TRUSS TOPOLOGY OPTIMIZATION WITH A PRESCRIBED MAXIMUM NUMBER OF ELEMENTS

BY

MR. ALIN SHAKYA

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY (ENGINEERING AND TECHNOLOGY)

SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY

THAMMASAT UNIVERSITY

ACADEMIC YEAR 2019

TRUSS TOPOLOGY OPTIMIZATION WITH A PRESCRIBED MAXIMUM NUMBER OF ELEMENTS

BY

MR. ALIN SHAKYA

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF DOCTOR
OF PHILOSOPHY (ENGINEERING AND TECHNOLOGY)
SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2019

THAMMASAT UNIVERSITY
SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY

DISSERTATION

BY

MR. ALIN SHAKYA

ENTITLED

TRUSS TOPOLOGY OPTIMIZATION WITH A PRESCRIBED
MAXIMUM NUMBER OF ELEMENTS

was approved as partial fulfillment of the requirements for
the degree of Doctor of Philosophy (Engineering and Technology)

on June 24, 2020

Chairperson	[illegible]
	(Associate Professor Nakhorn Poovarodom, Ph.D.)
Member and Advisor	[illegible]
	(Professor Pruettha Nanakorn, D.Eng.)
Member	[illegible]
	(Professor Teerapong Senjuntichai, Ph.D.)
Member	Chainarong Athisakul
	(Associate Professor Chainarong Athisakul, Ph.D.)
Member	Thunyaseth Sethaput
	(Assistant Professor Thunyaseth Sethaput, Ph.D.)
Director	[illegible]
	(Professor Pruettha Nanakorn, D.Eng.)

Thesis Title	TRUSS TOPOLOGY OPTIMIZATION WITH A PRESCRIBED MAXIMUM NUMBER OF ELEMENTS
Author	Mr. Alin Shakya
Degree	Doctor of Philosophy (Engineering and Technology)
Faculty/University	Sirindhorn International Institute of Technology/ Thammasat University
Thesis Advisor	Professor Pruettha Nanakorn, D.Eng.
Academic Years	2019

ABSTRACT

Truss topology optimization is one of the most active research areas in the field of structural optimization. Many studies on truss topology optimization have been done. Most of these studies employ the ground structure approach in performing truss topology optimization. In the ground structure approach, a grid of nodes is created, and a dense ground structure is usually formed by joining every node of the grid. The elements in the ground structure are then used as possible positions of truss elements. Since the elements are created by joining all the nodes of the grid, the search space becomes large quickly when the number of nodes is increased. In truss topology optimization problems, there are commonly many kinematically unstable trusses in their search spaces. Oftentimes, trusses are unstable only because of some locally unstable truss elements, such as disconnected elements. In addition, many trusses in a search space contain useless zero-force elements. Many of these kinematically unstable and zero-force elements can be easily identified. In the first part of this study, a new representation for truss topology optimization is proposed in which an element-removal algorithm is used to remove such types of unwanted elements from trusses. These elements are removed during the translation of a representation code into its corresponding truss in order that the resulting truss may become a stable one. As a result, more representation codes in the search space are mapped into kinematically stable and efficient trusses, and the level of competition among representation codes is increased. This process alleviates the problem of having large search spaces and encourages faster solution convergences. In the second part of the study, a new coding scheme is proposed in which a prescribed maximum number of elements is specified for truss optimization. In this scheme, there is no need for any ground structure, and only a grid of nodes is required. The value of the maximum number of elements is set to be lower than the number of elements in a complete ground structure. As a result, the number of elements that must be removed during the optimization process to obtain the optimal truss is reduced. Another advantage of the proposed scheme is that the designer can predefine a maximum limit for the number of members to be present in the final optimal truss. Both proposed representations are used in this study with a simple multi-population particle swarm optimization algorithm (MPSO). Several example problems are solved, and it is found that both proposed representation codes significantly improve the performance of the optimization process.

Keywords: Truss topology optimization, Coding scheme, Particle swarm optimization, Unwanted elements, Ground structure, Multi-population.

ACKNOWLEDGEMENTS

I would like to express my gratitude and sincere appreciation to my advisor, Prof. Dr. Pruettha Nanakorn, for his invaluable guidance, patience, and encouragement throughout the duration of my Ph.D. study. Apart from study, I have learnt from him to work hard and always keep one’s focus on the work at hand. Sincere appreciation and gratitude are given to Assoc. Prof. Dr. Nakhorn Poovarodom from Faculty of Engineering, Thammasat University, for serving as the chairperson of the examination committee and giving me useful comments during my presentations. Deep appreciation and gratitude are also extended to Prof. Dr. Teerapong Senjuntichai from Chulalongkorn University, Assoc. Prof. Dr. Chainarong Athisakul from King Mongkut’s University of Technology Thonburi, and Asst. Prof. Dr. Thunyaseth Sethaput from Sirindhorn International Institute of Technology (SIIT) for devoting their valuable time to serve as the members of the thesis committee. Special thanks are conveyed to Prof. Dr. Ali Kaveh, School of Civil Engineering, Iran University of Science and Technology, Tehran, Iran, for serving as the external examiner of this thesis.

I would like to thank all my colleagues in Dr. Pruettha’s research team, especially Mr. Pana Suttakul, Mr. Duy Vo, Mr. Watsuwat Petprakob, and Mr. Sujan Tripathi for helping me with all the problems. Special thanks go to Mr. Kasem Theerakittayakorn for organizing wonderful trips to various parts of Thailand and making us refreshed and familiar with Thai culture and traditions, during the entire study period.

This study is sincerely dedicated to my dad, whom I lost during the period of this study. He was an inspiration for me to work hard and to do good things. The support and encouragement given to me by my family even in the challenging times are really appreciable.

Finally, I would like to express my gratefulness to SIIT for providing me the Excellent Foreign Student (EFS) Scholarship, without which it would not be possible to pursue my study here in this institute.

Mr. Alin Shakya

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 General

Optimization is the process of obtaining the best result under given constraints from a set of available alternatives. There are different approaches available to solve optimization problems. Optimization methods, in general, can be broadly classified into two types, namely gradient-based methods and heuristic methods. Gradient-based methods are traditional methods of optimization, in which the gradients of objective functions are found using some mathematical algorithms and subsequently used to locate the points on the objective functions where their values are zero. Such points are associated with the optimal values of the functions, in their immediate localities. The Newton-Raphson method, steepest descent method, conjugate gradient method, and generalized reduced gradient method are some examples of gradient-based methods. The main disadvantage of gradient-based methods stems from the fact that it is usually difficult or computationally expensive to find, for real-life problems, the gradients of objective functions. Another disadvantage of these methods is that each of these gradient-based methods usually performs its search by iteratively improving one search point. Depending on the initial guess, the obtained result can be simply a local best solution. Due to these disadvantages, gradient-based methods may not be applicable to real-life problems. Especially, gradient-based methods may not be able to handle large problems that have many local optimums effectively.

Heuristic methods are search methods that, in general, do not require the gradients of objective functions. Most heuristic methods are conceptualized from natural phenomena, such as natural evolution, animal foraging behavior, and swarm intelligence. Heuristic methods commonly employ many search points, instead of one, at the same time. As a result, the chance of being trapped in local optimal regions is reduced. In general, heuristic optimization methods can be used with continuous or discrete optimization variables and any types of constraint without difficulty. Because of these flexible characteristics of these methods, they are now becoming popular among researchers. The biggest drawback of heuristic optimization methods may be that they are in general computationally expensive since many search points must be evaluated during their optimization processes. As a result, they may not handle significantly large problems well. However, the current and future advancement of computers and parallel computing can alleviate this drawback.

A conventional structural design procedure always aims at finding an acceptable or adequate design, which satisfies the functional and other requirements. In general, there might be more than one acceptable design, and the purpose of optimization is to choose the best one of these many acceptable designs. The definite way to get the best result is to compare all alternatives, but this is mostly not possible as the number of alternatives is usually remarkably high. To avoid this comparison problem, a structural design problem can be treated as an optimization problem and subsequently solved by an optimization algorithm. The objective of a structural design optimization problem is usually taken as minimizing the material cost. The structural design constraints straightforwardly become the constraints of the optimization problem.

The optimal design of a truss is concerned with the distribution of available materials among its structural members to carry a given set of loads as efficiently as possible, subjected to mechanical and technological constraints. To obtain the optimal design of a truss, three types of optimization can be employed, i.e. size, shape, and topology optimization. Size optimization considers the cross-sectional areas of truss elements as design variables while shape optimization considers the positions of truss joints as design variables. Fundamentally, topology optimization is concerned with the number and connectivity of truss elements and joints (Choensiridamrong, Watjatrakul & Prayote, 2014; Luh & Lin, 2011; Rajeev & Krishnamoorthy, 1992). However, topology optimization usually incorporates size optimization.

Many research works on truss topology optimization have been done (Deb & Gulati, 2001; Gilbert & Tyas, 2003; Hagishita & Ohsaki, 2009; Hajela & Lee, 1995; Hasançebi & Erbatur, 2002; Kaveh & Kalatjari, 2003; Kawamura, Ohmori & Kito, 2002; Luh & Lin, 2011; Martínez, Martí & Querin, 2007; Ohsaki, Fujisawa, Katoh & Kanno, 1999; Rajan, 1995; Sokół, 2011; Wu & Tseng, 2010). Most of the works consider stabilities, stresses, and displacements as the constraints of optimization (Deb & Gulati, 2001; Guo, Cheng & Olhoff, 2005; Wu & Tseng, 2010). To define all possible topologies, the ground structure approach, in which a highly interconnected truss is generated by joining some predefined nodes, is generally used. During the optimization process, members, whose areas are smaller than some minimum value, are removed from the truss (Hajela & Lee, 1995). Since only the removal of elements is possible, to get the real optimal structure, the initial ground structure must be sufficiently dense. In general, to construct a ground structure, all predefined nodes are connected the other nodes in order not to lose any single possible configuration. Considering densest trusses naturally requires large computational resources and time. Hence, researchers are now thinking of alternatives where it is not mandatory to construct densest ground structures (Gao, Liu, Li & Qiao, 2016; Gilbert & Tyas, 2003; Hagishita & Ohsaki, 2009; Kawamura, Ohmori & Kito, 2002; Sokól, 2014). The denseness of a ground structure can be reduced by not considering all possible node connections. For example, it is possible to consider only the connections between neighboring nodes as possible truss elements (Beckers & Fleury, 1997). The definition of neighborhood can be adjusted to cover a large or small region. In real practice, it is reasonable to use a sparse ground structure that does not have a large number of nodes and does not include long connections.

When the objective of a truss topology optimization problem is to minimize the volume of the truss under only stress constraints, the optimization problem can be formulated as a linear programming problem (Bołbotowski & Sokół, 2016; Gilbert & Tyas, 2003; Sokół, 2011). Solving linear programming problems is not computationally expensive, and, as a result, significantly large problems of this type can be solved efficiently and accurately. Truss topology optimization problems that also involve stability and displacement constraints are in general nonlinear programming problems. Nonlinear programming problems may contain local optimal solutions and are typically difficult to solve. Population-based heuristic optimization methods, whose applicability is not limited to certain types of optimization problem and can handle global optimization problems well, can be excellent choices for solving nonlinear truss topology optimization problems (Hajela & Lee, 1995; Hasançebi & Erbatur, 2002; Kaveh & Kalatjari, 2003; Luh & Lin, 2011; Rajan, 1995). Population-based heuristic optimization methods employ many search points simultaneously and their chance of being trapped in local optimal regions is low.

1.2 Statement of the problem

Most research works on truss topology optimization employ ground structures and eliminate unwanted truss members from the ground structures during the optimization processes. Although using ground structures to define possible topologies is convenient, the problem with this type of scheme is that, when the number of nodes in a ground structure is increased, the search space becomes large very quickly. Once the search space becomes large, it naturally becomes difficult to find the best solution by any optimization method. In addition, large optimization problems require large computational resources and time, especially when heuristic optimization methods are used to solve them.

In a ground structure having N nodes, if all possible node connections are considered, there are a total of $n = \binom{N}{2} = N(N-1)/2$ different truss members in the ground structure. If each element has C discrete choices of sections, the number of possible trusses in the search space is equal to $C^{N(N-1)/2}$. The size of the search increases more than exponentially with N. If each element has a continuous range of possible sectional areas, the number of dimensions of the continuous search space is equal to $N(N-1)/2$, which increases quadratically with N. However, it is observed that the number of elements in an optimal truss is always considerably less than the number of elements present in the original ground structure. This implies that most of the elements in the original ground structures are removed by the optimization process to obtain the optimal truss. Thus, it should be possible to develop an efficient truss optimization scheme in which only small numbers of truss elements are considered during the optimization process.

Another difficulty arising from the use of ground structures is that, ground structures always produce search spaces that include many kinematically unstable trusses. Unstable trusses in a search space range from obvious ones, such as those that have disconnected elements, to obscure ones that can only be identified during the analysis process. In most cases, kinematically unstable trusses occupy substantial portions of search spaces. Having many unstable trusses in the search space of a truss optimization problem naturally makes the problem difficult to solve. It is therefore desirable to reduce unstable trusses in search spaces. Some types of unstable element, such as disconnected elements, can be easily identified. If these elements are removed during the translation of a representation code into its corresponding truss, the resulting truss may become a stable one. This results in more representation codes in the search space that are mapped into kinematically stable trusses. This subsequently increases the competition among representation codes. In addition, in many cases, elements that have zero forces can be easily identified. These elements are not useful and can also be removed in the same manner as that for unstable elements, mentioned earlier. Removing these unwanted elements should alleviate the problem of having large search spaces using ground structures and encourages faster solution convergences.

1.3 Objectives of the study

To increase the efficiency of truss topology optimization using heuristic optimization methods, in this study, two new coding schemes are proposed. The objectives of this study are as follows:

1) To develop a coding scheme for removing unstable and zero-force elements that can be easily identified from trusses during the decoding process.
2) To develop a coding scheme for truss topology optimization with a prescribed maximum number of truss elements.

1.4 Scope of the study

The scope of this study is as follows:

1) Linear elasticity is assumed.
2) The objective of each truss optimization problem is to minimize the weight of the truss.
3) The constraints include stress and displacement constraints. Buckling constraints are not considered.
4) The weight of each truss is not considered in the calculation of stresses and displacements.
5) Multi-population particle swarm optimization (PSO) is used as the optimization algorithm.

1.5 Organization of the thesis

There are five chapters in this thesis. In the first chapter, the introduction to the study is given. The second chapter presents the literature review of this research work. The third chapter gives the major theoretical background required for this research. The fourth chapter gives the details of the proposed coding schemes. The fifth chapter shows the results of this effort. Finally, the conclusion of the study is presented.

CHAPTER 2
REVIEW OF LITERATURE

Truss topology optimization is one of the popular areas of research in the field of structural optimization, and many studies have been done in this area by using heuristic optimization methods (Deb & Gulati, 2001; Ghoddosian, Riyahi Vezvari, Sheikhi Azqandi & Karimi, 2018; Hajela & Lee, 1995; Nanakorn & Meesomklin, 2001; Rajan, 1995; Rajeev & Krishnamoorthy, 1992; Tsiptsis, Liimatainen, Kotnik & Niiranen, 2019). This is probably because solutions of truss topology optimization can frequently be used in real practice. In most of these research works, a ground structure is formed, and the constraints considered are stresses and displacements. When the ground structure approach is employed, a uniform grid, in which all nodes are linked, is usually used to form a ground structure. Some research works have also been done using alternative approaches to form ground structures (Gao, Liu, Li & Qiao, 2016; Ghoddosian, Riyahi Vezvari, Sheikhi Azqandi & Karimi, 2018; Hagishita & Ohsaki, 2009; McKeown, 1998; Smith, 1994). For example, in the work by Gao, Liu, Li and Qiao (2016), the locations of nodes in a ground structure are determined from the values of principal stresses that are obtained from continuum analysis. Kirsch (1996) proposed reduction and expansion processes for topology optimization of trusses. In the reduction process, members and joints are eliminated from an initial structural topology. In contrast, the expansion process adds members and joints to an initial structural topology. An initial structural topology in the expansion process is formed with a limited number of members and joints. During the expansion process, modified structures are introduced successively by adding members and joints. In the work by Hagishita and Ohsaki (2009), a growing ground structure method is used. In this method, the initial ground structure does not connect all the nodes with each other. Rather, elements are added iteratively during successive generations by using various strategies. Three types of problem, i.e. size, shape, and topology optimization problems are solved using this method. In the work by Ghoddosian, Riyahi Vezvari, Sheikhi Azqandi and Karimi (2018), another growing ground structure method, named the minimum growing ground structure method (MGGSM), is used for optimization of discrete truss structures. In MGGSM, a minimum stable structure is first constructed instead of constructing a complete ground structure, in which all nodes are linked. In this step, nodes that have forces are linked with nodes that represent supports to generate this minimum stable structure. In the second step, elements are added to the minimum stable structure until the structure is feasible with respect to stress and displacement constraints. In the third step, a heuristic optimization method is used to perform topology optimization on the structure obtained from the second step. The results by MGGSM are compared with those by other methods and the efficiency of MGGSM is shown. The main advantage of MGGSM is the reduction in computational cost specially for structural analysis. In the work by Kaveh and Kalatjari (2003), topology optimization of trusses is done using a genetic algorithm with the help of the force method and the graph theory. A graph with all pairs of truss nodes being connected by single members is named as a complete graph. A graph with no multiple members and loops is known as a simple graph. A graph with no member is called a null graph. After studying many basic graphs, a star graph is found to be one of the most suitable forms for topology optimization. In a star graph, every node is connected to the neighboring nodes only.

Truss optimization problems mostly have large search spaces and can have either continuous or discrete design variables, or both. In addition, they tend to have many local optimal solutions. As a result, heuristic optimization methods are more suitable for these problems than traditional gradient-based optimization methods. Many heuristic optimization algorithms have been used to solve truss optimization problems. Some popular ones include genetic algorithms (GAs), particle swarm optimization (PSO), and firefly algorithms (FAs).

GAs have been extensively used for solving truss optimization problems (Cui, An & Huang, 2018; Deb & Gulati, 2001; Han & Wang, 2019; Rajan, 1995; Rajeev & Krishnamoorthy, 1992). In the work by Rajeev and Krishnamoorthy (1992), a GA is first used for truss optimization. In their work, the objective of a truss optimization problem is to minimize the weight of a truss under stress and displacement constraints. The objective function is modified to incorporate constraints. The fitness of an individual is obtained by subtracting the modified objective function from a large constant. After the work by Rajeev and Krishnamoorthy (1992), GAs have been continuously used to solve size, shape, and topology optimization problems (Cui, An & Huang, 2018; Deb & Gulati, 2001; Neeraja, Kamireddy, Kumar & Reddy, 2017; Rajan, 1995; Serpik, Alekseytsev & Balabin, 2017). In size optimization problems, design variables are the cross-sectional areas of elements (Assimi, Jamali & Nariman-zadeh, 2017; Rajan, 1995). The objective of a size optimization problem is simply to find the best combination of cross-sectional areas that corresponds to the optimal truss, without removing any members. In shape optimization, nodal coordinates are considered as design variables, and the optimized shape of a truss is the one that has the minimum weight and does not violate its constraints (Deb & Gulati, 2001; Ede et al., 2018; Rajan, 1995). In topology optimization, truss topology and element cross-sectional areas that result in optimized trusses are to be determined (Arnaot, 2019; Assimi, Jamali & Nariman-zadeh, 2017; Deb & Gulati, 2001; Ede et al., 2018; Neeraja, Kamireddy, Kumar & Reddy, 2017). When the ground structure approach is used, the design variable representing the cross-sectional area of an element is allowed to become zero, or even negative. The presence or absence of an element in a ground structure is determined by comparing the design variable, representing the cross-sectional area of the element, with a user-defined small critical cross-sectional area. If the value of the design variable is smaller than the critical area, the element is considered absent. In the work by Han and Wang (2019), a GA is used to optimize bridge structures, where topology optimization of top lateral bracing configurations is done. In the study by Cui, An and Huang (2018), a GA is used for truss topology optimization with local buckling constraints. Their algorithm also restricts intersections and overlapping of members. Some other studies have also been done to eliminate overlapping members (Hagishita & Ohsaki, 2009; Kawamura, Ohmori & Kito, 2002).

In the course of time, many other modifications have been done in truss optimization by GAs (Kawamura, Ohmori & Kito, 2002; Nanakorn & Meesomklin, 2001; Richardson, Adriaenssens, Bouillard & Coelho, 2012; Tsiptsis, Liimatainen, Kotnik & Niiranen, 2019). For example, in the work by Nanakorn and Meesomklin (2001), an adaptive penalty scheme for structural design optimization by GAs is proposed. In GAs, penalty functions are generally used to consider constraints. To penalize infeasible structures, their fitness values are simply reduced. The relative importance between objective and constraint functions are commonly adjusted via user-defined coefficients that are usually problem-dependent and have no physical meanings. The adaptive penalty scheme by Nanakorn and Meesomklin (2001) employs a coefficient that has a clear physical meaning and allows the penalty to automatically adjust itself during the evolution. In the study by Kawamura, Ohmori and Kito (2002), a modified GA for eliminating overlapping elements in optimal trusses is proposed. Combinations of connected triangles are used instead of single line elements to generate trusses. As a result, unstable structures are avoided automatically by using triangles. In this modified method, elements are not generated all at once as in the ground structure scheme. Rather, triangular elements are generated progressively until all base nodes are selected. In their study, a new modified coding system is developed to represent triangular elements. In the study by Richardson, Adriaenssens, Bouillard and Coelho (2012), a kinematic stability repair approach for GAs is proposed. In their study, kinematically unstable trusses are not penalized. Instead, their chromosomes are modified to add or remove elements to finally obtain stable trusses. This implies that some representation codes are not considered since, when these codes appear, they are changed into different ones. It is shown that their method can significantly increase the rate of convergence of the algorithm. In the study by Nimtawat and Nanakorn (2009) and Nimtawat and Nanakorn (2010), the concept of removing unwanted elements in topology or layout optimization is introduced using GAs. A grid is superimposed onto a floor plan and serves as a ground structure for beam-slab layouts. Beam patterns created from a grid always include patterns that are not valid. Beam patterns are not valid if they contain some invalid beams, such as disconnected beams. In these works by Nimtawat and Nanakorn (2009) and Nimtawat and Nanakorn (2010), invalid beams are removed from their patterns during the decoding process in order to obtain valid beam-slab layouts. As a result, the representation of beam-slab layouts includes only valid beam-slab layouts. In short, the mapping between the representation codes and the corresponding layouts is specially tailored to give only valid beam-slab layouts.

PSO, developed by Kennedy and Eberhart (1995), is another popular heuristic optimization method that is frequently used for truss optimization. This method is generally used for continuous variables with continuous nonlinear objective and constraint functions. PSO mimics bird flocking, fish schooling, and swarming behavior of animals. The concept of PSO is very simple and it can be implemented in a few lines of codes. Truss topology optimization using PSO has also been developed by many researchers (Ghoddosian, Riyahi Vezvari, Sheikhi Azqandi & Karimi, 2018; Luh & Lin, 2011; Nanakorn, Petprakob & Naga, 2014; Tsiptsis, Liimatainen, Kotnik & Niiranen, 2019). In the study by Luh and Lin (2011), two PSO algorithms are used for truss topology optimization. First, a binary particle swarm optimization (BPSO) algorithm is used to find the best topology. After that, an algorithm based on so-called attractive and repulsive particle swarm optimization (ARPSO) is used to find the optimal size and shape variables for the obtained topology. In the study by Tsiptsis, Liimatainen, Kotnik and Niiranen (2019), PSO is used with isogeometric tools for optimization of truss and frame towers. The objective of the study is to explore structural optimization by a PSO-based optimizer having non-uniform rational B-spline (NURBS) integrated in both the structural analysis solver and PSO. This integration increases the accuracy of results. It also reduces computational effort and makes parametric studies easy to perform. Multi-population particle swarm optimization (MPSO) is shown to be more effective than conventional PSO in the work by Nanakorn, Petprakob and Naga (2014). An object-oriented programming (OOP) language is used to implement an MPSO algorithm for truss topology optimization. In their paper, it is shown that the OOP paradigm can be used suitably for implementing PSO algorithms. In OOP, the communications and actions between objects are considered. In PSO, there are particles, which can be treated as objects. The communications and actions between them can be considered easily under the OOP paradigm. In MPSO, particles are swapped among different populations every some iterations. Since many populations are used, MPSO reduces the probability of the algorithm to be trapped in local optimal regions. Using many populations of particles can be computationally expensive. Nevertheless, parallel computing techniques can be used to make the computation more effective and efficient.

FAs, first developed by Yang (2009), are among those heuristic optimization algorithms that are considered powerful. FAs mimic the flashing behavior of fireflies. Light flashing of fireflies can be formulated in such a way that it is associated with the objective functions of optimization problems. There are three basic rules in FAs. The first rule is that all fireflies are unisex. Fireflies are attracted to other fireflies regardless of their genders. The second rule is that attractiveness of a firefly increases with its brightness. Fireflies that are less bright will move towards brighter fireflies. The third rule is that the brightness of a firefly increases with its merit, which is determined from the values of its objective and constraint functions. In the work by Miguel, Lopez and Miguel (2013), FAs are used to perform size, shape, and topology optimization of truss structures. The examples considered in their work emphasize the capabilities of FAs in simultaneous optimization of size, shape, and topology of trusses in a single-stage procedure. In the problems considered, nodal coordinates are represented by continuous variables and cross-sectional areas are taken from a set of discrete variables. As a result, each problem has mixed types of variable. FAs are found to be effective in solving this type of problems. In the study by Baghlani, Makiabadi and Rahnema (2013), an accelerated firefly algorithm (AFA) for size optimization of truss structures is developed. In AFA, randomness in the motion of fireflies is reduced to improve the stability and performance of the standard FA. In the study by Wu, Li, Hu and Borgart (2017), an improved FA (IFA) is developed for size and topology optimization of trusses with discrete design variables. In IFA, the positions of initial fireflies and the position update formula are modified from the standard ones. Two examples are solved to verify the feasibility and efficiency of the algorithm.

Many other heuristic optimization algorithms have been used for solving truss optimization problems (Li & Zhen, 2019; Luh & Lin, 2008; Sönmez, 2011). For example, in the work by Sönmez (2011), an artificial bee colony (ABC) algorithm for optimization of trusses is developed, and, in the work by Li and Zhen (2019), a bat algorithm (BA) is used to optimize trusses. Among all of these algorithms, the most popular methods are probably GAs and PSO (Peydro Rasero, Sellés Cantó, Martínez Sanz, Plá Ferrando & Sánchez Caballero, 2012). Both methods have been used to solve truss topology optimization problems, in which the ground structure approach is employed. The ground structure approach is used even for non-orthogonal unstructured and concave domains (Zegard & Paulino, 2014). The ground structure approach has been found to give good results for small problems. However, for large problems, use of ground structures can be ineffective. The main drawback of the ground structure approach is that truss elements to be considered in this approach are generally and intrinsically dense. This simply means that search spaces can be exceptionally large for large problems. That is why this approach is suitable only for small problems.

CHAPTER 3

THEORETICAL BACKGROUND

3.1 Particle swarm optimization (PSO)

This optimization method, developed by Kennedy and Eberhart (1995), is modified and used in this study for truss topology optimization. PSO mimics the swarming behavior of animals, such as birds and fishes. Birds in a flock know the positions of other birds in the flock. As a result, anything interesting that any bird finds is transmitted to the entire swarm. All the members become acquainted with and, consequently, benefited from the transmitted information. Effectively, there is an information pool that is available to every bird. This intelligence helps every bird find food for itself. A population in PSO is called a swarm, and each individual in the population is called a particle. Each particle represents a position in the search space, and it can be thought of as a position of a bird in its flock. The optimal solution can be thought of as the position of food that the flock is trying to find. In each iteration, a particle moves by considering its own experience and the experience of the whole population in order to improve itself.

Consider an optimization problem in a D-dimensional search space. In the simple PSO algorithm, the movement of particles is governed by the following expressions, i.e.

$$V_{Id}(t+1) = WV_{Id}(t) + \phi_1 r_{1Id}[pB_{Id}(t) - X_{Id}(t)] + \phi_2 r_{2Id}[gB_d(t) - X_{Id}(t)]; \quad (3.1)$$

$$X_{Id}(t+1) = X_{Id}(t) + V_{Id}(t+1). \quad (3.2)$$

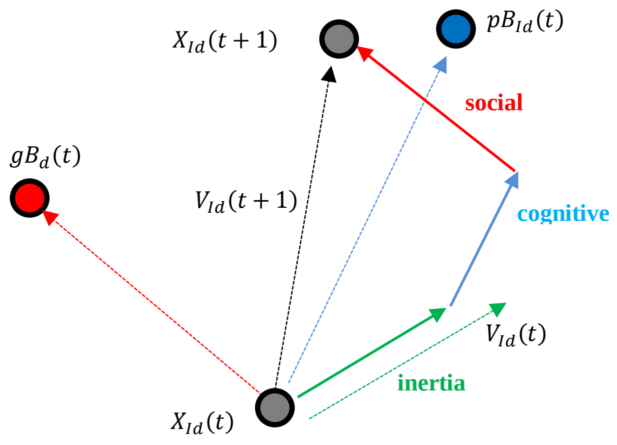


Figure 3.1 Particle movement in PSO.

Here, $V_{Id}(t)$ and $X_{Id}(t)$ represent, respectively, the velocity and position of particle I in dimension d at iteration t. In addition, W, ϕ_1, and ϕ_2 are user-defined positive constants while $r_{1Id} = rnd[0,1]$ and $r_{2Id} = rnd[0,1]$ are uniformly distributed random numbers on [0,1]. In Equation (3.1), $pB_{Id}(t)$ denotes the position in dimension d of the best solution that particle I has ever experienced, and this solution is called the particle best of particle I. In addition, $gB_d(t)$ denotes the position in dimension d of the best solution that the whole population has ever experienced, and this solution is called the global best.

The velocity V_{Id} is allowed to vary within a closed interval $[-V_{dmx}, V_{dmx}]$, where V_{dmx} is a user-defined positive constant. If the velocity $V_{Id}(t+1)$ becomes larger than V_{dmx}, it is set to V_{dmx}. If it becomes smaller than $-V_{dmx}$, it is set to $-V_{dmx}$. In addition, the position X_{Id} is varied within a user-defined closed interval $[X_{dmn}, X_{dmx}]$. If the position $X_{Id}(t+1)$ is larger than X_{dmx}, it is set to X_{dmx}. If it is smaller than X_{dmn}, it is set to X_{dmn}. In both of these cases, the velocity $V_{Id}(t+1)$ is also set to its negative.

To begin the algorithm at $t = 1$, each $V_{Id}(1)$ is randomly selected from $[-V_{dmx}, V_{dmx}]$ while each $X_{Id}(1)$ is randomly selected from $[X_{dmn}, X_{dmx}]$, i.e.

$$V_{Id}(1) = rnd[-V_{dmx}, V_{dmx}]; \tag{3.3}$$

$$X_{Id}(1) = rnd[X_{dmn}, X_{dmx}]. \tag{3.4}$$

As shown in Equation (3.1) and Figure 3.1, the new position of a particle is directed by three terms, namely the inertia, the particle best, and the global best. Over each iteration, a particle flies towards its particle best and global best positions. Since all particles always move and change their positions, the particle and global bests either become better or remain the same. The objective of the algorithm is that, at the end, all particles converge to the global optimal solution of the considered problem. The inertia weight W controls the influence of the previous velocity of a particle. A larger value of W has the greater global search ability and enhances exploration, whereas a smaller value of W has a greater local search ability and enhances exploitation. It is found that the value of W from 0.9 to 0.4 provides good results (Bansal et al., 2011). Several researchers have offered different formulas to dynamically adjust the value of W

(Bansal et al., 2011; Gao, An & Liu, 2008; Malik, Rahman, Mohd Hashim & Ngah, 2007; Xin, Chen & Hai, 2009). In addition, many researchers have proposed different modifications to the standard PSO algorithm, such as adaptive PSO (Hossen, Rabbi & Rahman, 2009), self-organizing PSO (Ratnaweera, Halgamuge & Watson, 2004), and emotional PSO (Ge & Rubo, 2005).

PSO technique is very simple when compared to GAs or FAs, and it is considered to be one of the most popular techniques of optimization (Peydro Rasero, Sellés Cantó, Martínez Sanz, Plá Ferrando & Sánchez Caballero, 2012). Particles in PSO have memories and these memories are used during search. GAs do not utilize any memory during search. Although PSO was initially developed to optimize functions with continuous design variables, with some modifications, it can be used with discrete variables (Kennedy & Eberhart, 1997; Li, Huang & Liu, 2009).

3.2 Multi-population PSO

Evolutionary and swarm optimization algorithms consider many search points at the same time. Using many search points, instead of one, reduces the chance that the search is trapped in local optimal regions. To reduce the chance of being trapped even further, multiple populations of search points can be used. Certainly, using more populations increases the computational time. However, this disadvantage can be alleviated by parallel computing. The performance of PSO is found to be improved when multiple populations are used by many researchers (Jiang, Hu, Huang & Wu, 2007; Nanakorn, Petprakob & Naga, 2014; Niu, Zhu, He & Shen, 2008). In the study by Jiang, Hu, Huang and Wu (2007), a population is partitioned into several sub-swarms and information sharing is done among these sub-swarms. In the study by Niu, Zhu, He and Shen (2008), a population, consisting of one master swarm and several slave swarms, are considered. The particles in the master swarm enhance themselves by using their own knowledge and also the knowledge of the particles in the slave swarms. In multi-population PSO proposed by Nanakorn, Petprakob and Naga (2014), two different populations are used. During the swam evolution process, some particles from the two populations are swapped with a probability of p_s at a certain number of interval of iterations.

In this study, the simple two-population PSO algorithm by Nanakorn, Petprakob and Naga (2014) is employed. In the algorithm, the two populations can have either the same or different numbers of particles. Each population employs the simple PSO algorithm. They progress independently, except when their particles are swapped with a probability of p_s at every K iterations, where p_s and K are user-defined parameters. This is done simply by pairing individuals from the two populations and setting each pair to have the probability of being swapped equal to p_s. The particles that are swapped remember their own particle bests, and these pieces of information are used in their respective new populations. Figure 3.2 shows the flowchart of the two-population PSO algorithm.

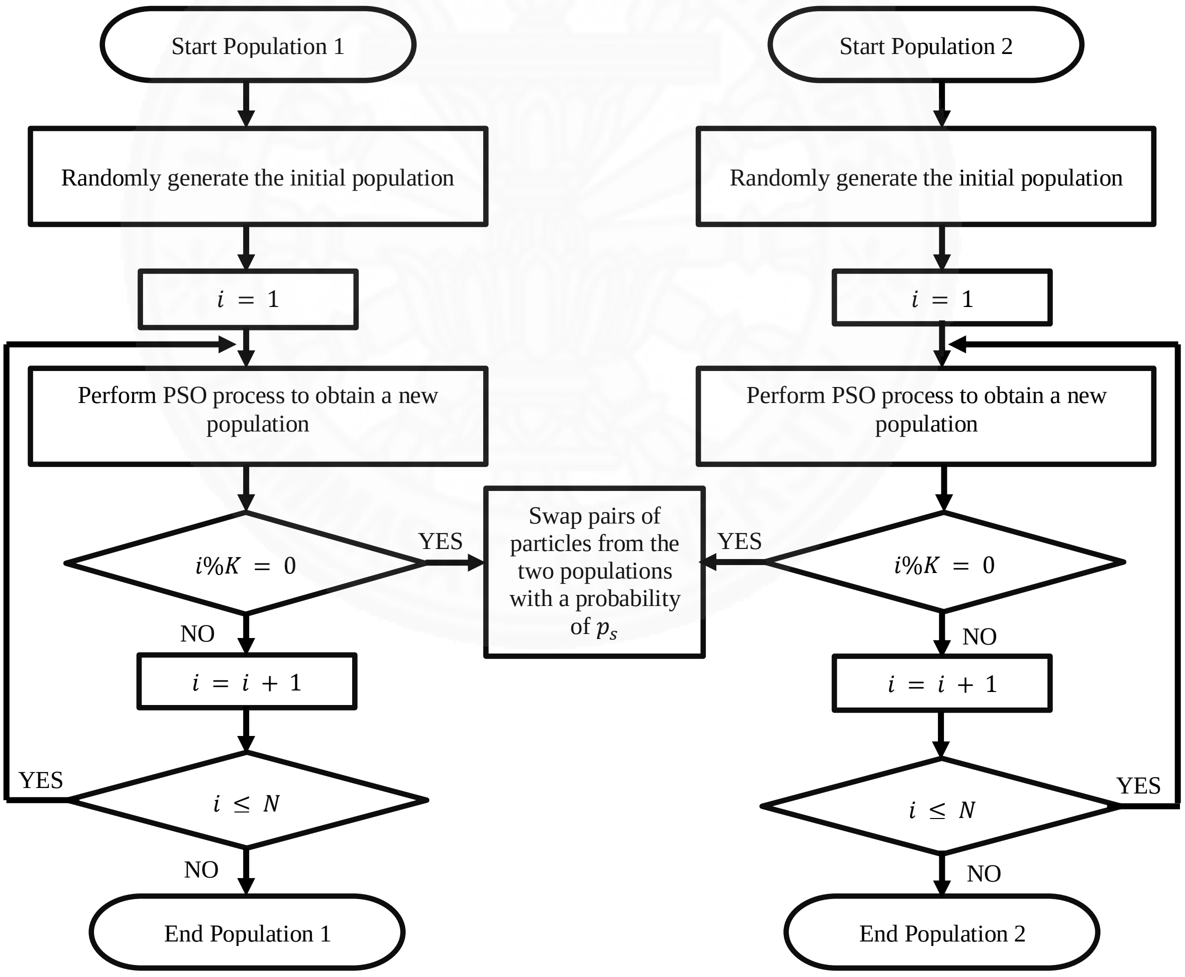

Start Population 1
Start Population 2
Randomly generate the initial population
Randomly generate the initial population
$i = 1$
$i = 1$
Perform PSO process to obtain a new population
Perform PSO process to obtain a new population
Swap pairs of particles from the two populations with a probability of p_s
YES
$i\%K = 0$
YES
$i\%K = 0$
NO
NO
$i = i + 1$
$i = i + 1$
YES
$i \leq N$
YES
$i \leq N$
NO
NO
End Population 1
End Population 2

Figure 3.2 Two-population PSO algorithm (Nanakorn, Petprakob & Naga, 2014).

3.3 Truss topology optimization

As aforementioned, the objective of each truss optimization problem considered in this study is to minimize the weight of the truss under stress and displacement constraints. The formal definition of this class of problems is given here. Consider a truss topology optimization problem in which boundary conditions and applied forces are specified. A ground structure can be created to define all possible topologies that are available for topology optimization. Let M be the number of all possible elements in the ground structure. Let S_i be the set of all available sections for element i of M elements. The set S_i can be a discrete set or a continuous set. However, S_i must include a zero-area section that in fact represents the absence of element i. Let $\mathbf{\Psi} = S_1 \times S_2 \times \ldots \times S_M$ be the set of all possible combinations of sections for the whole truss. Let $s_i \in S_i$ and $\boldsymbol{s} = (s_1, s_2, \ldots, s_M) \in \mathbf{\Psi}$. In addition, let $A_i = \{\text{area of } s_i\}$ and $\mathbf{\Phi} = A_1 \times A_2 \times \ldots \times A_M$ be the set of all possible combinations of sectional areas for the whole truss. Moreover, let $a_i \in A_i$, which means that $\boldsymbol{\alpha} = (a_1, a_2, \ldots, a_M) \in \mathbf{\Phi}$.

Designate nodes in the ground structure where boundary conditions or applied forces are specified as “conditioned nodes.” Designate nodes that are connected to no element as “disconnected nodes.” In addition, designate disconnected nodes that are not conditioned nodes as “unused nodes.” A truss that results from a combination of sections $\boldsymbol{s} \in \mathbf{\Psi}$ as a truss in which the section of element i is s_i and all unused nodes are removed. From this definition, if there is a disconnected conditioned node that is free to move in the resulting truss, the node will render the truss unstable. On the contrary, disconnected conditioned nodes that are not free to move simply represent hinge supports that are not used.

The stiffness matrix equation of a truss can be written as

$$\mathbf{K}(\boldsymbol{s})\mathbf{U} = \begin{bmatrix} \mathbf{K}_{11}(\boldsymbol{s}) & \mathbf{K}_{12}(\boldsymbol{s}) \\ \mathbf{K}_{21}(\boldsymbol{s}) & \mathbf{K}_{22}(\boldsymbol{s}) \end{bmatrix} \begin{Bmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{Bmatrix} = \mathbf{F} \tag{3.5}$$

where $\mathbf{K}$ is the stiffness matrix while $\mathbf{U}$ and $\mathbf{F}$ are, respectively, the displacement and force vectors. In addition, $\mathbf{U}_1$ and $\mathbf{F}_2$ denote, respectively, the unknown displacement and force vectors while $\mathbf{U}_2$ and $\mathbf{F}_1$ denote, respectively, the prescribed displacement and force vectors.

A truss topology optimization problem, considered in this study, is written as follows:

$$\underset{s\in\boldsymbol{\Psi}}{\text{minimize}}\ W = \sum_{i=1}^{M} \rho_i L_i a_i \tag{3.6}$$

Subjected to

$\mathbf{K}_{11}(\boldsymbol{s}) \succ 0$; i.e. $\mathbf{K}_{11}(\boldsymbol{s})$ is positive definite

$$\mathbf{K}(\boldsymbol{s})\mathbf{U} = \mathbf{F} \tag{3.7}$$

$\sigma_i \le \sigma_{a,i}$ $\quad i = 1,2,\dots,m$

$\delta_i \le \delta_{a,i}$ $\quad i = 1,2,\dots,ndof$

where

m:	number of elements in the truss
$ndof$:	number of displacement degrees of freedom of the truss
ρ_i:	weight density of element i of M possible elements
L_i:	length of element i of M possible elements
W:	weight of the whole truss
σ_i:	stress of element i of m elements of the truss
$\sigma_{a,i}$:	allowable stress for element i
δ_i:	displacement degree of freedom i of $ndof$ displacement degrees of freedom of the truss
$\delta_{a,i}$:	allowable displacement degree of freedom for displacement degree of freedom i.

The positive definiteness of $\mathbf{K}_{11}(\boldsymbol{s})$ means that the optimal truss must be kinematically stable. In the stress and displacement constraints, the magnitudes of stresses and displacements, which are positive, are used. In addition, constraints that are automatically satisfied by appropriate choices of input data, such as the ranges of sectional areas, are not included as constraints in the above optimization problem.

CHAPTER 4
METHODOLOGY

4.1 Truss topology optimization by two-population PSO

In this study, the optimization problem in Equations (3.6) and (3.7) is written for PSO as

$$\underset{\boldsymbol{\xi}\in\boldsymbol{\Gamma}}{\text{minimize}}\ W = \sum_{i=1}^{M} \rho_i L_i a_i \tag{4.1}$$

Subjected to

$Er = 0$.

Here, Er is the total degree of constraint violation defined as

$$Er = \begin{cases} \sum_{i=1}^{m} \frac{\max(0, \sigma_i - \sigma_{a,i})}{\sigma_{a,i}} + \sum_{i=1}^{ndof} \frac{\max(0, \delta_i - \delta_{a,i})}{\delta_{a,i}} & \text{if } \mathbf{K}_{11} \succ 0; \\ 10^{10} & \text{otherwise.} \end{cases} \tag{4.2}$$

If a truss is not feasible, its Er is greater than zero. If a truss is kinematically unstable and its stiffness matrix is not positive definite, Er is given a large value of 10^{10}. This strategy of giving a large value of Er to kinematically unstable structures has been employed successfully by many researchers (Deb & Gulati, 2001; Wu & Tseng, 2010). If a truss is kinematically stable, its Er is computed from the stress and displacement constraint violation. It can be seen that, if a truss is kinematically stable and does not violate the stress and displacement constraints, its Er is zero.

Given two solutions A and B, PSO needs only to know whether A is better than B or vice versa. PSO does not need to know how much exactly one solution is better than the other one. Consider two trusses, $Truss_I$ and $Truss_J$. Their weights are W_I and W_J and their total degrees of constraint violation are Er_I and Er_J. In this study, $Truss_I$ is better than $Truss_J$ when

1. $Er_I < Er_J$, or
2. $Er_I = Er_J$ and $W_I < W_J$.

In Equation (4.2), a large positive value of Er, used for kinematically unstable trusses, makes certain that these kinematically unstable trusses are always considered worse than kinematically stable trusses that violate the stress or displacement constraint.

In this study, the stability of each truss is first checked by using a variant of Grubler's equations (Deb & Gulati, 2001). The variant is given as

$$ndof_{rigid} = ndim \times n - m - ndofp. \tag{4.3}$$

Here, $ndof_{rigid}$ is the number of rigid-body degrees of freedom of the truss. In addition, $ndim$ is the number of the spatial dimensions of the problem. Moreover, n and m are, respectively, the number of nodes and the number of elements of the truss while $ndofp$ is the number of prescribed displacement degrees of freedom. If $ndof_{rigid}$ is found to be greater than zero, the truss is a mechanism and is not stable. Its stiffness matrix is not positive definite, and the truss is infeasible.

Trusses whose values of $ndof_{rigid}$ in Equation (4.3) are not positive can still be unstable trusses. The stiffness matrices of all trusses with non-positive values of $ndof_{rigid}$ are constructed by the finite element method (FEM). The positive definiteness of the obtained stiffness matrix of each truss is then checked during the matrix equation solving process within the finite element analysis process. In this study, the Cholesky decomposition is used in the matrix equation solving process. The Cholesky decomposition can be used only with symmetric positive definite matrices and its algorithm allows matrices that are not positive definite to be detected. If the stiffness matrix of a truss is found not to be positive definite by the Cholesky decomposition algorithm, the truss is infeasible, and the finite element analysis process is stopped. Otherwise, the truss is completely analyzed by FEM for displacements and stresses. Since the displacements and stresses are obtained from finite element analysis, the stiffness matrix equation, which is one of the constraints, is automatically satisfied. Similar to other numerical algorithms, the accuracy of the Cholesky decomposition algorithm in detecting non-positive definiteness depends on the machine precision used. It is possible that, due to numerical errors, some non-positive definite stiffness matrices are successfully decomposed by the Cholesky decomposition algorithm and their non-positive definiteness cannot be detected. However, if these happen, the obtained displacements will generally be very large. As a result, these trusses will not satisfy the displacement constraint.

4.2 Element-removal algorithm

In this study, the idea of removing invalid or unwanted elements from topologies to create better or acceptable topologies (Nimtawat & Nanakorn, 2009; Nimtawat & Nanakorn, 2010) is used to create a representation of trusses for truss topology optimization. To develop the representation, unwanted truss elements are first defined. These unwanted elements include kinematically unstable elements and useless zero-force elements. After that, an algorithm to remove unwanted elements from trusses is created, based on the definition of unwanted elements. The algorithm is used in the translation of representation codes into corresponding trusses. During the translation of a representation code into its corresponding truss, the unwanted elements are removed from the original topology to obtain the final topology. Since the removal of unwanted elements is included in the coding scheme, this results in more representation codes in the search space that are mapped into kinematically stable and efficient trusses. This increases the competition among representation codes. In addition, it allows the comparison between some representation codes to be done more meaningfully. For instance, it is meaningless to compare two representation codes that are both translated into kinematically unstable trusses. However, if the two codes are remapped by the element-removal algorithm into two kinematically stable trusses, the codes can be compared meaningfully. Removing unwanted elements alleviates the problem of having large search spaces using ground structures and encourages faster solution convergences.

In this section, the development of a representation of trusses with an element-removal algorithm is shown. First, unwanted truss elements that are to be removed in the decoding process are defined. After that, an algorithm to remove these unwanted elements is introduced. The formal description of the proposed representation is also presented. Small translation examples, showing how the proposed representation works, are also shown.

4.2.1 Unwanted elements

Consider a joint of a truss that is not a support. Unwanted elements are defined as follows:

For 2D trusses:

(1) If there is only one element connected to the joint, the element is kinematically unstable and is considered as an unwanted element.

(2) If there are two elements connected to the joint and the joint has no external applied force, the two elements are either kinematically unstable or zero-force members. If the two elements are in the same alignment, they are kinematically unstable. If they are not in the same alignment, they are zero-force members. Thus, they are considered as unwanted elements.

For 3D trusses:

(1) If there are less than three elements connected to the joint, the elements are kinematically unstable and are considered as unwanted elements.

(2) If there are three elements connected to the joint and the joint has no external applied force, the three elements are either kinematically unstable or zero-force members. If the three elements are in the same plane, they are kinematically unstable. If they are not in the same plane, they are zero-force members. Thus, they are considered as unwanted elements.

Figure 4.1 shows examples of unwanted elements. There are certainly other patterns of elements that are kinematically unstable or have no force. The above definition does not include all of them. Rather, the definition includes only patterns that can be easily identified and elements that are clearly known, without much detailed checking, to be useless. This is in order to not create too much computational effort during the decoding process.

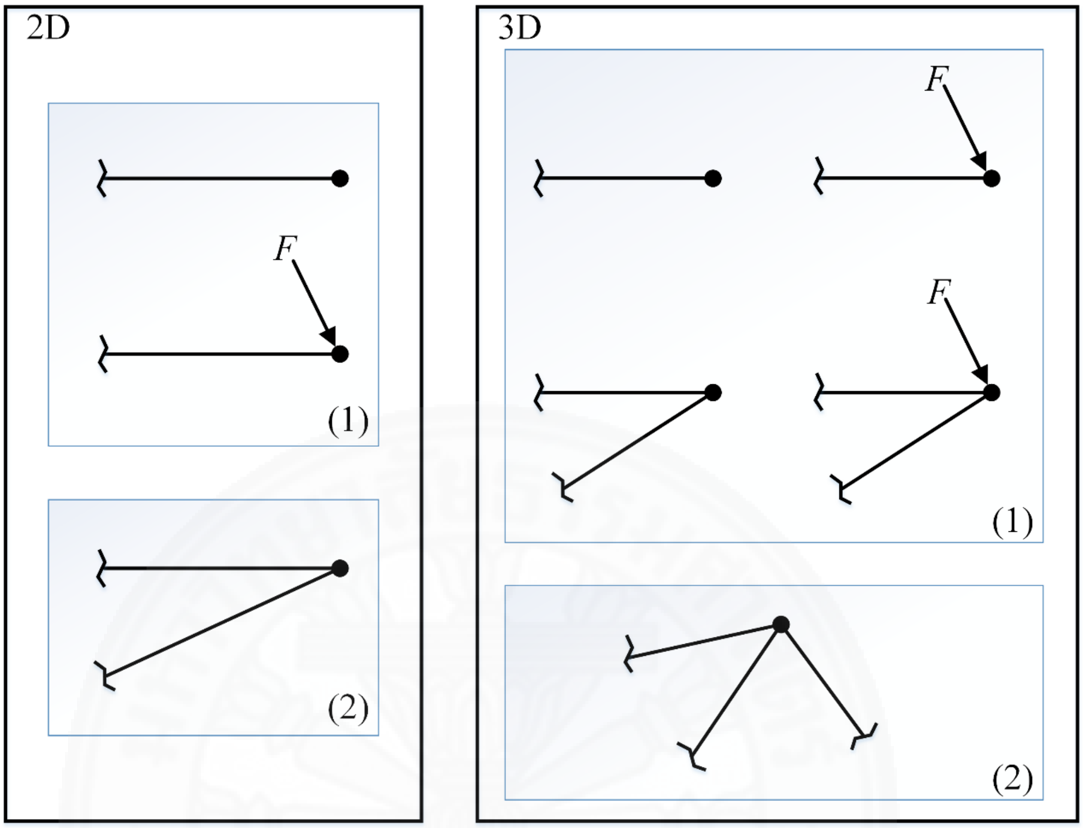


Figure 4.1 Unwanted truss elements.

4.2.2 Removal of unwanted elements

In ground-structure-based coding, a code is attached to each possible truss element, created from the employed ground structure. Each element code can be either a real code or a binary code, and it is usually decoded to give two pieces of information. First, each element code tells whether the element is present or absent. Second, if the element is present, the element code also provides information about the element section. For topology optimization, successful results have been observed when an element code provides roughly the same probability of the presence and absence of an element (Deb & Gulati, 2001; Wu & Tseng, 2010).

As an example, consider a truss element that has two possible choices of predefined sections, namely sections $sect_1$ and $sect_2$. The element code for this element can be a two-bit string. The codes 00 and 01 can be set to mean that the element is absent. In addition, 10 and 11 can be set to mean that the element is present, and its section is equal to $sect_1$ and $sect_2$, respectively. Instead of the two choices of sections, if the area of the element can be selected from a minimum limit of 1 cm^2 to a maximum limit of 225 cm^2, a real code, varied from −225 to 225, can be used as the element code. When the value of the code is less than 1, it can be set to mean that the element is absent. If the value is from 1 to 225, the value can be used to represent the area directly.

The translation of representation codes into corresponding truss structures without element removal can be written formally as follows. Let C_i be the set of all possible element codes of element i in the employed ground structure. Let $\mathbf{\Gamma} = C_1 \times C_2 \times \ldots \times C_M$ be the set of all possible codes for the whole truss. Note that M is the number of all possible elements in the ground structure, defined earlier. Let $Tr_i: C_i \rightarrow S_i$ be a function that maps C_i into S_i, where S_i is the set of all available sections for element i, also defined earlier. These Tr_i's can be used to create a function $\boldsymbol{Tr}: \mathbf{\Gamma} \rightarrow \mathbf{\Psi}$ that maps $\mathbf{\Gamma}$ into $\mathbf{\Psi}$. The function $\boldsymbol{Tr}$ is a translation function that defines how representation codes in $\mathbf{\Gamma}$ are translated into corresponding truss structures in $\mathbf{\Psi}$.

Let $\xi_i \in C_i$ and let $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_M) \in \mathbf{\Gamma}$ denote a representation code. Without an element removal process, the code is decoded into its corresponding truss $\boldsymbol{Tr}(\boldsymbol{\xi}) \in \mathbf{\Psi}$. Although the corresponding truss $\boldsymbol{Tr}(\boldsymbol{\xi})$ can, in general, be a stable or unstable structure, it can be used directly in the optimization process without any alteration. If the truss is not stable, it can be penalized in the optimization process. If the truss includes some useless zero-force elements, the weight of these elements reduces the merit of the truss in the optimization process. As a result, these elements are expected to be later removed by the optimization process. In this study, $\boldsymbol{Tr}(\boldsymbol{\xi})$ is considered as an intermediate truss and is not used directly in the optimization process. Instead, $\boldsymbol{Tr}(\boldsymbol{\xi})$ has to pass through an element-removal algorithm, denoted by $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$, that removes unwanted elements from $\boldsymbol{Tr}(\boldsymbol{\xi})$ to obtain the final truss for the optimization process. To introduce the element-removal algorithms for 2D and 3D trusses, some variables used in the algorithms are introduced below.

N:	number of nodes in the ground structure
$Node_j$:	node *j* where $j = 1,2, \ldots, N$
NEJ_j:	number of elements that are connected to node *j*
E_{ij}:	element i of node *j*
$\mathbf{F}_{ej}$:	external force at node *j*

The algorithms for 2D and 3D trusses are given as follows:

Algorithm: $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$ for 2D trusses

Input: An intermediate truss, $\boldsymbol{Tr}(\boldsymbol{\xi})$

Output: A final truss, $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$

```
1:  Do
2:      $count = 0;$
3:      For $j = 1$ to $N$
4:          If $Node_j \neq \text{Support}$
5:              If $\left(NEJ_j = 1\right)$
6:                  Remove $E_{1j}$;
7:                  $count = count + 1;$
8:              Else If $\left(NEJ_j = 2\right)\&\left(\mathbf{F}_{ej} = \mathbf{0}\right)$
9:                  Remove $E_{1j}$ and $E_{2j}$;
10:                 $count = count + 2;$
11:             End If
12:         End If
13:     End For
14: While $(count > 0)$
```

Algorithm: $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$ for 3D trusses

Input: An intermediate truss, $\boldsymbol{Tr}(\boldsymbol{\xi})$

Output: A final truss, $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$

```
1:  Do
2:      $count = 0;$
3:      For $j = 1$ to $N$
4:          If $Node_j \neq \text{Support}$
5:              If $(0 < NEJ_j < 3)$
6:                  Remove $E_{ij}$, where $i = 1$ to $NEJ_j$;
7:                  $count = count + NEJ_j;$
8:              Else If $\left(NEJ_j = 3\right)\&\left(\mathbf{F}_{ej} = \mathbf{0}\right)$
```

```
9:                      Remove $E_{ij}$, where $i = 1$ to 3;
10:                     $count = count + 3$;
11:                 **End If**
12:             **End If**
13:         **End For**
14:     **While** $(count > 0)$
```

In summary, the whole mapping can be written as

$$\boldsymbol{Truss} = \boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})] = (\boldsymbol{Rem} \circ \boldsymbol{Tr})(\boldsymbol{\xi}). \tag{4.4}$$

Here, $\boldsymbol{Truss} \in \boldsymbol{\Psi}$ is the truss obtained from the element-removal algorithm. The input of the composite function $\boldsymbol{Rem} \circ \boldsymbol{Tr}$ is the representation code $\boldsymbol{\xi}$. In contrast to the chromosome repairing algorithm in the work by Richardson, Adriaenssens, Bouillard and Coelho (2012), in the proposed methodology, the input representation code is not modified. Only the mapping between the representation codes and the corresponding trusses is changed from using $\boldsymbol{Tr}$ to using $\boldsymbol{Rem} \circ \boldsymbol{Tr}$. If an input representation code is modified, it means that the location of this search point in the search space is moved.

In general, the range of $\boldsymbol{Rem} \circ \boldsymbol{Tr}$ is a subset of $\boldsymbol{\Psi}$ that does not include all members of $\boldsymbol{\Psi}$. The members of $\boldsymbol{\Psi}$ that are not included in the range of $\boldsymbol{Rem} \circ \boldsymbol{Tr}$ are those trusses that contain unwanted elements. Although the size of the search space, defined by $\boldsymbol{\Gamma}$, is not reduced by the proposed methodology, the number of resulting trusses is reduced by reinterpreting some unstable and inferior trusses as potentially better ones. Consequently, more individuals in the search space can be meaningfully compared. For example, it is difficult to meaningfully compare two kinematically unstable trusses and decide which one is better. However, if the trusses become stable and efficient trusses after their unwanted elements are removed, then it is straightforward to decide which truss is better, based on the employed objective. Having a smaller number of corresponding trusses for the search space, and more stable and efficient trusses among them, results in faster convergences to optimal solutions. Note that removing unwanted elements defined above does not guarantee the stability of the resulting trusses.

Figure 4.2 shows an example 2D ground structure having four nodes and six possible elements. Assume that the area of each element can vary from 0 to 225 cm^2. Let the set of all possible element codes of element i, C_i, be defined as

$$C_i = \{x | -225 \le x \le 225\}. \tag{4.5}$$

Consequently, the set of all possible codes for the whole truss $\mathbf{\Gamma}$ is $C_1 \times C_2 \times \ldots \times C_6$. Since the area of each element varies from 0 to 225 cm^2, the set of all available sections for element i, S_i, can be expressed directly in terms of area as

$$S_i = \{area | 0 \le area \le 225\}. \tag{4.6}$$

Clearly, the set of all possible trusses $\mathbf{\Psi}$ becomes $S_1 \times S_2 \times \ldots \times S_6$.

The translation function $Tr_i: C_i \to S_i$ is simply defined as

$$s_i = Tr_i(x) = \begin{cases} 0 & \text{if } x < 0; \\ x & \text{if } x \ge 0 \end{cases} \tag{4.7}$$

where $-225 \le x \le 225$. Note that, in this example, all Tr_i's are the same. The translation function $\boldsymbol{Tr}: \mathbf{\Gamma} \to \mathbf{\Psi}$ is written as

$$\boldsymbol{Tr}(\boldsymbol{\xi}) = \big(Tr_1(\xi_1), Tr_2(\xi_2), \ldots, Tr_6(\xi_6)\big) \tag{4.8}$$

where $-225 \le \xi_i \le 225$ and $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_6) \in \mathbf{\Gamma}$.

It can be seen from Equations (4.5), (4.6), (4.7), and (4.8) that, within the range of $\xi_i \in C_i$ from -225 to 225, $-225 \le \xi_i \le 0$ represents the absence of element i while $0 < \xi_i \le 225$ represents the nonzero area of element i. In this example, the translation from the value of ξ_i to the area of element i provides the same probability of the presence and absence of element i.

Some translation examples are shown in Figure 4.2. In case I in Figure 4.2, the representation code $\boldsymbol{\xi} = (\boldsymbol{\xi_1}, \boldsymbol{\xi_2}, \ldots, \boldsymbol{\xi_6})$ is equal to $(100{,}150, -10, -90{,}200{,}35)$. By using the translation function $\boldsymbol{Tr}(\boldsymbol{\xi})$ in Equation (4.8), an intermediate truss, defined by the areas of the elements, is obtained as $\boldsymbol{Tr}(\boldsymbol{\xi}) = (100{,}150{,}0{,}0{,}200, 35)$. This intermediate truss is put through the element-removal algorithm $\boldsymbol{Rem}$, and the final truss $\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$ is obtained. It can be seen from Figure 4.2 that, although the representation codes of cases I, II, and III are different, their final trusses are the same. This means that these representation codes are mapped into the same truss. During the translation process, each input representation code $\boldsymbol{\xi}$ is not modified in any way, and its position in the search space does not change. Only the translation of the code is modified by the element-removal algorithm $\boldsymbol{Rem}$.

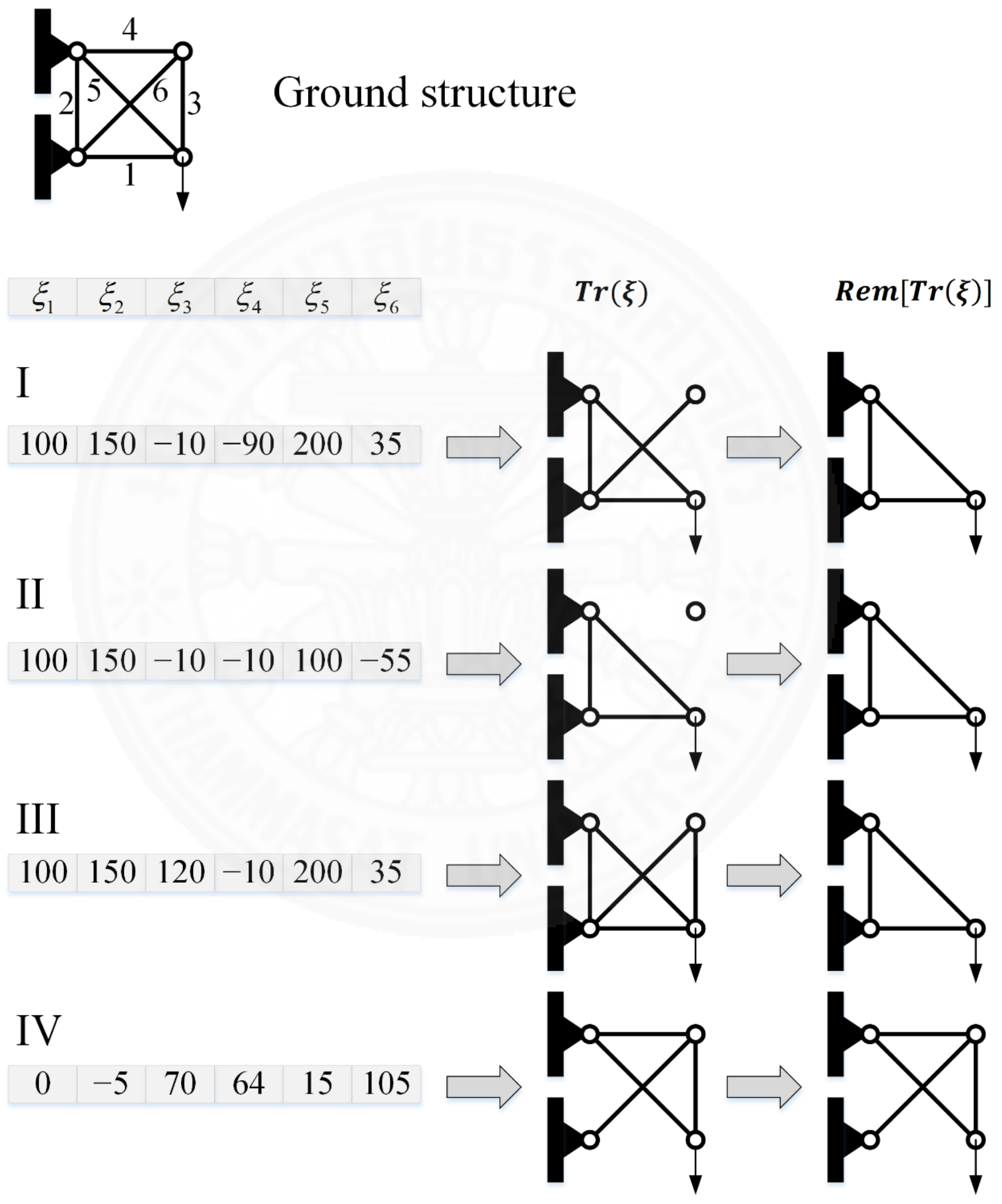
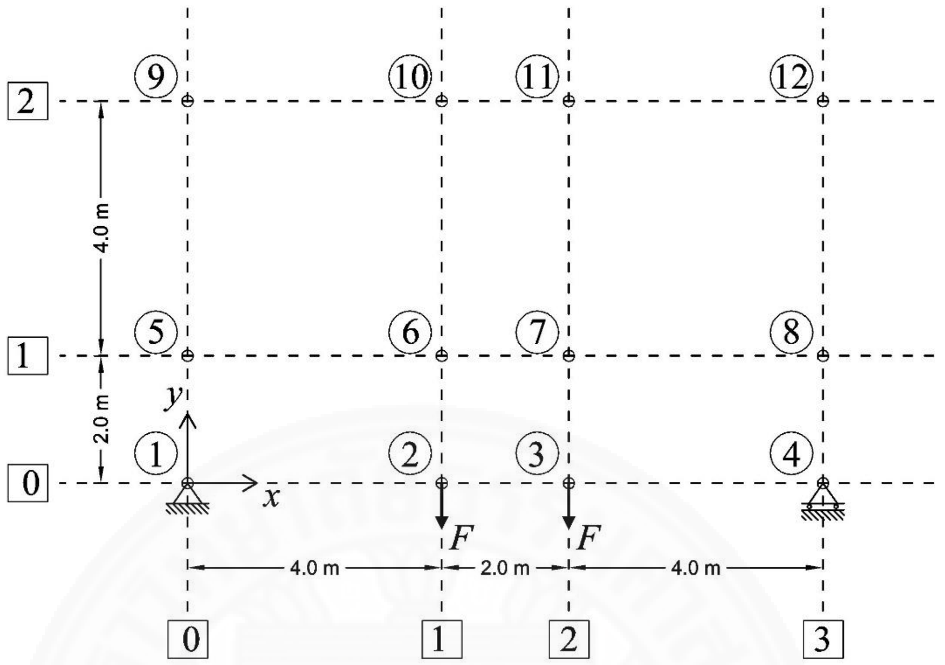

Ground structure
1
2
3
4
5
6
ξ_1 ξ_2 ξ_3 ξ_4 ξ_5 ξ_6
$\boldsymbol{Tr}(\boldsymbol{\xi})$
$\boldsymbol{Rem}[\boldsymbol{Tr}(\boldsymbol{\xi})]$
I
100 150 −10 −90 200 35
II
100 150 −10 −10 100 −55
III
100 150 120 −10 200 35
IV
0 −5 70 64 15 105

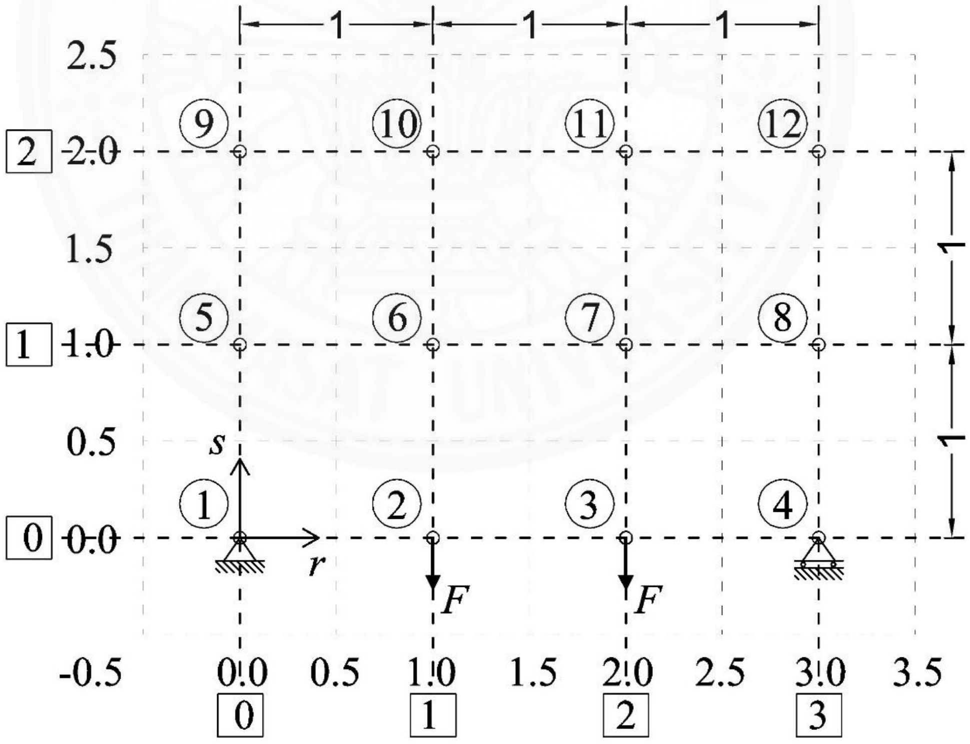
Figure 4.2 Translation examples.

4.3 Truss topology optimization with a prescribed maximum number of elements

In addition to the element-removal algorithm discussed above, in this study, a new coding scheme for truss topology optimization, in which no ground structure is used, is proposed. For a truss topology optimization problem, instead of joining all nodes in a selected ground structure to create possible positions of truss elements, the maximum number of elements that can appear in the truss is initially set. Practically, this number is to be set by the designer of the truss. The main concept of the proposed methodology is to use the end positions of these elements and their cross-sectional areas as the design variables of the problem. However, if the end positions of these elements can vary continuously, it is virtually impossible that these end positions end up at some same positions and, subsequently, connect the elements to form a truss. To avoid this problem, the end positions are allowed to be located only at some grid points.

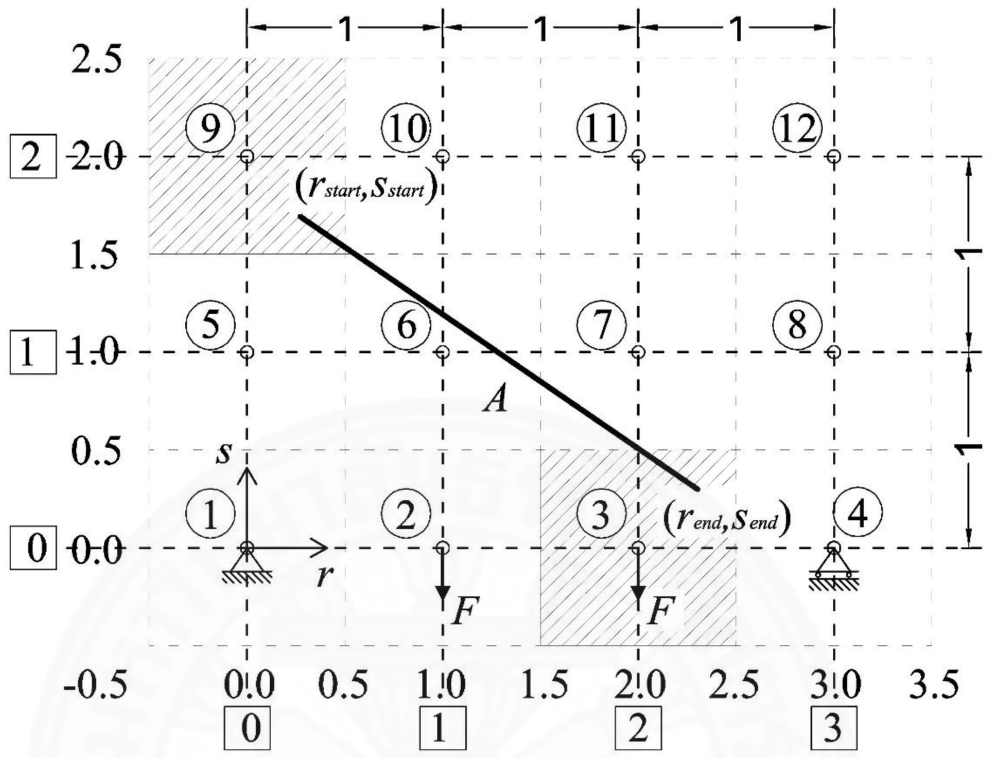


(a)

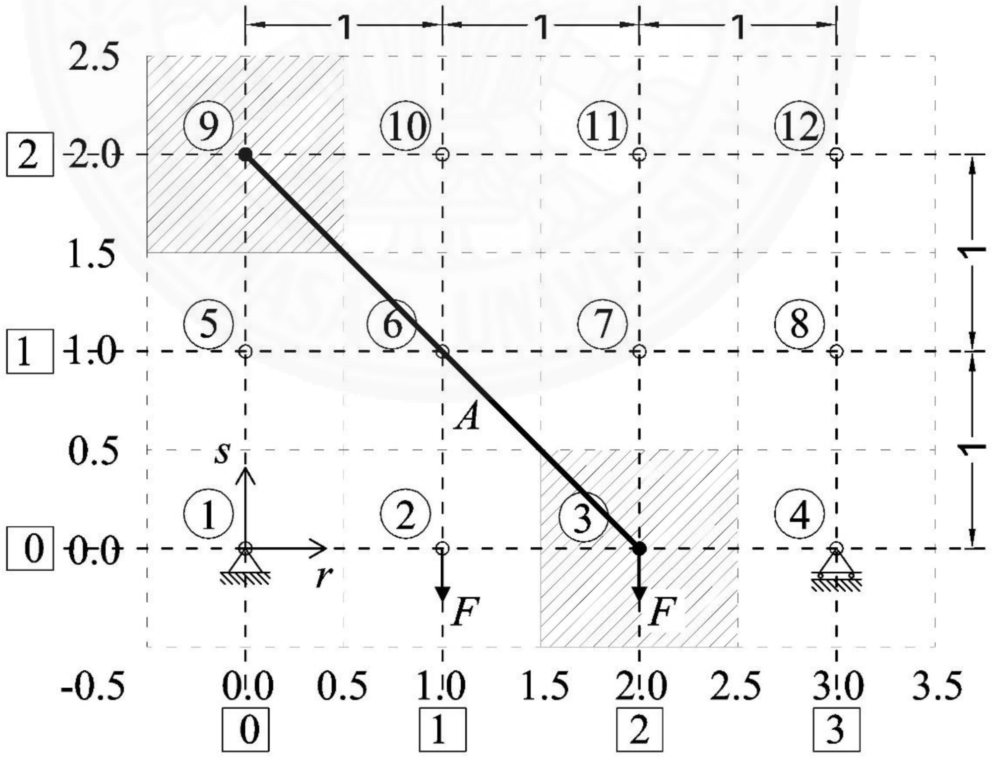


(b)

Figure 4.3 Grid mapping: (a) Grid for the original problem; (b) Substitute grid with uniform unit spacing.



(a)



(b)

Figure 4.4 Element snapping: (a) Original element position; (b) Snapping of element nodes to the nearest grid points.

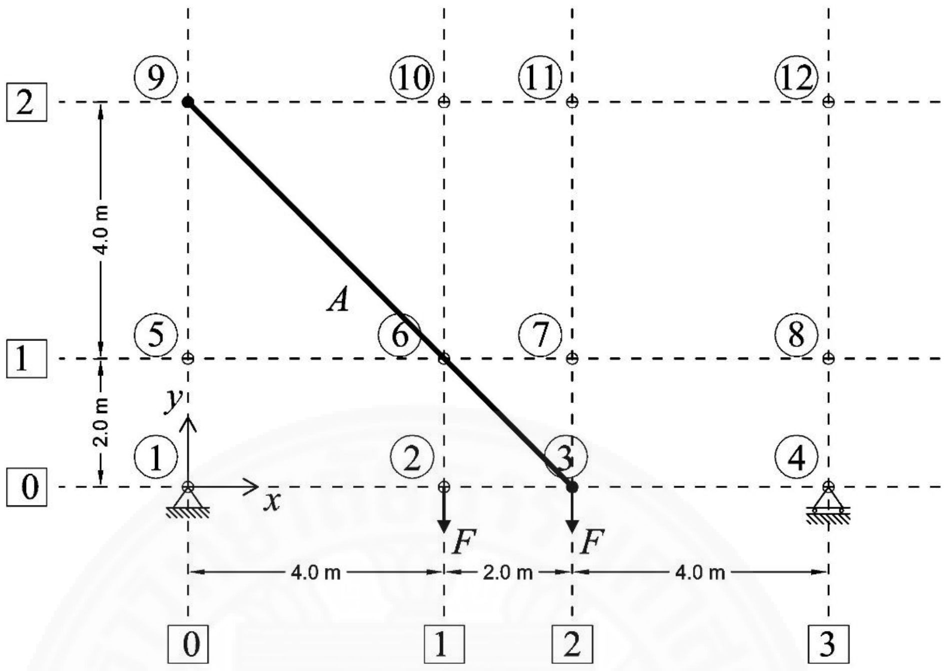
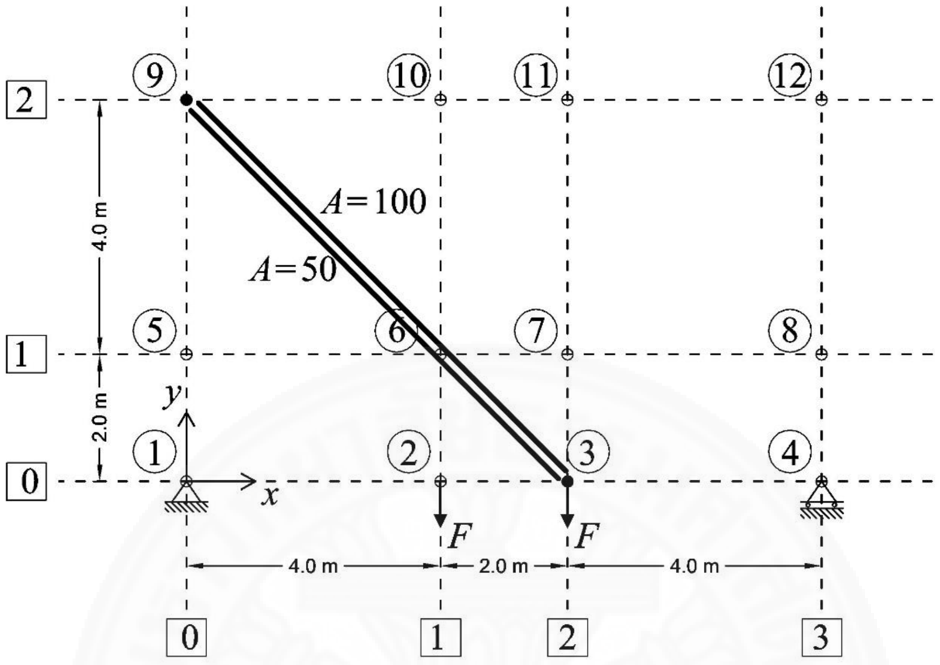


Figure 4.5 Position of element in the real space after snapping.

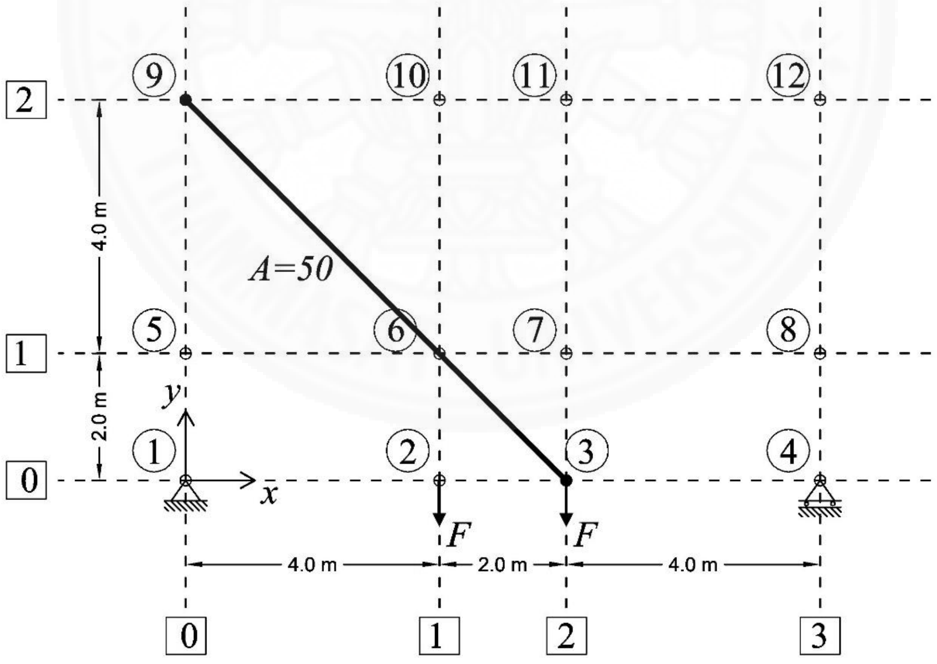
To better explain the proposed concept, a topology optimization problem of a 2D truss in Figure 4.3 (a) is considered. The truss to be optimized has two supports and two applied forces, as shown in the figure. To optimize the truss, a 4×3 grid is constructed. The grid is intentionally made nonuniform. The vertical and horizontal grid lines are separately numbered. In the figure, each of these grid line numbers is enclosed in a square. The 12 grid points of the 4×3 grid are used as the probable positions of the joints of the truss. The grid points are also numbered and, in the figure, each of the grid point numbers is enclosed in a circle.

In this study, the end positions of an element are coded using continuous variables. During the decoding process, the end positions are snapped into predefined grid points. For the problem in Figure 4.3 (a), these predefined grid points are the 12 grid points shown in Figure 4.3 (a). The values of the end positions of an element can be made to vary continuously in the real space of the problem and the two ends of the element can be simply made to snap into the nearest grid points in the real space. However, this simple strategy has a flaw when a nonuniform grid, such as the one in Figure 4.3 (a), is used. When a nonuniform grid is used, the probabilities of a node snapping into different grid points can be different. This creates biases in the optimization process. To remove the flaw, the original grid must be mapped into a substitute uniform grid. Figure 4.3 (b) shows a substitute grid with uniform unit spacing for the original grid in Figure 4.3 (a). The coordinates of element ends can then be made to vary in the space of the substitute grid instead of the real space. The coordinates of element ends must be set to vary beyond the minimum and maximum coordinates of the substitute grid in order that the probabilities of a node snapping into the boundary grid points are the same as those of the internal grid points. For example, the minimum and maximum r coordinates of the substitute grid in Figure 4.3 (b) are, respectively, equal to 0.0 and 3.0. However, the r coordinate of an element end has to be set to vary from -0.5 to 3.5. Similarly, the minimum and maximum s coordinates of the substitute grid are, respectively, equal to 0.0 and 2.0. The s coordinate of an element end has to be set to vary from -0.5 to 2.5.

Figure 4.4 (a) shows an example position of an element before the snapping process is employed. It can be seen that the nearest grid point to one end of the element is grid point 9 and the nearest grid point to the other end is grid point 3. Thus, the two ends are, respectively, snapped into grid points 9 and 3, as shown in Figure 4.4 (b). In this study, if the position of an element end in a certain coordinate lies exactly between two grid points, the end is snapped into the grid point that has the higher value of that coordinate. Figure 4.5 shows the position of the element in Figure 4.4 in the real space after snapping. It is possible that, in a truss, there are many elements occupying exactly the same position although they can have different cross-sectional areas. In order to create the final truss from the representation code, among these elements, only the one that has the minimum cross-sectional area is retained. The rest is removed from the final truss. Note that, similar to the element-removal algorithm, during the snapping process and the removal of elements that occupy the same position, the representation code is not modified. Finally, if the two nodes of an element are at the same grid point, it is simply interpreted that there is no element.



(a)



(b)

Figure 4.6 Area selection for repeated elements: (a) Two elements in the same position with different cross-sectional areas; (b) Only the element with the smallest cross-sectional area retained.

As aforementioned, the other design variable that is attached to an element, in addition to its end positions, is the cross-sectional area A of the element. Thus, in a 2D problem, there are five design variables attached to each truss element, i.e. $r_{start}, s_{start}, r_{end}, s_{end}$, and A, where r and s are the coordinates of the substitute grid, as shown in Figure 4.4. In a 3D problem, there are seven design variables attached to an element, i.e. $r_{start}, s_{start}, u_{start}, r_{end}, s_{end}, u_{end}$, and A. Here, $r, s,$ and u denote the coordinates of the 3D substitute grid. The cross-sectional areas of elements can either be continuous or discrete variables. In this study, the cross-sectional areas of elements are assumed to be continuous variables. If the number of initial elements is set to n, the total number of design variables becomes $5 \times n$ for a 2D truss and $7 \times n$ for a 3D truss. Note that the number of design variables is not a function of the number of grid points. Thus, the employed grid can be refined without increasing the size of search space. On the contrary, if the ground structure approach is used, refining the employed grid quickly increases the size of search space. As mentioned earlier, in the ground structure approach, the total number of possible truss elements is $N(N-1)/2$, where, N is the number of grid points. In the proposed approach, the size of search space increases only linearly with respect to the number of initial elements.

The design variable that represents the cross-sectional area of an element is in fact used not only to represent the area but also to identify whether the element is absent or present. The range of the design variable must cover two groups of values that represent, respectively, the element being absent and present. The available ranges of the two groups used for an element define the probability of the element being absent and present. In most studies, the cross sectional area of an element varies from $A_{critical}$ to A_{max}, where $A_{critical}$ is the minimum non-zero area and A_{max} is the maximum area. However, as shown in Figure 4.7, the corresponding design variable X is usually set to vary from a negative number X_{mn} to a positive number X_{mx}. In addition, X_{mx} is always

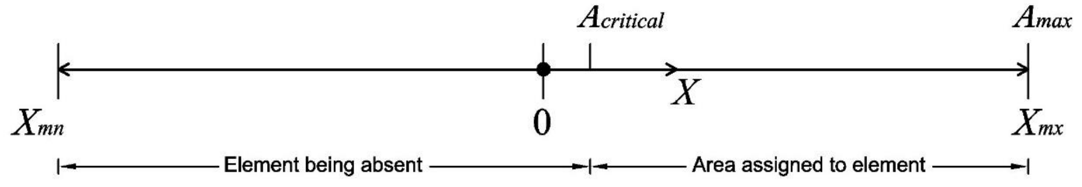


Figure 4.7 Range of an area design variable.

set to be equal to A_{max} while X_{mn} is usually set to be equal to $-X_{mx}$. When the design variable X is less than $A_{critical}$, the cross-sectional area is taken as zero and the element is absent. When X is not less than $A_{critical}$, X is usually interpreted directly as the cross-sectional area. Figure 4.7 demonstrates this coding concept for cross-sectional areas. Since $A_{critical}$ is normally small, setting $X_{mn} = -X_{mx}$ results in the same probability of the presence and absence of an element.

Generally, the number of elements that remain in the optimal topology of a truss is much smaller than the total number of elements found in any reasonable ground structures that can be used with that truss problem. This means that, in order to obtain the optimal topology, most of the elements in the employed ground structure must be removed. It follows that the chance of an element being removed from the final topology cannot be set too small. This is the reason that, in most studies that employ the ground structure approach, the probability of an element being absent is usually set, by setting X_{mn}, to be approximately equal to 0.5. In the method proposed in this study, the maximum number of elements is prescribed, and this number is generally assumed to not be significantly larger than the number of elements to be remained in the optimal truss. This means that the probability of an element being absent can be reduced from those used in the ground structure approach. In this study, a term called the minimum area factor, $A_{minFactor}$, is introduced for adjustments of the probability of an element being absent, i.e.

$$\frac{-X_{mx}}{A_{minFactor}} \le X \le X_{mx} \qquad (4.9)$$

In this study, when the maximum number of elements is prescribed, the values of $A_{minFactor}$ that are greater than one are used.

Some of the points to be considered when selecting the maximum number of elements are as follows:

1. The maximum number of elements must be enough to create a stable truss under the prescribed supports and forces.
2. The maximum number of elements must be greater than the number of elements in the optimal solution in order that the optimal solution can be obtained. Since the optimal solution is not known, educated guesswork must be employed.

3. The maximum number of elements should be quite smaller than the number of elements generated by joining all the grid points in order to take full advantage of the proposed method.

CHAPTER 5
RESULTS

In this chapter, the two proposed coding schemes, namely the element-removal algorithm and the truss optimization algorithm with a prescribed maximum number of elements, are tested and their results are shown. To demonstrate the effectiveness of the proposed representations, they are used in the two-population PSO algorithm (Nanakorn, Petprakob & Naga, 2014) to solve several truss topology optimization problems. Since the employed PSO algorithm includes stochastic processes, for each of the considered problems, the PSO algorithm is run for 1000 times. This is done in order to explore both the quality and uniformity of the obtained results. Each set of these 1000 runs is called a calculation set. The best solution of a run is called the run solution. Since there are 1000 runs in a calculation set, there are 1000 run solutions from these 1000 runs. The best solution of these 1000 run solutions is the best solution of the calculation set or the calculation-set best.

The PSO parameters employed for all the example problems are shown in Table 5.1. Different values of W, ϕ_1, and ϕ_2 are used for the two populations so that they progress differently, and the enhancement of the exploration is therefore ensured. The values of $W = 0.729$ and $\phi_1 = \phi_2 = 1.494$, used in one of the populations, are from the constriction method (Clerc & Kennedy, 2002; Eberhart & Shi, 2000). In this study, V_{dmx} for each optimization dimension is set to one quarter of the range of all possible codes for that dimension.

Table 5.1 PSO parameters.

Parameter	Population	
	1	2
Population size	10	10
W	0.500	0.729
$\phi_1 = \phi_2$	2.000	1.494
V_{dmx}	$(X_{dmx} - X_{dmn})/4$	$(X_{dmx} - X_{dmn})/4$
Number of iterations	1000	1000
p_s	0.4	0.4
K	50	50

5.1 Element-removal algorithm

To demonstrate the effectiveness of the proposed element-removal algorithm, six problems of truss topology optimization, including two 3D problems, are solved. Here, the ground structure approach is still used in order that the advantages of the element-removal algorithm can be clearly observed. The problems are solved with and without the proposed element-removal algorithm, and their results are compared. This is to see how the element-removal algorithm can improve the optimization performance. In addition, the obtained results are also compared with those from the literature, wherever possible. The comparison with the literature is done only to make certain that the results obtained from this study are satisfactory. It is not in the scope of this study to discuss the performance of the whole optimization process, which also includes the performance of the employed optimization method, namely the two-population PSO algorithm.

5.1.1 45-element, 10-node truss

The first problem is a 2D truss optimization problem from a ground structure with 10 nodes. The ground-structure nodes, forces, supports, and problem dimensions are shown in Figure 5.1 (Deb & Gulati, 2001; Wu & Tseng, 2010). Here, all possible node connections are considered. In this study, the symmetry along the middle vertical line of the solutions is assumed. Consequently, the number of the design variables is reduced from 45 to 25. The problem parameters are as follows (Deb & Gulati, 2001; Wu & Tseng, 2010):

Young's modulus: $E = 10^4$ ksi

Weight density: $\rho = 0.1\ \mathrm{lb/in^3}$

Allowable stress: $\sigma_a = 25$ ksi

Allowable displacement: $\delta_a = 2$ in

Area: $A = 0.09$ to $1\ \mathrm{in^2}$

The magnitudes of the stresses in all elements, for tension and compression, are limited to σ_a. The magnitudes of all displacement degrees of freedom are limited to δ_a.

The design variables of the problem are the areas of all possible elements. Since the area of each element varies from 0.09 to 1 $\mathrm{in^2}$, the set of all available areas for element i, S_i, is expressed as

$$S_i = \{A \mid (A = 0) \text{ or } (0.09 \le A \le 1)\}. \tag{5.1}$$

The set of all possible element codes of element i, C_i, is defined as

$$C_i = \{x \mid X_{imn} \le x \le X_{imx}\} = \{x \mid -1 \le x \le 1\}. \tag{5.2}$$

The translation function $Tr_i: C_i \to S_i$ is defined as

$$s_i = Tr_i(x) = \begin{cases} 0 & \text{if } x < 0.09; \\ x & \text{if } x \ge 0.09. \end{cases} \tag{5.3}$$

For this problem, the value of V_{dmx} is given as, $V_{dmx} = [1 - (-1)]/4 = 0.5$ for all optimization dimensions.

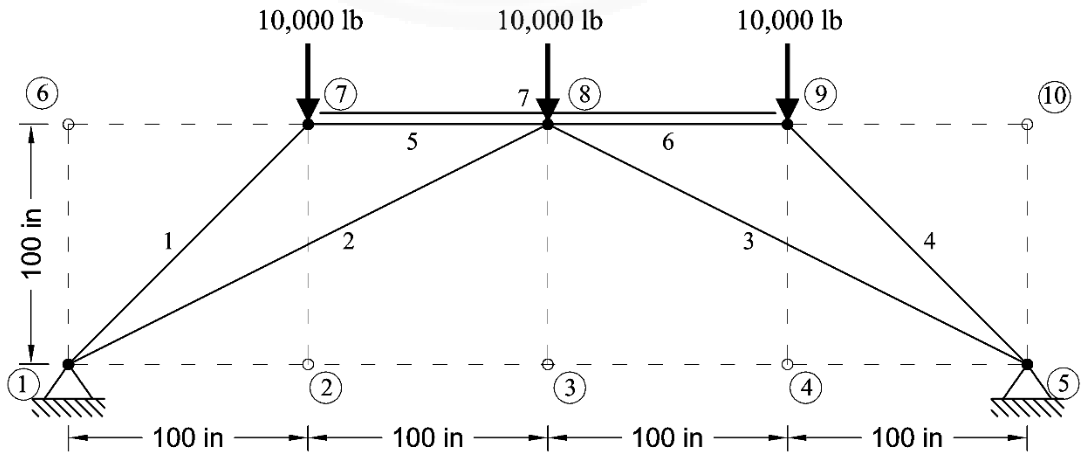


Figure 5.1 Problem 5.1.1: Problem details, ground-structure nodes, and best topology.

It is found from the results that all run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.2 shows the statistics of the run solutions obtained with and without the element-removal algorithm. In the table, the minimum, maximum, average, and standard deviation of the weights of the 1000 run solutions from the 1000 runs of each calculation set are shown. The minimum weight obtained with the element-removal algorithm is exactly the same as the one obtained without the element-removal algorithm. However, the average and maximum weights obtained with the element-removal algorithm are significantly smaller than those obtained without the algorithm. The same is true for the standard deviations. Table 5.2 also shows the percentage of the run solutions in a calculation set that are not heavier than the best solution of the calculation set by greater than 2%. It is considered in this study that any weight difference of not more than 2% is insignificant. It can be seen from Table 5.2 that, when the element-removal algorithm is employed, 78.9% of the run solutions are not heavier than the best solution of the calculation set by more than 2%. When the element-removal algorithm is not used, only 50.5% of the run solutions are not heavier than the calculation-set best by more than 2%. These results show that the element-removal algorithm consistently gives high-quality results.

Table 5.2 Problem 5.1.1: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (lb)	44.000	44.000
Average weight (lb)	45.851	49.283
Maximum weight (lb)	103.685	120.006
SD of weights (lb)	4.994	7.909
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	78.9	50.5

The best solutions of the two calculation sets, with and without the element-removal algorithm, are found to have the same topology that is shown in Figure 5.1. This topology is the same as the results by Deb and Gulati (2001) and Wu and Tseng (2010). Table 5.3 shows the details of the best solutions from this study and from Deb and Gulati (2001) and Wu and Tseng (2010). The results by Deb and Gulati (2001) and Wu and Tseng (2010) are presented as originally reported. Note that the symmetry of the problem is not considered by Deb and Gulati (2001) and Wu and Tseng (2010). It can be seen that the best solutions from this study compare satisfactorily with the best solutions from Deb and Gulati (2001) and Wu and Tseng (2010). In fact, the best solution of the present study, obtained with the element-removal algorithm, can be considered to be same as the result by Wu and Tseng (2010). It can be seen that the summation of the areas of elements 5 and 7 and the summation of the areas of elements 6 and 7 in the results by this study and Wu and Tseng (2010) are the same. The difference in the total weights from this study and Wu and Tseng (2010), shown in Table 5.3, is only from rounding.

Table 5.3 Problem 5.1.1: Best solutions.

Member	Area (in^2)			
	This study		Deb and Gulati (2001)[b]	Wu and Tseng (2010)[b]
	With the element-removal algorithm[a]	Without the element-removal algorithm[a]		
1	0.566	0.566	0.566	0.566
2	0.447	0.447	0.477	0.447
3	0.447	0.447	0.477	0.447
4	0.566	0.566	0.566	0.566
5	0.118	0.100	0.082	0.09
6	0.118	0.100	0.080	0.09
7	0.282	0.300	0.321	0.31
Weight (lb)	44.000	44.000	44.033	43.99
Max stress (ksi)	25.000	25.000	–	25.0
Max displacement (in)	1.250	1.250	–	1.108

[a]Symmetry is assumed.

[b]Symmetry is not assumed.

The convergences of runs with the element-removal algorithm are found to be faster than those without the algorithm. Figure 5.2 shows typical convergences of the best weights for runs with and without the element-removal algorithm. The plots clearly show that the solution converges faster when the element-removal algorithm is used.

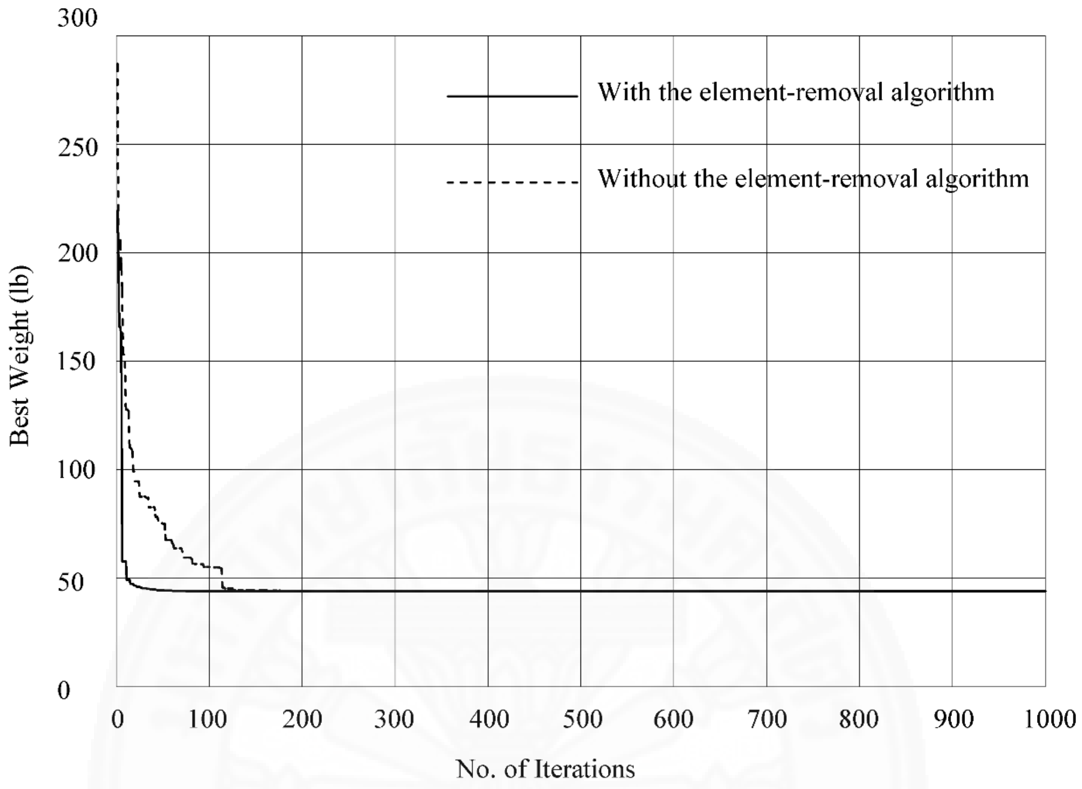


Figure 5.2 Problem 5.1.1: Typical convergences of the weights.

5.1.2 39-element, 12-node truss

The second problem is the 2D truss optimization problem in Figure 5.3 (Deb & Gulati, 2001; Wu & Tseng, 2010). The ground structure has 12 nodes, located on grid points of 5×3 grid lines. The circled numbers 1 to 15 in Figure 5.3 are grid point numbers. There are no nodes at grid points 8, 11, and 15. Moreover, only node connections shown in Figure 5.3 are considered. In this study, the symmetry along the middle vertical line of the solutions is assumed. As a result, the number of the design variables is reduced from 39 to 21. The problem parameters are as follows (Deb & Gulati, 2001; Wu & Tseng, 2010):

Young's modulus: $E = 10^4$ ksi

Weight density: $\rho = 0.1 \text{ lb/in}^3$

Allowable stress: $\sigma_a = 20$ ksi

Allowable displacement: $\delta_a = 2$ in

Area: $A = 0.05$ to 2.25 in^2

In the PSO, the lower and upper boundaries of the design variables in all optimization dimensions are set, respectively, to -2.25 and 2.25 in^2. If the value of a design variable is less than 0.05, it is interpreted as a section of zero area. In addition, V_{dmx} is set to $[2.25-(-2.25)]/4 = 1.125$ for all optimization dimensions.

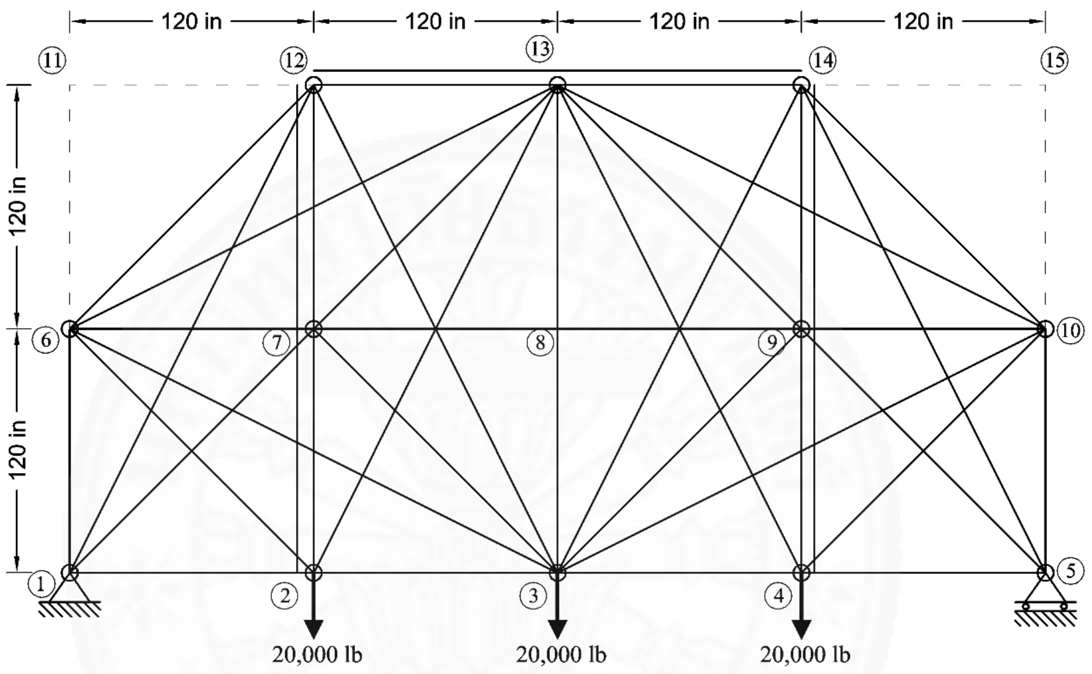


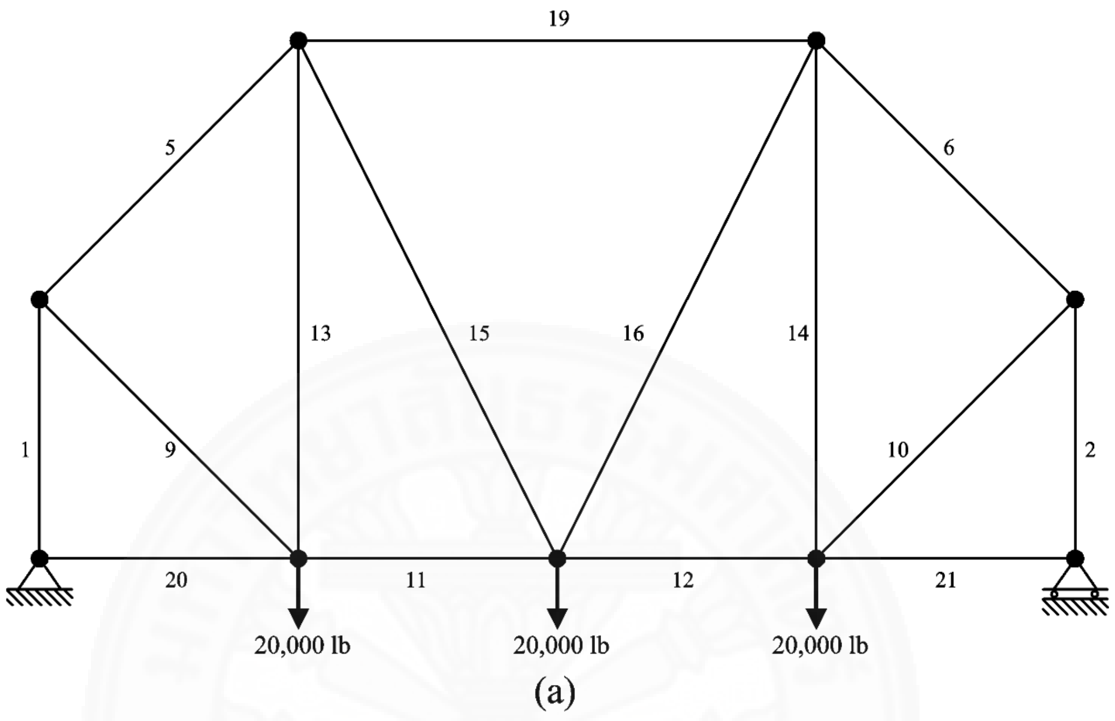
Figure 5.3 Problem 5.1.2: Problem details, ground-structure nodes and elements.

It is found from the results that all run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.4 shows the statistics of the run solutions obtained with and without the element-removal algorithm. The minimum weight obtained with the element-removal algorithm is the same as the one obtained without the element-removal algorithm. The average weight obtained with the element-removal algorithm is slightly smaller than the one obtained without the algorithm. The maximum weight obtained with the element-removal algorithm is found to be greater than that obtained without the algorithm. The standard deviation of the weights obtained with the element-removal algorithm is slightly greater than that obtained without the algorithm. It can be seen from Table 5.4 that, when the element-removal algorithm is employed, 17.7% of the run solutions are not heavier than the best solution of the calculation set by more than 2%. On the contrary, when the element-removal algorithm is not used, only 9.0% of the run solutions are not heavier than the calculation-set best by more than 2%. The element-removal algorithm significantly increases the number of excellent solutions in the calculation set.

Table 5.4 Problem 5.1.2: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (lb)	193.200	193.200
Average weight (lb)	228.183	228.602
Maximum weight (lb)	467.072	408.842
SD of weights (lb)	26.485	25.062
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	17.7	9.0

The best solutions of the two calculation sets, with and without the element-removal algorithm, are found to have the same topology that is shown in Figure 5.4 (a). This topology is the same as the result by Wu and Tseng (2010) but is different from the result by Deb and Gulati (2001). The best topology obtained by Deb and Gulati (2001) is shown in Figure 5.4 (b). Table 5.5 shows the details of the best solutions from this study and from Deb and Gulati (2001) and Wu and Tseng (2010). The results by Deb and Gulati (2001) and Wu and Tseng (2010) are presented as originally reported. The symmetry of the problem is also considered by Deb and Gulati (2001) and Wu and Tseng (2010). It can be seen that the best solutions from this study compare satisfactorily with the best solutions from Deb and Gulati (2001) and Wu and Tseng (2010). The difference in the total weights from this study and Wu and Tseng (2010) is negligible.



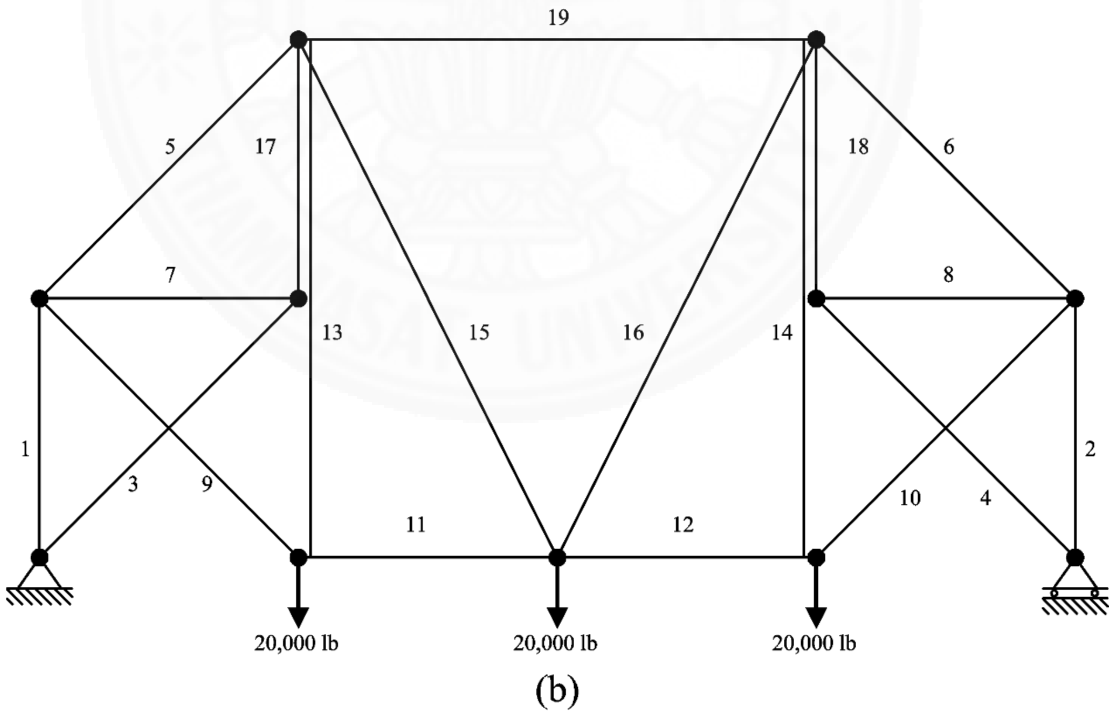


Figure 5.4 Problem 5.1.2: (a) Best topology by this study; (b) Best topology by Deb and Gulati (2001).

Table 5.5 Problem 5.1.2: Best solutions.

Member	Area (in^2)			
	This study		Deb and Gulati (2001)[a]	Wu and Tseng (2010)[a]
	With the element-removal algorithm[a]	Without the element-removal algorithm[a]		
1,2	1.500	1.500	1.502	1.500
3,4	–	–	0.051	–
5,6	1.061	1.061	1.063	1.060
7,8	–	–	0.051	–
9,10	1.061	1.061	1.061	1.060
11,12	0.750	0.750	0.751	0.750
13,14	0.250	0.250	0.251	0.250
15,16	0.559	0.559	0.559	0.559
17,18	–	–	0.052	–
19	1.000	1.000	1.005	1.000
20,21	0.050	0.050	–	0.05
Weight (lb)	193.200	193.200	196.546	193.199
Max stress (ksi)	20.000	20.000	–	20.000
Max displacement (in)	1.440	1.440	–	–

[a]Symmetry is assumed.

Figure 5.5 shows typical convergences of the best weights for runs with and without the element-removal algorithm. Similar to the previous problem, the solution converges faster when the element-removal algorithm is used.

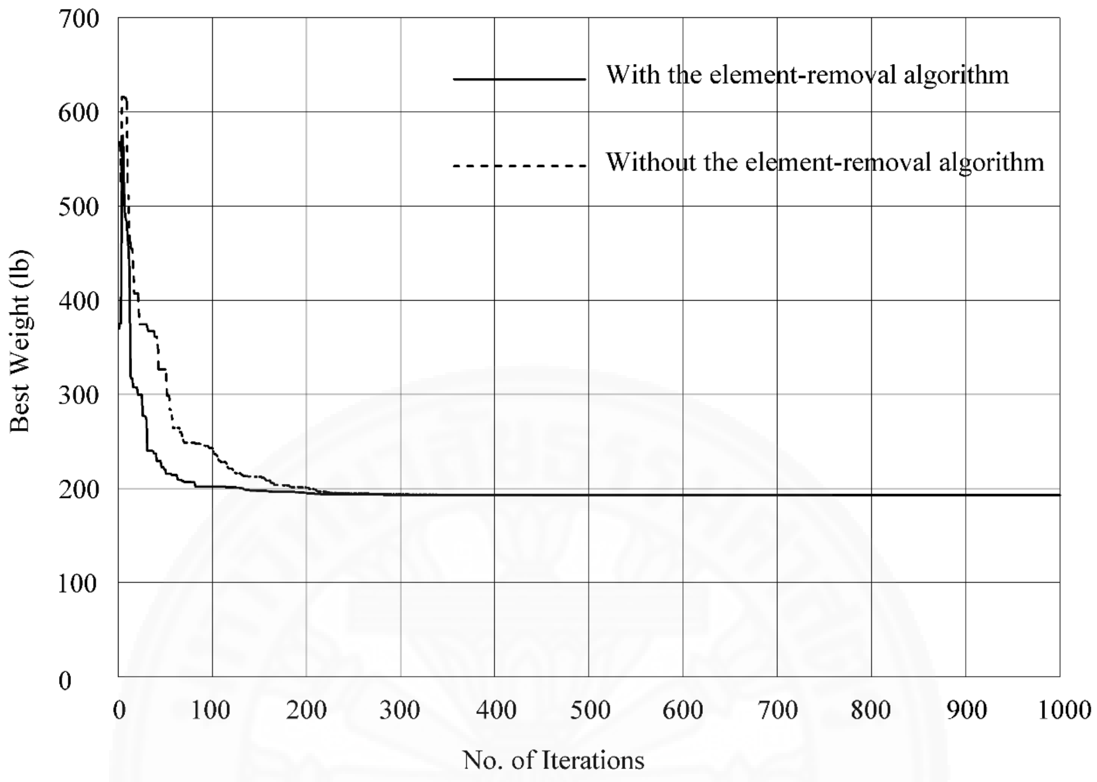


Figure 5.5 Problem 5.1.2: Typical convergences of the weights.

5.1.3 30381-element, 247-node truss

The third problem is a 2D truss optimization problem from a ground structure with 247 nodes. The ground structure nodes, force, supports, and problem dimensions are shown in Figure 5.6. The grid spacing of the ground structure is uniform. The problem is taken from the work by Faramarzi and Afshar (2012), except for the ground structure. In the work by Faramarzi and Afshar (2012), a coarser ground structure having 12 nodes is used. In this study, all 30381 connections between 247 nodes are considered. Since the behavior of truss elements in tension and compression is considered the same, the symmetry along the middle horizontal line of the solutions is assumed. This results in the number of design variables equal to 15,333. The problem parameters are as follows (Faramarzi & Afshar, 2012):

Young's modulus: $E = 10^4$ ksi

Weight density: $\rho = 0.1\ \mathrm{lb/in^3}$

Allowable stress: $\sigma_a = 5$ ksi

Allowable displacement: $\delta_a = 0.6$ in

Area: $A = 0.09$ to 35 in^2

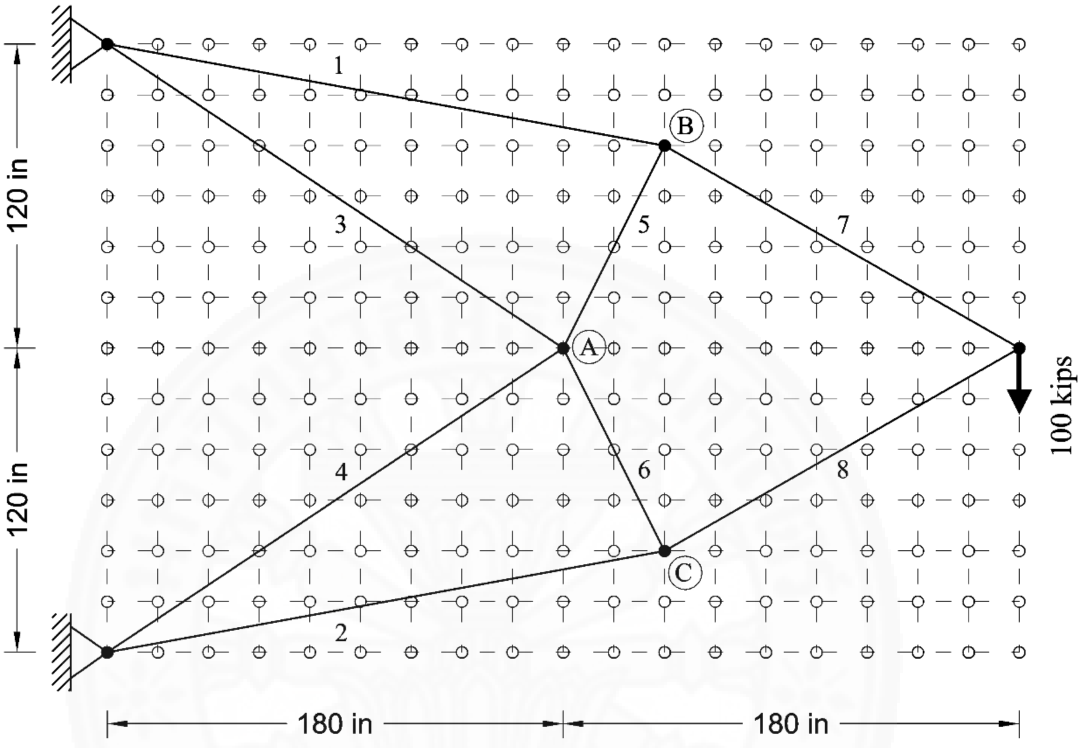


Figure 5.6 Problem 5.1.3: Problem details, ground-structure nodes, and best topology.

In PSO, the lower and upper boundaries of the design variables in all optimization dimensions are set, respectively, to -560 and 35. If the value of a design variable is less than 0.09, it is interpreted as a section of zero area. The very low values of the lower boundaries are used to increase the probability of elements being absent. This is because there are a large number of elements in the ground structure of this problem and, as a result, the probability of elements being removed has to be increased. Note that the values of the lower boundaries equal to -560 are obtained by trial and error. In this problem, V_{dmx} is set to $[35-(-560)]/4 = 148.75$ for all optimization dimensions.

Table 5.6 Problem 5.1.3: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (lb)	2200.006	100709.914
Average weight (lb)	5939.536	161614.786
Maximum weight (lb)	92324.122	239670.624
SD of weights (lb)	13236.397	22980.185
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	0.9	0.1
Percentage of the run solutions that are not heavier than 2400.000 lb (%)	81.1	0.0

All run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.6 shows the statistics of the run solutions obtained with and without the element-removal algorithm. All results in the table, obtained with the algorithm, are considerably better than those obtained without the algorithm. The minimum weight obtained with the element-removal algorithm is much lower than the one obtained without the algorithm. In fact, the minimum weight obtained without the element-removal algorithm is very large. In addition, the minimum weight of 2200.006 lb obtained in this study is lower than the minimum weight of 2400.001 lb obtained with a coarser ground structure having 12 nodes by Faramarzi and Afshar (2012). Even when the element-removal algorithm is used, the percentage of the run solutions that are not heavier than the best solution of the calculation set by more than 2% is found to be quite low. This is due to the large size of the search space, which makes the problem difficult to solve. Nevertheless, when the element-removal algorithm is employed, 81.1% of the run solutions are not heavier than

2400.000 lb, which is a rounded down value of the minimum weight obtained by Faramarzi and Afshar (2012). When the element-removal algorithm is not used, all run solutions are much heavier than 2400.000 lb.

The topology of the best solution obtained with the element-removal algorithm is shown in Figure 5.6. Table 5.7 shows the details of this best solution. The best solution obtained without the element-removal algorithm contains a large number of elements. Therefore, it is not shown.

Table 5.7 Problem 5.1.3: Best solution obtained with the element-removal algorithm.

Member	Area (in^2)[a]
1	20.963
2	20.963
3	11.267
4	11.267
5	6.988
6	6.988
7	20.156
8	20.156
Weight (lb)	2200.006
Max stress (ksi)	5.000
Max displacement (in)	0.550

[a]Symmetry is assumed.

Much faster convergences are observed when the element-removal algorithm is used, as demonstrated by Figure 5.7.

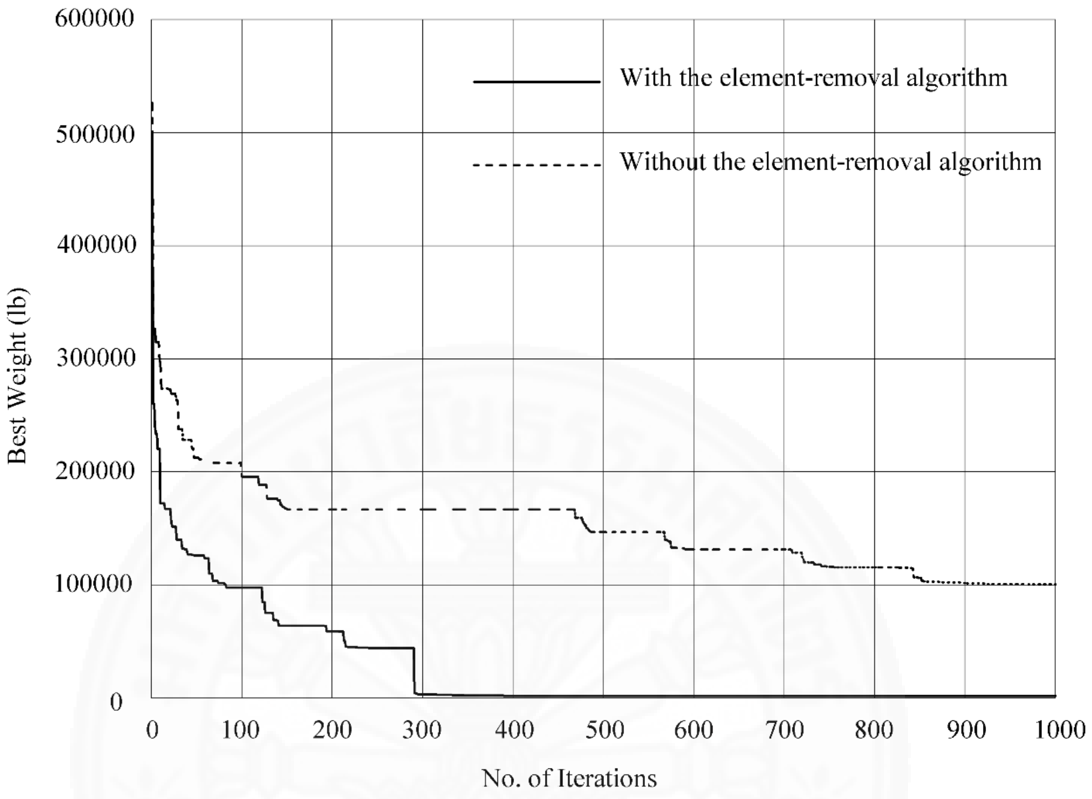


Figure 5.7 Problem 5.1.3: Typical convergences of the weights.

5.1.4 30381-element, 247-node truss with a reduced allowable displacement

In the previous problem, the maximum displacement of the best solution is equal to 0.550 in, which is smaller than the allowable displacement of 0.6 in. The maximum stress, however, is found to be equal to the allowable value. This means that the stress constraint is the critical constraint of the previous problem. In order to explore the situation where the displacement constraint is the only critical constraint, in this problem, the allowable displacement in the previous problem is reduced from 0.6 to 0.5 in. The other parameters are set to be the same as those used in the previous problem. The symmetry along the middle horizontal line of the solutions is also assumed.

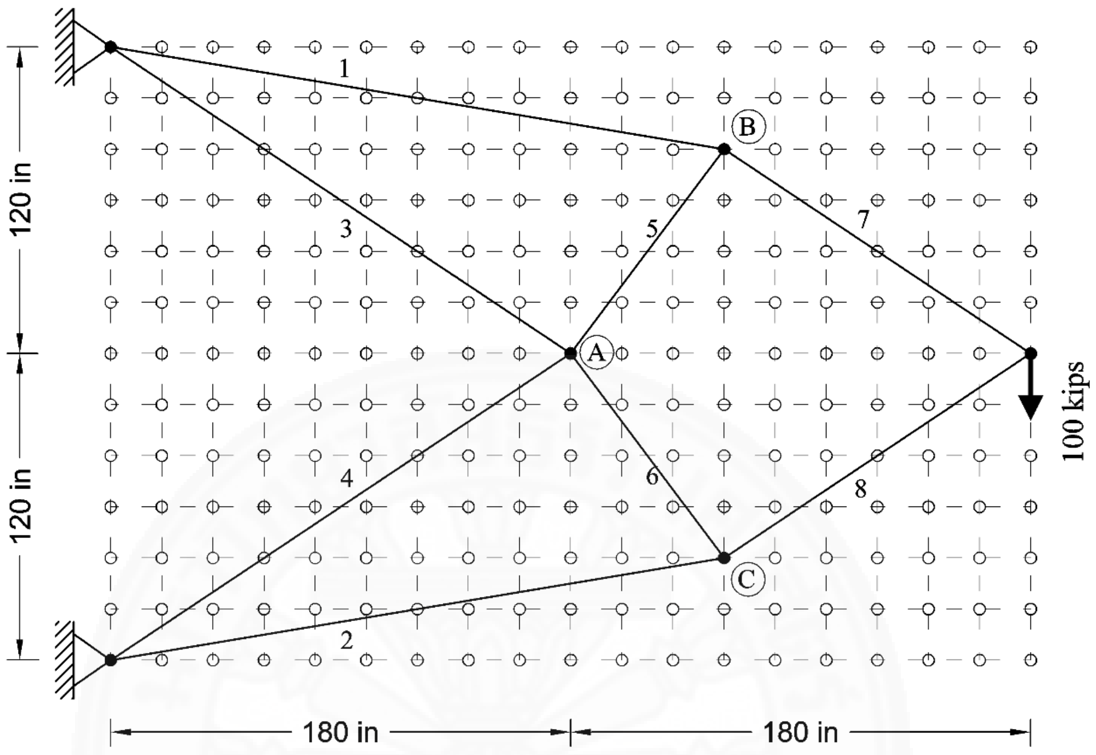


Figure 5.8 Problem 5.1.4: Problem details, ground-structure nodes, and best topology.

All run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.8 shows the statistics of the run solutions obtained with and without the element-removal algorithm. The minimum weight obtained with the element-removal algorithm is much lower than the one obtained without the algorithm. Again, the minimum weight obtained without the element-removal algorithm is very large. With the element-removal algorithm, the minimum weight obtained with the reduced allowable displacement is larger than that obtained with the allowable displacement of 0.6 in in the previous problem. This is expected because the new allowable displacement of 0.5 in is lower than the maximum displacement of the best solution obtained in the previous problem. The average and maximum weights obtained with the element-removal algorithm are also much lower than those obtained without the algorithm. The standard deviation of the weights obtained with the element-removal algorithm is slightly greater than that obtained without the algorithm. This is because, when the algorithm is not used, the run solutions consistently have high weights. On the contrary, when the algorithm is used, the weights of the run solutions are distributed over a large range, from low weights to high weights. Similar to the previous problem, the percentages of the run solutions that are not heavier than the best solution of the calculation set by more than 2%, obtained with and without the algorithm, are found to be quite low. The percentage obtained with the algorithm is in fact lower than that obtained without the algorithm. However, for this problem, the best solutions obtained with and without the element-removal algorithm are very different. Thus, the comparison is in fact not meaningful.

Table 5.8 Problem 5.1.4: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (lb)	2405.823	103321.059
Average weight (lb)	20798.427	162890.811
Maximum weight (lb)	133687.446	233046.017
SD of weights (lb)	25037.719	22334.623
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	0.1	0.2

The topology of the best solution obtained with the element-removal algorithm is shown in Figure 5.8. Table 5.9 shows the details of this best solution. The best solution obtained without the element-removal algorithm contains a large number of elements. Therefore, it is not shown. The topology in Figure 5.8 is very similar to the topology of the best solution of the previous problem in Figure 5.6. The only small differences are the positions of joints B and C of the two topologies. It can be seen from Table 5.9 that the maximum displacement of the best solution of this problem is equal to the allowable displacement while the maximum stress is lower than the allowable stress. Therefore, for this problem, the displacement constraint is the only critical constraint as expected.

Table 5.9 Problem 5.1.4: Best solution obtained with the element-removal algorithm.

Member	Area (in^2)[a]
1	21.971
2	21.971
3	13.227
4	13.227
5	9.482
6	9.482
7	19.926
8	19.926
Weight (lb)	2405.823
Max stress (ksi)	4.614
Max displacement (in)	0.500

[a]Symmetry is assumed.

Much faster convergences are observed when the element-removal algorithm is used, as demonstrated by Figure 5.9

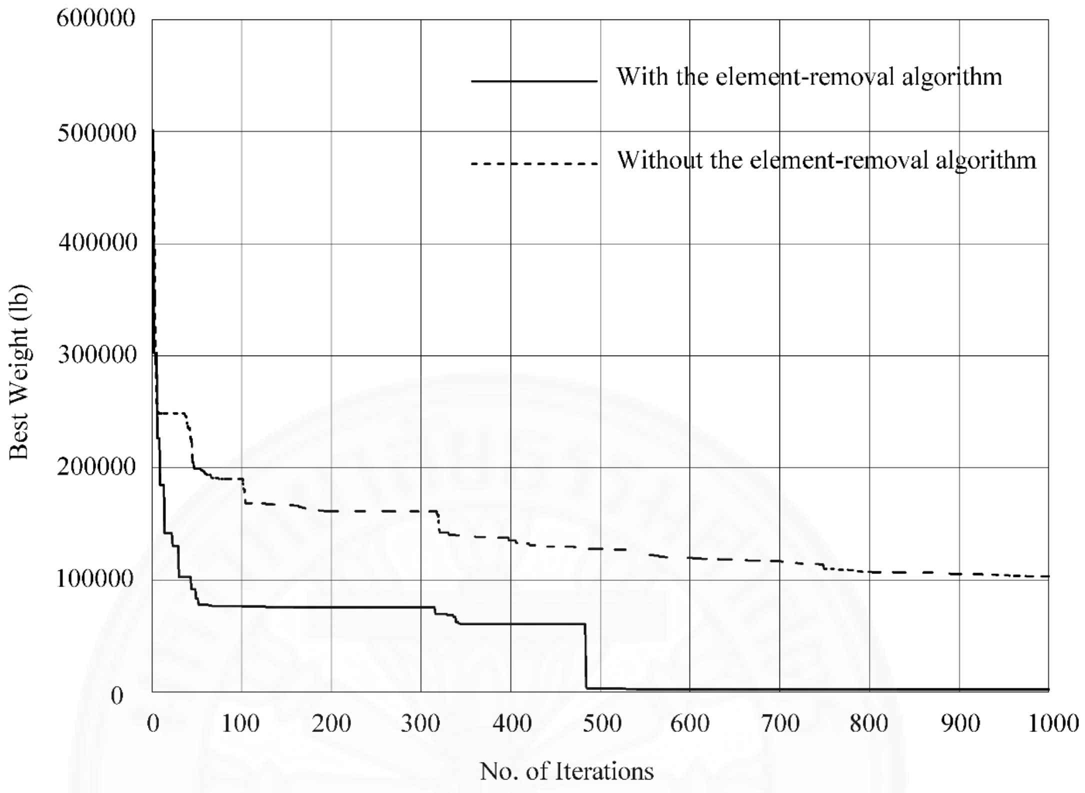


Figure 5.9 Problem 5.1.4: Typical convergences of the weights.

5.1.5 153-element, 18-node truss

The fifth problem is a 3D problem from a ground structure with 18 nodes. The ground structure is constructed from $3 \times 3 \times 2$ grid lines, shown in Figure 5.10. The ground structure nodes, force, supports, and problem dimensions are all shown in Figure 5.10. All 153 node connections are considered. The truss can be considered as a symmetrical truss, but in this study, the symmetry is not considered. The problem parameters are as follows:

Young's modulus: $E = 200$ GPa

Weight density: $\rho = 77.0\ \mathrm{kN/m^3}$

Allowable stress: $\sigma_a = 150$ MPa

Allowable displacement: $\delta_a = 20$ mm

Area: $A = 0$ to $2000\ \mathrm{mm}^2$

In PSO, the lower and upper boundaries of the design variables in all optimization dimensions are set, respectively, to -2000 and 2000. If the value of a design variable is non-positive, it is interpreted as a section of zero area. In addition, V_{dmx} is set to $[2000 - (-2000)]/4 = 1000$ for all optimization dimensions.

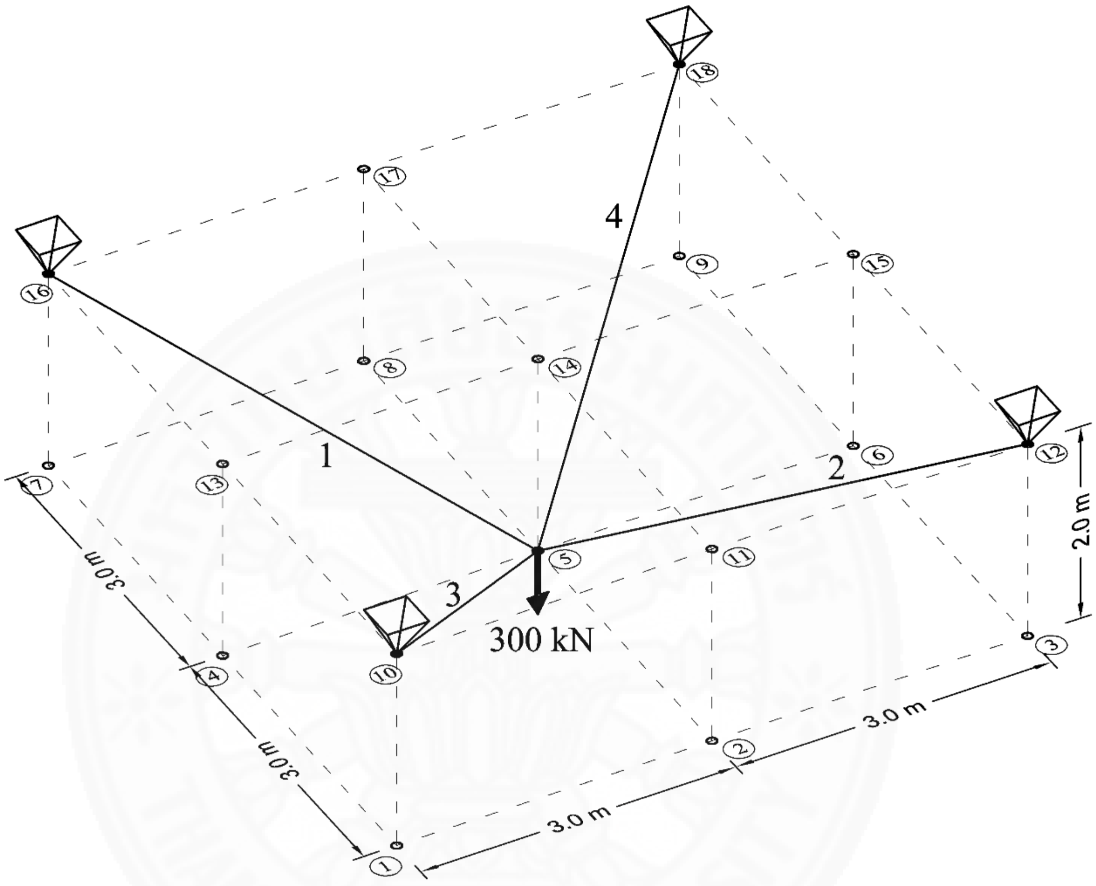


Figure 5.10 Problem 5.1.5: Problem details, ground-structure nodes, and best topology.

All run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.10 shows the statistics of the run solutions obtained with and without the element-removal algorithm. The minimum weight obtained with the element-removal algorithm is much lower than the one obtained without the algorithm. In fact, the minimum weight obtained with the element-removal algorithm is the exact optimal weight of this problem. The average weight and the standard deviation obtained with the element-removal algorithm are lower than those obtained without the algorithm. The maximum weight obtained with the algorithm is slightly higher than the one obtained without the algorithm. It can be seen from Table 5.10 that, when the element-removal algorithm is employed, 69.6% of the run solutions are not heavier than the best weight of the calculation set by more than 2%. On the contrary, when the element-removal algorithm is not used, only 0.2% of the run solutions are not heavier than the best weight of the calculation set by more than 2%. These results show clearly that the element-removal algorithm significantly improves the quality of the obtained results.

Table 5.10 Problem 5.1.5: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (N)	1694.000	1928.329
Average weight (N)	2194.763	3176.176
Maximum weight (N)	8976.507	8934.177
SD of weights (N)	986.153	1102.573
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	69.6	0.2

The topology of the best solution obtained with the element-removal algorithm is shown in Figure 5.10. For this problem, the best solutions with a weight of 1694.000 N, obtained with the element-removal algorithm, are not unique, regarding the individual member areas, although they all have the same topology, as shown in Figure 5.10. In each of these best solutions, the areas of members 1 and 2 are identical. The areas of members 3 and 4 are also identical. However, it is not necessary that all four members within each best solution have the same area. The topology of the best solution obtained without the element-removal algorithm is found to be complex and different from Figure 5.10.

Much faster convergences are again observed when the element-removal algorithm is used, as demonstrated by Figure 5.11.

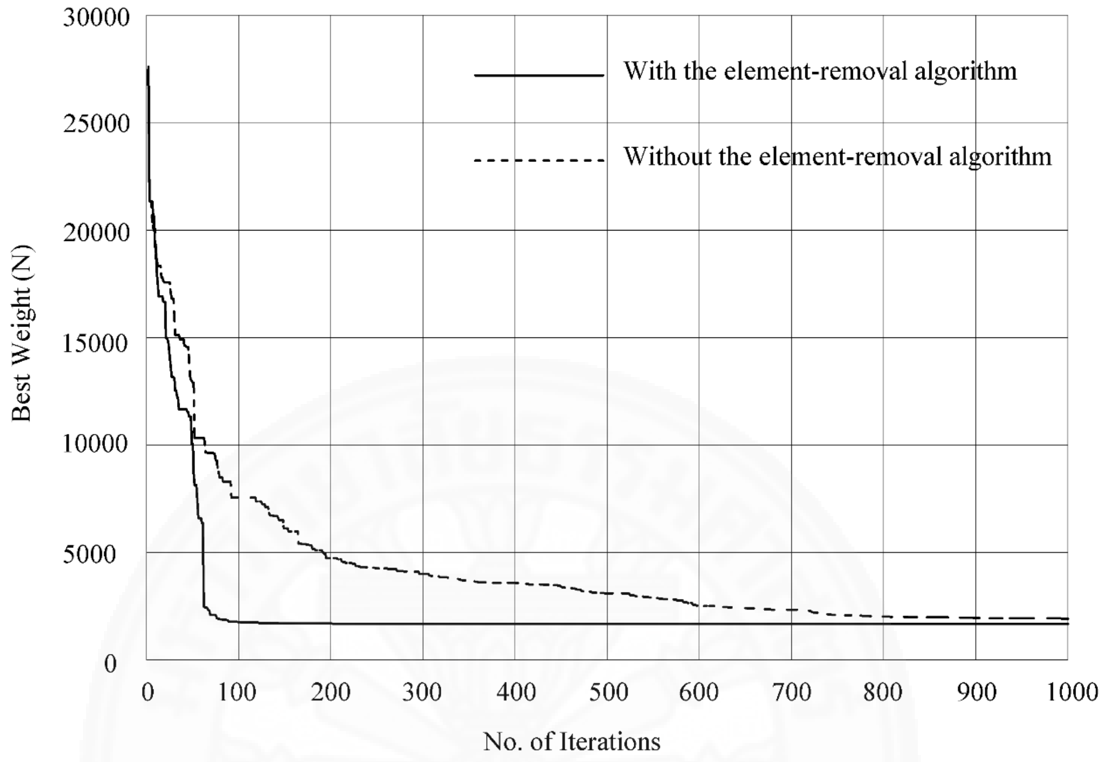


Figure 5.11 Problem 5.1.5: Typical convergences of the weights.

5.1.6 153-element, 18-node truss

The last problem is generated from problem 5.1.5 by making it unsymmetrical, as shown in Figure 5.12. All possible node connections are also considered in this problem. All problem and PSO parameters are the same as those of problem 5.1.5.

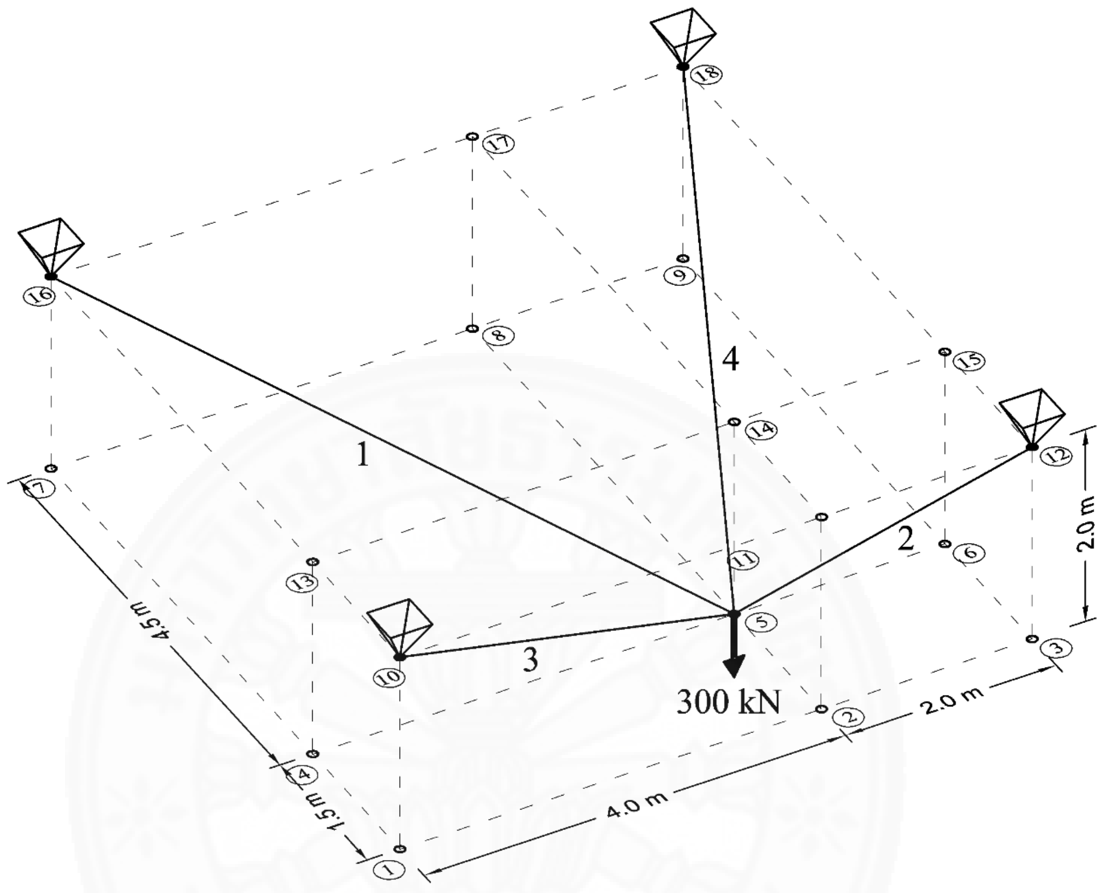


Figure 5.12 Problem 5.1.6: Problem details, ground-structure nodes, and best topology.

Again, it is found that all run solutions in the two calculation sets, with and without the element-removal algorithm, are feasible. Table 5.11 shows the statistics of the run solutions obtained with and without the element-removal algorithm. All results in the table, obtained with the algorithm, are considerably better than those obtained without the algorithm. The minimum weight obtained with the element-removal algorithm is the exact optimal weight of this problem. When the element-removal algorithm is used, the percentage of the run solutions that are not heavier than the best weight of the calculation set by greater than 2% is 68.0. When the element-removal algorithm is not used, the percentage of the run solutions that are not heavier than the best weight of the calculation set by more than 2% is only 0.4.

Similar to the previous problem, the best solutions with a weight of 1443.750 N, obtained with the element-removal algorithm, are not unique, regarding the individual member areas. Their topologies have four members, as shown in Figure 5.12, or three members, with member 1 in Figure 5.12 removed. The topology of the best solution obtained without the element-removal algorithm is found to be complex and different.

Table 5.11 Problem 5.1.6: Statistics of the run solutions.

	Calculation set of 1000 runs	
	With the element-removal algorithm	Without the element-removal algorithm
Minimum weight (N)	1443.750	1712.682
Average weight (N)	1837.071	2811.178
Maximum weight (N)	7734.456	9727.629
SD of weights (N)	825.887	956.491
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	68.0	0.4

Similar to the other problems, faster convergences are observed when the element-removal algorithm is used, as demonstrated by Figure 5.13.

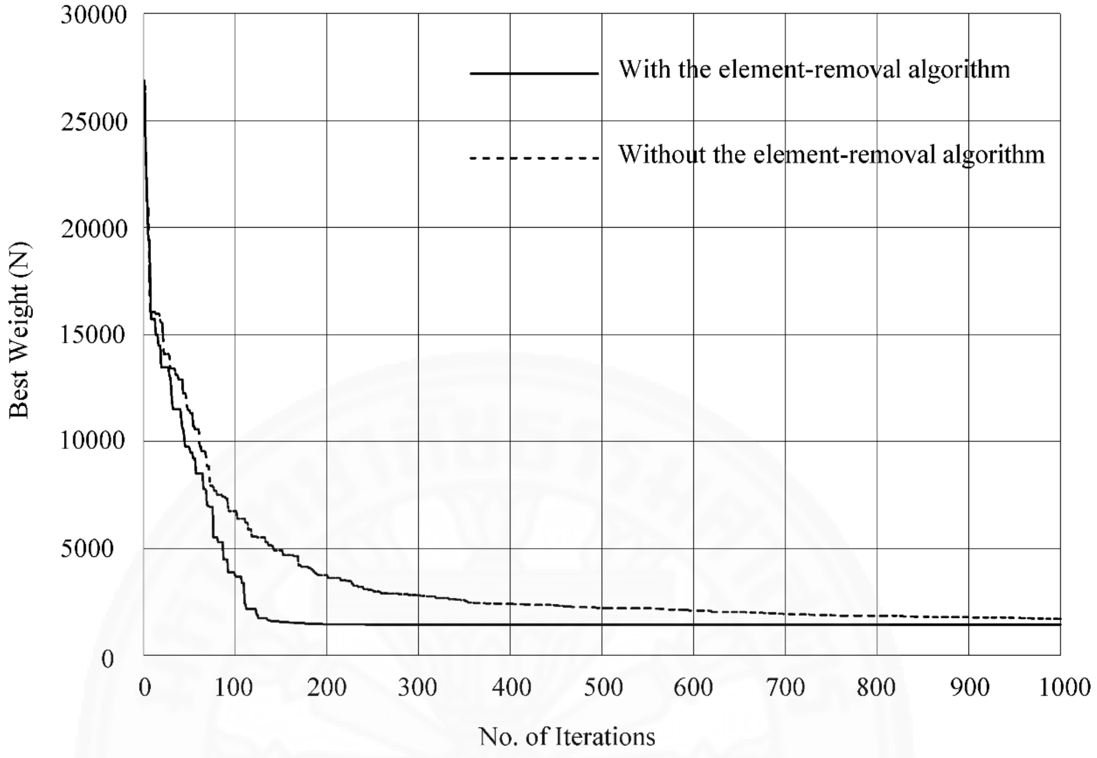


Figure 5.13 Problem 5.1.6: Typical convergences of the weights.

5.2 Truss topology optimization with a prescribed maximum number of elements

To demonstrate the effectiveness of the proposed truss optimization algorithm with a prescribed maximum number of elements, it is used to solve four truss topology optimization problems. The problems are also solved using the ground structure approach and the obtained results are compared. In both approaches, the two-population PSO algorithm is again used. The obtained optimal weights are also compared with some existing studies, wherever possible. As aforementioned, the comparison with the literature is done only to make certain that the results obtained from this study are satisfactory. It is not in the scope of this study to discuss the performance of the whole optimization process, which includes the performance of the two-population PSO algorithm. The element-removal algorithm developed in section 4.2 is also used while solving the problems in this section.

When the proposed algorithm is used, the value of the minimum area factor, $A_{minFactor}$, is taken as 8 for all the examples presented in this section. The value of the minimum area factor of 8 is found, by trial and error, to consistently give excellent results when the proposed algorithm is employed. On the contrary, the minimum area factor is taken as 1 for all the examples when the ground structure approach is used. In the proposed algorithm, the maximum number of elements is generally assumed not to be significantly larger than the number of elements to be remained in the optimal truss. As a result, the probability of an element being absent can be reduced from those used in the ground structure approach. As mentioned earlier, V_{dmx} for each optimization dimension is set to one quarter of the range of all possible codes for that dimension.

5.2.1 15-element, 6-node truss

The first problem considered is a simple 2D problem studied by Deb and Gulati (2001) and Wu and Tseng (2010). The forces, supports, and problem dimensions are shown in Figure 5.14. When the problem is solved by the ground structure approach, the ground structure in Figure 5.14, which has 6 nodes and 15 elements, is used and the design variables are the areas of all the elements present in the ground structure. When the problem is solved by the proposed approach, the six nodes of the ground structure in Figure 5.14 are used as the probable positions of the truss joints. This original grid of nodes in Figure 5.14 is mapped into a substitute grid with uniform unit spacing as stated in section 4.3.

The problem parameters are given as follows (Deb & Gulati, 2001; Wu & Tseng, 2010):

Young's modulus: $E = 10^4$ ksi

Weight density: $\rho = 0.1\ \mathrm{lb/in^3}$

Allowable stress: $\sigma_a = 25$ ksi

Allowable displacement: $\delta_a = 2$ in

Area: $A = 0.09$ to $35\ \mathrm{in^2}$

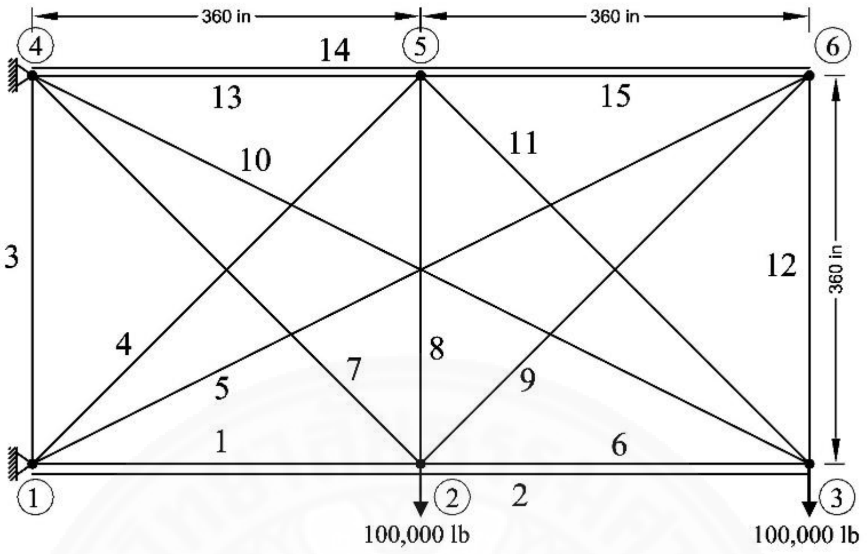


Figure 5.14 Problem 5.2.1: Problem details, ground-structure nodes and elements.

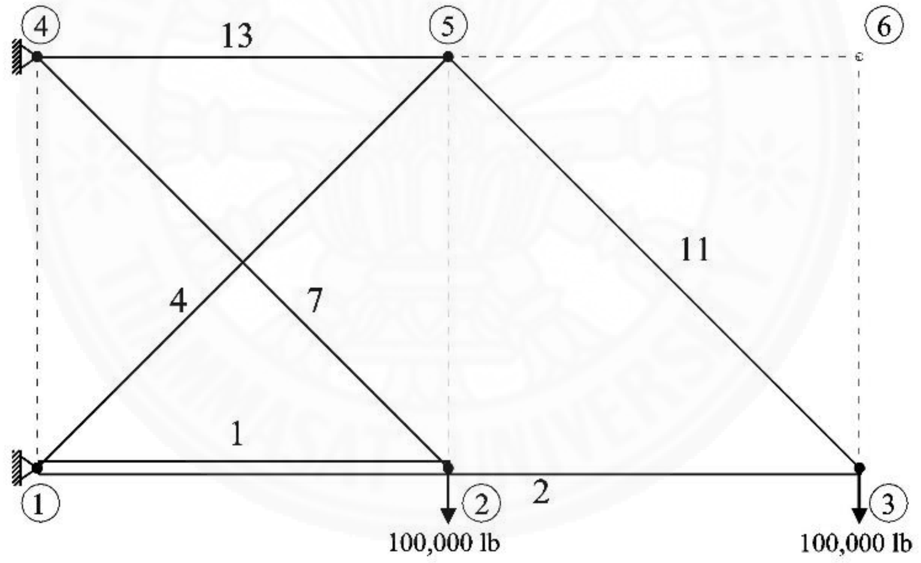


Figure 5.15 Problem 5.2.1: Best topology.

In the proposed algorithm, different values for the maximum number of elements are tried, and the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -4.375 and 35 in^2. In addition, V_{dmx} is set to $[35-(-4.375)]/4 = 9.844$ for all optimization dimensions. When the ground structure approach is used, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -35 and 35 in^2, and V_{dmx} is set to $[35-(-35)]/4 = 17.5$ for all optimization dimensions.

Table 5.12 Problem 5.2.1: Statistics of the run solutions.

	Calculation set of 1000 runs				
	Ground structure approach	Proposed algorithm			
The maximum number of elements	–	6	8	10	12
Minimum weight (lb)	4730.530	4730.553	4730.499	4730.507	4730.413
Average weight (lb)	5234.861	5841.331	5339.962	5073.561	5002.485
Maximum weight (lb)	7416.295	6885.384	7153.192	7114.694	6621.327
SD of weights (lb)	639.671	379.088	575.636	446.956	362.066
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	37.6	4.1	23.7	33.4	36.4

Table 5.12 shows the results obtained with the ground structure approach and the proposed approach. All 1000 run solutions are found to be feasible for all the cases in Table 5.12 except for the case when the maximum number of elements is equal to 6, in which some run solutions are found to be infeasible. The best weights obtained from the ground structure and proposed approaches are comparable. The average weights from the proposed algorithm with the maximum number of elements equal to 10 and 12 are better than that obtained by the ground structure approach. All the maximum weights and the standard deviations of the weights from the proposed algorithm are

better than those obtained by the ground structure approach. All the best solutions from the ground structure approach and the proposed approach are found to have the same topology that is shown in Figure 5.15. Table 5.12 also shows the percentages of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2%. For this performance indicator, the ground structure approach seems to perform better than the proposed algorithm. The percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% obtained by the proposed algorithm increases when the maximum number of elements increases. When the maximum number of elements is equal to 12, the obtained percentage is comparable to that obtained by the ground structure approach.

The minimum weights given by Deb and Gulati (2001) and Wu and Tseng (2010) are 4731.65 lb and 4730.68 lb, respectively, which are comparable with the present results. The best topologies obtained by Deb and Gulati (2001) and by Wu and Tseng (2010) are exactly the same as that obtained from the proposed algorithm shown in Figure 5.15. Table 5.13 shows the details of the best solutions from this study, with the maximum number of elements equal to 12, and from Deb and Gulati (2001) and Wu and Tseng (2010). The results by Deb and Gulati (2001) and Wu and Tseng (2010) are presented as originally reported. Note that there are 6 elements in the best topology in Figure 5.15. It follows that the optimal topology in Figure 5.15 cannot be obtained if the maximum number of elements is set to be less than 6.

Table 5.13 Problem 5.2.1: Best solutions.

Member	Area (in^2)			
	This study		Deb and Gulati (2001)	Wu and Tseng (2010)
	Ground structure approach	Proposed algorithm with the maximum number of elements = 12		
1	5.315	5.377	5.219	5.533
2	14.363	14.375	14.593	14.342
4	20.321	20.407	20.310	20.281
7	7.698	7.653	7.772	7.545
11	20.338	20.363	20.650	20.356
13	28.974	28.792	28.817	29.046
Weight (lb)	4730.530	4730.413	4731.650	4730.68
Max stress (ksi)	18.814	18.597	19.161	18.701
Max displacement (in)	2.000	2.000	2.000	2.000

5.2.2 45-element, 10-node truss

The second problem is the same 2D truss optimization problem considered in section 5.1.1. When the problem is solved using the ground structure approach, all possible node connections are considered. When the problem is solved by the proposed approach, the ten nodes of the ground structure are considered as the probable position of the truss joints. Symmetry is considered along the middle vertical line while solving the problem using both approaches.

When the proposed approach is used, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -0.125 and 1 in^2, and V_{dmx} is set to $[1-(-0.125)]/4 = 0.281$ for all optimization dimensions. When the ground structure approach is used, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -1 and 1 in^2, and V_{dmx} is set to $[1-(-1)]/4 = 0.5$ for all optimization dimensions.

Table 5.14 shows the results obtained with the ground structure approach and the proposed approach. The result obtained with the ground structure approach is the same as in Table 5.2. It is reported here again for comparison purpose only. All 1000 run solutions are found to be feasible for all the cases in Table 5.14 except for the case when the maximum number of elements is equal to 6, in which some run solutions are found to be infeasible. Although not shown here, when the maximum number of elements is specified less than 6, all the run solutions are found to be infeasible. This is expected since at least 6 elements are required to create a stable truss.

Table 5.14 Problem 5.2.2: Statistics of the run solutions.

	Calculation set of 1000 runs					
	Ground structure approach	Proposed algorithm				
The maximum number of elements	–	6	8	10	20	30
Minimum weight (lb)	44.000	44.000	44.000	44.000	44.000	44.000
Average weight (lb)	45.851	45.586	45.150	44.753	44.142	44.183
Maximum weight (lb)	103.685	67.871	52.000	52.000	52.000	52.000
SD of weights (lb)	4.994	3.246	2.798	2.328	1.043	1.142
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	78.9	80.1	85.5	90.5	98.2	97.2

The best weights obtained from the ground structure and proposed approaches are the same for all the cases as shown in Table 5.14. Figure 5.1 shows the best topology of the truss obtained using the ground structure approach and the proposed algorithm, which contains 7 elements. This topology is the same as the best topologies given by Deb and Gulati (2001) and Wu and Tseng (2010). Table 5.14 clearly shows that the proposed approach gives the results that are more consistent than those of the ground structure approach. The standard deviations of the weights obtained with the proposed approach are considerably lower than that obtained with the ground structure approach. Also, when the proposed approach is used, the numbers of run solutions that are not heavier than the best solution of the calculation set by more than 2% are found to be considerably high. The minimum weights reported by Deb and Gulati (2001) and Wu and Tseng (2010) are 44.033 lb and 43.99 lb, respectively. The details of the best solutions from the proposed approach when the maximum number of elements is set to 8, 10, 20, and 30 are exactly the same as those obtained in this study using the ground structure approach, which is shown in Table 5.3. The same table also shows the details of the best solutions by Deb and Gulati (2001) and Wu and Tseng (2010). When the maximum number of elements is set to 6, the optimized topology is as shown in Figure 5.16.

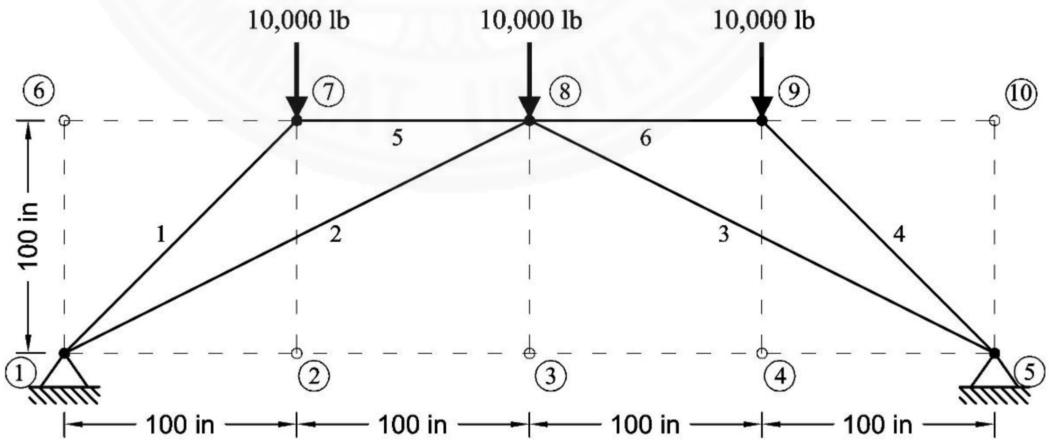


Figure 5.16 Problem 5.2.2: Best topology with 6 elements.

Table 5.15 shows the details of the best solution from the proposed algorithm when the maximum number of elements is set to 6.

Table 5.15 Problem 5.2.2: Best solution obtained with 6 elements.

Member	Area (in^2)[a]
1	0.566
2	0.447
3	0.447
4	0.566
5	0.4
6	0.4
7	$-$
Weight (lb)	44.000
Max stress (ksi)	25.000
Max displacement (in)	1.250

5.2.3 66-element, 12-node truss

The third problem is from Faramarzi and Afshar (2012). The ground structure used by Faramarzi and Afshar (2012), which has 12 nodes and 66 elements, is also used in this study. The problem dimensions, boundary conditions and ground-structure nodes and elements are shown in Figure 5.17. When the problem is solved by the proposed approach, the 12 nodes of the ground structure are used as the probable positions of the truss joints. Since the behavior of truss elements in tension and compression is considered the same, the symmetry along the middle horizontal line of the solutions is assumed when the problem is solved by both approaches.

The problem parameters are also taken from the work by Faramarzi and Afshar (2012), which are as follows:

Young's modulus: $E = 10^4$ ksi

Weight density: $\rho = 0.1\ \text{lb/in}^3$

Allowable stress: $\sigma_a = 5$ ksi

Allowable displacement: $\delta_a = 0.6$ in

Area: $A = 0.09$ to $35\ \text{in}^2$

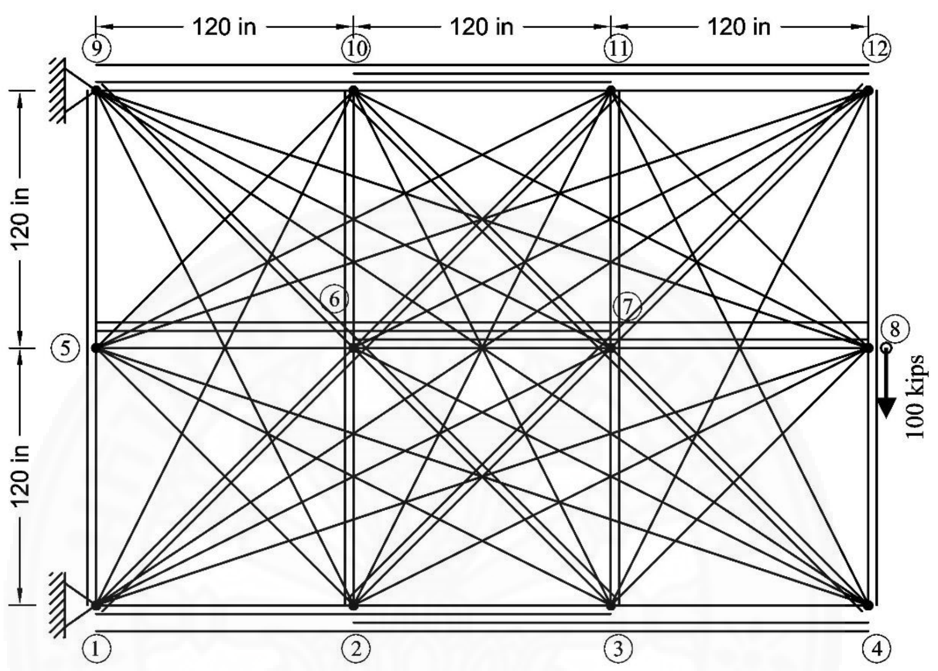
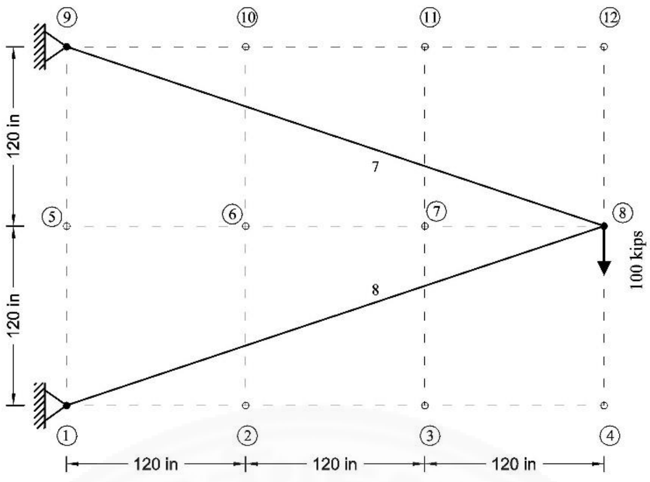
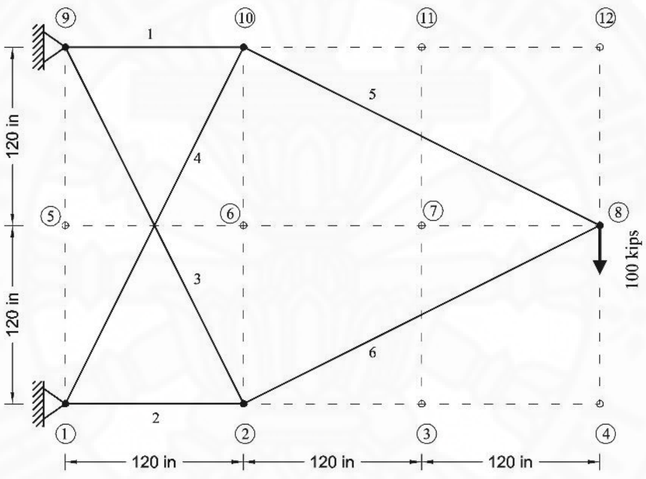


Figure 5.17 Problem 5.2.3: Problem details, ground-structure nodes and elements.

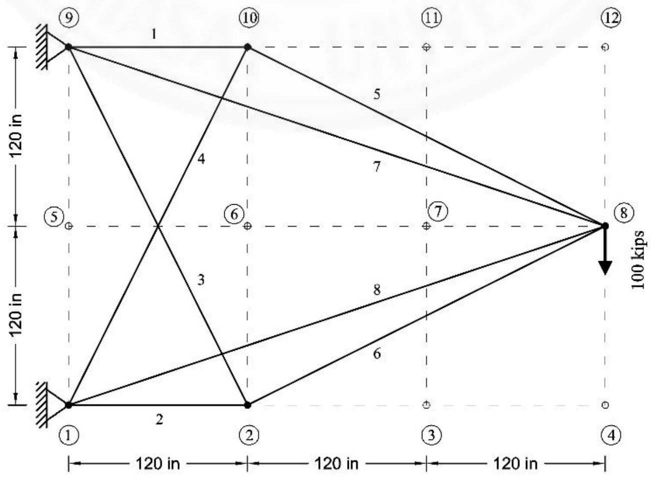
When the problem is solved using the proposed approach, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -4.375 and $35\ \text{in}^2$, and V_{dmx} is set to $[35-(-4.375)]/4 = 9.844$ for all optimization dimensions. When the ground structure approach is used, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -35 and $35\ \text{in}^2$, and V_{dmx} is set to $[35-(-35)]/4 = 17.5$ for all optimization dimensions.



(a)



(b)



(c)

Figure 5.18 Problem 5.2.3: Example best topologies: (a) maximum number of elements ≥ 2; (b) maximum number of elements ≥ 6; (c) maximum number of elements ≥ 8.

Table 5.16 shows the results obtained with the ground structure approach and the proposed approach. All 1000 run solutions are found to be feasible for all the cases in Table 5.16. In the proposed approach, the value of the maximum number of elements is varied from 2 to 66. Note that 66 is the total number of elements that are possible in the ground structure. When the value of the maximum number of elements is from 2 to 30, all run solutions in each calculation set have the same weight of 2,400 lb. This fact can be clearly seen from the values of the standard deviation of zero for these cases, shown in Table 5.16. It also follows that, for these cases, the percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% is 100. The results are slightly changed when the values of the maximum number of elements is set to 40 and 50. As seen from Table 5.16, the best weights are still the same and the other statistics are still better than those obtained by the ground structure approach. When the maximum number of elements is set to 66, the best weight is still the same. However, the other statistics are no longer better than those given by the ground structure approach. For this problem, several optimized topologies having the same minimum weight are obtained from both approaches. Three example best topologies are shown in Figure 5.18.

Table 5.16 Problem 5.2.3: Statistics of the run solutions.

	Calculation set of 1000 runs				
	Ground structure approach	Proposed algorithm			
The maximum number of elements	–	2, 4, 6, 8, 10, 20, 30	40	50	66
Minimum weight (lb)	2400.000	2400.000	2400.000	2400.000	2400.000
Average weight (lb)	2434.195	2400.000	2404.235	2415.563	2458.558
Maximum weight (lb)	3463.753	2400.000	3134.283	3262.440	4006.070
SD of weights (lb)	124.191	0	46.117	78.742	152.579
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	90.7	100	99.1	95.4	82

Table 5.17 shows the details of the element areas of the optimized topologies shown in Figure 5.18 (a) and Figure 5.18 (b). The element combinations for the optimized topology in Figure 5.18 (c) are not unique. As a result, the details of the element areas for this topology are not shown. The minimum weight obtained by Faramarzi and Afshar (2012) for this problem is 2400.001 lb. The optimized topology given by Faramarzi and Afshar (2012) is the same as that shown in Figure 5.18 (c). The details of the best solution from Faramarzi and Afshar (2012) are also listed in Table 5.17.

Table 5.17 Problem 5.2.3: Best solutions.

Member	Area (in^2)		
	This study[a] Ground structure and proposed approaches		Faramarzi and Afshar (2012)[b]
	Figure 5.18 (a)	Figure 5.18 (b)	
1	–	25.000	20.000
2	–	25.000	7.0263
3	–	11.180	3.1423
4	–	11.180	8.9442
5	–	22.361	17.8885
6	–	22.361	6.2845
7	31.623	–	9.0597
8	31.623	–	20.000
Weight (lb)	2400.000	2400.000	2400.001
Max stress (ksi)	5.000	5.000	5.000
Max displacement (in)	0.60	0.60	0.5999

[a]Symmetry is assumed.

[b]Symmetry is not assumed.

This problem is also solved with a finer grid in this study. As shown in Figure 5.19, the number of nodes in the finer grid is equal to 35, which results in 595 possible elements in the ground structure. All other problem parameters are kept unchanged. When solving this problem with the proposed approach, the nodes of the ground structure in Figure 5.19 are used as the probable positions of the truss joints. In addition, the value of the maximum number of elements is set to 100. Table 5.18 shows the results obtained with the ground structure approach and the proposed approach for the problem with 35 nodes. All 1000 run solutions are found to be feasible for all the cases in Table 5.18. The optimized topology obtained with the proposed approach is the same as that shown in Figure 5.19, which contains 8 elements. From the proposed approach, the minimum weight obtained for this 35-node problem is 2232.266 lb, which is lower than the one obtained for the 12-node problem. This is generally expected since finer grids provide more choices of topologies than rougher grids. The exceptions are those cases where the locations of nodes in finer grids cannot represent the optimal solutions while those in rougher grids can. For this problem, the best weight obtained with the ground structure approach is 2449.642 lb, which is greater than the best weight obtained with the proposed approach. The average weight obtained with the proposed approach is lower than that obtained with the ground structure approach as also shown in Table 5.18. The maximum weight and the standard deviation of the weights obtained with the proposed approach are found to be slightly higher than those obtained with the ground structure approach. When the ground structure approach and the proposed approach are used, the percentages of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% are 0.2 and 1.3, respectively. The topology of the best solution obtained with the ground structure approach is found to be complex, with 22 elements, and different from Figure 5.19.

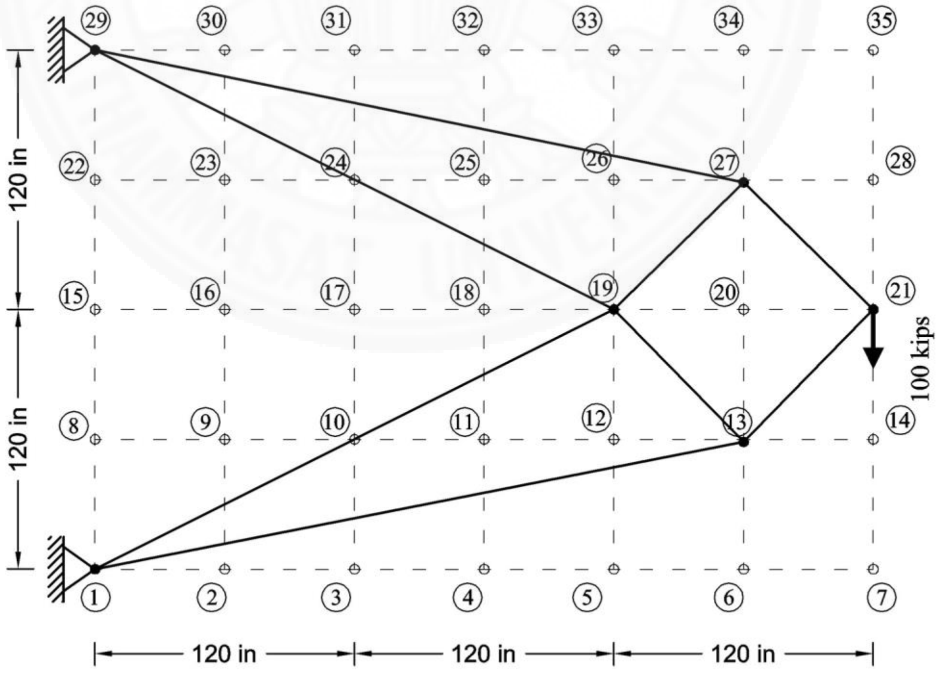


Figure 5.19 Problem 5.2.3: Problem details, ground-structure nodes, and best topology with a finer grid.

Table 5.18 Problem 5.2.3: Statistics of the run solutions for the problem with 35 nodes.

	Calculation set of 1000 runs	
	Ground structure approach	Proposed algorithm
The maximum number of elements	−	100
Minimum weight (lb)	2449.642	2232.266
Average weight (lb)	4294.971	2866.186
Maximum weight (lb)	6844.925	9042.204
SD of weights (lb)	722.519	759.233
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	0.2	1.3

5.2.4 153-element, 18-node truss

The last problem considered here is a 3D problem with 18 nodes and 153 elements from problem 5.1.5, and the $3 \times 3 \times 2$ grid used is shown in Figure 5.10. Here, the symmetry of the truss is not considered. When solving the problem using the ground structure approach, all 153 node connections are considered. When the problem is solved by the proposed approach, the 18 ground-structure nodes are considered as the probable position of the truss joints. All the problem parameters considered for this problem are also taken from problem 5.1.5.

When solving the problem using the proposed approach, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -250 and 2000 mm^2. The value of V_{dmx} is set to $[2000-(-250)]/4=562.5$ for all optimization dimensions. When the ground structure approach is used, the lower and upper boundaries of the cross-sectional area in all optimization dimensions are set, respectively, to -2000 and 2000 mm^2, and V_{dmx} is set to $[2000-(-2000)]/4=1000$ for all optimization dimensions.

The results obtained with the ground structure approach and the proposed approach are shown in Table 5.19. All 1000 run solutions are found to be feasible for all the cases in Table 5.19. The result obtained with the ground structure approach is the same as in Table 5.10. It is reported here again for comparison purpose only. The best solutions from the two approaches have the same topology, shown in Figure 5.10, and the same minimum weight of 1694 kN. Again, the best solutions are not unique, regarding the individual member areas although they all have the same topology as shown in Figure 5.10. When the values of the maximum number of elements are 20 to 40, all run solutions in each calculation set have the same weight of 1694 kN. As a result, the values of the standard deviation is equal to zero for these cases, as shown in Table 5.19. It follows that, for these cases, the percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% is 100. When the maximum number of elements is set to 50 and 60, the results are slightly changed. The best weights are still the same and the other statistics are still better than those obtained by the ground structure approach. When the maximum number of elements is set to 100, the best weight is still the same. However, the other statistics are no longer better than those given by the ground structure approach. It can be seen from Table 5.19 that, the percentage of the run solutions that are not heavier than the best weight of the calculation set by more than 2% is equal to 69.6 when the ground structure approach is used. When the proposed approach is used, this value is always higher than that of the ground structure approach, except when the maximum number of elements is set to 100, as shown in Table 5.19. These results clearly show that the proposed approach generally gives better results than the ground structure approach.

Table 5.19 Problem 5.2.4: Statistics of the run solutions.

	Calculation set of 1000 runs				
	Ground structure approach	Proposed algorithm			
The maximum number of elements	–	20, 30, 40	50	60	100
Minimum weight (lb)	1694.000	1694.000	1694.000	1694.000	1694.000
Average weight (lb)	2194.763	1694.000	1694.073	1706.473	2650.065
Maximum weight (lb)	8976.507	1694.000	1757.451	4381.400	7383.833
SD of weights (lb)	986.153	0	2.011	148.188	1227.485
Percentage of the run solutions that are not heavier than the minimum weight of the calculation set by greater than 2% (%)	69.6	100	99.9	98.8	23.5

CONCLUSIONS

In the first part of the study, a new representation for truss topology optimization is proposed. The proposed representation uses an element-removal algorithm to remove unwanted elements from trusses to obtain final trusses. These unwanted elements include kinematically unstable elements and useless zero-force elements. The element-removal algorithm does not consider all patterns of elements that are kinematically unstable or have no force. Rather, it considers only those patterns that can be easily identified without much detailed checking. The element-removal algorithm is used in the translation of representation codes into corresponding trusses. As a result, more representation codes in the search space are mapped into kinematically stable and efficient trusses, and the level of competition among representation codes is increased. Six example problems are solved, and the results are discussed. It has been found that the proposed coding scheme makes the optimization process more effective and efficient.

In the second part of the study, a new coding scheme is proposed in which a prescribed maximum number of elements is specified for truss optimization. The main advantage of this proposed method is that, for each truss topology optimization problem, a dense ground structure is not used. As a result, the number of elements that must be removed during the optimization process to obtain the optimal truss is reduced. Another advantage of the proposed scheme is that the designer can predefine a maximum limit for the number of members to be present in the final design. This type of provision is generally not possible when the ground structure approach is used. The performance of the proposed scheme is found to depend upon the prescribed value of the maximum number of elements. It can be concluded from the examples solved in this study that the proper value for the maximum number of elements depends upon the total number of elements in the ground structure and the number of elements in the real optimal solution. Since the number of elements in the real optimal solution is not known, values that are larger than those that may be used in real practice for the considered truss may be selected. However, the selected value should not be set too large since it will result in a large search space and the employed optimization algorithm may not be able to effectively solve the optimization problem. It can be seen from the results of the examples solved in this study that, when a denser ground structure is employed, a smaller ratio of the prescribed maximum number of elements to the number of elements in the ground structure has to be used in order that good results can be obtained. Some of the obtained results obtained from both proposed coding schemes are compared with the existing results found in the literature. Good agreement is observed.

In this research, the proposed algorithms are used only with continuous variables. Further researches can be done by applying the proposed algorithms to solve the problems with discrete variables. Another limitation of this study is that the proposed algorithms are implemented using only two-population PSO. The algorithms can be further implemented with different optimization algorithms, such as simple PSO, GAs, and FAs. Moreover, in this study, only stresses and displacements are considered as design constraints. Further studies can be done by considering other additional constraints.

REFERENCES

Arnaot, F. H. (2019). Effect of loading on optimum weight of planer trusses using genetic algorithms. *Proceeding of 2019 2nd International Conference on Engineering Technology and its Applications, IICETA 2019* (pp. 37-42).

Assimi, H., Jamali, A., & Nariman-zadeh, N. (2017). Sizing and topology optimization of truss structures using genetic programming. *Swarm and Evolutionary Computation, 37*, 90-103. 10.1016/j.swevo.2017.05.009

Baghlani, A., Makiabadi, M. H., & Rahnema, H. (2013). A new accelerated firefly algorithm for size optimization of truss structures. *Scientia Iranica, 20*(6), 1612-1625.

Bansal, J. C., Singh, P. K., Saraswat, M., Verma, A., Jadon, S. S., & Abraham, A. (2011). Inertia weight strategies in particle swarm optimization. *Proceeding of Proceedings of the 2011 3rd World Congress on Nature and Biologically Inspired Computing, NaBIC 2011* (pp. 633-640).

Beckers, M., & Fleury, C. (1997). A primal-dual approach in truss topology optimization. *Computers and Structures, 64*(1-4), 77-88.

Bołbotowski, K., & Sokół, T. (2016). New method of generating strut and tie models using truss topology optimization. *Proceeding of Advances in Mechanics: Theoretical, Computational and Interdisciplinary Issues - 3rd Polish Congress of Mechanics, PCM 2015 and 21st International Conference on Computer Methods in Mechanics, CMM 2015* (pp. 97-100).

Choensiridamrong, C., Watjatrakul, B., & Prayote, A. (2014). A simultaneous topology and sizing optimization for plane trusses. *Proceeding of 2014 11th Int. Joint Conf. on Computer Science and Software Engineering: "Human Factors in Computer Science and Software Engineering" - e-Science and High Performance Computing: eHPC, JCSSE 2014* (pp. 111-116).

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation, 6*(1), 58-73. 10.1109/4235.985692

Cui, H., An, H., & Huang, H. (2018). Truss topology optimization considering local buckling constraints and restrictions on intersection and overlap of bar members. *Structural and Multidisciplinary Optimization*, *58*(2), 575-594. 10.1007/s00158-018-1910-x

Deb, K., & Gulati, S. (2001). Design of truss-structures for minimum weight using genetic algorithms. *Finite Elements in Analysis and Design*, *37*(5), 447-465. 10.1016/S0168-874X(00)00057-3

Eberhart, R. C., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. *Proceeding of Proceedings of the 2000 Congress on Evolutionary Computation, CEC 2000* (pp. 84-88).

Ede, A. N., Oshokoya, O. O., Oluwafemi, J. O., Oyebisi, S. O., Olofinnade, O. M., & Akpabot, A. I. (2018). Weight optimization of square hollow steel trusses using genetic algorithm. *Proceeding of IOP Conference Series: Materials Science and Engineering* (pp.

Faramarzi, A., & Afshar, M. H. (2012). Application of cellular automata to size and topology optimization of truss structures. *Scientia Iranica*, *19*(3), 373-380. 10.1016/j.scient.2012.04.009

Gao, G., Liu, Z. Y., Li, Y. B., & Qiao, Y. F. (2016). A new method to generate the ground structure in truss topology optimization. *Engineering Optimization*, 1-17. 10.1080/0305215X.2016.1169050

Gao, Y., An, X., & Liu, J. (2008). A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. *Proceeding of 2008 International Conference on Computational Intelligence and Security* (pp. 61-65).

Ge, Y., & Rubo, Z. (2005). An emotional particle swarm optimization algorithm. *Proceeding of Lecture Notes in Computer Science* (pp. 553-561).

Ghoddosian, A., Riyahi Vezvari, M., Sheikhi Azqandi, M., & Karimi, M. A. (2018). Topology optimisation of the discrete structures with the minimum growing ground structure method. *International Journal of Structural Engineering*, 9, 38. 10.1504/IJSTRUCTE.2018.090749

Gilbert, M., & Tyas, A. (2003). Layout optimization of large-scale pin-jointed frames. *Engineering Computations (Swansea, Wales)*, *20*(7-8), 1044-1064.

Guo, X., Cheng, G. D., & Olhoff, N. (2005). Optimum design of truss topology under buckling constraints. *Structural and Multidisciplinary Optimization*, *30*(3), 169-180. 10.1007/s00158-004-0511-z

Hagishita, T., & Ohsaki, M. (2009). Topology optimization of trusses by growing ground structure method. *Structural and Multidisciplinary Optimization*, *37*(4), 377-393. 10.1007/s00158-008-0237-4

Hajela, P., & Lee, E. (1995). Genetic algorithms in truss topological optimization. *International Journal of Solids and Structures*, *32*(22), 3341-3357. 10.1016/0020-7683(94)00306-H

Han, L., & Wang, Y. (2019). Topology optimization of top lateral bracing for steel tub girder systems using genetic algorithm. *Proceeding of Structural Stability Research Council Annual Stability Conference 2019, SSRC 2019* (pp. 51-58).

Hasançebi, O., & Erbatur, F. (2002). Layout optimisation of trusses using simulated annealing. *Advances in Engineering Software*, *33*(7-10), 681-696. 10.1016/S0965-9978(02)00049-2

Hossen, S., Rabbi, F., & Rahman, M. (2009). Adaptive particle swarm optimization (apso) for multimodal function optimization. *International Journal of Engineering and Technology, 1*(3), 98-103.

Jiang, Y., Hu, T., Huang, C., & Wu, X. (2007). An improved particle swarm optimization algorithm. *Applied Mathematics and Computation*, *193*(1), 231-239. 10.1016/j.amc.2007.03.047

Kaveh, A., & Kalatjari, V. (2003). Topology optimization of trusses using genetic algorithm, force method and graph theory. *International Journal for Numerical Methods in Engineering*, *58*(5), 771-791. 10.1002/nme.800

Kawamura, H., Ohmori, H., & Kito, N. (2002). Truss topology optimization by a modified genetic algorithm. *Structural and Multidisciplinary Optimization*, *23*(6), 467-472. 10.1007/s00158-002-0208-0

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceeding of IEEE International Conference on Neural Networks - Conference Proceedings* (pp. 1942-1948).

Kennedy, J., & Eberhart, R. C. (1997). Discrete binary version of the particle swarm algorithm. *Proceeding of Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (pp. 4104-4108).

Kirsch, U. (1996). Integration of reduction and expansion processes in layout optimization. *Structural Optimization*, *11*, 13-18. 10.1007/BF01279649

Li, L. J., Huang, Z. B., & Liu, F. (2009). A heuristic particle swarm optimization method for truss structures with discrete variables. *Computers and Structures*, *87*(7-8), 435-443. 10.1016/j.compstruc.2009.01.004

Li, Y., & Zhen, Y. (2019). Application of improved bat algorithm in truss optimization. *KSCE Journal of Civil Engineering*, *23*, 10.1007/s12205-019-2119-2

Luh, G. C., & Lin, C. Y. (2008). Optimal design of truss structures using ant algorithm. *Structural and Multidisciplinary Optimization*, *36*(4), 365-379. 10.1007/s00158-007-0175-6

Luh, G. C., & Lin, C. Y. (2011). Optimal design of truss-structures using particle swarm optimization. *Computers and Structures*, *89*(23-24), 2221-2232. 10.1016/j.compstruc.2011.08.013

Malik, R., Rahman, T., Mohd Hashim, S., & Ngah, R. (2007). New particle swarm optimizer with sigmoid increasing inertia weight. *International Journal of Computer Science and Security*, *1*,

Martínez, P., Martí, P., & Querin, O. M. (2007). Growth method for size, topology, and geometry optimization of truss structures. *Structural and Multidisciplinary Optimization*, *33*(1), 13-26. 10.1007/s00158-006-0043-9

McKeown, J. J. (1998). Growing optimal pin-jointed frames. *Structural Optimization*, *15*(2), 92-100.

Miguel, L., Lopez, R., & Miguel, L. (2013). Multimodal size, shape, and topology optimisation of truss structures using the firefly algorithm. *Advances in Engineering Software*, *56*, 23-37. 10.1016/j.advengsoft.2012.11.006

Nanakorn, P., & Meesomklin, K. (2001). An adaptive penalty function in genetic algorithms for structural design optimization. *Computers and Structures*, *79*(29-30), 2527-2539. 10.1016/S0045-7949(01)00137-7

Nanakorn, P., Petprakob, W., & Naga, V. C. K. (2014). Object-oriented programming for topology optimization of steel truss structures by multipopulation particle swarm optimization. *Proceeding of Proceedings of the 12th International Conference on Steel, Space and Composite Structures*.

Neeraja, D., Kamireddy, T., Kumar, P. S., & Reddy, V. S. (2017). Weight optimization of plane truss using genetic algorithm. *Proceeding of IOP Conference Series: Materials Science and Engineering*.

Nimtawat, A., & Nanakorn, P. (2009). Automated layout design of beam-slab floors using a genetic algorithm. *Computers and Structures*, *87*(21-22), 10.1016/j.compstruc.2009.06.007

Nimtawat, A., & Nanakorn, P. (2010). A genetic algorithm for beam-slab l design of rectilinear floors. *Engineering Structures*, *32*(11), 3488-3500. DOI: 10.1016/j.engstruct.2010.07.018

Niu, B., Zhu, Y., He, X., & Shen, H. (2008). A multi-swarm optimizer based fuzzy modeling approach for dynamic systems processing. *Neurocomputing*, *71*(7-9), 1436-1448. 10.1016/j.neucom.2007.05.010

Ohsaki, M., Fujisawa, K., Katoh, N., & Kanno, Y. (1999). Semi-definite programming for topology optimization of trusses under multiple eigenvalue constraints. *Computer Methods in Applied Mechanics and Engineering*, *180*(1–2), 203-217. http://dx.doi.org/10.1016/S0045-7825(99)00056-0

Peydro Rasero, M. Á., Sellés Cantó, M. Á., Martínez Sanz, A. V., Plá Ferrando, R., & Sánchez Caballero, S. (2012). Recent advances in structural optimization. *Annals of The University of Oradea*, *1*(21), 118-127.

Rajan, S. D. (1995). Sizing, shape, and topology design optimization of trusses using genetic algorithm. *Journal of Structural Engineering*, *121*(10), 1480-1487. 10.1061/(ASCE)0733-9445(1995)121:10(1480)

Rajeev, S., & Krishnamoorthy, C. S. (1992). Discrete optimization of structures using genetic algorithms. *Journal of Structural Engineering (United States)*, *118*(5), 1233-1250. 10.1061/(ASCE)0733-9445(1992)118:5(1233)

Ratnaweera, A., Halgamuge, S. K., & Watson, H. C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, *8*(3), 240-255. 10.1109/TEVC.2004.826071

Richardson, J. N., Adriaenssens, S., Bouillard, P., & Coelho, R. F. (2012). Multiobjective topology optimization of truss structures with kinematic stability repair. *Structural and Multidisciplinary Optimization*, *46*(4), 513-532. 10.1007/s00158-012-0777-5

Serpik, I. N., Alekseytsev, A. V., & Balabin, P. Y. (2017). Mixed approaches to handle limitations and execute mutation in the genetic algorithm for truss size, shape and topology optimization. *Periodica Polytechnica Civil Engineering*, *61*(3), 471-482. 10.3311/PPci.8125

Smith, O. (1994). Interactive system for truss topology design with automatic generation of the ground structure. *Proceeding of International Conference on Computational Structures Technology - Proceedings* (pp. 103-112).

Sokól, T. (2014). Multi-load truss topology optimization using the adaptive ground structure approach. *Proceeding of Recent Advances in Computational Mechanics - Proceedings of the 20th International Conference on Computer Methods in Mechanics, CMM 2013* (pp. 9-16).

Sokół, T. (2011). A 99 line code for discretized michell truss optimization written in mathematica. *Structural and Multidisciplinary Optimization*, *43*(2), 181-190. 10.1007/s00158-010-0557-z

Sönmez, M. (2011). Artificial bee colony algorithm for optimization of truss structures. *Appl. Soft Comput.*, 11, 2406-2418. 10.1016/j.asoc.2010.09.003

Tsiptsis, I. N., Liimatainen, L., Kotnik, T., & Niiranen, J. (2019). Structural optimization employing isogeometric tools in particle swarm optimizer. *Journal of Building Engineering*, 10.1016/j.jobe.2019.100761

Wu, C. Y., & Tseng, K. Y. (2010). Truss structure optimization using adaptive multi-population differential evolution. *Structural and Multidisciplinary Optimization*, *42*(4), 575-590. 10.1007/s00158-010-0507-9

Wu, Y., Li, Q., Hu, Q., & Borgart, A. (2017). Size and topology optimization for trusses with discrete design variables by improved firefly algorithm. *Mathematical Problems in Engineering*, 2017, 10.1155/2017/1457297

Xin, J., Chen, G., & Hai, Y. (2009). *A particle swarm optimizer with multi-stage linearly-decreasing inertia weight*.

Yang, X. S. (2009). *Firefly algorithms for multimodal optimization*.

Zegard, T., & Paulino, G. H. (2014). Grand — ground structure based topology optimization for arbitrary 2d domains using matlab. *Structural and Multidisciplinary Optimization*, *50*(5), 861-882. 10.1007/s00158-014-1085-z

BIOGRAPHY

Name	Mr. Alin Shakya
Date of Birth	August 05, 1974
Education	1999: Bachelor of Engineering (Civil Engineering)
	Pulchowk Campus, Institute of Engineering
	Tribhuvan University
	2003: Master of Science (Structural Engineering)
	Pulchowk Campus, Institute of Engineering
	Tribhuvan University

Publications

Shakya, A., Nanakorn, P., & Petprakob, W. (2018). A ground-structure-based representation with an element-removal algorithm for truss topology optimization. *Structural and Multidisciplinary Optimization*, *58*(2), 657-675. 10.1007/s00158-018-1917-3

Shakya, A., and Nanakorn, P. (2017) Two-dimensional truss topology optimization with no overlapping elements by multi-population particle swarm optimization. *In Proceedings of the 15th East Asia-Pacific Conference on Structural Engineering and Construction, EASEC-15* 11-13 October 2017, Xi'an, P. R. China, 1648-1655.