การออกแบบอย่างเหมาะสมสำหรับโครงข้อหมุนเหล็กด้วยวิธีความฉลาดแบบกลุ่ม
Optimum Design of Steel Truss by Particle Swarm Optimization

เพทาย วงศ์ศิลากิจ[1] และ รศ. ดร. วัฒนชัย สมิทธากร[2]

[1,2]ภาควิศวกรรมศาสตร์โยธา คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย จ. กรุงเทพฯ

บทคัดย่อ

โครงงานนี้จัดทำเพื่อนำเสนอการออกแบบโครงสร้างโครงข้อหมุนอย่างเหมาะสมด้วยภาษาไพทอน โดยจุดประสงค์หลักคือการพัฒนาโปรแกรมออกแบบหน้าตัดโครงเหล็กที่สามารถใช้งานได้จริงและนำไปพัฒนาเพื่อใช้ในโครงสร้างประเภทอื่นต่อได้ โปรแกรมถูกพัฒนาด้วยการทำงานร่วมกันของโปรแกรมวิเคราะห์โครงสร้าง และ อัลกอริทึมปัญญาประดิษฐ์, Particle Swarm Optimization(PSO), เป็นอัลกอริทึมที่จำลองธรรมชาติของ การออกหาอาหารของฝูงนก มีจุดเด่นคือการประมวลผลโดยใช้เวกเตอร์ในการคาดการณ์คำตอบ ซึ่งสอดคล้องกับการออกแบบโครงสร้างอย่าง เหมาะสม ที่จำเป็นต้องออกแบบซ้ำเพื่อหาหน้าตัดที่ประหยัดที่สุด โดยกระบวนการประมวลผลต้องใช้เวลานาน จึงนำอัลกอริทึม PSO เข้ามาช่วย เพิ่มประสิทธิภาพในการออกแบบ ทั้งเรื่องเวลาและความแม่นยำ นอกจากนี้ ภาษาไพทอนมีคุณสมบัติที่ง่ายต่อการทำความเข้าใจและนำไปพัฒนา ต่อ เป็นภาษาที่กระชับ สามารถปรับแต่งได้โดยง่ายตามวัตถุประสงค์ที่ต้องการ ในโครงงานนี้จะเขียนโปรแกรมออกแบบด้วยภาษาไพทอนด้วย อัลกอริทึม PSO และนำโครงสร้างที่ออกแบบได้มาเปรียบเทียบกับความประหยัดกับโครงเหล็กที่ได้ออกแบบไว้ด้วยวิธีในอดีตเพื่อหาประสิทธิภาพ ของโปรแกรมออกแบบ

คำสำคัญ: การออกแบบอย่างเหมาะสม, โครงข้อหมุนเหล็ก, วิธีความฉลาดแบบกลุ่ม

Abstract

In this study, a structural optimization program is developed using Python programming language. The main purpose is to apply to real-world steel truss problems and to develop for other structural types. The program development uses a structural analysis program paired with an AI algorithm, Particle Swarm Optimization (PSO) designed to imitate a flock of birds searching for food. PSO uses vectors to predict optimum answers, therefore, it is similar to the traditional optimizing method, which repeats the calculations to solve for optimum design. As a result, PSO is adopted to improve the efficiency and accuracy of the calculation. Moreover, Python language is simple, customizable, and easy to maintain. In conclusion, the program uses Python programming language with PSO algorithm to optimize steel truss structures, then the results are compared to the traditional optimization to evaluate the efficiency.

Keywords: Optimum design, Steel truss, Particle Swarm Optimization

1. บทนำ

การออกแบบอย่างเหมาะสมถือเป็นปัจจัยสำคัญในขั้นตอนการก่อสร้างสิ่งปลูกสร้าง เนื่องจากวิศวกร-โครงสร้างมีหน้าที่ในการค้นหารูปแบบการสร้างสิ่งปลูกสร้างที่ประหยัดค่ามากที่สุด ในขณะที่ต้องคำนึงถึงความปลอดภัยและความเหมาะสมของโครงสร้างอีกด้วย ในศาสตร์ของวิศวกรรมโยธาจึงมีการค้นหาวิธีการมากมายในการสร้างเครื่องมือที่สามารถออกแบบโครงสร้างที่มีความแข็งแรงและประหยัดค่าใช้จ่ายในเวลาเดียวกัน เพื่อที่จะสร้างโครงสร้างที่ปลอดภัยให้ผู้ใช้สามารถใช้งานได้อย่างเชื่อมั่น สามารถประหยัดเวลาในการออกแบบ และสามารถช่วยลดปริมาณทรัพยากรและต้นทุนที่ใช้ในการก่อสร้าง ซึ่งเป็นสิ่งที่สำคัญมากในโลกยุคสมัยใหม่ การสร้างเครื่องมือในการออกแบบตามจุดประสงค์นี้ จึงเป็นการสร้างทางเลือกที่วิศวกรโยธาสามารถนำไปใช้งาน

ได้ ณ ปัจจุบัน วิศวกรโยธาได้สร้างโปรแกรมคอมพิวเตอร์ที่ใช้ในการคำนวณออกแบบโครงสร้างที่เป็นที่นิยม เช่น Extended3D Analysis of building System (ETABS) และ Structural Analysis Program2000 (SAP2000) เป็นต้น โดยทั้งสองโปรแกรมถูกเขียนด้วยภาษาคอมพิวเตอร์ที่เป็นกรรมสิทธิ์เฉพาะของบริษัท ไม่สามารถให้บุคคลภายนอกนำมาพัฒนาต่อได้อย่างอิสระ อีกทั้งโปรแกรมออกแบบโครงสร้างที่นิยมในงานวิศวกรรมโยธานั้น ทางตัววิศวกรเองต้องใช้วิธีการดังเดิมอย่าง Trial and Error เพื่อค้นหาคำตอบของชิ้นส่วนของโครงสร้างที่คุ้มค่าและปลอดภัยที่สุด จึงเป็นที่มาของการพัฒนาโปรแกรมในโครงงานนนี้

โดยการพัฒนาโปรแกรมในโครงงานนี้เป็นการใช้ภาษาคอมพิวเตอร์ไพทอนในการเขียนโปรแกรมออกแบบโครงข้อหมุนเหล็กอย่างเหมาะสมตามมาตรฐานความปลอดภัย AISC 360-16 วิธีการออกแบบ LRFD [1] ด้วยหน้าตัดเหล็กมาตรฐาน มอก. ของประเทศไทย ด้วยเหตุผลที่ภาษานี้ง่ายต่อการทำความเข้าใจทั้งผู้ใช้งาน และผู้พัฒนาโปรแกรม โดยในส่วนของการวิเคราะห์โครงสร้างนั้น จะใช้โปรแกรมคำนวณแรงในโครงสร้าง Project-Python3D [2] ที่ถูกพัฒนามาในภาษาไพทอนเช่นกัน ทางผู้วิจัยจึงเลือกใช้โปรแกรมนี้ในการพัฒนาต่อให้สามารถออกแบบอย่างเหมาะสมได้ในตัวโปรแกรมเดียว

2. งานวิจัยในอดีตและทฤษฎีที่เกี่ยวข้อง

ทฤษฎีการออกแบบเพื่อหาผลลัพธ์ที่เกี่ยวข้องกับ ขนาดหน้าตัด, รูปร่างหน้าตัด, วัสดุ และองค์ประกอบอื่น ๆ ที่ไม่เกี่ยวข้องกับรูปร่างหรือการเรียงตัวของโครงสร้างโดยรวม ในกรณีที่มีโจทย์เป็นรูปแบบการเรียงตัวของโครงสร้างและตำแหน่งรับน้ำหนัก ซึ่งเป็นการออกแบบที่พิจารณาแรงภายในของโครงสร้างและเปลี่ยนแปลงหน้าตัดทั้งในด้านของ ความกว้าง, ความลึก หรือรูปร่างของหน้าตัด เพื่อหาผลลัพธ์คุณสมบัติหน้าตัดที่ประหยัดที่สุดในการออกแบบโครงสร้างนั้น

การหาคำตอบของการออกแบบอย่างเหมาะสมมีความซับซ้อนมากขึ้นตามยุคสมัย เนื่องด้วยการออกแบบโครงสร้างที่ซับซ้อนขึ้น และเทคโนโลยีการก่อสร้างที่พัฒนาขึ้น ทำให้ปัญหาการออกแบบมีพฤติกรรมแบบ Non-Linear และมีตัวแปรจำนวนมาก จึงมีการคิดค้นการแก้ปัญหาแบบฮิวริสติกและเมตะฮิวริสติกขึ้นมา

วิธีการฮิวริสติก คือ การหาคำตอบแบบเจาะจงเฉพาะคำถาม (problem-specific) โดยไม่การันตีคำตอบที่ถูกต้องที่สุด แต่มีประสิทธิภาพพอที่จะหาคำตอบที่ดีพอสำหรับปัญหานั้น โดยในอีกวิธีการหนึ่งคือ มีวิธีการเมตะฮิวริสติกซึ่งเป็นวิธีการขั้นสูงกว่าในการออกแบบอย่างเหมาะสมที่สามารถใช้ในการแก้ปัญหาได้หลากหลายกว่า โดยวิธีการเมตะฮิวริสติกเป็นวิธีการแก้ปัญหาทั่วไปที่ออกแบบมาเพื่อหาคำตอบที่ใกล้เคียงค่าที่ถูกต้องที่สุดอย่างมีประสิทธิภาพแต่ไม่การันตีค่าที่ถูกต้องที่สุดอย่างแท้จริง วิธีการนี้ส่วนใหญ่จะเป็นการเลียนแบบธรรมชาติไม่ว่าจะเป็น การเลียนแบบวิวัฒนาการ, การเลียนแบบฝูงสัตว์ และการหลอมละลายของโลหะ วิธีการเมตะฮิวริสติกค้นหาคำตอบด้วยการคำนวณซ้ำและพิจารณาค่าที่ดีที่สุดในแต่ละการทำซ้ำ จนได้คำตอบที่พึงพอใจ

การออกแบบอย่างเหมาะสมของโครงสร้างนั้นได้นำความรู้ทางคณิตศาสตร์ และพฤติกรรมของธรรมชาติมากมายมาประยุกต์ใช้เข้าด้วยกัน ไม่ว่าจะเป็นการลอกเลียนแบบพฤติกรรมสัตว์ หรือ การสังเกตกฎของธรรมชาติต่าง ๆ โดยในการออกแบบโครงสร้างอย่างเหมาะสมได้เคยมีการใช้อัลกอริทึมมากมายในการออกแบบ เช่น Genetic Algorithm, GA ซึ่งเป็นการเลียนแบบวิวัฒนาการของสิ่งมีชีวิตผ่านการคัดเลือกทางธรรมชาติ และนำมาปรับใช้กับการออกแบบด้วยการคัดเลือกประชากรที่ให้ผลลัพธ์ที่ดีที่สุดและทำการผสมโครโมโซมประชากรที่ดีเข้ากับประชากรอื่น, Firefly Alogrithm, FA ที่เป็นการเลียนแบบพฤติกรรมของหิ่งห้อยที่จะมีการดึงดูดโดยแสงกระพริบของหิ่งห้อยตัวอื่น ค่าความสว่างของแสงและระยะห่างระหว่างหิ่งห้อยจะเป็นตัวแปรในการหาคุณภาพของประชากร, Harmony Search Algorithm, การเลียนแบบการด้นสดขณะเล่นดนตรีของวงดนตรีแจ๊ส โดยแก้ปัญหาทางโครงสร้างจากการหาทำนองที่กลมกลืนกันของเพลงแต่ละท่อน และ Ant Colony Optimization, ACO ซึ่งเป็นการเลียนแบบพฤติกรรมของการหาอาหารของอาณานิคมมด ซึ่งมดจะทิ้งร่องรอยฟีโรโมนเพื่อให้มดตัวอื่นสามารถเดินตามเส้นทางตนเองไปหาคำตอบที่เป็นผลลัพธ์ที่ดีที่สุดของปัญหาได้

3 วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO)

วิธีความฉลาดแบบกลุ่ม เป็นวิธีการแบบเมตะฮิวริสติกที่ถูกสร้างโดย Eberhert and Kennedy ในปี ค.ศ. 1995 [3] โดยได้รับแรงบันดาลใจมาจากการบินเป็นฝูงของนก และการว่ายเป็นฝูงของปลา โดยพวกเขาสังเกตพฤติกรรมของปลาและนกในเรื่องของการเดินทางเป็นฝูงแบบที่มีความสัมพันธ์ระหว่างประชากรในฝูง และนำมาปรับใช้ในการแก้ปัญหาการออกแบบอย่างเหมาะสม

R.E. Perez and K. Behdinan [4ได้นำวิธีความฉลาดแบบกลุ่มมาพัฒนา ปรับค่าพารามิเตอร์ต่าง ๆ เพื่อค้นหาความแตกต่างของผลลัพธ์และเทียบกับวิธีการออกแบบเหมาะสมแบบดั้งเดิม โดยทดสอบกับโครงสร้างโครงข้อหมุนเหล็ก โดยจากการทดลองพบว่า พารามิเตอร์แต่ละตัวให้ค่าผลลัพธ์ที่แตกต่างกันทำให้เส้นทางการเคลื่อนตัวของอนุภาคเปลี่ยนไปและจำนวนครั้งของการทำซ้ำของโปรแกรมมีผลกับคำตอบที่ดีที่ที่สุด

3.1 หลักการของวิธีความฉลาดแบบกลุ่ม

วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO) [5] เป็นวิธีการออกแบบอย่างเหมาะสมที่เป็นแบบ stochastic ที่มีพื้นฐานมาจากการเคลื่อนตัวแบบเป็นงฝูง ลอกเลียนแบบพฤติกรรมของฝูงนกและฝูงปลา

ประชากรแต่ละตัวในฝูงจะมีการเคลื่อนตัวที่เปลี่ยนแปลงรูปแบบไปเรื่อย ๆ ตามประสบการณ์ของประชากรตัวนั้น โดยมีพื้นฐานการสร้างระบบจำลองฝูงประชากรเหล่านี้ให้มีประสิทธิภาพ โดยมีหลักการดังนี้

1. ความใกล้ชิด: กลุ่มประชากรที่สร้างขึ้นมาต้องมีความสามารถในการกระจายตัวในพื้นที่ค้นหาที่เหมาะสมในระยะเวลาที่ใช้คำนวณ
2. คุณภาพ: กลุ่มประชากรควรมีความสามารถในการสัมผัสถึงความเปลี่ยนแปลงทางคุณภาพของสภาพแวดล้อมและตอบสนองอย่างถูกต้อง
3. ความหลากหลายในการตอบสนอง: กลุ่มประชากรไม่ควรจำกัดเส้นทางของตนในการค้นหาคำตอบในพื้นที่ที่เล็กเกินไป
4. ความเสถียร: กลุ่มประชากรไม่ควรเปลี่ยนความคิดของตนในทุก ๆ สภาพแวดล้อมที่เปลี่ยนไป
5. การปรับตัว: กลุ่มประชากรควรเปลี่ยนวิธีการคิดเมื่อการเปลี่ยนแปลงนั้นมีคุณค่ามากพอ

3.2 ขั้นตอนการทำงานของวิธีความฉลาดแบบกลุ่ม

ในขั้นตอนแรกต้องทำการสมมติให้ตัวแปรออกแบบ x อยู่ในตำแหน่งเริ่มต้นในพื้นที่ค้นหา และมี v เป็นเวกเตอร์ความเร็วในการเคลื่อนที่ ระยะทางระหว่างตำแหน่งของอนุภาคกับเป้าหมายจะถูกใช้ในการวัดคุณภาพของผลลัพธ์ของอนุภาคแต่ละตัว โดยในกรณีการออกแบบอย่างเหมาะสมกล่าวคือ มวลรวมโครงสร้างยิ่งมีค่าน้อยแสดงว่า ผลลัพธ์มีค่าที่ดี ในทางตรงข้ามกันยิ่งมวลรวมมากผลลัพธ์ยิ่งมีค่าแย่ โดยให้อนุภาคแต่ละตัวมีความสามารถในการจดจำตำแหน่งที่ดีที่สุดของมันได้ เราเรียกมันว่า personal best, $pbest$ โดยตัวแปรความเร็วในการเคลื่อนที่จะเปลี่ยนแปลงไปเรื่อย ๆ และจะมีตัวแปร r_1 ซึ่งจะให้ค่าเป็นเลขสุ่มระหว่าง 0-1 และ c1 เป็นค่าคงตัวน้ำหนักของเส้นทางที่ตามความทรงจำของอนุภาค การเปลี่ยนแปลงของตัวแปรความเร็วจากความทรงจำส่วนตัวนี้ว่าเรียกว่า “Cognitive Velocity” โดยสามารถอธิบายออกมาเป็นสมการได้ดังนี้

$$cognitive\ velocity = c_1 r_1 (pbest - x) \quad (1)$$

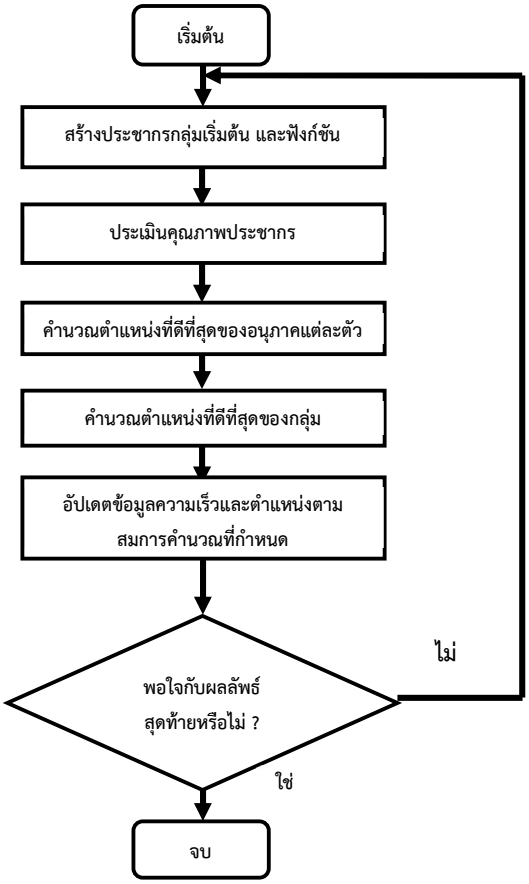
และถ้าหากเพิ่มสมมติฐานเข้าไปว่า อนุภาคแต่ละตัวสามารถสื่อสารกันได้ โดยอนุภาคแต่ละตัวสามารถจำตำแหน่งที่ดีที่สุดของสมาชิกในฝูงได้ และเรียกมันว่า global best, $gbest$ และให้ตัวแปร c_2 เป็นค่าคงตัวน้ำหนักของเส้นทางตามความทรงจำของฝูง และให้ r_2 เป็นเลขที่สุ่มระหว่าง 0-1 แยกกับ r_1 การเปลี่ยนแปลงของความเร็วจากความทรงจำกลุ่มนี้เรียกว่า “Social Velocity” เขียนเป็นสมการได้ว่า

$$social\ velocity = c_2 r_2 (gbest - x) \quad (2)$$

หากทำการรวมอิทธิพลระหว่างความทรงจำของอนุภาคแต่ละตัวกับความทรงจำแบบกลุ่ม เราสามารถสร้างสมการความเร็วที่เปลี่ยนไปในการเดินทางของอนุภาคได้ว่า

$$v = c_1 r_1 (pbest - x) + c_2 r_2 (gbest - x) \quad (3)$$

หากให้สมมติฐานว่าอนุภาคแต่ละตัวไม่มีมวลและไม่มีปริมาตร และใช้อิทธิพลจากตำแหน่งและความเร็วในการเคลื่อนที่เท่านั้น จะเกิดเป็นอัลกอริทึม Particle Swarm Optimization



รูปที่ 1 ขั้นตอนการออกแบบวิธี PSO

3.3 การคำนวณของวิธีความฉลาดแบบกลุ่ม

ให้ n คือขนาดของกลุ่มอนุภาค (swarm size) โดยอนุภาคแต่ละตัวจะมีมิติของตัวแปร คือ D ได้ว่า

ตำแหน่งของอนุภาค คือ

$$X_i = \left(x_{i1}, x_{i2}, \ldots, x_{id}, \ldots, x_{iD} \right) \quad (4)$$

เวกเตอร์ความเร็ว คือ

$$V_i = \left(v_{i1}, v_{i2}, \ldots, v_{id}, \ldots, v_{iD} \right) \quad (5)$$

ตำแหน่งที่ดีที่สุดส่วนตัว คือ

$$P_i = \left(p_{i1}, p_{i2}, \ldots, p_{id}, \ldots, p_{iD} \right) \quad (6)$$

ตำแหน่งที่ดีของกลุ่มอนุภาค คือ

$$P_i = \left(p_{g1}, p_{g2}, \ldots, p_{gd}, \ldots, p_{gD} \right) \quad (7)$$

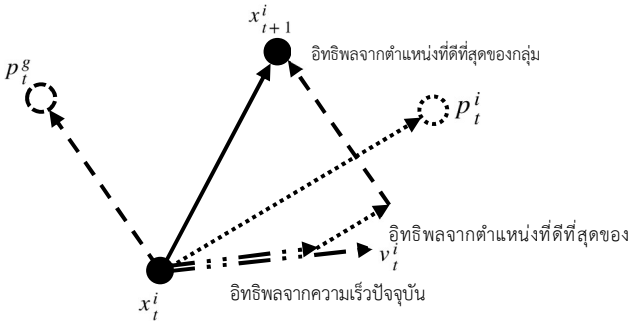
จากนั้นสามารถอัปเดตข้อมูลคุณภาพของอนุภาคจากการดำเนินการผ่านฟังก์ชันจุดประสงค์ได้ ดังนี้

$$p^d_{i.t+1} = \begin{cases} x^d_{i,t+1}, & if\ (X_{i,t+1}) < f(P_{i,t}) \\ p^d_{i,t}, & otherwise \end{cases} \quad (8)$$

ซึ่งตำแหน่งที่ดีที่สุดของกลุ่มอนุภาคือตำแหน่งที่ดีที่สุดเมื่อเทียบกับตำแหน่งที่ดีที่สุดของอนุภาคทุกตัวจาก สมการที่ (8) ในการอัปเดตความเร็วและตำแหน่งในการคำนวณแต่ละครั้ง เพื่อเพิ่มประสิทธิภาพของวิธีความฉลาดแบบกลุ่ม สามารถใส่ตัวแปรน้ำหนักของเส้นทางเพิ่มไปอีกหนึ่งตัวแปรได้ คือ (inertia weight, w) ซึ่งจะใช้ถ่วงน้ำหนักของเวกเตอร์ความเร็วในการคำนวณครั้งปัจจุบัน เพื่อใช้คำนวณเวกเตอร์ความเร็วที่เปลี่ยนแปลงในรอบการดำเนินการครั้งต่อไป

$$v^d_{i,t+1} = w \cdot v^d_{i,t} + c_1 r_1 \left(p^d_{i,t} - x^d_{i,t} \right) + c_2 r_2 \left(p^d_{g,t} - x^d_{i,t} \right) \quad (9)$$

จากสมการทั้งหมดสามารถสร้างรูปภาพที่อธิบายเส้นทางการเดินทางของอนุภาคใน PSO ได้ดังรูปที่ 2



รูปที่ 2 วิธีการเคลื่อนตัวของอนุภาค

การเคลื่อนตัวของอนุภาคในวิธีความฉลาดแบบกลุ่มจะเคลื่อนตัวโดยมีอิทธิพลมาจาก 3 ตัวแปรสำคัญคือ ตำแหน่งที่ดีที่สุดของอนุภาค (cognitive factor), ตำแหน่งที่ดีที่สุดของกลุ่มอนุภาค (social factor) และ ความเร็วปัจจุบันของอนุภาคนั้น ๆ โดยถ้าหากเราทำการคำนวณซ้ำผ่านฟังก์ชันวัตถุประสงค์ในอนุภาคทุกตัวในกลุ่ม จะทำให้เห็นภาพของการเคลื่อนตัวของอนุภาคที่จะมุ่งหน้าเข้าสู่เป้าหมายที่เราต้องการ กล่าวคือ เป็นการคำนวณการออกแบบอย่างเหมาะสมผ่านการแทนค่าตัวแปรและคำนวณอย่างมีหลักการและไม่ใช่การสุ่มอย่างแท้จริง

3.4 การออกแบบอย่างปลอดภัยตามมาตรฐาน

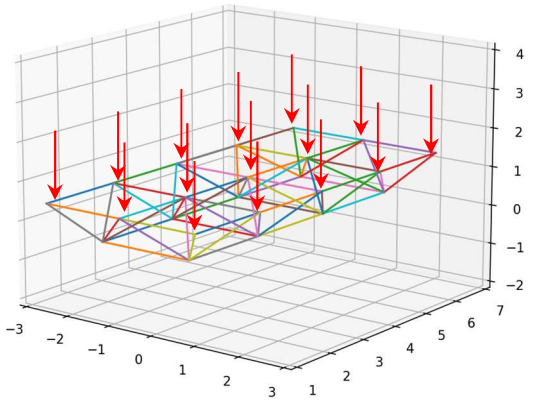
การเรียกใช้โปรแกรมที่ใช้อัลกอริทึม PSO จะเป็นการคำนวณแรงในโครงสร้างและออกแบบหน้าตัดอย่างต่อเนื่อง ทำให้ถ้าหากการสุ่มหน้าตัดของโครงข้อหมุนเหล็กได้หน้าตัดที่ไม่ปลอดภัยต้องทำการใส่ฟังก์ชันลงโทษ โดยในการคำนวณแรงภายในโครงสร้าง ถ้าหากแรงภายในมีค่ามากกว่าความต้านทานระบุปรับแก้แล้ว การคำนวณน้ำหนักของโครงสร้างที่ตำแหน่งอนุภาคนั้น จะถูกลงโทษด้วยการเพิ่มน้ำหนักโครงสร้างรวมที่มากขึ้นตามปริมาณแรงภายในที่มีค่าเกินมาจากความต้านทานระบุปรับแก้ โดยในโปรแกรมออกแบบอย่างเหมาะสมได้มีการสร้างคำสั่งในการตรวจสอบความปลอดภัยของชิ้นส่วนของโครงสร้างในการคำนวณซ้ำแต่ละรอบตามมาตรฐานการออกแบบ AISC 360-16 LRFD นอกจากนี้โปรแกรมออกแบบยังมีข้อมูลคุณสมบัติหน้าตัดเหล็กท่อกลมตามมาตรฐาน มอก. ทุกขนาดเพื่อใช้ในการออกแบบโครงสร้างโครงข้อหมุนเหล็ก โดยให้ตัวแปรที่ใช้ที่ใช้ออกแบบดังนี้

Modulus of Elasticity	=	200	GPa
ความหนาแน่นเหล็ก	=	7860	กิโลกรัม/ลูกบาศก์เมตร
หน่วยแรงคราก (F_y)	=	235	MPa
กำลังดึงประลัย (F_u)	=	360	MPa

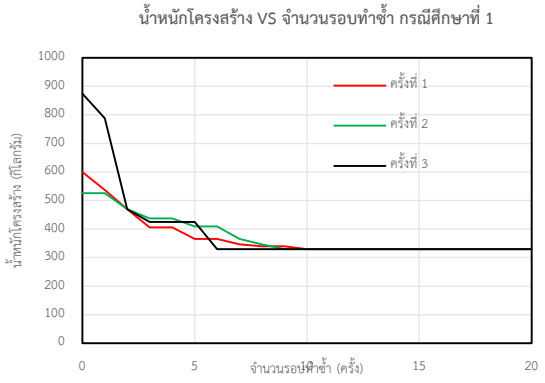
4. ผลการดำเนินงาน

4.1 กรณีตัวอย่างที่ 1

ตัวอย่างการออกแบบนี้จะออกแบบโครงสร้าง space truss 3 มิติมีระยะ span 6 เมตร ความสูง 1 เมตร ระยะห่าง support 2 เมตรที่โครงสร้างด้านล่างทั้ง 4 มุม โดยมีรูปร่างโครงสร้างดังรูป 3 โดยมีน้ำหนักบรรทุกปรับแก้ (P_u) ที่พิกัดจุดของโครงสร้างด้านบนจุดละ 20,000 นิวตัน เป็น Point Load และให้ตัวแปรออกแบบ 3หน้าตัดคือ โครงสร้างโครงข้อหมุนเหล็กด้านบน (Top Chord), โครงข้อหมุนด้านล่าง (Bottom Chord) และโครงข้อหมุนแนวทแยง (Diagonal Chord) โดยในกรณีตัวอย่างนี้เป็นตัวอย่างเพื่อทดสอบความสามารถในการออกแบบอย่างเหมาะสมในโครงสร้างที่ตัวแปรหน้าตัดน้อยของโปรงแกรม



รูปที่ 3 กรณีตัวอย่างที่ 1



รูปที่ 4 น้ำหนักของโครงสร้างตามจำนวนครั้งที่ทำซ้ำ

จากการคำนวณโปรแกรมออกแบบอย่างเหมาะสม 3 ครั้ง พบว่าได้ผลลัพธ์เป็นคำตอบเป็นโครงสร้างหน้าตัดเดียวกัน คือ โครงสร้างมีน้ำหนักรวม 330.096 กิโลกรัม และได้หน้าตัดโครงสร้างดังนี้

Top Chord = เหล็กท่อกลม 48.6 มม. หนา 2.3 มิลลิเมตร

Bottom Chord = เหล็กท่อกลม 34.0 มม. หนา 2.3 มิลลิเมตร

Diagonal Chord = เหล็กท่อกลม 48.6 มม. หนา 3.2 มิลลิเมตร

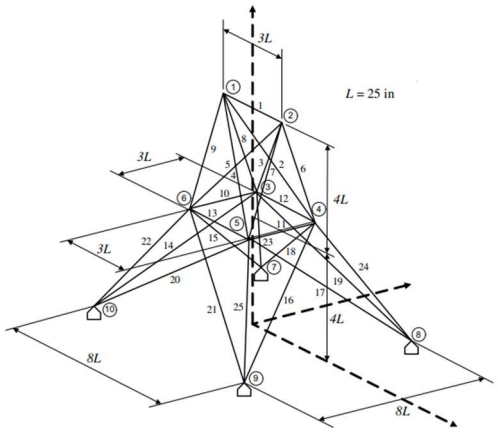
โดยเมื่อเทียบกับการออกแบบด้วยวิธีการ Trial and Error ซึ่งการออกแบบได้คำตอบเหมือนกัน ในโครงสร้างที่ตัวแปรการออกแบบจำนวนน้อย (3 ตัวแปร) จะเห็นได้ว่าโปรแกรมออกแบบอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่มสามารถออกแบบโครงสร้างได้โดยผู้ใช้ไม่ต้องทำการแทนค่าหาคำตอบด้วยตนเองในโครงสร้างที่ตัวแปรหน้าตัดไม่มาก

4.2 กรณีศึกษาที่ 2

โครงสร้าง 25-bar Truss Tower โดย Schmit and Fleury [6] ใช้ในการตรวจสอบประสิทธิภาพการทำงานของโปรแกรม เป็นตัวอย่างโครงสร้างที่ใช้อย่างแพร่หลายในการทดสอบระบบออกแบบอย่างเหมาะสมด้วยวิธีคำนวณต่าง ๆ เนื่องจากเป็นโครงสร้างที่มีความซับซ้อนสูง โดยมีตัวแปรการออกแบบหน้าตัด 8 ตัวแปร และการหาหน้าตัดการออกแบบที่ประหยัดที่สุดเป็นไปได้ยาก จึงเหมาะสมกับการทดสอบความแตกต่างของผลลัพธ์ของการออกแบบจากค่าพารามิเตอร์เริ่มต้นที่แตกต่างกันของวิธีความฉลาดแบบกลุ่ม

ตารางที่ 1 ตัวแปรหน้าตัดการออกแบบ

กลุ่มหน้าตัด	ชิ้นส่วนโครงข้อหมุนเหล็ก
A1	1
A2	2-5
A3	6-9
A4	10, 11
A5	12, 13
A6	14-17
A7	18-21
A8	22-25



รูปที่ 5 25 bar-truss-tower

ตารางที่ 2 แรงกระทำโครงสร้าง

Node	F_x , (นิวตัน)	Fx , (นิวตัน)	Fz , (นิวตัน)
1	4448.22	-44482.22	-44482.22
2	0	-44482.22	-44482.22
3	2224.11	0	0
6	2668.93	0	0

โดยจากการโปรแกรมออกแบบซ้ำจำนวน 200 ครั้งด้วยตัวแปร c_1, c_2, และ w ที่แตกต่างกันโดยมีตัวแปรคงที่คือ n = 40 และ จำนวนครั้งในการทำซ้ำ 100 ครั้ง พบว่า มีผลลัพธ์การออกแบบที่ได้น้ำหนักโครงสร้างต่ำที่สุด 436.63 กิโลกรัม โดยมีหน้าตัดของโครงข้อหมุนเหล็กดังนี้

ตารางที่ 3 หน้าตัดออกแบบ

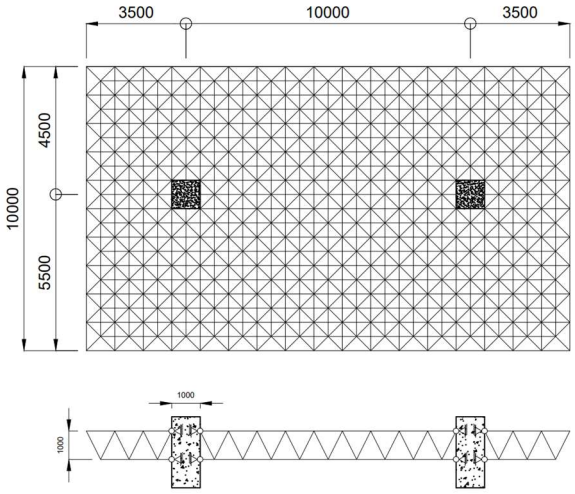
กลุ่มหน้าตัด	เหล็กท่อกลม (เส้นผ่านศูนย์กลาง x ความหนา)(มม.)
A1	21.7 x 2.0
A2	60.5 x 3.2
A3	76.3 x 3.2
A4	21.7 x 2.0
A5	21.7 x 2.0
A6	76.3 x 3.2
A7	76.3 x 3.2
A8	89.1 x 3.2

เนื่องด้วยกรณีตัวอย่างนี้มีจำนวนตัวแปรออกแบบหน้าตัดจำนวนมาก จึงต้องทำการออกแบบซ้ำหลายรอบ เพื่อหาคำตอบที่ดีที่สุด เนื่องจากโปรแกรมออกแบบมีผลจากการสุ่มข้อมูลเริ่มต้นสูง แต่ถ้าหากออกแบบซ้ำหลายรอบผลลัพธ์ที่ได้จะมีคุณภาพมาก

4.3 กรณีศึกษาที่ 3

กรณีศึกษานี้เป็นการออกแบบโครงสร้างในการทำงานจริง คือ โครงสร้างโครงหลังคา Space Truss โดยโครงสร้างเป็นโครงสร้างโครงข้อหมุนเหล็กขนาดกว้าง 17 เมตร ยาว 10 เมตร โดยมีเสาอยู่ตรงกลาง 2 ต้น ขนาด 1 เมตร x 1 เมตร และมี span สูงสุด 9 เมตรระหว่างเสาสองต้น

โดยพิกัดจุดของโครงข้อหมุนเหล็กอยู่ห่างกัน 1 เมตรทั้งแนวตั้งและแนวนอน มีน้ำหนักบรรทุกจร (Live Load) ก่อนคูณค่าปรับแก้ 750 นิวตัน/ตารางเมตร โดยใส่เป็น Point Load ที่คูณค่าปรับแก้ 1.6 ที่ตำแหน่งพิกัดจุดของโครงข้อหมุนด้านบน ดังรูป



รูปที่ 6 กรณีศึกษาที่ 3

โดย support ของโครงสร้างนี้จะอยู่ที่จุดที่โครงข้อหมุนชนกับเสาพอดี เป็น pinned support ทั้งหมด 8 จุด และแบ่งตัวแปรหน้าตัดเป็น 3 ตัวแปรคือ Top Chord, Bottom Chord และ Diagonal Chord ซึ่งเป็นการกำหนดหน้าตัดให้สามารถนำไปใช้ในการออกแบบและก่อสร้างจริง

ตารางที่ 4 ผลลัพธ์การออกแบบ

การออกแบบ	น้ำหนักของโครงสร้าง	หน้าตัดโครงข้อหมุน (เส้นผ่าศูนย์กลาง x ความหนา)		
		มิลลิเมตร x มิลลิเมตร		
ครั้งที่	กิโลกรัม	Top	Bottom	Diagonal
1	3164.27	34.0 x 2.3	48.6 x 3.2	34.0 x 2.3
2	2749.72	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
3	4215.37	27.2 x 2.3	101.6 x 3.2	34.0 x 2.3
4	3072.79	42.7 x 2.3	48.6 x 2.3	34.0 x 2.3
5	2749.72	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
6	3892.38	60.5 x 3.2	48.6 x 2.3	34.0 x 2.3
7	2749.72	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
8	2749.72	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
9	5725.37	114.3 x 3.2	48.6 x 3.2	34.0 x 2.3
10	3072.79	42.7 x 2.3	48.6 x 2.3	34.0 x 2.3

โดยจากการออกแบบอย่างเหมาะสมในกรณีศึกษานี้พบว่า โปรแกรมออกแบบใช้ระยะเวลานานในการออกแบบเนื่องจากโครงสร้างมีความซับซ้อนสูง ทำให้ถ้าหากต้องการออกแบบอย่างมีประสิทธิภาพต้องทำการออกแบบซ้ำหลายรอบ ซึ่งจะใช้เวลาในการทำงานมาก

5. สรุปผล

โครงงานนี้ได้พัฒนาโปรแกรมการออกแบบอย่างเหมาะสมด้วยวิธีเมตะ-ฮิวริสติก โดยใช้วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO) ในการออกแบบโครงสร้างโครงข้อหมุนเหล็กอย่างเหมาะสม เพื่อออกแบบโครงสร้างที่มีน้ำหนักเหล็กน้อยที่สุดเท่าที่ทำได้ โดยได้ทำการออกแบบด้วยมาตรฐาน AISC 360-16 และใช้หน้าตัดเหล็กท่อกลมมาตรฐาน มอก. ของประเทศไทย เพื่อให้สามารถนำโปรแกรมที่พัฒนาขึ้นมาไปใช้ในงานการออกแบบจริงได้อย่างมีประสิทธิภาพ โดยโปรแกรมทำการคำนวณแรงภายในของโครงสร้างด้วยโปรแกรมตั้งต้น Project-Python3D และทำการออกแบบอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่ม ในวิธีนี้มีขั้นตอนการดำเนินงานไม่ยาก ทำให้ใช้ทรัพยากรการทำงานของคอมพิวเตอร์น้อย และสามารถประแก้พารามิเตอร์ได้ตามต้องการ ขึ้นอยู่กับรูปแบบผลลัพธ์ที่ต้องการ วิธีนี้สามารถหาคำตอบของโครงสร้างที่มีจำนวนตัวแปรหน้าตัดน้อยและมากได้ โดยถ้าหากมีตัวแปรหน้าตัดน้อยจะสามารถหาคำตอบที่เหมาะสมได้ในการคำนวณซ้ำน้อยครั้ง ด้วยความที่โปรแกรมนี้เป็นโปรแกรม modified PSO แบบมีการปรับแต่งพัฒนาเพียงเล็กน้อย คือการใช้ค่า Inertia Weight, w ซึ่งเป็นตัวแปรตัวเดียวที่ถูกเพิ่มมาจาก PSO แบบดั้งเดิม ทำให้ความละเอียดและแม่นยำในการค้นหาคำตอบอาจไม่ดีเท่าอัลกอริทึมที่มีการปรับแต่งขั้นสูงกว่านี้ โดยอัลกอริทึม PSO สามารถใช้ในการออกแบบโครงสร้างได้จริง เนื่องจากการออกแบบโครงสร้างโครงข้อหมุนเหล็กในการทำงานจริง จะใช้ตัวแปรหน้าตัดจำนวนไม่มากเพราะเหตุผลในการทำงานก่อสร้างหน้างาน เนื่องด้วยถ้าหากตัวแปรมากเกินไปอาจทำให้การอ่านแบบเพื่อก่อสร้างสับสนและเกิดความผิดพลาด ซึ่งการออกแบบที่ตัวแปรไม่มากนี้ทำให้โปรแกรมนี้สามารถใช้งานได้อย่างมีประสิทธิภาพ และสะดวกต่อการใช้งาน ผู้ใช้งานสามารถปรับแต่งค่าคงที่ของโปรแกรม และสามารถใส่ข้อมูลโครงสร้างได้ตามต้องการ จากนั้นสามารถกดเรียกใช้โปรแกรมและได้ output ออกมาเป็นหน้าตัดที่ถูกออกแบบอย่างเหมาะสม และน้ำหนักของโครงสร้างได้โดยง่าย และสามารถทำการคำนวณซ้ำได้ตามที่ต้องการ

เอกสารอ้างอิง

[1] American Institute of Steel Construction (2016), Specification for Structural Steel Buildings (ANSI/AISC 360-16), AISC Committee on Specifications

[2] ชานน อธิปัญญา และ ณัฐชยา ศิริมาตย์ (2562), การพัฒนาโปรแกรมวิเคราะห์โครงสร้าง Truss และ Frame ใน 3 มิติ, ปริญญานิพนธ์จุฬาลงกรณ์มหาวิทยาลัย

[3] James Kennedy and Russel Eberhart (1995), Particle Swarm Optimization, Purdue School of Engineering and Technology Indianapolis, IN 46202-5160.

[4] K.Behdinan and R.E Perez (2007), Particle swarm approach for structural design optimization, Computers and Structures 85 (2007) 1579–1588.

[5] Dapei Tan, Dongshu Wang and Lei Liu (2018), Particle swarm optimization algorithm: an overview, Soft Comput (2018) 22:387–408.

[6] Fleury C and Schmit L. (1980). Discrete-continuous variable structural synthesis using dual methods. AIAA J 1980;18:1515–24.

การออกแบบอย่างเหมาะสมสำหรับโครงข้อหมุนเหล็กด้วยวิธีความฉลาดแบบกลุ่ม

นาย เพทาย วงศ์ศิลากิจ 6230391721

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาวิชาโครงงานวิศวกรรมโยธา

ตามหลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมโยธา

ภาควิศวกรรมโยธา คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2565

Optimum Design of Steel Truss by Particle Swarm Optimization

Mr. Pathay Wongsilakij 6230391721

Senior Project Submitted in Partial Fulfillment of the Requirements

For the Degree of Bachelor of Engineering Program in Civil Engineering

Department of Civil Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2022

หัวข้อโครงงาน	การออกแบบอย่างเหมาะสมสำหรับโครงข้อหมุนเหล็กด้วยวิธีความฉลาดแบบกลุ่ม
โดย	นาย เพทาย วงศ์ศิลากิจ
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ ดร. วัฒนชัย สมิทธากร

บทคัดย่อ

โครงงานนี้จัดทำเพื่อนำเสนอการออกแบบโครงสร้างโครงข้อหมุนเหล็กอย่างเหมาะสมด้วยภาษาไพทอน โดยจุดประสงค์หลักคือการพัฒนาโปรแกรม ออกแบบหน้าตัดโครงเหล็กที่สามารถใช้งานได้จริงและนำไปพัฒนาเพื่อใช้ในโครงสร้างประเภทอื่นต่อได้ โปรแกรมถูกพัฒนาด้วยการทำงานร่วมกัน ของโปรแกรมวิเคราะห์โครงสร้าง และอัลกอริทึมปัญญาประดิษฐ์, Particle Swarm Optimization(PSO), เป็นอัลกอริทึมที่จำลองธรรมชาติของ การออกหาอาหารของฝูงนก มีจุดเด่นคือการประมวลผลโดยใช้เวกเตอร์ในการคาดการณ์คำตอบ ซึ่งสอดคล้องกับการออกแบบโครงสร้างอย่าง เหมาะสม ที่จำเป็นต้องออกแบบซ้ำเพื่อหาหน้าตัดที่ประหยัดที่สุด โดยกระบวนการประมวลผลต้องใช้เวลานาน จึงนำอัลกอริทึม PSO เข้ามาช่วย เพิ่มประสิทธิภาพในการออกแบบ ทั้งเรื่องเวลาและความแม่นยำ นอกจากนี้ ภาษาไพทอนมีคุณสมบัติที่ง่ายต่อการทำความเข้าใจและนำไปพัฒนา ต่อ เป็นภาษาที่กระชับ สามารถปรับแต่งได้โดยง่ายตามวัตถุประสงค์ที่ต้องการ ในโครงงานนี้จะเขียนโปรแกรมออกแบบด้วยภาษาไพทอนด้วย อัลกอริทึม PSO และนำโครงสร้างที่ออกแบบได้มาเปรียบเทียบกับความประหยัดกับโครงเหล็กที่ได้ออกแบบไว้ด้วยวิธีดั้งเดิมเพื่อหาประสิทธิภาพ ของโปรแกรมออกแบบ

คำสำคัญ: การออกแบบอย่างเหมาะสม, โครงข้อหมุนเหล็ก, วิธีความฉลาดแบบกลุ่ม

Title	Optimum Design of Steel Truss by Particle Swarm Optimization
Students	Mr. Pathay Wongsilakij
Advisor	Assoc. Prof. Dr. Wattanachai Smittakorn

Abstract

In this study, a structural optimization program is developed using Python programming language. The main purpose is to apply to real-world steel truss problems and to develop for other structural types. The program development uses a structural analysis program paired with an AI algorithm, Particle Swarm Optimization (PSO) designed to imitate a flock of birds searching for food. PSO uses vectors to predict optimum answers, therefore, it is similar to the traditional optimizing method, which repeats the calculations to solve for optimum design. As a result, PSO is adopted to improve the efficiency and accuracy of the calculation. Moreover, Python language is simple, customizable, and able for further development. In conclusion, the program uses Python programming language with PSO algorithm to optimize steel truss structures, then the results are compared to the traditional optimization to evaluate the efficiency.

Keywords: Optimum design, Steel truss, Particle Swarm Optimization

กิตติกรรมประกาศ

โครงงานพัฒนาโปรแกรมออกแบบโครงสร้างอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่ม สำเร็จลุล่วงไปได้ด้วยดี ด้วยความช่วยเหลือจาก รองศาสตราจารย์ ดร. วัฒนชัย สมิทธากร ผู้เป็นอาจารย์ที่ปรึกษาโครงงาน ที่คอยให้คำแนะนำ ความช่วยเหลือ แสดงข้อคิดเห็นต่าง ๆ ที่เป็นประโยชน์ต่อผู้วิจัย และให้ความรู้เพื่อแก้ปัญหาจากข้อสงสัยที่ผู้วิจัยมี ได้คอยสละเวลาเพื่อเข้ามาช่วยเหลือผู้วิจัย ตลอดจนคอยปรับแก้ข้อบกพร่องของโครงงานนี้ด้วยความใส่ใจเป็นอย่างมาก ผู้วิจัยขอกราบขอบพระคุณท่านอาจารย์ ขอบพระคุณ บิดา มารดา ที่ให้กำลังใจในการดำเนินงานโครงการ และนี้กราบขอบพระคุณอาจารย์ทุกท่านที่เคยได้ถ่ายทอดความรู้อันเป็นประโยชน์ต่อผู้วิจัยสั่งสอนและพัฒนาให้ผู้วิจัยมีความรู้มีความสามารถในด้านวิศวกรรมโยธา เพื่อพัฒนาโครงงานนี้ให้เสร็จสิ้นสมบูรณ์ได้

หากโครงงานนี้มีข้อผิดพลาดประการใด ผู้วิจัยขอรับผิดชอบพียงผู้เดียว และพร้อมนำข้อติชมมาปรับปรุงแก้ไข และพัฒนาโครงงานนี้ต่อไป

เพทาย วงศ์ศิลากิจ

ผู้วิจัย

สารบัญ

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

การออกแบบอย่างเหมาะสมถือเป็นปัจจัยสำคัญในขั้นตอนการก่อสร้างสิ่งปลูกสร้าง เนื่องจากวิศวกร-โครงสร้างมีหน้าที่ในการค้นหารูปแบบการสร้างสิ่งปลูกสร้างที่ประหยัดค่ามากที่สุด ในขณะที่ต้องคำนึงถึงความปลอดภัยและความเหมาะสมของโครงสร้างอีกด้วย ในศาสตร์ของวิศวกรรมโยธาจึงมีการค้นหาวิธีการมากมายในการสร้างเครื่องมือที่สามารถออกแบบโครงสร้างที่มีความแข็งแรงและประหยัดค่าใช้จ่ายในเวลาเดียวกัน เพื่อที่จะสร้างโครงสร้างที่ปลอดภัยให้ประชากรสามารถใช้งานได้อย่างเชื่อมั่น สามารถประหยัดเวลาในการออกแบบ และสามารถช่วยลดปริมาณทรัพยากรและต้นทุนที่ใช้ในการก่อสร้าง ซึ่งเป็นสิ่งที่สำคัญมากในโลกยุคสมัยใหม่ การสร้างเครื่องมือในการออกแบบตามจุดประสงค์นี้ จึงเป็นการสร้างทางเลือกที่วิศวกรโยธาสามารถนำไปใช้งานได้

ด้วยยุคสมัยนี้เป็นยุคสมัยที่มีการพัฒนาอย่างรวดเร็วในด้านของเทคโนโลยี ในทางวิศวกรรมโยธาจึงพัฒนานำเทคโนโลยียุคใหม่มาใช้ในศาสตร์ที่เก่าแก่ โดยการประยุกต์การคำนวณออกแบบโครงสร้างด้วยกระดาษและดินสอ เปลี่ยนมาเป็นการใช้คอมพิวเตอร์ ซึ่งคอมพิวเตอร์นั้นถูกใช้ในงานวิศวกรรมโยธามายาวนานและถูกพัฒนามาตามยุคสมัย ณ ปัจจุบัน วิศวกรโยธาได้สร้างโปรแกรมคอมพิวเตอร์ที่ใช้ในการคำนวณออกแบบโครงสร้างที่เป็นที่นิยม เช่น Extended3D Analysis of building System (ETABS) และ Structural Analysis Program2000 (SAP2000) เป็นต้น โดยทั้งสองโปรแกรมถูกเขียนด้วยภาษาคอมพิวเตอร์ที่เป็นกรรมสิทธิ์เฉพาะของบริษัท ไม่สามารถให้บุคคลภายนอกนำมาพัฒนาต่อได้อย่างอิสระ จึงเป็นไปได้ยากที่วิศวกรโยธาจะนำมาพัฒนาต่อในทิศทางที่ตนเองต้องการ นอกจากนี้ภาษาคอมพิวเตอร์ในยุคปัจจุบันที่ได้รับความนิยมมีหลายภาษาเช่น C, C++, Java และ Python ภาษาเหล่านี้มีข้อดีและข้อเสียที่แตกต่างกัน แต่ทุกภาษานั้นวิศวกรสามารถเรียนรู้ได้ด้วยตนเองและสามารถพัฒนาโปรแกรมออกแบบคำนวณโครงสร้างอย่างเหมาะสมได้ เพื่อทำให้งานโครงสร้างที่ออกแบบมามีความประหยัดมากที่สุด

อีกทั้งโปรแกรมออกแบบโครงสร้างที่นิยมในงานวิศวกรรมโยธานั้น ทางตัววิศวกรเองต้องใช้วิธีการดังเดิมอย่าง Trial and Error เพื่อค้นหาคำตอบของชิ้นส่วนของโครงสร้างที่คุ้มค่าและปลอดภัยที่สุด และการทำงานในขั้นตอนนี้อาจใช้เวลาในการดำเนินการนานตามความซับซ้อนของโครงสร้าง เนื่องจากต้องทำการแทนค่าตัวแปรทั้งหมดเพื่อออกแบบชิ้นส่วนแต่ละชิ้น และถ้าหากมีตัวแปรตัวใดตัวหนึ่งเปลี่ยนแปลงค่า ตัวแปรอื่นก็ต้องทำการ

คำนวณใหม่ทุกครั้งที่ทำซ้ำ ซึ่งโจทย์ส่วนใหญ่ในการวิเคราะห์ออกแบบโครงสร้างนั้นเป็นแบบ Non-Linear จึงเป็นเรื่องยากที่จะหาคำตอบด้วยวิธีนี้ ทางผู้วิจัยจึงได้ศึกษาเกี่ยวกับอัลกอริทึมต่าง ๆ เช่น Genetic Algorithm (GA), Firefly Algorithm (FA), Harmony Search Algorithm(HSA), Ant Colony optimization (ACO) และ Particle Swarm Optimization (PSO) ซึ่ง PSO เป็นอัลกอริทึมที่นิยมในการออกแบบเหมาะสมและเป็นต้นแบบให้อัลกอริทึมแยกย่อยอีกมากมาย โดยเลือกใช้ PSO เนื่องจากมีการสุ่มชุดข้อมูลแบบต่อเนื่องและเข้ากับการออกแบบโครงสร้างที่มีคุณสมบัติหน้าตัดตายตัว โดยอัลกอริทึมนี้จะเปรียบการหาคำตอบสุดท้ายโดยการจำลองการออกหาอาหารของฝูงนก ที่เริ่มต้นจะมีการกระจายตัวและสุดท้ายจะมุ่งหน้าเข้าหาเป้าหมายเดียวกัน โดยคำตอบที่ได้จะเป็นคำตอบแบบ metaheuristic หมายความว่าเป็นคำตอบที่ใกล้เคียงค่าที่ดีที่สุด แต่ไม่สามารถทราบค่าที่ดีที่สุดที่แท้จริงได้

โดยการพัฒนาโปรแกรมในโครงงานนี้ยังใช้ภาษาคอมพิวเตอร์ไพทอนในการเขียนโปรแกรมออกแบบอย่างเหมาะสม เพราะภาษานี้ง่ายต่อการทำความเข้าใจทั้งผู้ใช้งาน และผู้พัฒนาโปรแกรม โดยในส่วนของการวิเคราะห์โครงสร้างนั้น จะใช้โปรแกรมคำนวณแรงในโครงสร้าง Project-Python3D ของ รศ. ดร. วัฒนชัย สมิทธากร ที่ถูกพัฒนามาในภาษาไพทอนเช่นกัน ทางผู้วิจัยจึงเลือกใช้โปรแกรมนี้ในการพัฒนาต่อให้สามารถออกแบบอย่างเหมาะสมได้ในตัวโปรแกรมเดียว และสามารถนำโปรแกรมไปพัฒนาต่อได้โดยวิศวกรที่มีความรู้ด้านวิศวกรรมโยธาและภาษาไพทอน

1.2 วัตถุประสงค์

โครงงานนี้จัดทำเพื่อพัฒนาโปรแกรมคำนวณแรงในโครงสร้างโครงข้อหมุนเหล็กสามมิติให้สามารถออกแบบอย่างเหมาะสมได้ ทำการพัฒนาอัลกอริทึมด้วยวิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO) ในการคำนวณออกแบบหน้าตัดที่ประหยัดที่สุดที่สามารถรับแรงได้อย่างปลอดภัย และการออกแบบสามารถนำไปใช้กับงานออกแบบโครงสร้างจริงได้ และเนื่องจากโปรแกรมภาษาไพทอนผู้วิจัยยังคาดว่าโปรแกรมสามารถถูกนำไปพัฒนาต่อให้สามารถคำนวณออกแบบอย่างเหมาะสมกับโครงสร้างชนิดอื่นได้ โดยใช้อัลกอริทึมที่ผู้พัฒนาโปรแกรมได้สร้างไว้

1.3 ขอบเขตการวิจัย

โครงการพิจารณาการออกแบบโครงสร้างโครงข้อหมุน โดยนำค่าแรงภายในโครงข้อหมุนแต่ละชิ้นมาทำการออกแบบอย่างเหมาะสมด้วยอัลกอริทึมความฉลาดแบบกลุ่ม มีขอบเขตพิจารณาดังนี้

1. ใช้แรงที่ได้จากการคำนวณแรงภายในของโปรแกรมวิเคราะห์โครงสร้างสามมิติ Project-Python3D เพื่อนำมาวิเคราะห์ออกแบบหน้าตัดโครงข้อหมุนในโครงสร้าง
2. วิเคราะห์ออกแบบเหมาะสมด้วยภาษาไพทอน
3. การออกแบบใช้มาตรฐานการออกแบบ AISC 360-16
4. ใช้การออกแบบเหล็กแบบ Load and Resistance Factor Design (LRFD)
5. ออกแบบด้วยหน้าตัดโครงข้อหมุนเหล็ก ใช้เหล็กท่อกลมมาตรฐาน มอก.
6. ใช้อัลกอริทึมในการออกแบบด้วย Particle Swarm Optimization (PSO)
7. การออกแบบเหมาะสมครอบคลุมโครงสร้างโครงข้อหมุนทั้งสองมิติและสามมิติ
8. วิเคราะห์แรงในโครงสร้างโครงข้อหมุนด้วย Point Load, คำนวณแรงภายในเป็นแรงดึง และแรงอัดเท่านั้น ไม่คิดค่าการกระจัดของพิกัดจุดต่อ (Node) ของโครงสร้าง
9. สามารถให้ output ได้ในตัวโปรแกรมออกมาเป็นหน้าตัดเหล็กท่อกลมตามชิ้นส่วนที่กำหนด

1.4 แผนการดำเนินการ

1. ศึกษาการทำงานและโค้ดของโปรแกรมวิเคราะห์แรงภายในโครงสร้าง และภาษาไพทอน
2. พัฒนาโปรแกรมวิเคราะห์โครงสร้างให้สามารถนำแรงภายในมาตรวจสอบความปลอดภัยตามมาตรฐาน AISC 360-16 ด้วยวิธีการออกแบบ LRFD
3. ศึกษาและพัฒนาอัลกอริทึมออกแบบเหมาะสม PSO
4. พัฒนาโปรแกรมวิเคราะห์แรงภายในโครงสร้างเข้ากับการออกแบบเหมาะสมด้วยอัลกอริทึม PSO

1.5 ขั้นตอนการดำเนินการ

1. ศึกษาภาษาไพทอน และนำความรู้มาใช้ในการศึกษาขั้นตอนการทำงานอย่างละเอียดของโปรแกรม Project-Python3D เพื่อให้ง่ายต่อการพัฒนาให้สามารถออกแบบเหมาะสมได้
2. นำโปรแกรมมาพัฒนาให้สามารถตรวจสอบความปลอดภัยตามมาตรฐาน AISC 360-16 และสร้างชุดข้อมูลของหน้าตัดเหล็กท่อกลมตามมาตรฐาน มอก.

3. พัฒนาโปรแกรมวิเคราะห์โครงสร้างให้สามารถวิเคราะห์ข้อมูลซ้ำได้โดยไม่เกิดการทับซ้อนของแรงภายใน
4. ศึกษาวิธีการออกแบบเหมาะสมด้วย PSO เพื่อนำพื้นฐานของอัลกอริทึมมาใช้ในการพัฒนาโปรแกรมออกแบบ
5.. วิเคราะห์โครงสร้างตัวอย่างเพื่อหาหน้าตัดที่ประหยัดที่สุดด้วยวิธีการ Trial and Error เพื่อใช้หน้าตัดเป็นหน้าตัดอ้างอิงในการตรวจสอบประสิทธิภาพของการออกแบบด้วยอัลกอริทึม PSO
6. พัฒนาโปรแกรมให้ออกแบบเหมาะสมด้วยอัลกอริทึม PSO และตรวจสอบความประหยัดเทียบกับการออกแบบด้วยวิธีดั้งเดิม เพื่อวิเคราะห์และสรุปผลการออกแบบอย่าง เหมาสม
7. จัดทำรูปเล่มปริญญานิพนธ์

1.6 ประโยชน์ที่ได้รับ

โครงงานนี้ได้พัฒนาโปรแกรมวิเคราะห์โครงสร้าง Project-Python3D ให้สามารถออกแบบโครงข้อหมุนที่มีหน้าตัดประหยัดที่สุดเป็นโปรแกรมที่สามารถนำไปใช้งานได้จริง โดยสามารถออกแบบได้ตามมาตรฐาน AISC 360-16 โดยการนำอัลกอริทึม Particle Swarm Optimization (PSO) มาใช้ออกแบบแทนการใช้วิธีการดังเดิมอย่าง Trial and Error ด้วยมนุษย์ ซึ่งสามารถประหยัดเวลาในการออกแบบได้อย่างมากและใช้แรงงานในการออกแบบน้อยลง อีกทั้งในขั้นตอนการออกแบบเหมาสมในโปรแกรมยังสามารถใช้งานและปรับแต่งค่าพารามิเตอร์แต่ละตัว ได้อย่างอิสระ และไม่ซับซ้อน สามารถเลือกจำนวนครั้งในการทำซ้ำของโปรแกรมหรือจำนวนอนุภาคที่ใช้ในการค้นหาข้อมูล เพื่อให้เหมาะสมกับประสิทธิภาพของหน่วยความจำคอมพิวเตอร์ของผู้ใช้งาน เพื่อให้สามารถใช้งานได้กับประสิทธิภาพคอมพิวเตอร์ทุกเครื่อง นอกจากนี้โปรแกรมที่พัฒนาออกมายังสามารถนำไปใช้พัฒนาต่อให้สามารถออกแบบโครงสร้างชนิดอื่น เช่น Steel Frame, โครงสร้างคอนกรีตเสริมเหล็ก, โครงสร้างผสม เป็นต้น เนื่องจากโปรแกรมเป็นโปรแกรมที่วิเคราะห์แรงภายในโครงสร้างในตัว จึงสามารถพัฒนาได้ตั้งแต่ขั้นตอนการวิเคราะห์แรงจนถึงขั้นตอนการออกแบบหน้าตัดโครงสร้าง

บทที่ 2
งานวิจัยในอดีต

2.1 ทฤษฎีการออกแบบอย่างเหมาะสม (Structural Optimization)

การออกแบบโครงสร้างอย่างเหมาะสมคือการนำวิธีการทางคณิตศาสตร์มาใช้เพื่อ ออกแบบโครงสร้างให้มีความประหยัดมากที่สุด ทั้งในด้านของวัสดุที่ใช้, เวลาที่ใช้ในการก่อสร้าง หรืองบประมาณก่อสร้างได้โดยโครงสร้างผ่านมาตรฐานความปลอดภัย โดยเป็นกระบวนการที่สำคัญอย่างมากในอุตสาหกรรมการก่อสร้าง ซึ่งวิศวกรหลายท่านได้พยายามคิดค้นวิธีการมากมายเพื่อทำให้การออกแบบเหมาะสมมีประสิทธิภาพมากที่สุด

โดยในการออกแบบโครงสร้างโครงข้อหมุนเหล็ก (Steel Truss) นั้นสำคัญอย่างมากในการออกแบบอย่างเหมาะสม เนื่องจากเป็นโครงสร้างที่สามารถมีตัวแปรหน้าตัดและรูปร่างของโครงสร้างได้หลายตัวแปรตามวิศวกรกำหนด หรือตามความสะดวกเข้าใจง่ายในการทำงานหน้างานก่อสร้าง ซึ่งทำให้การออกแบบอย่างเหมาะสมนั้นสามารถทำได้หลากหลายประเภท เนื่องด้วยโครงสร้างเหล็กมีความอิสระในการออกแบบสูง สามารถปรับแต่งองศาการจัดเรียงตัว หรือรูปร่างและขนาดของหน้าตัดได้ตามต้องการ โดยอยู่ภายใต้มาตรฐานความปลอดภัยที่สามารถรับน้ำหนักในสถานการณ์การใช้งานจริงได้

การออกแบบอย่างเหมาะสมมีขั้นตอนทางคณิตศาสตร์ที่สามารถสร้างออกมาเป็นฟังก์ชันได้ดังนี้

• **ฟังก์ชันวัตถุประสงค์ (Objective function, f)**: ฟังก์ชันที่ใช้ในการออกแบบโครงสร้างที่ต้องการ โดยทุกความเป็นไปได้ในการออกแบบ f คือผลลัพธ์ของคำตอบที่บ่งชี้คุณภาพของคำตอบ โดยทั่วไปการออกแบบอย่างเหมาะสมจะเลือกค่า f ที่ให้คำตอบน้อยที่สุด (ยกตัวอย่างเช่น การออกแบบเพื่อหาน้ำหนักของโครงสร้างที่น้อยที่สุด หมายความว่าใช้หน้าตัดโครงสร้างที่เล็กที่สุด) ส่วนใหญ่แล้วค่า f จะใช้วัดน้ำหนัก, การกระจัดในทิศทางที่กำหนด, แรงภายในประสิทธิผล หรือต้นทุนในการก่อสร้างโครงสร้างนั้น

• **ตัวแปรการออกแบบ (Design variables, x)**: ฟังก์ชันหรือเวกเตอร์ที่ใช้อธิบายการออกแบบนั้น และสามารถเปลี่ยนแปลงระหว่างการออกแบบอย่างเหมาะสมได้ ตัวแปรนี้อาจระบุรูปทรงทางเลขาคณิตหรือตัวเลือกวัสดุที่ใช้ในการออกแบบ ถ้าหาก x เป็นตัวแปรที่ระบุรูปทรงทางเลขาคณิต อาจระบุเป็น รูปร่าง, หน้าตัดของชิ้นส่วนของโครงสร้าง หรือความหนาของแผ่นชิ้นส่วนโครงสร้าง

• **ตัวแปรสถานะ (State variable, y)**: ในโครงสร้างที่กำหนด x, y คือฟังก์ชันหรือเวกเตอร์ที่บ่งบอกความตอบสนองของโครงสร้าง โดยในกลศาสตร์ของโครงสร้าง ความตอบสนองหมายถึง การกระจัด, ความเค้น, ความเครียด หรือแรงกระทำ

ในการออกแบบเหมาะสมทั่วไป ปัญหาของโครงสร้างสามารถระบุได้ดังนี้

$$(\mathbb{SO}) \quad \begin{cases} \text{minimize } f(x, y) \text{ with respect to } x \text{ and } y \\ \text{subject to} \quad \begin{cases} \text{behavioral constraints on } y \\ \text{design constraints on } x \\ \text{equilibrium constraint.} \end{cases} \end{cases}$$

เราสามารถออกแบบปัญหาของโครงสร้างโดยมี objective function หลายตัวได้ โดยเรียกมันว่า Vector Optimization

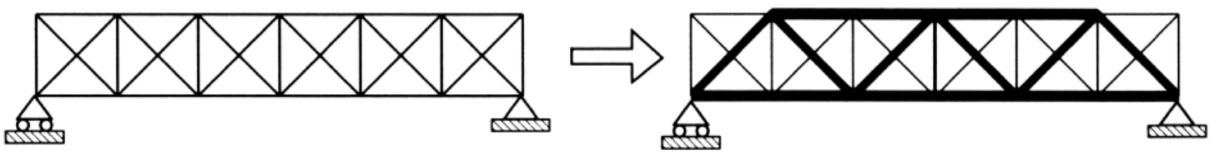
$$\text{minimize } (f_1(x, y), f_2(x, y), \ldots, f_l(x, y)),$$

โดยที่ l คือจำนวนของ objective function และข้อจำกัดต่าง ๆ ของปัญหาจะเหมือนกัน

โดยจากทฤษฎีข้างต้นสามารถแบ่งวิธีการออกแบบอย่างเหมาะสมออกเป็น 3 ประเภทหลัก ได้แก่

2.1.1 การออกแบบขนาดอย่างเหมาะสม (Sizing Optimization)

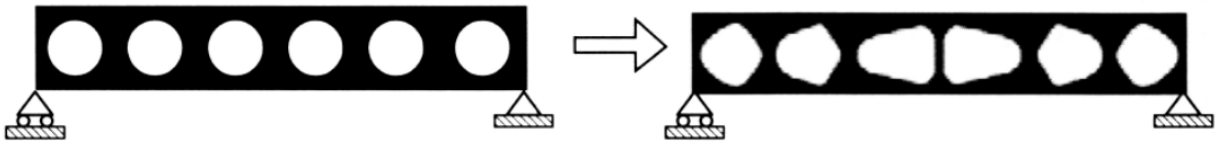
ทฤษฎีการออกแบบเพื่อหาผลลัพธ์ที่เกี่ยวข้องกับ ขนาดหน้าตัด, รูปร่างหน้าตัด, วัสดุ และองค์ประกอบอื่น ๆ ที่ไม่เกี่ยวข้องกับรูปร่างหรือการเรียงตัวของโครงสร้างโดยรวม ในกรณีที่มีโจทย์เป็นรูปแบบการเรียงตัวของโครงสร้างและตำแหน่งรับน้ำหนัก ซึ่งเป็นการออกแบบที่พิจารณาแรงภายในของโครงสร้างและเปลี่ยนแปลงหน้าตัดทั้งในด้านของ ความกว้าง, ความลึก หรือรูปร่างของหน้าตัด เพื่อหาผลลัพธ์คุณสมบัติหน้าตัดที่ประหยัดที่สุดในการออกแบบโครงสร้างนั้น



รูปที่ 2-1 การออกแบบขนาดอย่างเหมาะสม

2.1.2 การออกแบบรูปร่างอย่างเหมาะสม (Shape Optimization)

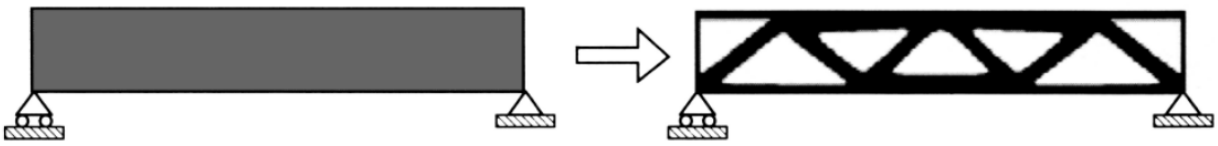
ทฤษฎีการออกแบบที่หาผลลัพธ์ในรูปแบบของรูปแบบการจัดเรียงตัวของโครงสร้าง โดยการออกแบบจะมีรูปร่างคร่าว ๆ , หน้าตัด, แรงกระทำ และวัสดุที่ใช้ในโครงสร้างเป็นโจทย์เริ่มต้น และทำการคำนวณด้วยวิธีของผู้ออกแบบเพื่อหารูปแบบการจัดเรียงโครงสร้างให้ประหยัดที่สุด ในการออกแบบจะทำการหารูปร่างที่เหมาะสมกับโครงสร้างมากที่สุดโดยคงรูปแบบเริ่มต้นไว้ แต่วัสดุและหน้าตัดชิ้นส่วนโครงสร้างยังคงเดิม



รูปที่ 2-2 การออกแบบรูปร่างอย่างเหมาะสม

2.1.3 การออกแบบอย่างเหมาะสมโดยใช้โทโพโลยี (Topology Optimization)

ทฤษฎีการออกแบบที่มีอิสระในการออกแบบสูงที่ใช้การคำนวณตามหลัก Finite Element เป็นหลัก โดยโจทย์เริ่มต้นมีเพียงพื้นที่ที่ต้องการสร้างโครงสร้าง, วัสดุ และแรงกระทำในสร้างโครงสร้าง จากนั้นทำการคำนวณ Stress Analysis เพื่อหาคำตอบของรูปร่างที่เหมาะสมที่สุดที่มีความประหยัดมากที่สุดและอยู่ในมาตรฐานความปลอดภัย โดยวิธีการนี้เป็นการออกแบบเหมาะสมที่คำนวณอย่างละเอียดโดยจะทำการนำส่วนที่รับแรงน้อยมากออกจากโครงสร้างและกระจายแรงเข้าสู่ส่วนอื่นที่มีพื้นที่รับแรงมากจนเหลือน้อยที่สุด ทำให้ผลลัพธ์ที่ได้เป็นโครงสร้างที่มีรูปร่างที่ประหยัดที่สุด



รูปที่ 2-3 การออกแบบอย่างเหมาะสมโดยใช้โทโพโลยี

2.2 วิธีการออกแบบอย่างเหมาะสม (Optimization Method)

ในยุคอดีต การออกแบบอย่างเหมาะสมของโครงสร้างนานาชนิดนั้นถูกออกแบบด้วยการคำนวณจากมนุษย์ โดยวิศวกรต้องทำการออกแบบโครงสร้างด้วยตนเอง และทำการลองผิดลองถูกจนกว่าจะได้โครงสร้างที่ปลอดภัยและประหยัดที่สุด กระบวนการทำงานนี้เป็นกระบวนการที่ใช้เวลาในการทำงานมาก อีกทั้งยังใช้แรงงานวิศวกรในการวิเคราะห์ข้อมูลโครงสร้างเป็นจำนวนมากอีกด้วย โดยต่อมาคอมพิวเตอร์ได้มีบทบาทเข้ามาช่วยเหลือในการ

คำนวณออกแบบ ทั้งการวิเคราะห์แรงภายในของโครงสร้างและการออกแบบอย่างเหมาะสมด้วยโปรแกรมคอมพิวเตอร์ ทำให้ขั้นตอนการออกแบบโครงสร้างนั้นมีความสะดวกสบายและประหยัดเวลาได้อย่างมาก วิศวกรจึงพยายามหาทางในการออกแบบโปรแกรมคอมพิวเตอร์ที่ใช้ออกแบบโครงสร้างให้มีประสิทธิภาพมากขึ้นตลอดมา

การออกแบบอย่างเหมาะสมในในยุคปัจจุบันได้มีรูปแบบกระบวนการการคำนวณมากมาย เนื่องด้วยความนิยมของการใช้เทคโนโลยีคอมพิวเตอร์ที่สูงขึ้นทำให้เกิดวิธีการมากมายในการออกแบบแทนการออกแบบด้วยวิธี Trial and Error ซึ่งจุดประสงค์ในการสร้างวิธีการออกแบบอย่างเหมาะสมหลากหลายรูปแบบคือ เพื่อให้สามารถออกแบบโครงสร้างได้ในเวลาที่รวดเร็วและเกิดประสิทธิภาพมากที่สุด ด้วยความต้องการนี้ทำให้เกิดการคิดค้นวิธีการออกแบบอย่างเหมาะสมที่ใช้พื้นฐานที่แตกต่างกัน ไม่ว่าจะเป็น การลอกเลียนแบบพฤติกรรมของสัตว์, การใช้พื้นฐานทางคณิตศาสตร์เป็นหลัก หรือการใช้กฎของธรรมชาติ เข้ามาช่วยเป็นตัวกำหนดขั้นตอนการออกแบบ ในแต่ละวิธีการออกแบบจะมีจุดเด่นและจุดด้อยที่ไม่เหมือนกัน ทำให้จุดประสงค์, ประสิทธิภาพ, ความแม่นยำ, เวลาที่ใช้ในการออกแบบ และความซับซ้อนของโปรแกรม เป็นตัวแปรสำคัญในการเลือกใช้วิธีการเหล่านี้

การหาคำตอบของการออกแบบอย่างเหมาะสมมีความซับซ้อนมากขึ้นตามยุคสมัย เนื่องด้วยการออกแบบโครงสร้างที่ซับซ้อนขึ้น และเทคโนโลยีการก่อสร้างที่พัฒนาขึ้น ทำให้ปัญหาการออกแบบมีพฤติกรรมแบบ Non-Linear และมีตัวแปรจำนวนมาก จึงมีการคิดค้นการแก้ปัญหาแบบฮิวริสติกและเมตะฮิวริสติกขึ้นมา

วิธีการฮิวริสติก คือ การหาคำตอบแบบเจาะจงเฉพาะคำถาม (problem-specific) โดยไม่การันตีคำตอบที่ถูกต้องที่สุด แต่มีประสิทธิภาพพอที่จะหาคำตอบที่ดีพอสำหรับปัญหานั้น โดยในอีกวิธีการหนึ่งคือ มีวิธีการเมตะฮิวริสติกซึ่งเป็นวิธีการขั้นสูงกว่าในการออกแบบอย่างเหมาะสมที่สามารถใช้ในการแก้ปัญหาได้หลากหลายกว่า โดยวิธีการเมตะฮิวริสติกเป็นวิธีการแก้ปัญหาทั่วไปที่ออกแบบมาเพื่อหาคำตอบที่ใกล้เคียงค่าที่ถูกต้องที่สุดอย่างมีประสิทธิภาพแต่ไม่การันตีค่าที่ถูกต้องที่สุดอย่างแท้จริง วิธีการนี้ส่วนใหญ่จะเป็นการเลียนแบบธรรมชาติไม่ว่าจะเป็น การเลียนแบบวิวัฒนาการ, การเลียนแบบฝูงสัตว์ และการหลอมละลายของโลหะ วิธีการเมตะฮิวริสติกค้นหาคำตอบด้วยการคำนวณซ้ำและพิจารณาค่าที่ดีที่สุดในแต่ละการทำซ้ำ จนได้คำตอบที่พึงพอใจ

โดยในวิธีการออกแบบเหมาะสมส่วนใหญ่ในยุคปัจจุบันจะคำนวณด้วยคอมพิวเตอร์จะเป็นการเขียนโปรแกรมขึ้นมาใช้ในการค้นหาคำตอบที่ดีที่สุด ซึ่งสามารถยกตัวอย่างวิธีการออกแบบอย่างเหมาะสมที่นิยมใช้ในการอออกแบบอย่างเหมาะสมโดยสังเขปได้ดังนี้

2.2.1 วิธีการคำนวณเชิงพันธุกรรม (Genetic Algorithm, GA)

วิธีการนี้ถูกคิดค้นโดย John Holland, นักวิทยาศาสตร์คอมพิวเตอร์และศาสตราจารย์ที่ University of Michigan, ในช่วงปี ค.ศ. 1960-1970 เพื่อเป็นการสร้างแบบจำลองการคัดเลือกทางธรรมชาติและการวิวัฒนาการ ซึ่งเขาได้สนใจในการสร้างวิธีการที่สามารถเรียนรู้และปรับตัวเข้ากับการเปลี่ยนแปลงของสิ่งแวดล้อมรอบข้างได้ โดยหลักการทำงานเบื้องต้นของวิธีการนี้ จะมีการสร้างประชากรกลุ่มเริ่มต้นมาที่สามารถให้คำตอบของปัญหาที่แตกต่างกัน และทำขั้นตอนการทำซ้ำด้วยการผสมพันธุ์ในกลุ่มประชากรเพื่อหาคำตอบใหม่ ๆ ผ่านการคัดเลือก, การผสมข้ามสายพันธุ์ และการกลายพันธุ์ ในขั้นตอนการคัดเลือกการผสมพันธุ์นี้จะมีข้อกำหนดการเลือกประชากรจากคุณภาพของผลลัพธ์คำตอบ ซึ่งเป็นตัวบ่งชี้ในประสิทธิภาพของประชากรนั้นที่มีต่อ objective function ประชากรที่มีผลลัพธ์ที่ดีก็จะมีโอกาสถูกคัดเลือกมากกว่าในการผสมพันธุ์เพื่อให้ได้ประชากรรุ่นต่อไปออกมา และจากการคัดเลือกจะทำให้เกิดการผสมกันของยีนส์ประชากรที่แตกต่างกันทำให้ประชากรรุ่นต่อไปมีคำตอบที่ออกมาและการกระทำซ้ำ ๆ เหล่านี้จะทำให้เกิดประชากรที่ให้ผลลัพธ์คำตอบที่ดีที่สุดออกมา

W. M. Jenkins (1991), Towards Structural Optimization via the Genetic Algorithm จาก University of Leeds เป็นบทความทางวิชาการ ที่ทดสอบการประยุกต์ใช้ของวิธีการคำนวณเชิงพันธุกรรมในการออกแบบโครงสร้างหลังคาคานและโครงข้อหมุนอย่างเหมาะสม ใช้ขั้นตอนการทำงานแบบ stochastic ซึ่งเป็นการผนวกเข้ากันระหว่างการคำนวณของแคลคูลัสกับความน่าจะเป็นในการสร้างกลุ่มประชากรเริ่มต้นและใช้หลักการคัดเลือกทางธรรมชาติและความอยู่รอดในการพัฒนาการออกแบบ โดยพิจารณาหลักการออกแบบและขั้นตอนการทำงานพื้นฐานคือ การคัดเลือก, การผสมข้ามสายพันธุ์, การกลายพันธุ์ และตัวแปรที่ปรับแต่งได้

โดยจากการทดลองดังกล่าวให้ผลลัพธ์ว่า การใช้วิธีการคำนวณเชิงพันธุกรรมนั้นให้ประโยชน์อย่างมากในการลดการทำงานของมนุษย์เนื่องจากเป็นวิธีการที่ใช้คอมพิวเตอร์เป็นหลัก เป็นวิธีการที่ให้อิสระกับผู้ใช้งานในการควบคุมตัวแปรต่าง ๆ ได้ด้วยตนเอง สามารถปรับแต่งค่าพารามิเตอร์ให้เหมาะสมตามต้องการ โดยการคำนวณแรงในโครงสร้างถ้าหากให้ผลลัพธ์ที่ไม่ผ่านมาตรฐานความปลอดภัยจะมีฟังก์ชันค่าการลงโทษเพื่อปรับปรุงค่าให้ประชากรยุคต่อไปวิวัฒนาการเข้าหาค่าคำตอบที่มีความปลอดภัยทางโครงสร้าง และการคำนวณเชิงพันธุกรรมสามารถนำไปพัฒนาต่อเป็นพื้นฐานของวิธีการออกแบบอย่างเหมาะสมได้อีกในอนาคต

2.2.2 วิธีการอัลกอริทึมหิ่งห้อย (Firefly Algorithm, FA)

ในปี ค.ศ. 2008 Xin-She Yang, นักคณิตศาสตร์และนักวิทยาศาสตร์คอมพิวเตอร์, ได้คิดค้นและสร้างวิธีการออกแบบเหมาะสมแบบเมตะฮิวริสติกขึ้นมาชื่อว่าอัลกอริทึมหิ่งห้อย โดยได้รับแรงบันดาลใจมาจาก

พฤติกรรมของหิ่งห้อยตามธรรมชาติ เขาสนใจในการพัฒนาเทคนิคการออกแบบอย่างเหมาะสมใหม่ขึ้นมาที่สามารถแก้ปัญหาที่มีความยากและซับซ้อน

อัลกอริทึมหิ่งห้อยมีพื้นฐานการทำงานที่จำลองพฤติกรรมการกระพริบแสงของตัวหิ่งห้อยออกมาเพื่อดึงดูดหิ่งห้อยตัวอื่น วิธีการนี้หิ่งห้อยแต่ละตัวเป็นตัวแทนของการแทนค่าตัวแปรในการหาคำตอบที่ดีที่สุดของปัญหาที่ต้องการ ความสว่างของหิ่งห้อยเปรียบเป็นคุณภาพของคำตอบที่ตัวมันมี หิ่งห้อยแต่ละตัวจะเคลื่อนตัวเข้าหาหิ่งห้อยตัวที่สว่างกว่าตนเอง และความน่าดึงดูดของหิ่งห้อยจะถูกนิยามโดยความสว่างและระยะห่างระหว่างตัวหิ่งห้อย วิธีการออกแบบนี้เป็นวิธีการแบบทำซ้ำเพื่ออัปเดตข้อมูลตำแหน่งของหิ่งห้อยแต่ละตัวซึ่งขึ้นอยู่กับความน่าดึงดูดของตัวมันเอง เพื่อที่จะหาคำตอบที่ดีที่สุดของโจทย์ปัญหานั้น โดยอัลกอริทึมนี้ได้ถูกแสดงศักยภาพของมันผ่านการแก้ปัญหาออกแบบอย่างเหมาะสมทั้งในด้านวิศวกรรม, การประมวลผลรูปภาพ และการเลือกคุณสมบัติ และได้มีการพัฒนาอัลกอริทึมต่ออีกหลากหลายรูปแบบเพื่อเพิ่มประสิทธิภาพของมัน

Siamak Talatahari, Amir Hossein Gandomi and Gun Jin Yun (2012), Optimum design of tower structures using Firefly Algorithm, บทความทางวิชาการนี้ทำการทดสอบอัลกอริทึมเมตะฮิวริสติกอัลกอริทึมหิ่งห้อย เพื่อใช้ออกแบบโครงสร้างหอสูงอย่างเหมาะสม ได้กล่าวไว้ว่าอัลกอริทึมนี้มีความสามารถในการหาคำตอบของปัญหาที่มีความยากและเป็นโครงสร้างขนาดใหญ่ โดยให้ข้อกำหนดว่า หิ่งห้อยแต่ละตัวเป็นสัตว์สองเพศสามารถถูกดึงดูดโดยหิ่งห้อยตัวอื่นโดยไม่สนเพศ, ค่าความดึงดูดขึ้นอยู่กับความสว่างแต่จะลดลงตามระยะห่างระหว่างหิ่งห้อยที่ถูกดึงดูด และได้กล่าวว่าวิธีการนี้มีความคล้ายคลึงกับการใช้ GA

ในการตัวอย่างการทดลองหนึ่งได้กำหนดตัวแปรเป็นโครงข้อหมุนเหล็ก 25 ตัวและหน้าตัด 8 รูปแบบและทำการเปรียบเทียบผลลัพธ์การออกแบบอย่างเหมาะสมของอัลกอริทึมหิ่งห้อยและวิธีการอื่นได้ดังนี้

Table 4. Performance comparison for the 25-bar spatial truss.

Method		Optimal cross-sectional areas (in^2)									
		GA	PSO	SA	GP	ABC	CP	CSS	OC	BB–BC	FA
1	A_1	0.10	0.010	0.01	0.01	0.01	0.01	0.010	0.01	0.010	0.01
2	$A_{2–5}$	1.80	2.121	1.987	1.986	1.979	1.998	2.003	1.987	1.993	2.0432
3	$A_{6–9}$	2.30	2.893	2.994	2.961	3.003	2.983	3.007	2.994	3.056	3.0017
4	$A_{10–11}$	0.20	0.010	0.01	0.01	0.01	0.01	0.010	0.01	0.010	0.01
5	$A_{12–13}$	0.10	0.010	0.01	0.01	0.01	0.01	0.010	0.01	0.010	0.01
6	$A_{14–17}$	0.80	0.671	0.684	0.806	0.69	0.684	0.687	0.684	0.665	0.6831
7	$A_{18–21}$	1.80	1.611	1.677	1.68	1.679	1.675	1.655	1.677	1.642	1.6231
8	$A_{22–25}$	3.0	2.717	2.662	2.53	2.652	2.667	2.660	2.662	2.679	2.6725
Weight	(lb)	546	545.21	545.23	545.66	545.19	545.37	545.10	545.16	545.16	545.13
	(kg)	247.88	247.53	247.53	247.65	247.52	247.60	247.25	247.50	247.50	247.27

GA,Genetic Algorithm; PSO, Particle Swarm Optimizer; SA, Simulated Annealing; GP, Geometrical Programming; ABC, Artificial Bee Colony; CP, Center Points method; CSS, Charged System Search; OC, Optimality Criteria; BB–BC, Big Bang–Big Crush; FA, Firefly Algorithm.

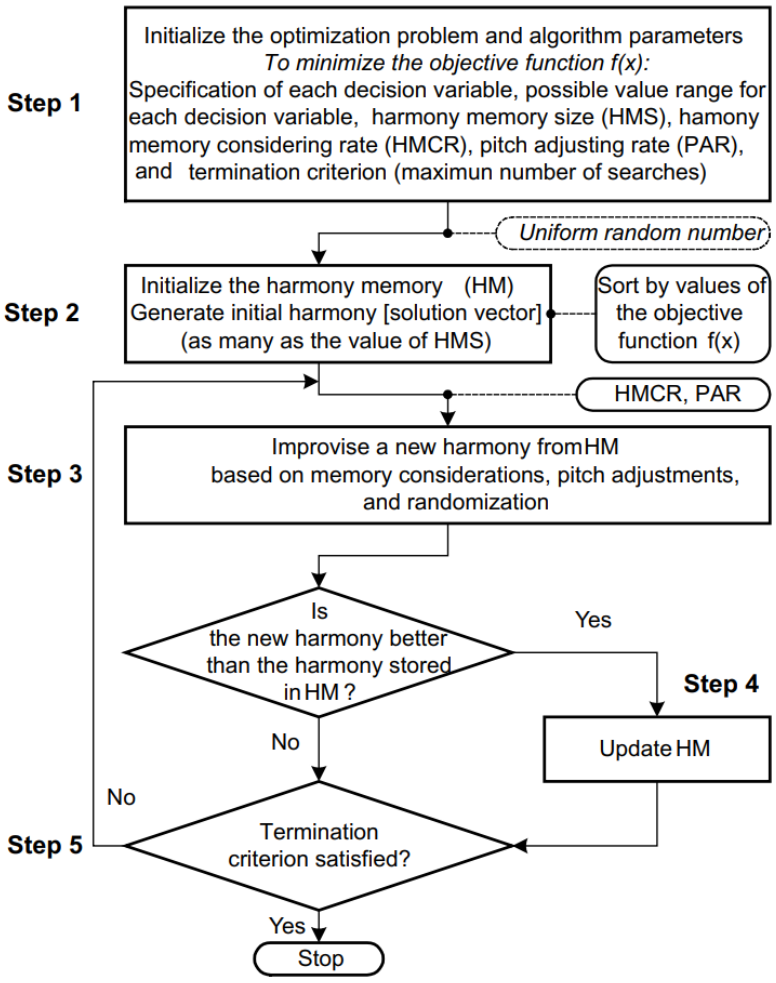
ตารางที่ 2-1 เปรียบเทียบผลลัพธ์การออกแบบอย่างเหมาะสมวิธีต่าง ๆ

เห็นได้ว่าแต่ละวิธีการถ้าหากมีตัวแปรจำนวนไม่มาก ผลลัพธ์ที่ได้นั้นมีค่าต่างกันไม่มาก

2.2.3 วิธีการอัลกอรทึมฮาร์โมนีเสิร์ช (Harmony Search Algorithm, HSA)

วิธีการนี้ถูกพัฒนาขึ้นโดย Zong Woo Geem วิศวกรโยธาและศาสตราจารย์ที่ University of Seoul เมื่อปี ค.ศ. 2001 เป็นกวิธีการเมตะฮิวริสติกที่ได้รับแรงบันดาลใจมาจากการด้นสดในการเล่นดนตรี โดยเขาได้สนใจและพัฒนาอัลกอริทึมนี้เพื่อใช้ในการแก้ปัญหาในการออกแบบของวิศวกร พื้นฐานของอัลกอริทึมนี้มาจากการด้นสดขณะเล่นดนตรี โดยที่นักดนตรีจะสร้างทำนองเพลงขึ้นมาใหม่ให้สามารถกลมกลืนกับท่อนเพลงอื่น ๆ ในเพลงได้ โดยอัลกอริทึมนี้ ทำนองเพลงแต่ละท่อนเป็นตัวแทนของคำตอบในการออกแบบอย่างเหมาะสม ซึ่งคุณภาพของคำตอบจะวัดจาก objective function จะมีประชากรเริ่มต้นเป็นทำนองเพลงแบบสุ่มและทำกระบวนการทำซ้ำเพื่ออัปเดตประชากรโดยสร้างทำนองเพลงใหม่ขึ้นมาผ่านการผสมผสานระหว่างการสำรวจและหาช่องทาง วิธีการนี้ถูกใช้ในการแก้ปัญหาออกแบบอย่างเหมาะสมทั้งในด้านวิศวกรรม, การจัดตารางเวลา และด้านการเงิน อีกทั้งยังถูกพัฒนาเพื่อเพิ่มประสิทธิภาพให้ดียิ่งขึ้น

Kang Seok Lee, Zong Woo Geem (2004), A new structural optimization method based on the harmony search algorithm บทความทางวิชาการจากผู้คิดค้นอัลกอริทึมนี้ กล่าวว่า อัลกอริทึมนี้ถูกพัฒนามาจากวิธีการออกแบบอย่างเหมาะสมเมตะฮิวริสติกอื่น โดยมีพื้นฐานมาจากการแสดงดนตรี เช่นการแสดงแบบด้นสดของเพลงแจ๊ส การกระทำนี้จะหาทำนองเพลงที่สามารถเข้ากันได้โดยขึ้นอยู่กับมาตรฐานทางดนตรี และเรียกคำตอบที่ดีที่สุดว่า perfect state ซึ่งจะถูกพิจารณาจาก objective function และในการทดลองนี้ได้ทำการทดสอบกับโครงสร้างโครงรข้อหมุนเหล็กในรูปแบบต่าง ๆ ด้วยขั้นตอนดังนี้



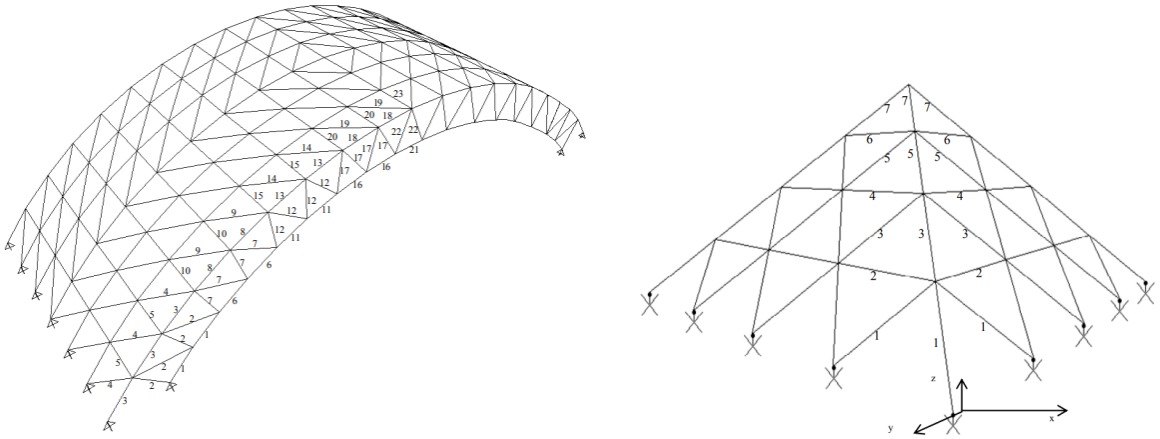
รูปที่ 2-4 แผนภาพขั้นตอนการทำงานของอัลกอริทึมฮาร์โมนีเสิร์ช

2.2.4 วิธีการระบบอาณานิคมมด (Ant Colony Optimization, ACO)

วิธีการนี้ถูกพัฒนาขึ้นโดย Marco Dorigo, นักวิทยาศาสตร์คอมพิวเตอร์, ในวิทยานิพนธ์ระดับปริญญาเอกของเขาในปี ค.ศ. 1992 โดยวิธีการนี้ได้รับแรงบันดาลใจมาจากพฤติกรรมการหาอาหารของมด เป็นวิธีการที่พัฒนาขึ้นมาเพื่อใช้แก้ปัญหาที่ซับซ้อนและผสมผสาน เขาได้สังเกตว่ามดสามารถหาเส้นทางที่สั้นที่สุดระหว่างรังของมันกับแหล่งอาหารได้โดยพวกมันจะทิ้งฟีโรโมนของมันไว้เป็นทางเพื่อเป็นการบ่งบอกให้มดตัวอื่นรู้ จากการสังเกตของเขาทำให้เขานำไปใช้ในการแก้ปัญหาการออกแบบอย่างเหมาะสม ในวิธีการนี้จะมีการสร้างกลุ่มมดจำลองขึ้นมาเพื่อหาคำตอบโดยการเดินผ่านเส้นทางในพื้นที่ค้นหาคำตอบนั้น ๆ มดพวกนี้จะทิ้งร่องรอยฟีโรโมนไว้เพื่อระบุเส้นทางของพวกมัน และฟีโรโมนเหล่านี้จะระเหยตามกาลเวลา มดเหล่านี้มีโอกาสที่จะเลือกเส้นทางที่มีฟี

โรโมนสูงกว่า ซึ่งเป็นเส้นทางที่แสดงถึงคำจอบที่ดีกว่า อัลกอริทึมนี้จะทำซ้ำไปเรื่อย ๆ จนกว่าจะได้คำตอบที่พึงพอใจ

O. Hasançeb and S. Çarbaş (2011), Ant colony search method in practical structural optimization บทความทางวิชาการนี้เป็นการนำ ACO มาใช้ในการออกแบบหน้าตัดเหมาะสมของโครงสร้างโครงข้อหมุนเหล็ก เพื่อทำการหาน้ำหนักเหล็กที่น้อยที่สุดที่ใช้ในการออกแบบโครงสร้าง โดยรูปร่างโครงสร้างดังนี้



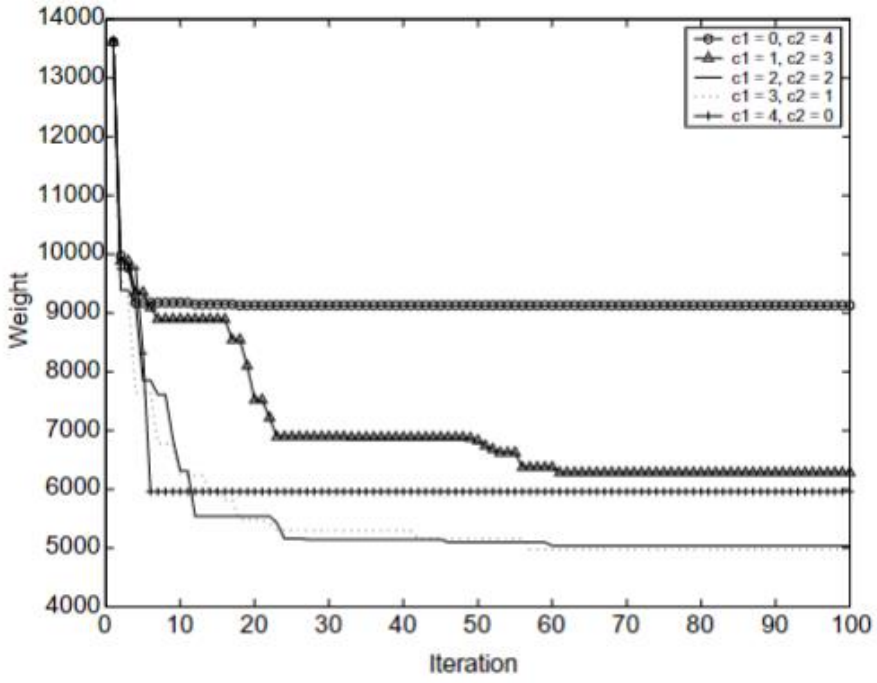
รูปที่ 2-5 ตัวอย่างโครงสร้างโครงข้อหมุนเหล็กที่ใช้ทดสอบ ACO

2.2.5 วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO)

วิธีความฉลาดแบบกลุ่ม เป็นวิธีการแบบเมตะฮิวริสติกที่ถูกสร้างโดย Eberhert and Kennedy ในปี ค.ศ. 1995 โดยได้รับแรงบันดาลใจมาจากการบินเป็นฝูงของนก และการว่ายเป็นฝูงของปลา พวกเขามีความสนใจในการพัฒนาวิธีการออกแบบเหมาสม โดยพวกเขาสังเกตพฤติกรรมของปลาและนกในเรื่องของการเดินทางเป็นฝูงแบบที่มีความสัมพันธ์ระหว่างประชากรในฝูง และนำมาปรับใช้ในการแก้ปัญหาการออกแบบอย่างเหมาะสม

พื้นฐานของวิธีความฉลาดแบบกลุ่มนั้นเป็นการจำลองพฤติกรรมของฝูงอนุภาคที่เคลื่อนตัวผ่านพื้นที่ค้นหา โดยอนุภาคแต่ละตัวเป็นตัวแทนของคำตอบของปัญหา อนุภาคเหล่านี้จะเคลื่อนตัวโดยมีอิทธิพลมาจากตำแหน่งที่ดีที่สุดของตนเอง และตำแหน่งที่ดีที่สุดของสมาชิกในฝูง ซึ่งจะเป็นวิธีการทำซ้ำไปเรื่อย ๆ เพื่อหาคำตอบที่พึงพอใจ โดยตั้งแต่วิธีการนี้ได้ถูกคิดค้นขึ้นมาก็มีการนำไปประยุกต์ใช้กับปัญหามากมายทั้งด้านวิศวกรรม, การทำนายสถานะทางการเงิน เป็นต้น นอกจากนี้วิธีการนี้ยังถูกนำไปพัฒนาต่อเพื่อใช้หาคำตอบได้อย่างมีประสิทธิภาพมากขึ้น เช่น Hybrid particle swarm optimization (HPSO), Adaptive particle swarm optimization (APSO) และ Multi-objective particle swarm optimization (MOPSO) เป็นต้น

R.E. Perez, K. Behdinan (2007), Particle swarm approach for structural optimization บทความทางวิชาการนี้ได้นำวิธีความฉลาดแบบกลุ่มมาพัฒนา ปรับค่าพารามิเตอร์ต่าง ๆ เพื่อค้นหาความแตกต่างของผลลัพธ์ และเทียบกับวิธีการออกแบบเหมาะสมแบบดั้งเดิม โดยทดสอบกับโครงสร้างโครงข้อหมุนเหล็ก โดยจากการทดลองพบว่า พารามิเตอร์แต่ละตัวให้ค่าผลลัพธ์ที่แตกต่างกันทำให้เส้นทางการเคลื่อนตัวของอนุภาคเปลี่ยนไป และจำนวนครั้งของการทำซ้ำของโปรแกรมมีผลกับคำตอบที่ดีที่สุดดังภาพ



รูปที่ 2-6 แผนภูมิแสดงผลลัพธ์จากพารามิเตอร์ที่ต่างกัน

โดยจากตัวอย่างตัวแปร c1 คือน้ำหนักของเส้นทางที่ดีที่สุดของอนุภาค และ c2 น้ำหนักของเส้นทางที่ดีที่สุดของกลุ่มอนุภาค

บทที่ 3
ทฤษฎีและขั้นตอนการดำเนินงาน

การออกแบบอย่างเหมาะสมจะมีหลักการพื้นฐานคือ การมีฟังก์ชันวัตถุประสงค์, ตัวแปรการออกแบบ และตัวแปรสถานะ ซึ่งตัวแปรเหล่านี้เป็นตัวกำหนดขั้นตอนการดำเนินการของฟังก์ชันเพื่อหาผลลัพธ์ของการออกแบบอย่างเหมาะสม สามารถเขียนเป็นสมการทางคณิตศาสตร์ได้ดังนี้

Minimize $f(x)$ (3.1)

Subject to $g_j(x) \leq 0 \quad j = 1, 2, 3, \ldots, J$ (3.2)

$h_k(x) = 0 \quad k = 1, 2, 3, \ldots, K$ (3.3)

And $x \in R^n : x^{min}{}_i \leq x_i \leq x^{max}{}_i, i = 1, 2, 3 \ldots, n$

โดยที่ $f(x)$ คือ ฟังก์ชันวัตถุประสงค์ (Objective function)

$g_i(x)$ คือ ฟังก์ชันข้อจำกัดที่ไม่เท่ากัน (Equality constrain function)

$h_k(x)$ คือ ฟังก์ชันข้อจำกัดที่เท่ากัน (Inequality constrain function)

x_i คือ ตัวแปรออกแบบ (Design Variable)

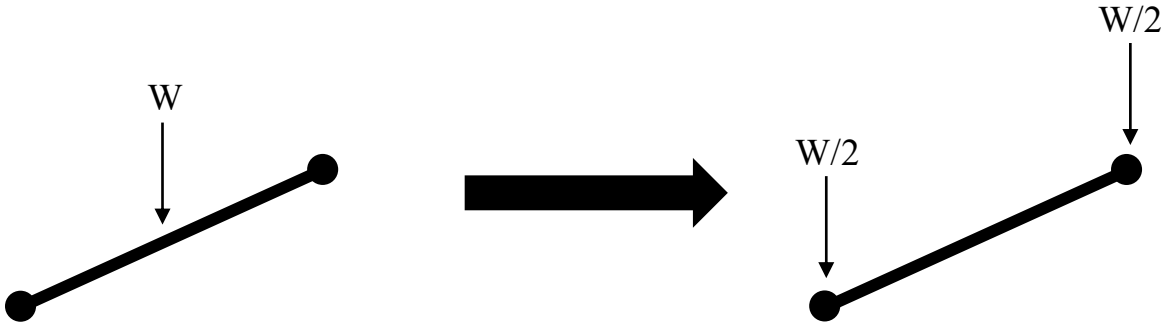
R^n คือ พื้นที่ค้นหา (Search Space)

เมื่อทำการคำนวณตามสมการนี้จะเป็นวิธีการที่นำไปสู่การหาคำตอบที่ดีที่สุดของการออกแบบอย่างเหมาสมได้

3.1 การออกแบบโครงข้อหมุนเหล็กตามมาตรฐาน

การออกแบบโครงข้อหมุนเหล็กในทางทฤษฎีจะเป็นการคิดแรงภายในของชิ้นส่วนโครงข้อหมุนเหล็ก โดยไม่คำนวณโมเมนต์ที่เกิดขึ้นต่อชิ้นส่วนนั้นจึงทำให้ต้องพิจารณาแรงภายในทั้งสองรูปแบบคือ แรงดึง และแรงอัด โดยในโครงงานนี้จะใช้วิธีการออกแบบ LRFD ตามมาตรฐาน AISC 360-16 โดยในกรณีของโครงงานนี้ จะคิดแรง

กระทำจากน้ำหนักของตัวโครงสร้างโดยการถ่ายโอนน้ำหนักไปที่พิกัดจุดต่อของโครงข้อหมุนเหล็กชิ้นส่วนนั้นแทนการกระจายเป็น uniform load หรือ point load ที่จุดศูนย์กลางมวล



รูปที่ 3-1 วิธีการคำนวณน้ำหนักของโครงสร้าง

3.1.1 วิธีตัวคูณความต้านทานและน้ำหนักบรรทุก (Load and Resistance Factor Design, LRFD)

การออกแบบโครงสร้างเหล็กด้วยวิธี LRFD จะออกแบบใช้สภาวะจำกัดเป็นเกณฑ์ ภายใต้น้ำหนักบรรทุกปรับค่าแรงที่คำนวณได้จากการวิเคราะห์ จะต้องมีค่าน้อยกว่าหรือเท่ากับค่าความต้านทานระบุชิ้นส่วนของโครงสร้างคูณด้วยความต้านทาน สามารถเขียนเป็นสมการได้ ดังนี้

$$R_u = \sum \gamma_i P_i \leq \phi R_n \qquad (3.4)$$

โดยที่ R_u คือ แรงต่าง ๆ เนื่องจากน้ำหนักบรรทุกปรับค่า เช่น แรงดึง แรงอัด

P_i คือ แรงต่าง ๆ จากน้ำหนักบรรทุกใช้งาน

γ_i คือ ตัวคูณน้ำหนักบรรทุก (Load Factor)

ϕ คือ ตัวคูณความต้านทาน (Resistance Factor)

R_n คือ ความต้านทานระบุ (Nominal Resistance)

ค่าแรงต่าง ๆ เนื่องจากน้ำหนักบรรทุกปรับค่า (P_u) เป็นค่าที่มากที่สุดที่ได้จากการรวมน้ำหนักบรรทุกประเภทต่าง ๆ ดังนี้

1.4D

$1.2D + 1.6L + 0.5L_r$

$1.2D + 1.6L_r + (f_1L \text{ or } 0.8W)$

$1.2D + 1.6W + 0.5L_r$

$1.2D \pm 1.0E + f_1L$

$0.9D \pm (1.6W \text{ or } 1.0E)$

โดยที่ D คือ น้ำหนักบรรทุกคงที่

L คือ น้ำหนักบรรทุกจร

L_r คือ น้ำหนักบรรทุกจรชั้นหลังคา

W คือ แรงลม

E คือ แรงแผ่นดินไหว

f_1 มีค่าเป็น 1.0 สำหรับอาคารจอดรถ พื้นที่ห้องโถงสาธารณประโยชน์ และพื้นที่ที่มีน้ำหนักบรรทุกจรมากกว่า 500 กิโลกรรม/ตารางเมตร นอกนั้นมีค่าเป็น 0.5

3.1.2 การออกแบบโครงสร้างเหล็กสำหรับรับแรงดึง (Tensile Strength Design)

การออกแบบกำลังการรับแรงดึง, ϕP_n, ชิ้นส่วนของโครงสร้างต้องมีค่าแรงดึงภายในต่ำกว่าค่าความแข็งแรงในการรับแรงดึง ซึ่งถูกกำหนดด้วยแรงคราก(จากการดึง (Tensile Yielding) ในพื้นที่รับแรงทั้งหมด (gross section) และแรงดึงประลัย (Tensile Rupture) ในพื้นที่รับแรงสุทธิ (net section)

สำหรับหน่วยแรงครากในพื้นที่รับแรงทั้งหมด

$$P_n = F_y A_g \qquad (3.5)$$

$$\phi_t = 0.90 \ (LRFD)$$

สำหรับแรงดึงประลัยในพื้นที่รับแรงสุทธิ

$$P_n = F_u A_e \qquad (3.6)$$

$$\phi_t = 0.75 \ (LRFD)$$

โดยที่ A_e คือ พื้นที่รับแรงสุทธิประสิทธิผล

A_g คือ พื้นที่รับแรงทั้งหมดของชิ้นส่วนโครงสร้าง

F_y คือ หน่วยแรงคราก

F_u คือ กำลังดึงประลัย

ในโครงงานนี้กำหนดให้ F_y = 235 MPa และ F_u = 360 MPa

โดยกำลังรับแรงดึงของโครงสร้างคือ ค่าที่น้อยที่สุดที่คำนวณได้ระหว่างแรงดึงประลัย และแรงดึงคราก

3.1.3 การออกแบบโครงสร้างเหล็กสำหรับรับแรงอัด (Compression Strength Design)

ความยาวประสิทธิผล (Effective Length, L_c) ใช้ในการคำนวณความชะลูดของชิ้นส่วนโครงสร้าง (L_c/r) โดยสามารถพิจารณาได้ตามตาราง 3-2

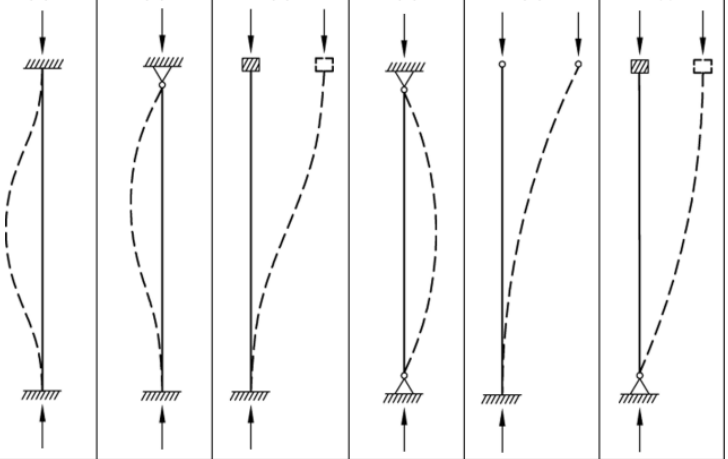
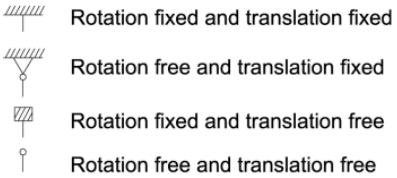
โดยที่ K คือ ตัวคูณความยาวประสิทธิผล

$L_c = KL$ คือ ความยาวประสิทธิผล

L คือ ความยาว

r คือ รัศมีไจเรชัน

TABLE C-A-7.1
Approximate Values of Effective Length Factor, *K*

	(a)	(b)	(c)	(d)	(e)	(f)
Buckled shape of column is shown by dashed line						
Theoretical *K* value	0.5	0.7	1.0	1.0	2.0	2.0
Recommended design value when ideal conditions are approximated	0.65	0.80	1.2	1.0	2.1	2.0
End condition code	 Rotation fixed and translation fixed Rotation free and translation fixed Rotation fixed and translation free Rotation free and translation free					

ตารางที่ 3-1 ค่าประมาณตัวคูณความยาวประสิทธิผล (K)

ในกรณีโครงสร้างโครงข้อหมุนเหล็กสมมติให้ปลายทั้งสองข้างของชิ้นส่วนโครงสร้างเป็นข้อหมุน (pinned support) ทำให้ในโครงงานนี้ใช้ค่า K มีค่าเท่ากับ 1.0 ในการคำนวณแรงภายอัดภายในโครงสร้าง

โดยการวิบัติของโครงสร้างเหล็กจากแรงอัดจะวิบัติเนื่องจากแรงกดทำให้เกิดการโก่งเดาะ (buckling) ขึ้นที่โครงสร้างแต่สามารถคำนวณการวิบัติได้จากแรงอัดที่เกิดขึ้นในโครงสร้างนั้น โดยกำลังแรงอัดระบุ (P_n) สามารถคำนวณได้ดังนี้

$$P_n = F_{cr} A_g \tag{3.6}$$

โดยที่ หน่วยแรงอัดวิกฤต (F_{cr}) สามารถคำนวณได้จาก

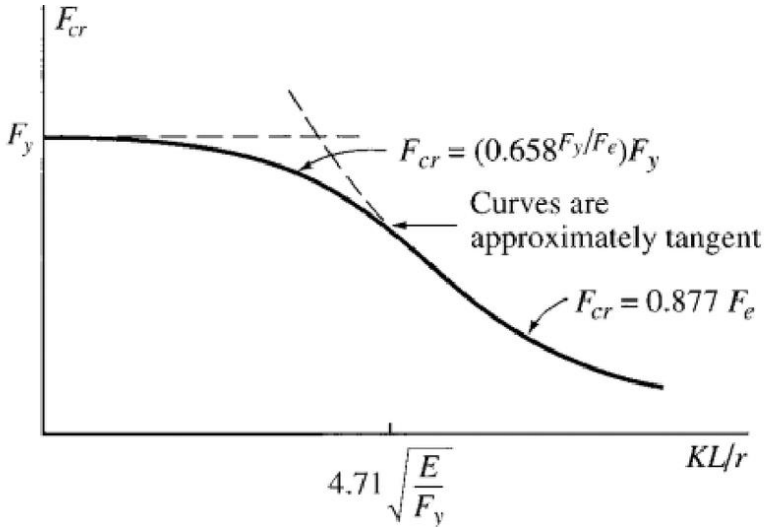
เมื่อ $\frac{L_c}{r} \leq 4.71\sqrt{\frac{E}{F_y}}$ *or* $\frac{F_y}{F_e} \leq 2.25$ **Inelastic Buckling**

$$F_{cr} = 0.658^{\frac{F_y}{F_e}} F_y \quad (3.7)$$

เมื่อ $\frac{L_c}{r} \geq 4.71\sqrt{\frac{E}{F_y}}$ *or* $\frac{F_y}{F_e} \geq 2.25$ **Elastic Buckling**

$$F_{cr} = 0.877F_e \quad (3.8)$$

โดยที่ $$F_e = \pi^2 E(KL/r)^2$$



รูปที่ 3-2 แผนภูมแสดงพฤติกรรมมของค่าหน่วยแรงอัดออยเลอร์ (F_e)

โดยที่ A_g คือ พื้นที่หน้าตัดทั้งหมด

E คือ โมดูลัสยืดหยุ่นของเหล็ก

F_e คือ หน่วยแรงอัดออยเลอร์

F_y คือ หน่วยแรงคราก

r คือ รัศมีไจเรชัน

โดยในโครงงานนี้กำหนดให้ E = 200 GPa

การคำนวณค่ากำลังรับแรงอัดแบบ LRFD นั้น ต้องคูณกับตัวคูณปรับลดกำลัง

โดยที่ $\phi_c = 0.90\ (LRFD)$

ได้ว่า $\phi_c P_n$ คือ กำลังรับแรงอัด

ในการคำนวณแรงอัดนั้นตามมาตรฐาน AISC 360-16 ต้องคำนวณผลกระทบจากการบิดของชิ้นส่วนโครงสร้างเนื่องจากความไม่สมมาตรของหน้าตัด แต่ในโครงงานนี้ได้เลือกใช้หน้าตัดเหล็กท่อกลม ซึ่งมีความสมมาตรในทุกแกนจึงไม่นำผลกระทบจากแรงบิดมาใช้ในการคำนวณความปลอดภัยจากการรับน้ำหนักบรรทุกของโครงสร้างโครงข้อหมุนเหล็ก

อีกทั้งโครงข้อหมุนเหล็กในอุดมคติยังไม่มีการรับแรงจากด้านข้าง ทำให้ไม่เกิดแรงจากการดัดและแรงเฉือนกับชิ้นส่วนโครงสร้าง ทำให้ในโครงงานนี้ไม่คำนวณการวิบัติจากกรณีทั้งสองที่ได้กล่าวมา

3.2 การออกแบบอย่างเหมาะสม

3.2.1 การออกแบบหน้าตัดอย่างเหมาะสม

ในกรณีที่ปัญหาโครงสร้างต้องการหาคำตอบเป็นหน้าตัดที่เหมาะสม ตัวแปรออกแบบจะสื่อถึงคุณสมบัติในรูปแบบใดรูปแบบหนึ่งของหน้าตัดชิ้นส่วนของโครงสร้าง ในกรณีออกแบบโครงข้อหมุนเหล็กอย่างเหมาะสมในโครงงานนี้จะกำหนดตัวแปรนี้เป็นหน้าตัดเหล็กท่อกลมตามมาตรฐาน มอก. ของประเทศไทยเพื่อใช้ในการออกแบบอย่างเหมาะสมตามปัญหาที่ต้องการตามตาราง 3-1

ขนาด	เส้นผ่าศูนย์กลางภายนอก (D)	น้ำหนัก	ความหนา (t)	พื้นที่หน้าตัด (A)	โมเมนต์ความเฉื่อย (I)	โมดูลัสหน้าตัด (S)	โมดูลัสพลาสติก (Z)	r	D/t	ค่าคงที่การบิด (C_w)
	มม.	กก./มม.	ซม.	ตร.ซม.	ซม.^{4}	ซม.^{3}	ซม.^{3}	ซม.		ซม.
15	21.7	0.972	2.0	1.24	0.81	0.56	0.71	0.70	10.85	1.22
20	27.2	1.41	2.3	1.80	1.41	1.03	1.31	0.88	11.83	2.24
25	34	1.8	2.3	2.29	2.89	1.70	2.16	1.12	14.78	3.63
32	42.7	2.29	2.3	2.92	5.97	2.80	3.55	1.43	18.57	5.90
40	48.6	2.63	2.3	3.35	8.99	3.70	4.70	1.64	21.13	7.74
		3.58	3.2	4.56	11.8	4.86	6.18	1.61	15.19	10.4
50	60.5	4.52	3.2	5.76	23.7	7.84	9.96	2.03	18.91	16.5
		5.57	4.0	7.10	28.5	9.41	12.0	2.00	15.13	20.1
65	76.3	5.77	3.2	7.35	49.2	12.9	16.4	2.59	23.84	26.9
		7.13	4.0	9.09	59.5	15.6	19.8	2.56	19.08	32.8
80	89.1	6.78	3.2	8.64	79.8	17.9	22.7	3.04	27.84	37.0
		8.39	4.0	10.69	97	21.8	27.7	3.01	22.28	45.5
90	101.6	7.75	3.2	9.89	120	23.6	30.0	3.48	31.75	48.7
		9.63	4.0	12.26	146	28.8	36.6	3.45	25.40	59.9
100	114.3	8.77	3.2	11.17	172	30.2	38.3	3.93	35.72	62.0
		12.2	4.5	15.52	234	41	52.1	3.89	25.40	85.2
		15.0	5.6	19.12	283	49.6	62.9	3.85	20.41	104
125	139.8	15.0	4.5	19.13	438	62.7	79.6	4.79	31.07	129
		19.8	6.0	25.22	566	80.9	103	4.74	23.30	169
150	165.2	17.8	4.5	22.72	734	88.9	113	5.68	36.71	183
		23.6	6.0	30.01	952	115	146	5.63	27.53	239
175	190.7	22.9	5.0	29.17	1258	132	188	6.57	38.14	271
		31.7	7.0	40.4	1707	179	227	6.50	27.24	371
200	216.3	31.1	6.0	39.64	2193	203	258	7.44	36.05	417
		41.1	8.0	52.35	2844	263	334	7.37	27.04	545
250	267.4	38.7	6.0	49.27	4211	315	400	9.24	44.57	644
		51.2	8.0	65.19	5489	411	521	9.18	33.43	846

ตารางที่ 3-2 ตารางคุณสมบัติหน้าตัดเหล็กท่อกลมมาตรฐาน มอก.

3.2.2 ฟังก์ชันวัตถุประสงค์

ฟังก์ชันวัตถุประสงค์ในการออกแบบเหมาะสมโครงข้อหมุนเหล็กในโครงงานนี้เป็นการออกแบบเหมาะสมโดยคิดจากมวลรวมของโครงสร้างและพยายามให้ค่าฟังก์ชันวัตถุประสงค์ได้ค่าออกมาน้อยที่สุดเท่านที่เป็นไปได้ โดยโครงสร้างสามารถรับน้ำหนักตามที่กำหนดได้ตามมาตรฐาน AISC 360-16 สามารถเขียนเป็นสมการได้ดังนี้

Minimize $$m = \rho_{steel} \sum_{i=1}^{n} A_i L_i \qquad (3.9)$$

โดยที่ m คือ มวลรวมของโครงสร้างโครงข้อหมุนเหล็ก (กิโลกรัม)

ρ_{steel} คือ ความหนาแน่นของเหล็ก (กิโลกรัม/ลูกบาศก์เมตร)

A_i คือ พื้นที่หน้าตัดของโครงข้อหมุนชิ้นที่ i (ตารางเมตร)

L_i คือ ความยาวของโครงข้อหมุนชิ้นที่ i (เมตร)

โดยในโครงงานนี้กำหนดให้ ρ_{steel} = 7860 กิโลกรัม/ลูกบาศก์เมตร และทำการคิดน้ำหนักจากการออกแบบอย่างเหมาะสมออกมาเป็นห่วย กิโลกรัม เพื่อเปรียบเทียบค่าน้ำหนักระหว่างการคำนวณฟังก์ชันด้วยตัวแปรออกแบบต่าง ๆ ในการออกแบบเหมาะสมด้วยหน้าตัดเหล็กที่มีคุณสมบัติชัดเจนตามมาตรฐานนั้น ตัวแปรออกแบบ (x) ซึ่งตามฟังก์ชันวัตถุประสงค์คือพื้นที่หน้าตัด นั้นจะมีค่าที่ชัดเจนทำให้พื้นที่การค้นหาคำตอบแคบลง และทำให้การหาคำตอบการออกแบบอย่างเหมาะสมที่ดีที่สุดเป็นไปได้ง่ายและสามารถนำไปใช้กับการออกแบบในสภาพงานจริงได้ดียิ่งขึ้น

3.2.3 ตัวแปรออกแบบ

ตัวแปรออกแบบคือ ชุดข้อมูลที่ใช้ในการแทนค่าในฟังก์ชันวัตถุประสงค์ เพื่อใช้คำนวณผลลัพธ์ตามวัตถุประสงค์ที่ต้องการ โดยตัวแปรออกแบบจะต้องมีขอบเขตของค่าที่นำไปใช้ในกระบวนการคำนวณในการออกแบบโครงสร้างโครงข้อหมุนเหล็กในโครงงานนี้จะใช้คุณสมบัติหน้าตัดในการออกแบบตามตารางที่ 3-2 โดยที่จะเก็บข้อมูลคุณสมบัติของหน้าตัดเหล็กทุกตัวในตารางเพื่อนำข้อมูลที่มี ดึงไปใช้ในการคำนวณการออกแบบอย่างเหมาะสมโดยมีขอบเขตการออกแบบที่ชัดเจน

3.2.4 ฟังก์ชันข้อจำกัด

การออกแบบโครงสร้างที่มีความปลอดภัยต้องมีข้อจำกัดในการออกแบบ โดยฟังก์ชันข้อจำกัดจะเป็นตัวบ่งบอกว่า ผลลัพธ์ที่ได้ออกมามีความถูกต้องหรือไม่ ซึ่งในการออกแบบโครงสร้างโครงข้อหมุนเหล็กนั้นมีข้อจำกัดคือ เหล็กที่นำมาใช้ออกแบบต้องสามารถรับน้ำหนักบรรทุกตามโจทย์ปัญหาที่กำหนดได้อย่างปลอดภัยตามมาตรฐานการออกแบบ โดยความสามารถในการรับแรงต้องมีความแข็งแรงพอในการรับแรงที่กำหนด ซึ่งเป็นไปตามสมการ (3.4)

ถ้าหากแบ่งกรณีเป็นกรณีโครงสร้างรับแรงดึงและแรงอัด ได้ว่า

ในกรณีโครงสร้างโครงข้อหมุนเหล็กรับแรงดึง

$$\phi_t P_n \geq T_u \tag{3.10}$$

เมื่อ T_u คือ แรงดึงจากน้ำหนักบรรทุกปรับค่า

P_n คือ กำลังรับแรงดึงระบุ

ในกรณีโครงสร้างโครงข้อหมุนเหล็กรับแรงอัด

$$\phi_c P_n \geq P_u \tag{3.11}$$

เมื่อ P_u คือ แรงดึงจากน้ำหนักบรรทุกปรับค่า

P_n คือ กำลังรับแรงอัดระบุ

ในทั้งสองกรณี ถ้าหากสมการ (3.10) และ (3.11) เป็นจริง แสดงว่าโครงสร้างสามารถรับน้ำหนักบรรทุกได้อย่างปลอดภัย แต่ถ้าหากมีสมการตัวใดตัวหนึ่งไม่เป็นจริงแสดงว่า ผลลัพธ์ที่ได้จากฟังก์ชันวัตถุประสงค์นั้นไม่ถูกต้องต้องทำการปรับแก้ค่าในการคำนวณต่อไป

3.3 วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO)

วิธีความฉลาดแบบกลุ่มเป็นวิธีการออกแบบอย่างเหมาะสมที่เป็นแบบ stochastic ที่มีพื้นฐานมาจากการเคลื่อนตัวแบบเป็นงฝูง ลอกเลียนแบบพฤติกรรมของฝูงนกและฝูงปลา ประชากรแต่ละตัวในฝูงจะมีการเคลื่อนตัวที่เปลี่ยนแปลงรูปแบบไปเรื่อย ๆ ตามประสบการณ์ของประชากรตัวนั้น โดยมีพื้นฐานการสร้างระบบจำลองฝูงประชากรเหล่านี้ให้เกิดประสิทธิภาพ โดยมีหลักการดังนี้

1. ความใกล้ชิด: กลุ่มประชากรที่สร้างขึ้นมาต้องมีความสามารถในการกระจายตัวในพื้นที่ค้นหาที่เหมาะสมในระยะเวลาที่ใช้คำนวณ
2. คุณภาพ: กลุ่มประชากรควรมีความสามารถในการสัมผัสถึงความเปลี่ยนแปลงทางคุณภาพของสภาพแวดล้อมและตอบสนองอย่างถูกต้อง
3. ความหลากหลายในการตอบสนอง: กลุ่มประชากรไม่ควรจำกัดเส้นทางของตนในการค้นหาคำตอบในพื้นที่ที่เล็กเกินไป
4. ความเสถียร: กลุ่มประชากรไม่ควรเปลี่ยนวิธีการคิดในทุก ๆ สภาพแวดล้อมที่เปลี่ยนไป
5. การปรับตัว: กลุ่มประชากรควรเปลี่ยนวิธีการคิดเมื่อการเปลี่ยนแปลงนั้นมีคุณค่ามากพอ

โดยข้อ 4. และ ข้อ 5. นั้นมีทั้งข้อดีและข้อเสียเป็นของตนเอง ทั้ง 5 หลักการนี้เป็นคุณลักษณะหลักของการจำลองพฤติกรรม และเป็นหลักการที่ทำให้เกิดการจำลองอนุภาคกลุ่มขึ้นมา ในวิธีความฉลาดแบบกลุ่มนั้น อนุภาคแต่ละตัวจะทำการอัปเดตตำแหน่งและความเร็วของตนเองขึ้นอยู่กับสภาพแวดล้อมที่เปลี่ยนไป และในวิธีการนี้จะไม่จำกัดอิสระภาพในการเคลื่อนไหวของอนุภาค แต่จำทำการเคลื่อนที่หาคำตอบอย่างต่อเนื่อง อนุภาคในกลุ่มจะทำการเคลื่อนไหวอย่างต่อเนื่องในพื้นที่ค้นหาคำตอบ ในขณะที่วิธีการเคลื่อนที่สามารถเปลี่ยนแปลงได้โดยปรับตัวเข้าหาสภาพแวดล้อม ทำให้วิธีความฉลาดแบบกลุ่มเป็นไปตามหลักการทั้ง 5 ข้างต้น

3.3.1 หลักการการทำงานของวิธีความฉลาดแบบกลุ่ม

ในขั้นตอนแรกต้องทำการสมมติให้ตัวแปรออกแบบ x อยู่ในตำแหน่งเริ่มต้นในพื้นที่ค้นหา และมี v เป็นเวกเตอร์ความเร็วในการเคลื่อนที่ ระยะทางระหว่างตำแหน่งของอนุภาคกับเป้าหมายจะถูกใช้ในการวัดคุณภาพของผลลัพธ์ของอนุภาคแต่ละตัว โดยในกรณีการออกแบบอย่างเหมาะสมกล่าวคือ มวลรวมโครงสร้างยิ่งมีค่าน้อยแสดงว่า ผลลัพธ์มีค่าที่ดี ในทางตรงข้ามกันยิ่งมวลรวมมากผลลัพธ์ยิ่งมีค่าแย่ โดยให้อนุภาคแต่ละตัวมีความสามารถในการจดจำตำแหน่งที่ดีที่สุดของมันได้ เราเรียกมันว่า personal best, $pbest$ โดยตัวแปรความเร็ว

ในการเคลื่อนที่จะเปลี่ยนแปลงไปเรื่อย ๆ และจะมีตัวแปร r_1 ซึ่งจะให้ค่าเป็นเลขสุ่มระหว่าง 0-1 และ c_1 เป็นค่าคงตัวน้ำหนักของเส้นทางที่ตามความทรงจำของอนุภาค การเปลี่ยนแปลงของตัวแปรความเร็วจากความทรงจำส่วนตัวนี้ว่าเรียกว่า “Cognitive Velocity” โดยสามารถอธิบายออกมาเป็นสมการได้ว่า

$$cognitive\ velocity = c_1 r_1 (pbest - x) \tag{3.12}$$

และถ้าหากเพิ่มสมมติฐานเข้าไปว่า อนุภาคแต่ละตัวสามารถสื่อสารกันได้ โดยอนุภาคแต่ละตัวสามารถจำตำแหน่งที่ดีที่สุดของสมาชิกในฝูงได้ และเรียกมันว่า global best, $gbest$ และให้ตัวแปร c_2 เป็นค่าคงตัวน้ำหนักของเส้นทางตามความทรงจำของฝูง และให้ r_2 เป็นเลขที่สุ่มระหว่าง 0-1 แยกกับ r_1 การเปลี่ยนแปลงของความเร็วจากความทรงจำกลุ่มนี้เรียกว่า “Social Velocity” เขียนเป็นสมการได้ว่า

$$social\ velocity = c_2 r_2 (gbest - x) \tag{3.13}$$

หากทำการรวมอิทธิพลระหว่างความทรงจำของอนุภาคแต่ละตัวกับความทรงจำแบบกลุ่ม เราสามารถสร้างสมการความเร็วที่เปลี่ยนไปในการเดินทางของอนุภาคได้ว่า

$$v = c_1 r_1 (pbest - x) + c_2 r_2 (gbest - x) \tag{3.14}$$

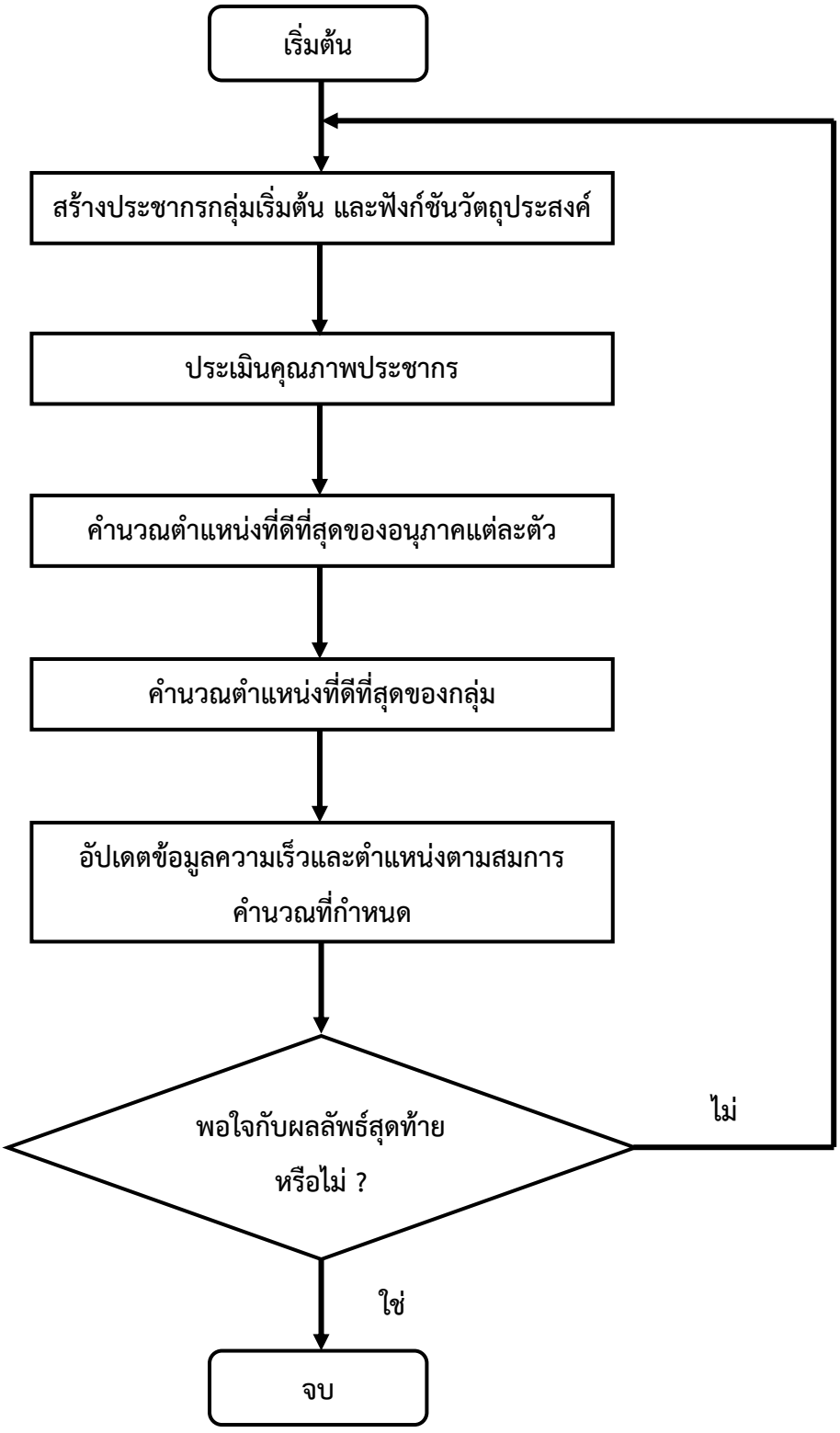
หากให้สมมติฐานว่าอนุภาคแต่ละตัวไม่มีมวลและไม่มีปริมาตร และใช้อิทธิพลจากตำแหน่งและความเร็วในการเคลื่อนที่เท่านั้น จะเกิดเป็นอัลกอริทึม Particle Swarm Optimization

3.3.2 ขั้นตอนการทำงานของวิธีความฉลาดแบบกลุ่ม

วิธีการความฉลาดแบบกลุ่มมีขั้นตอนการทำงาน 6 ขั้นตอน ดังนี้

1. สร้างกลุ่มประชากรเริ่มต้นโดยกำหนดจำนวนอนุภาคและค่าพารามิเตอร์ต่าง ๆ เพื่อให้โปรแกรมทำการสุ่มคุณสมบัติประชากรเริ่มต้น และกำหนดฟังก์ชันวัตถุประสงค์เพื่อใช้คำนวณออกแบบอย่างเหมาะสม
2. ประเมินคุณภาพประชากรโดยคำนวณผ่านฟังก์ชันวัตถุประสงค์ เพื่อหาคุณภาพของตำแหน่งปัจจุบันของอนุภาคแต่ละตัว
3. คำนวณตำแหน่งที่ดีที่สุดของอนุภาคแต่ละตัว โดยทำการเปรียบเทียบคุณภาพของตำแหน่งปัจจุบันและเทียบกับตำแหน่งที่ดีที่สุดที่อนุภาคนั้นเคยอยู่ ถ้าหากตำแหน่งปัจจุบันดีกว่าจะทำการบันทึกตำแหน่งที่ดีที่สุดใหม่
4. คำนวณตำแหน่งที่ดีที่สุดของกลุ่มประชากร โดยเปรียบเทียบคุณภาพของตำแหน่งของอนุภาคแต่ละตัว และหาว่าตำแหน่งปัจจุบันมีอนุภาคตัวไหนที่คุณภาพดีกว่าตำแหน่งที่ดีที่สุดที่กลุ่มอนุภาคเคยอยู่ไหม ถ้าหากมีจะทำการบันทึกตำแหน่งที่ดีที่สุดของกลุ่มประชากร
5. อัปเดตข้อมูลความเร็วตามสมการคำนวณความเร็วที่กำหนด เพื่อใช้ในการคำนวณตำแหน่งที่อนุภาคแต่ละตัวจะเคลื่อนตัวในการคำนวณซ้ำครั้งต่อไป และทำการคำนวณผ่านฟังก์ชันวัตถุประสงค์ซ้ำ
6. ได้ผลลัพธ์การออกแบบอย่างเหมาะสมออกมา ถ้าหากพอใจกับผลลัพธ์สามารถหยุดการคำนวณซ้ำได้ แต่ถ้าหากยังไม่พึ่งพอใจ สามารถทำการเริ่มจากขั้นตอนที่ 1 ใหม่เพื่อทำการดำเนินการคำนวณใหม่

โดยอัลกอริทึมนี้มีขั้นตอนการทำงานที่ไม่ซับซ้อน สามารถเข้าใจได้ง่ายจาก Flowchart ดังรูปที่ 3-3



รูปที่ 3-3 Flowchart ขั้นตอนการทำงานของ PSO

3.3.3 การคำนวณในวิธีความฉลาดแบบกลุ่ม

จากข้อมูลในหัวข้อ 3.2.1 ให้ n คือขนาดของกลุ่มอนุภาค (swarm size) โดยอนุภาคแต่ละตัวจะมีมิติของตัวแปร คือ D ได้ว่า

ตำแหน่งของอนุภาค คือ $X_i = \left(x_{i1}, x_{i2}, \ldots, x_{id}, \ldots, x_{iD} \right)$

เวกเตอร์ความเร็ว คือ $V_i = \left(v_{i1}, v_{i2}, \ldots, v_{id}, \ldots, v_{iD} \right)$

ตำแหน่งที่ดีที่สุดส่วนตัว คือ $P_i = \left(p_{i1}, p_{i2}, \ldots, p_{id}, \ldots, p_{iD} \right)$

ตำแหน่งที่ดีของฝูง คือ $P_i = \left(p_{g1}, p_{g2}, \ldots, p_{gd}, \ldots, p_{gD} \right)$

โดยที่ i คือ ลำดับอนุภาคในกลุ่มประชากร

จากนั้นสามารถอัปเดตข้อมูลคุณภาพของอนุภาคจากการดำเนินการผ่านฟังก์ชันจุดประสงค์ได้ ดังนี้

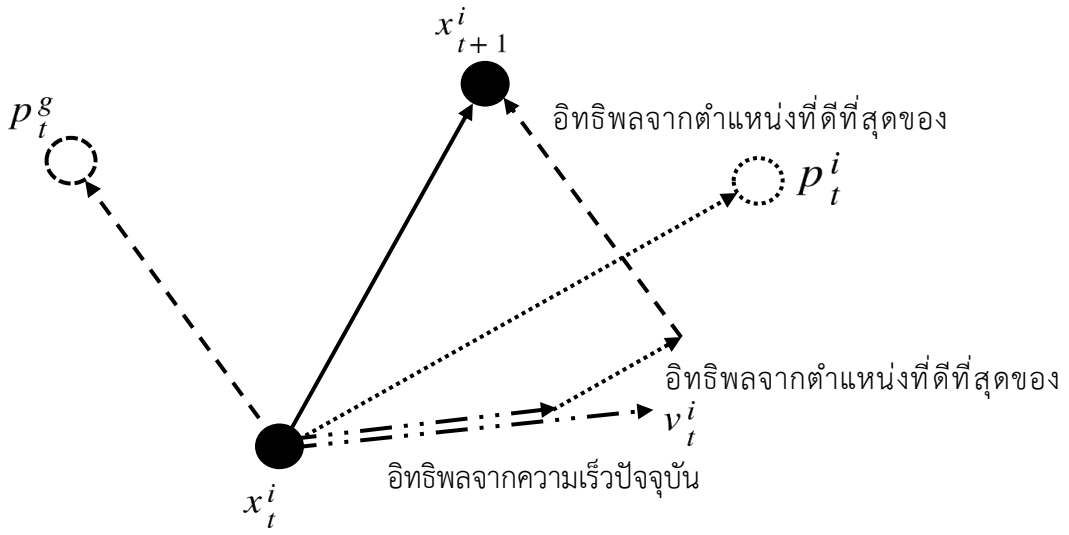
$$p^d_{i.t+1} = \begin{cases} x^d_{i,t+1}, & if \ \left(X_{i,t+1} \right) < f\left(P_{i,t} \right) \\ p^d_{i,t}, & otherwise \end{cases} \qquad (3.15)$$

โดยที่ t คือ ลำดับครั้งในการคำนวณผ่านฟังก์ชันจุดประสงค์

ซึ่งตำแหน่งที่ดีที่สุดของกลุ่มอนุภาคือตำแหน่งที่ดีที่สุดเมื่อเทียบกับตำแหน่งที่ดีที่สุดของอนุภาคทุกตัวจากสมการ (3.15) ในการอัปเดตความเร็วและตำแหน่งในการคำนวณแต่ละครั้ง เพื่อเพิ่มประสิทธิภาพของวิธีความฉลาดแบบกลุ่ม สามารถใส่ตัวแปรน้ำหนักของเส้นทางเพิ่มไปอีกหนึ่งตัวแปรได้ คือ (inertia weight, w) ซึ่งจะใช้ถ่วงน้ำหนักของเวกเตอร์ความเร็วในการคำนวณครั้งปัจจุบัน เพื่อใช้คำนวณเวกเตอร์ความเร็วที่เปลี่ยนแปลงในรอบการดำเนินการครั้งต่อไป โดย

$$v^d_{i,t+1} = w \cdot v^d_{i,t} + c_1 r_1 \left(p^d_{i,t} - x^d_{i,t} \right) + c_2 r_2 \left(p^d_{g,t} - x^d_{i,t} \right) \qquad (3.16)$$

จากสมการทั้งหมดสามารถสร้างรูปภาพที่อธิบายเส้นทางการเดินทางของอนุภาคใน PSO ได้ดังรูปที่ 3-4



รูปที่ 3-4 วิธีการเคลื่อนตัวของอนุภาค

การเคลื่อนตัวของอนุภาคในวิธีความฉลาดแบบกลุ่มจะเคลื่อนตัวโดยมีอิทธิพลมาจาก 3 ตัวแปรสำคัญคือ ตำแหน่งที่ดีที่สุดของอนุภาค (cognitive factor), ตำแหน่งที่ดีที่สุดของกลุ่มอนุภาค (social factor) และ ความเร็วปัจจุบันของอนุภาคนั้น ๆ โดยถ้าหากเราทำการคำนวณซ้ำผ่านฟังก์ชันวัตถุประสงค์ในอนุภาคทุกตัวในกลุ่ม จะทำให้เห็นภาพของการเคลื่อนตัวของอนุภาคที่จะมุ่งหน้าเข้าสู่เป้าหมายที่เราต้องการ กล่าวคือ เป็นการคำนวณการออกแบบอย่างเหมาะสมผ่านการแทนค่าตัวแปรและคำนวณอย่างมีหลักการและไม่ใช่การสุ่มอย่างแท้จริง

3.3.4 ตัวอย่างการคำนวณของวิธีความฉลาดแบบกลุ่ม

หาค่าที่ต่ำที่สุดสำหรับ $f(x, y)$ ภายใต้ x และ y มีค่าตั้งแต่ 0 ถึง 5 โดยที่

$f(x, y) = (x - 3.14)^2 + (y - 2.72)^2 - \sin(3x + 1.41) + \sin(4y - 1.73)$ (Adrian Tram, 2021)

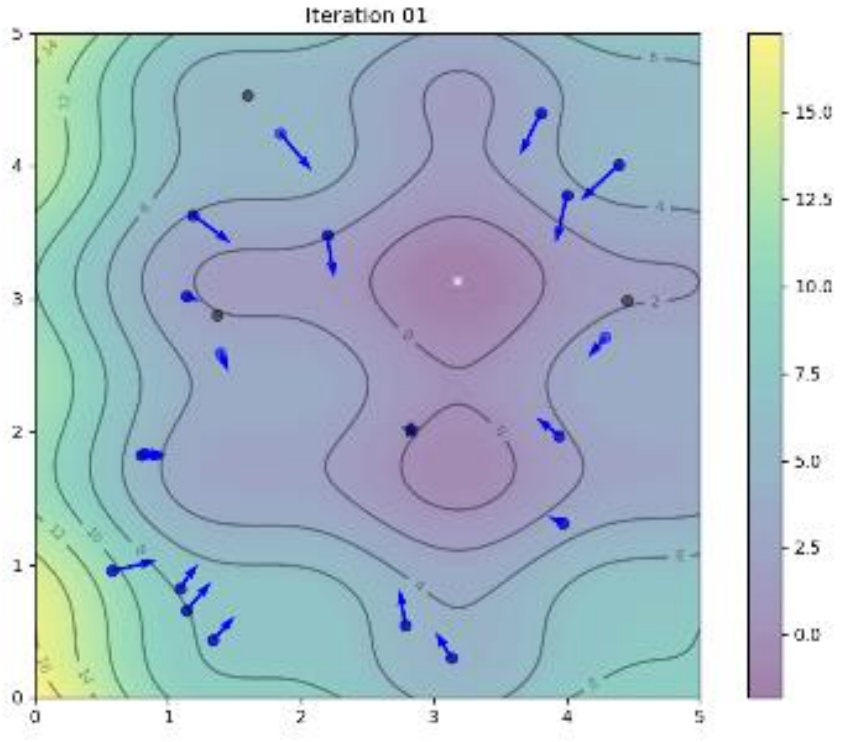
และมีการกำหนดพารามิเตอร์ของวิธีความฉลาดแบบกลุ่มดังนี้

n = 20

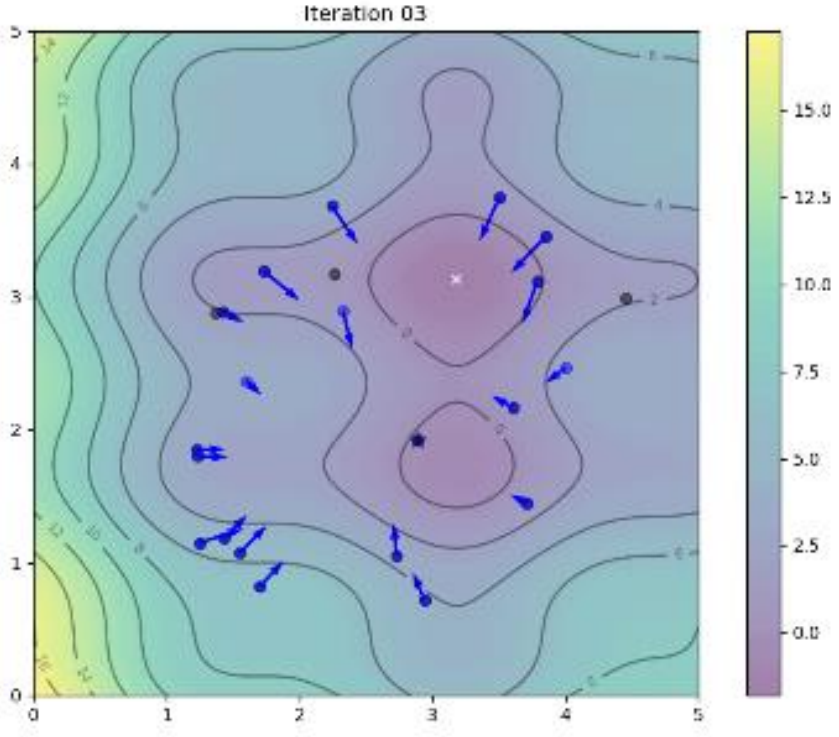
c_1 = 0.1

c_2 = 0.1

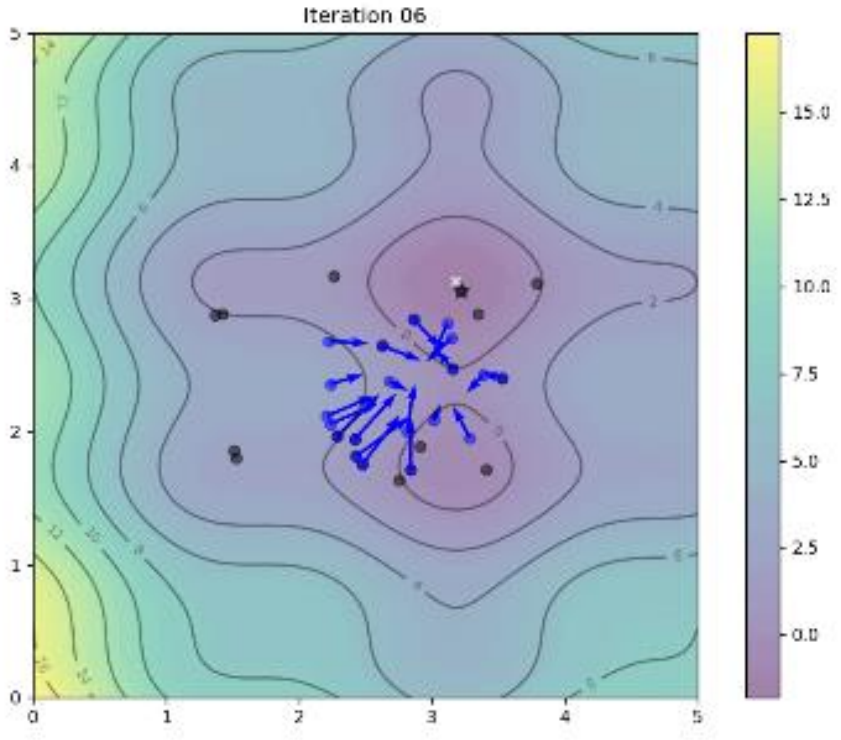
w = 0.8



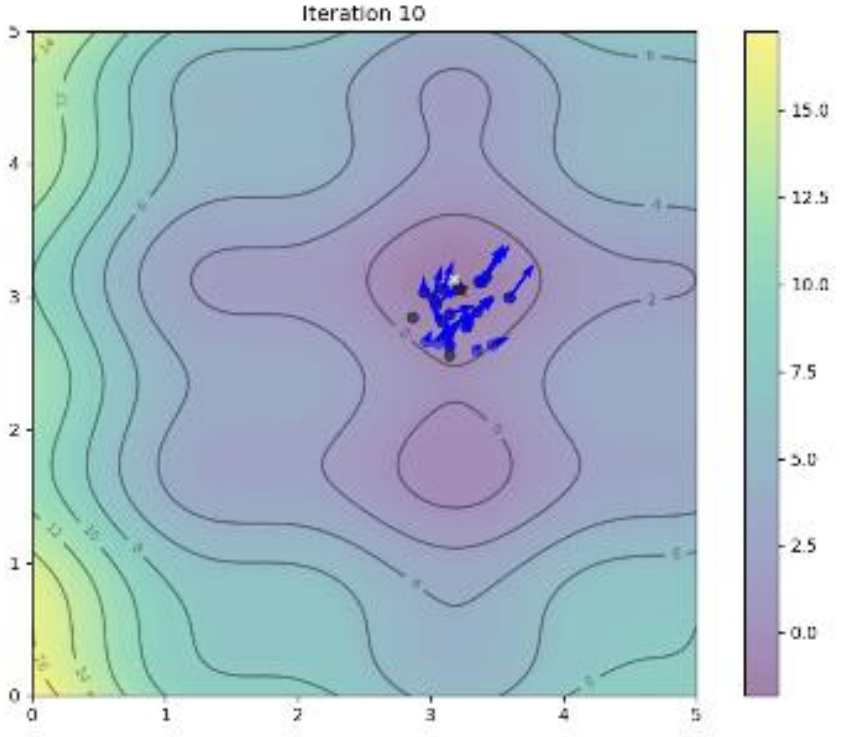
รูปที่ 3-5 การเคลื่อนตัวครั้งที่ 1 ของวิธีความฉลาดแบบกลุ่ม



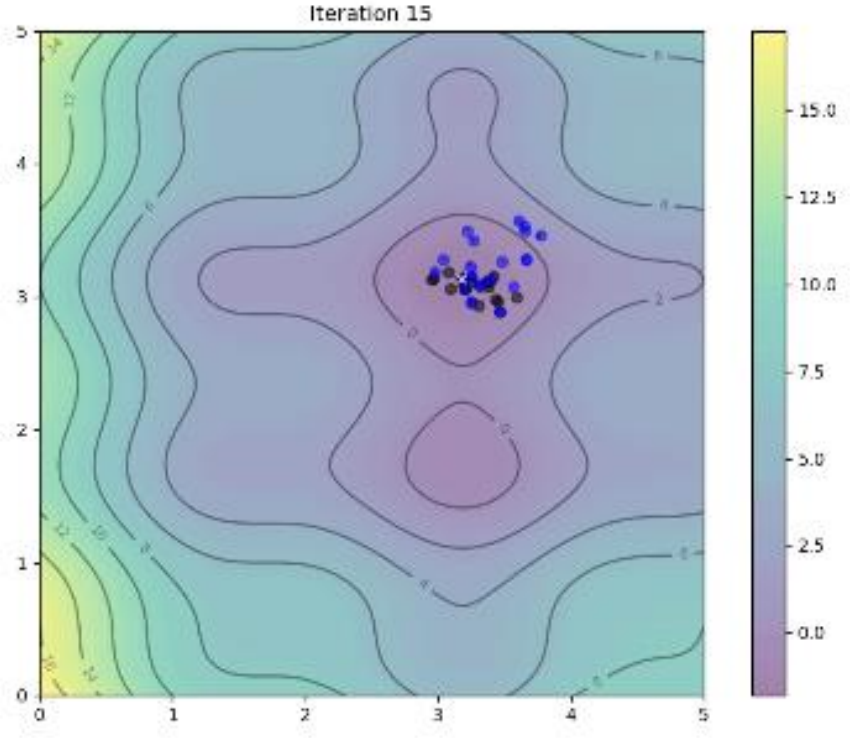
รูปที่ 3-6 การเคลื่อนตัวครั้งที่ 1 ของวิธีความฉลาดแบบกลุ่ม



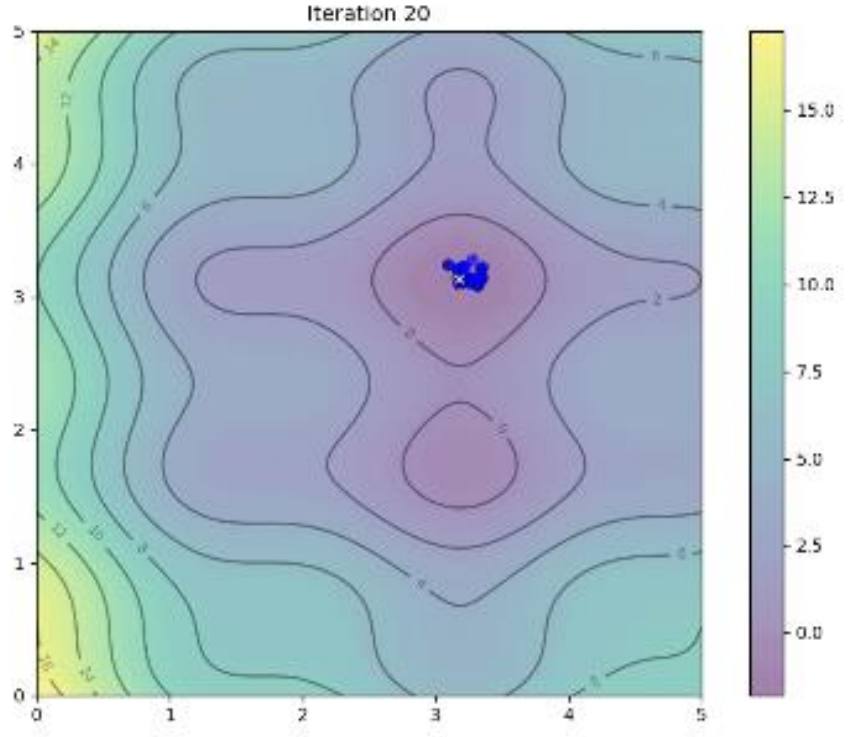
รูปที่ 3-7 การเคลื่อนตัวครั้งที่ 6 ของวิธีความฉลาดแบบกลุ่ม



รูปที่ 3-8 การเคลื่อนตัวครั้งที่ 10 ของวิธีความฉลาดแบบกลุ่ม



รูปที่ 3-9 การเคลื่อนตัวครั้งที่ 15 ของวิธีความฉลาดแบบกลุ่ม



รูปที่ 3-10 การเคลื่อนตัวครั้งที่ 20 ของวิธีความฉลาดแบบกลุ่ม

เมื่อทำการคำนวณซ้ำตามจำนวนครั้งที่กำหนด จะสามารถแสดงออกมาเป็นแผนภาพได้ดังรูปที่ 3-5 ถึงรูปที่3-10 โดยภาพตัวอย่างเป็นการหาคำตอบของฟังก์ชันวัตถุประสงค์ด้วยวิธีความฉลาดแบบกลุ่ม โดยทำการคำนวณหาคำตอบ 20 ครั้ง จะเห็นว่าอนุภาคแต่ละตัวจะมีจุดเริ่มต้นที่แตกต่างกัน โดยเกิดจากการสุ่มประชากรเริ่มต้นว่าอนุภาคตัวไหนจะมีตำแหน่งเริ่มต้นที่ไหน และมีการเคลื่อนตัวเข้าหาผลลัพธ์ที่ดีที่สุดโดยมีการเคลื่อนที่ที่คล้ายกับฝูงสัตว์ที่มุ่งหน้าเข้าหาเป้าหมาย ซึ่งตามธรรมชาติอาจเป็นแหล่งอาหาร หรือตำแหน่งที่ต้องการอพยพไปหา แต่ในมุมมองของการออกแบบอย่างเหมาะสมแล้ว จุดมุ่งหมายนั้นคือผลลัพธ์ที่ดีที่สุดของการออกแบบนั่นเอง

โดยจากปัญหาตัวอย่าง ได้ว่า

ตำแหน่งที่ดีที่สุด $(x, y) = (3.1854, 3.1297)$

ผลลัพธ์ที่ดีที่สุด $Global\ Optima = -1.8084$

3.4 ขั้นตอนการปฏิบัติงาน

3.4.1 ศึกษาขั้นตอนการทำงานของโปรแกรม Project-Python3D

โปรแกรมการคำนวณแรงภายในโครงสร้าง Project-Python3D เป็นโปรแกรมภาษาไพทอนที่สามารถคำนวณแรงภายในของโครงสร้าง Truss และ Frame ได้แต่ไม่สามารถคำนวณโครงสร้างผสมได้ด้วยการคำนวณแบบ Direct Stiffness Method ในโครงงานนี้ใช้เพียงโครงสร้างโครงข้อหมุนเหล็กจึงสามารถนำโปรแกรมมาใช้ในการออกแบบอย่างเหมาะสมได้โดยไม่ต้องแก้ไขโปรแกรมในขั้นตอนการกำหนดคลาส Truss หรือ Frame

ในการศึกษาขั้นตอนการทำงานนี้จะทำการศึกษาด้วยตัวอย่างโครงสร้างโครงข้อหมุนเหล็กที่มีตำแหน่งพิกัดจุดต่อดังนี้

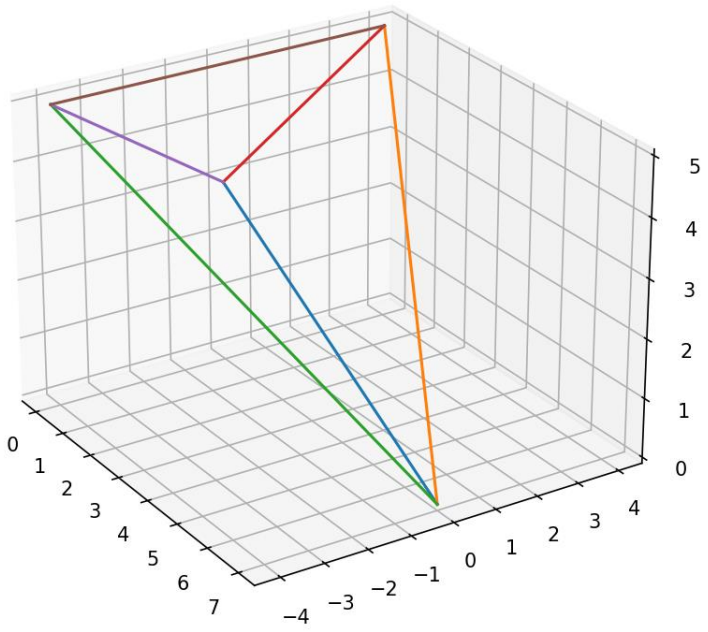
A = (7, 0, 0)

B = (0, 0, 3)

C = (0, 4, 5)

D = (0, -4, 5)

โดยมีโครงข้อหมุนเหล็กคือ AB, AC, AD, BC, BD, CD ดังรูปที่ 3-11



รูปที่ 3-11 โครงสร้างตัวอย่างที่ใช้ในการศึกษาโปรแกรม Project-Python3D

โดยขั้นตอนการทำงานของโปรแกรม Project-Python3D มีขั้นตอนการทำงานดังนี้

1. ขั้นตอนแรกของการใช้โปรแกรม โปรแกรมต้องทำการ Import ข้อมูลและคำสั่งจากไฟล์ต่าง ๆ ที่ใช้ในการคำนวณแรงภายใน

```
import numpy as np
from Material import Material
from Ball import Ball
from Structure import Structure
from Truss import Truss
from StructureAssemble import StructureAssemble
from Section import Section
from matplot import matplot
```

รูปที่ 3-12 การ Import ข้อมูลและคำสั่ง

2. กำหนดวัสดุที่ใช้ในการคำนวณแรงภายใน

```
mat = Material()
mat.Steel()
```

รูปที่ 3-13 กำหนดวัสดุ

3. กำหนดหน้าตัดชิ้นส่วนของโครงสร้าง โดยคำสั่งนี้ถ้าหากผู้ใช้ใส่ input เป็นข้อมูลตัวเลขข้อมูลเดียวจะเป็นการกำหนดพื้นที่หน้าตัดหน่วยเป็นตารางเมตร

```
sectA = Section(1.0)
sectB = Section(1.5)
```

รูปที่ 3-14 กำหนดหน้าตัด

4. กำหนดตำแหน่งพิกัดจุดต่อของโครงสร้างโดยหน่วยระยะทางเป็นเมตร

```
A = Ball(7.0,0.0,0.0,'A')
B = Ball(0.0,0.0,3.0,'B')
C = Ball(0.0,4.0,5.0,'C')
D = Ball(0.0,-4.0,5.0,'D')
```

รูปที่ 3-15 กำหนดพิกัดจุดต่อ

5. กำหนดโครงข้อหมุนเหล็กจากตำแหน่งพิกัดจุดต่อ

```
AB = Truss(A,B,mat,sectA,'AB')
AC = Truss(A,C,mat,sectA,'AC')
AD = Truss(A,D,mat,sectA,'AD')
BC = Truss(B,C,mat,sectB,'BC')
BD = Truss(B,D,mat,sectB,'BD')
CD = Truss(C,D,mat,sectB,'CD')
```

รูปที่ 3-16 กำหนดโครงข้อหมุน

6. กำหนดน้ำหนักบรรทุก โดยขั้นตอนนี้ไม่สามารถแบ่งน้ำหนักบรรทุกเป็น น้ำหนักบรรทุกคงที่และน้ำหนักบรรทุกจรได้ ผู้ใช้ต้องทำการคำนวณด้วยตนเอง ถ้าหากต้องการออกแบบด้วยวิธี ASD หรือ LRFD ตามมาตรฐาน AISC 360-16 โดยเป็นการกำหนด Point Load ที่ตำแหน่งพิกัดจุดต่อที่ต้องการ ที่ได้กำหนดไว้ในขั้นตอนที่ 4 โดยน้ำหนักบรรทุกที่ใส่จะมีหน่วยเป็นนิวตัน

```
A.load(0,0,0,-200)
```

รูปที่ 3-17 กำหนดน้ำหนักบรรทุก

7. กำหนดรูปแบบของ support ที่ตำแหน่งต่าง ๆ ของโครงสร้าง โดยสามารถกำหนด support เป็น roller, pinned หรือ fixed ได้

```
B.rollerYZ()
C.socket()
D.rollerY()
```

รูปที่ 3-18 กำหนด support

8. การประกอบโครงสร้างและทำการคำนวณแรงภายใน โดยผู้ใช้สามารถกำหนดว่าจะคำนวณน้ำหนักของโครงสร้างหรือไม่ จากคำสั่ง addSelfWeight ถ้าหากไม่ป้อนคำสั่งนี้โปรแกรมจะไม่คำนวณน้ำหนักที่เกิดจากโครงสร้าง และคำสั่ง addSelfWeight ตามที่โปรแกรมใช้คำนวณน้ำหนักจะใช้ค่าแรงโน้มถ่วง (g_z = 10 m/s^2)

```
Structure.init()
t1 = StructureAssemble()
# t1.addSelfWeight(0)
t1.solve()
```

รูปที่ 3-19 ประกอบโครงส้รางและคำนวณแรงภายใน

9. การแสดงผลลัพธ์ โดยโปรแกรมนี้จะทำการคำนวณและให้ผลลัพธ์ออกมาในรูปแบบของรูปโครงสร้างสามมิติแสดงใน matplot และแรงภายในและการกระจัดของโครงสร้างแสดงในไฟล์โปรแกรม Microsoft Excel

LOAD_CAS	0											
Nodal_Dis	A		7	0	0	u=	-1.45E-07	v=	-2.00E-09	w=	-2.44E-07	deform=	2.84E-07
Nodal_Dis	B		0	0	3	u=	0	v=	-2.00E-09	w=	2.08E-08	deform=	2.09E-08
Nodal_Dis	C		0	4	5	u=	0	v=	0	w=	0	deform=	0
Nodal_Dis	D		0	-4	5	u=	0	v=	-4.00E-09	w=	0	deform=	4.00E-09
Member_	AB	P=	-761.577										
Member_	AC	P=	474.3416										
Member_	AD	P=	474.3416										
Member_	BC	P=	-335.41										
Member_	BD	P=	-335.41										
Member_	CD	P=	100										

รูปที่ 3-20 ผลลัพธ์แรงภายในและการกระจัด

3.4.2 พัฒนาโปรแกรม Project-Python3D ให้สามารถตรวจสอบความปลอดภัย

ขั้นตอนนี้จะทำการเขียนโปรแกรมภาษาไพทอนเพื่อให้สามารถใช้ในการตรวจสอบความปลอดภัยตามมาตรฐานการออกแบบโครงสร้างเหล็ก AISC 360-16 ด้วยวิธีการออกแบบ LRFD โดยโปรแกรมจะให้ผลลัพธ์ของแรงภายในโครงสร้างโครงข้อหมุนเหล็กออกมาดังรูปที่ 3-20 ซึ่งถ้าหากเป็นแรงดึง ผลลัพธ์จะมีค่าเป็นค่าบวก ส่วนแรงอัดจะมีค่าเป็นแรงลบ โดยจากทฤษฎีการออกแบบโครงสร้างเหล็กจากแรงภายในที่เป็นแรงดึงและแรงอัดที่ได้กล่าวไว้ในหัวข้อ 3.1.2 และ 3.1.3 สามารถนำไปใช้ในการเขียนโปรแกรมตรวจสอบความปลอดภัยของโครงสร้างได้ดังนี้

การรับแรงดึง

การตรวจสอบการรับแรงดึงสามารถแบ่งการวิบัติเป็นสองกรณีคือ

1. การวิบัติจากหน่วยแรงคราก ตามสมการ (3.5)

2. การวิบัติจากแรงดึงประลัย ตามสมการ (3.6)

โดยความสามารถในการรับแรงดึง จะเลือกใช้ค่าที่น้อยกว่าในการใช้ตรวจสอบความปลอดภัย สามารถเขียนออกมาเป็นภาษาไพทอนได้ดังรูปที่ 3-21

```
class Tension :
        def __init__(self,f,mat,sect,status = '',Ty=0,Tu=0) :
            self.f = f
            self.mat = mat
            self.sect = sect
            self.Ty = 0
            self.Tu = 0
            self.Ty = 0.90*mat.Fy*sect.A
            self.Tu = 0.75*mat.Fu*sect.A
            self.status = "OK"
            if f>min(self.Ty,self.Tu) :
                if self.Ty <= self.Tu :
                    self.status = "TYF"
                else :
                    self.status = "TRF"
            elif f<=min(self.Ty,self.Tu) :
                self.status = "OK"
            else :
                self.status = "error"
```

รูปที่ 3-21 โค้ดตรวจสอบแรงดึง

จากโค้ดดังกล่าวความหมายของตัวแปรที่ใช้คำนวณแต่ละตัวคือ

self.f	คือ	แรงภายในที่เกิดในโครงข้อหมุนเหล็ก	(นิวตัน)
self.mat	คือ	วัสดุที่ใช้เป็นโครงสร้าง	
self.Ty	คือ	กำลังดึงคราก	(นิวตัน)
self.Tu	คือ	กำลังดึงประลัย	(นิวตัน)
self.sect.A	คือ	พื้นที่หน้าตัด	(ตารางเมตร)

โดยจะนำผลลัพธ์จากกการคำนวณแรงภายในมาเปรียบเทียบกับค่ากำลังที่รับได้เพื่อตรวจสอบความปลอดภัย และให้ output ออกมาเป็นสถานะความปลอดภัย คือ self.status ถ้าหากเป็น “TYF” หมายความว่าเกิด Tension Yielding Failure และถ้าเป็น “TRF” แสดงว่าเกิด Tension Rupture Failure

การรับแรงอัด

การตรวจสอบความสามารถในการรับการรับแรงอัดสามารถทำได้ตามสมการ (3.6) และสามารถเขียนเป็นโค้ดการตรวจสอบความปลอดภัยในการรับแรงอัดได้ดังรูปที่ 3-22 โดยที่จะแบ่งเป็นกรณี Elastic Buckling และ Inelastic Buckling ตามสมการ (3.7) และ (3.8) และทำการคำนวณ P_u ออกมาเพื่อเปรียบเทียบกับแรงภายในที่เกิดขึ้นและตรวจสอบความปลอดภัยของชิ้นส่วนโครงข้อหมุนเหล็ก โดยจะทำการตรวจสอบตามขั้นตอนและให้ output ออกมาเป็นสถานะของชิ้นส่วนดังกล่าว ถ้าหากชิ้นส่วนวิบัติจะได้สถานะ “CF” หมายความว่าเกิด Compression Failure

```
class Compression :
        def __init__(self,f,mat,sect,L,status,Pu=0) :
            self.f = f
            self.mat = mat
            self.sect = sect
            self.L = L
            self.Pu = 0
            self.status = "OK"
            r = sect.r
            Fy = mat.Fy
            E = mat.E
            K = 1
            Slen = K*L/r
            Fe = math.pi**2*E/(Slen)**2
#Inelastic Buckling
            if Slen <= 4.71*math.sqrt(E/Fy) or Fy/Fe <= 2.25 :
                Fcr = (0.658**(Fy/Fe))*Fy
#Elastic Buckling
            else :
                Fcr = 0.877*Fe
            self.Pu = 0.9*Fcr*sect.A
            if -1*f >= self.Pu :
                self.status = "CF"
            else :
                self.status = "OK"
```

รูปที่ 3-22 โค้ดตรวจสอบแรงอัด

3.4.3 สร้างชุดข้อมูลหน้าตัดเหล็กท่อกลม

ทำการสร้างชุดข้อมูลของหน้าตัดเหล็กขึ้นมาตามตาราง 3-2 โดยที่มีคุณสมบัติของหน้าตัดเหล็กที่ใช้ในการคำนวณแรงของโครงงสร้างและกำลังของชิ้นส่วนแต่ละชิ้นตามมาตรฐาน มอก. โดยทำการสร้างคลาสใหม่คือ HollowBar ซึ่งใช้คำนวณคุณสมบัติหน้าตัดของเหล็กท่อกลมโดยมี input เพียงขนาดเส้นผ่าศูนย์กลางและความหนาของท่อเหล็กในหน่วยมิลลิเมตร โดยมีขั้นตอนการคำนวณดังรูปที่ 3-23

```
class HollowBar(Section):
    def __init__(self,diameter=0,thickness=0):
        self.diameter = diameter
        self.thickness = thickness
        d = diameter*1e-3
        t = thickness*1e-3
        area = math.pi*(d*d-(d-2*t)**2)/4
        Ia = Ib = math.pi*(d**4-(d-2*t)**4)/64
        J = Ia+Ib
        Sa = Sb = math.pi*(d**4-(d-2*t)**4)/d/32
        rx = ry = r = math.sqrt(Ia/area)
        Section.__init__(self,area,Ia,Ib,J,Sa,Sb,r,d,t,self.name)
```

รูปที่ 3-23 คลาส HollowBar

จากนั้นนำโค้ดการคำนวณคุณสมบัติหน้าตัดไปใช้สร้างชุดข้อมูลหน้าตัดเหล็กตามมาตรฐาน มอก. เพื่อใช้ในการออกแบบอย่างเหมาะสม

```
from Section import HollowBar

HSS = [HollowBar(21.7,2.0),
       HollowBar(27.2,2.3),
       HollowBar(34.0,2.3),
       HollowBar(42.7,2.3),
       HollowBar(48.6,2.3),
       HollowBar(48.6,3.2),
       HollowBar(60.5,3.2),
       HollowBar(60.5,4.0),
       HollowBar(76.3,3.2),
       HollowBar(76.3,4.0),
       HollowBar(89.1,3.2),
       HollowBar(89.1,4.0),
       HollowBar(101.6,3.2),
       HollowBar(101.6,4.0),
       HollowBar(114.3,3.2),
       HollowBar(114.3,4.5),
       HollowBar(114.3,5.6),
       HollowBar(139.8,4.5),
       HollowBar(139.8,6.0),
       HollowBar(165.2,4.5),
       HollowBar(165.2,6.0),
       HollowBar(190.7,5.0),
       HollowBar(190.7,7.0),
       HollowBar(216.3,6.0),
       HollowBar(216.3,8.0),
       HollowBar(267.4,6.0),
       HollowBar(267.4,8.0)]
```

รูปที่ 3-24 สร้างชุดข้อมูลหน้าตัดเหล็ก

โดยคุณสมบัติของเหล็กในโปรแกรมสามารถปรับแต่งได้ แต่ในโครงงานนี้ใช้ค่าคุณสมบัติต่าง ๆ ของวัสดุเหล็กดังนี้

Modulus of Elasticity	=	200	GPa
Modulus of Rigidity	=	80	GPa
ความหนาแน่น	=	7860	กิโลกรัม/ลูกบากศ์เมตร
หน่วยแรงคราก (F_y)	=	235	MPa
กำลังดึงประลัย (F_u)	=	360	MPa

3.4.4 พัฒนาโปรแกรม Project-Python3D ให้ไม่เกิดการทับซ้อนของแรงภายใน

ในตัวโปรแกรม Project-Python3D จะมีขั้นตอนการ input แรงสองวิธีคือ การให้ข้อมูลแทนที่น้ำหนักและการเพิ่มน้ำหนัก ทำให้การทำซ้ำของวิธีความฉลาดแบบกลุ่มเกิดปัญหาคือ การน้ำหนักที่กระทำต่อโครงสร้างโครงข้อหมุนเหล็กจะเพิ่มขึ้นเรื่อย ๆ ในแต่ละครั้งที่ทำการคำนวณซ้ำและทำให้ไม่สามารถออกแบบอย่างเหมาะสมได้

โดยในตัวโปรแกรมเริ่มต้นคลาส Truss จะมีคำสั่ง Truss.addWeight ซึ่งเป็นการเพิ่มน้ำหนักชิ้นส่วนของโครงสร้างเข้าพิกัดจุดต่อของโครงข้อหมุนเหล็ก จึงต้องสร้างคำสั่งใหม่เพื่อลบน้ำหนักที่ใส่เพิ่มในการคำนวณซ้ำแต่ละรอบโดยคำสั่งคือ Truss.subtractWeight

```
def addWeight(self,loadCase,gx,gy,gz):
    mass = 1.2*self.mat.Rho*self.sect.A*self.L
    halfWeightX = mass*gx/2
    halfWeightY = mass*gy/2
    halfWeightZ = mass*gz/2
    self.pI.addLoad(loadCase,halfWeightX,halfWeightY,halfWeightZ)
    self.pJ.addLoad(loadCase,halfWeightX,halfWeightY,halfWeightZ)

def subtractWeight(self,loadCase,gx,gy,gz):
    mass = 1.2*self.mat.Rho*self.sect.A*self.L
    halfWeightX = -mass*gx/2
    halfWeightY = -mass*gy/2
    halfWeightZ = -mass*gz/2
    self.pI.addLoad(loadCase,halfWeightX,halfWeightY,halfWeightZ)
    self.pJ.addLoad(loadCase,halfWeightX,halfWeightY,halfWeightZ)
```

รูปที่ 3-25 คำสั่ง subtractWeight

โดยน้ำหนักดังกล่าวจะถูกนำไปใช้อีกทีตอนทำการคำนวณโครงสร้างรวมในคลาส StructureAssemble ซึ่งมีคำสั่ง addSelfWeight จึงต้องทำการสร้างคำสั่งเพื่อล้างข้อมูลนั้นคือ subtractSelfWeight

```
def addSelfWeight(self,loadCase=0,gx=Material.gravityX,gy=Material.gravityY,gz=Material.gravityZ):
    truss = Truss.start
    while truss != None:
        truss.addWeight(loadCase,gx,gy,gz)
        truss = truss.next
    frame = Frame.start
    while frame != None:
        frame.addWeight(loadCase,gx,gy,gz)
        frame = frame.next

def subtractSelfWeight(self,loadCase=0,gx=Material.gravityX,gy=Material.gravityY,gz=Material.gravityZ):
    truss = Truss.start
    while truss != None:
        truss.subtractWeight(loadCase,gx,gy,gz)
        truss = truss.next
    frame = Frame.start
    while frame != None:
        frame.subtractWeight(loadCase,gx,gy,gz)
        frame = frame.next
```

รูปที่ 3-26 คำสั่ง subtractSelfWeight

นอกจากน้ำหนักของชิ้นส่วนโครงสร้างที่เป็นปัญหาต่อการคำนวณวิธีความฉลาดแบบกลุ่มแล้ว แรงภายในของโครงสร้างยังมีการเพิ่มขึ้นเรื่อย ๆ เนื่องจากตัวโปรแกรมไม่มีการรีเซ็ตข้อมูลหลังจากคำนวณเสร็จในแต่ละรอบ จึงต้องทำการรีเซ็ตข้อมูลแรงภายในของโครงข้อหมุนเหล็กด้วยคำสั่ง Truss.reset

```
def reset(self):
    Truss.start = None
    Truss.current = None
```

รูปที่ 3-27 คำสั่ง Truss.reset

3.4.5 การสร้างฟังก์ชันการลงโทษ

สำหรับการออกแบบเหมาะสมด้วยวิธีความฉลาดแบบเหมาะสมในโครงงานนี้ เป็นวิธีที่ไม่สามารถดึงข้อมูลที่ผลลัพธ์ไม่ผ่านมาตรฐานออกจากการคำนวณระหว่างทางได้ จึงต้องสร้างฟังก์ชันการลงโทษ

ซึ่งก่อนจะตัดสินว่าโครงข้อหมุนตัวไหนจะมีการลงโทษ จะต้องผ่านการตรวจสอบความปลอดภัยในคลาส Truss ก่อน ดังคำสั่งต่อไปนี้

```
def CheckTension(self):
    TENS = Tension(self.P,self.mat,self.sect,'',0,0)
    self.status = TENS.status
    self.Ty = TENS.Ty
    self.Tu = TENS.Tu

def CheckCompression(self):
    COMP = Compression(self.P,self.mat,self.sect,self.L,'',0)
    self.status = COMP.status
    self.Pu = COMP.Pu

def checkFailures(self,loadCase,factor):
    self.factor(loadCase,factor)
    self.status = "OK"
    if self.P > 0 :
        self.CheckTension()
    elif self.P < 0 :
        self.CheckCompression()
    elif self.P == 0 :
        self.status = "No Force"
    self.m = m = self.mat.Rho*self.sect.A*self.L
```

รูปที่ 3-28 คำสั่งตรวจสอบความปลอดภัย

โดยจะให้ output ออกมาเป็นสถานะของโครงข้อหมุนว่ามีการวิบัติหรือไม่ อย่างไร

ด้วยความที่ฟังก์ชันวัตถุประสงค์ต้องการหาโครงสร้างที่น้ำหนักน้อยที่สุดและมีความปลอดภัยตามมาตรฐาน เพราะฉะนั้นฟังก์ชันลงโทษสำหรับชิ้นส่วนของโครงสร้างที่เกิดการวิบัติจึงต้องเป็นการเพิ่มน้ำหนักของโครงสร้างตามปริมาณชิ้นส่วนที่วิบัติ และตามแรงภายในที่เกินกำลังที่รับได้ของชิ้นส่วนเหล่านั้น จึงทำให้เกิดฟังก์ชันการลงโทษในคลาส StructureAssemble ดังนี้

```
def Process(self,file,loadCase=0,factor=[],mass = 0):
        y = 0
        self.x = x = 0
        self.structmass = 0
        self.truss = Truss.start
        self.penalty = penalty = 0
        self.totalpenalty = totalpenalty = 0
        while self.truss != None:
            self.truss.status = "OK"
            self.truss.checkFailures(loadCase,factor)
            penalty = 0
            if self.truss.status != "OK" :
                if self.truss.status == "TYF" :
                    penalty = (self.truss.P-self.truss.Ty)/9.81
                elif self.truss.status == "TRF" :
                    penalty = (self.truss.P-self.truss.Tu)/9.81
                elif self.truss.status == "CF" :
                    penalty = (-self.truss.P-self.truss.Pu)/9.81
                elif self.truss.status == "No Force" :
                    pass
            y = y + self.truss.m + penalty
            self.totalpenalty = self.totalpenalty + penalty
            penalty = 0
            self.truss = self.truss.next
        x = self.structmass = y
        self.x = x
        y = None
        self.structmass = None
        self.penalty = None
        self.totalpenalty = None
```

รูปที่ 3-29 ฟังก์ชันกาลงโทษ

โดยที่	penalty	คือ	น้ำหนักที่ลงโทษของชิ้นส่วน	(กิโลกรัม)
	totalpenalty	คือ	น้ำหนักที่ลงโทษรวมในโครงสร้าง	(กิโลกรัม)

บทลงโทษจะมีปริมาณเท่ากับแรงภายในที่เกินขีดจำกัดแรงที่สามารถรับได้ของโครงข้อหมุนแต่ละตัว โดยที่แปลงหน่วยจากแรงภายในที่ได้จากโปรแกรม (นิวตัน) เป็น กิโลกรัมเพื่อให้สามารถใช้กับค่าน้ำหนักโครงสร้างได้อย่างเหมาะสม

3.4.6 การเขียนโปรแกรมวิธีความฉลาดแบบกลุ่ม

จากหัวข้อที่ 3.3.2 ขั้นตอนทั้งหมด สามารถเขียนโปรแกรมออกมาได้ดังนี้

1. กำหนดพารามิเตอร์สร้างกลุ่มประชากร โดยมีพื้นที่การค้นหาตามชุดข้อมูลหน้าตัดเหล็กที่ได้ทำการสร้างไปในหัวข้อ 3.4.3 ซึ่งมีหน้าตัดเหล็ก 27 หน้าตัดจึงให้พื้นที่การค้นหาคือ ข้อมูลที่ตัวที่ 0-26 ตามตัวแปร lower_bound และ upper_bound

```
#Parameters
swarm_size = 10
iteration_num = 20
c1 = 1.5
c2 = 1.0
w = 0.8
#Search Space
lower_bound = 0
upper_bound = 26
dimension = 3
```

รูปที่ 3-30 พารามิเตอร์

นอกจากนี้ยังมีการกำหนดตัวแปร c_1, c_2, w, มิติการค้นหา (จำนวนตัวแปรออกแบบ), จำนวนครั้งในการทำซ้ำ , และขนาดกลุ่มอนุภาค และทำการสุ่มตำแหน่งเริ่มต้นของอนุภาคแต่ละตัวตามคำสั่งดังนี้

```
particles_position = np.zeros((swarm_size, dimension))
particles_velocity = np.zeros((swarm_size, dimension))
best_particle_position = np.zeros((swarm_size, dimension))
best_particle_fitness = np.zeros(swarm_size)
global_best_position = np.zeros(dimension)
global_best_fitness = math.inf
```

รูปที่ 3-31 สุ่มตำแหน่งเริ่มต้น

2. ประเมินคุณภาพประชากรสำหรับการคำนวณในรอบเริ่มต้นจะมีคำสั่งที่แตกต่างไปจากครั้งต่อ ๆ ไปที่ทำซ้ำ โดยมีคำสั่งดังนี้

```
for i in range(swarm_size):
    for j in range(dimension):
        particles_position[i, j] = int(random.uniform(lower_bound, upper_bound))
        particles_velocity[i, j] = random.uniform(-1, 1)

    Top = HSS[int(particles_position[i,0])]
    Bot = HSS[int(particles_position[i,1])]
    Dia = HSS[int(particles_position[i,2])]

    best_particle_position[i, :] = particles_position[i, :]
    best_particle_fitness[i] = objective_function(particles_position[i, :])
    if best_particle_fitness[i] < global_best_fitness:
        global_best_position = best_particle_position[i, :]
        global_best_fitness = best_particle_fitness[i]
```

รูปที่ 3-32 การประเมินคุณภาพเริ่มต้น

โดยจะทำการสุ่มตำแหน่งและความเร็วของอนุภาค และทำการหา *pbest* และ *gbest* ไปในตัวก่อนที่จะเกิดการทำซ้ำ

3. ขั้นตอนการทำซ้ำ ประเมินคุณภาพ และการอัปเดตประวัติตำแหน่งที่ดีที่สุดของอนุภาคและกลุ่มสามารถเขียนออกมาเป็นโค้ดได้ดังรูปที่ 3-33 โดยโปรแกรมจะทำการทำการคำนวณซ้ำจนกว่าจะครบจำนวนครั้งที่กำหนดเพื่อทำการออกแบบอย่างเหมาะสมออกมา และให้สามารถให้ผลลัพธ์เป็นน้ำหนักของโครงสร้างที่ดีที่สุด และตำแหน่งของหน้าตัดในชุดข้อมูลหน้าตัดเหล็กที่ได้สร้างไว้

```
#Iteration Process
for iter in range(iteration_num):
    for i in range(swarm_size):
        for j in range(dimension):

            r1 = random.random()
            r2 = random.random()

            cognitive_velocity = c1 * r1 * (best_particle_position[i, j] - particles_position[i, j])
            social_velocity = c2 * r2 * (global_best_position[j] - particles_position[i, j])
            particles_velocity[i, j] = w * particles_velocity[i, j] + cognitive_velocity + social_velocity

            particles_position[i, j] = int(particles_position[i, j] + particles_velocity[i, j])
            if particles_position[i, j] < lower_bound:
                particles_position[i, j] = lower_bound
            elif particles_position[i, j] > upper_bound:
                particles_position[i, j] = upper_bound

            Top = HSS[int(particles_position[i,0])]
            Bot = HSS[int(particles_position[i,1])]
            Dia = HSS[int(particles_position[i,2])]

        particle_fitness = objective_function(particles_position[i, :])

        if particle_fitness < best_particle_fitness[i]:
            best_particle_position[i, :] = particles_position[i, :]
            best_particle_fitness[i] = particle_fitness
        if best_particle_fitness[i] < global_best_fitness:
                global_best_position = best_particle_position[i, :]
                global_best_fitness = best_particle_fitness[i]
    print(global_best_fitness)

print("Best Truss Section: ", global_best_position)
print("Lowest Structure Weight: ", global_best_fitness,"kg")
```

รูปที่ 3-33 ขั้นตอนการทำซ้ำ

บทที่ 4
ผลการดำเนินงาน

4.1 การดำเนินการทดสอบการออกแบบอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่ม

จากการพัฒนาโปรแกรม Project-Python3D เพื่อให้สามารถออกแบบโครงสร้างโครงข้อหมุนเหล็กอย่างเหมาะสมได้ มีขั้นตอนการทำงานดังนี้

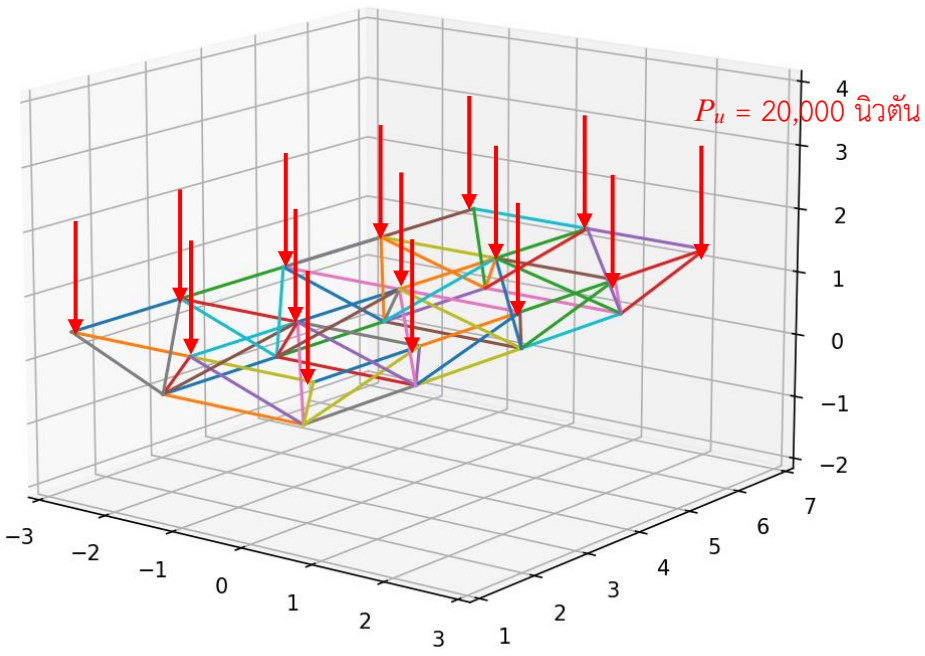
1. กำหนดวัสดุของโครงสร้างให้เป็นเหล็ก และกำหนดชุดข้อมูลหน้าตัดเหล็กท่อกลม
2. กำหนดพิกัดจุดต่อ และสร้างชิ้นส่วนของโครงสร้างโครงข้อหมุนเหล็กเชื่อมระหว่างพิกัดจุดต่อ
3. ใส่น้ำหนักบรรทุกที่พิกัดจุดแต่ละพิกัดได้รับ
4. กำหนดประเภท support ของโครงสร้างที่จุดต่าง ๆ
5. กำหนดพารามิเตอร์ต่าง ๆ ที่ใช้วิธีความฉลาดแบบกลุ่ม เช่น c_1, c_2, ขนาดกลุ่มอนุภาค เป็นต้น
6. ทำการคำนวณด้วยวิธีความฉลาดแบบกลุ่มเพื่อหาคำตอบ

โดยในโครงงานนี้จะทดลองการทำงานของวิธีความฉลาดแบบกลุ่มจากโครงสร้างโครงข้อหมุนเหล็กสามมิติ และเปรียบเทียบค่าการออกแบบอย่างเหมาะสมที่ได้ที่พารามิเตอร์ที่แตกต่างกันเพื่อเทียบผล และทำการเทียบประสิทธิภาพการทำงานเทียบกับโครงสร้างที่ออกแบบเหมาะสมด้วยวิธีการ Trial and Error โดยจะคำนวณออกแบบตามโครงสร้างตัวอย่างดังนี้

4.2 กรณีศึกษาการออกแบบอย่างเหมาะสม

4.2.1 กรณีศึกษาที่ 1

ตัวอย่างการออกแบบนี้จะออกแบบโครงสร้าง space truss 3 มิติมีระยะ span 6 เมตร ความสูง 1 เมตร ระยะห่าง support 2 เมตรที่โครงสร้างด้านล่างทั้ง 4 มุม โดยมีรูปร่างโครงสร้างดังรูป 4-1 โดยมีน้ำหนักบรรทุกปรับแก้ (P_u) ที่พิกัดจุดของโครงสร้างด้านบนจุดละ 20,000 นิวตันเป็น Point Load และให้ตัวแปรออกแบบ 3 หน้าตัดคือ โครงสร้างโครงข้อหมุนเหล็กด้านบน (Top Chord), โครงข้อหมุนด้านล่าง (Bottom Chord) และโครงข้อหมุนแนวทแยง (Diagonal Chord) โดยในกรณีตัวอย่างนี้เป็นตัวอย่างเพื่อทดสอบความสามารถในการออกแบบอย่างเหมาะสมในโครงสร้างที่ตัวแปรหน้าตัดน้อยของโปรงแกรม



รูปที่ 4-1 โครงสร้างกรณีศึกษาที่ 1

โดยตัวโปรแกรมมีขั้นตอนการ input ข้อมูลดังนี้

```
import numpy as np
from Material import Material
from Ball import Ball
from Structure import Structure
from Truss import Truss
from StructureAssemble import StructureAssemble
from Section import HollowBar
from matplot import matplot
from SteelSection import HSS
import math
import random
```

รูปที่ 4-2 การ import คำสั่งที่ใช้ในโปรแกรม

```
mat = Material()
mat.Steel()

Top = HSS[0]
Bot = HSS[0]
Dia = HSS[0]

BA1 = Ball(-1,0.707,0,'BA1')
BA2 = Ball(-1,2.707,0,'BA2')
BA3 = Ball(-1,4.707,0,'BA3')
BA4 = Ball(-1,6.707,0,'BA4')
BB1 = Ball(1,0.707,0,'BB1')
BB2 = Ball(1,2.707,0,'BB2')
BB3 = Ball(1,4.707,0,'BB3')
BB4 = Ball(1,6.707,0,'BB4')

TA1 = Ball(-1.707,0,1,'TA1')
TA2 = Ball(-1.707,1.8535,1,'TA2')
TA3 = Ball(-1.707,3.707,1,'TA3')
TA4 = Ball(-1.707,5.5605,1,'TA4')
TA5 = Ball(-1.707,7.414 ,1,'TA5')
TB1 = Ball(0,0,1,'TB1')
TB2 = Ball(0,1.8535,1,'TB2')
TB3 = Ball(0,3.707,1,'TB3')
TB4 = Ball(0,5.5605,1,'TB4')
TB5 = Ball(0,7.414 ,1,'TB5')
TC1 = Ball(1.707,0,1,'TC1')
TC2 = Ball(1.707,1.8535,1,'TC2')
TC3 = Ball(1.707,3.707,1,'TC3')
TC4 = Ball(1.707,5.5605,1,'TC4')
TC5 = Ball(1.707,7.414 ,1,'TC5')
```

รูปที่ 4-3 กำหนดพิกัดจุด (node)

```
def objective_function(self) :
    #Bot
    BA1BA2 = Truss(BA1,BA2,mat,Bot)
    BA1BB1 = Truss(BA1,BB1,mat,Bot)
    BA2BA3 = Truss(BA2,BA3,mat,Bot)
    BA2BB2 = Truss(BA2,BB2,mat,Bot)
    BA3BA4 = Truss(BA3,BA4,mat,Bot)
    BA3BB3 = Truss(BA3,BB3,mat,Bot)
    BA4BB4 = Truss(BA4,BB4,mat,Bot)
    BB1BB2 = Truss(BB1,BB2,mat,Bot)
    BB2BB3 = Truss(BB2,BB3,mat,Bot)
    BB3BB4 = Truss(BB3,BB4,mat,Bot)
```

รูปที่ 4-4 กำหนด objective_function (กำหนด Truss, Load และ support)

```
#Top
TA1TA2 = Truss(TA1,TA2,mat,Top)
TA1TB1 = Truss(TA1,TB1,mat,Top)
TA2TA3 = Truss(TA2,TA3,mat,Top)
TA2TB2 = Truss(TA2,TB2,mat,Top)
TA3TA4 = Truss(TA3,TA4,mat,Top)
TA4TA5 = Truss(TA4,TA5,mat,Top)
TA3TB3 = Truss(TA3,TB3,mat,Top)
TA4TA3 = Truss(TA4,TA3,mat,Top)
TA4TB4 = Truss(TA4,TB4,mat,Top)
TA5TB5 = Truss(TA5,TB5,mat,Top)
TC1TC2 = Truss(TC1,TC2,mat,Top)
TC2TC3 = Truss(TC2,TC3,mat,Top)
TC3TC4 = Truss(TC3,TC4,mat,Top)
TC4TC5 = Truss(TC4,TC5,mat,Top)
TC5TB5 = Truss(TC5,TB5,mat,Top)
TC4TB4 = Truss(TC4,TB4,mat,Top)
TC3TB3 = Truss(TC3,TB3,mat,Top)
TC2TB2 = Truss(TC2,TB2,mat,Top)
TC1TB1 = Truss(TC1,TB1,mat,Top)
TB1TB2 = Truss(TB1,TB2,mat,Top)
TB2TB3 = Truss(TB2,TB3,mat,Top)
TB3TB4 = Truss(TB3,TB4,mat,Top)
TB4TB5 = Truss(TB4,TB5,mat,Top)
```

```
#Diagonal&Vertical
BA1TB1 = Truss(BA1,TB1,mat,Dia)
BB1TB1 = Truss(BB1,TB1,mat,Dia)
BA1TB2 = Truss(BA1,TB2,mat,Dia)
BB1TB2 = Truss(BB1,TB2,mat,Dia)
BA1TA1 = Truss(BA1,TA1,mat,Dia)
BB1TC1 = Truss(BB1,TC1,mat,Dia)
BA2TA2 = Truss(BA2,TA2,mat,Dia)
BA3TA3 = Truss(BA3,TA3,mat,Dia)
BA4TA4 = Truss(BA4,TA4,mat,Dia)
BA4TA5 = Truss(BA4,TA5,mat,Dia)
BB4TB5 = Truss(BB4,TC5,mat,Dia)
BB2TC2 = Truss(BB2,TC2,mat,Dia)
BB3TC3 = Truss(BB3,TC3,mat,Dia)
BB4TC4 = Truss(BB4,TC4,mat,Dia)
BA1TA2 = Truss(BA1,TA2,mat,Dia)
BB1TC2 = Truss(BB1,TC2,mat,Dia)
BA2TA3 = Truss(BA2,TA3,mat,Dia)
BB2TC3 = Truss(BB2,TC3,mat,Dia)
BA3TA4 = Truss(BA3,TA4,mat,Dia)
BB3TC4 = Truss(BB3,TC4,mat,Dia)
TB2BA2 = Truss(TB2,BA2,mat,Dia)
TB2BB2 = Truss(TB2,BB2,mat,Dia)
TB3BA2 = Truss(TB3,BA2,mat,Dia)
TB3BB2 = Truss(TB3,BB2,mat,Dia)
TB3BA3 = Truss(TB3,BA3,mat,Dia)
TB3BB3 = Truss(TB3,BB3,mat,Dia)
TB4BA3 = Truss(TB4,BA3,mat,Dia)
TB4BB3 = Truss(TB4,BB3,mat,Dia)
TB4BA4 = Truss(TB4,BA4,mat,Dia)
TB4BB4 = Truss(TB4,BB4,mat,Dia)
TB5BA4 = Truss(TB5,BA4,mat,Dia)
TB5BB4 = Truss(TB5,BB4,mat,Dia)
```

```
#Add Load
TA1.load(0,0,0,-20000)
TA2.load(0,0,0,-20000)
TA3.load(0,0,0,-20000)
TA4.load(0,0,0,-20000)
TA5.load(0,0,0,-20000)
TB1.load(0,0,0,-20000)
TB2.load(0,0,0,-20000)
TB3.load(0,0,0,-20000)
TB4.load(0,0,0,-20000)
TB5.load(0,0,0,-20000)
TC1.load(0,0,0,-20000)
TC2.load(0,0,0,-20000)
TC3.load(0,0,0,-20000)
TC4.load(0,0,0,-20000)
TC5.load(0,0,0,-20000)

BA1.socket()
BB1.socket()
BA4.socket()
BB4.socket()

Structure.init()
t0=StructureAssemble()
t0.addSelfWeight()
t0.solve()
t0.Process('',0)
t0.subtractSelfWeight(0,0,0)

Truss.reset(self)
```

รูปที่ 4-5 กำหนด objective_function (กำหนด Truss, Load และ support) (ต่อ)

```
#Parameters
swarm_size = 10
iteration_num = 20
c1 = 1.0
c2 = 1.0
w = 0.5
#Search Space
lower_bound = 0
upper_bound = 26
dimension = 3
```

รูปที่ 4-6 กำหนดพารามิเตอร์ของ PSO

```
particles_position = np.zeros((swarm_size, dimension))
particles_velocity = np.zeros((swarm_size, dimension))
best_particle_position = np.zeros((swarm_size, dimension))
best_particle_fitness = np.zeros(swarm_size)
global_best_position = np.zeros(dimension)
global_best_fitness = math.inf

for i in range(swarm_size):
    for j in range(dimension):
        particles_position[i, j] = int(random.uniform(lower_bound, upper_bound))
        particles_velocity[i, j] = random.uniform(-1, 1)

    Top = HSS[int(particles_position[i,0])]
    Bot = HSS[int(particles_position[i,1])]
    Dia = HSS[int(particles_position[i,2])]

    best_particle_position[i, :] = particles_position[i, :]
    best_particle_fitness[i] = objective_function(particles_position[i, :])
    if best_particle_fitness[i] < global_best_fitness:
        global_best_position = best_particle_position[i, :]
        global_best_fitness = best_particle_fitness[i]
print(global_best_fitness)
```

รูปที่ 4-7 คำนวณกลุ่มอนุภาคเริ่มต้น

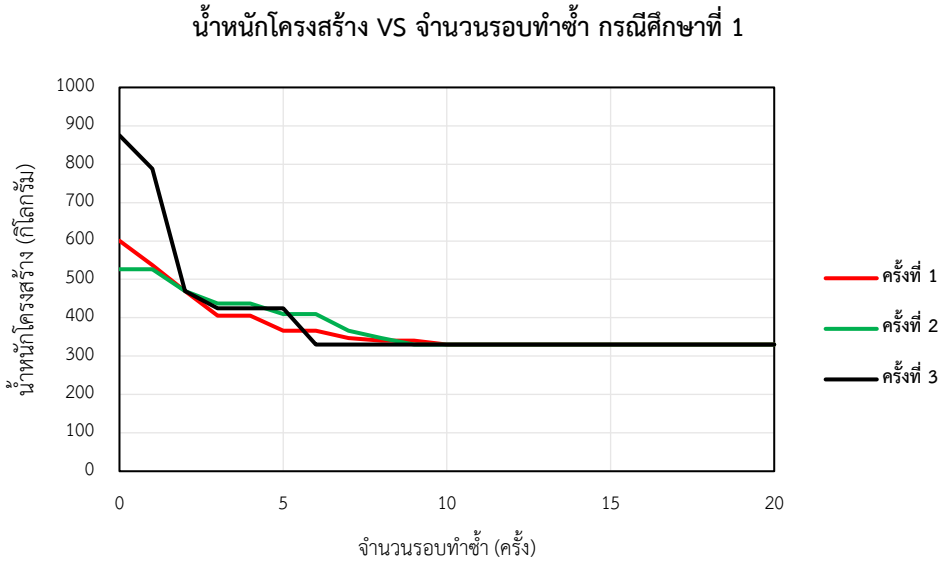
```
#Iteration Process
for iter in range(iteration_num):
    for i in range(swarm_size):
        for j in range(dimension):

            r1 = random.random()
            r2 = random.random()

            cognitive_velocity = c1 * r1 * (best_particle_position[i, j] - particles_position[i, j])
            social_velocity = c2 * r2 * (global_best_position[j] - particles_position[i, j])
            particles_velocity[i, j] = w * particles_velocity[i, j] + cognitive_velocity + social_velocity

            particles_position[i, j] = int(particles_position[i, j] + particles_velocity[i, j])
            if particles_position[i, j] < lower_bound:
                particles_position[i, j] = lower_bound
            elif particles_position[i, j] > upper_bound:
                particles_position[i, j] = upper_bound

            Top = HSS[int(particles_position[i,0])]
            Bot = HSS[int(particles_position[i,1])]
            Dia = HSS[int(particles_position[i,2])]

        particle_fitness = objective_function(particles_position[i, :])

        if particle_fitness < best_particle_fitness[i]:
            best_particle_position[i, :] = particles_position[i, :]
            best_particle_fitness[i] = particle_fitness
        if best_particle_fitness[i] < global_best_fitness:
                global_best_position = best_particle_position[i, :]
                global_best_fitness = best_particle_fitness[i]
    print(global_best_fitness)

print("Best Truss Section: ", global_best_position)
print("Lowest Structure Weight: ", global_best_fitness,"kg")
```

รูปที่ 4-8 การทำซ้ำและแสดงผลลัพธ์

```
899.2046071285944
453.97797356631906
453.97797356631906
453.97797356631906
346.67947627843154
346.67947627843154
339.97781826342316
339.97781826342316
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
330.0957123768854
Best Truss Section:  [4. 2. 5.]
Lowest Structure Weight:  330.0957123768854 kg
```

รูปที่ 4-9 ผลลัพธ์จากการเรียกใช้โปรแกรม

โดยผลลัพธ์ที่ได้ในแต่ละบรรทัดคือค่าที่ดีที่สุดของกลุ่มอนุภาคในการทำซ้ำแต่ละรอบ และผลสุดท้ายจะให้ผลเป็นหน้าตัดที่ดีที่สุด และน้ำหนักโครงสร้างที่น้อยที่สุดจากการคำนวณ โดยตัวอย่าง Section [4, 2, 5] คือตำแหน่งของข้อมูลหน้าตัดในไฟล์ชุดข้อมูลหน้าตัดดังภาพ 3-24 ที่ได้ทำการบันทึกข้อมูลหน้าตัดไว้



รูปที่ 4-10 ผลลัพธ์การออกแบบกรณีศึกษาที่ 1

ในการออกแบบเหมาะสมกรณีศึกษานี้ ให้พารามิเตอร์ของโปรแกรมดังนี้

ขนาดกลุ่มอนุภาค (n) = 10 อนุภาค

จำนวนครั้งทำซ้ำ = 20 ครั้ง

c_1 = 1.0

c_2 = 1.0

w = 0.5

จากการคำนวณโปรแกรมออกแบบอย่างเหมาะสม 3 ครั้ง พบว่าได้ผลลัพธ์เป็นคำตอบเป็นโครงสร้างหน้าตัดเดียวกัน คือ โครงสร้างมีน้ำหนักรวม 330.096 กิโลกรัม และได้หน้าตัดโครงสร้างดังนี้

Top Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 48.6 มิลลิเมตร หนา 2.3 มิลลิเมตร

Bottom Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 34.0 มิลลิเมตร หนา 2.3 มิลลิเมตร

Diagonal Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 48.6 มิลลิเมตร หนา 3.2 มิลลิเมตร

โดยเมื่อเทียบกับการออกแบบด้วยวิธีการ Trial and Error ได้หน้าตัดดังนี้

Top Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 48.6 มิลลิเมตร หนา 2.3 มิลลิเมตร

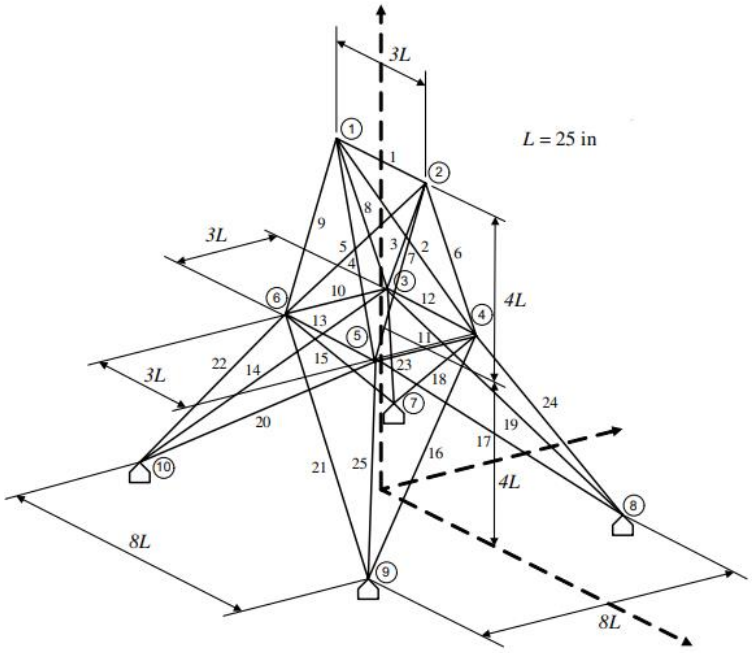
Bottom Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 34.0 มิลลิเมตร หนา 2.3 มิลลิเมตร

Diagonal Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 48.6 มิลลิเมตร หนา 3.2 มิลลิเมตร

ซึ่งทั้งสองการออกแบบได้คำตอบเหมือนกัน ในโครงสร้างที่ตัวแปรการออกแบบจำนวนน้อย (3 ตัวแปร) จะเห็นได้ว่าโปรแกรมออกแบบอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่มสามารถออกแบบโครงสร้างได้โดยผู้ใช้ไม่ต้องทำการแทนค่าหาคำตอบด้วยตนเองในโครงสร้างที่ตัวแปรหน้าตัดไม่มาก

4.2.2 กรณีศึกษาที่ 2

โครงสร้าง 25-bar Truss Tower โดย Schmit and Fleury ใช้ในการตรวจสอบประสิทธิภาพการทำงานของโปรแกรม เป็นตัวอย่างโครงสร้างที่ใช้อย่างแพร่หลายในการทดสอบระบบออกแบบอย่างเหมาะสมด้วยวิธีคำนวณต่าง ๆ เนื่องจากเป็นโครงสร้างที่มีความซับซ้อนสูง โดยมีตัวแปรการออกแบบหน้าตัด 8 ตัวแปร และการหาหน้าตัดการออกแบบที่ประหยัดที่สุดเป็นไปได้ยาก จึงเหมาะสมกับการทดสอบความแตกต่างของผลลัพธ์ของการออกแบบจากค่าพารามิเตอร์เริ่มต้นที่แตกต่างกันของวิธีความฉลาดแบบกลุ่ม



รูปที่ 4-11 25-bar Truss Tower

มีตัวแปรหน้าตัดของโครงข้อหมุนเหล็กดังนี้

กลุ่มหน้าตัด	ชิ้นส่วนโครงข้อหมุนเหล็ก
A1	1
A2	2-5
A3	6-9
A4	10, 11
A5	12, 13
A6	14-17
A7	18-21
A8	22-25

ตารางที่ 4-1 ข้อมูลตัวแปรหน้าตัด

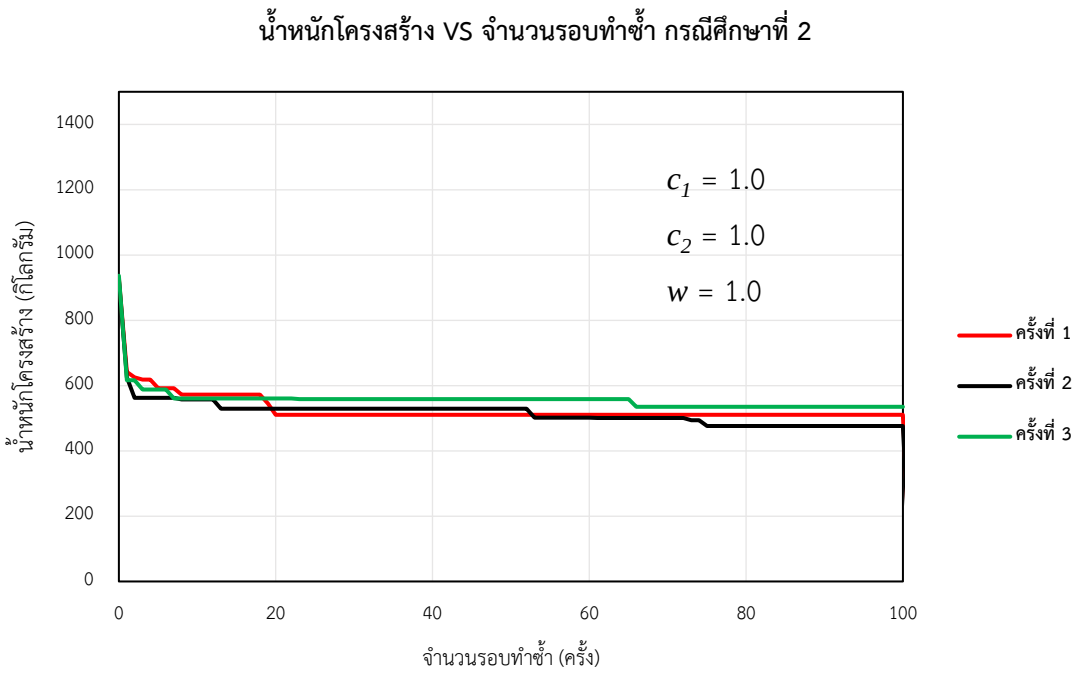
โดยมีแรงกระทำ P_u ที่ตำแหน่งต่าง ๆ ของโครงสร้างดังนี้

Node	F_x , (นิวตัน)	Fx , (นิวตัน)	Fz , (นิวตัน)
1	4448.22	-44482.22	-44482.22
2	0	-44482.22	-44482.22
3	2224.11	0	0
6	2668.93	0	0

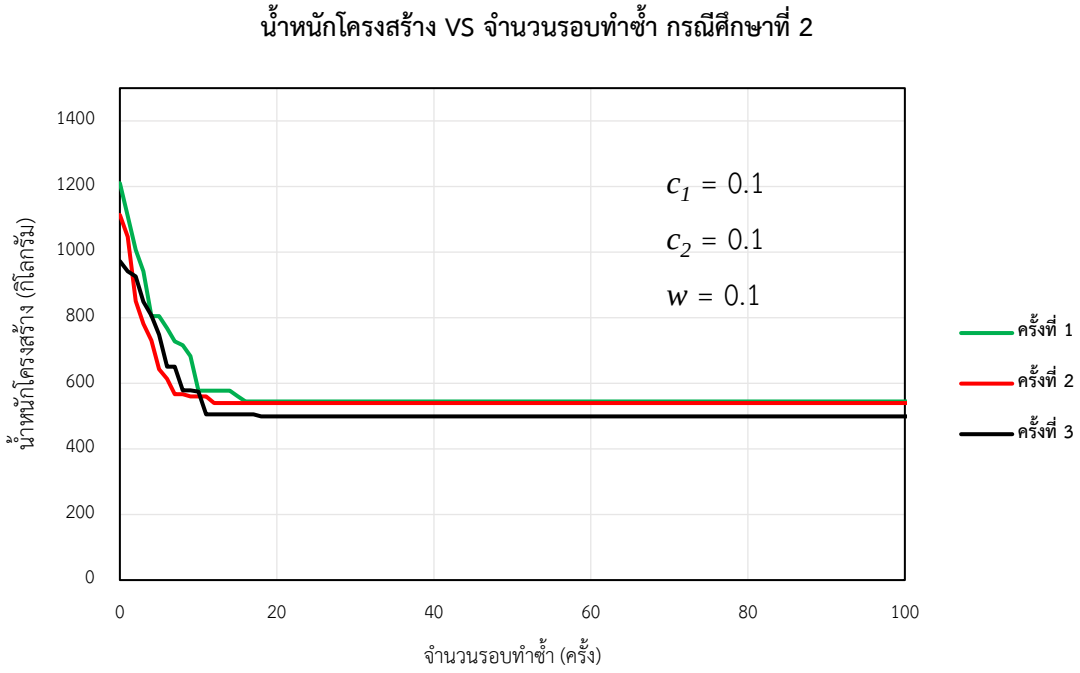
ตารางที่ 4-2 แรงกระทำต่อโครงสร้าง

ตามปกติแล้วโครงสร้างนี้ที่ใช้ในการออกแบบอย่างเหมาะสมจะมีการให้ค่า allowable stress, modulus of elasticity, material density และ displacement limit ของโครงสร้างมาให้ แต่ในโครงงานนี้เป็นการทดสอบตามมาตรฐาน AISC 360-16 ตามเหล็กท่อกลมมาตรฐาน มอก. และไม่ทำการคิดค่าการเคลื่อนตัวของพิกัดจุด ทำให้ไม่สามารถเปรียบเทียบค่าการออกแบบกับการทดลองในอดีตได้ จึงใช้โครงสร้างนี้ในการหาความแตกต่างของพารามิเตอร์แต่ละตัว

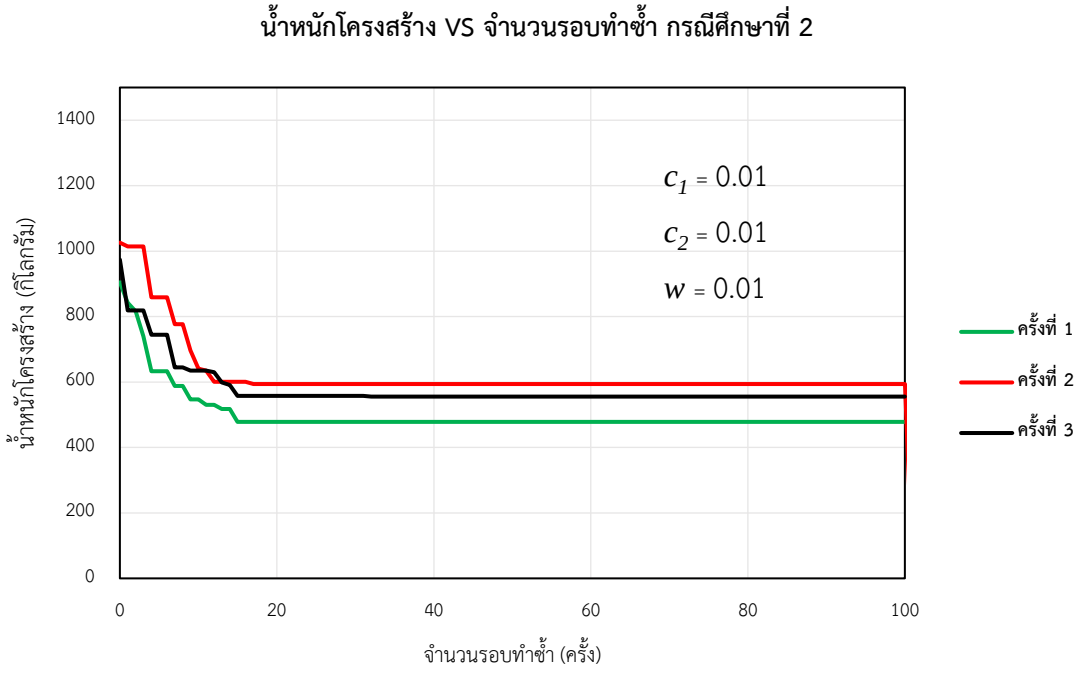
โดยจากการทดลองคำนวณออกแบบด้วยวิธีความฉลาดแบบกลุ่มด้วยค่าพารามิเตอร์ c_1, c_2 และ w ที่แตกต่างกัน และมีขนาดกลุ่มอนุภาค และจำนวนครั้งในการทำซ้ำเท่ากันคือ 40 อนุภาค และ 100 ครั้ง ตามลำดับ ได้ผลลัพธ์ดังนี้



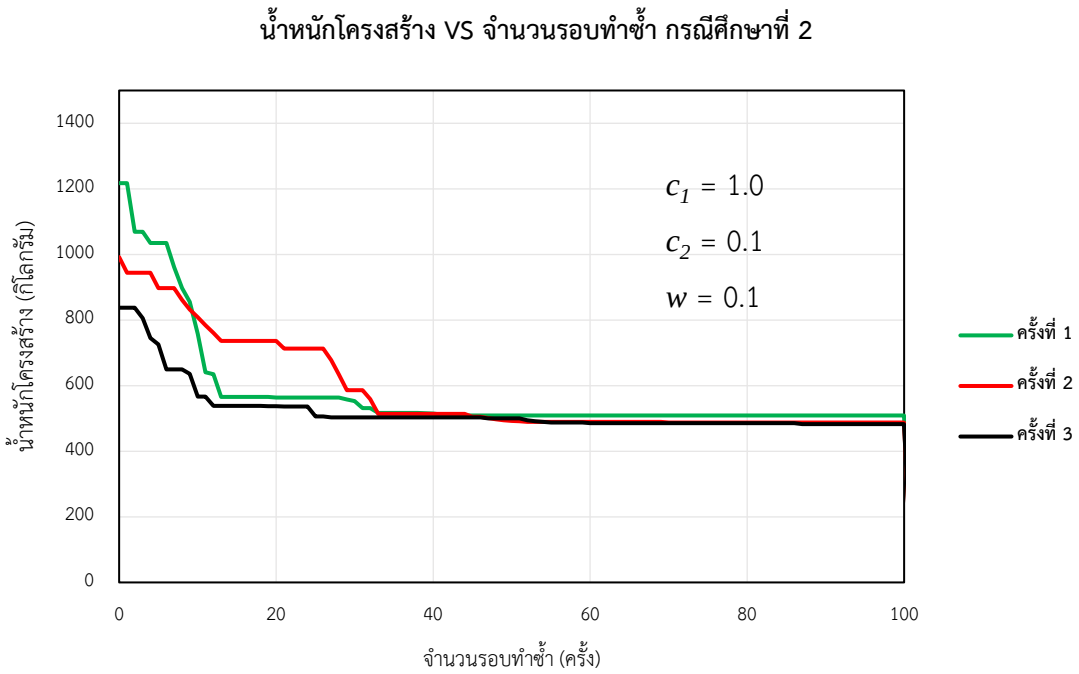
รูปที่ 4-12 c_1 = 1.0, c_2 = 1.0, w = 1.0



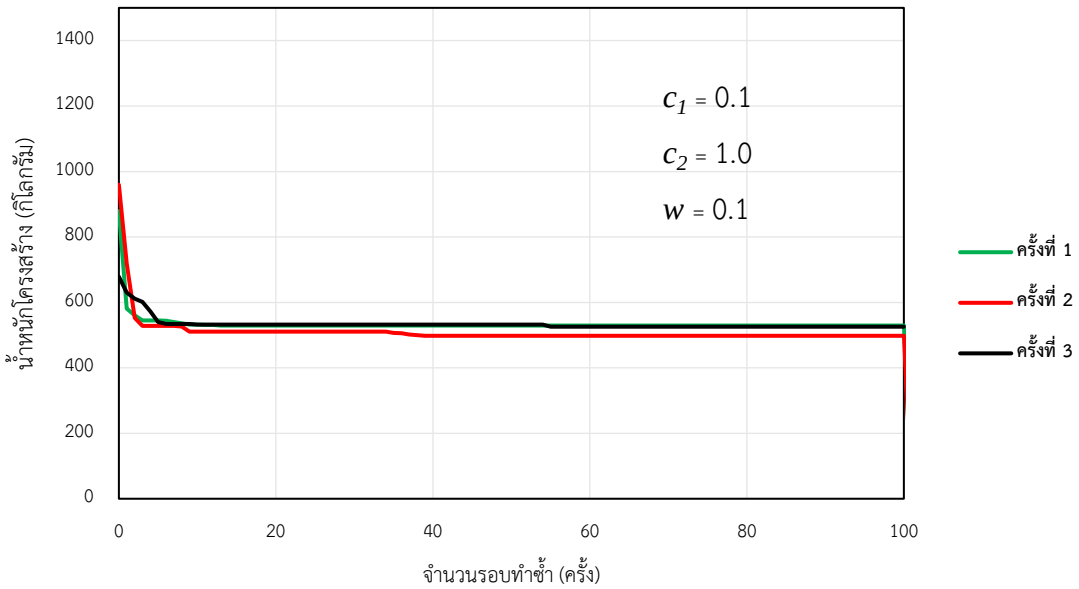
รูปที่ 4-13 c_1 = 0.1, c_2 = 0.1, w = 0.1



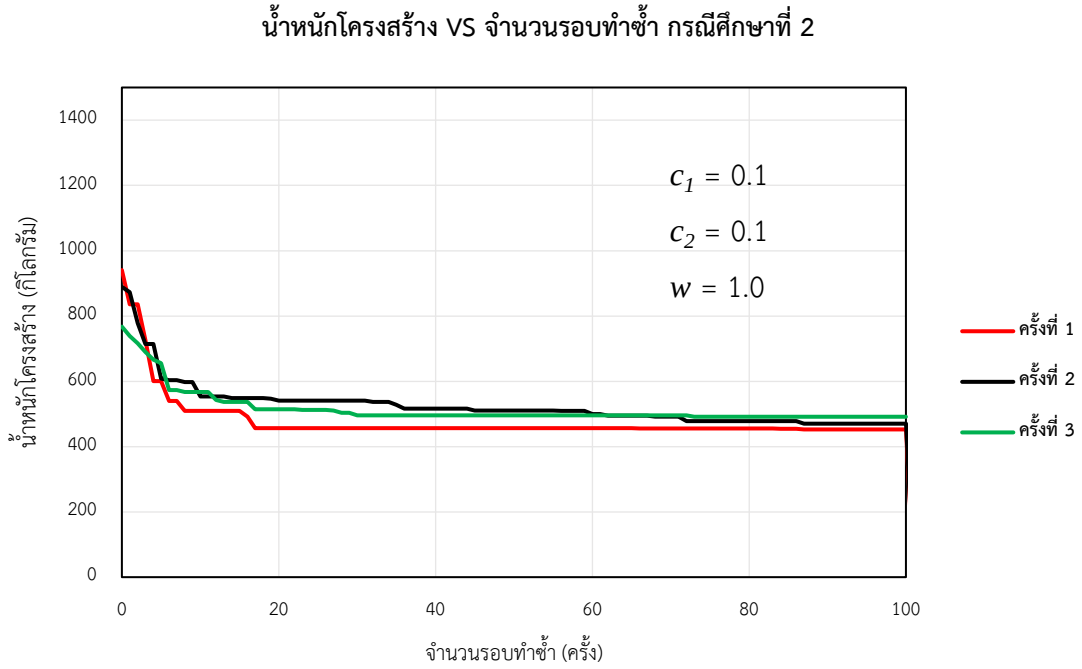
รูปที่ 4-14 c_1 = 0.01, c_2 = 0.01, w = 0.01



รูปที่ 4-15 c_1 = 1.0, c_2 = 0.1, w = 0.1



รูปที่ 4-16 c_1 = 0.1 c_2 = 1.0, w = 0.1



รูปที่ 4-17 c_1 = 0.1, c_2 = 0.1, w = 1.0

โดยจากแผนภาพเหล่านี้ เป็นการยกตัวอย่างความแตกต่างของพฤติกรรมการเปลี่ยนแปลงค่าออกแบบที่ดีที่สุดในแต่ละพารามิเตอร์ที่การทำซ้ำครั้งต่าง ๆ จะเห็นว่าพารามิเตอร์บางชุดสามารถเคลื่อนตัวเข้าหาผลลัพธ์ที่มีค่าต่ำที่สุดได้ในเวลาอันรวดเร็ว บางชุดก็เคลื่อนตัวได้ช้า แต่คุณภาพคำตอบที่ได้ก็จะแตกต่างกันตามพารามิเตอร์ที่เปลี่ยนไป แต่ถ้าหากต้องการทดสอบค่าพารามิเตอร์อย่างละเอียด สามารถทำได้ โดยในกรณีตังอย่างนี้ ปรับแต่งค่าพารามิเตอร์ที่ 0.25-1.00 ของค่าพารามิเตอร์แต่ละตัวและเปรียบเทียบ โดยที่ค่าพารามิเตอร์ต่าง ๆ จะทำการคำนวณ 2 ครั้งด้วยขนาดกลุ่มอนุภาค 40 อนุภาคและทำซ้ำ 100 ครั้ง ได้ดังนี้

c_1	c_2	w	น้ำหนักโครงสร้าง (กิโลกรัม)				c_1	c_2	w	น้ำหนักโครงสร้าง (กิโลกรัม)			
			ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	เฉลี่ย				ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	เฉลี่ย
0.25	0.25	0.25	497.30	498.03	466.54	487.29	0.75	0.25	0.25	473.15	458.21	461.30	464.22
0.25	0.25	0.50	506.54	485.51	480.61	490.88	0.75	0.25	0.50	461.30	441.42	475.63	459.45
0.25	0.25	0.75	466.15	441.42	441.42	449.66	0.75	0.25	0.75	475.63	436.63	462.54	458.27
0.25	0.25	1.00	439.78	470.70	456.78	455.75	0.75	0.25	1.00	462.54	447.73	460.09	456.79
0.25	0.50	0.25	515.41	511.29	511.29	512.67	0.75	0.50	0.25	491.34	486.75	489.49	489.19
0.25	0.50	0.50	461.52	452.40	470.84	461.58	0.75	0.50	0.50	489.49	476.85	441.42	469.25
0.25	0.50	0.75	436.63	436.63	436.63	436.63	0.75	0.50	0.75	441.42	463.54	462.62	455.86
0.25	0.50	1.00	495.73	475.67	503.91	491.77	0.75	0.50	1.00	462.62	468.22	466.38	465.74
0.25	0.75	0.25	463.99	487.90	489.18	480.36	0.75	0.75	0.25	436.63	467.34	501.50	468.49
0.25	0.75	0.50	463.99	463.99	441.42	456.46	0.75	0.75	0.50	501.50	463.99	436.63	467.37
0.25	0.75	0.75	436.63	436.63	436.63	436.63	0.75	0.75	0.75	436.63	436.63	436.63	436.63
0.25	0.75	1.00	496.87	488.55	450.75	478.72	0.75	0.75	1.00	436.63	515.10	480.27	477.33
0.25	1.00	0.25	441.42	441.42	511.08	464.64	0.75	1.00	0.25	458.68	436.63	436.63	443.98
0.25	1.00	0.50	436.63	436.63	436.63	436.63	0.75	1.00	0.50	436.63	436.63	441.42	438.22
0.25	1.00	0.75	436.63	436.63	436.63	436.63	0.75	1.00	0.75	441.42	436.63	436.63	438.22
0.25	1.00	1.00	488.56	462.81	497.78	483.05	0.75	1.00	1.00	436.63	535.61	475.88	482.70
0.50	0.25	0.25	480.56	490.60	520.25	497.14	1.00	0.25	0.25	436.63	441.42	441.42	439.82
0.50	0.25	0.50	441.42	490.28	469.29	467.00	1.00	0.25	0.50	441.42	452.28	441.42	445.04
0.50	0.25	0.75	436.63	436.63	455.38	442.88	1.00	0.25	0.75	441.42	436.63	436.63	438.22
0.50	0.25	1.00	496.31	450.12	490.02	478.82	1.00	0.25	1.00	436.63	474.23	495.93	468.93
0.50	0.50	0.25	436.63	493.08	452.22	460.64	1.00	0.50	0.25	452.40	441.42	471.32	455.05
0.50	0.50	0.50	514.04	441.42	486.98	480.81	1.00	0.50	0.50	471.32	441.42	458.78	457.17
0.50	0.50	0.75	436.63	436.63	436.63	436.63	1.00	0.50	0.75	458.78	436.63	436.63	444.01
0.50	0.50	1.00	505.76	503.72	467.97	492.48	1.00	0.50	1.00	436.63	456.33	456.23	449.73
0.50	0.75	0.25	495.24	531.31	436.63	487.72	1.00	0.75	0.25	474.24	468.34	452.40	464.99
0.50	0.75	0.50	436.63	436.63	447.60	440.28	1.00	0.75	0.50	452.40	436.63	436.63	441.88
0.50	0.75	0.75	447.60	436.63	447.60	443.94	1.00	0.75	0.75	436.63	436.63	436.63	436.63
0.50	0.75	1.00	447.60	513.05	495.88	485.51	1.00	0.75	1.00	436.63	483.04	477.99	465.88
0.50	1.00	0.25	462.54	462.54	458.78	461.29	1.00	1.00	0.25	436.63	436.63	436.63	436.63
0.50	1.00	0.50	462.54	436.63	458.78	452.65	1.00	1.00	0.50	436.63	436.63	462.54	445.26
0.50	1.00	0.75	458.78	436.63	436.63	444.01	1.00	1.00	0.75	462.54	436.63	436.63	445.26
0.50	1.00	1.00	436.63	465.02	536.36	479.34	1.00	1.00	1.00	436.63	480.96	560.41	492.66

ตารางที่ 4-3 ผลลัพธ์การออกแบบที่ค่าพารามิเตอร์ต่าง ๆ

จากผลลัพธ์การออกแบบที่ค่าพารามิเตอร์ต่าง ๆ ที่ได้ พบว่าที่ค่าพารามิเตอร์บางชุด จะมีการออกแบบอย่างเหมาะสมที่ผลลัพธ์ดีที่สุด คือ น้ำหนักโครงสร้างเท่ากับ 436.63 กิโลกรัมทุกครั้งที่เรียกใช้โปรแกรม จากการวิเคราะห์สามารถให้เหตุผลได้ว่า เนื่องจากการเรียกใช้ชุดข้อมูลของหน้าตัดโครงข้อหมุนเหล็กที่เป็นท่อกลม โดยชุดข้อมูลเป็นหน้าตัดของเหล็กต่าง ๆ ซึ่งตอนเรียกใช้ข้อมูลจะเรียกใช้เป็นตำแหน่งของข้อมูลที่เป็นจำนวนเต็ม คือ ข้อมูลตำแหน่งที่ 0 ถึง ตำแหน่งที่ 26 ทำให้เมื่ออนุภาคเคลื่อนตัวต้องเคลื่อนตัวในระยะที่เป็นจำนวนเต็มเท่านั้น ทำให้ค่าพารามิเตอร์ต่าง ๆ คูณกับระยะเวกเตอร์ความเร็วแล้วจะต้องทำการปัดเศษขึ้นหรือลง เพื่อให้ได้ความเร็วที่เป็นจำนวนเต็ม ด้วยเหตุนี้ทำให้ค่าพารามิเตอร์ชุดที่ใช้ต้องเป็นค่าที่ไม่มากหรือไม่น้อยเกินไป เพื่อประสิทธิภาพของผลลัพธ์ที่ดี

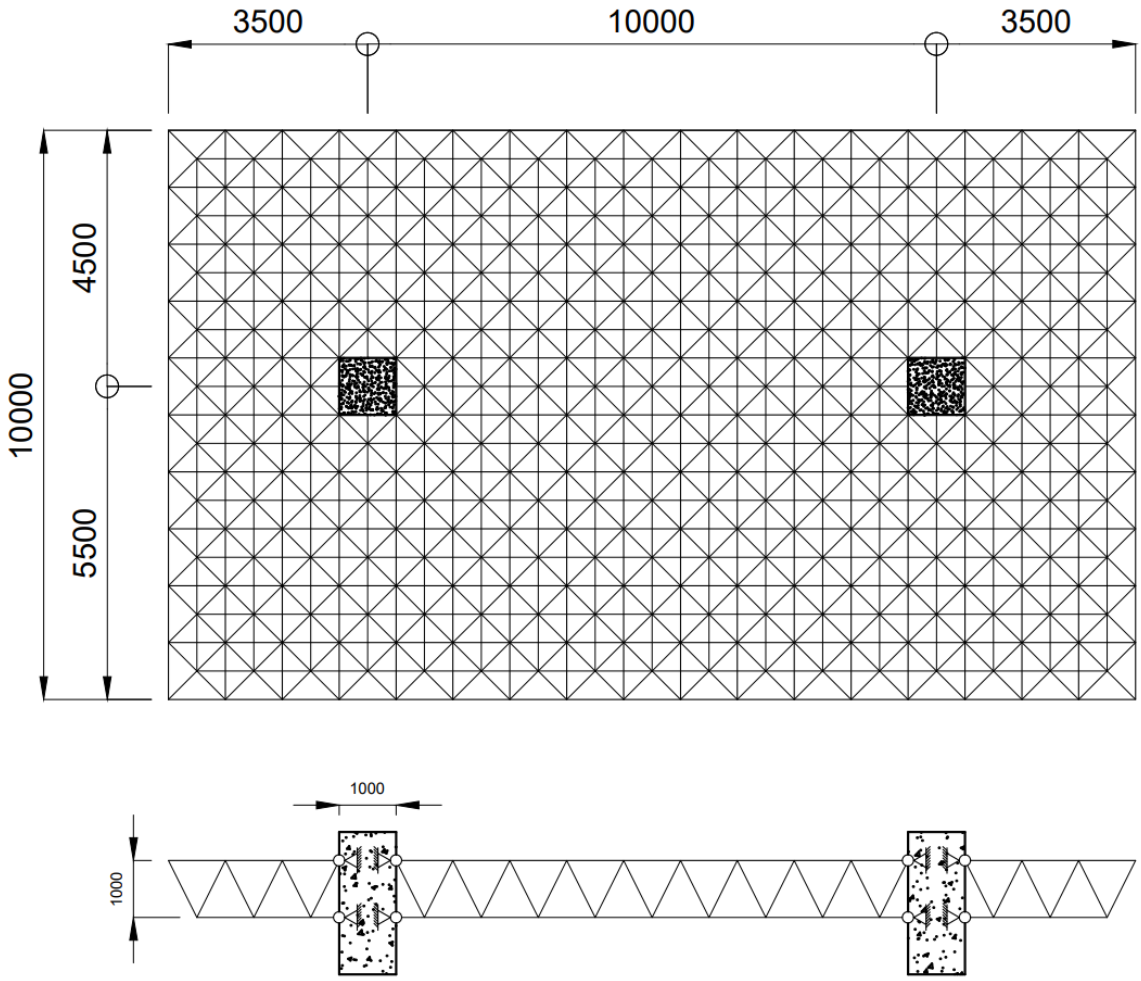
เมื่อทำการคำนวณด้วยพารามิเตอร์ทั้งหมดนี้ ทำให้ได้ค่าการออกแบบอย่างเหมาะสมโดยมีน้ำหนักโครงสร้าง 436.63 กิโลกรัมและมีหน้าตัดของโครงสร้างโครงข้อหมุนเหล็กดังนี้

กลุ่มหน้าตัด	หน้าตัดเหล็กท่อกลม (เส้นผ่านศูนย์กลาง x ความหนา)(มิลลิเมตร)
A1	21.7 x 2.0
A2	60.5 x 3.2
A3	76.3 x 3.2
A4	21.7 x 2.0
A5	21.7 x 2.0
A6	76.3 x 3.2
A7	76.3 x 3.2
A8	89.1 x 3.2

ตารางที่ 4-4 หน้าตัดเหล็กจากการออกแบบอย่างเหมาะสม

4.2.3 กรณีศึกษาที่ 3

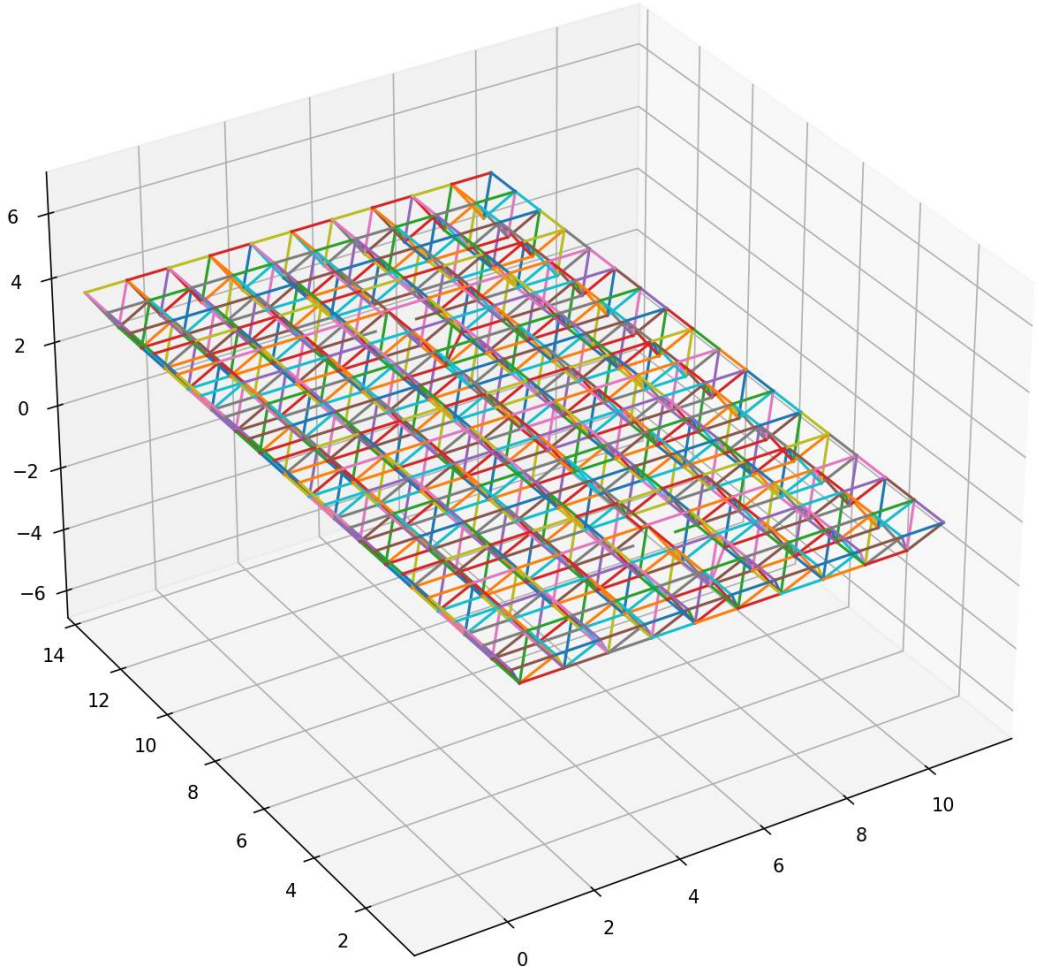
กรณีศึกษานี้เป็นการออกแบบโครงสร้างในการทำงานจริง คือ โครงสร้างโครงหลังคา Space Truss ที่ใช้เป็นกันสาดที่อาคารใหม่ของโรงเรียนสาธิต จุฬาลงกรณ์ มหาวิทยาลัย ฝ่านมัธยมศึกษา โดยโครงสร้างเป็นโครงสร้างโครงข้อหมุนเหล็กขนาดกว้าง 17 เมตร ยาว 10 เมตร โดยมีเสาอยู่ตรงกลาง 2 ต้นขนาด 1 เมตร x 1 เมตร และมี span สูงสุด 9 เมตรระหว่างเสาสองต้น โดยพิกัดจุดของโครงข้อหมุนเหล็กอยู่ห่างกัน 1 เมตรทั้งแนวตั้งและแนวนอน มีน้ำหนักบรรทุกจร (Live Load) ก่อนคูณค่าปรับแก้ 750 นิวตัน/ตารางเมตร โดยใส่เป็น Point Load ที่คูณค่าปรับแก้ 1.6 ที่ตำแหน่งพิกัดจุดของโครงข้อหมุนด้านบน ดังรูป



รูปที่ 4-18 โครงสร้างหลังค่า Space Truss

โดย support ของโครงสร้างนี้จะอยู่ที่จุดที่โครงข้อหมุนชนกับเสาพอดี เป็น pinned support ทั้งหมด 8 จุด

กรณีศึกษานี้เป็นโครงสร้างขนาดใหญ่ที่มีจำนวนชิ้นส่วนโครงข้อหมุนเหล็ก 1327 ชิ้นส่วน ซึ่งในขั้นตอนการคำนวณแรงภายในโครงสร้างของโปรแกรม Project-Python3D นั้นใช้เวลาในการคำนวณนานมากต่อการคำนวณฟังก์ชันจุดประสงค์ 1 ครั้ง แต่โครงสร้างนี้เป็นการยกตัวอย่างที่ดีของประสิทธิภาพในการทำงานของอัลกอริทึม PSO ที่ได้พัฒนาขึ้นมา เป็นเครื่องมือในการตรวจสอบการทำงานที่มีความซับซ้อนในการคำนวณสูง โดยในกรณีศึกษานี้จะแบ่งตัวแปรหน้าตัดเป็น 3 ตัวแปรคือ Top Chord, Bottom Chord และ Diagonal Chord ซึ่งเป็นการกำหนดหน้าตัดให้สามารถนำไปใช้ในการออกแบบและก่อสร้างจริงได้



รูปที่ 4-19 โมเดลสามมิติโครงสร้าง Space Truss

ด้วยเหตุผลที่กรณีศึกษานี้มีความซับซ้อนของโครงสร้างสูง จึงทำให้การเรียกใช้โปรแกรมในการออกแบบแต่ละครั้งใช้เวลาประมาณ 1 ชั่วโมง 30 นาทีเพื่อหาคำตอบการออกแบบเหมาะสมหนึ่งครั้ง และเป็นโครงสร้างที่น้ำหนักของโครงสร้างมีผลกับแรงภายในของโครงข้อหมุนเหล็กมาก เนื่องจากมีจำนวนโครงข้อหมุนเหล็กมาก ทำให้การหาคำตอบการออกแบบที่เหมาะสมที่สุดจึงใช้เวลานานมากถ้าหากต้องการออกแบบใหม่หลายครั้ง ในการหาคำตอบการออกแบบเหมาะสมของโครงสร้างนี้จึงทำการออกแบบซ้ำเพียง 10 รอบ โดยใช้ค่าพารามิเตอร์ที่ให้ผลลัพธ์ที่ดีที่สุดจากกรณีศึกษาที่ 2 คือ c_1 = 0.25, c_2 = 0.75 และ w = 0.75 และจากการออกแบบซ้ำ 10 รอบ ได้ผลลัพธ์การออกแบบดังนี้

การออกแบบ	น้ำหนักของโครงสร้าง	หน้าตัดเหล็กท่อกลม (เส้นผ่าศูนย์กลาง x ความหนา) มิลลิเมตร x มิลลิเมตร		
ครั้งที่	กิโลกรัม	Top Chord	Bottom Chord	Diagonal Chord
1	3164.27	34.0 x 2.3	48.6 x 3.2	34.0 x 2.3
2	**2749.72**	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
3	4215.37	27.2 x 2.3	101.6 x 3.2	34.0 x 2.3
4	3072.79	42.7 x 2.3	48.6 x 2.3	34.0 x 2.3
5	**2749.72**	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
6	3892.38	60.5 x 3.2	48.6 x 2.3	34.0 x 2.3
7	**2749.72**	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
8	**2749.72**	27.2 x 2.3	48.6 x 2.3	34.0 x 2.3
9	5725.37	114.3 x 3.2	48.6 x 3.2	34.0 x 2.3
10	3072.79	42.7 x 2.3	48.6 x 2.3	34.0 x 2.3

ตารางที่ 4-5 ผลลัพธ์การออกแบบกรณีศึกษาที่ 3

จากผลลัพธ์ทั้งหมดที่ได้ พบว่ากค่าการออกแบบเหมาะสมที่ประหยัดที่สุดของโครงสร้างนี้คือ

Top Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 27.2 มิลลิเมตร หนา 2.3 มิลลิเมตร

Bottom Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 48.6 มิลลิเมตร หนา 2.3 มิลลิเมตร

Diagonal Chord = เหล็กท่อกลมเส้นผ่านศูนย์กลาง 34.0 มิลลิเมตร หนา 2.3 มิลลิเมตร

ซึ่งจากการออกแบบซ้ำทั้ง 10 รอบ คำตอบการออกแบบที่ดีที่สุดมีอยู่ 4 ครั้งที่สามารถออกแบบได้ แสดงให้เห็นว่าโปรแกรมออกแบบด้วยวิธีความฉลาดแบบกลุ่มที่พัฒนาขึ้นจะมีประสิทธิภาพมากที่สุดถ้าทำการออกแบบซ้ำและวิเคราะห์หาคำตอบที่ดีที่สุด โดยจำนวนครั้งในการออกแบบซ้ำขึ้นอยู่กับความต้องการของผู้ใช้งาน

บทที่ 5
บทสรุป

5.1 สรุปผล

โครงงานนี้ได้พัฒนาโปรแกรมการออกแบบอย่างเหมาะสมด้วยวิธีเมตะฮิวริสติก โดยใช้วิธีความฉลาดแบบกลุ่ม (Particle Swarm Optimization, PSO) ในการออกแบบโครงสร้างโครงข้อหมุนเหล็กอย่างเหมาะสม เพื่อออกแบบโครงสร้างที่มีน้ำหนักเหล็กน้อยที่สุดเท่าที่ทำได้ ซึ่งหมายความว่าเป็นโครงสร้างที่ประหยัดค่าวัสดุได้มากที่สุด โดยได้ทำการออกแบบด้วยมาตรฐาน AISC 360-16 และใช้หน้าตัดเหล็กท่อกลมมาตรฐาน มอก. ของประเทศไทย เพื่อให้สามารถนำโปรแกรมที่พัฒนาขึ้นมาไปใช้ในงานการออกแบบจริงได้อย่างมีประสิทธิภาพ โดยโปรแกรมทำการคำนวณแรงภายในของโครงสร้างด้วยโปรแกรมตั้งต้น Project-Python3D และทำการออกแบบอย่างเหมาะสมด้วยวิธีความฉลาดแบบกลุ่ม ซึ่งทำงานคล้ายกับพฤติกรรมของฝูงสัตว์ที่มุงหน้าเข้าหาเป้าหมายของกลุ่มได้ผ่านการสื่อสารในรูปแบบบางรูปแบบ ในวิธีนี้มีขั้นตอนการดำเนินงานไม่ยาก ทำให้ใช้ทรัพยากรการทำงานของคอมพิวเตอร์น้อย และสามารถประแก้พารามิเตอร์ได้ตามต้องการ ขึ้นอยู่กับรูปแบบผลลัพธ์ที่ต้องการ วิธีนี้สามารถหาคำตอบของโครงสร้างที่มีจำนวนตัวแปรหน้าตัดน้อยและมากได้ โดยถ้าหากมีตัวแปรหน้าตัดน้อยจะสามารถหาคำตอบที่เหมาะสมได้ในการคำนวณซ้ำน้อยครั้ง แต่ถ้าหากต้องการหาคำตอบของการออกแบบที่ตัวแปรจำนวนมากอาจต้องทำการกำหนดการทำซ้ำหลายครั้ง และอาจต้องทำการออกแบบหลายครั้งเพื่อหาค่าที่ดีที่สุด ด้วยความที่โปรแกรมนี้เป็นโปรแกรม modified PSO แบบมีการปรับแต่งพัฒนาเพียงเล็กน้อย คือการใช้ค่า Inertia Weight, w ซึ่งเป็นตัวแปรตัวเดียวที่ถูกเพิ่มมาจาก PSO แบบดั้งเดิม ทำให้ความละเอียดและแม่นยำในการค้นหาคำตอบอาจไม่ดีเท่าอัลกอริทึมที่มีการปรับแต่งขั้นสูงกว่านี้ เช่น Hybrid PSO, Dynamic PSO และ Multi-Objective PSO เป็นต้น ซึ่งทำการพัฒนาอัลกอริทึมด้วยการผสมผสานความรู้จากอัลกอริทึมต่าง ๆ เข้าด้วยกัน ทำให้อัลกอริทึมที่ผู้ทดลองได้พัฒนาขึ้นมาอาจมีประสิทธิภาพไม่มากเท่าอัลกอริทึมเหล่านี้ แต่สามารถใช้ในการออกแบบโครงสร้างได้จริง เนื่องจาก การออกแบบโครงสร้างโครงข้อหมุนเหล็กในการทำงานจริง จะใช้ตัวแปรหน้าตัดจำนวนไม่มากเพราะเหตุผลในการทำงานก่อสร้างหน้างาน เนื่องด้วยถ้าหากตัวแปรมากเกินไปอาจทำให้การอ่านแบบเพื่อก่อสร้างสับสน และเกิดความผิดพลาด ซึ่งการออกแบบที่ตัวแปรไม่มากนี้ทำให้โปรแกรมนี้สามารถใช้งานได้อย่างมีประสิทธิภาพ และสะดวกต่อการใช้งาน ผู้ใช้งานสามารถปรับแต่งค่าคงที่ของโปรแกรม และสามารถ

ใส่ข้อมูลโครงสร้างได้ตามต้องการ จากนั้นสามารถกดเรียกใช้โปรแกรมและได้ output ออกมาเป็นหน้าตัดที่ถูกออกแบบอย่างเหมาะสม และน้ำหนักของโครงสร้างได้โดยง่าย และสามารถทำการคำนวณซ้ำได้ตามที่ต้องการ

ถ้าหากดูจากกรณีศึกษาที่ 1 จะเห็นว่าโปรแกรมสามารถคำนวณหน้าตัดเหมาะสมได้อย่างรวดเร็ว ด้วยการทำซ้ำเพียง 20 รอบ ซึ่งในการเรียกใช้โปรแกรมซ้ำไม่ว่าจะกี่ครั้งก็ได้คำตอบเท่าเดิม ตัวอย่างนี้แสดงให้เห็นถึงการใช้งานในการออกแบบโครงสร้างจริง ที่มีตัวแปรจำนวนน้อย และโครงสร้างไม่ซับซ้อนมาก ซึ่งสามารถใช้งานได้อย่างมีประสิทธิภาพและรวดเร็ว โดยการการคโดยจากกรณีศึกษาที่ 2 จะเห็นว่า การที่ผู้ใช้ปรับแต่งค่าพารามิเตอร์ต่าง ๆ นั้นขึ้นอยู่กับจุดประสงค์ที่ต้องการจากโปรแกรม โดยชุดพารามิเตอร์บางตัวสามารถให้ค่าออกแบบเหมาะสมได้ในการทำซ้ำน้อยครั้ง แต่คุณภาพคำตอบอาจไม่ดีเท่าชุดพารามิเตอร์ที่หาคำตอบได้ด้วยการทำซ้ำจำนวนหลายครั้ง นอกจากนี้ยังทำให้เห็นว่า ถ้าหากมีตัวแปรการออกแบบจำนวนมากจะทำให้โปรแกรมไม่สามารถหาคำตอบผลลัพธ์ที่ดีที่สุดได้อย่างแม่นยำ ต้องทำการเรียกใช้โปรแกรมหลายครั้ง เพื่อยืนยันว่าคำตอบที่ดีที่สุดมีค่าเท่าไร และถ้าพิจารณาที่กรณีศึกษาที่ 3 จะเห็นถึงประสิทธิภาพในการออกแบบโครงสร้างที่มีจำนวนชิ้นส่วนจำนวนมาก ซึ่งโปรแกรมสามารถคำนวณคำตอบในการทำซ้ำแต่ละรอบที่ใช้เวลานาน ซึ่งโปรแกรมสามารถออกแบบอย่างเหมาะสมได้แต่ไม่สามารถทำได้ในประสิทธิภาพที่ดีมาก ผู้ใช้อาจต้องรอผลลัพธ์เป็นเวลานาน ซึ่งเกิดจากขั้นตอนการคำนวณแรงภายในโครงสร้างที่ใช้เวลาในการคำนวณมาก เพราะต้องทำการคำนวณเมตริกที่มีมิติสูงมาก ทำให้การหาแรงภายในนาน นอกจากนั้นยังต้องทำการแทนค่าตามวิธีความฉลาดแบบกลุ่มทีมีจำนวนอนุภาคจำนวนหนึ่งและทำการคำนวณซ้ำอีกหลายรอบ จึงอาจไม่เหมาะสมสำหรับการออกแบบโครงสร้างที่ชิ้นส่วนเยอะมาก ๆ เท่าที่ควร

โดยจากการศึกษาจากการออกแบบอย่างเหมาะสมที่ได้ทำในโครงงานนี้ พบว่า การออกแบบอย่างเหมาะสมในโครงสร้างที่ไม่ซับซ้อน เช่น จำนวนชิ้นส่วนน้อยและตัวแปรหน้าตัดน้อยอย่างกรณีศึกษาที่ 1 การออกแบบด้วยวิธีดั้งเดิมอาจเป็นทางเลือกที่ง่ายสำหรับผู้ออกแบบที่ไม่ต้องการศึกษาขั้นตอนการใช้โปรแกรม เนื่องจากสามารถทำการแทนค่าหน้าตัดและคำนวณออกมาเพื่อตรวจสอบความปลอดภัยด้วยและทำซ้ำเพียง 2 – 3 ครั้งก็สามารถหาคำตอบการออกแบบอย่างเหมาะสมได้ แต่การใช้โปรแกรมก็สามารถประหยัดเวลาในการทำงานส่วนนี้ได้เช่นกัน แต่ถ้าหากทำการคำนวณ โครงสร้างที่เป็นแบบ Statically Indeterminate ที่แรงภายในโครงสร้างมีพฤติกรรมที่แตกต่าง และถ้าหากมีตัวแปรหน้าตัดจำนวนมาก การใช้โปรแกรมออกแบบด้วยวิธีความฉลาดแบบกลุ่มก็สามารถออกแบบอย่างมีประสิทธิภาพได้ถ้าทำการออกแบบหลายครั้งและนำผลลัพธ์มาเปรียบเทียบกัน เพื่อให้ได้โครงสร้างที่มีคุณภาพตามมาตรฐานความปลอดภัย และมีความประหยัดมากที่สุดเท่าที่เป็นไปได้

เอกสารอ้างอิง

Aboul Ella Hassanien, M.A. El-Shorbagy (2018), Particle Swarm Optimization from Theory to Applications, IGI Global.

American Institute of Steel Construction (2016), Specification for Structural Steel Buildings (ANSI/AISC 360-16), AISC Committee on Specifications.

Amir Hossein Gandomi, Gun Jin Yun and Siamak Talatahari (2014), Optimum design of tower structures using Firefly Algorithm, Struct. Design Tall Spec. Build.23, 350–361.

Anders Klarbring, Peter W. Christensen (2008), An Introduction to Structural Optimization, Linköping University.

A.W. Gebisa, H.G. Lemu(2017). A case study on topology optimized design for additive manufacturing. IOP Conf. Series: Materials Science and Engineering 276 (2017) 012026.

Dapei Tan, Dongshu Wang and Lei Liu (2018), Particle swarm optimization algorithm: an overview, Soft Comput (2018) 22:387–408.

Fleury C,Schmit L. (1980). Discrete-continuous variable structural synthesis using dual methods. AIAA J 1980;18:1515–24.

Frans vau den Bergh (2006), An Analysis of Particle Swarm Optimizers, University of Pretoria.

Gerhard Venter (2003), Particle Swarm Optimization, NASA Langley Research Center, Hampton, Virginia 23681-2199.

James Kennedy and Russel Eberhart (1995), Particle Swarm Optimization, Purdue School of Engineering and Technology Indianapolis, IN 46202-5160.

O. Hasançebi, d S. Çarbaş (2011), Ant Colony Search Method in Practical Structural Optimization, Int. J. Optim. Civil Eng., 2011; 1:91-105.

Jason Brownlee (2021), Optimization for Machine Learning , Machine Learning Mastery.

Kang Seok Lee , Zong Woo Geem, A new structural optimization method based on the harmony search algorithm, Comput Struct 2004;84(9-10):781-98.

K.Behdinan, R.E Perez (2007), Particle swarm approach for structural design optimization, Computers and Structures 85 (2007) 1579–1588.

Keith M. MacBain, William R. Spillers (2009), Structural Optimization, Springer Dordrecht Heidelberg London New York.

Raphael T. Haftka, Zafer Gürdal (1992), Elements of Structural Optimization, Springer-Science+Business Media, B.V.

Singiresu S. Rao (2009), Engineering Optimization Theory and Practice, พิมพ์ครั้งที่ 4, JOHN WILEY & SONS, INC.

W. M. Jenkins (1991), Towards Structural Optimization Via the Genetic Algorithm, Computers & Structure Vol. 40 No. 5, pp. 1321-1327, (1991).

ทักษิณ เทพชาตรี, อัครวัชร เล่นวารี (2016), พฤติกรรมและการออกแบบโครงสร้างเหล็ก ปรับปรุงครั้งที่ 4, สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย.