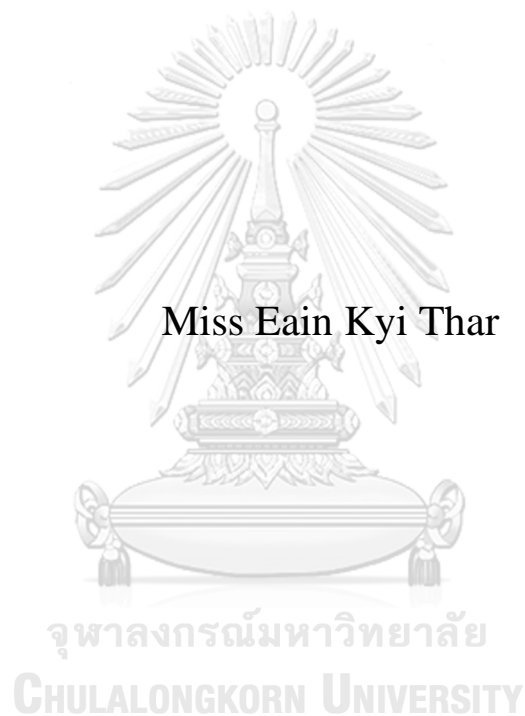


**CHAOS GAME OPTIMIZATION METHOD FOR WEIGHT
MINIMIZATION OF STEEL TRUSS STRUCTURE**



A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Civil Engineering
Department of Civil Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2021
Copyright of Chulalongkorn University

วิธีการเพิ่มประสิทธิภาพเกม **Chaos** สำหรับการลดน้ำหนักของโครงสร้างโครงเหล็ก



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมโยธา ภาควิชาวิศวกรรมโยธา
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2564
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

Thesis Title CHAOS GAME OPTIMIZATION METHOD
FOR WEIGHT MINIMIZATION OF STEEL TRUSS
STRUCTURE
By Miss Eain Kyi Thar
Field of Study Civil Engineering
Thesis Advisor Associate Professor SAWEKCHAI
TANGARAMVONG, Ph.D.

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University
in Partial Fulfillment of the Requirement for the Master of Engineering

..... Dean of the FACULTY OF
ENGINEERING
(Professor SUPOT TEACHAVORASINSKUN, Ph.D.)

THESIS COMMITTEE

..... Chairman
(Associate Professor PITCHA JONGVIVATSAKUL,
Ph.D.)

..... Thesis Advisor
(Associate Professor SAWEKCHAI
TANGARAMVONG, Ph.D.)

..... External Examiner
(Assistant Professor Pakawat Sancharoen, Ph.D.)



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

เอน จี ทาร์ : วิธีการเพิ่มประสิทธิภาพเกม Chaos สำหรับการลดน้ำหนักของโครงสร้างโครงเหล็ก. (CHAOS GAME OPTIMIZATION METHOD FOR WEIGHT MINIMIZATION OF STEEL TRUSS STRUCTURE) อ.ที่ปรึกษาหลัก : รศ. ดร. เสวกชัย ตั้งอร่ามวงศ์

งานวิจัยนี้ได้นำเสนอวิธีการที่เรียกว่า การหาความเหมาะสมที่สุดของผลลัพธ์เกมโกลาหล (Chaos Game Optimozation ; CGO) เพื่อออกแบบขนาดที่เหมาะสมที่สุดของโครงถักเหล็ก ภายใต้ระบบแรง การหาความเหมาะสมที่สุดของผลลัพธ์มีจุดมุ่งหมายเพื่อลดน้ำหนักโดยรวม (เกี่ยวข้องโดยตรงกับฟังก์ชันต้นทุน) ของโครงสร้างที่มีจุดรองรับ (Constrain) ที่อธิบายถึงการตอบสนองของโครงสร้างที่สอดคล้องกับข้อกำหนดของสภาพการจำกัด (Limit-State) แนวทางการวิเคราะห์โดยCGO มีแนวคิดมาจาก ทฤษฎีความโกลาหลพื้นฐาน โดยจะกำหนดรูปแบบทางเลขาคณิต (Fractal) ในการตอบสนองต่อระบบที่เป็นตัวแทนของพวกเขาระบบพลวัตที่จัดระเบียบด้วยตนเอง รูปแบบทางเลขาคณิตถูกสร้างขึ้นเป็นสามเหลี่ยม Sierpinski (เช่นพื้นที่ค้นหาจำนวนของผลลัพธ์) โดยมีจุดเริ่มต้นที่เลือกแบบสุ่มเพื่อทำแผนที่ของจุดและรูปร่างโดยรวมของสามเหลี่ยม ในการหาค่าตอบที่ดีที่สุดจะใช้เวลาน้อยในการคำนวณหาความเหมาะสมที่สุดของผลลัพธ์ ในการออกแบบขนาดที่เหมาะสมของโครงเหล็กภายใต้แรงที่ระบุแสดงให้เห็นถึงประสิทธิภาพและความแม่นยำของวิธีการ CGO โดยการเปรียบเทียบกับผลลัพธ์มาตรฐานอ้างอิง (Bench Mark) ที่มีอยู่บางส่วน (รวมถึงข้อมูลทางสถิติของผลลัพธ์) ที่แก้ไขได้ด้วยเทคนิค meta-heuristic คำตอบที่ดีที่สุดของผลลัพธ์มาตรฐานซึ่งมีสามตัวอย่างของโครงถักได้แก่ โครงถัก 10 ชั้น 72 ชั้น และ 200 ชั้น ได้รับการทดสอบและจับลงด้วยวิธีที่มีประสิทธิภาพด้วยการหาความเหมาะสมที่สุดของผลลัพธ์ โดย CGO และผลลัพธ์เหล่านี้จะถูกเปรียบเทียบกับวิธีการอื่นๆ การหาความเหมาะสมที่สุดของผลลัพธ์ โดย CGO พิสูจน์ให้เห็นว่าสามารถให้ผลลัพธ์ที่ดี และมีประสิทธิภาพกว่าวิธีการหาความเหมาะสมที่สุดของผลลัพธ์ จากงานวิจัยที่พัฒนาขึ้นก่อนหน้านี้

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

สาขาวิชา วิศวกรรมโยธา
ปีการศึกษา 2564

ลายมือชื่อนิสิต
ลายมือชื่อ อ.ที่ปรึกษาหลัก

6270366621 : MAJOR CIVIL ENGINEERING

KEYWORD Chaos Game Optimization, Meta-Heuristic Algorithm, ; Optimal
D: Sizing Design, Steel Trusses, Sierpinski Triangle

Eain Kyi Thar : CHAOS GAME OPTIMIZATION METHOD
FOR WEIGHT MINIMIZATION OF STEEL TRUSS STRUCTURE.

Advisor: Assoc. Prof. SAWEKCHAI TANGARAMVONG, Ph.D.

The research proposes the so-called chaos game optimization (CGO) method to perform the optimal sizing design of steel truss structures under applied loading regimes. The optimization problem aims at the minimization of the total weight (directly related to the cost function) of the designed structure subjected to the constraints describing the intrinsic structural responses complying with limit-state specifications. The CGO approach is based on the underlying chaos theory that establishes the primary patterns as fractals in responses to the systems representing them as self-organized dynamical systems. The fractal is generated as an initial Sierpinski triangle (i.e., search space of solution candidates) with randomly selected initial points to map out the sequence of points and hence the overall shape of the triangle. The optimal solutions can be obtained at a small amount of searching efforts. The applications in the optimal sizing design of steel trusses under specified forces illustrate the efficiency and accuracy of the CGO method through the good comparisons with some available benchmarks (including the statistical data of solutions) solved by other meta-heuristic techniques. The optimal solution of the benchmark results with three examples of 10-bar, 72-bar, 200-bar truss structures are tested which ends in an effective way with the CGO optimization and these results are compared to other methods. The Chaos Game Optimization proved that it is capable of providing the good result and outperforming the previously developed optimization methods.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Field of Study: Civil Engineering

Student's Signature

Academic Year: 2021

Advisor's Signature

Year:

.....

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deep and sincere gratitude to my advisor, Assoc. Prof. Dr. Sawekchai Tangaramvong for his continuous support of my study and research. His prompt inspirations, timely suggestions with kindness, dynamism, enthusiasm and motivation have enabled me to complete my thesis. It was a great privilege and honor to work and study under his guidance.

Besides my advisor, I would like to thank my thesis committee members for their encouragement and insightful comments.

I am extending my heartfelt thanks to all the administrative staffs in the Civil Engineering Department for their co-operation, helpfulness and furtherance.

My thanks and appreciations also go to my teachers, seniors, colleagues and people who have willingly helped me throughout my study and research period

I honestly thank Chulalongkorn University for offering the scholarship of Master's degree.

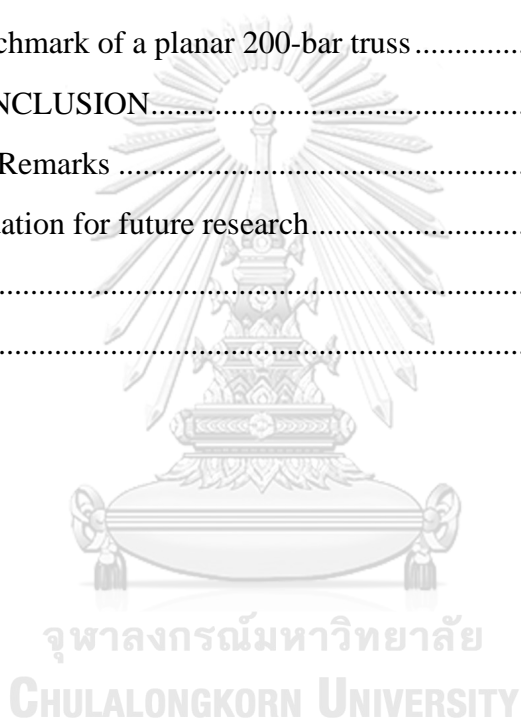
Finally, I owe a deep sense of gratitude to my supportive parents who provide unending inspirations.

Eain Kyi Thar

TABLE OF CONTENTS

	Page
.....	iii
ABSTRACT (THAI)	iii
.....	iv
ABSTRACT (ENGLISH).....	iv
ACKNOWLEDGEMENTS.....	v
TABLE OF CONTENTS.....	vi
LIST OF TABLES.....	viii
LIST OF FIGURES	ix
CHAPTER 1 INTRODUCTION.....	1
1.1 Background.....	1
1.2 Objectives of the Study.....	3
1.3 Scope of the Study	3
1.4 Methodology.....	4
CHAPTER 2 LITERATURE REVIEW.....	5
2.1 Overview of Structural Optimization	5
2.2 Optimum Design of Truss Structures	6
2.3 Problem Formulation of the Optimum Design Problem.....	8
2.4 Direct Stiffness Method.....	10
2.5 Structural Optimization Methods	12
2.5.1 Deterministic Techniques.....	12
2.5.1.1 Mathematical Programming Methods	12
2.5.1.2 Optimality criteria Methods	12
2.5.2 Stochastic Search Methods.....	13
2.6 Application of Meta-Heuristic Algorithms for Steel Truss Design.....	14
2.7 Application of CGO Algorithm for Steel Truss Design	17

CHAPTER 3 RESEARCH METHODOLOGY	18
3.1 Overview of Chaos Game Theory	18
3.2 Background of Chaos Game Theory	19
3.3 Mathematical model	21
3.4 Chaos Game Optimization Algorithm (CGO).....	27
CHAPTER 4 RESULTS AND DISCUSSIONS.....	29
4.1. Test on Benchmark of 10-bar Steel Truss Structure.....	29
4.2 Test on Benchmark of a space 72-bar truss	31
4.3 Test on Benchmark of a planar 200-bar truss	35
CHAPTER 5 CONCLUSION.....	41
5.1 Concluding Remarks	41
5.2 Recommendation for future research.....	41
REFERENCES	42
VITA.....	47



LIST OF TABLES

	Page
Table 1. Meta-Heuristic Algorithms	13
Table 2. Comparison of Optimization results obtained for the planar 10-bar plane truss	31
Table 3. Load cases of the 72-bar space truss structure.....	32
Table 4. Comparison of optimized designs of the 72-bar space truss	33
Table 5. Load cases of the 200-bar planar truss.....	35
Table 6. Element data of the 200-bar planar truss	36
Table 7. Comparison of optimized designs of the 200-bar planar truss	38

LIST OF FIGURES

	Page
Figure 1 Three main categories of structural optimization problems	5
Figure 2 Schematic configuration of an optimization process.....	6
Figure 3 Typical Two-node Truss element in Global and Local coordinates.....	10
Figure 4 unit displacement applying in global coordinate to determine corresponding displacement in local coordinate system, (i) at node 1, (ii) at node 2.....	11
Figure 5 Self-similarity of Mandelbrot set in different scales	20
Figure 6 The methodology of chaos game for creating Sierpinski triangle.....	20
Figure 7 The final shape and self-similarity of the Sierpinski triangle in different scales	20
Figure 8 The schematic view of creating temporary triangles.....	21
Figure 9 The schematic view of temporary triangles in the search space.	21
Figure 10 The schematic view of position update for (a) first seed , (b) second seed, (c) third seed and (d) fourth seed in the search space	25
Figure 11 Pseudo-code of the CGO algorithm	26
Figure 12 Flowchart of the CGO algorithm.....	28
Figure 13 A 10-bar truss	29
Figure 14 Solution convergency of 10-bar truss.....	30
Figure 15 A 72-bar space truss	32
Figure 16 Solution convergency of 72-bar space truss.....	34
Figure 17 A 200-bar planar truss	36
Figure 18 Solution convergency of 200-bar planar truss.....	40

CHAPTER 1

INTRODUCTION

1.1 Background

The majority of design issues in nature can be categorized as optimization issues, which necessitate the use of appropriate optimization techniques and algorithms. These days, design issues are so complicated that traditional optimization methods founded on mathematical concepts are unable to deliver some adequate outcomes in a fair amount of time. One of these mathematical approaches is gradient-based algorithms, which configures the optimization issue using the gradient of the objective function. Dealing with the shortcomings of traditional optimization algorithms and proposing new effective optimization algorithms have been major concerns over the past few decades. The introduction of new optimization algorithms that produce a high efficiency, great accuracy, and enhanced speed rate in handling challenging optimization issues is becoming more and more popular as a result of recent technological advancements. Additionally, there are certain other challenges that need to be addressed, such as the local optima problems and the non-smoothness and non-convexity of the search spaces, which have been quite problematic in this area [1].

These worries regarding optimization algorithms have prompted researchers and industry professionals to offer new, "Metaheuristic," algorithms for dealing with various optimization challenges. This phrase, which is made up of a core word (Heuristics) and a suffix (Meta), both of which have their roots in Greek words, was initially proposed by Glover in 1986 [2]. The word "heuristic" derives from the Greek word "heuriskein," which originally meant to discover new rules (strategies) in order to deal with various difficulties. The word "meta" refers to some higher-level techniques in nature. The term "metaheuristics" refers to a unique class of problem-solving techniques that take advantage of higher-level approaches to carry out a search process while taking into account certain special capabilities (such as avoiding local optimal results) in order to identify suitable solutions. Metaheuristic algorithms also make it possible to take into consideration design constraints by fusing an optimization process with precise engineering analysis. Trajectory-based algorithms and population-based algorithms are two categories of metaheuristic algorithms. Simulated Annealing (SA) method developed by Kirkpatrick et al. [3] is a trajectory-based algorithm,

while Harmony Search (HS) [4], Genetic Algorithm (GA) [5], Cuckoo Search [6], Particle Swarm Optimization (PSO)[7], Ant Colony Optimization (ACO)[8] are all population-based algorithms. New metaheuristic algorithms are also being created in order to enhance convergence behavior and optimization capabilities. For instance, Yang recently created the Flower Pollination Algorithm (FPA), a population-based metaheuristic technique [9], which imitates the nature of flower pollination.

Numerous techniques have been used to optimize truss structures. To discover a local optimum of the approximate problem, Adeli and Kamal, for instance, used a dual simplex technique to optimize space trusses while repeatedly solving the initial problem [10]. Discrete variables and GA with a penalty parameter based on constraint violation were employed by Rajeev and Krishnamoorthy [11]. For the ideal design of frame structures, Cao also used GA. [12]. To optimize the size and configuration of truss structures, Schutte and Groenwold employed PSO [13]. To reduce the total weight of the structure subject to stress and deflection limitations, Camp and Bichon used ACO [14]. By utilizing the HS method and continuous design variables, Lee and Geem created trusses that performed optimally under various loading circumstances [15]. Camp used the Big bang–big crunch (BB–BC) algorithm developed by Erol and Eksin [16] to create space trusses with the best possible design. Based on the particle swarm optimizer with passive congregation and an HS scheme, Li et al. created a heuristic particle swarm optimizer. This approach was effectively used to optimize the design of planar and spatial truss structures [17].

The chaos game optimization (CGO) method is recommended for performing the best sizing design for steel truss structures under the applied loading regimes. The objective of the optimization issue is to minimize the overall weight, which is directly related to the cost function, of the designed structure under the conditions of constraints specifying the intrinsic structural responses that satisfy limit-state requirements. The CGO method is founded on the underlying chaos theory, which establishes the fundamental patterns as fractals in reaction to the systems portraying them as self-organized dynamical systems [1]. The fractal is produced as an initial Sierpinski triangle (i.e., search space of solution candidates) with initially chosen starting points that are chosen at random in order to map out the points' order and subsequently the overall shape of the triangle. With minimal research effort, the optimal answers can be found. trusses Through accurate comparisons with various existing

benchmarks (including the statistical data of solutions) resolved by other meta-heuristic techniques, the applications in the optimal sizing design of steel under specified stresses demonstrate the effectiveness and correctness of the CGO method.

1.2 Objectives of the Study

The following are the objectives of this investigation.

- (a) To outline a useful and effective strategy based on the Chaos Game Optimization (CGO) that can optimize the design of truss structures.
- (b) To reduce the weight of the truss structure by using the Chaos Game Optimization (CGO) algorithm.
- (c) To satisfy the structural constraints that are applied to the building which are stresses, deflections, and lateral displacement by finite element analysis (FEA).

1.3 Scope of the Study

To accomplish the scope of this research, numerous parameters are investigated as follows:

- (a) An optimization test has been made on three different truss structures which are considered as the structural size optimization problems using the proposed method.
- (b) The stress and displacement serve as the design constraints in these issues.
- (c) This study has tested the optimization tool for truss structures with continuous variables.
- (d) The MATLAB program is applied to optimize and analyze with an iterative manner to minimize the total weight of the structure.
- (e) The violations of these constraints are considered as penalty functions and will have accounted them for the total weight.

1.4 Methodology

In this paper, a Chaos Game Optimization (CGO) is suggested. The central concept is founded on the ideas of chaos theory, and as inspiration, fractal configuration and self-similarity problems are used. The algorithm is tested on several well-known benchmark truss problems for sizing optimization and the obtained results were compared to those of some well-known meta-heuristics which successfully solved benchmarks highlighted efficiency optimization problems.

To investigate the efficiency and viability of the proposed CGO method, three prominent benchmarks of trusses will be explored for sizing optimization. The goal is to find the optimum cross-sections of the members by minimizing the weight of the structure. For the numerical purpose, the 10-bar, 72-bar, 200-bar truss structures as three of the benchmark problems are considered as design examples.

In addition, the direct stiffness method is applied to analyze the mass and stiffness of all benchmark structures. The coding procedure is implemented with MATLAB program. The optimization procedure terminates when the minimum cross-sections are obtained without violating its given constraints at the maximum number of iterations. The penalty function method will be applied to the weight minimization process. The outcomes of the suggested approach will be contrasted with those of other meta-heuristic optimization methods that have just been published in the literature.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of Structural Optimization

The three key categories for the best skeletal structure design are optimization of size, shape, and topology as shown in Figures 1.

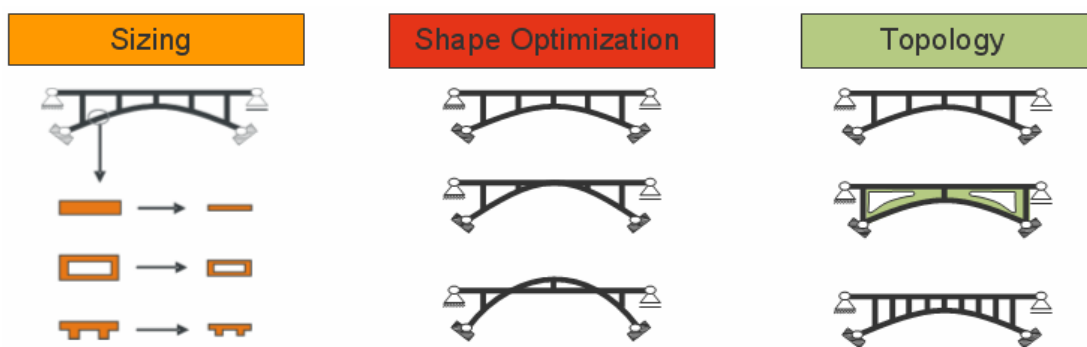


Figure 1 Three main categories of structural optimization problems

In shape optimization, the target is to find the best shape of the structure. In topology optimization, the target is to find the optimum structure by changing the amount and the location of material or components in the structure. While the cross-sectional areas of structural members are regarded as design variables in sizing optimization, they can be further classified into two subcategories, such as continuous and discrete [18]. In real-world applications, where structural components must be chosen from a range of offered sections by manufacturers, this is typically not the case for the continuous sizing optimization. In contrast, the most typical situation for structural optimization in practice is discrete optimization issues. However, the discrete variable space is hardly organized for algorithms to converge to good solutions of steel frame design optimization [19].

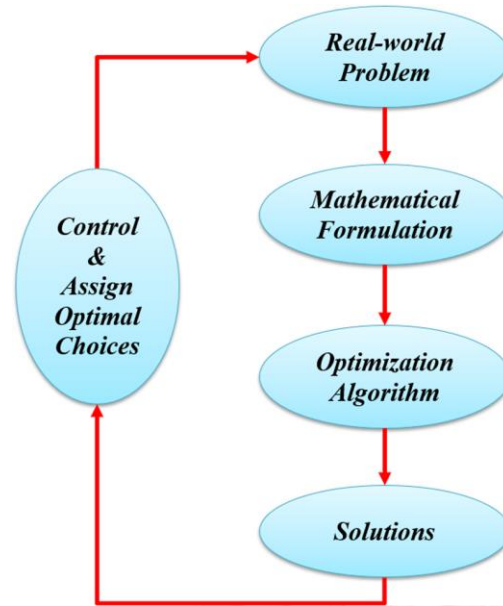


Figure 2 Schematic configuration of an optimization process.

2.2 Optimum Design of Truss Structures

The structural structures known as trusses are made up of N bars connected by nodes. The system is vulnerable to outside forces that are applied at the joints. The main goal of truss system structural optimization is to reduce the overall weight of the system. The optimization procedure and the truss structural analysis are both included in the suggested technique. The latter utilizes the stiffness method, and nodal displacements are computed in accordance with

$$\Delta = K^{-1}P \quad (1)$$

K and P in Eq. (1) stand for the external load vector, the system stiffness matrix, and the nodal displacement vector, respectively. By combining the element stiffness matrices in global coordinates and removing the row and column conditions, the system stiffness matrix is created. A bar element with three degrees of freedom at each node has the following stiffness matrix:

$$K_i = EA_i \begin{bmatrix} l^2 & lm & nl & -l^2 & -lm & -nl \\ lm & m^2 & mn & -lm & -m^2 & -mn \\ nl & mn & n^2 & -nl & -mn & -n^2 \\ -l^2 & -lm & -nl & l^2 & lm & nl \\ -lm & -m^2 & -mn & lm & m^2 & mn \\ -nl & -mn & n^2 & nl & mn & n^2 \end{bmatrix} \quad (2)$$

where,

$$l = \frac{L_{xi}}{L_i}, \quad m = \frac{L_{yi}}{L_i} \quad \text{and} \quad n = \frac{L_{zi}}{L_i} \quad (3)$$

The coordinates of the nodes and boundaries of the elements, which are established as the design constants, are used in Eqs. (2) and (3) to compute the total length of the bars (L_i) and the dimensions of the length in x, y, and z coordinates (L_{xi} , L_{yi} and L_{zi}). Additionally, design constants for bar materials include density (γ) and elasticity modulus (E). The design variables (X) of the optimization problem are the areas of the bars (A_i) (from $i = 1$ to N). The optimization's goal is to reduce the overall structural weight. That is

$$\min \quad W = \sum_{i=1}^N \lambda L_i A_i \quad (A_i \in R) \quad (4)$$

for the design variables:

$$X^T = [A_1, A_2, \dots, A_N] \quad (5)$$

within the ranges of

$$A^L \leq A_i \leq A^U \quad i = 1, N \quad (5')$$

subject to the stress ($g_1(X) \leq 0$) and displacement ($g_2(X) \leq 0$) constraints

$$\begin{aligned} g_1(X) : -\sigma^L \leq \sigma_i \leq \sigma^U & \quad i = 1, N_i \\ g_2(X) : \delta^L \leq \delta_j \leq \delta^U & \quad j = 1, N_j \end{aligned} \quad (6)$$

The lower and upper limits of the design variable solution ranges are designated as A^L and A^U , respectively. The displacement limits, δ^L and δ^U , often have equal absolute values but the opposite signs. The displacement of nodes (δ_j) from $j=1$ to N_j (for a system with j nodes) are the components of the displacement vector:

$$\Delta = \begin{bmatrix} \delta_1 \\ \delta_2 \\ \cdot \\ \cdot \\ \cdot \\ \delta_{N_j-1} \\ \delta_N \end{bmatrix} \quad (\delta_{1,N_j} \in R) \quad (7)$$

σ^L and σ^U are two different types of stress limits which are for compression (σ^L in - sign) and tension (σ^U in + sign). The stresses of a bar (σ_i^G) in global coordinate are calculated by

$$\sigma_j^G = \frac{K_i \Delta_i}{A_i}, \quad i = 1, N \quad (8)$$

where Δ_i is the vector of the nodal displacements of i^{th} bar. Global stresses are multiplied by directional cosines to determine the axial stress on a bar (σ_i). The same rows are home to various design factors while creating the system stiffness matrix (K). Since there are several design variables in the equilibrium equations obtained in each row, the optimization issue is coupled. It is impossible to achieve nodal displacements according to Eq. (1) and stresses according to Eq. (8) without making these design variables assumptions. These numbers can be seen as the upper and lower bounds of the design restrictions given by $g_1(X)$ and $g_2(X)$, however the best outcome never involves maximizing all nodal displacements and bar stresses.

Additionally, bar stresses may be higher than the limit when all displacements are at their limits. Different loads and time periods may stimulate the structures in different ways. When design variables are altered in accordance with various loading scenarios, the maximum stresses change as a result of the interaction of the bars. Large structural systems might not be able to handle the quantity of design variations. All of these factors can make it impossible for us to use mathematical optimization techniques to solve structural design issues. In this case, metaheuristic approaches that employ randomly generated design variables are appropriate options for completing such jobs. Thus, it is possible to resolve the linked equilibrium equation. The best design variables can be discovered by performing iterative studies.

2.3 Problem Formulation of the Optimum Design Problem

The optimization problem endeavors to minimize the total weight (W) of the structure under the limited design constraints. The member cross-sectional areas, namely A_d for each d -th member are the design variables and the constraints consider the stresses developed in the element (i.e., tensile, compression and buckling) and the nodal displacement of joints. This can be mathematically described as follows:

$$\begin{aligned}
&\text{Find} && A_d \text{ for } \forall d \in \{1, \dots, n_d\} \\
&\text{Minimize} && W = \sum_{d=1}^{n_d} \rho_d A_d L_d \\
&\text{Subjected to} && \sigma_{\min} \leq \omega_d \leq \sigma_{\max}, \\
&&& \delta_{\min} \leq \delta_d \leq \delta_{\max}, \\
&&& A_{\min} \leq A_d \leq A_{\max}
\end{aligned} \tag{9}$$

where n_d is the total number of (pin-jointed) truss members, ρ_d is the material density of the structure, L_d is the length of a generic d -th member, σ_d is the stress developed in each d -th member, and δ_d is the nodal displacement of joints of each d -th member, σ_{\min} and σ_{\max} are the lower and upper limits on the element stresses, δ_{\min} and δ_{\max} are the lower and upper limits on the nodal displacements and A_{\min} and A_{\max} the lower and upper limits on the available sectional areas, respectively.

Truss structure designs must adhere to the limited design constraints in order to be optimal. In this study, a penalty approach f has been applied to the total weight (W) of the structure to be accounted for the design infeasibility. Then, the problem in Eq. (9) is reformulated as:

$$\left. \begin{aligned}
f &= W(1 + \varepsilon_1 C)^{\varepsilon_2} \\
C &= \sum_{i=1}^{n_d} c_d^\sigma + \sum_{i=1}^{n_d} c_d^\delta
\end{aligned} \right\} \tag{10}$$

$$\text{where } c_d^\sigma = \begin{cases} \left| \frac{\sigma_d}{\sigma_d^*} \right| - 1, & \text{if } \sigma_d > \sigma_d^* \\ 0, & \text{if } \sigma_d \leq \sigma_d^* \end{cases}, \quad c_d^\delta = \begin{cases} \left| \frac{\delta_d}{\delta_d^*} \right| - 1, & \text{if } \delta_d > \delta_d^* \\ 0, & \text{if } \delta_d \leq \delta_d^* \end{cases},$$

The penalty factor C is associated with the violation of the design constraints and the parameters ε_1 and ε_2 are set to 1 and 2, respectively. c_d^σ and c_d^δ are the parameters that indicates the satisfaction or violation of the stress and nodal displacement. σ_d^* and δ_d^* denote as the allowable stress and the allowable nodal displacement.

2.4 Direct Stiffness Method

The direct stiffness method is carried out to perform the truss element stiffness depending on their material properties, section properties, and member configurations assuming that the truss is loaded at the joints as the concentrated loads and the member of the truss is subjected to axial forces only which remain constant along the length of the member. Additionally, the joints are also presumed as frictionless pins or internal hinges. The element local stiffness matrix can be indicated as:

$$[k_e'] = \frac{EA_d}{L_d} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

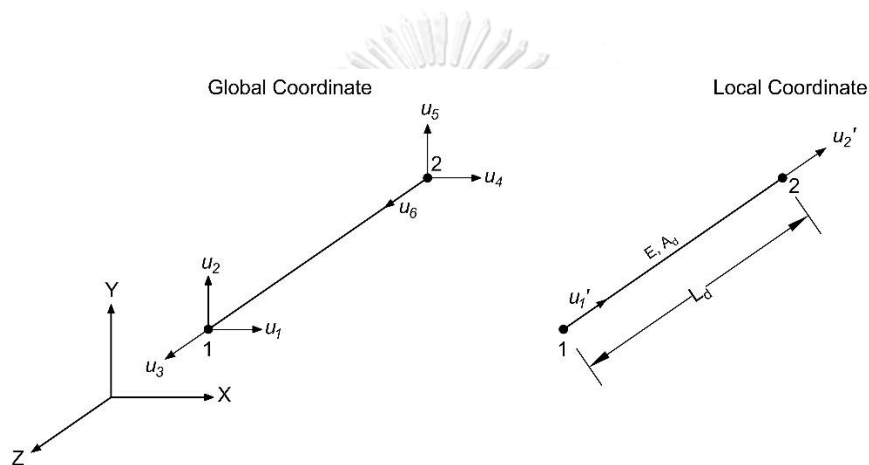


Figure 3 Typical Two-node Truss element in Global and Local coordinates.

A unit displacement is applied in the global coordinate for the purpose of determining the corresponding displacement in local coordinate. The relationship between the displacements of local and global coordinates is computed as follows:

$$\begin{Bmatrix} u_1' \\ u_2' \end{Bmatrix} = \begin{bmatrix} C_x & C_y & C_z & 0 & 0 & 0 \\ 0 & 0 & 0 & C_x & C_y & C_z \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{Bmatrix}$$

$$C_x = \cos \alpha, C_y = \cos \beta, C_z = \cos \gamma$$

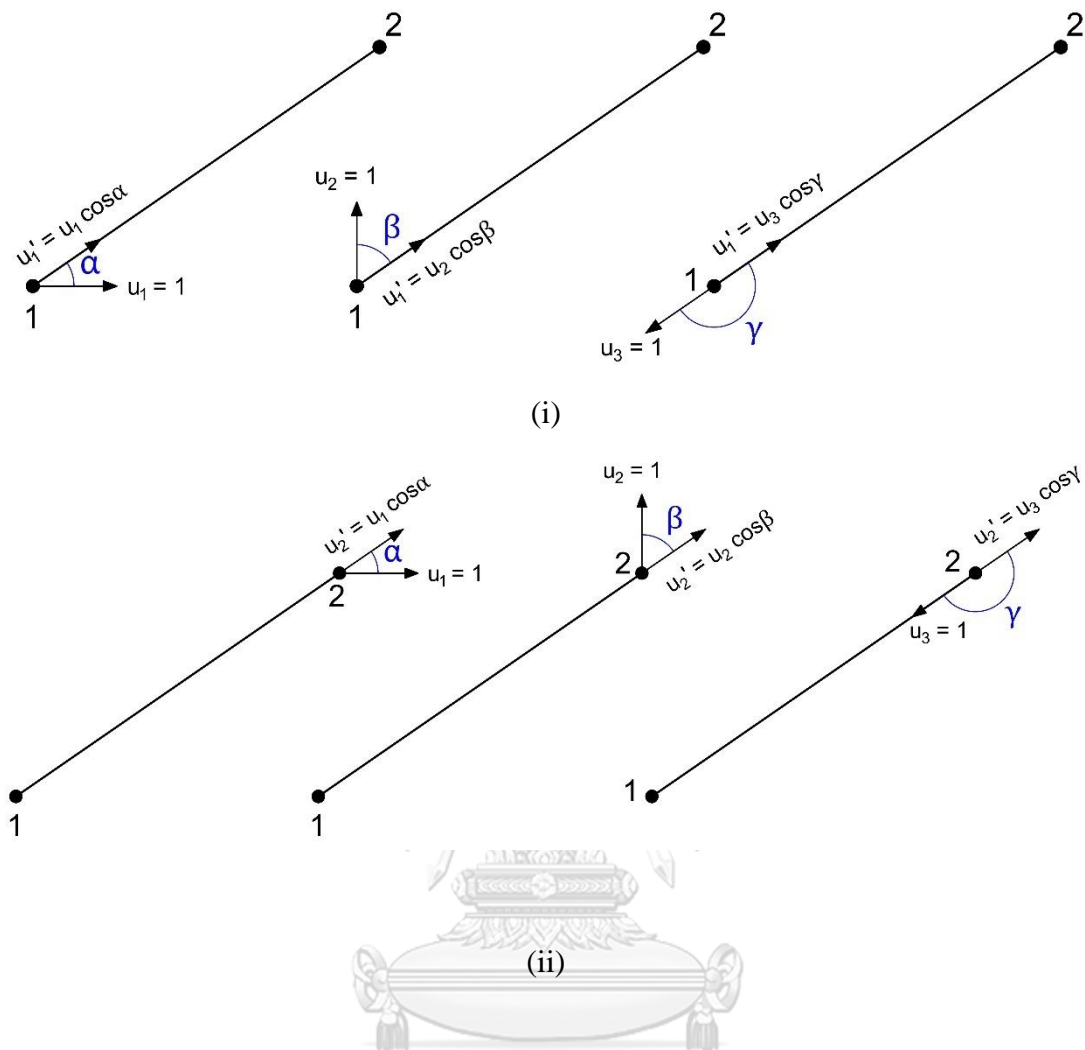


Figure 4 unit displacement applying in global coordinate to determine corresponding displacement in local coordinate system, (i) at node 1, (ii) at node 2.

Then, the global stiffness matrix for a truss element can be stated as follows:

$$[k_e] = \frac{EA_d}{L_d} \begin{bmatrix} C_x & 0 \\ C_y & 0 \\ C_z & 0 \\ 0 & C_x \\ 0 & C_y \\ 0 & C_z \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} C_x & C_y & C_z & 0 & 0 & 0 \\ 0 & 0 & 0 & C_x & C_y & C_z \end{bmatrix}$$

$$[k_e] = \frac{EA_d}{L_d} \begin{bmatrix} C_x^2 & C_x C_y & C_x C_z & -C_x^2 & -C_x C_y & -C_x C_z \\ C_x C_y & C_y^2 & C_y C_z & -C_x C_y & -C_y^2 & -C_y C_z \\ C_x C_z & C_y C_z & C_z^2 & -C_x C_z & -C_y C_z & -C_z^2 \\ -C_x^2 & -C_x C_y & -C_x C_z & C_x^2 & C_x C_y & C_x C_z \\ -C_x C_y & -C_y^2 & -C_y C_z & C_x C_y & C_y^2 & C_y C_z \\ -C_x C_z & -C_y C_z & -C_z^2 & C_x C_z & C_y C_z & C_z^2 \end{bmatrix}$$

2.5 Structural Optimization Methods

The objective and constraint functions are non-smooth and non-convex optimization problems in the optimal design of steel frames in order to be optimal. Over the past years, many different algorithms for optimization have been developed. Most of these algorithms are based on numerical methods which can be typically classified into two groups including deterministic and stochastic techniques [20].

2.5.1 Deterministic Techniques

2.5.1.1 Mathematical Programming Methods

The objective function's gradient information as well as constraints pertaining to the design variables are necessary for programming methodologies. In the case of minimization problems, to find the next point until there is no major discrepancy between the design variable values within two consecutive iterations, they take a step in the negative direction of the gradient of the objective function. There are various mathematical programming methods for solving complicated optimization problems. Several techniques include sequential linear programs, penalty function methods and gradient methods. However, when using these techniques to the design real-size practical steel frames, numerical difficulties were met [20].

2.5.1.2 Optimality criteria Methods

The way solving the optimum design problem of optimality criteria methods is different from that of the mathematical programming methods. While mathematical programming techniques exert to minimize the objective function directly taking into account the constraint conditions, the optimality criteria methods derive a criterion based on intuitive such as fully stressed design or a mathematical statement such as Kuhn-Tucker conditions. The objective function of optimality criteria methods is formulated in form of a Lagrangian function instead of the original one. They then establish an iteration procedure to achieve this

criterion. The mathematical programming techniques are more general, but optimality criteria methods are computationally more effective. However, there are some circumstances when they might not lead to the best answer [20].

2.5.2 Stochastic Search Methods

Both continuous and discrete optimization problems can be successfully solved using stochastic search strategies. The fundamental idea behind these methods is to mimic natural phenomena like the cooling process, immune system, swarm intelligence, and the survival of the fittest. These techniques deviate from conventional stochastic search and do not require knowledge of the gradient or the convexity of the objective function and constraints. Furthermore, they employ probabilistic transition rules rather than deterministic ones [20]. Numerous of researcher have paid much attention on the meta-heuristic search procedures. The meta-heuristic techniques are successfully used in the optimum design steel frames. Ali Kaveh and Ghazaan [21] reported that some of the well-known methods being a genetic (GA) is inspired by Darwin's theory about biological evolution. The Simulated Annealing (SA) algorithm makes use of the energy minimization that takes place during the cooling of molten metals. The notion of the Harmony Search (HS) algorithm was inspired by the musical practice of seeking for the ideal condition of harmony. Charged system search (CSS) directs the charged particles according to the Newtonian rules of mechanics and the electric laws of physics.

Especially, Swarm intelligence (SI) based algorithm is one of the best choices to obtain the optimum solutions by using special strategies. The algorithm based on collective behaviors of animals such as birds, insects, or fishes. Particle swarm optimization (PSO), which models the social interaction behavior of flocking birds and schooling fish, is one of the most popular swarm intelligence algorithms. Ant colony optimization (ACO) mimics how ant colonies determine the quickest path between the food source and their nest. The flashing patterns and behaviors of fireflies serve as the basis for the Firefly Algorithm (FA). Table.1 given below is to illustrate the development of different algorithms with respect to their development years [22].

Table 1. Meta-Heuristic Algorithms

2010	Charge system, Bat Search Algorithm,
2009	Cuckoo Algorithm
2007	Firefly Algorithm, Improved Harmony Search
2005	Bee Colony Algorithm, Glowworm Swarm Optimization
2001	Harmony Search

1995	Particle Swarm Optimization
1992	Ant Colony Optimization
1989	Swarm Intelligence
1986	Tubu Search, Artificial Immune System
1983	Simulated Annealing
1970	Genetic Algorithm

According to Saka and Geem [20], they also have the following negative aspects. The first is that it is impossible to show whether the optimum solution they achieve is the global optimum or is close to the global optimum because they do not employ mathematical derivations. The second is that they work with random numbers, and they have several param needing to be given values by the user. The third drawback is that they need many structural analyses which becomes computationally expensive for the large size steel frames. It is therefore difficult to predict which of these strategies will be adopted as the norm for the design tools in the finite element programs.

2.6 Application of Meta-Heuristic Algorithms for Steel Truss Design

A huge number of studies have been performed by applying meta-heuristic methods to solve the optimum design of steel trusses. An extensive review of metaheuristic techniques employed in developing optimum design algorithms for steel trusses in the literature until now.

Lamberti [23] and Saka [24] Lamberti recently examined the state-of-the-art for the use of metaheuristic algorithms in weight or cost optimization of skeletal systems. Dorigo et al. [25] first introduced the ACO for optimization issues. The procedure imitates how ant colonies forage in the real world.

By using stochastic combinatorial optimization, the ACO aims to simulate some of the key traits seen in ant behavior [26]. By employing pheromone trails, ants can create the quickest route between their colony and the source of their food and return [23]. The approach has been utilized for structural system design optimization in addition to its other uses. For example, truss structures were optimized by Camp and Bichon [14], Capriles et

al. [27], Serra and Venini [28], and Hasancebi et al. [29]. Frame structures were optimized by Camp et al. [26], Kaveh and Shojaee [30], Hasancebi et al. [31], Kaveh and Talatahari [32].

For the purpose of resolving combinatorial optimization issues, Geem et al. [4] developed HS. The approach is based on a comparison between the search for optimal harmony in music and the search for answers to optimization issues. For instance, the optimum design method aims to discover the optimum solution as indicated by the goal function, similar to jazz improvisation which seeks to achieve musically satisfying harmony [26]. Following works by Lee and Geem [15] and Lee et al. [33] that employed HS to optimize truss structures, HS has now been used to a number of structural optimization issues, such as the best design for geodesic domes [34], [grillage](#) systems [35], steel frames and trusses. In addition to the traditional HS algorithm implementation created several unique HS characteristics. For instance, Lamberti and Pappalettere [36] developed a better harmony search formulation in which trial designs are produced with knowledge of the cost function gradients. Compared to conventional harmony search and other meta-heuristic optimization algorithms, the novel HS formulation finished the optimization process with much less iterations [38]. An adaptive harmony search approach for structure optimization was put out by Hasancebi et al. [37]. Internal constant values that are acceptable are assigned in the typical implementation of HS. As a result, the chosen parameter value set directly affects how effective HS is. In order to find the most effective optimization procedure, Hasancebi et al. 's harmony search algorithm [37] adopts a novel method for automatically altering internal parameters.

The PSO method was developed by Kennedy and Eberhart [7]. It is predicated on the idea that social information exchange among species members provides an evolutionary benefit [38]. An objective function's search space is initialized at random with a number of particles that represent the swarm. Each member of the swarm is a potential answer to the optimal design issue. The current position, a velocity vector, and a time increment are used to update the positions of the particles as they move across the search space [26]. PSO has been used in optimization of skeletal structures [39], [13], [17], [40]. Researchers updated the PSO standard For the purpose of sizing optimization of truss structures, For the purpose of sizing optimization of truss structures, For the purpose of sizing optimization of truss structures, For the purpose of sizing optimization of truss structures, implementation with new functionality.

A PSO strategy and an HS method are combined in the heuristic particle swarm optimizer (HPSO) developed by Li et al. [19], [42]. For truss constructions with both discrete [43] and continuous variables, Particle swarmer, ant colony optimization, and harmony search are inventions of Kaveh and Talatahari [41], [42]. The strategy combines a particle swarm optimizer with harmony search technique, passive congregation (PSOPC), and ant colony optimization (ACO) (HS).

The BB-BC model that Erol and Eksin presented simulates theories about the development of the cosmos. This hypothesis states that the Big Bang phase is characterized by chaos and energy dissipation, whereas the Big Crunch phase sees the order of randomly scattered particles being brought into it [43]. To optimize the size of truss constructions, the BB-BC algorithm was used [44]. In order to improve convergence capability of standard BB-BC algorithm, Kaveh and Talatahari [45] created the hybrid BB-BC (HBB-BC) algorithm to increase the convergence capabilities of the ordinary BB-BC algorithm and to optimize space trusses and ribbed domes. Two stages make up the HBB-BC method: a Big Bang phase in which potential solutions are dispersed randomly throughout the search space, and a Big Crunch phase acting as a convergence operator where the center of mass is produced [45].

For the purpose of optimizing numerical functions, Karaboga [46] invented the ABC approach initially. The ABC is an optimization technique based on the brilliant behavior of a swarm of honey bees. Each food source that the bees utilize in the ABC approach indicates a potential resolution to a specific optimization challenge. The design variables and fitness function are represented by the position and quantity of nectar from the flower patch, respectively [47]. Truss structure size optimization using the ABC has been done effectively for both continuous [48] and discrete variables. When results from the ABC algorithm are compared to those from other meta-heuristic techniques, it is clear that the ABC algorithm produces results that are at least as excellent as those of other optimization algorithms for truss structure optimization [48].

For limited mechanical design optimization issues, Rao et al. [48] have developed a brand-new optimization technique named "teaching-learning-based optimization (TLBO). The approach is based on how learners interact with one another and how a teacher influences them. In order to show the robustness of TLBO, Rao et al. [50] provided five distinct restricted benchmark test functions. The design example results were compared to

those from various meta-heuristic optimization techniques. The comparisons revealed that the TLBO outperformed other meta-heuristic optimization techniques while requiring less computer work. The TLBO approach was created by Rao et al. [49] for solving large-scale non-linear optimization issues. The TLBO approach is used to optimize five distinct benchmark problems, and the outcomes are compared to those obtained using the GA, ant colony system, bee algorithm, and grenade explosion method. The outcomes demonstrated the TLBO method's efficacy in terms of computing effort, consistency, and providing solutions that are close to optimal. The TLBO was utilized for the best design of planar steel frames [50] following the groundbreaking work of Rao et al. [49]. Three steel frames that had previously been optimized by the GA, HS, and enhanced ACO were used to demonstrate the effectiveness of the approach. The TLBO approach outperformed the GA, ACO, HS, and improved ACO in terms of the number of analyses and the outcomes for the frames included in the research [52].

2.7 Application of CGO Algorithm for Steel Truss Design

In this study, the so-called Chaos Game Optimization (CGO) metaheuristic is proposed. The CGO algorithm's basic idea is based on some chaos theory concepts, which put fractal self-similarity concerns and fractal configuration using chaos game approach into perspective. Different metaheuristic algorithms have used the fractals as the primary or secondary notion, such as the Stochastic Fractal Search (SFS) method developed by Salimi [51], Fractal-Based Algorithm (FBA) developed by Kaedi [52], Fractal Decomposition-Based Algorithm (FDA) presented by Nakib et al. [53], and Fractal Triangle Search (FTS) algorithm proposed by Rodrigues et al. [54]. In addition, some more enhanced versions of metaheuristic algorithms have been put out in which the general formulation of these algorithms for various purposes incorporates the chaos theory.

It should be mentioned that the technique of the CGO algorithm is entirely distinct from the prior research, according to the provided literature study. The Chaos Game Theory (CGO) algorithm bases its general formulation on the game theory and uses it as its primary conceptual framework.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Overview of Chaos Game Theory

The study of dynamical systems with unique properties that are very sensitive to their beginning conditions is the focus of the mathematical field known as chaos theory. Chaos theory indicates the existence of some fundamental patterns in the behavior of these dynamical systems, such as similar loops, repeated templates, fractals, and multiple sub-systems, which represent them as self-similar and self-organized dynamical systems despite the randomness of these systems [55]. The chaos theory demonstrates how a dynamical system's dependency on its starting conditions causes even slight changes to its beginning conditions to have severe effects on the system's future states. According to this theory, the current state of a system could predict its future state, as opposed to the approximate present state of a system, which cannot predict its future state precisely. The majority of chaotic processes have fractal graphic forms. A fractal is a subset of Euclidean space in mathematics where a certain geometric design is reproduced on various scales. Fractals are self-similar systems because they have roughly similar shapes at various scales. One of the most well-known fractals is the Mandelbrot set, as shown in Fig. 5, represents a complex infinite border in which various recursive aspects are gradually displayed at various scales.

In this study, the CGO is put forth as a metaheuristic algorithm that is conceptualized in terms of chaos game theory, a well-known method for making strategic decisions in mathematics. To put it another way, the CGO technique is based on this theory, which is seen as a general iterated system for simulating the decision-making of different individuals. Furthermore, the mathematical representation of this approach is a fully detailed model that includes both general and specific aspects of the chaos game theory. The difficulties in producing exceptional or even acceptable outcomes while dealing with various optimization problems can be overcome by creating a metaheuristic with solid inspiration and a mathematical model, such as CGO. The suggested CGO technique is a parameter-free metaheuristic algorithm, meaning that no internal parameter needs to be found during the optimization process. The parameter algorithm of this technique is one of its most outstanding features, to put it simply. Additionally, the solution's location updating method

allows for more precise local and global searching across the whole search space, producing outstanding results.

3.2 Background of Chaos Game Theory

The chaotic game is a method for making fractals in mathematics that starts with a polygonal shape and a randomly chosen initial point. To produce a sketch with a comparable shape at various scales, the primary goal is to iteratively build a series of points [55]. In this regard, it is first important to correctly position the vertices of a polygon, which is thought to define the main shape of the fractal. Afterwards, a randomly chosen initial point is chosen to serve as the fractal's starting point. The following point in the series, which is based on the original point, is calculated as a portion of the distance between the initial point and a randomly chosen polygonal vertex in each iteration. A fractal is produced by continuously repeating this process while taking into account the random initial point and the random vertex selection in each iteration. By utilizing three vertices with the factor of $1/2$, a Sierpinski triangle is created. When the fractal's initial vertex count reaches N , a Sierpinski Simplex with $N-1$ dimensions can be produced.

The systematic construction of a Sierpinski triangle using the chaos game methodology is shown as a straightforward example. In order to start building the main shape of the fractal, which in this case is a triangle, three vertices must first be chosen. One of the red, blue, or green colors designates each of the chosen vertices. It is decided to roll a die with two red, two blue, and two green faces. In this example, the initial random point that serves as the fractal's starting point or seed is chosen. The beginning point's seed is moved toward the linked vertex by half the distance between them as the dice are rolled, depending on whatever color comes up. The seed is transferred to the appropriate vertex in accordance with the new position of the seed, which is used as the starting point for the next roll of the dice. The ultimate shape is the Sierpinski triangle, which is created by repeatedly rolling the dice. Fig. 6 shows the proposed methodology's schematic view, and Fig. 7 shows the Sierpinski triangle's final form and its self-similarity at various scales.

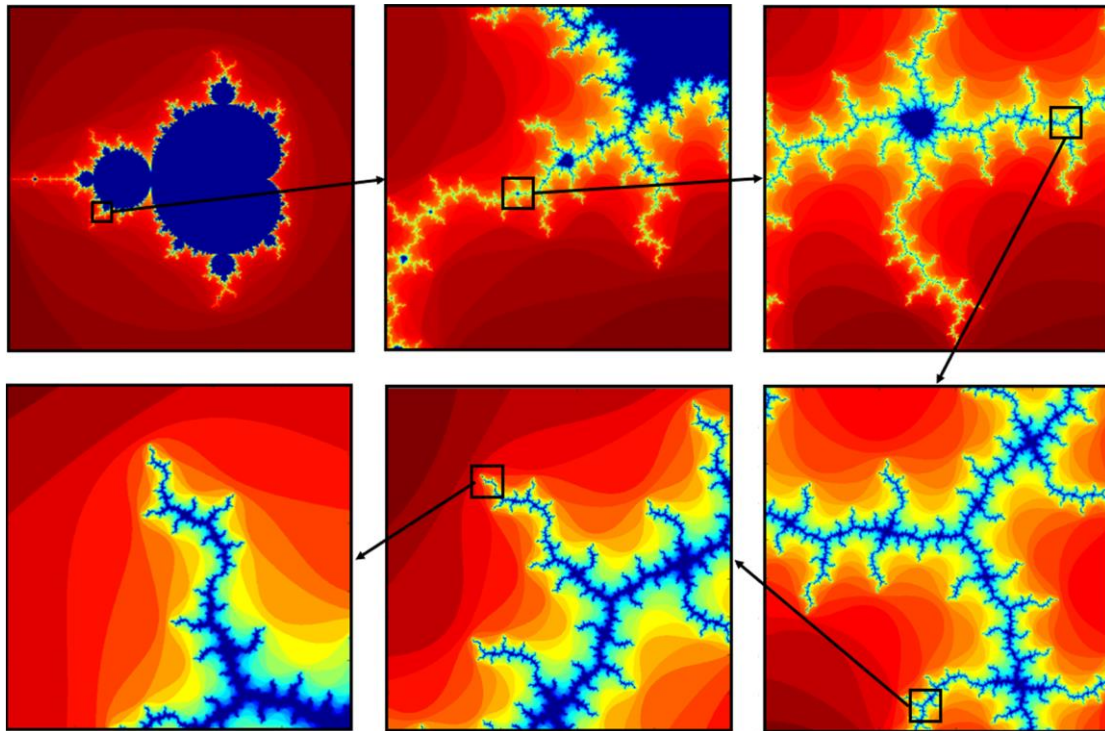


Figure 5 Self-similarity of Mandelbrot set in different scales

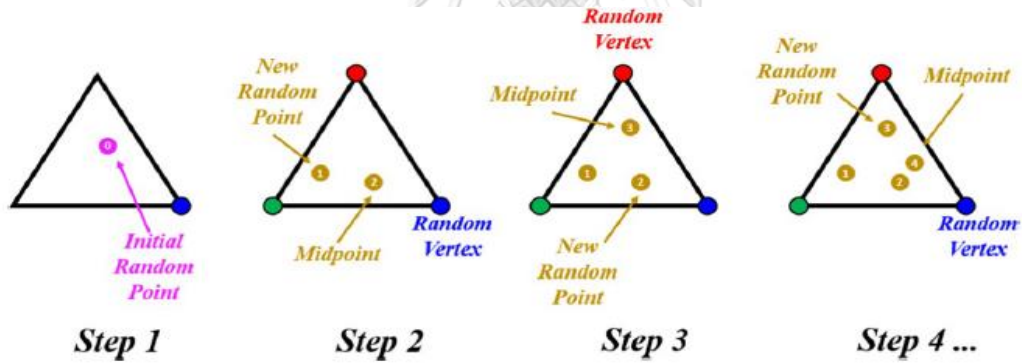


Figure 6 The methodology of chaos game for creating Sierpinski triangle

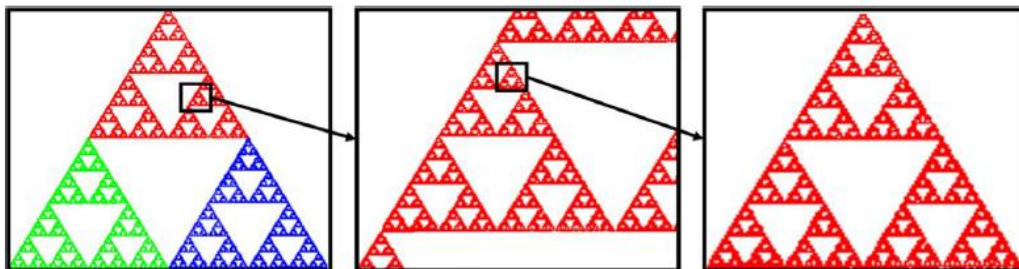


Figure 7 The final shape and self-similarity of the Sierpinski triangle in different scales

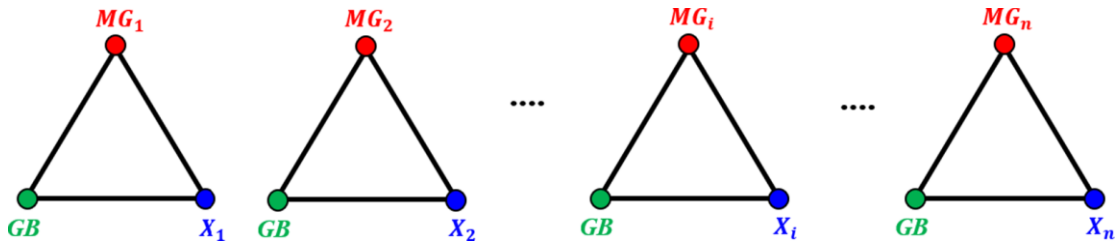


Figure 8 The schematic view of creating temporary triangles.

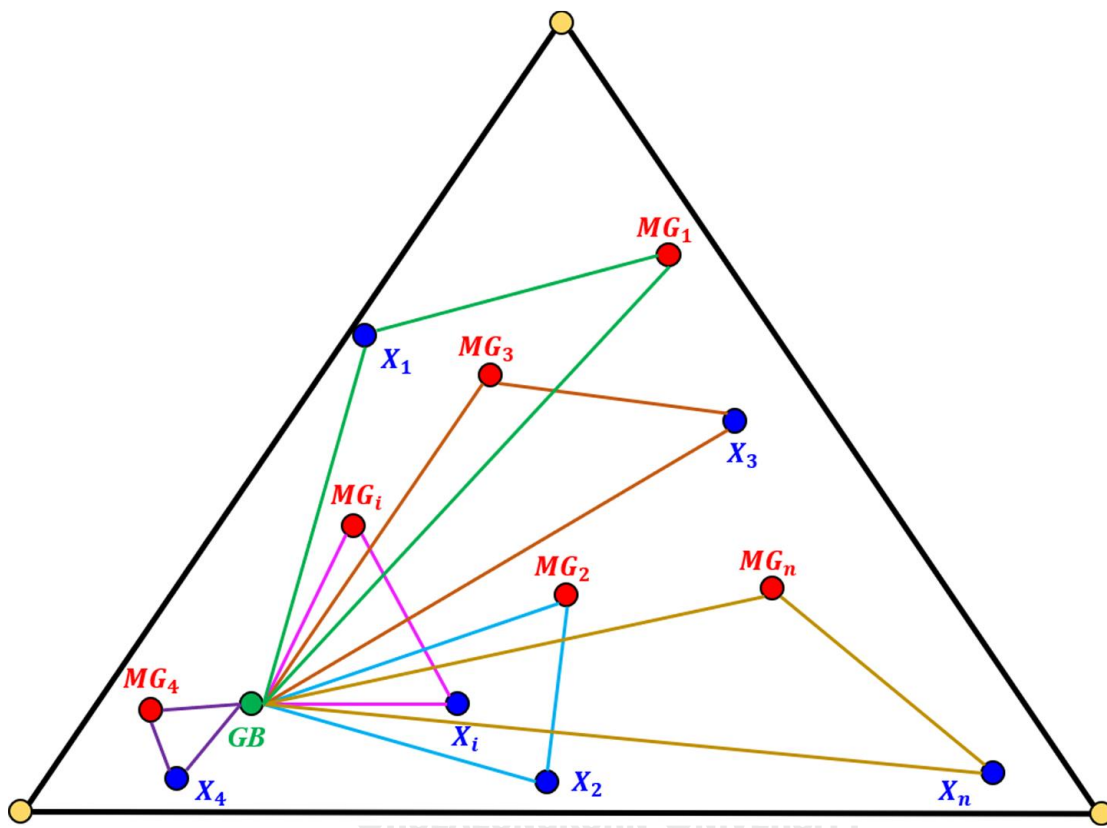


Figure 9 The schematic view of temporary triangles in the search space.

3.3 Mathematical model

The fundamentals of the chaos theory are discussed in this part, and an optimization strategy is suggested. The basic concepts of the chaotic game and fractals are used to create a mathematical model for the CGO algorithm. Due to the fact that many natural evolution algorithms preserve a population of solutions that are evolved through random changes and selection, the CGO technique takes into consideration a number of solution candidates (X) in this regard that represent some eligible seeds inside a Sierpinski triangle [55]. Each solution candidate (X_i) in this technique is made up of some decision variables (x_i^j) that indicate

where these eligible seeds are located inside a Sierpinski triangle. In the optimization algorithm, the Sierpinski triangle is regarded as the search space for potential solutions. These aspects are presented mathematically as follows:

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^j & \cdots & x_1^d \\ x_2^1 & x_2^2 & \cdots & x_2^j & \cdots & x_2^d \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_i^1 & x_i^2 & \cdots & x_i^j & \cdots & x_i^d \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n^1 & x_n^2 & \cdots & x_n^j & \cdots & x_n^d \end{bmatrix}, \quad \begin{cases} i = 1, 2, \dots, n. \\ j = 1, 2, \dots, d. \end{cases} \quad (1)$$

where n is the total number of potential seeds (solution candidates) contained within the Sierpinski triangle (search space), and d is their dimension.

These qualifying seeds' starting places in the search space are chosen at random and are as follows:

$$x_i^j(0) = x_{i,\min}^j + \text{rand.} \cdot (x_{i,\max}^j - x_{i,\min}^j), \quad \begin{cases} i = 1, 2, \dots, n. \\ j = 1, 2, \dots, d. \end{cases} \quad (2)$$

where $x_i^j(0)$ determines the initial position of the eligible seeds; $x_{i,\min}^j$ $x_{i,\max}^j$ are the minimum and maximum allowable values for the j th decision variable of the i th solution candidate; rand is a random number in the interval of $[0,1]$.

As was previously said, the underlying patterns in the behavior of dynamical systems that demonstrate that they are self-similar and self-organized systems are what the principles of chaos theory are concerned with. The produced initial seeds, also known as "eligible seeds," represent the fundamental patterns of the chaotic dynamical systems. An optimization problem's solution candidates (X) can be used to describe the self-similarity of these seeds, which determines their suitability to serve as the primary patterns. The greatest and worst fitness values are represented, respectively, by the solution candidates with the highest and lowest eligibility levels.

In order to complete the overall Sierpinski triangle form, the essential idea behind this mathematical model is to create many suitable seeds within the search area. In this sense, the Sierpinski triangle-based seed-creation approach is also applied. A temporary triangle is created with three seeds as follows for each of the eligible seeds in the search space (X_i);

- The position of the so far found Global Best (GB),

- The position of the Mean Group (MG_i),
- The position of the i th solution candidate (X_i) as the selected seed.

The most qualified candidate for the best solution found thus far is referred to as the GB . The term MG_i refers to the average values of certain randomly chosen eligible seeds, each of which has an equal chance of being the currently accepted initial eligible seed (X_i). The three vertices of a Sierpinski triangle are the GB , MG_i , and the chosen eligible seed (X_i). As previously stated, a temporary triangle is formed for each of the initial eligible seeds in the search space with the intention of producing some additional seeds inside the search space that could be regarded as fresh eligible seeds for finishing the Sierpinski triangle. In Figures 8 and 9, respectively, the schematic perspective and thorough schematic description of the process of manufacturing temporary triangles are shown.

In order to produce fresh viable seeds in the search space, temporary triangles are primarily used as follows [55]. Four methods are created to achieve this goal. The n suitable seeds that were obtained in the previous iteration are included in the i th temporary triangle (i th iteration), along with three Sierpinski triangle vertices: the GB (green seed), MG_i (red seed), and X_i (blue seed). A dice, three seeds, and the chaotic game methodology are used to generate fresh seeds in this temporary triangle. The first seed is placed in the X_i , followed by the second in the GB , and the third in the MG_i . Using a dice with three green faces and three red faces, the first seed is placed. The seed in the X_i is shifted toward the GB (green face) or the MG_i (red face) depending on which color (red or green) appears on the dice. The dice is rolled and based on which color comes up (green or red), the seed in the X_i is moved toward the GB (green face) or the MG_i (red face). This feature is represented by a random integer generating function that only generates the values 0 and 1 and allows the user to choose between green or red faces. If the green face comes up, the seed positioned in the X_i is moving toward the GB but if the red face comes up, the seed positioned in the X_i is moving toward the MG_i . The possibility of producing two identical random integers for the GB and the MG_i is also taken into consideration, with the seed in the X_i moving toward a point of the connected lines between the GB and the MG_i , despite the fact that each green or red face has an equal chance of appearing in the game. Some randomly generated factorials are used in this purpose to control this element because the mobility of the seeds in the search space should be restricted as a result of the chaos game methodology. Figure 10a provides a

schematic representation of the first seed's described process, and the following equations represent it mathematically:

$$Seed_i^1 = X_i + \alpha_i \times (\beta_i \times GB - \gamma_i \times MG_i), \quad i = 1, 2, \dots, n \quad (3)$$

where X_i is the i^{th} solution candidate, GB is the so far found global best, and MG_i is the mean values of some selected eligible seeds. α_i is the randomly generated factorial for modelling the movement limitations of the seeds while each of the β_i and γ_i represent a random integer of 0 or 1 for modelling the possibility of rolling a dice.

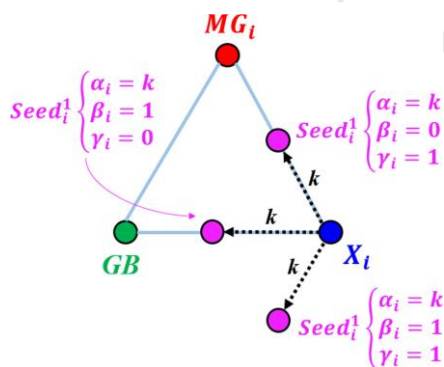
For second seed and third seed is also performed by the same process and the following equations are utilized.

$$Seed_i^2 = GB + \alpha_i \times (\beta_i \times X_i - \gamma_i \times MG_i), \quad i = 1, 2, \dots, n \quad (4)$$

$$Seed_i^3 = MG_i + \alpha_i \times (\beta_i \times X_i - \gamma_i \times GB), \quad i = 1, 2, \dots, n \quad (5)$$

Another procedure is also used to produce the fourth seed in order to implement the mutation phase in the position updates of the eligible seeds in the search space [55]. This seed's location updates are dependent on some random changes to the randomly selected decision factors. Figure 10 d provides a visual representation of the specified procedure for the fourth seed while the following equations represent this feature mathematically:

$$Seed_i^4 = X_i(x_i^k + R), \quad k = [1, 2, \dots, d]$$



(6)

where k is a random integer in the interval of $[1, d]$ and R is a uniformly distributed random number in the interval of $[0,1]$.

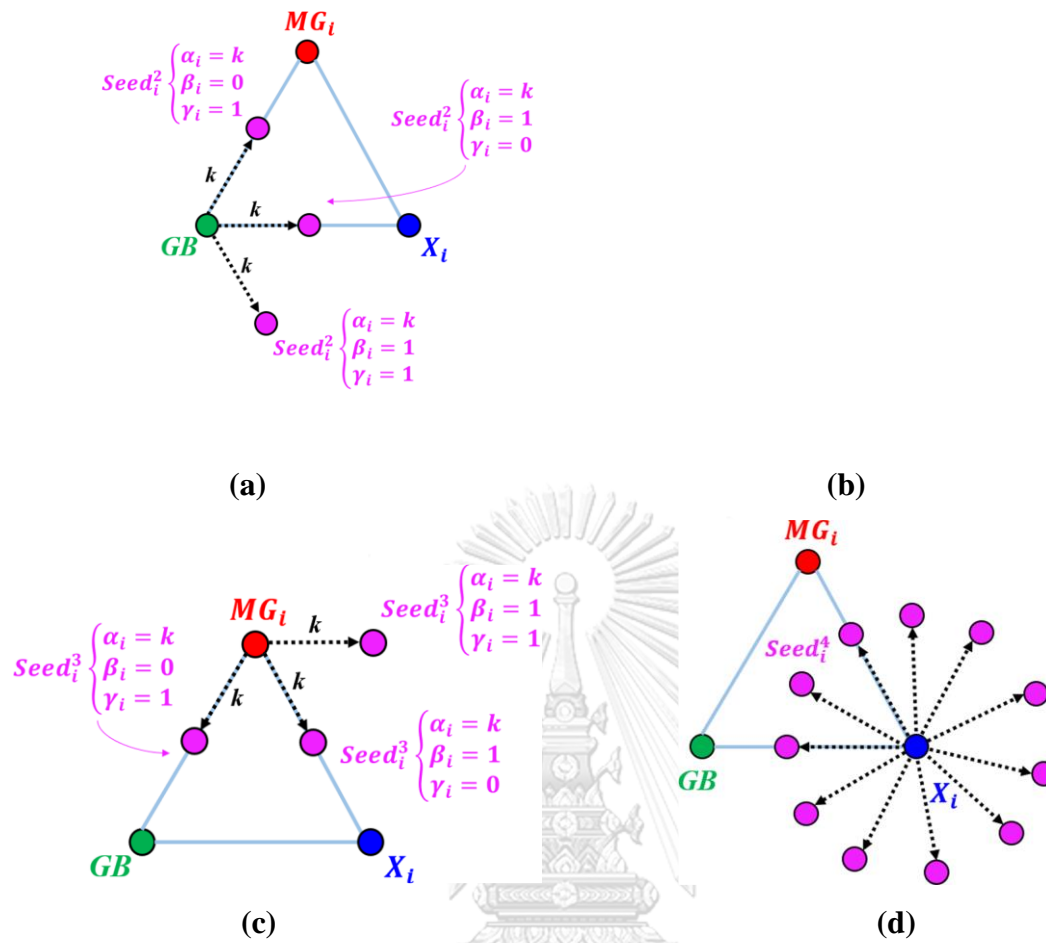


Figure 10 The schematic view of position update for (a) first seed , (b) second seed, (c) third seed and (d) fourth seed in the search space

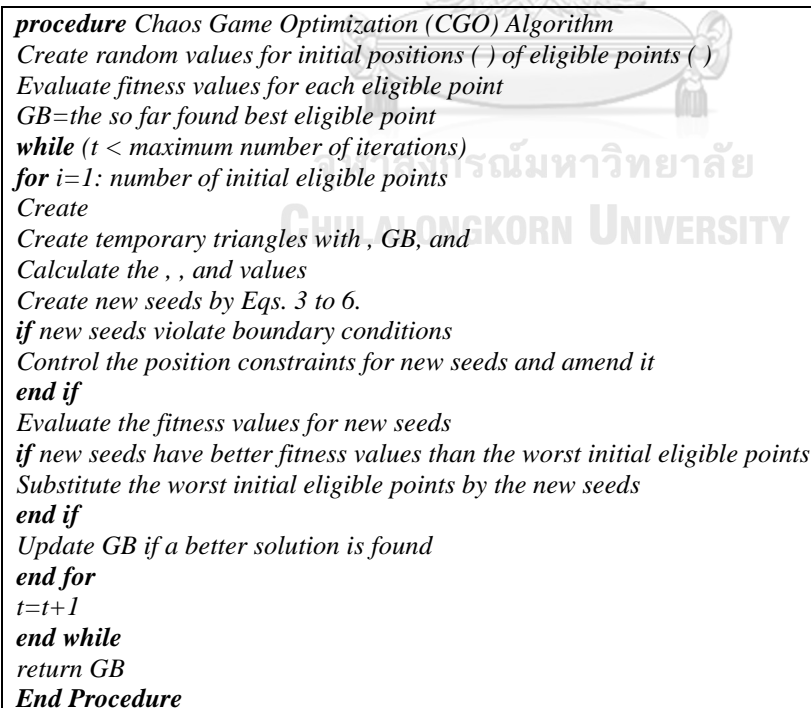
Four separate equations are used to determine the value of the variable α_i which is concerned with modeling the mobility constraints of the seeds, in the proposed innovative CGO algorithm in order to regulate and adapt the exploration and exploitation rate [55]. Each of the equations (3) to (5), which determine the order of the first to third seeds, uses one of these equations at random.

$$\alpha_i = \begin{cases} Rand \\ 2 \times Rand \\ (\delta \times Rand) + 1 \\ (\varepsilon \times Rand) + (\varepsilon) \end{cases} \quad (7)$$

where $Rand$ is a uniformly distributed random number in the interval of $[0,1]$, while δ and ε are random integers in the interval of $[0,1]$.

To decide whether or not the new seeds should be included in the total eligible points in the search space based on the self-similarity problems in the fractals, one must take into account the eligibility of the starting points as well as the newly created seeds by the chaotic game idea. When comparing new solution candidates to starting ones, the new points (seeds) should be swapped out for the initial points with the lowest fitness values, which correspond to the lowest levels of self-similarity. It should be emphasized that the mathematical approach uses the replacement process to reduce the complexity of the mathematical model. In actuality, the general shape of the Sierpinski triangle is formed by using all of the eligible points that have been discovered in the search area so far.

In order to cope with solution variables (x_i') that violate the boundary conditions of the variables, a mathematical flag is constructed for (x_i') outside the variables range, which directs a boundary adjustment for the violating variables. The termination criterion is the maximum number of iterations allowed before the optimization process is terminated after a predefined number of iterations [55]. The CGO algorithm's step-by-step configuration is as follows, and Fig. 11 shows the algorithm's pseudo-code. Additionally, Fig. 12 displays the CGO algorithm's flowchart.



```

procedure Chaos Game Optimization (CGO) Algorithm
  Create random values for initial positions ( ) of eligible points ( )
  Evaluate fitness values for each eligible point
  GB=the so far found best eligible point
  while (t < maximum number of iterations)
  for i=1: number of initial eligible points
    Create
    Create temporary triangles with , GB, and
    Calculate the , , and values
    Create new seeds by Eqs. 3 to 6.
    if new seeds violate boundary conditions
      Control the position constraints for new seeds and amend it
    end if
    Evaluate the fitness values for new seeds
    if new seeds have better fitness values than the worst initial eligible points
      Substitute the worst initial eligible points by the new seeds
    end if
    Update GB if a better solution is found
  end for
  t=t+1
  end while
  return GB
End Procedure

```

Figure 11 Pseudo-code of the CGO algorithm

3.4 Chaos Game Optimization Algorithm (CGO)

The step-by-step procedure of the CGO algorithm is as follows:

- *Step 1.* The initial positions of solution candidates (X) or the initial seeds that are eligible are set at random in the search space.
- *Step 2.* Based on the self-similarity of the initial eligible seeds, the fitness values of the initial solution candidates are computed.
- *Step 3.* The seed with the greatest levels of eligibility is identified as the Global Best
- *Step 4.* A Mean Group (MG_i) for each valid seed (X_i) in the search space is established.
- *Step 5.* A temporary triangle is created in the search space with the three vertices X_i , GB , and MG_i for each acceptable seed (X_i).
- *Step 6.* For each of the temporary triangles, α_i , β_i , and γ_i values are calculated.
- *Step 7.* Four seeds are made based on the Eqs. 3-6 for every temporary triangle.
- *Step 8.* A boundary condition check is performed for the fresh seeds whose x_i^j is outside the variable range.
- *Step 9.* Based on self-similarity problems, the fitness values of the new seeds are computed.
- *Step 10.* The new seeds are used to replace the available eligible seeds with the lowest fitness values and lowest degrees of self-similarity.
- *Step 11.* It is tested against the terminating criterion.

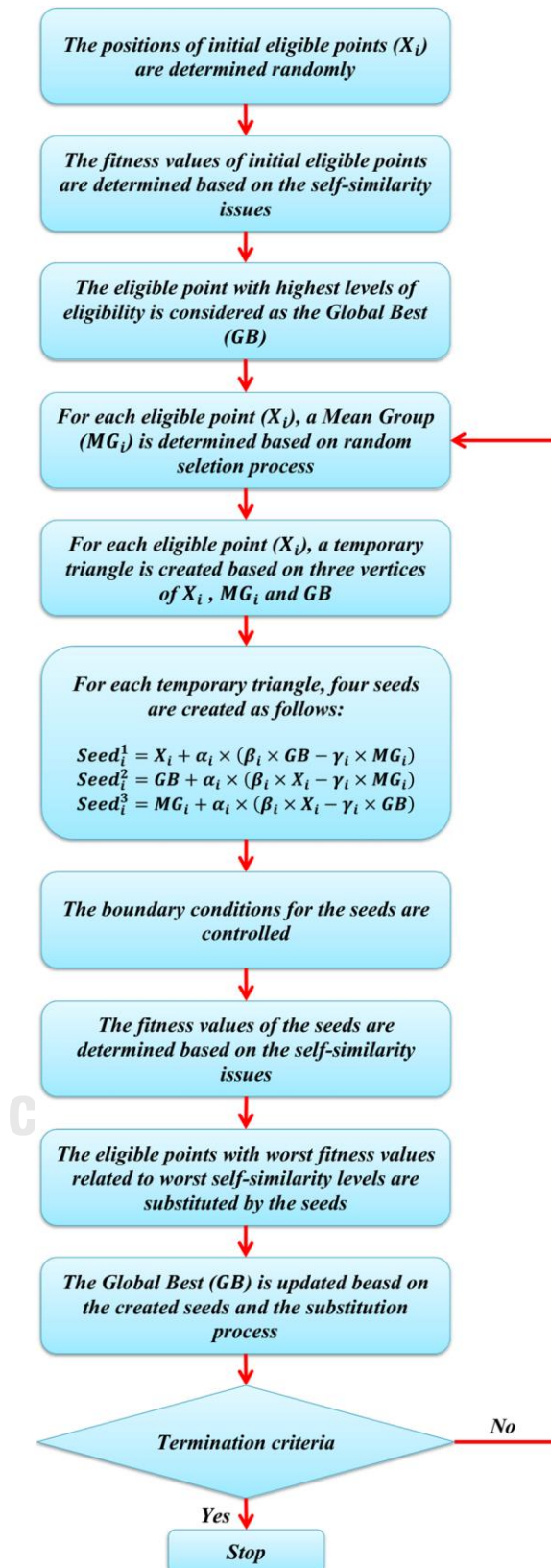


Figure 12 Flowchart of the CGO algorithm

CHAPTER 4

RESULTS AND DISCUSSIONS

In this chapter, the optimal sizing designs of three popular benchmark truss problems of 10-bar, 72-bar and 200-bar were successfully performed by the proposed CGO method.

4.1. Test on Benchmark of 10-bar Steel Truss Structure

The geometry of the planar 10-bar truss is presented in Fig. 13. For all members, the unit weight of material is 0.1 lb/in^3 and the modulus of elasticity is 10^7 psi . All members are subjected to the stress constraints of $\pm 25 \text{ ksi}$. The maximum allowable displacement is $\pm 2 \text{ in}$ in both vertical and horizontal directions for all nodes. Two vertical loads, P of the magnitude of 10^5 lb are applied at the node 2 and 4. The cross-sectional areas of each member are considered as design variables which are varying from 0.1 in^2 to 35 in^2 .

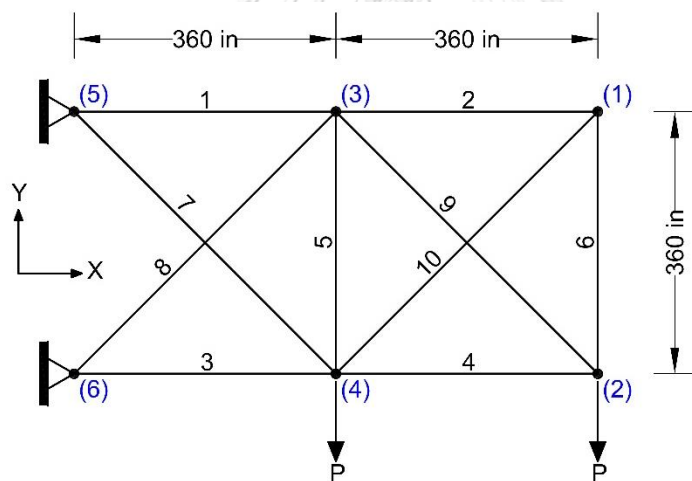


Figure 13 A 10-bar truss

The effectiveness of the CGO approach is tested on the 10-bar truss. The results confirm the validity and provide a comparison between the performance of the proposed algorithm and the other meta-heuristic optimization in the literature.

Table 2 summarizes the resulting areas of members and the optimal weight $W = 5061.09 \text{ lb}$ by the CGO method in evaluation with the solutions from other metaheuristic methods. More explicitly, the designed weight values are 5060.92 lb by heuristic particle

swarm optimization (HPSO) [17] , 5057.88 lb by harmony search algorithm (HSA) [15], 5060.82 lb by improved harmony search (IHS) [36], 5024.21 lb by particle swarm optimization (PSO) [38] , 5060.88 lb by artificial bee colony (ABC-AP) [47], 5062.39 lb by efficient harmony search (EHS) [56] and 5061.42 lb by self-adaptive harmony search (SAHS) [56] , and 5060.96 lb by teaching learning based optimization (TLBO) [57] . The optimal result from the proposed method is comparatively close to the other methods with the modest numerical efforts. The solution convergence with the number of analysis (up to 100) iterations for this example is represented in Fig. 14.

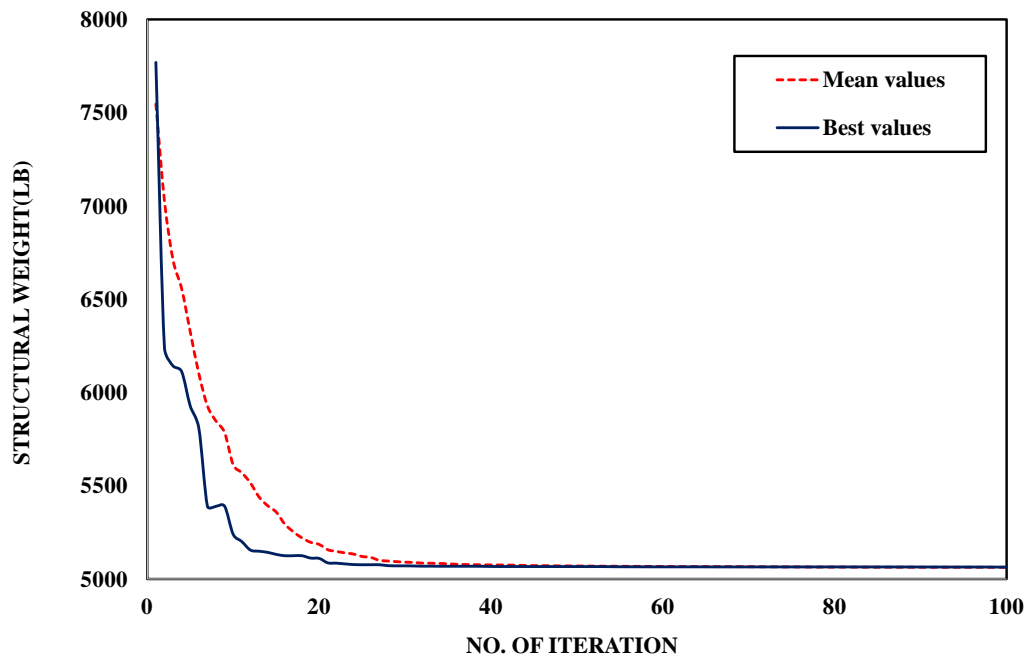


Figure 14 Solution convergency of 10-bar truss

The figure 14 of solution convergency and table 2 of results obtained from the 10-bar truss and the table of comparison result, the optimization results and the convergence history developed and compared with applying different algorithms for the 10-bar steel truss problem. The algorithm took only couple of minutes to converge the optimum solution within 50 analysis iterations. In this study, the optimum weight has the good optimization results along with the other algorithms.

Table 2. Comparison of Optimization results obtained for the planar 10-bar plane truss

Design variables (areas)	Li et al. [17]	Lee et al. [15]	Lamberti and Pappalettere [36]	Perez and Bedhinan [38]	Sonmez [47]	Degertekin [56]		Degertekin and Hayalioglu [57]	Present CGO
	HPSO	HSA	IHS	PSO	ABC-AP	EHS	SAHS	TLBO	CGO
A ₁	30.704	30.15	30.522	33.5	30.548	30.208	30.394	30.429	30.523
A ₂	0.1	0.102	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A ₃	23.167	22.71	23.201	22.766	23.18	22.698	23.098	23.244	23.394
A ₄	15.183	15.27	15.223	14.417	15.218	15.275	15.491	15.368	15.273
A ₅	0.1	0.102	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A ₆	0.551	0.544	0.551	0.1	0.551	0.529	0.529	0.575	0.55
A ₇	7.46	7.541	7.457	7.534	7.463	7.558	7.488	7.440	7.414
A ₈	20.978	21.56	21.037	20.467	21.058	21.559	21.189	20.967	20.819
A ₉	21.508	21.45	21.529	20.392	21.501	21.491	21.342	21.533	21.619
A ₁₀	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Weight(lb)	5060.92	5057.88	5060.82	5024.21	5060.88	5062.39	5061.42	5060.96	5061.09
Number of analyses	1250000	20000	1350		500000	9791	7081	16872	2500
Worst Weight(lb)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	5063.02
Mean Weight(lb)	N/A	N/A	N/A	N/A	N/A	5063.73	5061.95	N/A	5062.02

4.2 Test on Benchmark of a space 72-bar truss

The geometry of the space 72-bar truss is shown in Fig. 15. The material has a unit weight of 0.1 lb/in³ and the modulus of elasticity of 10⁷ psi for all members. The maximum allowable stresses of ± 25 ksi are considered as stress constraints for all members of the truss. The maximum allowable nodal displacements of ± 0.25 in both vertical and horizontal directions are considered as displacement constraints for all nodes of the truss. The continuous design variables are the cross-sectional areas of each member varying from 0.1 in² to 100 in². The load distribution on nodes is listed in Table 3. The members are divided into 16 groups: (1) A₁-A₄, (2) A₅-A₁₂, (3) A₁₃-A₁₆, (4) A₁₇-A₁₈, (5) A₁₉-A₂₂, (6) A₂₃-A₃₀, (7) A₃₁-

A₃₄, (8) A₃₅-A₃₆, (9) A₃₇-A₄₀, (10) A₄₁-A₄₈, (11) A₄₉-A₅₂, (12) A₅₃-A₅₄, (13) A₅₅-A₅₈, (14) A₅₉-A₆₆, (15) A₆₇-A₇₀, and (16) A₇₁-A₇₂.

Table 3. Load cases of the 72-bar space truss structure

Nodes	Load case 1			Load case 2		
	P _x (kips)	P _y (kips)	P _z (kips)	P _x (kips)	P _y (kips)	P _z (kips)
17	5	5	-5	0	0	-5
18	0	0	0	0	0	-5
19	0	0	0	0	0	-5
20	0	0	0	0	0	-5

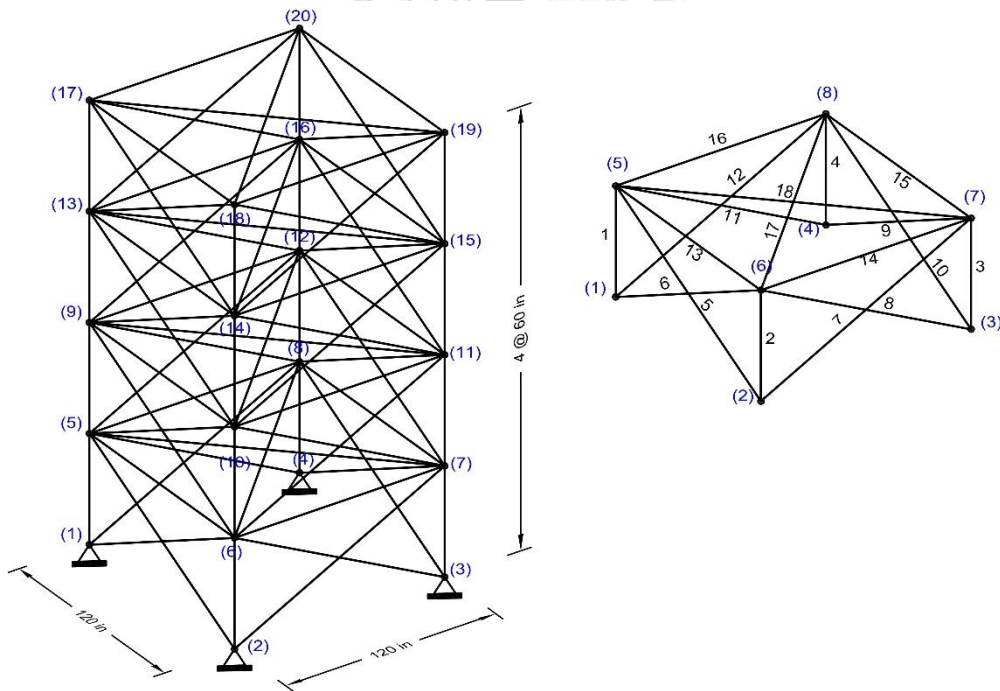


Figure 15 A 72-bar space truss

Table 4. Comparison of optimized designs of the 72-bar space truss

Design variables	Li et al.	Lee et al.	Perez and Bedinan	Camp	Jalili and Hosseinzadeh	Degertekin		Degertekin and Hayalioglu	Present
(areas)	[17]	[15]	[38]	[44]	[58]	[56]		[57]	
	HPSO	HSA	PSO	BB-BC	CA	EHS	SAHS	TLBO	CGO
A₁-A₄	1.857	1.79	1.743	1.858	1.861	1.967	1.86	1.906	1.867
A₅-A₁₂	0.505	0.521	0.519	0.506	0.509	0.51	0.521	0.506	0.514
A₁₃-A₁₆	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A₁₇-A₁₈	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A₁₉-A₂₂	1.255	1.229	1.308	1.248	1.263	1.293	1.271	1.262	1.241
A₂₃-A₃₀	0.503	0.522	0.519	0.527	0.504	0.511	0.509	0.511	0.511
A₃₁-A₃₄	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A₃₅-A₃₆	0.1	0.1	0.1	0.101	0.1	0.1	0.1	0.1	0.1
A₃₇-A₄₀	0.496	0.517	0.514	0.521	0.523	0.499	0.485	0.532	0.486
A₄₁-A₄₈	0.506	0.504	0.546	0.517	0.525	0.501	0.501	0.516	0.511
A₄₉-A₅₂	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A₅₃-A₅₄	0.1	0.101	0.11	0.101	0.103	0.1	0.1	0.1	0.1
A₅₅-A₅₈	0.1	0.156	0.162	0.157	0.156	0.16	0.168	0.156	0.1
A₅₉-A₆₆	0.524	0.547	0.509	0.551	0.553	0.522	0.584	0.549	0.518
A₆₇-A₇₀	0.4	0.442	0.497	0.392	0.42	0.478	0.433	0.41	0.377
A₇₁-A₇₂	0.534	0.59	0.562	0.592	0.562	0.591	0.52	0.57	0.532
Weight(lb)	369.65	379.27	381.91	379.85	379.69	381.03	380.62	379.63	369.88
Number of analyses	125000	20000		19621	18460	15044	13742	19709	25000
Worst Weight(lb)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	370.56
Mean Weight(lb)	N/A	N/A	N/A	382.08	380.86	383.51	382.42	380.2	370.02

Table 4 represents the best structural weight of the CGO algorithm in comparison of other metaheuristic algorithms such as heuristic particle swarm optimization (HPSO) [17] , harmony search algorithm (HSA) [15] , particle swarm optimization (PSO) [38] , big bang-

big crunch (BB-BC) [44] , cultural algorithm (CA) [58], efficient harmony search (EHS) [56], self-adaptive harmony search (SAHS) [56] , and teaching learning based optimization (TLBO) [57] . In this design example, the CGO algorithm gives the best result of (369.88 lb) in all other metaheuristic algorithms. The optimum weights of other algorithms are HPSO (369.65 lb), HSA (379.27 lb), PSO (381.91 lb), BB-BC (379.85 lb), CA (379.69 lb), EHS (381.03 lb), SAHS (380.62 lb), and TLBO (379.63 lb) individually.

The CGO algorithm progressively performs the number of structural analyses than most of the algorithms in the literature. The optimization history of the CGO algorithm is described in Fig. 16 with respect to the variation of the best penalized weight and the number of iterations. The optimal design of the steel space truss was successfully performed by the proposed CGO method within 40 analysis iterations. The solution (total weight) convergence with the number of analysis (up to 100) iterations is clearly depicted in Fig. 16. More explicitly, the second minimum weight of 369.88 lbs was occurred during the early number of iterations within a few minutes.

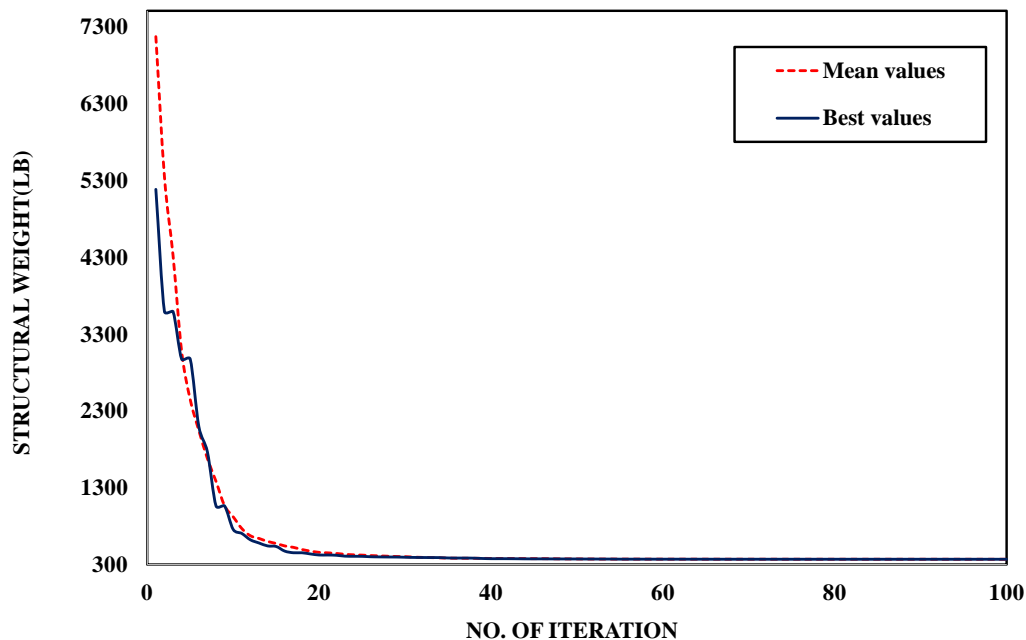


Figure 16 Solution convergency of 72-bar space truss

4.3 Test on Benchmark of a planar 200-bar truss

The geometry of the planar 200-bar truss is shown in Fig. 17. The material has a unit weight of 0.283 lb/in^3 and the modulus of elasticity of $3 \times 10^7 \text{ psi}$ for all members. The maximum allowable stresses of $\pm 10 \text{ ksi}$ are considered as stress constraints for all members of the truss. The load distribution on nodes is listed in Table 5. The members are divided into 29 groups. The element data is presented in Table 6.

Table 5. Load cases of the 200-bar planar truss

Load case	Description	Node
1	1 kip acting in the positive x direction	1, 6, 15, 20, 29, 34, 43, 48, 57, 62 and 71
2	10 kips acting in the negative y direction	1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 24, 26, 28, 29, 30, 31, 32, 33, 34, 36, 38, 40, 42, 43, 44, 45, 46, 47, 48, 50, 52, 54, 56, 57, 58, 59, 60, 61, 62, 64, 66, 68, 70, 71, 72, 73, 74 and 75
3	A combination of the case 1 and case 2 together	

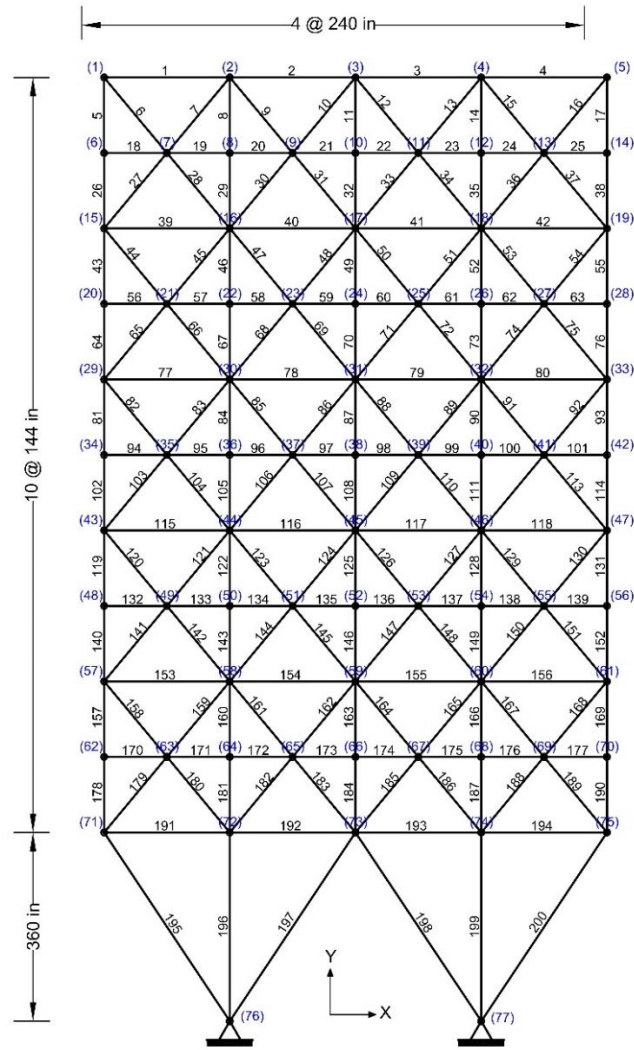


Figure 17 A 200-bar planar truss
 จุฬาลงกรณ์มหาวิทยาลัย
 CHULALONGKORN UNIVERSITY

Table 6. Element data of the 200-bar planar truss

Elem. group	Node		Elem. no.	Node		Elem. no.	Node		Elem. group	Node		Elem. no.		
	1	2		1	2		1	2		1	2			
1	1	2	1	8	15	20	43	40	46	111	53	59	147	
	2	3	2		16	22	46	42	47	114	53	60	148	
	3	4	3		17	24	49	16	29	35	82	55	60	150
	4	5	4		18	26	52	30	35	83	55	61	151	
2	1	6	5		19	28	55	30	37	85	22	57	58	153

	2	8	8	9	21	22	57		31	37	86		58	59	154
	3	10	11		22	23	58		31	39	88		59	60	155
	4	12	14		23	24	59		32	39	89		60	61	156
	5	14	17		24	25	60		32	41	91	23	57	62	157
3	7	8	19		25	26	61		33	41	92		58	64	160
	8	9	20		26	27	62		35	43	103		59	66	163
	9	10	21	10	20	29	64		35	44	104		60	68	166
	10	11	22		22	30	67		37	44	106		61	70	169
	11	12	23		24	31	70		37	45	107	24	63	64	171
	12	13	24		26	32	73		39	45	109		64	65	172
4	6	7	18		28	33	76		39	46	110		65	66	173
	13	14	25	11	15	21	44		41	46	112		66	67	174
	20	21	56		16	21	45		41	47	113		67	68	175
	27	28	63		16	23	47	17	43	44	115		68	69	176
	34	35	94		17	23	48		44	45	116	25	62	71	178
	41	42	101		17	25	50		45	46	117		64	72	181
	48	49	132		18	25	51		46	47	118		66	73	184
	55	56	139		18	27	53	18	43	48	119		68	74	187
	62	63	170		19	27	54		44	50	122		70	75	190
	69	70	177		21	29	65		45	52	125	26	57	63	158
5	6	15	26		21	30	66		46	54	128		58	63	159
	8	16	29		23	30	68		47	56	131		58	65	161
	10	17	32		23	31	69	19	49	50	133		59	65	162
	12	18	35		25	31	71		50	51	134		59	67	164
	14	19	38		25	32	72		51	52	135		60	67	165
6	1	7	6		27	32	74		52	53	136		60	69	167
	2	7	7		27	33	75		53	54	137		61	69	168
	2	9	9	12	29	30	77		54	55	138		63	71	179
	3	9	10		30	31	78	20	48	57	140		63	72	180
	3	11	12		31	32	79		50	58	143		65	72	182
	4	11	13		32	33	80		52	59	146		65	73	183

4	13	15	13	29	34	81		54	60	149		67	73	185
5	13	16		30	36	84		56	61	152		67	74	186
7	15	27		31	38	87	21	43	49	120		69	74	188
7	16	28		32	40	90		44	49	121		69	75	189
9	16	30		33	42	93		44	51	123	27	71	72	191
9	17	31	14	35	36	95		45	51	124		72	73	192
11	17	33		36	37	96		45	53	126		73	74	193
11	18	34		37	38	97		46	53	127		74	75	194
13	18	36		38	39	98		46	55	129	28	71	76	195
13	19	37		39	40	99		47	55	130		73	76	197
7	15	16	39	40	41	100		49	57	141		73	77	198
16	17	40	15	34	43	102		49	58	142		75	77	200
17	18	41		36	44	105		51	58	144	29	72	76	196
18	19	42		38	45	108		51	59	145		74	77	199

Table 7. Comparison of optimized designs of the 200-bar planar truss

Design variables (areas)	Toğan & Daloğlu	Sonmez	Kaveh & Bakshp oori	Kaveh & Zolghadr	Kaveh &	Zakian	Present
	[59]	[47]	[60]	[61]	[62]		
	GA	(ABC-AP)	WEO	CPA	(GWO)	(IGWO)	CGO
A₁	0.3469	0.1039	0.1144	0.1721	1.3363	0.1024	0.1036
A₂	1.0810	0.9463	0.9443	0.9553	2.7525	0.9654	0.9281
A₃	0.1000	0.1037	0.1310	0.1000	0.5923	0.1391	0.1018
A₄	0.1000	0.1126	0.1016	0.1004	0.5258	0.1741	0.1000
A₅	2.1421	1.9520	2.0353	1.9662	5.0281	1.9613	1.9545
A₆	0.3470	0.2930	0.3126	0.3055	0.4945	0.2899	0.2629
A₇	0.1000	0.1064	0.1679	0.1000	1.7505	0.1294	0.1000
A₈	3.5650	3.1249	3.1541	3.1618	3.3725	3.1511	2.8854

A ₉	0.3470	0.1077	0.1003	0.1152	0.2057	0.1251	0.6690
A ₁₀	4.8050	4.1286	4.1005	4.2405	4.3035	4.0627	3.8836
A ₁₁	0.4400	0.4250	0.4350	0.4046	0.7077	0.4131	0.5074
A ₁₂	0.4400	0.1046	0.1148	0.1000	0.1212	0.4043	0.9424
A ₁₃	5.9520	5.4803	5.3823	5.4132	6.6465	5.3357	5.2916
A ₁₄	0.3470	0.1060	0.1607	0.1545	0.1000	0.2632	0.1073
A ₁₅	6.5720	6.4853	6.4152	6.3976	6.9236	6.3226	6.2830
A ₁₆	0.9540	0.5600	0.5629	0.5555	0.8096	0.7972	1.0162
A ₁₇	0.3470	0.1825	0.4010	0.4425	0.1943	0.1791	0.1141
A ₁₈	8.5250	8.0445	7.9735	8.0928	7.9800	8.1286	8.0176
A ₁₉	0.1	0.1026	0.1092	0.1004	0.9110	0.1141	1.9519
A ₂₀	9.3000	9.0334	9.0155	8.9918	10.7262	9.1337	9.2295
A ₂₁	0.9540	0.7844	0.8628	0.8925	1.0542	0.8000	1.3999
A ₂₂	1.7639	0.7506	0.2220	0.2544	0.2809	0.2487	0.1226
A ₂₃	13.3006	11.3057	11.0254	11.1214	15.0000	11.2008	12.0314
A ₂₄	0.3470	0.2208	0.1397	0.1000	0.1310	0.1136	0.1446
A ₂₅	13.3006	12.2730	12.0340	12.3304	15.0000	12.1703	13.2845
A ₂₆	2.1421	1.4055	1.0043	0.0110	0.9469	0.9947	1.3095
A ₂₇	4.8050	5.1600	6.5762	6.4103	7.8886	6.3377	5.0915
A ₂₈	9.3000	9.9930	10.7265	10.5814	15.0000	10.5338	9.2529
A ₂₉	17.1740	14.70144	13.9666	14.1288	13.8801	14.0917	15.6092
Weight(lb)	28,544.0	25,533.79	25674.83	25651.58	33137.452	25771.78	25650.37
Number of analyses	N/A	N/A	N/A	N/A	N/A	N/A	12500
Worst Weight(lb)	N/A	N/A	N/A	N/A	N/A	N/A	26904.99
Mean Weight(lb)	N/A	N/A	26,613.45	25957.15	33,137.452	25,771.78	26413

Table 7 represents the best structural weight of the CGO algorithm in comparison of other metaheuristic algorithms such as genetic algorithm (GA) [59], Artificial Bee Colony algorithm (ABC-AP) [47], water evaporation optimization (WEO) [60], Cyclical Parthenogenesis Algorithm (CPA) [61], GWO algorithm (GWO) [62] and Improved GWO algorithm (IGWO) [62]. The optimum weight comes from CGO (25650.37 lb) is better than GA (28544.0 lb), WEO (25674.83 lb), CPA (25651.58 lb), GWO (33137.78 lb) and IGWO (25771.78 lb) particularly whereas it is more than ABC (25533.79 lb). The optimization history of CGO is described in Fig. 18 with respect to the variation of the best penalized weight and the number of iterations.

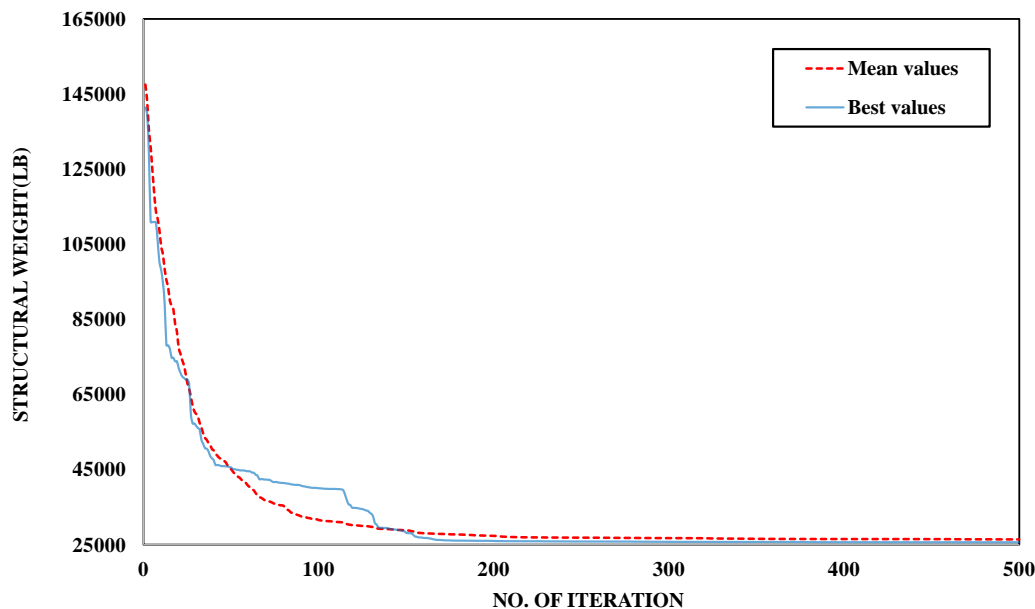


Figure 18 Solution convergency of 200-bar planar truss

The optimal design of the steel truss was successfully performed by the proposed CGO method within 400 analysis iterations with the minimum weight of 25650.37 lbs. The solution (total weight) convergence with the number of analysis (up to 500) iterations is clearly depicted in Fig. 18. In essence, the present CGO approach provides the second most minimum weight solution with the satisfaction of all constraints.

CHAPTER 5

CONCLUSION

5.1 Concluding Remarks

The optimization results are compared to those of some well-known meta-heuristics. The results obtained by the present algorithms is the same or lighter than all other reported results. The proposed method quickly converges to the optimum weight. The proposed method has been illustrated through 10-bar, 72-bar and 200-bar trusses where its accuracy and robustness are evidenced by the good comparisons with other available meta-heuristic algorithms.

The effectiveness and robustness of the proposed algorithm was successfully investigated on three popular sizing optimization benchmarks of steel trusses. The numerical result comparison obtained from these problems reveals that in most cases, the performance of the proposed CGO algorithm is remarkably better than the other optimization techniques available in the literature. It can be concluded that the proposed CGO algorithm is an effective and robust optimizer for solving various optimization problems in engineering fields and can be straightforwardly integrated into in the shape and topology optimization.

5.2 Recommendation for future research

The various application of Chaos Game Optimization algorithm (CGO) should be taken into consideration for upcoming challenges in light of the need to assess this algorithm's aptitude for solving some challenging practical optimization issues. The CGO method can be considered applicable in various technical domains. Moreover, the optimal design of vibration control systems in buildings and other engineering structures is one area where the CGO algorithm can be considered applicable. Additionally, the researchers' differing views on the CGO approach as it is now given should be taken into account when evaluating novel configurations of this algorithm.

REFERENCES

1. Talatahari, S. and M. Azizi, *Chaos Game Optimization: a novel metaheuristic algorithm*. Artificial Intelligence Review, 2021. **54**(2): p. 917-1004.
2. Talatahari, S., et al., *Crystal structure algorithm (CryStAl): a metaheuristic optimization method*. IEEE Access, 2021. **9**: p. 71244-71261.
3. Kirkpatrick, S., C.D. Gelatt Jr, and M.P. Vecchi, *Optimization by simulated annealing*. science, 1983. **220**(4598): p. 671-680.
4. Geem, Z.W., J.H. Kim, and G.V. Loganathan, *A new heuristic optimization algorithm: harmony search*. simulation, 2001. **76**(2): p. 60-68.
5. Holland, J.H., *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. 1992: MIT press.
6. Yang, X.-S. and S. Deb, *Cuckoo search: recent advances and applications*. Neural Computing and applications, 2014. **24**(1): p. 169-174.
7. Kennedy, J. and R. Eberhart. *Particle swarm optimization*. in *Proceedings of ICNN'95-international conference on neural networks*. 1995. IEEE.
8. Dorigo, M., V. Maniezzo, and A. Colomi, *Ant system: optimization by a colony of cooperating agents*. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 1996. **26**(1): p. 29-41.
9. Yang, X.-S. *Flower pollination algorithm for global optimization*. in *International conference on unconventional computing and natural computation*. 2012. Springer.
10. Adeli, H. and O. Kamal, *Efficient optimization of space trusses*. Computers & structures, 1986. **24**(3): p. 501-511.
11. Rajeev, S. and C. Krishnamoorthy, *Discrete optimization of structures using genetic algorithms*. Journal of structural engineering, 1992. **118**(5): p. 1233-1250.
12. Erbaturo, F., et al., *Optimal design of planar and space structures with genetic algorithms*. Computers & Structures, 2000. **75**(2): p. 209-224.
13. Schutte, J. and A. Groenwold, *Sizing design of truss structures using particle swarms*. Structural and Multidisciplinary Optimization, 2003. **25**(4): p. 261-269.
14. Camp, C.V. and B.J. Bichon, *Design of space trusses using ant colony optimization*. Journal of structural engineering, 2004. **130**(5): p. 741-751.
15. Lee, K.S. and Z.W. Geem, *A new structural optimization method based on the harmony search algorithm*. Computers & structures, 2004. **82**(9-10): p. 781-798.
16. Erol, O.K. and I. Eksin, *A new optimization method: big bang-big crunch*. Advances in Engineering Software, 2006. **37**(2): p. 106-111.
17. Li, L.-J., et al., *A heuristic particle swarm optimizer for optimization of pin connected structures*. Computers & structures, 2007. **85**(7-8): p. 340-349.
18. Azad, S.K., O. Hasançebi, and M. Saka, *Guided stochastic search technique for discrete sizing optimization of steel trusses: A design-driven heuristic approach*. Computers & Structures, 2014. **134**: p. 62-74.
19. Murren, P., *Development and implementation of a design-driven harmony search algorithm in steel frame optimization*. 2011, University of Notre Dame.
20. Saka, M.P. and Z.W. Geem, *Mathematical and metaheuristic applications in design optimization of steel frame structures: an extensive review*. Mathematical

- problems in engineering, 2013. **2013**.
21. Kaveh, A. and M.I. Ghazaan, *Optimum design of skeletal structures using PSO-Based algorithms*. Periodica Polytechnica Civil Engineering, 2017. **61**(2): p. 184-195.
 22. Gandomi, A.H., X.-S. Yang, and A.H. Alavi, *Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems*. Engineering with computers, 2013. **29**(1): p. 17-35.
 23. Lamberti, L. and C. Pappalettere, *Metaheuristic design optimization of skeletal structures: a review*. Computational technology reviews, 2011. **4**(1): p. 1-32.
 24. Saka, M., *Optimum design of steel frames using stochastic search techniques based on natural phenomena: a review*. Civil engineering computations: tools and techniques, Saxe-Coburg Publications, Stirlingshire, UK, 2007: p. 105-147.
 25. Colorni, A., M. Dorigo, and V. Maniezzo. *An Investigation of some Properties of an "Ant Algorithm"*. in *Ppsn*. 1992.
 26. Camp, C.V., B.J. Bichon, and S.P. Stovall, *Design of steel frames using ant colony optimization*. Journal of structural engineering, 2005. **131**(3): p. 369-379.
 27. Capriles, P.V., et al., *Rank-based ant colony algorithms for truss weight minimization with discrete variables*. Communications in Numerical Methods in Engineering, 2007. **23**(6): p. 553-575.
 28. Serra, M. and P. Venini, *On some applications of ant colony optimization metaheuristic to plane truss optimization*. Structural and Multidisciplinary Optimization, 2006. **32**(6): p. 499-506.
 29. Hasançebi, O., et al., *Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures*. Computers & Structures, 2009. **87**(5-6): p. 284-302.
 30. Kaveh, A. and S. Shojaee, *Optimal design of skeletal structures using ant colony optimization*. International Journal for Numerical Methods in Engineering, 2007. **70**(5): p. 563-581.
 31. Hasançebi, O., et al., *Comparison of non-deterministic search techniques in the optimum design of real size steel frames*. Computers & structures, 2010. **88**(17-18): p. 1033-1048.
 32. Kaveh, A. and S. Talatahari, *An improved ant colony optimization for the design of planar steel frames*. Engineering Structures, 2010. **32**(3): p. 864-873.
 33. Lee, K.S., et al., *The harmony search heuristic algorithm for discrete structural optimization*. Engineering Optimization, 2005. **37**(7): p. 663-684.
 34. Saka, M., *Optimum geometry design of geodesic domes using harmony search algorithm*. Advances in Structural Engineering, 2007. **10**(6): p. 595-606.
 35. Saka, M. and F. Erdal, *Harmony search based algorithm for the optimum design of grillage systems to LRFD-AISC*. Structural and Multidisciplinary Optimization, 2009. **38**(1): p. 25-41.
 36. Lamberti, L. and C. Pappalettere. *An improved harmony-search algorithm for truss structure optimization*. 2009. PROCEEDINGS OF INTERNATIONAL CONFERENCE ON CIVIL, STRUCTURAL AND
 37. Hasançebi, O., F. Erdal, and M.P. Saka, *Adaptive harmony search method for structural optimization*. Journal of structural engineering, 2010. **136**(4): p. 419-431.
 38. Perez, R.E. and K. Behdinan, *Particle swarm approach for structural design*

- optimization. *Computers & Structures*, 2007. **85**(19-20): p. 1579-1588.
39. Fourie, P. and A.A. Groenwold, *The particle swarm optimization algorithm in size and shape optimization*. *Structural and Multidisciplinary Optimization*, 2002. **23**(4): p. 259-267.
 40. Li, L., Z. Huang, and F. Liu, *A heuristic particle swarm optimization method for truss structures with discrete variables*. *Computers & structures*, 2009. **87**(7-8): p. 435-443.
 41. Kaveh, A. and S. Talatahari, *A particle swarm ant colony optimization for truss structures with discrete variables*. *Journal of Constructional Steel Research*, 2009. **65**(8-9): p. 1558-1568.
 42. Kaveh, A. and S. Talatahari, *Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures*. *Computers & Structures*, 2009. **87**(5-6): p. 267-283.
 43. Kaveh, A. and S. Talatahari, *Size optimization of space trusses using Big Bang–Big Crunch algorithm*. *Computers & structures*, 2009. **87**(17-18): p. 1129-1140.
 44. Camp, C.V., *Design of space trusses using Big Bang–Big Crunch optimization*. *Journal of Structural Engineering*, 2007. **133**(7): p. 999-1008.
 45. Kaveh, A. and S. Talatahari, *Optimal design of Schwedler and ribbed domes via hybrid Big Bang–Big Crunch algorithm*. *Journal of Constructional Steel Research*, 2010. **66**(3): p. 412-419.
 46. Karaboga, D., *An idea based on honey bee swarm for numerical optimization*. 2005, Technical report-tr06, Erciyes university, engineering faculty, computer
 47. Sonmez, M., *Artificial Bee Colony algorithm for optimization of truss structures*. *Applied Soft Computing*, 2011. **11**(2): p. 2406-2418.
 48. Rao, R.V., V.J. Savsani, and D. Vakharia, *Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems*. *Computer-aided design*, 2011. **43**(3): p. 303-315.
 49. Rao, R.V., V.J. Savsani, and D. Vakharia, *Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems*. *Information sciences*, 2012. **183**(1): p. 1-15.
 50. Toğan, V., *Design of planar steel frames using teaching–learning based optimization*. *Engineering Structures*, 2012. **34**: p. 225-232.
 51. Salimi, H., *Stochastic fractal search: a powerful metaheuristic algorithm*. *Knowledge-Based Systems*, 2015. **75**: p. 1-18.
 52. Kaedi, M., *Fractal-based algorithm: a new metaheuristic method for continuous optimization*. *Int J Artif Intell*, 2017. **15**(1): p. 76-92.
 53. Nakib, A., et al., *Deterministic metaheuristic based on fractal decomposition for large-scale optimization*. *Applied Soft Computing*, 2017. **61**: p. 468-485.
 54. Rodrigues, E.O., et al., *Fractal triangular search: a metaheuristic for image content search*. *IET Image Processing*, 2018. **12**(8): p. 1475-1484.
 55. Talatahari, S. and M. Azizi, *Optimization of constrained mathematical and engineering design problems using chaos game optimization*. *Computers & Industrial Engineering*, 2020. **145**: p. 106560.
 56. Degertekin, S., *Improved harmony search algorithms for sizing optimization of truss structures*. *Computers & Structures*, 2012. **92**: p. 229-241.
 57. Degertekin, S. and M. Hayalioglu, *Sizing truss structures using teaching-*

- learning-based optimization*. Computers & Structures, 2013. **119**: p. 177-188.
58. Jalili, S. and Y. Hosseinzadeh, *A cultural algorithm for optimal design of truss structures*. Latin American Journal of Solids and Structures, 2015. **12**: p. 1721-1747.
59. Toğan, V. and A.T. Daloğlu, *An improved genetic algorithm with initial population strategy and self-adaptive member grouping*. Computers & Structures, 2008. **86**(11-12): p. 1204-1218.
60. Kaveh, A. and T. Bakhshpoori, *A new metaheuristic for continuous structural optimization: water evaporation optimization*. Structural and Multidisciplinary Optimization, 2016. **54**(1): p. 23-43.
61. Kaveh, A. and A. Zolghadr, *Cyclical parthenogenesis algorithm: A new metaheuristic algorithm*. 2017.
62. Kaveh, A. and P. Zakian, *Improved GWO algorithm for optimal design of truss structures*. Engineering with Computers, 2018. **34**(4): p. 685-707.





จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

VITA

NAME Ms. Eain Kyi Thar

DATE OF BIRTH 10 May 1993

PLACE OF BIRTH Pyinmana, Myanmar

**INSTITUTIONS
ATTENDED** West Yangon Technological University

HOME ADDRESS No.1, 60th street, 7 Ward, Hlaing Township, Yangon,
Myanmar.



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY