200,000 |

In addition to the significant reduction in total weight, the method developed in this research effort provides a wide range of design alternatives that allows the user to better satisfy their specific design criteria. The user has the flexibility to restrict the trusses considered to a specific truss topology or geometry or to expand the search and consider a broader range of trusses.

## ESTABLISHING A BENCHMARK ROOF TRUSS DESIGN PROBLEM

Due to the unavailability of a benchmark long-span roof truss problem in the literature, a roof truss design problem is defined to assist in further evaluating the performance of the proposed design method. The problem domain dimensions are a span of 15.24 m (50 ft) and a height of 6.096 m (20 ft). The center to center truss spacing perpendicular to the truss is defined as 12.192 m (40 ft). The factored roof area design load is 4.788 MPa (100 psf). The member sections were selected from a set of 16 standard sections ranging between W4x13 to W8x24 from the AISC LRFD (2001) Manual. The material properties are those of steel (E = 201 GPa, $F_y$ = 344.736 MPa, and $\rho$ = 7851.03 $Kg/m^3$). The AISC LRFD design specifications (2001) were used, along with an allowable slenderness ratio of 300 for tension members and 200 for compression members and the maximum nodal displacement of 25.4 mm (1 inch).

## DISCUSSION OF BENCHMARK ROOF TRUSS RESULTS

Figure 39 presents fifty trusses produced using Stage I in one trial. All of these trusses lie within the specified problem domain, while also satisfying the complexity and shape requirements specified by the user.

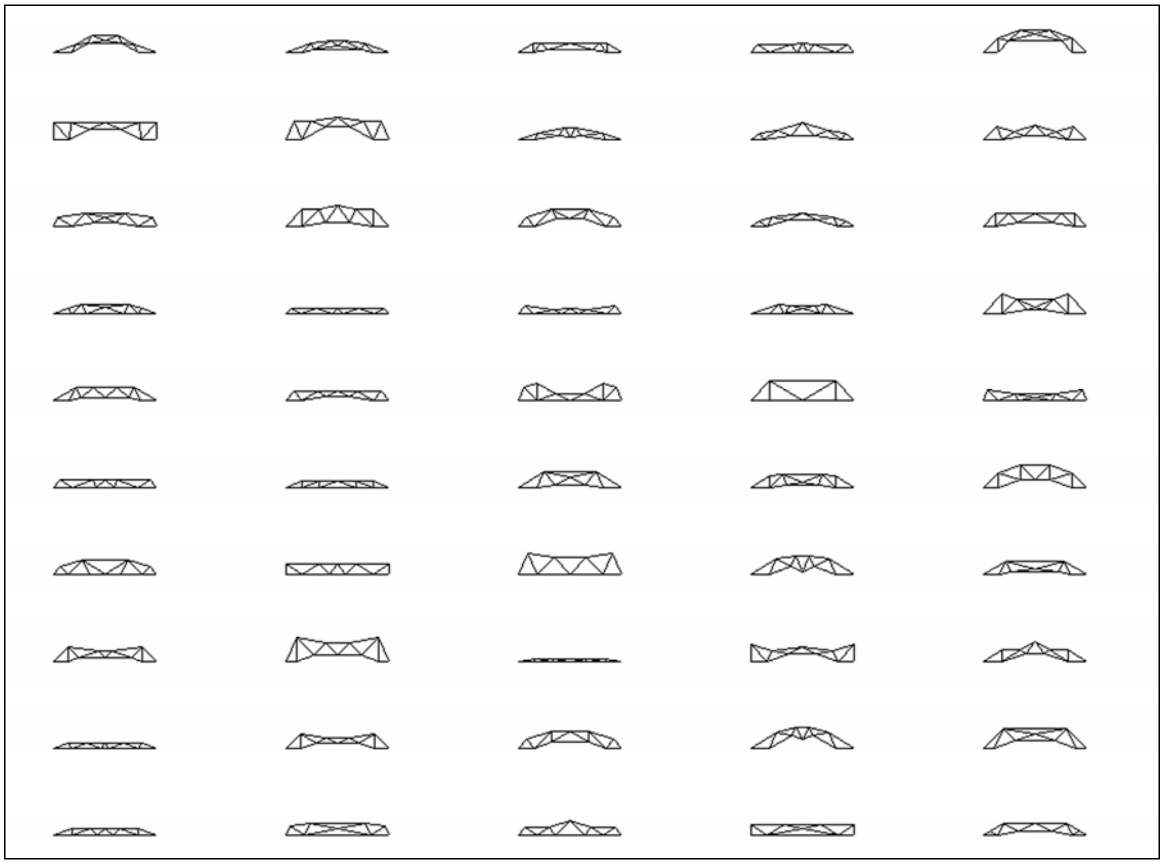

**Fig. 39. Truss topologies generated for the unstructured roof truss problem domain**

Nineteen of these fifty trusses were selected for optimization using MOGA in Stage II. Figure 40 identifies the near-optimal designs and their respective weights. All these trusses satisfy the stress, slenderness, and displacement criteria.

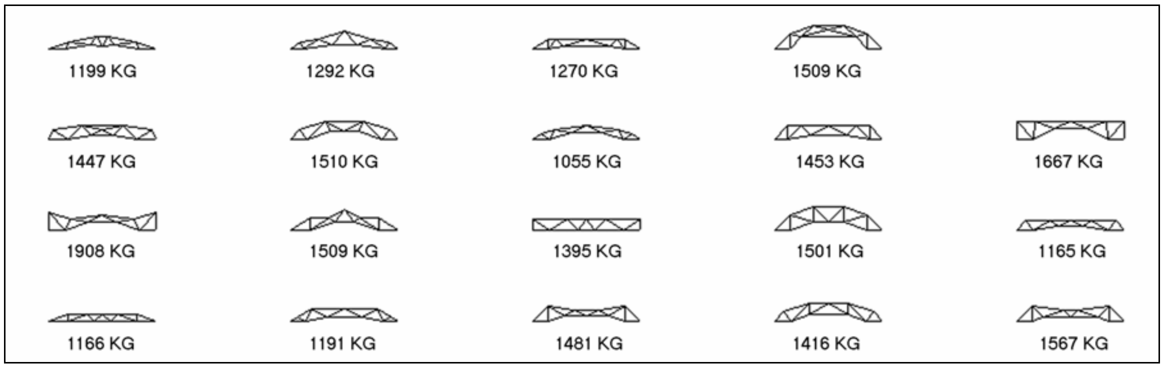

**Fig. 40. Optimized weight of roof truss designs obtained using the proposed MOGA method**

The most optimal truss obtained during Stage II was further optimized by using geometry perturbations in Stage III. A 2 % reduction in total weight was obtained. The final truss design details, including the nodal locations and loads, member properties, and support locations, are defined in Figure 41 and Table 8.

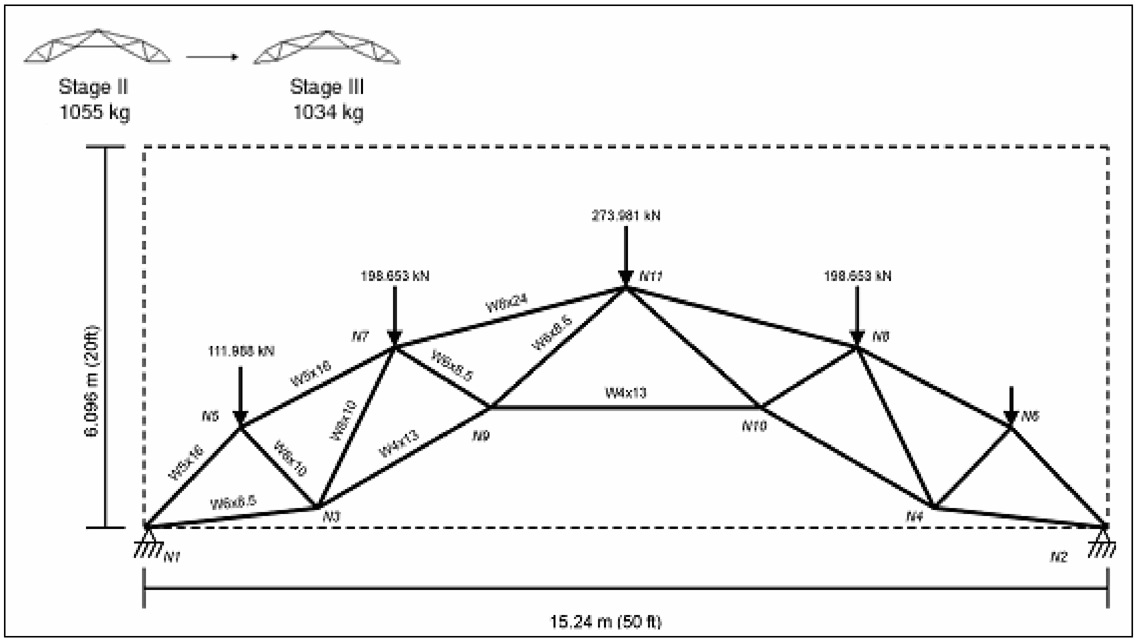

**Fig. 41. Details of a near-optimal design for the roof truss problem domain**

**Table 8. Stage III Nodal coordinates for the optimal roof truss design**

| Node | X(m) | Y(m) | Node | X(m) | Y(m) | Node | X(m) | Y(m) |
|---|---|---|---|---|---|---|---|---|
| N1 | 0 | 0 | N5 | 1.22 | 1.22 | N9 | 5.32 | 1.67 |
| N2 | 15.24 | 0 | N6 | 14.02 | 1.22 | N10 | 9.92 | 1.67 |
| N3 | 2.44 | 0.3 | N7 | 3.05 | 2.28 | N11 | 7.62 | 3.34 |
| N4 | 12.8 | 0.3 | N8 | 12.19 | 2.28 | | | |

# CHAPTER XII

# CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

## CONCLUSIONS

The primary objective of this research effort is to produce optimal or near-optimal truss structures in a user-defined search domain. The process of optimization and design is broken down into three stages. This staging helps in addressing topology, geometry and sizing optimization using different processes, while providing computationally efficiency. The staging helps to increase the performance of the overall method, as a single optimization procedure might not be suitable for addressing all of these objectives simultaneously. Along with this benefit to designers, the method developed in this research allows the user to consider a wide range of design alternatives, rather than just providing one final optimal solution. One of the strongest features of this research effort is the wide range of design alternatives being produced as the final result. In comparison to most other research studies where only a single solution is finally obtained, this research produces a range of optimal trusses. By allowing the user to input various parameters between the three stages, the final designs are being produced in consonance with the users' requirements. Along with minimizing the total weight of the overall truss, and staying within the stress, slenderness and displacement constraints, the final designs can also better match the aesthetics requirements of the designer. Optimizing, while considering the human component, is a novel feature of this work, as very few researchers have given emphasis to aesthetics in design.

The process of evolution of trusses in Stage I is pertinent for designing roof and bridge trusses in an unstructured problem domain. Using triangles, along with the applied constraints and closing mechanism always produces stable truss topologies. While working under numerous constraints, a large number of unique topologies can be produced simultaneously. The user defines the design domain within which these trusses are generated. By providing input that includes the maximum number of members and the coordinates of the node next to the support nodes, the user can influence the shape and complexity of the generated topologies. Along with the wide variety of geometries and topologies, the time taken for the production of these trusses is negligible due to this highly efficient algorithm.

Stage II uses a multi-objective genetic algorithm (MOGA) to perform sizing optimization of the members defining selected truss structures The MOGA imitates the process of natural evolution to arrive at optimal designs. A large population of designs is created. Crossover and mutation is performed out to produce new designs from existing designs in the current generation. The selection of parents to undergo reproduction is determined by their individual rank. The rank can be viewed by plotting the trade-offs that occur between the numerous objectives. The objectives considered in this research for truss systems are stresses, displacements, and total volume. Multi-objective genetic algorithms have been identified as an advanced implementation of simple GAs. The pareto-surface obtained between the various objectives helps in the better understanding of the trade-off that exists between these objectives. The solutions which lie on this pareto-surface are considered equally optimal for analysis during stage II. Hence, the MOGA provides a more efficient way to rank the population during intermediate generations. It also helps in providing independence from the fitness function formulation, which is often a significant problem with simple GAs.

The performance of two other heuristic methods was investigated for sizing optimization. Either simulated annealing or reactive taboo search can be employed in Stage II, in addition to the MOGA. Unlike GAs, these methods work with a single solution and apply iterative processes try to optimize this solution. Simulated annealing has similarities with the heating of metals to a high temperature and then reducing the temperature until the final product has an improved behavior. SA also derives its methodology from the Metropolis criteria for predicting the movement of atoms at equilibrium. Reactive taboo search, on the other hand, is an improved version of a local hill-climber that has a short term memory. By introducing a mutation strategy along with maintaining a taboo list of previously considered solutions, the RTS method provides a better search of the design space. A comparison study was conducted to evaluate the performance of all three methods with respect to their overall performance and computational efficiency. An evaluation was also made concerning their robustness to the initial parameter setting and their repeatability in obtaining the optimal solutions. It is for the first time in this research, that a benchmark roof truss problem was used to evaluate the relative strength and robustness of these three methods. The results obtained prove the superiority of using the MOGA over the other two heuristic techniques.

Local perturbations of the nodal locations are carried out in Stage III. This approach shifts the symmetric nodes vertically either one foot up or down. It is performed iteratively, while the solution remains within the allowable stress and displacement limits until no more improvement is made. In most cases, this local search strategy is able to further reduce the total weight of the truss design.

The implementation of the three stages on a four node parallel platform helped to in reduce the total computational time. Stage II, which consumes the maximum resources, is run in parallel on three computers. The parallel implementation also helps in simultaneously addressing sizing optimization and the spacing of trusses in the third dimension. Support for human-computer interactions also plays a vital role in this research by producing designs that agree with the designer's preference criteria.

A comparison of the research results with those obtained previously by other researchers on benchmark problems helps in establishing the strengths of the computational method developed. This method produces more optimal trusses while requiring less computational time.

Overall an effort has been made to make the design and optimization of truss systems more efficient using multi-objective genetic algorithms and local perturbation. An emphasis has been made to incorporate the subjective aesthetic criteria and to provide a large set of design alternatives to the user as the final product.

## FUTURE RESEARCH DIRECTIONS

This research envisages the formulation of a powerful design tool that would assist an engineer during the three dimensional design of complicated roof and bridge truss systems. Three dimensional design is addressed in this research by varying the truss spacing, which would have a direct application in warehouse systems like Walmarts, etc. Extending of this work is required in order to consider the design of dome shaped; radial and other complicated truss systems. The method developed would require the utilization of the same concepts used in Stage I, but modified to address a third dimension. The processes in Stage II and III would remain the same, except for requiring a 3-D finite element analysis implementation.

A selection aide could also be provided for selecting truss topologies at the end of Stage I. Research is currently being conducted under a project linked to this research to categorize the truss topologies produced during Stage I using a Kohenen neural network. The self-organizing maps created at the end of Stage I are able to categorize the trusses into groups of trusses that have similar features. This helps the users in selecting a group that matches their design preferences while minimizing the number of trusses that must be viewed. The selected groups of similar trusses are passed onto Stage II for size optimization.

The current commercial software programs available for structural engineering perform sizing optimization using sequential search algorithms that are highly inefficient and almost never provide a global optimal solution. The long term goal of this research effort is to provide a method that qualifies as a trusted resource to replace these sequential search algorithms. As is evident from the results obtained in this research, these methods could lead to huge savings in the overall cost of the structure while also allowing the designer to satisfy their design objectives to a much greater extent.

# REFERENCES

Agarwal, P., and Raich, A.M. (2004). “Design and optimization of steel trusses using genetic algorithms, parallel computing and computer human interactions.” paper presented at the *Advances in Structural Engineering and Mechanics*, Seoul, Korea.

AISC (1992). *Manual of Steel Construction : Allowable Stress Design.* American Institute of Steel Construction, Inc., Chicago, IL.

AISC (2001). *Manual of Steel Construction : Load and Resistance Factor Design.* American Institute of Steel Construction, Inc., Chicago, IL.

Azid, I.A., Kwan, A.S.K., and Seetharamu, K.N. (2002). “An evolutionary approach for layout optimization of a three-dimensional truss.” *Structural Multidisciplinary Optimization,* 24, 333-337.

Bennage, W.A., and Dhingra, A.K. (1995). “Optimization of truss topology using tabu search”. *International Journal for Numerical Methods in Engineering,* 38(23), 4035-4052.,

Bennett, J. A., and Botkin, M. E. (1985). “Structural shape optimization with geometric description and adaptive mesh refinement.” *AIAA J.* 23(3), 458-464.

Benyus, J. (1997). *Biomimicry: Innovation Inspired by Nature*. William Morrow Company, New York, NY.

Berke, L., and Khot, N. S. (1987). “Structural optimization using optimality criteria.” *Proc., Computer Aided Optimal Design : Structural and Mechanical System,* Soares C.A.M., Eds. ; Springer: Berlin, Germany, 235-269.

Bland, J.A., Dawson, G.P (1991). “Tabu search and design optimization”. *Computer Aided Design*, 23(3), 195-201.

Bland, J.J. (1995). “Discrete-variable optimal structural design using tabu-search.” *Structural Optimzation*, 10(2), 87-93.

Cai, W., and Cheng, G. (1998). “Simulated annealing algorithm for the topology optimization of truss.” *Huanan Ligong Daxue Xuebao/Journal of South China University of Technology (Natural Science),* 26(9), 78-84.

Cheng, F.Y., and Li, D. (1997). “Multi-objective optimization design with pareto genetic algorithms.” *Journal of Structural Engineering*, 123(9), 1252-1261.

Coello, C.A., and Christiansen, (2000). “Multi-objective optimization of trusses using genetic algorithms.” *Computers and Structures,* 75, 647-660.

Colorni, A., Dorigo, M., and Maniezzo, V. (1992), “Distributed optimization by ant colonies.” *Proc., First European Conference on Artificial Life*, Varela F., Bourgine P., Eds.; Elsevier: Paris, France, 134–142.

Corana., A., Marchesi., M., Martini, C., Ridella, S. (1987). “Minimizing multimodal functions of continuous variables with the 'simulated annealing' algorithm.” *ACM Transactions on Mathematical Software*, 13(3), 262-280.

Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*, John-Wiley & Sons Ltd., West Sussex, England.

Deb, K., and Gulati, S. (2001). “Design of truss-structures for minimum weight using genetic algorithms.” *Finite Elements in Analysis and Design,* 37, 447-465.

Dorigo, M., Di Caro, G., and Gambardella, L. (1999). “Ant algorithms for discrete optimization”. *Artificial Life,* 5(3), 137-172.

Eschenauer, H., Koski, J., and Osyczka, A. (Ed.) (1990*). Multicriteria Design Optimization*. Springer-Verlag, New York, NY.

Gage, P.J., Kroo, I.M., and Sobieski, I.P. (1995). “Variable-complexity genetic algorithm for topological design.” *AIAA J.*, 33(11), 2212-2217.

Glover, F. (1977) “Heuristics for integer programming using surrogate constraints”. *Design Science.* 8, 156-166.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison-Wesley, Reading, MA.

Grierson, D.E. (2001). “Optimal design of engineered structures using evolutionary computation.” *Tutorial, Genetic and Evolutionary Computation Conference – GECCO,.* Morgan Kaufmann, San Francisco, CA, 527-534.

Hajela, P., and Lee, E. (1995). “Genetic algorithm in truss topological optimization.” *International Journal of Solids and Structures,* 32(22), 3341-3357.

Hajela, P. and Lin, C.Y. (1992). “Genetic algorithm strategies in multicriterion optimal design.” *Structural Optimization,* 4, 99-107.

Hamza, K., Mahmoud, H., and Saitou, K. (2003). “Design optimization of N-shaped roof trusses using reactive taboo search.” *Applied Soft Computing*, 3, 221-235.

Hasancebi, O., and Erbatur F. (2002). “Layout optimization of trusses using simulated annealing.” *Advances in Engineering Software*, 33(7-10), 681-696.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, MI.

Jenkins, W.M. (1992). “Plane frame optimum design environment based on genetic algorithm.” *ASCE Journal of Structural Engineering,* 118(11), 3103-3112.

Jenkins, W.M. (1997). “On the application of natural algorithms to structural design optimization.” *Engineering Structures,* 19(4), 302-308.

Jenkins, W.M. (2002). “A decimal-coded evolutionary algorithm for constrained optimization.” *Computers and Structures*, 80, 471-480.

John, K.V., and Ramakrishnan, C.V. (1987). “Minimum weight design of trusses using improved move limit method of sequential linear programming.” *International Journal Comp. Struct.*, 27(5), 583-591.

Kennedy, J., and Eberhart, R.C. (1995). “Particle swarm optimization.” *Proc., IEEE International Conference on Neural Networks,* IEEE Service Center, Piscataway, NJ, IV, 1942–1948.

Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983). “Optimization by simulated annealing.” *Science,* 220(4598), 671-679.

Krishnamoorthy, C.S., Venkatesh, P.P., and Sudarshan, R. (2002). “Object-oriented framework for genetic algorithms with application to space truss optimization.” *Journal of Computing in Civil Engineering.* 16(1), 66-75.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). “Equations of state calculations by fast computing machines.” *J. Chem Phys.* 21, 1087.

Pareto, V., (1906). *Manuale d’Economia Politica*. Societa Editrice Libraria, Milano.

Queau, J. P. and Trompette, Ph. (1980). “Two dimensional shape optimal design by the finite element method.” *International Journal for Numerical Methods in Engineering,* 15(1l), 1603-1612.

Raich, A.M., and Agarwal, P., (2005). “Optimal design of bridge and roof truss systems using multi-objective genetic algorithms.” accepted at the *ASCE International Conference on Computing in Civil Engineering,* Cancun, Mexico.

Raich, A.M., and Ghaboussi, J. (1997). “Implicit redundant representation in genetic algorithms.” *Evolutionary Computation,* 5(3), 277-302.

Raich, A.M., and Ghaboussi, J. (2000). “Evolving structural design solutions using an implicit redundant Genetic Algorithm.” *Structural Multidisciplinary Optimization,* 20, 222-231.

Rajan, S.D. (1995). “Sizing, shape, and topology design optimization of trusses using genetic algorithm.” *Journal of Structural Engineering*, 121(10), 1480-1487.

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

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

Rao, S.S. (1984). “Multiobjective optimization in structural design with uncertain parameters and stochastic processes.” *AIAA J.*, 22(11), 1670-1678.

Rechenberg, I. (1965) *Cybernetic Solution Path of an Experimental Problem.* Royal Aircraft Establishment, Library Translation 1122, Farnborough, England.

Reddy, G., and Cagan, J. (1995). “An improved shape annealing algorithm for truss topology generation.” *Journal of Mechanical Design*, 117, 315-321.

Reynolds, J., and Azarm, S. (2002). “A mutli-objective heuristic-based hybrid genetic algorithm.” *Mechanics of Structures and Machines,* 30(4), 463-491.

Roston, G. P., and Sturges, R. H. (1996). “Using the genetic design methodology for structure configuration.” *Microcomputers in Civil Engineering,,*11, 175-183.

Ruy, W.H., Yang, Y.S., Kim, G.H., and Yeun, Y.S. (2001). “Topology design of truss structures in a multicriteria environment.” *Computer-Aided Civil and Infrastructure Engineering*, 16, 246-258.

Ryoo, J., and Hajela, P. (2004). “Handling variable lengths in GA-based structural topology optimization.” *Structural Multidisciplinary Optimization*, 26, 318-325.

Schaffer, J.D. (1985). “Multiple objective optimization with vector evaluated genetic algorithms.” *Proc., First International Conference on Genetic Algorithms,* Lawrence Erlbaum Associates, Hillsdale, NJ, 93-100.

Schmit, L. A. Jr. (1981). “Structural synthesis - Its genesis and development.” *AIAA J.*, 19(10), 1249-1263.

Shea, K., Cagan, J., and Fenves, S.J. (1997). “A shape annealing approach to optimal truss design with dynamic grouping of members.” *Transactions of the ASME Journal of Mechanical Design*, 199, 388-394.

Shrestha, S.M.. and Ghaboussi, J. (1998). "Evolution of optimal structural shapes using genetic algorithm." *Journal of Structural Engineering*, 124(8), 1331-1338.

Soh, C.K., and Yang, Y. (2000). "Genetic programming-based approach for structural optimization." *Journal of Computing in Civil Engineering*, 14(1), 31-37.

Srinvias, N., and Deb, K. (1994). "Multi-objective optimization using nondominated sorting in genetic algorithms." *Evolutionary Computation*, 2(3), 221-248.

Templeman, A.B., and Yates, D.F. (1983). "Segemental method for the discrete optimum design of structures." *Engineering Optimization*, 6(3), 145-155.

Yang, Y., and Soh, C.K. (2000). "Fuzzy logic integrated genetic programming for optimization and design." *Journal of Computing in Civil Engineering*, 14(4), 249-254.

Yeh, I.C. (1999). "Hybrid genetic algorithms for optimization of truss structures". *Computer-Aided Civil and Infrastructure Engineering*, 14, 199-206.

# VITA

BACKGROUND

| | |
|---|---|
| Name | : Pranab Agarwal |
| Address | : C – 153, East of Kailash, New Delhi – 110065, India |
| Phone | : 91-11-26835496 |
| Email | : pranab_a@yahoo.com |

EDUCATIONAL BACKGROUND

- Bachelor of Engineering
  Civil Engineering
  Delhi College of Engineering, INDIA
  Graduation : Spring 2003 ; GPA : 3.97 / 4.00
- Master of Science
  Civil Engineering (Structures)
  Texas A&M University, USA
  Graduation : Spring 2005 ; GPA : 4.00 / 4.00

RESEARCH PAPERS

1. Agarwal, P. and Raich, A.M. (2005). "Evaluating heuristic methods for the conceptual design of steel roof trusses", accepted at the *6th World Conference on Structural and Multi-disciplinary Optimization*, Rio de Janeiro, Brazil.
2. Raich, A.M. and Agarwal, P., (2005). "Optimal design of bridge and roof truss systems using multi-objective genetic algorithms." accepted at the *International Conference on Computing in Civil Engineering, ASCE,* Cancun, Mexico.
3. Agarwal, P. and Raich, A.M. (2004). "Design and optimization of steel trusses using genetic algorithms, parallel computing and computer human interactions." *Proc., Advances in Structural Engineering and Mechanics*, Seoul, Korea.
4. Pranab, Siddharth, Ankur and P.R. Bose (2003). "An analytical approach to jacketing of columns for retrofitting." *Proc., Retrofitting of Structures,* Indian Institute of Technology, Roorkee, India.
5. Pranab and Saurabh (2003). "Reconstructional survey of 3 villages: Selari, Kalyanpar, Sangramsar(District Rapar, Gujarat, India)." *Proc., Pacific Conference on Earthquake Engineering*, Christchurch, New Zealand.
6. Bose P.R., Sinvhal A., Bose A., Verma A., Pranab and Saurabh (2002). "Implications of planning and design decisions on damage during earthquakes." *Proc., International Symposium on Earthquake Engineering,* Indian Institute of Technology, Roorkee, India, 1, 561-568.