EVALUATION OF RECENT METAHEURISTIC SEARCH TECHNIQUES IN OPTIMUM DESIGN OF STEEL STRUCTURES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

AHMET ERAY KALAYCI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ENGINEERING SCIENCES

APRIL 2023

Approval of the thesis:

EVALUATION OF RECENT METAHEURISTIC SEARCH TECHNIQUES IN OPTIMUM DESIGN OF STEEL STRUCTURES

submitted by **AHMET ERAY KALAYCI** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Engineering Sciences, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Murat Dicleli
Head of the Department, **Engineering Sciences** _____

Prof. Dr. Murat Dicleli
Supervisor, **Engineering Sciences, METU** _____

Prof. Dr. Oğuzhan Hasançebi
Co-Supervisor, **Civil Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Zehra Çağnan Ertuğrul
Engineering Sciences, METU _____

Prof. Dr. Murat Dicleli
Engineering Sciences, METU _____

Assoc. Prof. Dr. Saeid Kazemzadeh
Civil Engineering, Atılım University _____

Date: 26.04.2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name : Ahmet Eray Kalaycı

Signature :

ABSTRACT

EVALUATION OF RECENT METAHEURISTIC SEARCH TECHNIQUES IN OPTIMUM DESIGN OF STEEL STRUCTURES

Kalaycı, Ahmet Eray
Master of Science, Engineering Sciences
Supervisor: Prof. Dr. Murat Dicleli
Co-Supervisor: Prof. Dr. Oğuzhan Hasançebi

April 2023, 84 pages

Meta-heuristic search techniques have become popular in the last decade due to advantages, such as preventing local optimum solutions and producing discrete solutions to structural systems. The main purpose of the optimization techniques is to achieve the minimum weight or cost design of a structural system. As there are so many decision variables while calculating the weight or cost of a structure, structural optimization is carried out to find the best design out of a large set of acceptable solutions. Five optimization techniques; namely Atomic Orbital Search, Honey Badger Algorithm, Nuclear Fission-Nuclear Fusion Algorithm, Pathfinder Algorithm, and Salp Swarm Algorithm are used in this study to optimize the investigated benchmark problems and structural test problems taken from the literature. These algorithms are initiated randomly, but each of them has different improvement steps to reach the optimum result. In this study, the results obtained with all these five different metaheuristic algorithms are compared with each other and also with the formerly reported solutions of the investigated problems in the previous studies.

Keywords: Structural Optimization, Salp Swarm Algorithm, Steel Truss Systems, Size Optimization

ÖZ

ÇELİK YAPILARIN OPTIMUM TASARIMINDA YENİ META-SEZGİSEL ARAMA TEKNİKLERİNİN DEĞERLENDİRİLMESİ

Kalaycı, Ahmet Eray
Yüksek Lisans, Mühendislik Bilimleri
Tez Yöneticisi: Prof. Dr. Murat Dicleli
Ortak Tez Yöneticisi: Prof. Dr. Oğuzhan Hasançebi

Nisan 2023, 84 sayfa

Meta-sezgisel teknikler, yerel optimum çözümleri engellemesi ve kolay uyarlanabilir yapılarının olması gibi avantajları nedeniyle son on yılda popüler hale geldi. Optimizasyon tekniklerinin temel amacı, yapısal bir sistem için minimum ağırlığı veya minimum maliyeti elde etmektir. Yapı maliyetinin hesaplanmasında çok fazla değişken olduğu için yapısal optimizasyonda ağırlığın minimize edilmesi amaçlanmaktadır. Bu Çalışmada, literatürdeki kıyaslama problemlerini ve yapısal test problemlerini optimize etmek için Atomik Orbital Arama, Bal Porsuğu Algoritması, Nükleer Fisyon-Nükleer Füzyon Algoritması, Pathfinder Algoritması ve Plantonik Tunikap Sürüsü Algoritması kullanılmıştır. Bu algoritmalar rastgele başlangıç değerlerine sahiptir, ancak her birinin optimum sonuca ulaşmak için farklı iyileştirme adımları vardır. Bu çalışmada tüm bu beş farklı metasezgisel algoritmanın sonuçları kendi içinde ve önceki çalışmalarla karşılaştırılmıştır

Anahtar Kelimeler: Yapısal Optimizasyon, Planktonik Tunikap Sürü Algoritması, Çelik Kafes Yapılar, Boyut Optimizasyonu

To My Family

ACKNOWLEDGMENTS

First of all, I want to offer my deepest gratitude to my supervisor Prof. Dr. Murat Dicleli and my co-supervisor Prof. Dr. Oğuzhan Hasançebi, for their invaluable guidance, comments, supports and thoughts throughout this study.

I wish to express my sincere gratitude to the examining committee members for their efforts in reviewing this thesis and useful suggestions.

I am grateful to my family for their endless support, patience, and encouragement.

I wholeheartedly thank my friends for their continuous motivation and encouraging conversation in all the difficult times for me.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xiii

LIST OF FIGURES xiv

LIST OF ABBREVIATIONS xvii

LIST OF SYMBOLS xxi

CHAPTERS

1 INTRODUCTION 1

1.1 Structural Optimization and Types 1

1.1.1 The Categories of Structural Optimization 1

1.1.1.1 Topology Optimization 3

1.1.1.2 Shape Optimization 4

1.1.1.3 Material Optimization 4

1.2 Methods of Structural Optimization 5

1.2.1 Traditional Optimization Techniques 5

1.2.2 Meta-heuristic Optimization Techniques 7

1.3 Aim of this study 8

2 FORMULATION OF OPTIMIZATION PROBLEM 9

2.1 Design Variables 9

2.2 Constraints 9

2.2.1 Constraints for Benchmark Problems 10
2.2.2 Constraints for Structural Design Problems 10
2.3 Objective Function 12
3 LITERATURE REVIEW 13
3.1 Studies on Structural Optimization 13
4 OPTIMIZATION METHODS 21
4.1 Atomic Orbital Search (AOS) Optimization Algorithm 21
4.1.1 Mathematical Model of AOS Algorithm 22
4.2 Honey Badger Algorithm 24
4.2.1 Mathematical Model of Honey Badger Algorithm 25
4.3 Nuclear Fission-Nuclear Fusion (N2F) Algorithm 27
4.3.1 Mathematical Model of N2F Optimization Algorithm 28
4.4 Pathfinder Algorithm 28
4.4.1 Mathematical Model of Pathfinder Algorithm 29
4.5 Salp Swarm Algorithm 31
4.5.1 Mathematical Model of Salp Swarm Algorithm 32
4.6 Progress of the Software 33
5 NUMERICAL EXAMPLES 35
5.1 Benchmark Test Problems 35
5.1.1 25- Member Truss 36
5.1.1.1 Results of Analyses 37
5.1.2 38- Member Truss 42
5.1.2.1 Results of Analyses 42
5.1.3 72-Member Truss 48

5.1.3.1 Results of Analyses 50
5.1.3.1.1 The First Design Case 50
5.1.3.1.2 The Second Design Case 55
5.2 Structural Problems 60
5.2.1 130- Member Tower 60
5.2.1.1 Results of Analyses 62
5.2.2 117- Member Cantilever Beam 67
5.2.2.1 Results of Analyses 68
6 CONCLUSIONS 75
6.1 Summary and Conclusion 75
6.2 Future Studies 79
7 REFERENCES 81

LIST OF TABLES

TABLES

Table 5.1 Member groups of the 25-member truss 36
Table 5.2 Loadings at 25-member truss 37
Table 5.3 Summary of the optimum results for the 25-bar benchmark truss problem 41
Table 5.4 Summary of the optimum results for the 38-bar cantilever truss problem 47
Table 5.5 The first loading case for the 72-member truss tower 48
Table 5.6 The second loading case for the 72-member truss tower 48
Table 5.7 Member groups of the 72-member truss tower 49
Table 5.8 Summary of the optimum results for the 72-bar cantilever truss problem (case 1) 54
Table 5.9 Summary of 72-bar benchmark results (case 2) 59
Table 5.10 Loadings of 130-member tower 62
Table 5.11 Summary of the optimum results for the 130-bar tower truss problem 67
Table 5.12 Loadings at 117-member truss 68
Table 5.13 Summary of the optimum results for the 117-bar cantilever truss problem 73

LIST OF FIGURES

FIGURES

Figure 1.1. Types of Structural Optimization (Kato, 2010) 2
Figure 1.2. Structural size optimization problem (Bhensdida, 2015)..... 2
Figure 1.3. Subclasses of the topology optimization, (a) geometric topology optimization, (b) discrete distribution, (c) continuous distribution (Maute, 1998) ... 3
Figure 1.4. The shape optimization sample (Crew and Salvio, 2010) 4
Figure 1.5. An illustration of the graphical method (Arora, 2004) 5
Figure 1.6. Classification of metaheuristic algorithms (Dhirman, 2017)..... 7
Figure 4.1. Electron motion models (Azizi, 2021). 22
Figure 4.2. The AOS algorithm's pseudo code (Azizi, 2021)..... 24
Figure 4.3. Pseudo-code of Honey Badger Algorithm (Hashim et al, 2021) 27
Figure 4.4. Sample collective movement of Pathfinder Algorithm (Yapici et al, 2019). 29
Figure 4.5. The Pseudo code of the Pathfinder Algorithm (Yapici et al, 2019)..... 31
Figure 4.6. (a) Single Salp, (b) Salps Herd (Mirjalili , 2017)..... 32
Figure 4.7. The Pseudo code of SSA (Mirjalili , 2017)..... 33
Figure 5.1. A 3-D view of the 25-member truss..... 36
Figure 5.2. Convergence curves for the 25-bar benchmark truss problem with AOS 37
Figure 5.3. Convergence curves for the 25-bar benchmark truss problem with HBA 38
Figure 5.4. Convergence curves for the 25-bar benchmark truss problem with N2F 39
Figure 5.5. Convergence curves for the 25-bar benchmark truss problem with PFA 40
Figure 5.6. Convergence curves for the 25-bar benchmark truss problem with SSA 40
Figure 5.7. 38-member cantilever truss 42

Figure 5.8. Convergence curves for the 38-bar cantilever truss problem with AOS 43
Figure 5.9. Convergence curves for the 38-bar cantilever truss problem with HBA 44
Figure 5.10. Convergence curves for the 38-bar cantilever truss problem with N2F 44
Figure 5.11. Convergence curves for the 38-bar cantilever truss problem with PFA 45
Figure 5.12. Convergence curves for the 38-bar cantilever truss problem with SSA 46
Figure 5.13. 72-member truss tower 49
Figure 5.14. Convergence curves for the 72-bar tower truss (case-1) with AOS ... 50
Figure 5.15. Convergence curves for the 72-bar tower truss (case-1) with HBA... 51
Figure 5.16. Convergence curves for the 72-bar tower truss (case-1) with N2F.... 52
Figure 5.17. Convergence curves for the 72-bar tower truss (case-1) with PFA.... 53
Figure 5.18. Convergence curves for the 72-bar tower truss (case-1) with SSA.... 53
Figure 5.19. Convergence curves for the 72-bar tower truss (case-2) with AOS ... 55
Figure 5.20. Convergence curves for the 72-bar tower truss (case-2) with HBA... 56
Figure 5.21. Convergence curves for the 72-bar tower truss (case-2) with N2F.... 57
Figure 5.22. Convergence curves for the 72-bar tower truss (case-2) with PFA.... 58
Figure 5.23. Convergence curves for the 72-bar tower truss (case-2) with SSA.... 58
Figure 5.24. 130-member tower..... 61
Figure 5.25. Loadings points of 130-member tower..... 61
Figure 5.26. Convergence curves for the 130-bar structural truss problem with AOS 62
Figure 5.27. Convergence curves for the 130-bar structural truss problem with HBA 63
Figure 5.28. Convergence curves for the 130-bar structural truss problem with N2F 64

Figure 5.29. Convergence curves for the 130-bar structural truss problem with PFA 65
Figure 5.30. Convergence curves for the 130-bar structural truss problem with SSA 66
Figure 5.31. 117-bar Cantilever Beam (Kazemzadeh Azad et.al, 2014) 67
Figure 5.32. Convergence curves for the 117-bar cantilever truss problem with AOS 69
Figure 5.33. Convergence curves for the 117-bar cantilever truss problem with HBA 69
Figure 5.34. Convergence curves for the 117-bar cantilever truss problem with N2F 70
Figure 5.35. Convergence curves for the 117-bar cantilever truss problem with PFA 71
Figure 5.36. Convergence curves for the 117-bar cantilever truss problem with SSA 72

LIST OF ABBREVIATIONS

ABBREVIATIONS

ABC	: Artificial Bee Colony
ABC-AP	: Artificial Bee Colony with Adaptive Penalty
AC	: Ant Colony Optimization
ACO	: Ant Colony Optimization
AISC	: American Institute of Steel Construction
ANSI	: American National Standards Institute
AOS	: Atomic Orbital Search
API	: Application Programming Interface
ARCGA	: Adaptive Real-Coded Genetic Algorithm
BB-BC	: Big Bang-Big Crunch
BFGS	: Broyden Fletcher Goldfarb Shanno
CA	: Cultural Algorithm
CBO	: Colliding Bodies Optimization
CMLPSA	: Corrected Multi-Level & Multi-Point Simulated Annealing
CPSO	: Cellular Automata hybridized with Particle Swarm Optimization
CS	: Cuckoo Search
CSP	: Chaotic Swarming of Particles
CSS	: Charged System Search
CSS-BBBC	: Charged System Search - Big Bang-Big Crunch

DAJA : Discrete Advanced Jaya Algorithm

DFP : Davidon Fletcher Powell

DHPSACO : Discrete Heuristic Particle Swarm Ant Colony Optimization

EBB-BC : Exponential Big Bang – Big Crunch

EHO : Elephant Herding Optimization

EHOC : Elephant Herding Optimization Cultural

ES : Evolution Strategies

ESs : Evolution Strategies

FA : Firefly Algorithm

FEAPGEN : Based on Genetic Algorithm with Finite Element Analysis Program

FFA-LS : Firefly Algorithm with Least Square

FPA : Flower Pollination Algorithm

FSD-ES : Fully Stressed Design based on Evolution Strategy

GA : Genetic Algorithm

GGP : General Geometric Programming

GNMS : Genetic-Nelder Mead Simplex

GSO : Group Search Optimizer

GSS_A : Guided Stochastic Search first Approach

GSS_B : Guided Stochastic Search second Approach

GWO : Grey Wolf Optimizer

HBA : Honey Badger Algorithm

HBB-BC : Hybrid Big Bang – Big Crunch

HPSACO : Heuristic Particle Swarm Ant Colony Optimization

HPSO : Heuristic Particle Swarm Optimization

HPSSO : Hybrid Particle Swallow Swarm Optimization

HS : Harmony Search

IGSO : Improved Group Search Optimizer

IGWO : Improved Grey Wolf Optimizer

ISCSO : International Student Competition in Structural Optimization

JA : Jaya Algorithm

LRFD : Load and Resistance Factor Design

MABC : Modified Artificial Bee Colony

MBB-BC : Modified Big Bang – Big Crunch

MSOS : Modified Symbiotic Organisms Search

MSPSO : Multi-Stage Particle Swarm Optimization

N2F : Nuclear Fission-Nuclear Fusion Algorithm

OAPI : Open Application Programming Interface

PFA : Pathfinder Algorithm

PSO : Particle Swarm Optimization

QP : Quadratic Programming

SA : Simulated Annealing

SAHS : Self-Adaptive Harmony Search

SASS : Self-Adaptive Step-Size

SCPSO : Sequential Cellular Particle Swarm Optimization

SGA : Simple Genetic Algorithm

SOPT : Simple Optimization Algorithm

SQP : Sequential Quadratic Programming

SSA : Salp Swarm Algorithm

TLBO : Teaching-Learning-Based Optimization

TS : Tabu Search

UBS : Upper Bound Strategy

WEO : Water Evaporation Optimization

cm : Centimeter

in : Inch

kg : Kilogram

kN : Kilonewton

ksi : Kilopound per Square Inch

kips : Kilopounds

lb : Libre

LIST OF SYMBOLS

SYMBOLS

A_e	: Effective Area of an Element
A_g	: Gross Ares of an Element
BE^k	: Binding Energy in kth layer
BS^k	: Binding State in kth layer
C	: A constant number
D_{ij}	: Distance between ith search agent and jth search agent
E_i	: Objective function of ith search agent
E^k_i	: Objective function in kth layer ith search agent
F	: Value of Flag
F_a	: Maximum stress
F_{cr}	: Critical Stress
F_e	: Elastic Buckling Stress
F_j	: The jth variable value of the best search agent
F_u	: Specified Minimum Tensile Strength
F_y	: Specified Minimum Yield Stress
I	: Intensity
I_i	: Intensity of smelling
I^k	: Fusibility index of kth member
K	: Factor of Effective Length

L_c : Effective Length

LE^k : Lowest Energy Level in kth layer

N^k : Random number between 0 and 1

P_n : Nominal Axial Strength

S : Strength of food source

X^k : Position of each variable

cos : Cosinus

c_1 : A calculation value for balance

c_2 : A random number between 0 and 1

c_3 : A random number between 0 and 1

d_i : Distance between the current search agent and the leader of swarm

f : Stress

f_{best} : Best result in search agents

f^k : Result of kth search agent

k : Number of Orbit

l : Unbraced Length

lb_i : Lower bound of the ith variable

m : Total number of candidates for whole search agents

p : Total number of candidates at the kth layer

p_{global} : Value of the variable of the global best candidate

p_{new} : New value of the variable of a candidate

p_{old} : Old value of the variable of a candidate

r : Gyration Radius

r_1 : A random number between 0 and 1

r_2 : A random number between 0 and 1

r_3 : A random number between 0 and 1

r_4 : A random number between 0 and 1

r_5 : A random number between 0 and 1

r_6 : A random number between 0 and 1

r_7 : A random number between 0 and 1

rand : A random number between 0 and 1

t : Current iteration number

t_{max} : Maximum iteration number

ub_i : Upper bound of the ith variable

u_1 : Random vector between [-1,1]

u_2 : Random vector between [-1,1]

x_c : Mass center

x_i : Value of ith variable

x_i^j : Position of ith search agent's jth variable

x_i^K : Position of ith search agent's Kth variable

$x_{i,min}^j$: Minimum allowable value of ith search agent's jth variable

$x_{i,max}^j$: Maximum allowable value of ith search agent's jth variable

x_j^i : Position jth search agent's ith variable

x_j^K : Position of jth search agent's Kth variable

x_{max} : Maximum value of each variable of current search agents

x_{min} : Minimum value of each variable of current search agents

x_p^K : Position of pth search agent's Kth variable

x_{prey} : Leader search agent of the swarm

α : Factor of density

α_i : A random number between 0 and 1 at the ith layer

β : Ability to reach food

β_i : Random number between 0 and 1 at the ith layer

μ : A parameter between 10^0 and 10^{20}

δ : Displacement

δ_i : Maximum displacement

ε : Term of vibration

γ_i : A random number between 0 and 1 ith layer of each group

λ : Slenderness

λ_a : Maximum slenderness

ρ : Amplification Factor between 1.01 and 1.10

ϕ : A random number between 0 and 1

ϕ_t : Resistance factor

ϕ_t : Constant value

π : pi

CHAPTER 1

INTRODUCTION

The minimum weight or cost optimization of steel structures has long been studied in the literature using a variety of different optimization techniques. In the past, mathematical programming techniques, which utilized information based on the derivative of the objective function and constraints with respect to the design variables, were usually implemented by the researchers. Recently, the applications of structural optimization have been attempted using newly emerging search methods that avoid the use of gradient information during the search, such as meta-heuristic search algorithms. Optimization is implemented for the designing process in engineering, the planning process in logistics, and solving different problems in various other disciplines.

1.1 Structural Optimization and Types

Structural Optimization is used during the design process of structures and systems in all disciplines of engineering, especially civil engineering, mechanical engineering, and aircraft engineering.

1.1.1 The Categories of Structural Optimization

The optimization method depends on the initial raw data such as usable materials, cross-sections of the elements, geographic data, and the expectation from the system to be analyzed. As shown in Figure 1.1, structural optimization can be implemented in four different types depending on the type of design variables employed; namely size optimization, shape optimization, topology optimization, and material

optimization. Optionally, these optimization types can also be implemented together to solve a problem.

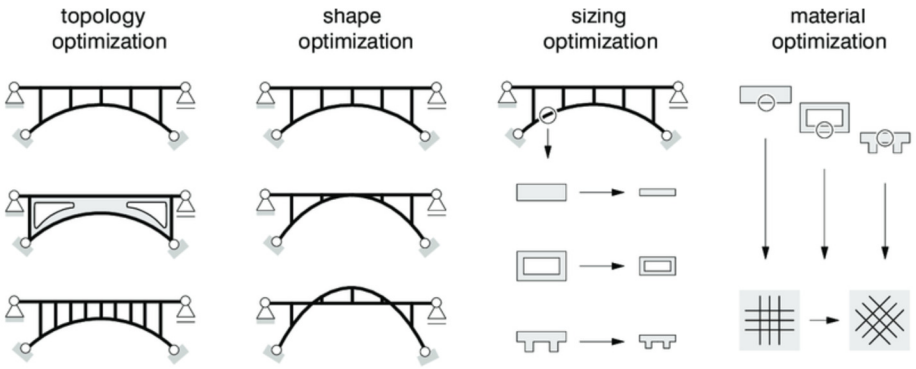

Figure 1.1. Types of Structural Optimization (Kato, 2010)

Size optimization is related to finding the best dimensions of the elements in a structure; for example, the thickness or height of the beam, wall thickness, diameter of a hollow tube, the thickness of a shell element, etc. The dimensions of structural members are employed as design variables during the optimum design process of a structural system. The optimization algorithm employed tries to find the optimum dimensions of the elements leading to the minimum weight or cost design of the system. At the end of the size optimization process, some structural elements would have larger or smaller cross-sections than the initially assigned ones as illustrated in Figure 1.2.

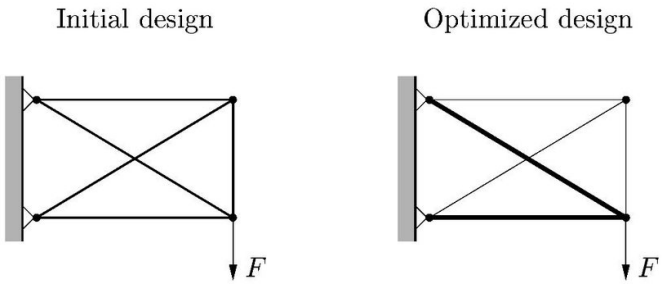

Figure 1.2. Structural size optimization problem (Bhensdida, 2015)

1.1.1.1 Topology Optimization

The purpose of topology optimization is to find the optimum topology of a structural system. In general, the term "topology" refers to the existence or non-existence of structural elements in a structural system.

Structural truss systems consist of bar elements, which are located between nodes. When applied to such systems, topology optimization tries to find which of these bar elements will remain in the structural model and which elements are not necessary and thus should be removed from the structural model. An efficiently implemented topology design optimization process will lead to the optimum design with the least weight of the structure.

The two subclasses of topology optimization are referred to as geometric topology optimization and material topology optimization, which are illustrated in Figure 1.3.

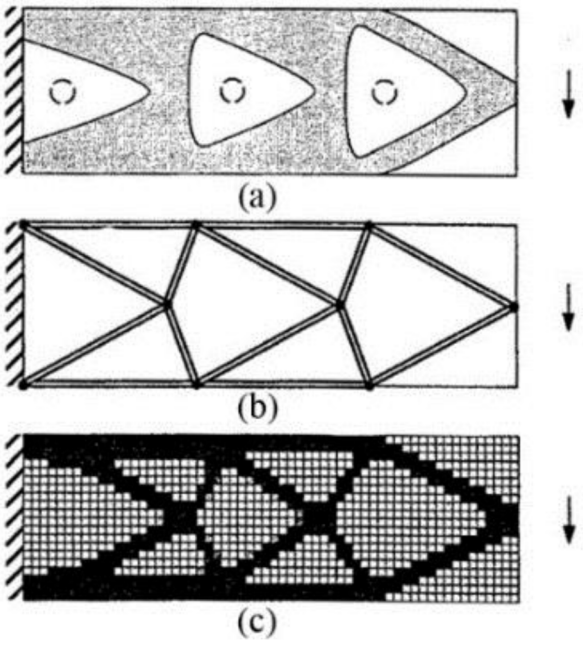

Figure 1.3. Subclasses of the topology optimization, (a) geometric topology optimization, (b) discrete distribution, (c) continuous distribution (Maute, 1998)

1.1.1.2 Shape Optimization

Shape optimization deals with finding the optimum geometry of structural systems by varying the coordinates of nodes. In other words, in shape optimization nodal point coordinates are employed as design variables, while the sizes and the member connectivities are left unchanged in the optimum design process.

This method tries to find the optimum node location in a finite element model. One of the first applications of shape optimization dates back to the work of Galileo in 1638, which produced a solution to the shape optimization problem shown in Figure 1.4.

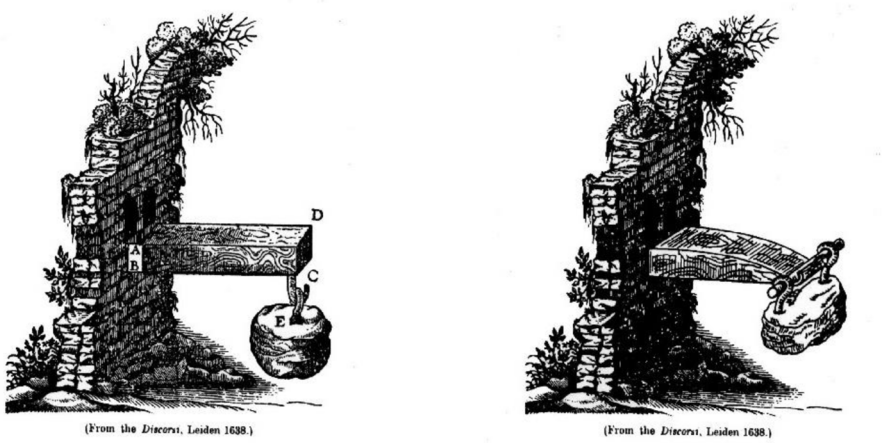

Figure 1.4. The shape optimization sample (Crew and Salvio, 2010)

1.1.1.3 Material Optimization

In general, a structural system may consist of various parts with different material types, such as steel, wood, or aluminum. Then, material optimization aims to find the best use of material types or properties for different parts of a structural system. Accordingly, material types or properties are employed as design variables during the optimization process. Material optimization can be used alone or used in conjunction with other optimization models.

1.2 Methods of Structural Optimization

A vast number of different optimization methods exist in the optimization literature. In general, the choice of an optimization method used to solve a problem is decided upon the nature of the problem at hand, such as whether the problem is constrained or not, whether the type of design variables are continuous, discrete, integer, or mixed, etc. In a broad sense, the optimization methods used in structural optimization can be collected under two categories as Traditional Optimization Techniques and Meta-heuristic Optimization Methods.

1.2.1 Traditional Optimization Techniques

The optimum solutions of given mathematical problems or functions can be obtained using the graphical method and various conventional search techniques. Although the graphical method guarantees to find the global optimum solution of a problem, it is applicable to problems that have two design variables only. In this method, the objective function and all the constraints functions are plotted on a graph sheet, and the optimum solution is identified with the help of objective function contours. An illustration of the graphical method is depicted in Figure 1.5.

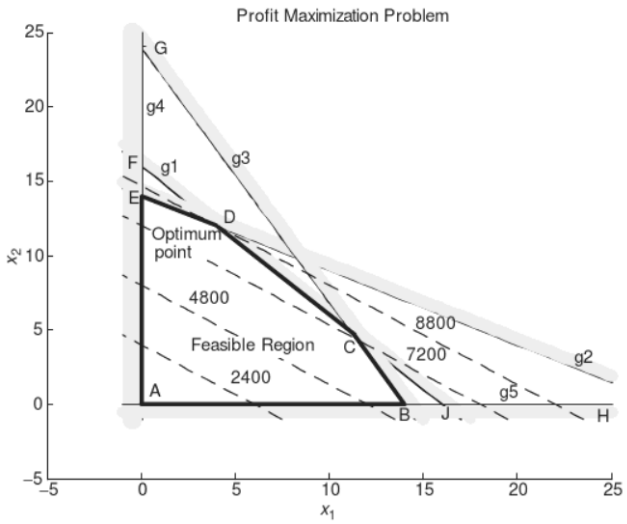

Figure 1.5. An illustration of the graphical method (Arora, 2004)

Optimization problems can be divided into two groups as linear programming and nonlinear programming optimization problems. In a linear programming problem, the optimization function as well as problem constraints are all expressed as a linear function of design variables. In a nonlinear programming problem, however, at least the objective function or one of the problem constraints is expressed as a nonlinear function of design variables. Similarly, optimization problems (and techniques) can also be divided into two groups as constrained or unconstrained optimization problems (and techniques) such that some constraints are involved in the former, and no constrained are involved in the latter.

The Simplex Method is the most widely used technique for linear programming problems. On the other hand, the unconstrained optimization methods are usually attempted by Steepest Descent Method, which is developed by Cauchy in 1847 (Arora, 2004). Newton's method, which uses Taylor's series of second-order expansions, obtains more accurate results than the search methods that use first-order derivatives. In 1963, the steepest descent method and Newton's method are improved by Marquart's modification (Arora, 2004).

Sequential Linear Programming and Sequential Quadratic Programming are two methods that are used to solve constrained optimization problems that have different initial conditions. The other methods are the Constrained Steepest Descent Method, Simplex Method for QP Problem, Quasi-Newton Method, and Gradient Projection Method. While using these methods, at the beginning of the optimization process, all the constraints must be normalized (Arora, 2004).

In addition, Steepest Descent Method, Conjugate Gradient Method, Newton Method, and Quasi-Newton Method are used for unconstrained problems. All of these methods can be used for better results with some modifications such as Scaling design variables with Hessian matrix, a modified Hessian formula as Marquart's modification, Inverse Hessian as DFP Method, and Direct Hessian as BFGS Method (Arora, 2004).

1.2.2 Meta-heuristic Optimization Techniques

Meta-heuristic optimization techniques have emerged to be powerful search tools and received increasing attention from all disciplines of science including structural optimization.

Meta-heuristic optimization techniques have a flexible and gradient-free structure and can avoid local optima. These features made them popular in the last decade (Mirjalili, 2017). As can be seen from Figure 1.6, they can be classified into five sub-branches, as evolutionary algorithms, physics-based algorithms, swarm-based algorithms, bio-inspired algorithms, and nature-inspired algorithms. All these algorithms employ different strategies or inspired methodologies to reach the optimum solution for the problems they are applied to.

Evolutionary Algorithms and Swarm-Based Algorithms (Mirjalili, 2017) are used most frequently while solving engineering design optimization problems. The most popular technique of Evolutionary algorithms is Genetic Algorithm, which uses a numerical solution model based on the evolution theory. At the beginning of the optimization process, the optimization process is initiated with randomly generated initial solutions. These solutions are improved using evolutionary operators, such as selection, crossover, and mutation so that the solution population is guided towards better regions of the design space throughout the successive iterations.

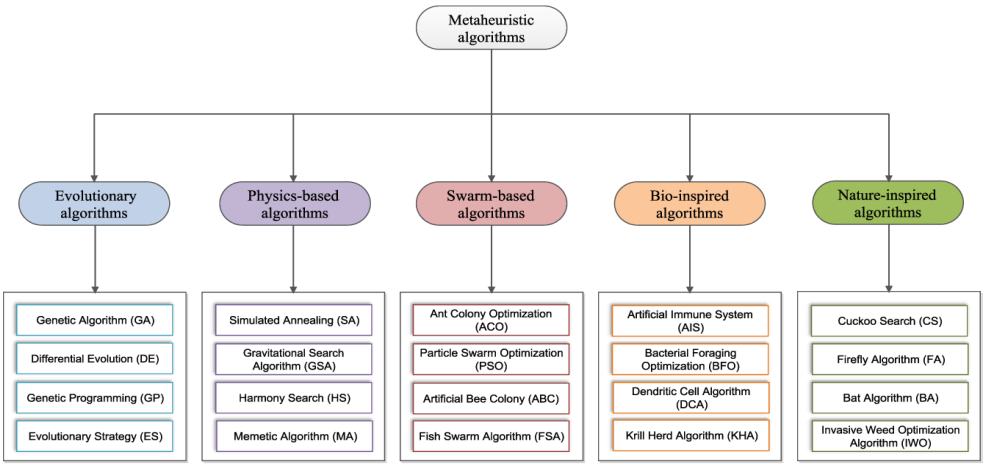

Figure 1.6. Classification of metaheuristic algorithms (Dhirman, 2017)

1.3 Aim of this study

The purpose of this thesis is to evaluate the search performances of several recent meta-heuristic search techniques in the optimum design of steel truss structures. In this regard, Atomic Orbital Search (AOS), Honey Badger Algorithm (HBA), Nuclear Fission-Nuclear Fusion Algorithm (N2F), Pathfinder Algorithm (PFA), and Salp Swarm Algorithm (SSA) are investigated. The optimum solutions produced to some benchmark and structural design problems using these five techniques are compared with each other and also with the formerly reported solutions of these problems in the literature.

The Atomic Orbital Search Algorithm, a physics-based algorithm, is inspired by electron movement in an atomic orbital range. The electrons revolve around the nucleus with different velocities and different distances from the core.

The second algorithm, Honey Badger Algorithm, is a basic swarm-based algorithm. This algorithm mimics the movement and food-searching behaviors of the honey badger swarm.

The Nuclear Fission-Nuclear Fusion Algorithm is another physics-based algorithm inspired by the nuclear reaction.

The Pathfinder Algorithm is a swarm-based algorithm and its design idea has a different movement characteristic. The algorithm imitates the group of animals that move collectively and follow the leader as well.

Finally, Salp Swarm Algorithm is a basic swarm algorithm mimicking the salp herd. The general movement behavior of the swarm is to follow the leader for reaching the food source.

CHAPTER 2

FORMULATION OF OPTIMIZATION PROBLEM

In this study, the sizing optimum design of steel trusses is studied in conjunction with five recent meta-heuristic search techniques using three benchmark test problems and two structural design problems. All optimization problems have the same problem description regarding the definitions of design variables, objective function, and constraints, which are discussed in the following sections.

2.1 Design Variables

In the context of sizing optimization of steel truss-type structures, design variables refer to the cross-sectional areas of truss elements. For practical purposes, referred to as discrete optimization, they are selected from the appropriate sections given in the profile lists specified in steel design codes. For continuous variable optimization problems, however, upper and lower bounds are specified for the design variables and they can be assigned to any value in between them.

2.2 Constraints

The constraints refer to a set of requirements that must be satisfied during a design process in order to create an acceptable design. In structural engineering applications, one can define two types of constraints. The first type is geometric constraints. The geometric constraints have nothing to do with the functionality or expected performance of a structural system, rather they are usually imposed due to fabrication or aesthetics requirements; such as the maximum or minimum slope of a roof. The second type of constraint is the so-called functional or behavior constraint. Functional or behavioral constraints are usually stipulated by a chosen code of

practice, such as maximum displacements or maximum tensile and compressive stresses on structural members.

Meta-heuristic optimization techniques are fundamentally unconstrained optimization algorithms. Hence, when meta-heuristic optimization techniques are applied to a constrained optimization problem, the problem should be turned into an unconstrained one by means of a penalty function approach. In this approach, penalty functions are defined for the constrained violations and they are integrated into the original objective function. Hence, when a constraint is violated by a design, a positive term is assigned associated with this violated constraint, and this term is added to the objective function of the corresponding design. Hence, the objective function of the design is increased for the violation of each constraint.

2.2.1 Constraints for Benchmark Problems

In this thesis, performance evaluation of implemented meta-heuristic techniques is implemented in conjunction with the benchmark optimization problems chosen from the structural optimization literature. In these benchmark optimization problems, the constraints are specified as the maximum nodal displacements in specified directions and/or maximum tension and compression stresses on members. In some problems, the displacement restrictions are imposed on more than one node, whereas in some others they are imposed on one node only. Similarly, some benchmark problems have only one stress limitation value under both tension and compression but in some other problems, more elaborate stress limitations are defined.

2.2.2 Constraints for Structural Design Problems

In structural design optimization problems investigated in this thesis, displacement, slenderness, and strength constraints are imposed. The displacement constraints are defined as the maximum nodal displacements in specified directions. The member strength constraints are imposed according to Load and Resistance Factor (LRFD)

design methodology specified by ANSI/AISC 360-16 design specification. Similarly, slenderness constraints on members are imposed according to ANSI/AISC 360-16 design specification. The formulations related to slenderness, stress, and displacement constraints are given in equations 2.1, 2.2, and 2.3, respectively.

$$\left|\frac{\lambda}{\lambda_a}\right| - 1 \leq 0 \tag{2.1}$$

$$\left|\frac{f}{F_a}\right| - 1 \leq 0 \tag{2.2}$$

$$\left|\frac{\delta}{\delta_i}\right| - 1 \leq 0 \tag{2.3}$$

While ANSI/AISC 360-16 imposes no upper limit for the slenderness of tension members, it is recommended that the slenderness ratio of tension members should be under 300. Similarly, the slenderness ratio of compression members is recommended to be should be under 200 according to ANSI/AISC 360-16. In general, the slenderness of a member can be formulated as given in equation 2.3. In this equation, l is the unbraced length, K is a factor of effective length, and r is the minimum gyration radius for the element.

$$\lambda = \frac{Kl}{r} \tag{2.3}$$

According to the LRFD-AISC 360-16, for tension members, the design strength is defined as $\phi_t P_n$. The design tensile strengths in gross and net sections are given in equations 2.4 and 2.5, respectively. In these equations, P_n, F_y, A_g, F_u, and A_e represent nominal axial strength, specified minimum yield stress, gross cross-sectional area, specified minimum tensile strength, and effective cross-sectional area, respectively.

$$\phi_t * P_n = 0.9 * F_y * A_g \tag{2.4}$$

$$\phi_t * P_n = 0.75 * F_u * A_e \tag{2.5}$$

The nominal strength for a compressive member is defined as P_n, and it is calculated with equation 2.6, where F_{cr} and A_g refer to critical compressive stress and gross cross-sectional area, respectively.

$$P_n = F_{cr} * A_g \tag{2.6}$$

The critical stress is calculated using two different formulas. When F_y/F_e is bigger than 2.25, equation 2.7 is used; otherwise, equation 2.8 is used. In these equations, F_y is the specified minimum yield stress, and F_e is the elastic buckling stress. Elastic buckling stress is determined using equation 2.9.

$$F_{cr} = \left(0.658^{\frac{F_y}{F_e}}\right) * F_y \quad \text{, if} \quad \frac{F_y}{F_e} \geq 2.25 \tag{2.7}$$

$$F_{cr} = 0.877 * F_e \quad \text{, if} \quad \frac{F_y}{F_e} < 2.25 \tag{2.8}$$

$$F_e = \frac{\pi^2 E}{\left(\frac{L_c}{r}\right)^2} \tag{2.9}$$

2.3 Objective Function

The objective function is defined according to the required performance from a structural system design. In the context of structural optimization, the objective function is usually chosen as minimizing the weight or cost of a structure. Although a direct relationship may be established between weight and cost for all types of steel structures, weight can be identified as a strong indication of cost for most steel structures. Besides, it is easier and more accurate to calculate weight rather than the cost which depends on many items, including material, construction, erection, transportation, etc. Therefore, structural weight is used for the objective function of the structural optimization problems studied here.

CHAPTER 3

LITERATURE REVIEW

In this chapter, some studies regarding the optimum design of steel trusses collected from the literature are briefly overviewed.

3.1 Studies on Structural Optimization

In Hasançebi et al. (2009) the performances of seven meta-heuristic algorithms in the size optimum design of steel trusses are investigated and compared. These search techniques are Genetic Algorithms (GA), Tabu Search (TS) , Particle Swarm Optimization (PSO), Evaluation Strategies (ESs), Harmony Search (HS), Simulated Annealing (SA) and Ant Colony Optimization (ACO). In this study, 25-bar tower benchmark problem, 113-bar plane bridge, 354-bar dome, 582-bar tower, and 960-bar double-layer grid are used as test problems, and the optimum designs produced to these problems with the aforementioned optimization techniques are compared.

Kaveh and Talatahari (2009) implemented the Big Bang-Big Crunch (BB-BC) algorithm for sizing optimization of steel space trusses. In this study, five different truss systems were analyzed, and the optimum designs produced for these structures were compared with those of the previous studies. These problems are 25-bar truss, 72-bar truss, 120-bar truss dome, 26-story truss tower, and a square double-layer grid system.

Degertekin (2011) used an improved version of the Harmony Search Algorithm (HSA) to achieve sizing optimization of truss systems. In this paper, 10-bar truss, 25-bar truss, 72-bar truss, and 200-bar were designed for the minimum weight. Two improved variants of the algorithm; namely, Efficient Harmony Search Algorithm (EHSA) and Self Adaptive Harmony Search Algorithm (SAHSA), were employed.

Kaveh and Khayatazad (2012) used the Ray Optimization algorithm (ROA) for the size and shape optimum design of truss structures. The implemented optimization algorithm mimics ray refraction, which is a basic physical phenomenon. In the study, the performance of the Ray Optimization algorithm was investigated with respect to the previous studies and other techniques using a problem suit of 72-bar truss, 120-bar truss, 200-bar truss, 37-bar truss, and a model of Forth Bridge.

Degertekin and Hayalioglu (2012) applied the Teaching-Learned-Based optimization (TLBO) method for the size optimization of truss systems. This optimization method is a meta-heuristic algorithm inspired by the relationship between a learner and a teacher. A problem suit of 10-bar truss, 25-bar truss, 72-bar truss, and 200-bar truss is used to implement numerical studies and investigations.

Hasançebi et al. (2013) used a Computationally Improved version of the SOPT Algorithm for the size optimization of truss systems. SOPT is a simple optimization algorithm, whose computational efficiency is improved with the aid of upper bound strategy (UBS). In this study, 10-bar truss, 17-bar truss, and 45-bar truss are tested and their results are compared with the results of the previous studies.

Hasançebi et al. (2013) employed the bat-inspired algorithm for sizing optimization of truss structure systems. This algorithm was designed mathematically designed by considering the behavior of bats while searching for prey. In this study, the following truss structures are used as test problems for investigating the performance of the algorithm: 25-bar truss, 354-bar dome truss, 693-bar-truss, and 942-bar truss tower. The optimum designs produced for these problems with the bat-inspired algorithm are compared with the formerly published solutions of these problems with other techniques, such as PSO, HS, SA, ESs, Ant Colony (AC), Simple Genetic Algorithm (SGA), TS, etc.

Kazemzadeh Azad et al. (2013) employed the Upper Bound Strategy (UBS)-integrated Big Bang-Big Crunch Algorithm for sizing optimization of steel trusses. This algorithm is mainly inspired by the theory of Big Bang-Big Crunch, which explains how the universe is generated. The optimum designs of the following

problems were investigated in the study: 10-bar cantilever truss, 45-bar truss bridge, and 120-bar truss dome. The optimum designs produced for these problems with the (UBS)-integrated Big Bang-Big Crunch Algorithm were compared with the formerly published solutions of these problems with other techniques, such as Adaptive Real-Coded Genetic Algorithm (ARCGA), Artificial Bee Colony (ABC), Modified Artificial Bee Colony (MABC), and Firefly Algorithm (FA).

Hybridized Firefly Algorithm was used by Baghlani and Makiabadi (2013) for the minimum weight design of truss systems. Firefly Algorithm mimics some behaviors of the firefly swarm's movement. In this study, Firefly Algorithm was hybridized with the so-called New Feasible Boundary Search Technique, which is an improvement to the former. The numerical applications were performed using five different structural design examples, which are 10-bar cantilever truss, 17-bar cantilever truss, 25-bar tower truss, 72-bar truss, and 120-bar truss dome. The results obtained were compared with those of the previous studies.

Kazemzadeh Azad and Hasançebi (2014) applied an elitist self-adaptive step-size search method for optimizing the truss structures. The numerical applications were performed using 17-bar cantilever truss, 45-bar bridge, 120-bar dome, and 200-bar truss examples. The optimum designs produced for these problems with the aforementioned optimization technique with the formerly published solutions of these problems with other techniques, such ARCGA, ABC, MABC, FA, and GA.

Kazemzadeh Azad et al. (2014) developed a so-called Guided stochastic search technique for discrete size optimization of steel truss systems. This algorithm utilizes the virtual work principle in conjunction with a fully strengthed design approach while determining members which have the highest influence on displacement quantities. The numerical applications, as well as performance comparison of the method, were illustrated using 10-bar truss benchmark problem in addition to four practical design examples, which are 117-bar cantilever truss, 130-bar tower truss, 392-bar double layer grid, and 354-bar dome truss. The optimum solutions obtained were compared with the results of PSO, BB-BC, Heuristic Particle Swarm

Optimization (HPSO), GA, Finite Element Analysis program based on Genetic Algorithm (FEAGEN), and Optimality Criteria algorithms.

Flager et al. (2014) introduced a general, flexible, and scalable method called the Fully Constrained Design Method for the size optimization of steel truss systems. The method was applied to the optimum design problems of 10-bar, 25-bar, and 200-bar trusses, and the results were compared with those of some Heuristic methods and the Optimality Criteria.

Adaptive Dimensional Search was proposed by Hasançebi and Kazemzadeh Azad (2015) for sizing optimization of steel truss systems. In this method, the stagnation of the method in a local optimum is avoided using several alternative approaches, such as Uphill move, Annealing Approach, and Penalty Relaxation. The application of the method was tested and illustrated using 10-bar, 200-bar truss, 113-bar bridge, 693-bar truss, and 960-bar grid systems and the results obtained were compared with the results of previous studies.

Kazemzadeh Azad and Hasançebi (2015) applied the Guided Stochastic Search Technique to the sizing optimization of truss structures subject to multiple displacement constraints. The numerical efficiency of the proposed method was investigated using some benchmark problems from the literature (i.e., 10-bar truss, 25-bar truss, 200-bar truss) in addition to several truss design problems, such as 117-bar cantilever, 130-bar tower, and 368-bar dome.

Kaveh and Ilchi Ghazaan (2015) developed an improved version of the Ray Optimization technique for the size and layout optimum design of truss structures. A problem suit of 10-bar cantilever beam, 37-bar bridge, 52-bar dome, 72-bar truss, and 120-bar dome is used in this study to examine and verify the numerical performance of the implemented technique. The results obtained to these problems are compared with the other reported solutions of the problems using Charged System Search (CSS), Charged System Search – Big Bang-Big Cruunch (CSS-BBBC), HS, and FA algorithms.

Bekdaş et al. (2015) employed the Flower Pollination Algorithm for the size optimization of steel truss structures. This algorithm mimics the efflorescence operation of phanerogam. Basically, there are two variants of the algorithm named cross-efflorescence and self-efflorescence. In this study, a problem suit of 25-bar tower, 72-bar truss, and 200-bar truss is used, and the results obtained are compared to those of the GA, ACO, BB-BC, Corrected Multi-Level & Multi-Point Simulated Annealing (CMLPSA), Hybrid Big Bang – Big Crunch (HBB-BC), Artificial Bee Colony with Adaptive Penalty (ABC-AP), TLBO, Hybrid Particle Swallow Swarm Optimization (HPSSO), Colliding Bodies Optimization (CBO), HS, Self-Adaptive Harmony Search (SAHS), Chaotic Swarming of Particles (CSP), General Geometric Programming (GGP) techniques.

Cheng et al. (2016) proposed Hybrid Harmony Search Algorithm for the optimum design of truss systems. This method differs from a standard Harmony Search algorithm in the sense that it employs the global best search characteristics of the PSO method. Six benchmark truss problems are numerically investigated, namely 10-bar truss, 15-bar truss, 25-bar truss, 52-bar truss, 72-bar truss, and 200-bar truss, and the results obtained are compared to the previously reported solutions of these problems by HS, HPSO, and Discrete Heuristic Particle Swarm Ant Colony Optimization (DHPSACO) algorithms.

Kazemzadeh Azad S. (2016) presented Enhanced hybrid metaheuristic algorithms for the size optimization of truss systems. The performances of ABC, MBB-BC, and Exponential Big Bang-Big Crunch (EBB-BC) methods are investigated comparatively using the following test problems: 117-bar truss, 200-bar truss, 354-bar truss, and 728-bar truss.

Do and Lee (2017) used a modified version of the Symbiotic Organisms Search algorithm for the optimum design of truss and tensegrity structures. The numerical applications are performed using 10-bar, 52-bar, 200-bar, 25-bar, and 160-bar truss problems in addition to 2-d hexagonal and 3-d truncated tetrahedral tensegrity structures.

Degertekin et al. (2017) investigated Jaya Algorithm (JA) in the optimum size, topology, and layout design of steel structures. For the numerical applications of the method, 200-bar, 942-bar, 1938-bar, and 25-bar truss systems were addressed, and the results obtained with the JA method were compared with SA, FFA-LS, SQP MATLAB algorithms, etc.

A comprehensive performance evaluation of metaheuristic algorithms is carried out by Pholdee and Bureerat (2017) on the size optimum design of truss systems. A total of eighteen different metaheuristic algorithms are considered and their solutions to 10-bar cantilever truss, 25-bar tower truss, 72-bar space truss, and 200-bar plane truss systems are compared with each other.

Assimi et al. (2018) used Genetic Programming with a new adaptive mutant operator for size and topology optimization of truss structures. The efficiency of the proposed method was determined using three numerical examples; namely 10-bar truss, 25-bar truss, and 56-bar truss.

In Sonmez (2018), the performances of eight different population-based metaheuristic algorithms are compared in the optimum design of space truss designs. The optimum solutions of the 10-bar cantilever plane truss and 582-bar tower truss problems are sought using these algorithms, and the results obtained are compared with each other to determine the most successful metaheuristic algorithms.

Multi-objective Colliding Bodies Algorithm was employed by Kaveh and Mahdavi (2018) for the design optimization of the truss systems. This algorithm is an extension of the Colliding Bodies Optimization Algorithm for multi-objective optimization problems. Together with some mathematical functions, the 120-bar truss dome and 582-bar truss tower problems were used to verify the efficiency of the optimization algorithm.

Jafari et al. (2018) proposed a hybrid algorithm based on the integration of the Cultural Algorithm (CA) into the Elephant Herding Optimization (EHO) method. A problem suit consisting of separately eight mathematical problems and four steel

truss structures (namely, 10-bar, 25-bar, 72-bar and 120-bar truss systems) were used to examine the numerical performance of the new hybrid algorithm EHOC (Elephant Herding Optimization Cultural) as well as individual main algorithms (CA) and (EHO). The results obtained with these three methods were also compared with the previously reported solutions to these problems using PSO, Multi-Stage Particle Swarm Optimization (MSPSO), and HPSSO algorithms.

An advanced version of the Jaya Algorithm was proposed and employed by Degertekin et al. (2019) for discrete sizing, layout, and topology optimization of truss systems. This algorithm, named Discrete Advanced Jaya Algorithm (DAJA), allows for producing discrete solutions to structural optimization problems. The efficiency of the proposed method was investigated using relatively a large set of test problems; namely, the 10-bar planar truss, 25-bar spatial tower, 47-bar truss tower, 72-bar spatial structure, 200-bar planar system, and 942-bar spatial structure. The optimum designs produced for these problems with the DAJA algorithm were compared with the results of the previous studies.

In Jawad et al (2021), Artificial Bee Colony Algorithm is implemented for sizing and layout optimization of truss systems. A problem suit consisting of 15-bar, 18-bar, 25-bar, and 47-bar truss systems is used to evaluate the performance of the algorithm as well as to compare the results with those of PSO, Cellular Automata hybridized with Particle Swarm Optimization (CPSO), GA, Group Search Optimizer (GSO), and Improved Group Search Optimizer (IGSO) methods.

Awad (2021) used the Political Optimizer algorithm for the size optimization of truss systems. The performance of the algorithm was identified using a large problem set, including the 10-bar cantilever truss, 18-bar cantilever truss, 200-bar planar truss, 22-bar cantilever truss, 25-bar tower truss, 72-bar truss, and 942-bar tower. The optimum solutions attained with the Political Optimizer algorithm were compared to the previously published results of these problems using SA, ES, Genetic-Nelder Mead Simplex (GNMS), FA, Cuckoo Search (CS), Grey Wolf Optimizer (GWO),

Improved Grey Wolf Optimizer (IGWO), JA, Flower Pollination Algorithm (FPA), and TLBO algorithms.

Liu J. and Xia Y. (2022) employed the Genetic Algorithm integrated with a Deep Learning Neural Network for the optimization of truss systems. In this algorithm, named Hybrid Intelligent Genetic Algorithm, Deep Learning Neural Network is used to substantially improve the computational efficiency of the optimization process with the Genetic Algorithm. The 10-bar cantilever truss, 25-bar tower truss, and 37-bar bridge truss were designed for the minimum weight using this algorithm, and the results obtained were compared to those of GA to verify the accuracy of the approximation through Deep Learning Neural Network.

CHAPTER 4

OPTIMIZATION METHODS

In this thesis, recently developed five metaheuristic search methods are examined in terms of their performances in the sizing optimization of truss structures. These algorithms refer to Atomic Orbital Search, Honey Badger Algorithm, Nuclear Fission-Nuclear Fusion Algorithm, PathFinder Algorithm, and Salp Swarm Algorithm. In this chapter, these methods are briefly introduced and their algorithms are explained in the following sections.

4.1 Atomic Orbital Search (AOS) Optimization Algorithm

The main idea of Atomic Orbital Search (AOS) lies in the quantum atomic model and some principles of quantum mechanics. The difference between the quantum-based model and the classical atom model is that in the classical atom model, the probability of electrons' locations is specific and the orbitals are accepted as a region around the nucleus, whereas in the quantum atom model, the probability of electrons' locations is varied by the energy level of each electron particle. The electron movement around the atomic nucleus will be seen as a wave when photographed with time exposure . All of these are illustrated in Figure 4.1, where Figure 4.1A is the classical atomic sketch; Figure 4.1B is the possible electron location; Figure 4.1C is the probability density versus distance between electron and nucleus; Figure 4.1D is the layer of the possible locations of electrons; and finally Figure 4.1E is the distribution of the radial probability of the locations of the electrons. Every electron layer of this quantum model has a different energy level and the positions of electrons change with their energy levels.

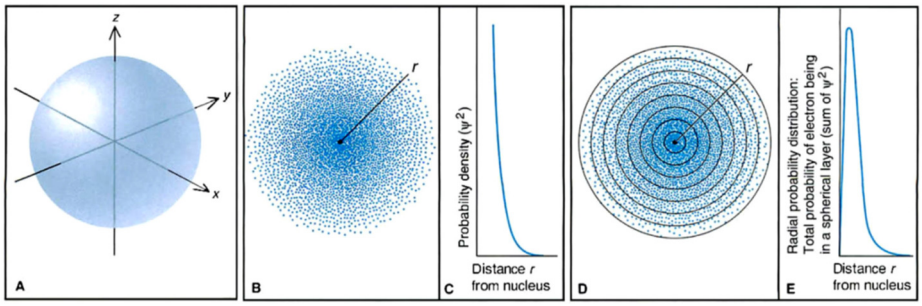

Figure 4.1. Electron motion models (Azizi, 2021).

4.1.1 Mathematical Model of AOS Algorithm

The mathematical model of the AOS algorithm is based on the orbital principles of the quantum atomic theory, which explains the location change of electrons by energy absorption or energy emission.

In this model, the initial values of all the candidates are determined randomly as given in equation 4.1, where i is the current number of the search agent and j is the current number of design variables, and $X_i^j(0)$ is the initial value of the variable j in the search agent (candidate) i.

$$x_i^j(0) = x_{i,min}^j + rand * \left(x_{i,max}^j - x_{i,min}^j\right) \qquad (4.1)$$

The second step is the evaluation of these initial candidates and the calculation of their objective functions. In this atomic model, there are some imaginary layers at every iteration and X^k and E^k are used to define the position and the objective function value at the kth imaginary layer, respectively.

Prior to evaluation, the candidates define the lowest energy level in each layer (LE^k) and calculate the binding state (BS^k) and the binding energy (BE^k) at the kth layer using equations 4.2 and 4.3. In these equations, p is the total number of candidates at the kth layer.

$$BS^k = \frac{\sum_{i=1}^{p} x_i^k}{p} \qquad (4.2)$$

$$BE^k = \frac{\sum_{i=1}^{p} E_i^k}{p} \quad (4.3)$$

The binding state and binding energy values are calculated for all the search agents when the objective functions are calculated for all candidates using equations 4.4 and 4.5. In these equations, m is the total number of candidates at whole search agents.

$$BS = \frac{\sum_{i=1}^{m} X_i}{m} \quad (4.4)$$

$$BE = \frac{\sum_{i=1}^{m} E_i}{m} \quad (4.5)$$

In this method, there is a parameter called Photon Rate and this parameter is initially set to 0.1 at the beginning of the optimization process. A random number (ϕ) is generated for every candidate and if it is greater than the Photon Rate, the candidate will continue to progress in the first path. In this first path, a comparison is carried out between Binding Energy and Objective Function. If Binding Energy is greater than Objective Function for a candidate, this candidate is improved using equation 4.6, otherwise, equation 4.7 is used instead.

$$X_{i+1}^k = X_i^k + \frac{\alpha_i * (\beta_i * LE - \gamma_i * BS)}{k} \quad (4.6)$$

$$X_{i+1}^k = X_i^k + \alpha_i * (\beta_i * LE^k - \gamma_i * BS^k) \quad (4.7)$$

If the random number ϕ is less than the Photon Number, then the second path is followed in which case the candidate is improved using equation 4.8.

$$X_{i+1}^k = X_i^k + r_i \quad (4.8)$$

The pseudo-code of the Atomic Orbital Search Algorithm is shown in Figure 4.2.

Procedure *Atomic Orbital Search (AOS) Algorithm*
Determine initial positions of solution candidates (X_i) in the search space with m candidates
Evaluate fitness values (E_i) for initial solution candidates
Determine binding state (BS) and binding energy (BE) of atom
Determine candidate with lowest energy level in atom (LE)
while *Iteration < Maximum number of iterations*
Generate n as the number of imaginary layers
Create imaginary layers
Sort solution candidates in an ascending or descending order
Distribute solution candidates in the imaginary layers by PDF
for *k=1: n*
Determine binding state (BS^k) and binding energy (BE^k) of the kth layer
Determine the candidate with lowest energy level in the kth layer (LE^k)
for *i=1:p*
Generate $\varphi, \alpha, \beta, \gamma$
Determine PR
if $\varphi \geq PR$
if $E_i^k \geq BE^k$
$X_{i+1}^k = X_i^k + \frac{\alpha_i \times (\beta_i \times LE - \gamma_i \times BS)}{k}$
else if $E_i^k < BE^k$
$X_{i+1}^k = X_i^k + \alpha_i \times (\beta_i \times LE^k - \gamma_i \times BS^k)$
end
else if $\varphi < PR$
$X_{i+1}^k = X_i^k + r_i$
end
end
end
Update binding state (BS) and binding energy (BE) of atom
Update candidate with lowest energy level in atom (LE)
end while
end Procedure

Figure 4.2. The AOS algorithm's pseudo code (Azizi, 2021)

4.2 Honey Badger Algorithm

The Honey Badger Algorithm is based on the movement of a honey badger swarm, which has an intelligent technique for searching food. This swarm behaviour with its effective search strategy is helpful to solve optimization problems and has two dynamic steps called digging and finding honey. The mathematical model of this algorithm consists of two phases as exploration and exploitation, developed to create an effective search strategy for arriving at the optimum solution,

The Swarm of the Honey Badger searches for food in two different methods. The first method is smelling around the possible food or prey location, and digging into this area. The second method is following the honey birds, which already found the

beehives but did not reach them. The Honey Badgers go after these birds and open the beehives with their claws, and then they eat this honey together with these birds.

4.2.1 Mathematical Model of Honey Badger Algorithm

The mathematical model of the Honey Badger Optimization (HBO) algorithm is inspired by the searching food behaviours of a Honey Badgers Swarm. The Honey Badger Optimization algorithm has two steps known as exploration and exploitation, similar to digging a possible food area and following the honey birds.

The initial values of all the candidates are determined randomly using equation 4.9, where i is the current number of design variables, and x_i is the initial value of the *i*th variable in each candidate.

$$x_i = lb_i + r_1 * (ub_i - lb_i) \quad (4.9)$$

The second step is the calculation of the intensity of smelling for the prey, and this is performed using equation 4.10, where d is defined as the distance between the current candidate badger and the leader of this swarm, as formulated in equation 4.11. The other variable of the intensity equation S, which is the strength of the food source, is shown in equation 4.12.

$$I_i = r_2 * \frac{S}{4*\pi*d_i} \quad (4.10)$$

$$d_i = x_{prey} - x_i \quad (4.11)$$

$$S = (x_i - x_{i+1})^2 \quad (4.12)$$

The third step is defining the factor of density, which decreases throughout the iteration. This factor is used for a soft transition between exploration and exploitation and is calculated using equation 4.13, where t_{max} is the maximum number of iterations, t is the current iteration number and C is a constant, which must be set to a value equal to or greater than 1.

$$\alpha = C * exp\left(\frac{-t}{t_{max}}\right) \quad (4.13)$$

The main phase is updating the candidates' location with the following two separate paths, which are digging and honey paths. Both of these paths use a flag F, which protects this algorithm from stagnation in a local optimum. This flag F is related to a random number r_6, and it manages the improving direction of the candidates' location (-1 or 1) as formulated in equation 4.14.

$$F = \begin{cases} 1, & if \ \ r_6 \leq 0.5 \\ -1 & else \end{cases} \quad (4.14)$$

At the digging path, equation 4.15 is used, which imitates the cardioid motion of the Honey Badger behaviour. At the honey path, equation 4.16 is used, which imitates the chase of the honey birds to find the beehives.

$$x_{new} = x_{prey} + F * \beta * I * x_{prey} + F * r_3 * \alpha * d_i * |\cos(2\pi r_4) * [1 - \cos(2\pi r_5)]| \quad (4.15)$$

$$x_{new} = x_{prey} + F * r_7 * \alpha * d_i \quad (4.16)$$

In equations (4.14) and (4.15), x_{prey} is the location of the leader candidate; r_3, r_4, r_5, and r_7 are random numbers between 0 and 1; α is a factor of density; I is intensity; and β is an ability to reach the food and it must be greater than 1.

The pseudo-code of the Honey Badger Algorithm is shown in Figure 4.3.

Algorithm 1 Pseudo code of HBA.

Set parameters t_{max}, N, β, C.
Initialize population with random positions.
Evaluate the fitness of each honey badger position x_i using objective function and assign to f_i, $i \in [1, 2, ..., N]$.
Save best position x_{prey} and assign fitness to f_{prey}.
while $t \leq t_{max}$ **do**
  Update the decreasing factor α using (3).
  for $i = 1$ to N **do**
    Calculate the intensity I_i using Eq. (2).
    if $r < 0.5$ **then** ▷ r is random number between 0 and 1
      Update the position x_{new} using Eq. (4).
    else
      Update the position x_{new} using Eq. (6).
    end if
    Evaluate new position and assign to f_{new}.
    if $f_{new} \leq f_i$ **then**
      Set $x_i = x_{new}$ and $f_i = f_{new}$.
    end if
    if $f_{new} \leq f_{prey}$ **then**
      Set $x_{prey} = x_{new}$ and $f_{prey} = f_{new}$.
    end if
  end for
end while Stop criteria satisfied.
Return x_{prey}

Figure 4.3. Pseudo-code of Honey Badger Algorithm (Hashim et al, 2021)

4.3 Nuclear Fission-Nuclear Fusion (N2F) Algorithm

The Nuclear Fission-Nuclear Fusion (N2F) Algorithm is an improved variant of the Big Bang-Big Crunch (BB-BC) Algorithm. This algorithm has features to avoid the search from unnecessary exploration of local minimums and also from jumping to ineffective regions of the design space. N2F Algorithm has two phases, which are named Nuclear Fission and Nuclear Fusion.

The first phase is Nuclear Fission, which is similar to the Big Bang step in the BB-BC algorithm. The task of this phase is to prevent the search steps from getting trapped at a local optimum solution. The second phase is Nuclear Fusion, which is similar to the Big Crunch step in the BB-BC algorithm. In this method, there is a magnification factor, which has an effective ability to exploit the design space.

4.3.1 Mathematical Model of N2F Optimization Algorithm

The Mathematical model of the N2F optimization algorithm imitates the physical phenomenon of Nuclear Fission-Nuclear Fusion. The N2F algorithm has been devised in a way to eliminate certain drawbacks encountered in the original BB-BC algorithm.

The first step in the implementation of the N2F optimization algorithm is to set the ρ and μ parameters and to randomly generate initial candidates of search agents using equation 4.17. The ρ parameter should be chosen between 1.0 and 3.0, and the μ parameter is set to a value between 10^0 and 10^{20}. The second step is to calculate the objective functions for all the candidates and then find the best result.

$$x_i = lb_i + rand * (ub_i - lb_i) \tag{4.17}$$

The third step is the Nuclear Fusion phase, where the location of the mass center (x_c) is determined using equations 4.18 and 4.19.

$$I^k = \left(\frac{f_{best}}{f^k}\right)^{\rho^{m-1}} \tag{4.18}$$

$$x_c = \frac{\sum_{k=1}^{K} x^k I^k}{\sum_{k=1}^{K} I^k} \tag{4.19}$$

The fourth step of this algorithm is the Nuclear Fission phase, where new candidates are created around the mass center (x_c) at random using equation 4.20. Only the best candidate, which is the fittest member of previous search agents, is still the same.

$$(x^k)_{new} = x_c + N^k * (x_{max} - x_{min}) * \left(\mu^{\frac{-m}{M}}\right) \tag{4.20}$$

4.4 Pathfinder Algorithm

Pathfinder Algorithm is another meta-heuristic algorithm investigated in this study. The algorithm is based on the collected movement patterns of an animal swarm to reach food. The algorithm mimics the searching, exploiting, and hunting behaviours

of an animal swarm. The leader must carry the swarm to the food or water resources. So, the leader is not a constant member of a swarm; instead, it is the member of the swarm which has the best movement of the current step. Therefore, the leader can change at every step on the way to the source path (Yapici et al, 2019).

In this algorithm, an interactive hierarchy is used in the social movement of the swarm, such as collective movement and hierarchic foraging. An example of the collective movement is shown in Figure 4.4

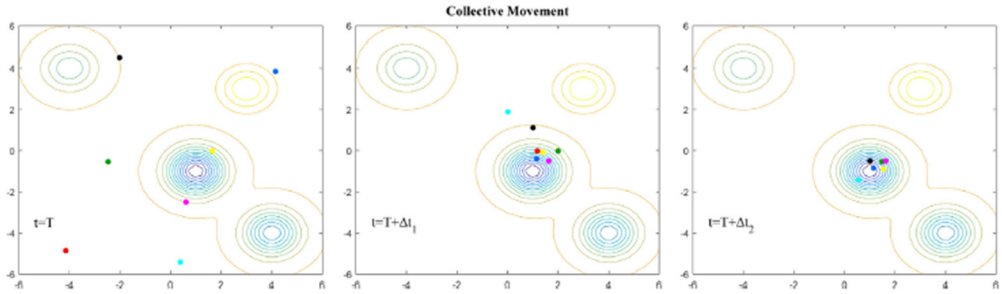

Figure 4.4. Sample collective movement of Pathfinder Algorithm (Yapici et al, 2019).

4.4.1 Mathematical Model of Pathfinder Algorithm

The Pathfinder algorithm's main improvement idea is the interactive leadership in the swarm, which means that any candidate which has the best result at the current iteration will be the leader of the swarm for that iteration.

The initial values of all candidates are determined randomly using equation 4.21, where i is the current number of design variables, and x_i is the initial value of the i-th variable in a candidate.

$$x_i = lb_i + rand * (ub_i - lb_i) \qquad (4.21)$$

The second step is to evaluate these initial candidates and calculate the objective functions for all candidates. The candidate, which has the best result in an iteration, will be the leader of the swarm at the next iteration.

After evaluation, all candidate locations are updated using two separate equations. Equation 4.22 is related to the current leader of this swarm, and equation 4.23 is for the followers.

$$x_p^{K+1} = x_p^K + 2 * r_3 * \left(x_p^K - x_p^{K-1}\right) + A \qquad (4.22)$$

$$x_i^{K+1} = x_i^K + R_1 * \left(x_j^K - x_i^K\right) + R_2 * \left(x_p^K - x_i^K\right) + \varepsilon \qquad (4.23)$$

In equations 4.22 and 4.23, R_1 is equal to $\alpha * r_1$, and R_2 is equal to $\beta * r_2$. The α and β values are chosen randomly between 1 and 2 for each candidate at the beginning of each iteration, and they affect the size steps for improving the candidates. r_1, r_2, and r_3 are random numbers between 0 and 1. The ε and A are calculated using the equations 4.24 and 4.25, respectively.

$$\varepsilon = \left(1 - \frac{t}{t_{max}}\right) * u_1 * D_{ij} \quad , D_{ij} = \left\|x_i - x_j\right\| \qquad (4.24)$$

$$A = u_2 * e^{\frac{-2K}{K_{max}}} \qquad (4.25)$$

In equations 4.24 and 4.25, u_1 and u_2 are vectors defined randomly between -1 and 1, t_{max} is the maximum iteration number, and t is the current iteration number. The pseudo-code of the Pathfinder Algorithm is shown in Figure 4.5.

```
Load PFA parameter
Initialize the population
Calculate the fitness of initial population
Find the pathfinder
while K < maximum number of iterations
    α and β = random number in [1,2]
    update the position of pathfinder using Equation (2.4) and check the bound
    if new pathfinder is better than old
        update pathfinder
    end
    for i=2 to maximum number of populations
        update positions of members using Equation (2.3) and check the bound
    end
    calculate new fitness of members
    find the best fitness
    if best fitness < fitness of pathfinder
        pathfinder = best member
        fitness = best fitness
    end
    for i=2 to maximum number of populations
        if new fitness of member (i) < fitness of member (i)
            update members
        end
    end
    generate new A and ε
end
```

Figure 4.5. The Pseudo code of the Pathfinder Algorithm (Yapici et al, 2019)

4.5 Salp Swarm Algorithm

Salp Swarm Algorithm was inspired by the behavior of the salp swarms. Since the living environments of salps are oceans, and they cannot be kept in a laboratory environment, it is very hard to understand their living and foraging behaviors. Salps usually can be seen in a swarm, which is named the salp chain, in deep oceans. This salp chain is illustrated in Figure 4.6. The researchers believe that this behavior model is the best way for achieving foraging and better locomotion using rapid coordinated changes (Mirjalili , 2017).

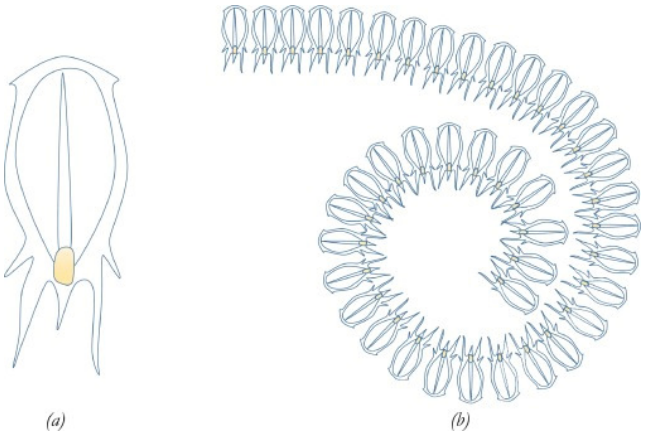

Figure 4.6. (a) Single Salp, (b) Salps Herd (Mirjalili , 2017).

4.5.1 Mathematical Model of Salp Swarm Algorithm

The mathematical model of the Salp Swarm Algorithm (SSA) is based on the searching food behaviours of Salp Swarm. The Salp Swarm Algorithm has two steps known as exploration and exploitation, similar to navigating in deep ocean and finding food.

The initial values of all the candidates are determined randomly using equation 4.26, where i is the current number of design variables. x_i is the initial value of the i-th variable in a candidate.

$$x_i = lb_i + rand * (ub_i - lb_i) \qquad (4.26)$$

The second step is to evaluate these initial candidates and calculate the objective functions of all the candidates. The candidates are sorted according to increasing values of their objective functions; hence the best candidate ranks on the first row. The candidate, which has the minimum objective function at a current iteration, is identified as the best salp.

The most important component of the algorithm is the c_1 parameter, which helps to balance between exploitation and exploration and is calculated using equation 4.27.

$$c_1 = 2 * e^{-\left(\frac{4*l}{L}\right)^2} \qquad (4.27)$$

In this method, the other two components of the algorithm are the c2 and c3 parameters, which are both set to a random value between 0 and 1. If c3 is greater than zero, the location of the current leader salp will be updated using equation 4.28, otherwise, equation 4.29 is used instead.

$$x_j^1 = F_j + c_1 * \left(\left(ub_j - lb_j \right) * c_2 + lb_j \right) \quad (4.28)$$

$$x_j^1 = F_j - c_1 * \left(\left(ub_j - lb_j \right) * c_2 + lb_j \right) \quad (4.29)$$

The follower salps are updated using equation 4.30.

$$x_j^i = \frac{1}{2} * \left(x_j^i + x_j^{i-1} \right) \quad (4.30)$$

The pseudo-code of the Salp Swarm Algorithm is shown in Figure 4.7.

```
Initialize the salp population xi (i = 1, 2, ..., n) considering ub and lb
while (end condition is not satisfied)
Calculate the fitness of each search agent (salp)
F=the best search agent
Update c1 by Eq. (3.2)
    for each salp (xi)
        if (i==1)
            Update the position of the leading salp by Eq. (3.1)
        else
            Update the position of the follower salp by Eq. (3.4)
        end
    end
    Amend the salps based on the upper and lower bounds of variables
end
return F
```

Figure 4.7. The Pseudo code of SSA (Mirjalili , 2017).

4.6 Progress of the Software

In this study, the numerical implementations of the optimization algorithms were coded in MATLAB R2021b program using MATLAB programming language. While solving the benchmark problems, the structural analyses were carried out with the aid of the so-called K-files, in which the responses of structures; i.e., member

forces and displacements are expressed as mathematical functions of cross-sectional areas of truss members. While solving the structural design problems, on the other hand, optimization codes are communicated with the SAP2000 structural analysis software for obtaining response calculations of designs generated in the course of the optimization process. This is done using the Open Application Programming interface available in SAP2000 software, which allows an external use of the structural analysis software by some other programs through the internal functions supported by various programming languages. The optimization algorithms are iterated until one of the following two stopping criteria is satisfied. Accordingly, the algorithms are terminated when a specified maximum number of iterations is reached, and when the best design is not improved over a specified number of successive iterations.

CHAPTER 5

NUMERICAL EXAMPLES

In this chapter, the numerical performances of the investigated metaheuristic optimization algorithms are examined and compared using three benchmark truss problems taken from the literature. In addition, two structural truss design examples that are sized for the minimum weight according to the provisions of LRFD-AISC 2016 design specification are also studied here using the five metaheuristic algorithms in order to observe the performances and convergence characteristics of the algorithms in discrete and challenging optimization problems of engineering design practice.

5.1 Benchmark Test Problems

Three benchmark truss problems taken from the literature are studied here. These problems are the 25-bar truss tower, 38-bar cantilever truss, and 72-bar truss tower. While solving a benchmark problem, each optimization algorithm is run ten times independently to achieve the size optimum design of the structure, and the best feasible weight attained is reported to be the minimum weight design of the structure achieved with each optimization algorithm. All the algorithms are initiated with randomly generated solutions at the beginning of the optimization process in each run. The maximum number of iterations in a single run is limited to 1000, and if the best feasible is not improved over 100 iterations, the algorithm is terminated prematurely.

5.1.1 25- Member Truss

The 25-member spatial truss shown in Figure 5.1 is an electric transmission tower. The truss will be designed for the minimum weight by selecting the truss members from a set of 30 discrete sections; namely 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 3.0, 3.2, 3.4 in^2. The material density is 0.1 lb/in^3 and the Young's Modulus is 10^4 ksi.

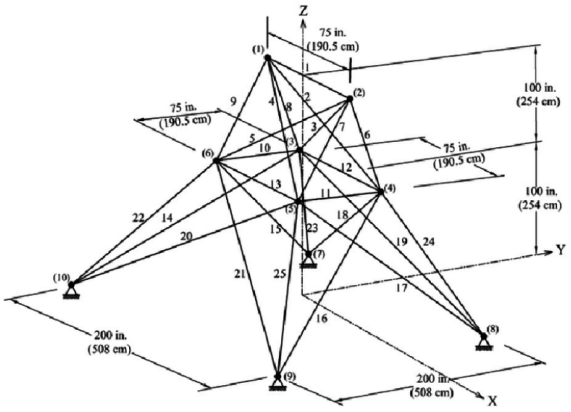

Figure 5.1. A 3-D view of the 25-member truss

This truss tower has 10 nodes and 25 members altogether. The truss members are collected under 8 member groups (design variables), as shown in Table 5.1.

Table 5.1 Member groups of the 25-member truss

Group Name	*Elements*
1	1
2	2-5
3	6-9
4	10-11
5	12-13
6	14-17
7	18-21
8	22-25

The loads applied on the nodes of the truss are summarized in Table 5.2. Both the stress and displacement constraints are imposed on the problem. The upper value of stress on every member is defined as ±40 ksi under tension and compression, and the displacements of nodes 1 and 2 are limited to a maximum value of ±0.35 in any direction.

Table 5.2 Loadings at 25-member truss

Nodes\Direction	*x*	*y*	*z*
1	1.0	-10.0	-10.0
2	0	-10.0	-10.0
3	0.5	0	0
4	0.6	0	0

5.1.1.1 Results of Analyses

The 25-member benchmark truss problem was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms.

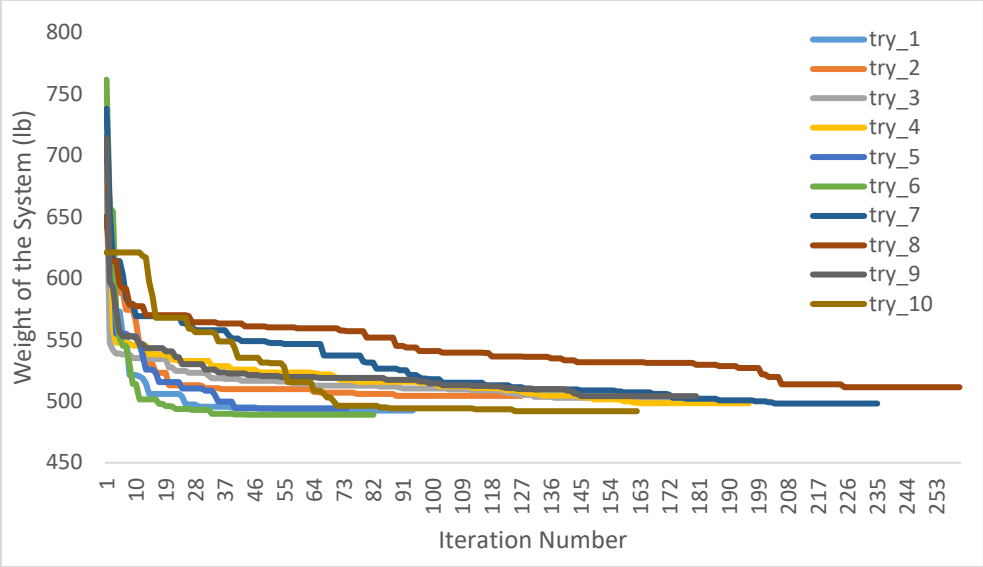

Figure 5.2. Convergence curves for the 25-bar benchmark truss problem with AOS

The best run of the Atomic Orbital Search algorithm has resulted in a design weight of 489,05 lb for the truss, while the optimum design weight in the worst run is 511,46 lb. For all the independent runs performed with the Atomic Orbital Search algorithm, the resulting convergence curves which show the variation of the best feasible design versus the iteration number performed are plotted in Figure 5.2.

The Honey Badger Algorithm has two parameters as Beta and C. These parameters significantly affect the speed of convergence and accuracy of the optimization process. After some trials, Beta and C parameters are chosen as 4.50 and 1.25, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 485,05 lb for the truss, while the worst run leads to a design weight of 492,31 lb. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.3.

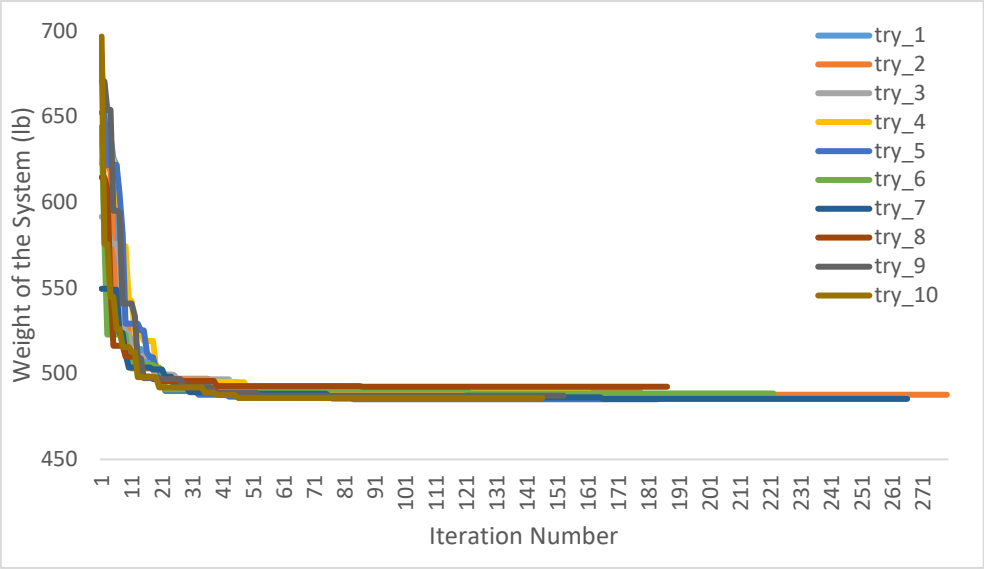

Figure 5.3. Convergence curves for the 25-bar benchmark truss problem with HBA

The μ and ρ parameters are used in Nuclear Fission-Nuclear Fusion Algorithm to establish a balance between searching around the design space and searching near the current best result. After trials, μ and ρ parameters are chosen as 10^{14} and 1.10

respectively. The best run of the Nuclear Fission-Nuclear Fusion Algorithm has resulted in a design weight of 485,05 lb for the truss, while the worst run leads to a design weight of 496,58 lb. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion Algorithm, the resulting convergence curves are plotted in Figure 5.4.

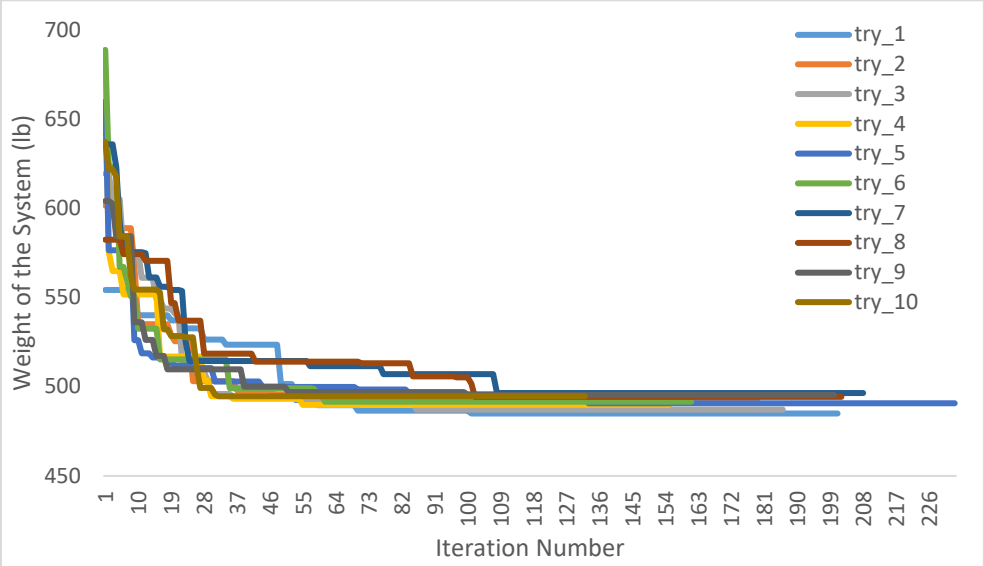

Figure 5.4. Convergence curves for the 25-bar benchmark truss problem with N2F

The Pathfinder Algorithm does not involve any parameter which requires an appropriate setting prior to the start of the optimization process. The best run of the Pathfinder Algorithm has resulted in a design weight of 486,96 lb for the truss, while the worst run leads to a design weight of 496,72 lb. For all the independent runs performed with the Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.5.

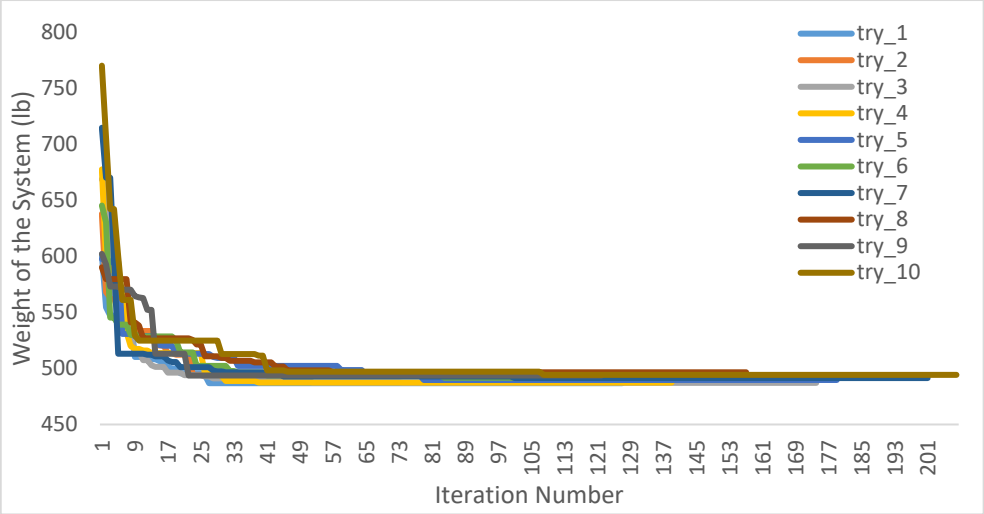

Figure 5.5. Convergence curves for the 25-bar benchmark truss problem with PFA

Similar to the Pathfinder Algorithm, the Salp Swarm Algorithm also does not involve any parameter which requires an appropriate setting prior to the start of the optimization process. In the runs performed here, the c1 parameter is multiplied by 0.1 here. The best run of the Salp Swarm Algorithm has resulted in a design weight of 485,35 lb for the truss, while the worst run leads to a design weight of 489,04 lb. For all the independent runs performed with the Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.6.

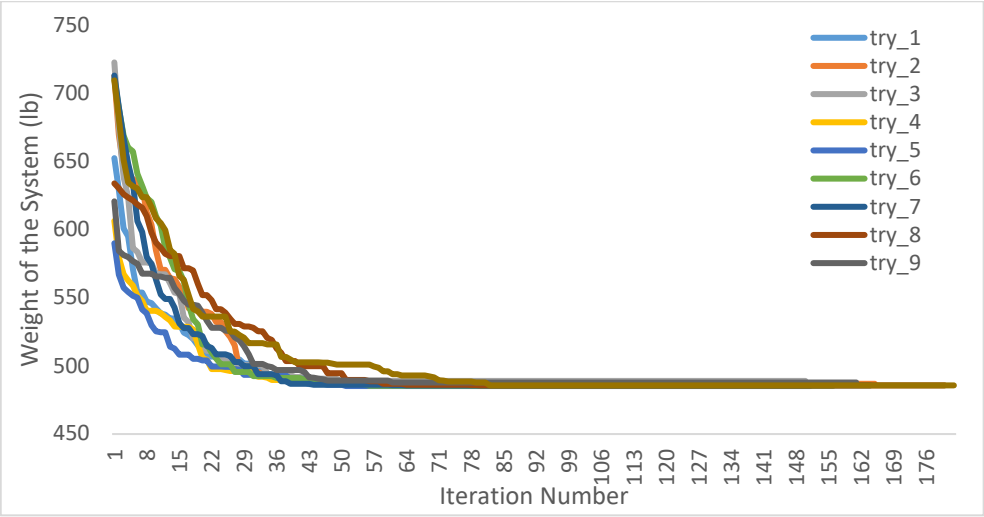

Figure 5.6. Convergence curves for the 25-bar benchmark truss problem with SSA

25-bar benchmark truss problem has formerly been studied by Hasançebi et al. (2009) using a large number of meta-heuristic search algorithms. The optimum designs of the problem attained with different optimization algorithms in that study are reproduced in Table 5.3 in terms of cross-sectional areas, the truss weight, and the number of analyses performed. Accordingly, the minimum weight of the truss is 484,85 lb, which is identified by the HS and SA algorithms by performing 2.100 and 6.624 structural analyses, respectively.

The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.3 in terms of cross-sectional areas, the truss weight, and the number of analyses performed. As can be seen from this table, the minimum weight of the truss, which is achieved in this study, is 485,05 lb. This solution has been attained by both the Honey Badger Algorithm and Nuclear Fission-Nuclear Fusion Algorithm by performing 9.200 and 10.050 structural analyses, respectively.

Table 5.3 Summary of the optimum results for the 25-bar benchmark truss problem

Design Variable		Best cross-section areas (in^2)									
		Hasançebi et al. (2009)					This study				
		HS	**SA**	**AC**	**SGA**	**TS**	**AOS**	**HBA**	**N2F**	**PFA**	**SSA**
1	A_1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1
2	A_2~A_5	0.3	0.3	0.5	0.2	0.4	0.5	0.5	0.5	0.2	0.7
3	A_6~A_9	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4
4	A_{10}~A_{11}	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
5	A_{12}~A_{13}	2.1	2.1	1.9	2.0	1.8	2.6	1.9	1.9	1.6	1.7
6	A_{14}~A_{17}	1.0	1.0	1.0	1.0	0.9	1.1	0.9	1.0	1.1	0.9
7	A_{18}~A_{21}	0.5	0.5	0.4	0.6	0.6	0.2	0.5	0.4	0.6	0.4
8	A_{22}~A_{25}	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4	3.4
Weight(lb)		484.85	484.85	485.05	485.38	485.57	489.05	485.05	485.05	487	485.3
Analysis number		2100	6624	10050	9050	1626	4100	9200	10050	6350	7600

5.1.2 38- Member Truss

The 38-member cantilever truss shown in Figure 5.7 is a test problem used in a student contest called "the International Student Competition in Structural Optimization" held in 2012. The cantilever truss consists of 21 nodes and 38 members altogether, and it will be designed for the minimum weight by selecting the truss members from a set of discrete sections; namely (0.1, 0.2, 0.3, …, 14.8, 14.9, 15.0 in^2). No member grouping is carried out for the truss elements. The material density is 0.283 lb/in^3 and the Young's Modulus is 30,000 ksi. The truss is subjected to a single load of P= -15 kips applied at node 21 in the y-direction only.

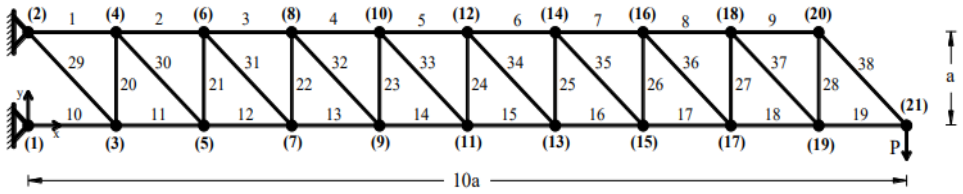

Figure 5.7. 38-member cantilever truss

Both the stress and displacement constraints are imposed on the problem. The upper value of stress on every member is defined as ±30 ksi under tension and compression, and the displacements of all nodes are limited to a maximum value of ±4.0 in both x and y-directions.

5.1.2.1 Results of Analyses

The 38-member cantilever truss problem was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms. The independent runs performed with the Atomic Orbital Search algorithm have led to an optimum design weight of the structure in the range between 6005,09 lb. and 6237,83 lb, and the algorithm has terminated at iteration numbers

between 450 and 960. For all the independent runs performed with the Atomic Orbital Search algorithm, the resulting convergence curves are plotted in Figure 5.8.

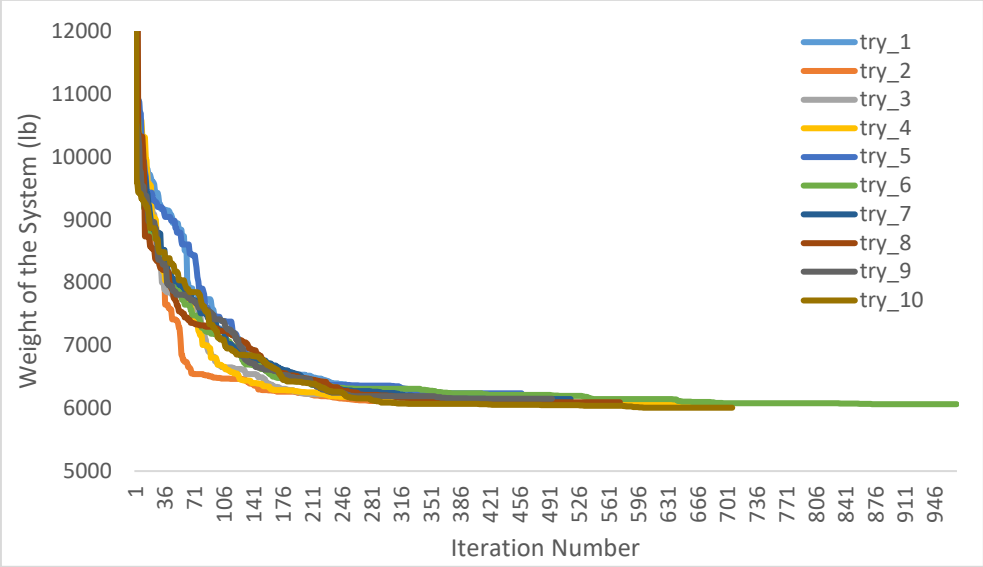

Figure 5.8. Convergence curves for the 38-bar cantilever truss problem with AOS

While solving the 38-member cantilever truss problem with the Honey Badger Algorithm, after some trials, the Beta and C parameters are chosen as 2.10 and 1.25, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 5975,09 lb for the truss, while the worst run leads to a design weight of 6252,30 lb. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.9.

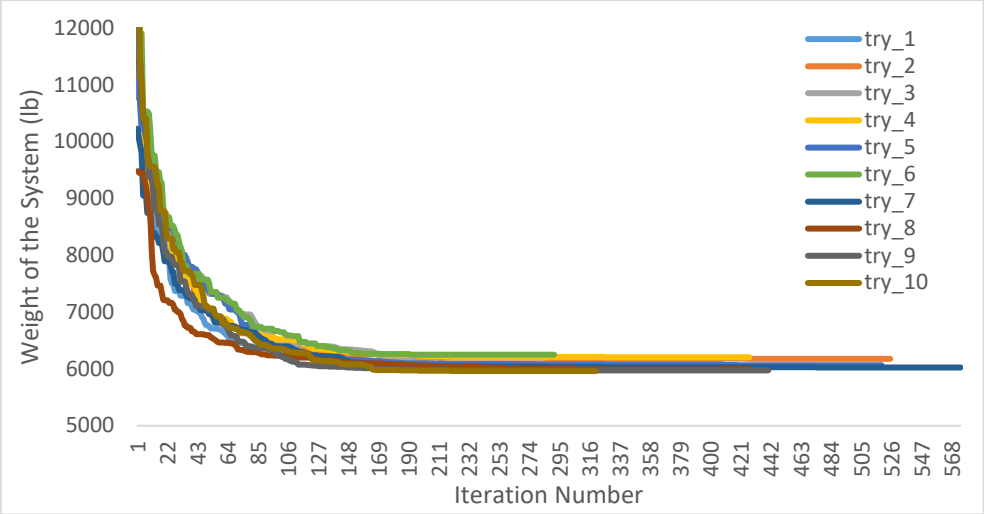

Figure 5.9. Convergence curves for the 38-bar cantilever truss problem with HBA

Nuclear Fission-Nuclear Fusion Algorithm has been implemented such that the μ and ρ parameters are set to 10^{15} and 1.13 respectively. The best run of the Nuclear Fission-Nuclear Fusion Algorithm has resulted in a design weight of 6038,52 lb for the truss, while the worst run leads to a design weight of 6439,08 lb, and the algorithm has terminated at iteration numbers between 250 and 570. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion Algorithm, the resulting convergence curves are plotted in Figure 5.10.

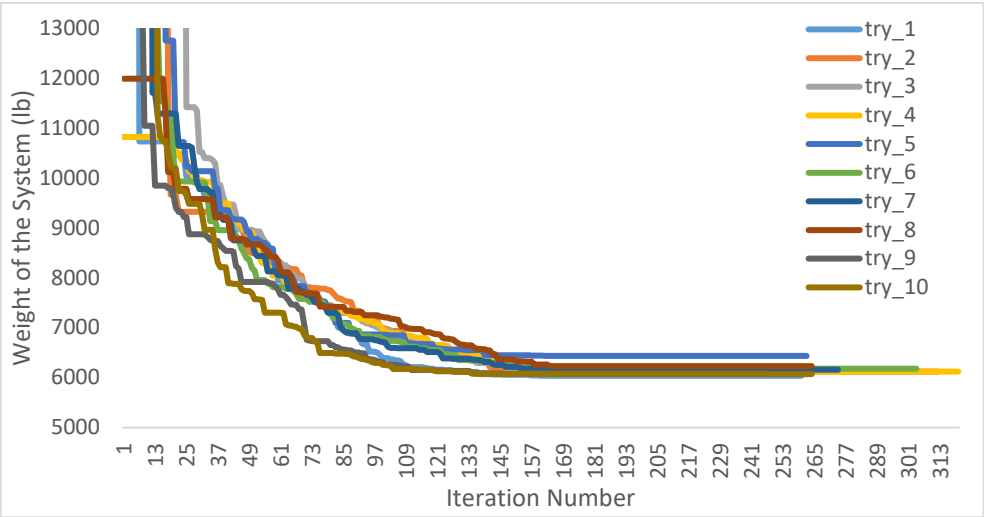

Figure 5.10. Convergence curves for the 38-bar cantilever truss problem with N2F

A standard formulation of The Pathfinder Algorithm could not find any feasible solution for the 38-bar cantilever truss problem. Therefore, for this example, a modification of the algorithm is performed as formulated in equation 5.1 to remedy its search performance.

$$p_{new} = p_{old} + \left((alpha * r1) * \left(p_{global} - p_{old}\right)\right) + \left((beta * r2) * \left(p_{old}^{i} - p_{old}^{i-1}\right)\right) + \begin{cases} (eps * d), & original \\ (0.5 * eps * d), & modified \end{cases} \quad (5.1)$$

The independent runs performed with the modified (improved) Pathfinder Algorithm have led to an optimum design weight of the structure in the range between 7094,79 lb and 8666,37 lb, and the algorithm has terminated at iteration numbers between 200 and 650. For all the independent runs performed with the modified Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.11.

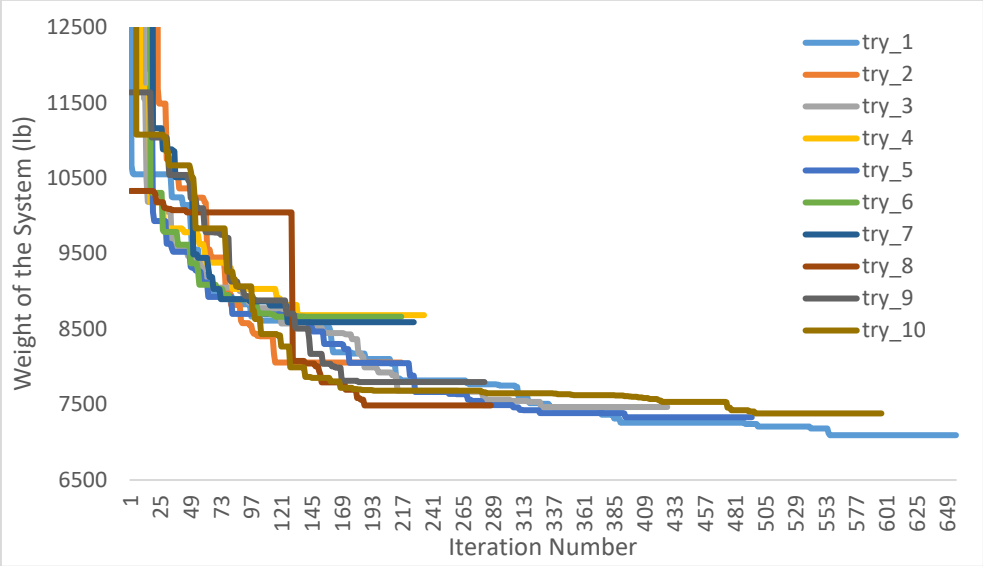

Figure 5.11. Convergence curves for the 38-bar cantilever truss problem with PFA

In the runs performed with the Salp Swarm Algorithm, the c1 parameter is multiplied by 0.15 here. The best run of the Salp Swarm Algorithm has resulted in a design weight of 5959,89 lb for the truss, while the worst run leads to a design weight of 6004,97 lb, and the algorithm has terminated at iteration numbers between 265 and 375. For all the independent runs performed with the Salp Swarm Algorithm, the resulting convergence curves are plotted in Figure 5.12.

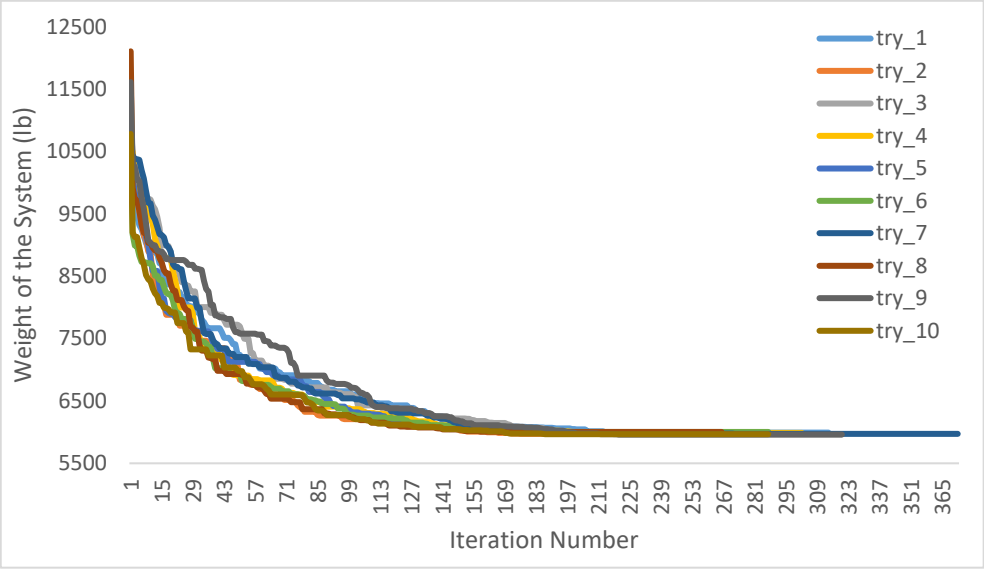

Figure 5.12. Convergence curves for the 38-bar cantilever truss problem with SSA

The 38-member cantilever truss problem has formerly been studied by Kazemzadeh Azad et al. (2016) using the standard BB-BC, modified BB-BC, and exponential BB-BC algorithms. The optimum designs of the problem attained with these algorithms are 5889,99 lb by the standard BB-BC, and 5891,16 lb by both modified BB-BC and exponential BB-BC. These optimum designs are reproduced in Table 5.4 in terms of cross-sectional areas and the truss weight.

The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.4. As can be seen from this table, the minimum weights of the truss achieved with the Salp Swarm Algorithm and Honey Badger Algorithm are comparatively lesser than those of other

implemented metaheuristic techniques, yet they are slightly higher than the solutions attained with the standard BB-BC, modified BB-BC, and exponential BB-BC algorithms as reported in Kazemzadeh Azad et al. (2016).

Table 5.4 Summary of the optimum results for the 38-bar cantilever truss problem

Design Variable		Best cross-section areas of each algorithm							
		Kazemzadeh Azad et al (2016)			This study				
		BB-BC Standard	**BB-BC Modified**	**BB-BC Exponential**	**AOS**	**HBA**	**N2F**	**PFA**	**SSA**
1	A_1	14.6	14.5	14.6	13.0	15.0	14.9	15.0	14.1
2	A_2	12.9	12.8	12.7	12.6	12.9	11.6	15.0	12.4
3	A_3	11.3	11.6	11.4	11.0	11.6	10.7	15.0	11.7
4	A_4	9.8	9.6	9.6	10.1	9.0	8.1	5.9	9.0
5	A_5	8.1	8.2	8.2	8.3	8.4	7.1	12.9	7.9
6	A_6	6.5	6.7	6.4	5.0	7.1	7.2	10.0	6.9
7	A_7	4.9	4.8	4.9	5.6	5.2	5.6	5.0	4.1
8	A_8	3.2	3.2	3.3	2.9	4.5	4.6	6.2	3.9
9	A_9	1.6	1.6	1.6	2.1	2.1	2.3	4.7	2.1
10	A_{10}	15.0	15.0	15.0	14.1	15.0	15.0	12.1	15.0
11	A_{11}	14.6	14.7	14.9	14.8	14.0	14.7	12.1	13.5
12	A_{12}	13.1	13.0	13.0	12.7	12.6	13.9	10.7	12.0
13	A_{13}	11.3	11.2	11.4	13.7	11.2	11.8	13.3	11.1
14	A_{14}	9.8	9.9	9.5	9.7	9.3	12.2	5.4	11.4
15	A_{15}	8.2	8.3	8.1	7.8	8.3	6.2	5.1	8.3
16	A_{16}	6.5	6.3	6.6	5.3	6.4	6.4	12.6	6.1
17	A_{17}	4.9	4.8	4.8	6.1	4.7	5.3	2.4	5.5
18	A_{18}	3.3	3.3	3.2	3.1	3.3	3.2	7.8	3.2
19	A_{19}	1.6	1.5	1.5	1.8	1.8	2.1	5.4	1.9
20	A_{20}	1.6	1.6	1.7	1.2	1.7	1.5	3.8	1.4
21	A_{21}	1.6	1.6	1.6	1.9	0.7	2.9	2.4	2.3
22	A_{22}	1.6	1.6	1.6	1.4	1.5	1.8	2.6	2.0
23	A_{23}	1.6	1.6	1.7	2.0	1.5	1.1	4.8	1.6
24	A_{24}	1.6	1.6	1.6	1.6	2.1	2.1	4.7	1.8
25	A_{25}	1.6	1.7	1.6	1.8	1.5	3.2	4.4	1.3
26	A_{26}	1.6	1.6	1.7	1.5	2.9	1.3	3.3	1.2
27	A_{27}	1.6	1.6	1.7	2.4	1.8	1.2	1.8	1.9
28	A_{28}	1.6	1.6	1.6	1.5	1.8	2.0	2.6	1.5
29	A_{29}	2.3	2.3	2.4	2.2	2.5	2.7	4.2	3.0
30	A_{30}	2.2	2.3	2.3	3.1	2.3	2.3	3.1	2.2
31	A_{31}	2.3	2.4	2.3	3.1	2.2	2.2	1.8	2.6
32	A_{32}	2.3	2.3	2.3	2.3	2.4	2.4	3.7	2.1
33	A_{33}	2.3	2.3	2.3	3.0	2.6	1.9	3.2	2.9
34	A_{34}	2.3	2.3	2.3	2.1	2.2	2.2	2.3	2.3
35	A_{35}	2.3	2.3	2.3	2.8	2.3	2.9	4.5	2.1
36	A_{36}	2.3	2.3	2.3	3.2	2.1	2.5	2.3	2.6
37	A_{37}	2.3	2.3	2.3	2.4	2.0	2.1	2.5	3.0
38	A_{38}	2.4	2.3	2.3	2.1	2.9	2.4	3.3	2.3
Weight(lb)		5889.99	5891.16	5891.16	6005.09	5975.09	6038.52	7094.79	5959.89

5.1.3 72-Member Truss

The 72-member space truss is a rectangular tower as shown in Figure 5.13. The truss consists of 20 nodes and 72 members altogether. The material density is 0.1 lb/in^3 and the Young's Modulus is 10,000 ksi. The truss tower is subjected to two different loading conditions. In the first load case, node 17 is subjected to a load of 5.0 kips in the x-direction, 5.0 kips in the y- direction, and -5.0 kips in the z-direction (Table 5.5), whereas in the second load case, a load of -5.0 kips is applied to the nodes 17, 18 and 19 in the z-direction (Table 5.6).

Table 5.5 The first loading case for the 72-member truss tower

Node	*x-direction*	*y-direction*	*z-direction*
17	5.0	5.0	-5.0

Both the stress and displacement constraints are imposed on the problem. The upper value of stress on every member is defined as ±25 ksi under tension and compression, and the displacements of the top nodes are limited to a maximum value of ±0.25 in all directions. The truss elements are collected under 16 member groups (sizing design variables) as given in Table 5.7.

Table 5.6 The second loading case for the 72-member truss tower

Node	*x-direction*	*y-direction*	*z-direction*
17	0.0	0.0	-5.0
18	0.0	0.0	-5.0
19	0.0	0.0	-5.0
20	0.0	0.0	-5.0

Two different design cases of the problem are considered. In the first design case, the minimum cross-sectional areas assigned to member groups are limited to 0.1 in^2, whereas in the second design case, the minimum cross-sectional areas assigned to

member groups are limited to 0.01 in^2. The other design requirements are all identical in both design cases.

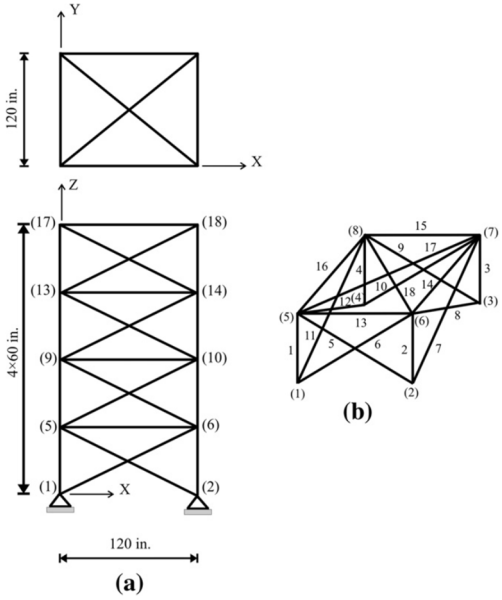

Figure 5.13. 72-member truss tower

Table 5.7 Member groups of the 72-member truss tower

Group Name	*Elements*	Group Name	*Elements*
1	1-4	9	37-40
2	5-12	10	41-48
3	13-16	11	49-52
4	17-18	12	53-54
5	19-22	13	55-58
6	23-30	14	59-66
7	31-34	15	67-70
8	35-36	16	71-72

5.1.3.1 Results of Analyses

5.1.3.1.1 The First Design Case

The 72-member tower truss problem (case-1) was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms. The independent runs performed with the Atomic Orbital Search algorithm have led to an optimum design weight of the structure in the range between 384,87 lb and 424.08 lb, and the algorithm has terminated at iteration numbers between 300 and 650. For all the independent runs performed with the Atomic Orbital Search algorithm, the resulting convergence curves are plotted in Figure 5.14.

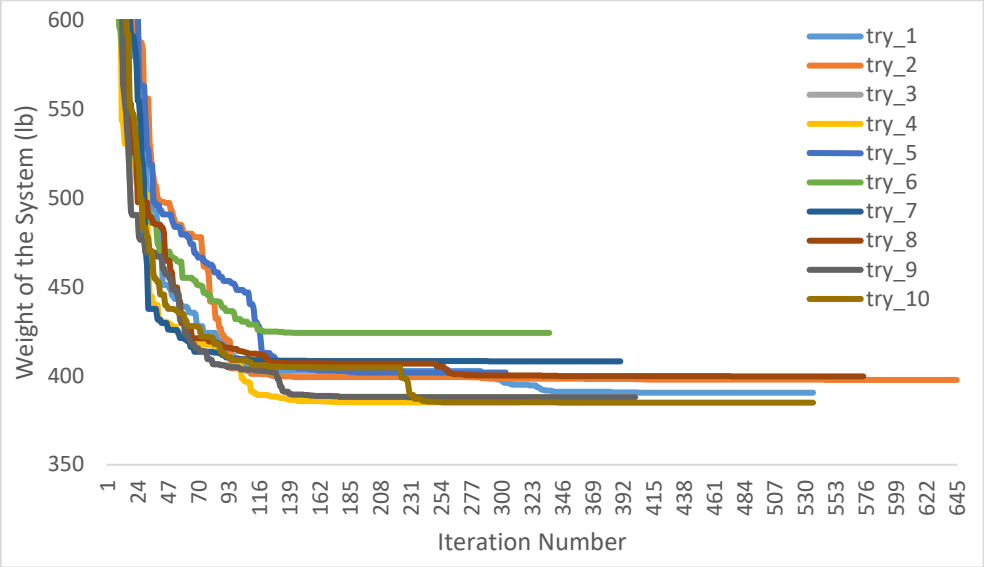

Figure 5.14. Convergence curves for the 72-bar tower truss (case-1) with AOS

While solving the 72-member tower truss problem with the Honey Badger Algorithm, after some trials, the Beta and C parameters are chosen as 1.65 and 1.15, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 380,75 lb for the truss, while the worst run leads to a design weight of 394,87 lb, and the algorithm has terminated at iteration numbers between 200 and

400. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.15.

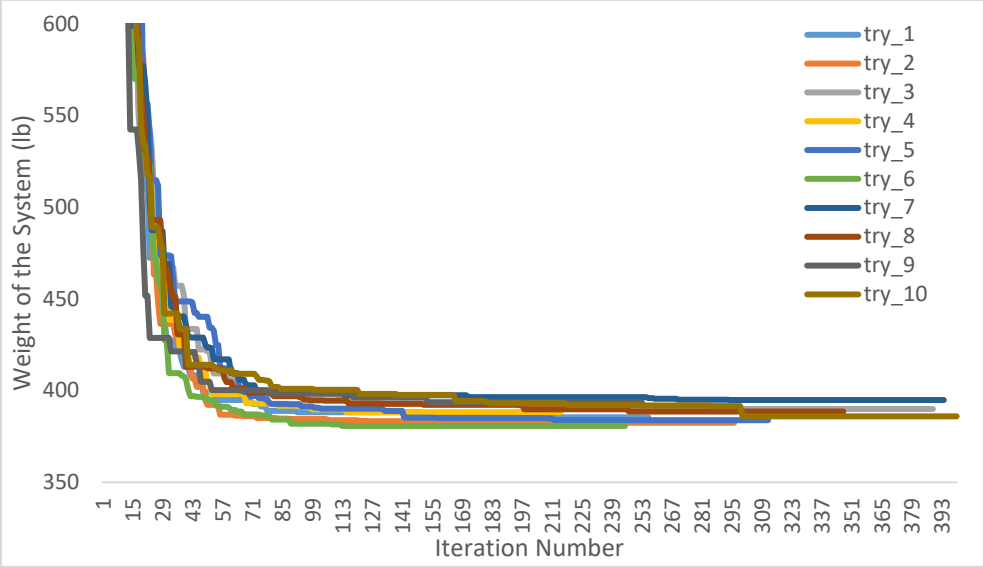

Figure 5.15. Convergence curves for the 72-bar tower truss (case-1) with HBA

Nuclear Fission-Nuclear Fusion Algorithm has been implemented such that the μ and ρ parameters are set to 10^{14} and 1.10, respectively. In this example, a modification of the standard algorithm is performed through equation 5.2 to remedy its search performance.

$$p_{new} = CM + \begin{cases} \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right), & original \\ \left(0.5 * \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right)\right), & modified \end{cases} \quad (5.2)$$

The independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm have led to an optimum design weight of the structure in the range between 388,81 lb and 414,48 lb, and the algorithm has terminated at iteration numbers between 300 and 400. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm, the resulting convergence curves are plotted in Figure 5.16.

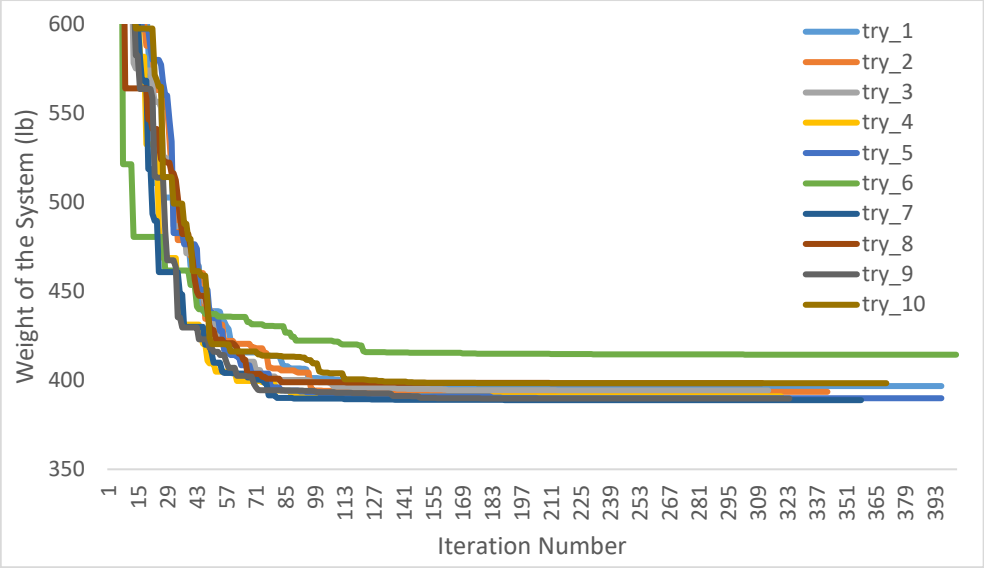

Figure 5.16. Convergence curves for the 72-bar tower truss (case-1) with N2F

While solving the 72-bar tower truss, a standard formulation of the Pathfinder Algorithm is not followed again; rather a modification of the algorithm is accomplished as implemented in equation 5.3.

$$p_{new} = p_{old} + \left((alpha * r1) * \left(p_{global} - p_{old}\right)\right) + \left((beta * r2) * \left(p_{old}^{i} - p_{old}^{i-1}\right)\right) + \begin{cases} (eps * d), & original \\ (0.1 * eps * d), & modified \end{cases} \quad (5.3)$$

The independent runs performed with the modified Pathfinder Algorithm have led to an optimum design weight of the structure in the range between 386,06 lb and 407,87 lb, and the algorithm has terminated at iteration numbers between 350 and 420. For all the independent runs performed with the modified Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.17.

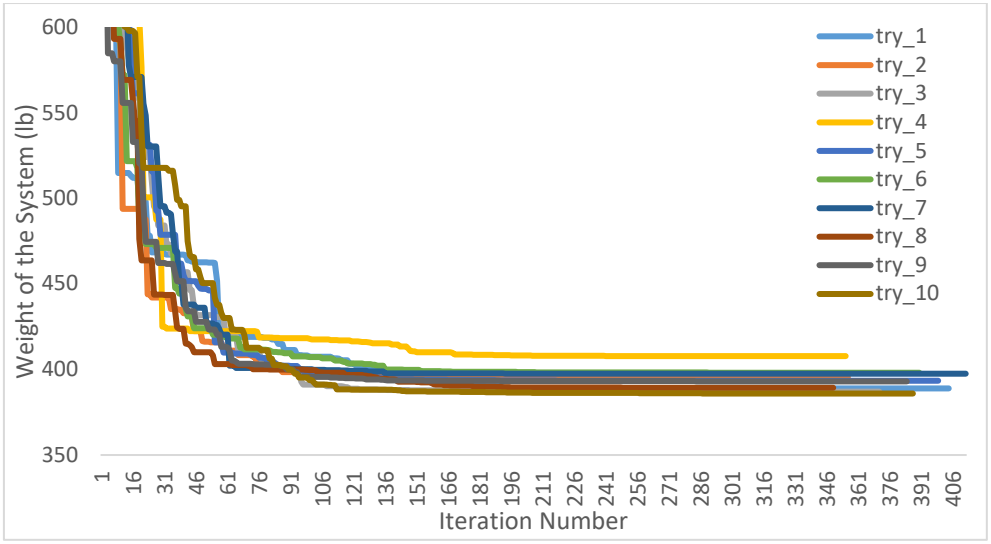

Figure 5.17. Convergence curves for the 72-bar tower truss (case-1) with PFA

In the runs performed with the Salp Swarm Algorithm, the c1 parameter is multiplied by 0.05 here. The best run of the Salp Swarm Algorithm has resulted in a design weight of 380,29 lb for the truss, while the worst run leads to a design weight of 385,84 lb, and the algorithm has terminated at iteration numbers between 280 and 560. For all the independent runs performed with the Salp Swarm Algorithm, the resulting convergence curves are plotted in Figure 5.18.

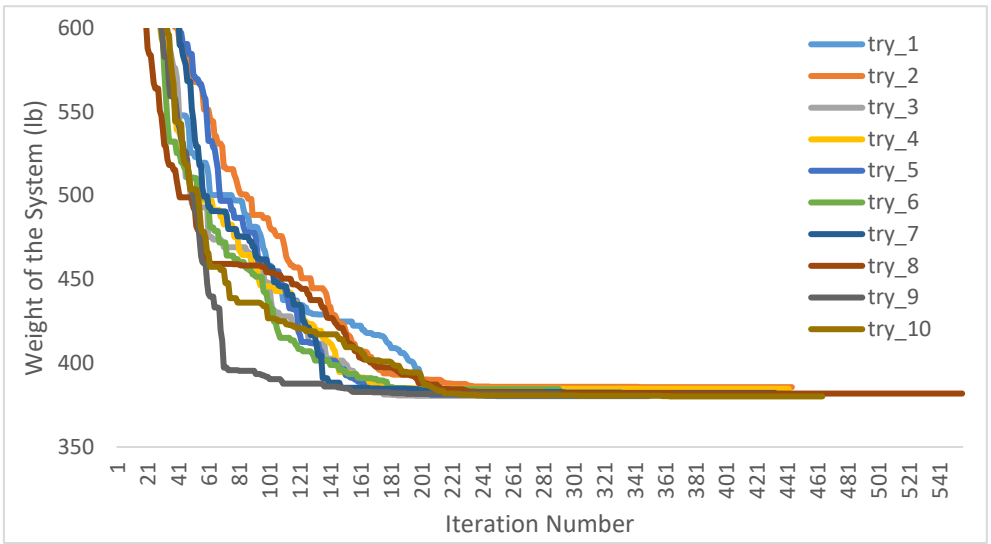

Figure 5.18. Convergence curves for the 72-bar tower truss (case-1) with SSA

The 72-bar tower truss problem (case-1) has formerly been addressed by various papers in the literature such as Lee and Geem (2004), Perez and Behdinan (2007), and Kaveh and Talatahari (2009). In these papers, the optimum designs of the problem were 379,22 lb by the HA technique obtained by performing 20,000 analyses, 381,91 lb by the PSO technique, and 379,66 lb by the HBB-BC technique obtained by performing 13.200 analyses. These optimum designs are reproduced in Table 5.8 in terms of cross-sectional areas and the truss weight. The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.8. Accordingly, the optimum designs of the truss reached in this study are 384,87 lb by the AOS technique obtained by performing 26.800 analyses; 380,75 lb by the HBA technique obtained by performing 12.250 analyses; 388,81 lb by the N2F technique obtained by performing 17.850 analyses; 386,06 lb by the PFA technique obtained by performing 19.300 analyses; and 380,29 lb by the SSA technique obtained by 23.150 analyses.

Table 5.8 Summary of the optimum results for the 72-bar cantilever truss problem (case 1)

Design Variables		Best cross-section areas of each algorithm (in^2)							
		Lee et al. (2004)	Perez et al. (2007)	Kaveh et al. (2009)	This Study				
		HS	**PSO**	**HBB-BC**	**AOS**	**HBA**	**N2F**	**PFA**	**SSA**
1	A_1~A_4	1.790	1.7427	1.9042	1.6894	1.8303	1.8304	1.9801	1.8063
2	A_5~A_{12}	0.521	0.5185	0.5162	0.5885	0.5428	0.5717	0.4487	0.4927
3	A_{13}~A_{16}	0.100	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
4	A_{17}~A_{18}	0.100	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
5	A_{19}~A_{22}	1.229	1.3079	1.2582	1.0908	1.2116	1.1025	1.2337	1.2296
6	A_{23}~A_{30}	0.522	0.5193	0.5035	0.4640	0.5156	0.5917	0.4571	0.5012
7	A_{31}~A_{34}	0.100	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
8	A_{35}~A_{36}	0.100	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
9	A_{37}~A_{40}	0.517	0.5142	0.5178	0.7363	0.5463	0.6652	0.6818	0.5636
10	A_{41}~A_{48}	0.504	0.5464	0.5214	0.5235	0.5067	0.4775	0.4870	0.5315
11	A_{49}~A_{52}	0.100	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
12	A_{53}~A_{54}	0.101	0.1095	0.1007	0.1000	0.1003	0.1000	0.1000	0.1000
13	A_{55}~A_{58}	0.156	0.1615	0.1566	0.1526	0.1566	0.1983	0.1508	0.1545
14	A_{59}~A_{66}	0.547	0.5092	0.5421	0.5774	0.5384	0.5690	0.6191	0.5787
15	A_{67}~A_{70}	0.442	0.4967	0.4132	0.5058	0.4846	0.3249	0.6199	0.3891
16	A_{71}~A_{72}	0.590	0.5619	0.5756	0.4957	0.5084	0.5990	0.5501	0.6217
Weight (lb)		379.27	381.91	379.66	384.87	380.75	388.81	386.06	380.29
Average weight (lb)		N/A	N/A	381.85	396.746	387.448	395.568	393.708	382.342
Analyses number		20,000	N/A	13,200	26,800	12,250	17,850	19,300	23,150

5.1.3.1.2 The Second Design Case

The 72-member tower truss problem (case-2) was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms. The independent runs performed with the Atomic Orbital Search algorithm have led to an optimum design weight of the structure in the range between 370,65 lb and 387,39 lb, and the algorithm has terminated at iteration numbers between 260 and 560. For all the independent runs performed with the Atomic Orbital Search algorithm, the resulting convergence curves are plotted in Figure 5.19.

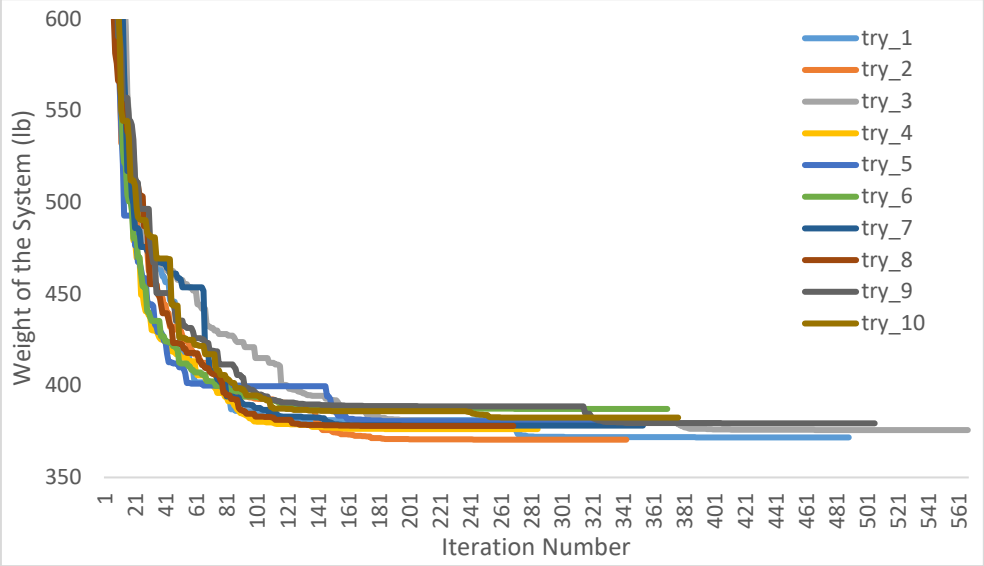

Figure 5.19. Convergence curves for the 72-bar tower truss (case-2) with AOS

While solving the 72-member tower truss problem (case-2) with the Honey Badger Algorithm, the Beta and C parameters are chosen as 1.65 and 1.15, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 367,54 lb for the truss, while the worst run leads to a design weight of 378,27 lb, and the algorithm has terminated at iteration numbers between 220 and 400. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.20.

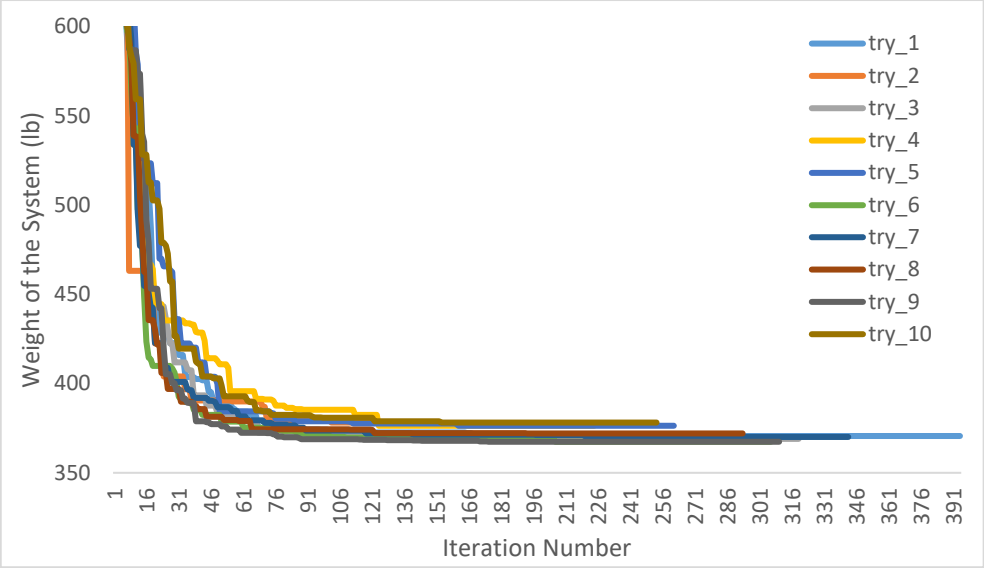

Figure 5.20. Convergence curves for the 72-bar tower truss (case-2) with HBA

Nuclear Fission-Nuclear Fusion Algorithm has been implemented such that the μ and ρ parameters are set to 10^{14} and 1.10, respectively. In this example, a modification of the standard algorithm is performed through equation 5.4 to improve its search performance.

$$p_{new} = CM + \begin{cases} \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right), & original \\ \left(0.5 * \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right)\right), & modified \end{cases} \quad (5.4)$$

The independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm have led to an optimum design weight of the structure in the range between 368,63 lb and 389,28 lb, and the algorithm has terminated at iteration numbers between 340 and 420. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm, the resulting convergence curves are plotted in Figure 5.21.

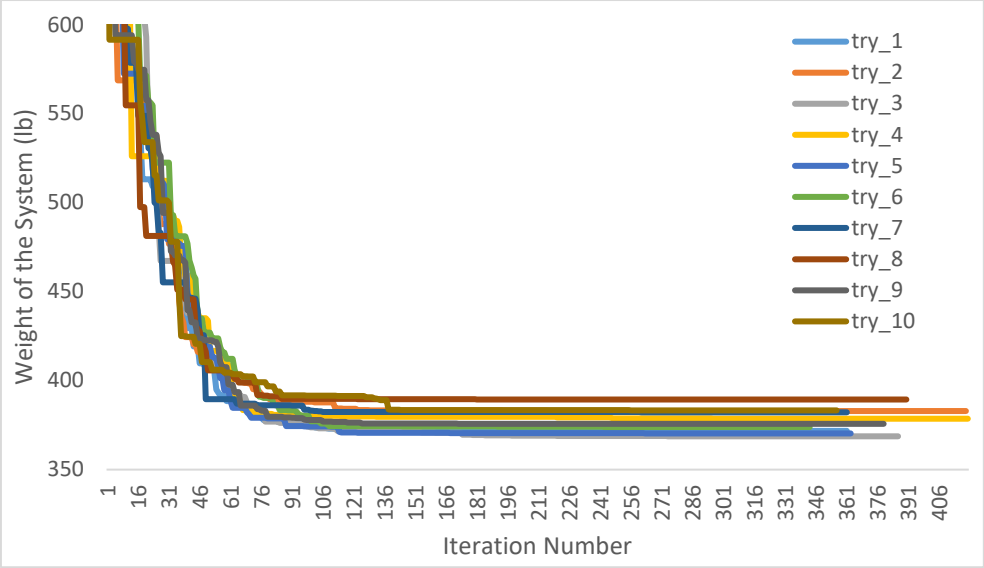

Figure 5.21. Convergence curves for the 72-bar tower truss (case-2) with N2F

While solving the 72-bar tower truss (case-2), a standard formulation of the Pathfinder Algorithm is not followed again; rather a modification of the algorithm is accomplished as implemented in equation 5.5.

$$p_{new} = p_{old} + \left((alpha * r1) * \left(p_{global} - p_{old}\right)\right) + \left((beta * r2) * \left(p_{old}^{i} - p_{old}^{i-1}\right)\right) + \begin{cases} (eps * d), & original \\ (0.1 * eps * d), & modified \end{cases} \quad (5.5)$$

The independent runs performed with the modified Pathfinder Algorithm have led to an optimum design weight of the structure in the range between 372,87 lb and 389,50 lb, and the algorithm has terminated at iteration numbers between 230 and 630. For all the independent runs performed with the modified Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.22.

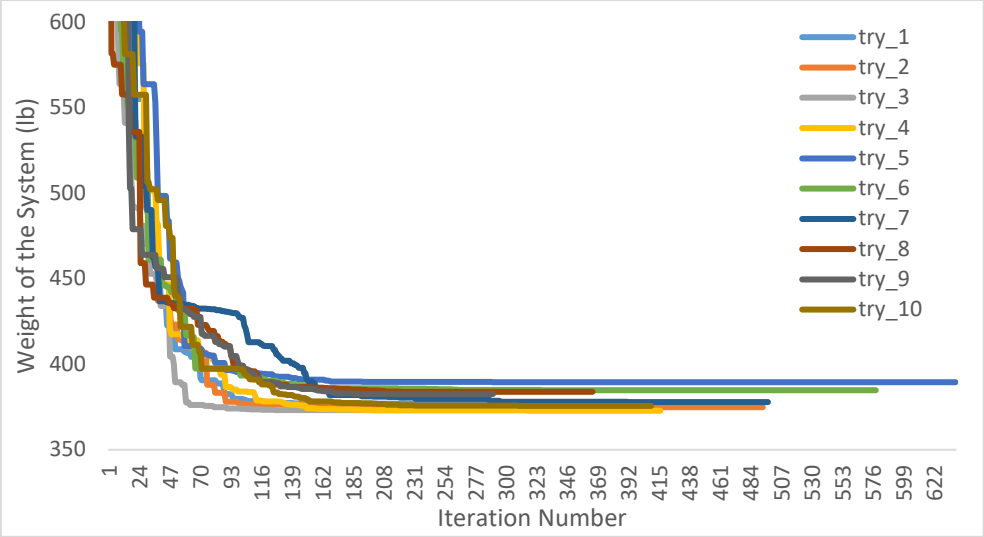

Figure 5.22. Convergence curves for the 72-bar tower truss (case-2) with PFA

In the runs performed with the Salp Swarm Algorithm, the c1 parameter is multiplied by 0.05 here. The best run of the Salp Swarm Algorithm has resulted in a design weight of 364,69 lb for the truss, while the worst run leads to a design weight of 367,99 lb, and the algorithm has terminated at iteration numbers between 240 and 470. For all the independent runs performed with the Salp Swarm Algorithm, the resulting convergence curves are plotted in Figure 5.23.

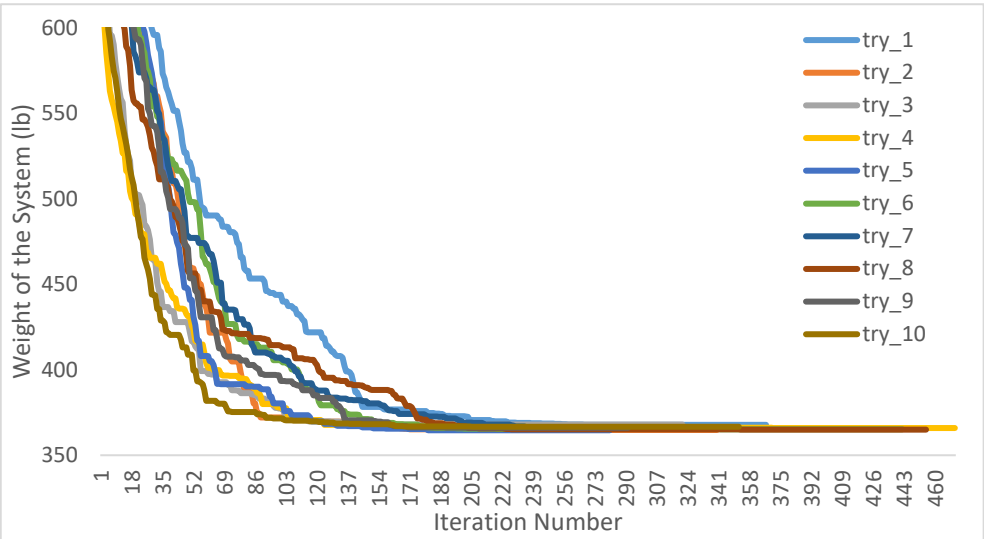

Figure 5.23. Convergence curves for the 72-bar tower truss (case-2) with SSA

Table 5.9 Summary of 72-bar benchmark results (case 2)

Design Variables		Best cross-section areas of each algorithm (in^2)							
		Lee and Geem (2004)	Lamberti L. (2008)	S.O. Degetekin (2012)	This Study				
		HS	**CMLPSA**	**EHS**	**AOS**	**HBA**	**N2F**	**PFA**	**SSA**
1	A_1~A_4	1,963	1,8866	1,8890	1,5044	1,7887	1,8648	2,0000	1,7311
2	A_5~A_{12}	0,481	0,5169	0,5020	0,5357	0,5706	0,5542	0,6181	0,5412
3	A_{13}~A_{16}	0,010	0,0100	0,0100	0,0100	0,0100	0,0100	0,0100	0,0100
4	A_{17}~A_{18}	0,011	0,0100	0,0100	0,0101	0,0100	0,0106	0,0100	0,0100
5	A_{19}~A_{22}	1,233	1,2903	1,2840	1,4614	1,2875	1,2656	1,4751	1,3584
6	A_{23}~A_{30}	0,506	0,5170	0,5260	0,4940	0,5797	0,5094	0,4691	0,5059
7	A_{31}~A_{34}	0,011	0,0100	0,0100	0,0100	0,0100	0,0100	0,0101	0,0100
8	A_{35}~A_{36}	0,012	0,0100	0,0100	0,0100	0,0100	0,0170	0,0100	0,0100
9	A_{37}~A_{40}	0,538	0,5207	0,5280	0,7234	0,5358	0,5733	0,5803	0,5697
10	A_{41}~A_{48}	0,533	0,5180	0,5250	0,5882	0,4705	0,4692	0,4568	0,5202
11	A_{49}~A_{52}	0,010	0,0100	0,0100	0,0152	0,0100	0,0688	0,0100	0,0100
12	A_{53}~A_{54}	0,167	0,1141	0,0630	0,0435	0,2230	0,0776	0,1430	0,1202
13	A_{55}~A_{58}	0,161	0,1665	0,1730	0,1718	0,1600	0,1641	0,1621	0,1665
14	A_{59}~A_{66}	0,542	0,5363	0,5500	0,5658	0,5204	0,5026	0,6418	0,5212
15	A_{67}~A_{70}	0,478	0,4460	0,4440	0,3540	0,3842	0,5839	0,3076	0,4615
16	A_{71}~A_{72}	0,551	0,5761	0,5920	0,6708	0,5588	0,6330	0,4508	0,5998
Weight (lb)		364,33	363,82	364,36	370,65	367,45	368,64	372,87	364,69
Average weight (lb)		N/A	N/A	366,79	378,152	372,604	377,617	379,18	366,077
Analyses number		20.000	900	13.755	17.050	15.450	19.250	20.700	14.000

The 72-bar tower truss problem (case-2) has formerly been addressed by various methods in the literature such as Harmony Search Algorithm in Lee and Geem (2004), Corrected Multi-Level & Multi-Point Simulated Annealing (CMLSA) in Lamberti (2008), Efficient Harmony Search (EHS) in Lamberti and Degertekin (2012). In these studies, the optimum designs of the problem were 364,33 lb by the HS technique obtained by performing 20,000 analyses, 363,82 lb by the CMLSA technique obtained by 900 analyses, and 364,36 lb by the EHS technique obtained by performing 13.755 analyses. These optimum designs are reproduced in Table 5.9 in terms of cross-sectional areas and the truss weight. The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.9. Accordingly, the optimum designs of the truss reached in this study are 370,65 lb by the AOS technique obtained by performing 17.0500 analyses; 367,45 lb by the HBA technique obtained by performing 15.450

analyses; 368,64 lb by the N2F technique obtained by performing 19.250 analyses; 372,87 lb by the PFA technique obtained by performing 20.700 analyses; and 364,69 lb by the SSA technique obtained by 14.000 analyses.

5.2 Structural Problems

Two structural truss problems taken from the literature are studied here. These problems are the 130-bar tower and 117- bar cantilever truss. While solving a structural design problem, each optimization algorithm is run ten times independently to achieve the size optimum design of the structure, and the best feasible weight attained is reported to be the minimum weight design of the structure achieved with each optimization algorithm. All the algorithms are initiated with randomly generated solutions at the beginning of the optimization process in each run. The maximum number of iterations in a single run is limited to 1000, and if the best feasible is not improved over 100 iterations, the algorithm is terminated prematurely.

5.2.1 130- Member Tower

The transmission tower truss shown in Figure 5.24 is an electric transmission tower, which consists of 33 nodes and 130 members altogether. This structure will be designed for the minimum weight by selecting the truss members from a database of pipe sections with the following discrete cross-sectional areas: (1.6129, 2.0645, 2.1484, 2.7935, 3.1871, 4.1226, 4.3161, 5.1548, 5.6839, 6.9032, 6.9032, 9.5484, 10.9677, 14.3871, 14.5161, 17.1613, 17.2903, 19.4838, 20.4516, 23.7419, 26, 27.7419, 28.4516, 35.2903, 36, 39.4193, 52.258, 54.1934, 54.1934, 72.9031, 76.774, 82.5805, 94.1934, 100.645, 103.8708, 123.8707, 137.4191 cm^2). No member grouping is carried out for the truss elements. The material density is 7850 kg/m^3 and the Young's Modulus is 200,000 MPa.

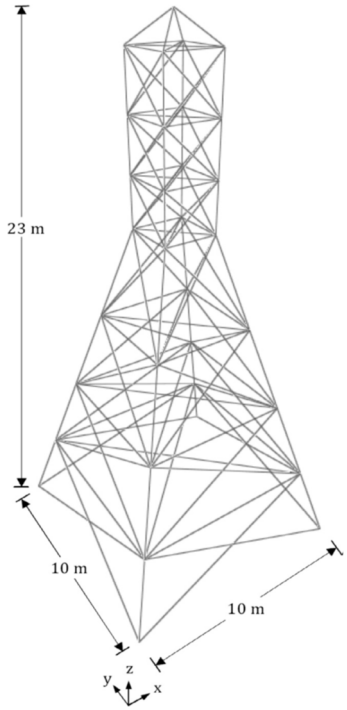

Figure 5.24. 130-member tower

The loads applied on the nodes of the truss are summarized in Table 5.10. Both the strength and displacement constraints are imposed on the problem. The strength constraints are imposed according to the provisions of LRFD-AISC (2016) specifications. On the other hand, the displacements of nodes 29, 30, 31, 32, and 33 are limited to a maximum value of ±3 cm in the x-direction.

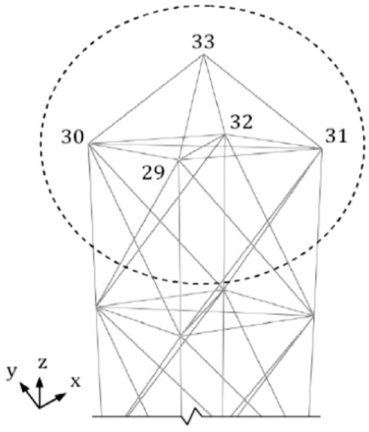

Figure 5.25. Loadings points of 130-member tower

Table 5.10 Loadings of 130-member tower

Node	*x-direction (kN)*	*y-direction (kN)*	*z-direction (kN)*
29	100	0	0
30	100	0	0
31	0	25	0
32	0	25	0
33	0	0	-50

5.2.1.1 Results of Analyses

The 130-member structural truss problem was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms.

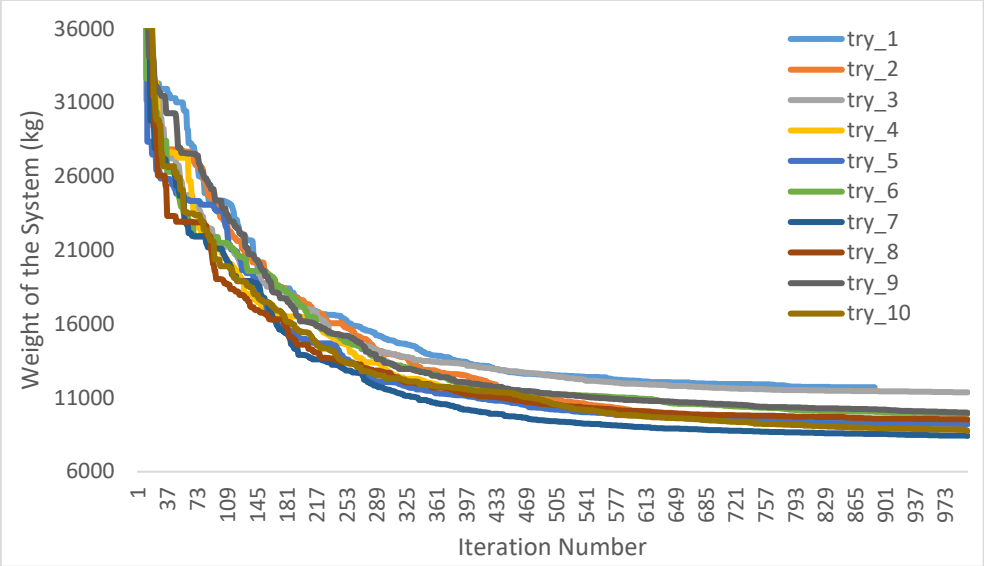

Figure 5.26. Convergence curves for the 130-bar structural truss problem with AOS

The best run of the Atomic Orbital Search algorithm has resulted in a design weight of 8427,53 kg for the truss, while the optimum design weight in the worst run is 11710,48 kg. For all the independent runs performed with the Atomic Orbital Search

algorithm, the resulting convergence curves which show the variation of the best feasible design versus the iteration number performed are plotted in Figure 5.26.

While solving the 130-member structural truss problem with the Honey Badger Algorithm, after some trials, the Beta and C parameters are chosen as 1.25 and 0.50, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 12206,85 kg for the truss, while the worst run leads to a design weight of 16313,69 kg. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.27.

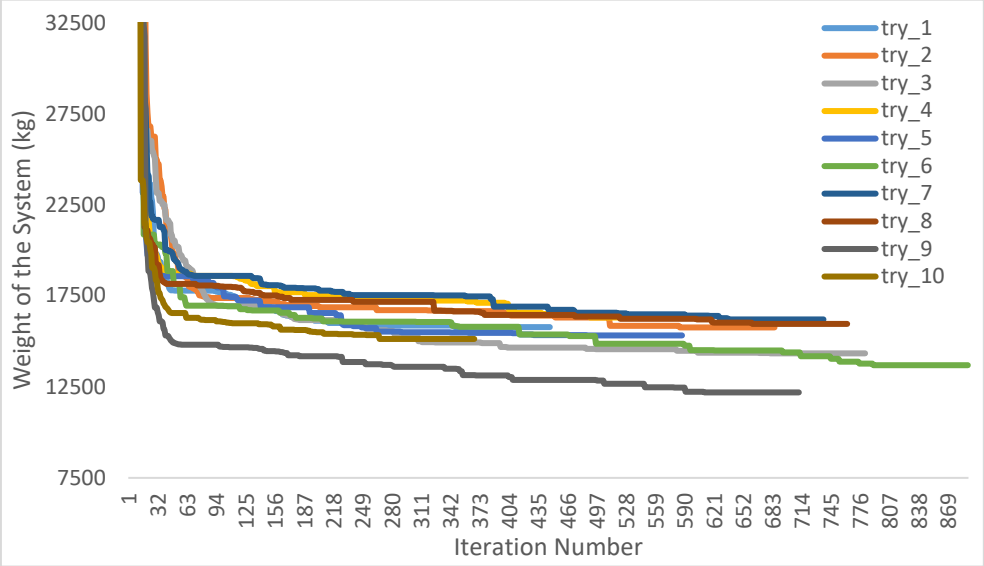

Figure 5.27. Convergence curves for the 130-bar structural truss problem with HBA

Nuclear Fission-Nuclear Fusion Algorithm has been implemented such that the μ and ρ parameters are set to 10^7 and 1.01, respectively. In this example, a modification of the standard algorithm is performed through equation 5.6 to remedy its search performance.

$$p_{new} = CM + \begin{cases} \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right), & original \\ \left(0.5 * \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right)\right), & modified \end{cases} \quad (5.6)$$

The independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm have led to an optimum design weight of the structure in the range between 11841,01 kg and 13731,87 kg, and the algorithm has terminated at iteration numbers between 550 and 1000. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm, the resulting convergence curves are plotted in Figure 5.28.

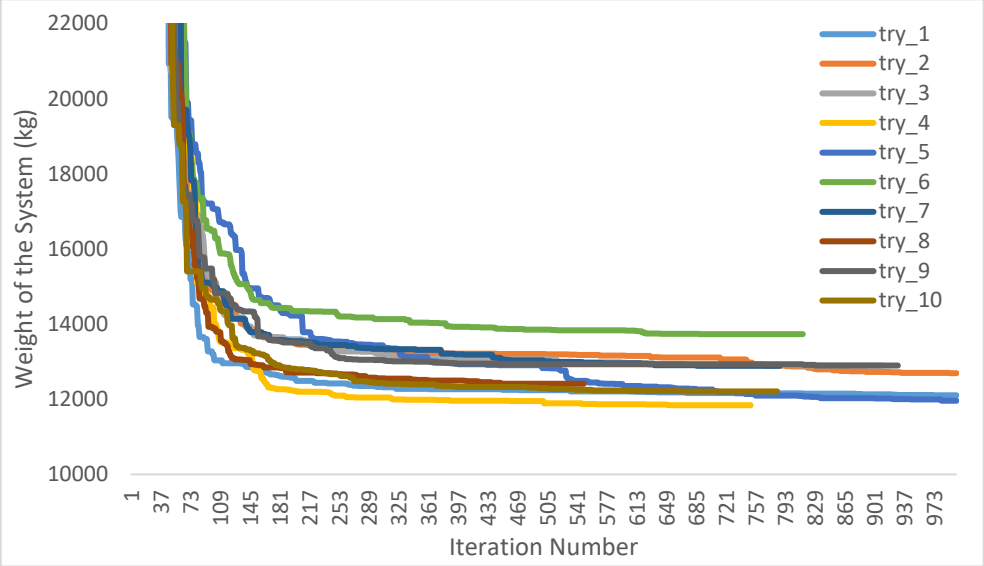

Figure 5.28. Convergence curves for the 130-bar structural truss problem with N2F

While solving the 130-bar tower truss, a standard formulation of the Pathfinder Algorithm is not followed again; rather a modification of the algorithm is accomplished as implemented in equations 5.7 and 5.8.

$$p_{new} = p_{old} + \Big((alpha * r1) * \big(p_{global} - p_{old}\big)\Big) + \Big((beta * r2) * \big(p_{old}^{i} - p_{old}^{i-1}\big)\Big) + \begin{cases} (eps * d), & original \\ (0.1 * eps * d), & modified \end{cases} \quad (5.7)$$

$$A = 5 * u1 * e^{\left(-2*\left(\frac{ti}{tmax}\right)\right)} \quad (5.8)$$

The independent runs performed with the modified Pathfinder Algorithm have led to an optimum design weight of the structure in the range between 13167,45 kg and 16743,27 kg, and the algorithm has terminated at iteration number 1000 in all the runs. For all the independent runs performed with the modified Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.29.

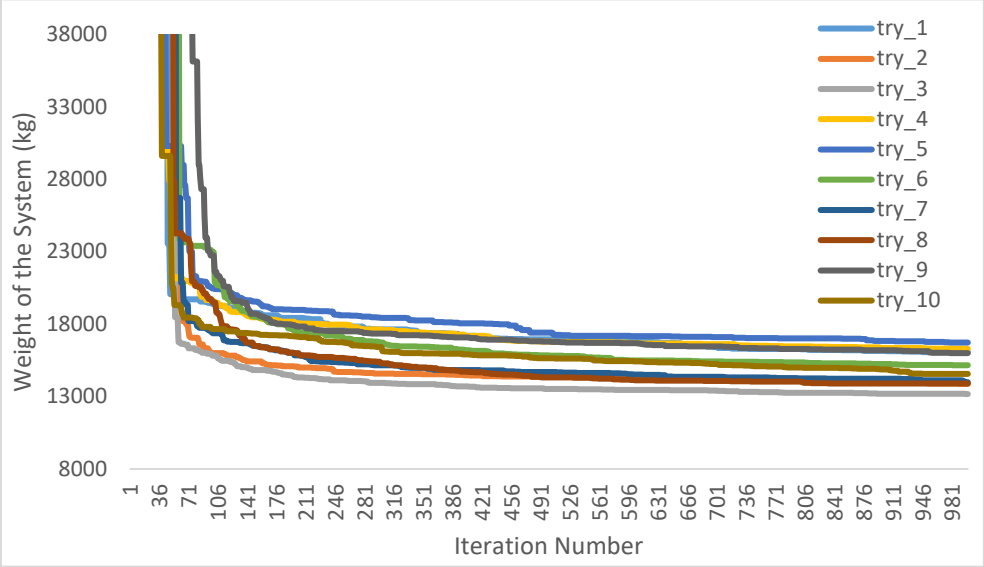

Figure 5.29. Convergence curves for the 130-bar structural truss problem with PFA

The best run of the Salp Swarm Algorithm has resulted in a design weight of 7876,37 kg for the truss, while the worst run leads to a design weight of 10504,57 kg. For all the independent runs performed with the Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.30.

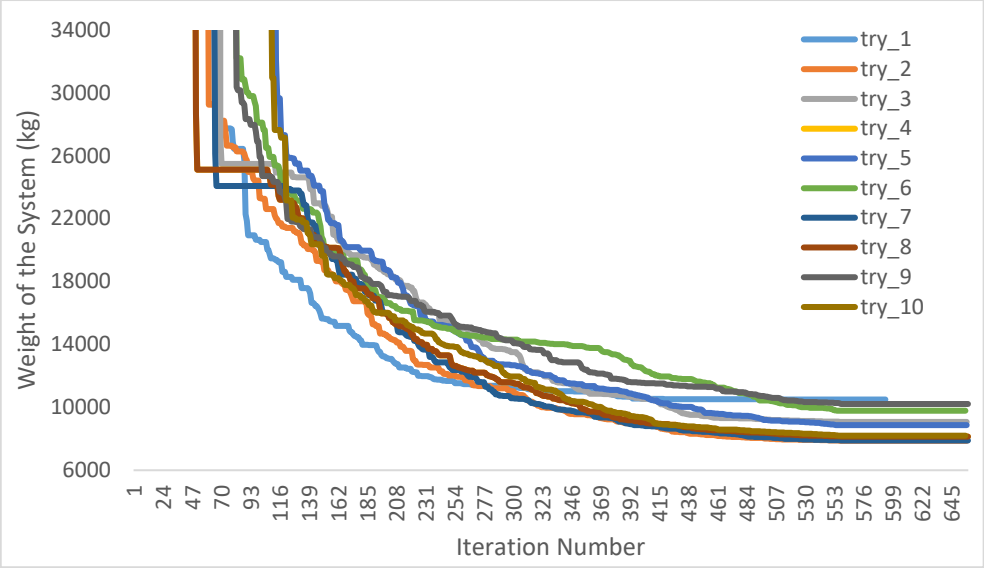

Figure 5.30. Convergence curves for the 130-bar structural truss problem with SSA

The 130-member transmission tower truss problem has formerly been studied by Kazemzadeh Azad et al. (2015) using the Particle Swarm Optimization (PSO), standard BB-BC, modified BB-BC, exponential BB-BC methods as well as Guided Stochastic Search in two different formulations referred to as (GSS_A) and (GSS_B). The optimum designs of the problem attained with these algorithms are 7344,9 kg by the PSO; 8686,1 kg by standard BB-BC; 7011,1 kg by modified BB-BC; 7060,7 kg by exponential BB-BC; 6485,6 kg by GSS_A; and finally 6448,1 kg by GSS_B. These optimum designs are reproduced in Table 5.8 in terms of the truss weight.

The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.8. As can be seen from this table, the minimum weights of the truss achieved with the Atomic Orbital Search and Salp Swarm Algorithm are comparatively lesser than those of other implemented metaheuristic techniques and the standard BB-BC, yet they are higher than the solutions attained with the PSO, modified BB-BC, exponential BB-BC, GSS_A, and GSS_B algorithms as reported in Kazemzadeh Azad et al. (2015).

Table 5.11 Summary of the optimum results for the 130-bar tower truss problem

Details	Kazemzadeh Azad et al. (2015)						This Study				
	PSO	BB-BC Standard	BB-BC Modified	BB-BC Exponential	GSS_A	GSS_B	AOS	HBA	N2F	PFA	SSA
Population size	50	50	50	50	1	1	50	50	50	50	50
Max. Iteration No.	1000	1000	1000	1000	200	200	1000	1000	1000	1000	1000
Analyzed number	50000	50000	50000	50000	330	330	50000	35500	37500	50000	32900
Best Results (kg)	7344.9	8686.1	7011.1	7060.7	6485.6	6448.1	8427.5	12206.85	11841.01	13167.45	7876.4
Worst Results (kg)	8869	9492.4	7808.2	7706.6	6706.2	6557.6	11710.5	16313.7	13731.87	16743.27	10224.7
Mean Results (kg)	7749.4	9127.1	7280.2	7307.2	6573.6	6509.9	9750.242	15074.18	12562.49	14979.98	8869.514

5.2.2 117- Member Cantilever Beam

The structural problem shown in Figure 5.31 is a cantilever truss, which consists of 30 joints and 117 members altogether. This structure will be designed for the minimum weight by selecting the truss members from a database of pipe sections with the following discrete cross-sectional areas: (1.6129, 2.0645, 2.1484, 2.7935, 3.1871, 4.1226, 4.3161, 5.1548, 5.6839, 6.9032, 6.9032, 9.5484, 10.9677, 14.3871, 14.5161, 17.1613, 17.2903, 19.4838, 20.4516, 23.7419, 26, 27.7419, 28.4516, 35.2903, 36, 39.4193, 52.258, 54.1934, 54.1934, 72.9031, 76.774, 82.5805, 94.1934, 100.645, 103.8708, 123.8707, 137.4191 cm^2). No member grouping is carried out for the truss elements. The material density is 7850 kg/m^3 and the Young's Modulus is 200,000 MPa.

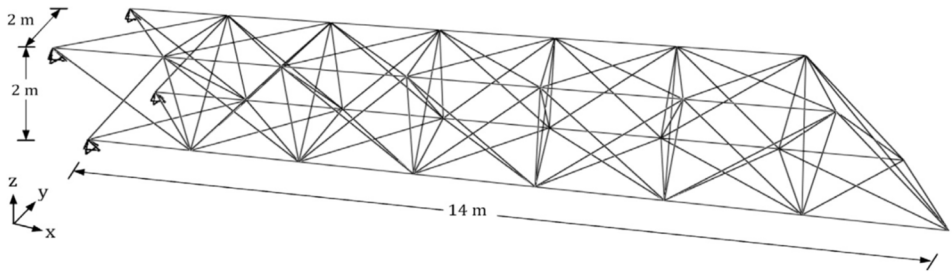

Figure 5.31. 117-bar Cantilever Beam (Kazemzadeh Azad et.al, 2014)

The loads applied on the nodes of the truss are summarized in Table 5.9. The strength constraints are imposed according to the provisions of LRFD-AISC (2016)

specifications. On the other hand, the displacements of all nodes are limited to a maximum value of ±4.0 cm in the x,y, and z-directions.

Table 5.12 Loadings at 117-member truss

Nodes \ Direction	*x (case 1)*	*y (case 2)*	*z (case 3)*
2-8	15	15	-15
10-16	15	15	-15
18-23	15	15	-15
25-30	15	15	-15

5.2.2.1 Results of Analyses

The 117-member cantilever truss problem was designed for the minimum weight by performing ten independent runs with each of the investigated metaheuristic search algorithms. The independent runs performed with the Atomic Orbital Search algorithm have led to an optimum design weight of the structure in the range between 4640,52 kg and 6434,27 kg, and the algorithm has terminated at iteration numbers between 500 and 100. For all the independent runs performed with the Atomic Orbital Search algorithm, the resulting convergence curves are plotted in Figure 5.32.

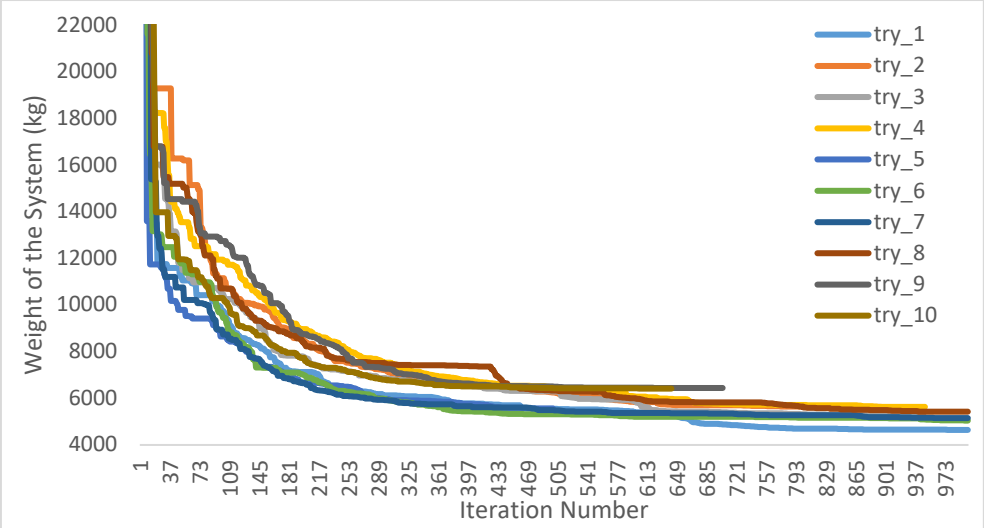

Figure 5.32. Convergence curves for the 117-bar cantilever truss problem with AOS

While solving the 117-member cantilever truss problem with the Honey Badger Algorithm, after some trials, the Beta and C parameters are chosen as 0.85 and 0.15, respectively. The best run of the Honey Badger Algorithm has resulted in a design weight of 6112,82 kg for the truss, while the worst run leads to a design weight of 9813,04 kg. For all the independent runs performed with the Honey Badger Algorithm, the resulting convergence curves are plotted in Figure 5.33.

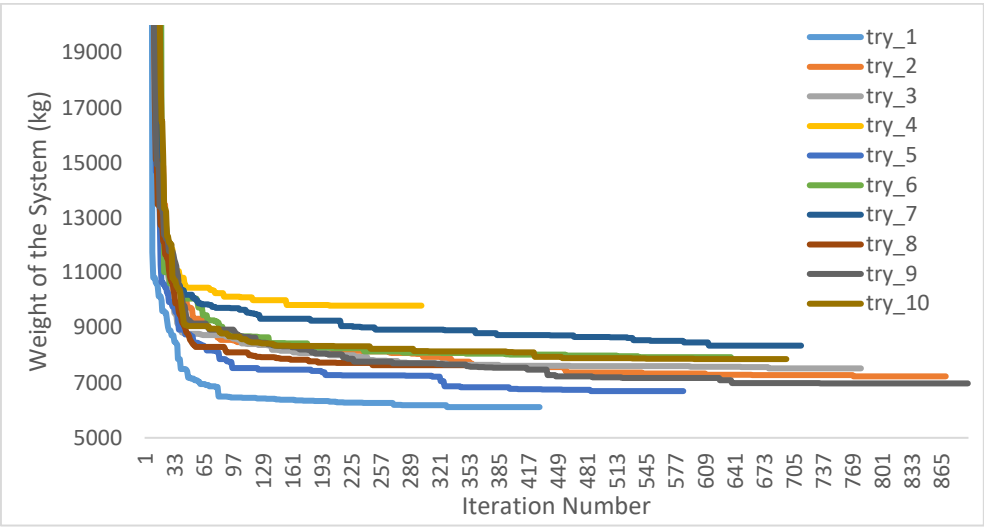

Figure 5.33. Convergence curves for the 117-bar cantilever truss problem with HBA

Nuclear Fission-Nuclear Fusion Algorithm has been implemented such that the μ and ρ parameters are set to 10^7 and 1.01, respectively. In this example, a modification of the standard algorithm is performed through equation 5.9 to remedy its search performance.

$$p_{new} = CM + \begin{cases} \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right), & original \\ \left(0.5 * \left(rand * (x_{max} - x_{min}) * (\mu^{\left(-\frac{t}{t_{max}}\right)})\right)\right), & modified \end{cases} \quad (5.9)$$

The independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm have led to an optimum design weight of the structure in the range between 5713,2 kg and 7611,84 kg, and the algorithm has terminated at iteration numbers between 350 and 1000. For all the independent runs performed with the Nuclear Fission-Nuclear Fusion algorithm, the resulting convergence curves are plotted in Figure 5.34.

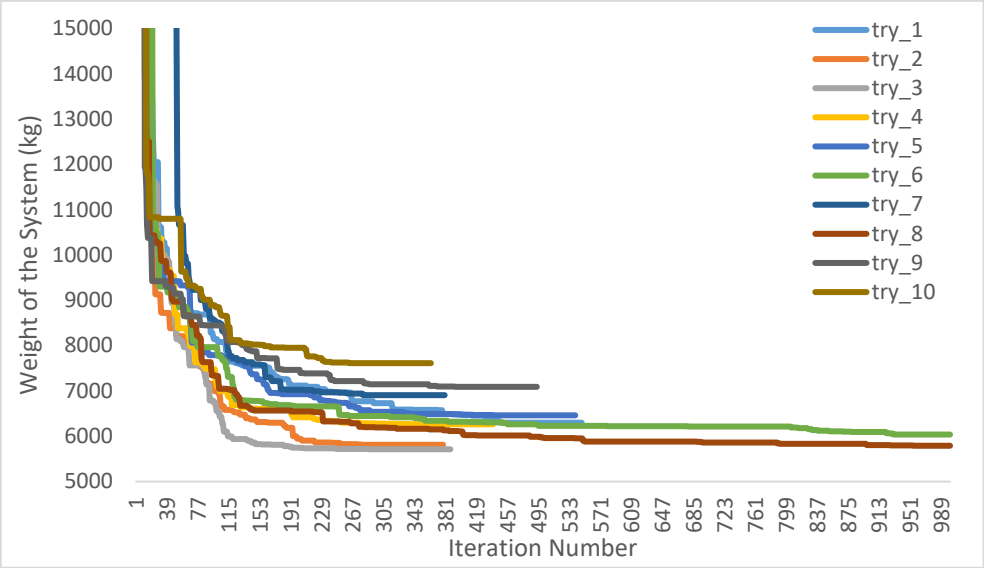

Figure 5.34. Convergence curves for the 117-bar cantilever truss problem with N2F

While solving the 117-bar cantilever truss, a standard formulation of the Pathfinder Algorithm is not followed again; rather a modification of the algorithm is accomplished as implemented in equation 5.10.

$$p_{new} = p_{old} + \left((alpha * r1) * \left(p_{global} - p_{old}\right)\right) + \left((beta * r2) * \left(p_{old}^{i} - p_{old}^{i-1}\right)\right) + \begin{cases} (eps * d), & original \\ (0.1 * eps * d), & modified \end{cases} \quad (5.10)$$

The independent runs performed with the modified Pathfinder Algorithm have led to an optimum design weight of the structure in the range between 6225,00 kg and 9685,83 kg, and the algorithm has terminated at iteration numbers between 750 and 1000. For all the independent runs performed with the modified Pathfinder Algorithm, the resulting convergence curves are plotted in Figure 5.35.

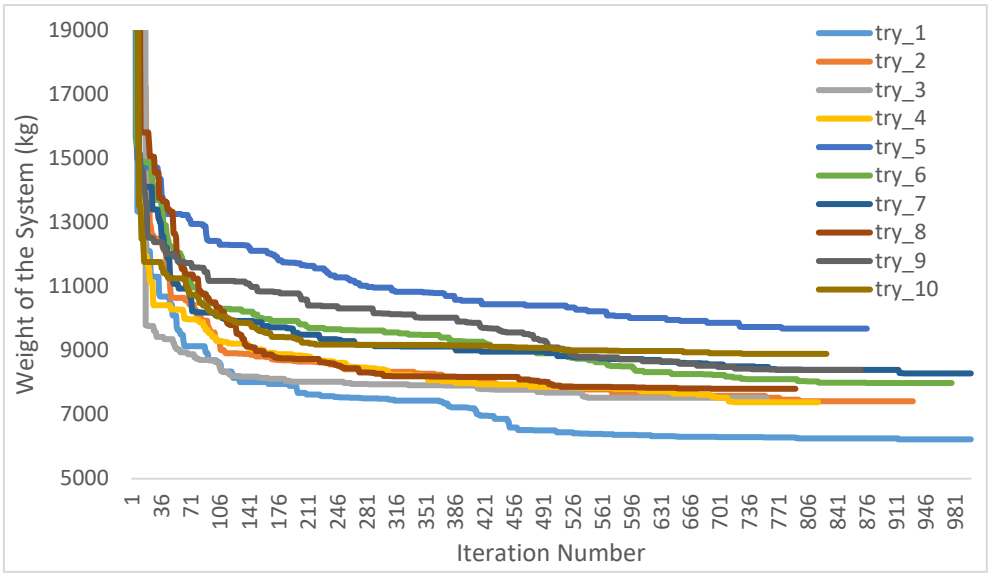

Figure 5.35. Convergence curves for the 117-bar cantilever truss problem with PFA

The best run of the Salp Swarm Algorithm has resulted in a design weight of 5066,63 kg for the truss, while the worst run leads to a design weight of 8645,80 kg, and the

algorithm has terminated at iteration numbers between 450 and 630. For all the independent runs performed with the Salp Swarm Algorithm, the resulting convergence curves are plotted in Figure 5.36.

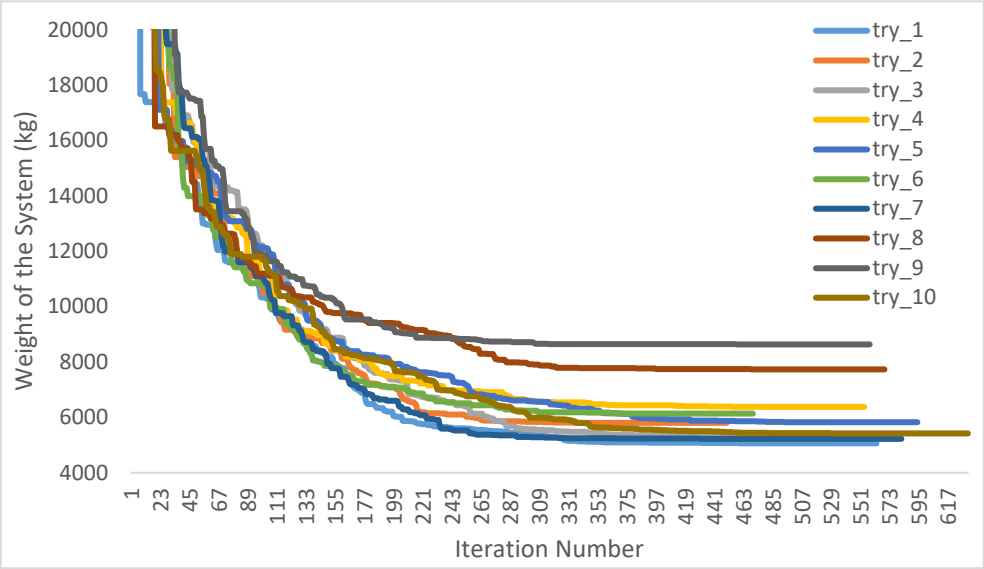

Figure 5.36. Convergence curves for the 117-bar cantilever truss problem with SSA

The 117-member cantilever truss problem has formerly been studied by Kazemzadeh Azad et al. (2019) using the standard Adaptive Dimensional Search (ADS), exponential BB-BC (EBB-BC), modified BB-BC (MBB-BC) methods as well as the modified Monitored Convergence Curve framework (MCC) versions of all these algorithms referred to as MCC-ADS, MCC-EB, and MCC-MB. The optimum designs of the problem attained with these algorithms are 3078,32 kg by the standard ADS; 3094,15 kg by MCC-ADS (SMP_{100}); 3077,79 kg by MCC-ADS (SMP_{150}); 3041,17 kg by exponential BB-BC; 3041,29 kg by MCC-EB (SMP_{100}); 3042,09 kg by MCC-EB (SMP_{150}); 3125,25 kg by modified BB-BC; 3052,88 kg by MCC-MB (SMP_{100}); and 3058,80 kg by MCC-MB (SMP_{150}). These optimum designs are reproduced in Table 5.10 in terms of the truss weight.

The optimum designs of the problem attained with the five implemented metaheuristic algorithms in this study are also reproduced in Table 5.10. As can be

seen from this table, the minimum weight of the truss achieved with the Atomic Orbital Search is comparatively lesser than those of other implemented metaheuristic techniques, yet they are higher than the solutions attained with the standard ADS, MCC-ADS, exponential BB-BC, MCC-EB, modified BB-BC, and MCC-MB algorithms as reported in Kazemzadeh Azad et al. (2019).

Table 5.13 Summary of the optimum results for the 117-bar cantilever truss problem

Details	Kazemzadeh Azad et al. (2019)									This Study				
	ADS	MCC-ADS SMP(100)	MCC-ADS SMP(150)	EBB-BC	MCC-EB SMP(100)	MCC-EB SMP(100)	MBB-BC	MCC-MB SMP(100)	MCC-MB SMP(100)	AOS	HBA	N2F	PFA	SSA
Best Results (kg)	3078.32	3094.15	3077.79	3041.17	3041.29	3042.09	3123.25	3052.88	3058.8	4640.25	6112.82	5713.2	6225	5066.63
Worst Results (kg)	3299.92	3261.39	3186.25	3692.74	3284.28	3254.03	3417.87	3286.99	3328.39	6434.27	9813.04	7611.8	9685.83	8645.8
Mean Results (kg)	3169.28	3145.34	3127.99	3199.7	3075.87	3095.63	3272.69	3098.57	3133.79	5516.54	7616.391	6399.26	7963.42	6161.046

CHAPTER 6

CONCLUSIONS

The objective of this thesis is to evaluate the performance of the following five recently emerged meta-heuristic search techniques in the sizing optimization of steel trusses: Atomic Orbital Search, Honey Badger Algorithm, Nuclear Fission-Nuclear Fusion Algorithm, PathFinder Algorithm, and Salp Swarm Algorithm. In this chapter, the results of this study are summarized, and future research areas are articulated.

6.1 Summary and Conclusion

A literature survey reveals that in the last three decades, more than two hundred meta-heuristic search techniques have been developed to solve mathematical and engineering optimization problems. These techniques are non-traditional, stochastic, and derivative-free methods that make use of ideas mostly inspired by nature, social or physical systems. This trend of developing new metaheuristic techniques is likely to continue for a while due to the obvious advantages of these techniques with respect to traditional optimization approaches that are based on the use of derivative tools. In this thesis, five of these techniques which are recently proposed; namely Atomic Orbital Search, Honey Badger Algorithm, Nuclear Fission-Nuclear Fusion Algorithm, Pathfinder Algorithm, and Salp Swarm Algorithm have been evaluated in the context of structural optimization and sizing optimization of steel trusses in particular.

The numerical implementations of the optimization algorithms were coded in MATLAB R2021b program using MATLAB programming language. The numerical performances of the methods were quantified using a problem test set consisting of selected benchmark truss problems as well as structural design problems. The

benchmark problems; namely the 25-bar truss tower, 38-bar cantilever truss, and 72-bar truss tower as well as structural design problems; namely the 130-bar truss tower, and 117-bar cantilever truss, were all taken from the literature. The structural analyses of these benchmark problems were performed with the aid of the so-called K-files. The K-files consist of the responses of structures; i.e., member forces and displacements are expressed as mathematical functions of cross-sectional areas of truss members. On the other hand, while solving the structural design problems, optimization codes were communicated with the SAP2000 structural analysis software for obtaining response calculations of designs, generated in the course of the optimization process.

While solving all the investigated benchmark problems and structural design problems, each optimization algorithm was run ten times independently to achieve the size optimum design of the structure. All the optimization algorithms were initiated with randomly generated solutions. The algorithms were terminated when a specified maximum number of iterations was reached or when the best design was not improved over 100 successive iterations.

In this study, the following results were obtained:

- The minimum truss weight of the 25-bar truss tower problem, which was also attained in this study, is 485,05 lb. Both the Honey Badger Algorithm and Nuclear Fission-Nuclear Fusion Algorithm attained this solution by performing 9.200 and 10.500 structural analyses, respectively. When the optimum designs of the search techniques are sorted in the order of the lightest design to the heaviest one, this order appears as HBA, NFA, SSA, PFA, and AOS.
- The optimum design results of the 38-bar cantilever truss problem attained with the studied techniques were 5959,89 lb by SSA; 5975,09 lb by HBA; 6005,09 lb by AOS; 6038,52 lb by N2F; and 7094,79 lb by PFA. The minimum truss weight was obtained by the Salp Swarm Algorithm, yet this

solution is not better than the optimum design of the problem achieved in the literature.

- The minimum design weight of the 72-bar truss tower problem (case-1) attained in this study was 384,87 lb by AOS; 380,75 lb by HBA; 388,81 lb by N2F; 386,06 lb by PFA; and 380,29 lb by SSA. The minimum design weights of the truss obtained by the Salp Swarm Algorithm and the Honey Badger Algorithm were better than the optimum solutions reported by the Particle Swarm Optimization technique in the literature. When the optimum designs of the search techniques are sorted in the order of the lightest design to the heaviest one, this order appears as SSA, HBA, AOS, PFA, and N2F.
- The optimum design results of the 72-bar truss tower problem (case-2) attained with the studied techniques were 370,65 lb by AOS 367,45 lb by HBA; 368,64 lb by N2F; 372,87 lb by PFA; and 364,69 lb by SSA. The minimum design weight of the truss was obtained by the Salp Swarm Algorithm, yet this solution is not better than the optimum design of the problem achieved in the literature. When the optimum designs of the search techniques are sorted in the order of the lightest design to the heaviest one, this order appears as SSA, HBA, N2F, AOS, and PFA.
- The optimum designs of the 130-bar truss tower problem attained with the studied techniques were 8427,5 kg by AOS; 12206,85 kg by HBA; 11841,01 kg by N2F; 13167,45 kg by PFA; and 7876,4 kg by SSA. The minimum design weights of the truss obtained by the Salp Swarm Algorithm and the Atomic Orbital Search technique were better than the optimum solution reported by the standard BB-BC technique in the literature. When the optimum designs of the search techniques are sorted in the order of the lightest design to the heaviest one, this order appears as SSA, AOS, N2F, HBA, and PFA.
- The optimum designs of the 117-bar cantilever truss problem attained with the studied techniques were 4640,25 kg by AOS; 6112,82 kg by HBA; 5713,2 kg by N2F; 6225,00 kg by PFA; and 5066,63 kg by SSA. The minimum

design weight of the truss was obtained by the Atomic Orbital Search technique, yet this solution is not better than the optimum design of the problem achieved in the literature. When the optimum designs of the search techniques are sorted in the order of the lightest design to the heaviest one, this order appears as AOS, SSA, N2F, HBA, and PFA.

- When the analysis results and parameters used in these analyses are evaluated, it is seen that the ability of some of the techniques used in this study, i.e. PFA, to reach the optimum design values in the literature decreases with the increase in design variables.
- For benchmark problems in which the number of design variables is not high, it is observed that meta-heuristic search techniques implemented in this study produced comparable solutions to the formerly reported optimum designs of these trusses by other techniques in the literature.
- In benchmark problems, the best overall performance (out of the five techniques tested) was exhibited by SSA since it produced the best solutions in the 38-bar truss problem and both design cases of the 72-bar truss problem. Another promising performance was exhibited by HBA such that it produced the best solution in the 25-bar truss problem and the second best solutions in the 38-bar truss problem and both design cases of the 72-bar truss problem. On the other hand, the least promising performance was exhibited by PFA such that it produced the highest design weights for the 38-bar truss and the second design case of the 72-bar truss, and the second highest weights for the 25-bar truss, and the first design case of the 72-bar truss.
- For structural design problems in which the number of design variables is rather high, it is observed that meta-heuristic search techniques implemented in this study produced considerably heavier solutions than the formerly reported optimum designs of these trusses by other techniques in the literature. It follows that all these techniques require a thorough reformulation for a successful application to large-scale structural optimization problems with numerous design variables.

- In structural design problems, the best overall performance (out of the five techniques tested) was exhibited by SSA and AOS. The SSA produced the least design weight for the 130-bar truss tower, and the second least design weight for the 117-bar cantilever truss. The AOS produced the second least design weight for the 130-bar truss tower, and the least design weight for the 117-bar cantilever truss. Again, the least promising performance was exhibited by PFA such that it produced the highest design weights for both the structural design problems.

6.2 Future Studies

For structural design problems which include a limited number of design variables, the performances of metaheuristic search techniques investigated in this study are found sufficient and comparable to other optimization techniques in the literature. However, for large-scale and challenging structural optimization problems consisting of numerous design variables, the techniques fall short of exhibiting a satisfactory search performance. Therefore, it is recommended that these techniques are reformulated or improved in a way to remedy their search performance in such problems.

REFERENCES

Arora J.S., (2004). Introduction to Optimum Design. Amsterdam; Boston. Elsevier/Academic Press.

Assimi H., Jamali A. & Nariman-zadeh N. (2018). Multi-Objective Sizing and Topology Optimization of Truss Structures using Genetic Programming based on a new Adaptive Mutant Operator, Springer, Neural Computing and Applications, Volume 31, 5729-5749.

Awad R. (2021). Sizing Optimization of Truss Structures using the Political Optimizer (PO) Algorithm, Structures, Volume 33, 4871-4894.

Azizi M. (2021). Atomic Orbital Search: A Novel Metaheuristic Algorithm, Applied Mathematical Modelling, Volume 93, 657-683.

Baghlani A. & Makiabadi M.H. (2014). Weight Optimization of Truss Structures by a New Feasible Boundary Search Technique Hybridized with Firefly Algorithm, KSCE Journal of Civil Engineering, Volume 18(4), 1105-1118

Bhensdadia V.H. (2015). 'Investigation of Size Optimization of Frame Structure using Grey Wolf Optimizer (GWO) And Stochastic Fractal Search (SFS) Algorithms', Msc Thesis, Department of Mechanical Engineering School of Engineering, Rk University, Rajkot, Gujarat

Bekdaş G., Nigdeli S.M. & Yang X. (2015). Sizing Optimization of Truss Structures using Flower Pollination Algorithm, Applied Soft Computing, Volume 37, 322-331.

Cheng M., Prayogo D., Wu Y. & Lukito M.M (2016). A Hybrid Harmony Search Algorithm for Discrete Sizing Optimization of Truss Structure, Automation in Construction, Volume 69, 21-33.

Degertekin S.O. (2012). Improved Harmony Search Algorithms for Sizing Optimization of Truss Structures, Computers and Structures, Volume (92-93), 229-241.

Degertekin S.O. & Hayalioglu M.S. (2013). Sizing Truss Structures using Teaching-Learning-Based Optimization, Computers and Structures, Volume 119, 177-188.

Degertekin S.O., Lamberti L. & Ugur I.B. (2017). Sizing, Layout and Topology Design Optimization of Truss Structures using the Jaya Algorithm, Applied Soft Computing, Volume 70, 903-928

Degertekin S.O., Lamberti L. & Ugur I.B. (2019). Discrete Sizing/Layout/Topology Optimization of Truss Structures with an Advanced Jaya Algorithm, Applied Soft Computing Journal, Volume 79, 363-390.

Dhiman G. & Kumar V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications, Advances in Engineering Software, Volume 114, 47-70.

Do D.T.T. & Lee J. (2017). A Modified Symbiotic Organisms Search (mSOS) Algorithm for Optimization of Pin-Jointed Structures, Applied Soft Computing, Volume 61, 683-699.

Flager F., Soremekun G., Adya A., Shea K., Haymaker J. & Fischer M. (2014). Fully Constrained Design: A General and Scalable Method for Discrete Member Sizing Optimization of Steel Truss Structures, Computers and Structures, Volume 140, 55-65.

Galilei, G. (1991). Dialogues Concerning Two New Sciences, translated by Henry Crew and Alfondo de Salvio, William Andrew Publishing.

Hasançebi O., Çarbaş S., Doğan E., Erdal F. & Saka M.P. (2009). Performance Evaluation of Metaheuristic Search Techniques in The Optimum Design of Real Size Pin Jointed Structures, Computers and Structures, Volume 87, 284-302.

Hasançebi O. & Kazemzadeh Azad S. (2015). Adaptive Dimensional Search: A new Metaheuristic Algorithm for Discrete Truss Sizing Optimization, Computers and Structures, Volume 154, 1-16.

Hasançebi O., Kazemzadeh Azad S. & Kazemzadeh Azad S. (2013). Automated Sizing of Truss Structures using a Computationally Improved Sopt Algorithm, International Journal of Optimization in Civil Engineering, Volume 3(2), 209-221.

Hasançebi O., Teke T & Pekcan O (2013). A Bat-inspired Algorithm for Structural Optimization, Computers and Structures, Volume 128, 77-90.

Hashimi F.A., Houssein E.H., Hussain K., Mobrouk M.S. & Al-Atabany W. (2022). Honey Badger Algorithm: New Metaheuristic Algorithm for solving Optimization Problems, Mathematics and Computers in Simulation, Volume 192, 84-110.

Jafari M., Salajegheh E. & Salajegheh J. (2019). An Efficient Hybrid of Elephant Herding Optimization and Cultural Algorithm for Optimal Design of Trusses, Engineering with Computers, Volume 35, 781-801.

Jawad F.K.J., Ozturk C., Dansheng W., Mahmood M., Al-Azzawi O. & Al-Jemely A. (2021). Sizing and Layout Optimization of Truss Structures with Artificial Bee Colony Algorithm, Structures, Volume 30, 546-559.

Kaveh A. & Ghazaan I. M. (2015). Layout And Size Optimization of Trusses with Natural Frequency Constraints using Improved Ray Optimization Algorithm, Iranian Journal of Science and Technology, Transactions of Civil Engineering, Volume 39, 395-408.

Kaveh A. & Khayatazad M. (2013) Ray Optimization for Size and Shape Optimization of Truss Structures, Computers and Structures, Volume 117, 82-94.

Kaveh A. & Mahdavi V.R. (2018). Multi-Objective Colliding Bodies Optimization Algorithm for Design of Trusses, Journal of Computational Design and Engineering, Volume 6, 49-59.

Kaveh A. & Talatahari S. (2009). Size Optimization of Space Trusses using Big Bang–Big Crunch Algorithm, Computers and Structures, Volume 87, 1129-1140.

Kazemzadeh Azad S. (2016). Enhanced Hybrid Metaheuristic Algorithms for Optimal Sizing of Steel Truss Structures with Numerous Discrete Variables, Springer, Structural and Multidisciplinary Optimization, Volume 55(6), 2159-2180.

Kazemzadeh Azad S. & Hasançebi O. (2014). An Elitist Self-Adaptive Step-Size Search for Structural Design Optimization, Applied Soft Computing, Volume 19, 226-235.

Kazemzadeh Azad S. & Hasançebi O. (2015). Discrete Sizing Optimization of Steel Trusses Under Multiple Displacement Constraints and Load Cases Using Guided Stochastic Search Technique, Springer, Structural and Multidisciplinary Optimization, Volume 52(2), 383-404.

Kazemzadeh Azad S., Hasançebi O., Kazemzadeh Azad S. & Erol O.K. (2013). Upper Bound Strategy in Optimum Design of Truss Structures: A Big Bang-Big Crunch Algorithm Based Application, Advances in Structural Engineering, Volume 16(6), 1035-1046.

Kazemzadeh Azad S., Hasançebi O. & Saka M.P. (2014). Guided Stochastic Search Technique for Discrete Sizing Optimization of Steel Trusses: A Design-Driven Heuristic Approach, Computers and Structures, Volume 134, 62-74.

Liu J. & Xia Y. (2022). A Hybrid Intelligent Genetic Algorithm for Truss Optimization Based on Deep Neutral Network, Swarm and Evolutionary Computation, Volume 73, 101-120.

Maute, K.K.C. (1998). Topologie- und Formoptimierung von dünnwandigen Tragwerken, Univ., Inst. für Baustatik, (p. 253).

Mirjalili S., Gandomi H.A., Mirjalili S.Z., Saremi S., Faris H. & Mirjalili S.M. (2017) A Bio-inspired Optimizer for Engineering Design Problems, Advances in Engineering Software, Volume 114, 1-29.

Pholdee N. & Bureerat S. (2018). A Comparative Study of Eighteen Self-adaptive Metaheuristic Algorithms for Truss Sizing Optimisation, KSCE Journal of Civil Engineering, Volume 22(8), 2982-2993.

Ramm E., Maute K. & Schwarz S. (1998). Conceptual design by structural optimization, In: Proceedings of the Euro–C 1998 Conference on Computational Modelling of Concrete (eds. Borst R. de et al.), Badgastein, Austria.

Sonmez M. (2018). Performance Comparison of Metaheuristic Algorithms for the Optimal Design of Space Trusses, Arabian Journal for Science and Engineering, Volume 43, 5264-5281.

Yalcin Y. & Pekcan O. (2020). Nuclear Fission-Nuclear Fusion Algorithm for Global Optimization: A Modified Big Bang-Big Crunch Algorithm, Neural Computing and Applications, Volume 32, 2751-2783.

Yapici H. & Cetinkaya N. (2019). A New Meta-Heuristic Optimizer: Pathfinder Algorithm, Applied Soft Computing Journal, Volume 78, 545-568.