บทความวิจัย (Research Article)

แอปพลิเคชันการจับคู่ในกราฟถ่วงน้ำหนักสำหรับปัญหาการจัดสรรงาน

An Application for Weighted Graph Matching in Job Assignment Problems

โรจนี ขุมมงคล[1], จินตหรา จารัตน์[1], ดำรงค์ ถาวร[2], นฤมล แย้มอาสา[1] และ วัชรินทร รักษาศักดิ์ชัย[1*]

Rojanee Khummongkol[1], Chintara Charat[1], Damrong Thavorn[2], Naruemon Yaemarsa[1] and Watcharintorn Ruksasakchai[1*]

[1]สาขาวิชาคณิตศาสตร์และสถิติ ภาควิชาวิทยาการคำนวณและเทคโนโลยีดิจิทัล คณะศิลปศาสตร์และวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน อำเภอกำแพงแสน จังหวัดนครปฐม 73140

[2]สาขาวิชาการตลาด ภาควิชาบริหารธุรกิจและการบัญชี คณะศิลปศาสตร์และวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน อำเภอกำแพงแสน จังหวัดนครปฐม 73140

[1]Division of Mathematics and Statistics, Department of Computational Science and Digital Technology, Faculty of Liberal Arts and Science, Kasetsart University Kamphaeng Saen Campus, Nakhon Pathom, 73140

[2]Division of Business Administration in Marketing, Department of Business Administration and Accountancy, Faculty of Liberal Arts and Science, Kasetsart University Kamphaeng Saen Campus, Nakhon Pathom, 73140

[*]Corresponding author email: faaswtr@ku.ac.th

วันที่รับบทความ (Received)	วันที่ได้รับบทความฉบับแก้ไข (Revised)	วันที่ตอบรับบทความ (Accepted)
28 ตุลาคม 2569	18 มีนาคม 2569	20 มีนาคม 2569

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์หลักเพื่อออกแบบและพัฒนาแอปพลิเคชันสำหรับการหาการจับคู่ที่เหมาะสมที่สุดในกราฟถ่วงน้ำหนักเพื่อแก้ปัญหาการจัดสรรงาน พร้อมทั้งพัฒนาชุดคำสั่งการจับคู่ให้ครอบคลุม 5 ลักษณะสำคัญเพื่อเพิ่มความยืดหยุ่นในการเลือกคำตอบที่เหมาะสมในบริบทที่หลากหลาย ตลอดจนมุ่งวิเคราะห์และตรวจสอบคุณสมบัติเชิงโครงสร้างของกราฟในด้านการเป็นกราฟสองส่วนและการมีอยู่ของการจับคู่สมบูรณ์ โดยเครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชันคือภาษาไพทอน ซึ่งมีการทดสอบผ่านกลุ่มตัวอย่างที่เป็นกรณีศึกษาจำลองปัญหาการจัดสรรงานจำนวน 5 กรณีศึกษา สำหรับเครื่องมือที่ใช้ในการวิจัยประกอบด้วยแอปพลิเคชันสำหรับการหาการจับคู่ที่เหมาะสมในกราฟถ่วงน้ำหนักและแบบบันทึกผลการทดสอบความถูกต้องของขั้นตอนวิธี โดยใช้สถิติในการวิเคราะห์คือการหาค่าร้อยละของความถูกต้องและเปรียบเทียบผลลัพธ์กับคำตอบที่เหมาะสมที่สุด

ผลการวิจัยพบว่าแอปพลิเคชันที่พัฒนาขึ้นสามารถหาการจับคู่ที่เหมาะสมที่สุดในกราฟถ่วงน้ำหนักได้ครบถ้วนทั้ง 5 ลักษณะ ประกอบด้วย (1) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมมากที่สุด (2) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมน้อยที่สุด (3) การจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด (4) การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด และ (5) การจับคู่ใหญ่สุดโดยไม่คำนึงถึงค่าน้ำหนักรวม ซึ่งการระบุเป้าหมายในการพัฒนาแอปพลิเคชันและขอบเขตการหาการจับคู่ในกราฟถ่วงน้ำหนักที่ชัดเจนนี้ ช่วยให้สามารถแก้ปัญหาการจัดสรรงานได้อย่างมีประสิทธิภาพและมีความยืดหยุ่นสูงกว่าแอปพลิเคชันทั่วไป อีกทั้งยังสามารถตรวจสอบคุณสมบัติความเป็นกราฟสองส่วนและการมีอยู่ของการจับคู่สมบูรณ์ได้อย่างแม่นยำ ซึ่งช่วยลดความซับซ้อนใน การวิเคราะห์ข้อมูลกราฟและเป็นเครื่องมือที่มีประสิทธิภาพในการสนับสนุนการตัดสินใจทั้งในด้านวิชาการและการวางแผนเชิงธุรกิจ

คำสำคัญ: การจับคู่, กราฟถ่วงน้ำหนัก, ปัญหาการจัดสรรงาน, แอปพลิเคชัน

Abstract

This research primarily aims to design and develop an application for finding optimal matchings in weighted graphs to address assignment problems, along with developing matching algorithms that cover five key characteristics to enhance flexibility in selecting appropriate solutions across diverse contexts. Furthermore, it aims to analyze and verify the structural properties of graphs, specifically regarding bipartite graph detection and the existence of perfect matchings. The application was developed using the Python programming language and tested through five simulated case studies involving various assignment problems. The research instruments consist of the developed application for optimal matching in weighted graphs and an algorithm validation record. The statistical analysis involves calculating the percentage of accuracy and comparing the results with known optimal solutions.

The results indicate that the developed application successfully identifies optimal matchings in weighted graphs across all five characteristics: (1) maximum matching with the highest total weight, (2) maximum matching with the lowest total weight, (3) matching with the highest total weight, (4) matching with the lowest total weight, and (5) maximum matching regardless of total weight. By clearly defining the application's goals and the scope of weighted graph matching, the system facilitates efficient assignment problem-solving with greater flexibility compared to typical applications. Moreover, the system accurately verifies bipartite graph properties and the existence of perfect matchings, thereby reducing complexity in graph analysis and serving as an effective decision-support tool for both academic and business planning purposes.

Keywords: Matching, Weighted graph, Job assignment problem, Application

บทนำ

การจัดสรรทรัพยากรและการมอบหมายงานถือเป็นปัญหาพื้นฐานที่พบได้อย่างแพร่หลายในเชิงธุรกิจ ลักษณะของปัญหาดังกล่าวมักเกี่ยวข้องกับการตัดสินใจที่ซับซ้อนภายใต้ข้อจำกัดด้านเวลา ต้นทุน และปริมาณทรัพยากรที่มีอยู่อย่างจำกัด หนึ่งในแนวทางที่ได้รับการยอมรับอย่างกว้างขวางในการแก้ปัญหาประเภทนี้ คือการใช้แบบจำลองทางคณิตศาสตร์ร่วมกับเทคนิคเชิงคอมพิวเตอร์เพื่อค้นหาคำตอบที่เหมาะสมที่สุด

แนวคิดสำคัญประการหนึ่งในการสร้างแบบจำลองเชิงคณิตศาสตร์ คือ การจับคู่ในกราฟ (Matching in Graphs) ซึ่งหมายถึงกระบวนการเลือกชุดของเส้นเชื่อมที่ไม่ทับซ้อนกัน โดยที่แต่ละจุดยอดจะถูกจับคู่กับจุดยอดอื่นได้เพียงหนึ่งครั้ง แนวคิดนี้ถูกนำไปประยุกต์ใช้ในการแก้ปัญหาหลากหลาย เช่น การจับคู่ในตลาดแรงงาน การกำหนดตารางการทำงาน และปัญหาการขนส่ง อย่างไรก็ตาม ในสถานการณ์จริง ความสัมพันธ์ระหว่างทรัพยากรและงานมักไม่เท่ากัน การใช้กราฟถ่วงน้ำหนัก (Weighted Graph) จึงช่วยสะท้อนคุณลักษณะเชิงปริมาณ เช่น ค่าใช้จ่าย เวลา หรือความสำคัญของการจับคู่ได้อย่างชัดเจน การหาการจับคู่ที่เหมาะสมที่สุดในกราฟถ่วงน้ำหนักจึงหมายถึงการค้นหาชุดของเส้นเชื่อมที่ไม่ทับซ้อนกันซึ่งให้ผลรวมของค่าน้ำหนักสูงสุดหรือต่ำสุด ขึ้นอยู่กับเป้าหมายของปัญหา เพื่อให้เห็นภาพชัดเจน พิจารณาตัวอย่างดังต่อไปนี้ บริษัท Anon ต้องการจัดพนักงาน 4 คน ไปประจำ 4 สาขา โดยที่พนักงานแต่ละคนสามารถสร้างกำไรได้แตกต่างกันในแต่ละสาขา จะเห็นว่าการเลือกจับคู่ที่ต่างกันสามารถนำไปสู่ผลลัพธ์ที่แตกต่างกันได้ ดังนั้นการหาการจับคู่ที่เหมาะสมที่สุดจึงเป็นสิ่งจำเป็นในเชิงธุรกิจและอุตสาหกรรม ไม่ว่าจะเพื่อเพิ่มผลกำไร ลดต้นทุน หรือจัดสรรทรัพยากรให้มีประสิทธิภาพสูงสุด

ในการหาการจับคู่ที่เหมาะสมในกราฟสามารถดำเนินการได้โดยใช้ขั้นตอนวิธีที่หลากหลายตามลักษณะของปัญหา เช่น ขั้นตอนวิธีฮังกาเรียน (Hungarian Algorithm) ซึ่งเป็นเครื่องมือสำคัญในการแก้ปัญหาการจัดสรรงาน โดย Shahriar Tanvir Alam และคณะ [1] รวมถึง Jayasree T G และ Malavika V [2] ได้นำขั้นตอนวิธีนี้มาใช้เพื่อหาคำตอบที่เหมาะสมที่สุดในการบริหารจัดการทรัพยากร ซึ่งโดยทั่วไปขั้นตอนวิธีฮังกาเรียนจะใช้สำหรับการจับคู่ในลักษณะหนึ่งต่อหนึ่ง (One-to-One) แต่ในปี 2023 Fatemeh Rajabi-Alni และ Alireza Bagheri [3] ได้พัฒนาต่อยอดให้สามารถรองรับการจับคู่ในลักษณะหลายต่อหลาย (Many-to-Many) เพื่อตอบโจทย์ความต้องการและขีดความสามารถที่ซับซ้อนขึ้น

นอกจากนี้ยังมี ขั้นตอนวิธีฮอปครอฟท์–คาร์พ (Hopcroft–Karp Algorithm) ซึ่ง M. Usha Devi และคณะ [4] ได้นำมาประยุกต์ใช้ในทางการแพทย์เพื่อหาการจับคู่ระหว่างผู้บริจาคและผู้รับตับในรูปแบบกราฟสองส่วนเพื่อให้เกิดจำนวนคู่ที่เหมาะสมที่สุด และในกรณีของกราฟทั่วไปที่มีความซับซ้อน Gabriel Cristian และคณะ [5] ได้นำเสนอการใช้ขั้นตอนวิธีบลอสซัม (Blossom Algorithm) สำหรับจัดการทรัพยากรในสภาวะวิกฤตของระบบจัดการเหตุฉุกเฉินทางการแพทย์

อย่างไรก็ตาม แต่ละวิธีดังกล่าวยังมีข้อจำกัดที่แตกต่างกัน เช่น ขั้นตอนวิธีฮังกาเรียนที่เหมาะสำหรับกราฟสองส่วนเท่านั้น และโดยทั่วไปขั้นตอนวิธีเหล่านี้มักมุ่งเน้นการหาการจับคู่ที่ใหญ่ที่สุดหรือมีค่าน้ำหนักรวมสูงสุดเพียงด้านใดด้านหนึ่ง ซึ่งอาจไม่สามารถตอบโจทย์ปัญหาในสถานการณ์จริงที่มีความหลากหลายได้ครบถ้วน ด้วยเหตุนี้ งานวิจัยฉบับนี้จึงมุ่งเน้นการพัฒนาแอปพลิเคชันที่บูรณาการลักษณะการจับคู่ที่แตกต่างกันถึง 5 ลักษณะ เพื่อเพิ่มความยืดหยุ่นและประสิทธิภาพในการสนับสนุนการตัดสินใจให้ครอบคลุมทุกบริบทของปัญหา

วัตถุประสงค์ของการวิจัย

1. เพื่อออกแบบและพัฒนาแอปพลิเคชันสำหรับการหาการจับคู่ที่เหมาะสมที่สุดในกราฟถ่วงน้ำหนักสำหรับแก้ปัญหาการจัดสรรงาน
2. เพื่อพัฒนาชุดคำสั่งการจับคู่ให้ครอบคลุม 5 ลักษณะสำคัญ ซึ่งจะช่วยเพิ่มความยืดหยุ่นในการเลือกคำตอบที่เหมาะสมในบริบทที่หลากหลาย
3. เพื่อวิเคราะห์และตรวจสอบคุณสมบัติเชิงโครงสร้างของกราฟผ่านแอปพลิเคชัน โดยมุ่งเน้นระบบตรวจสอบความเป็นกราฟสองส่วนและการมีอยู่ของการจับคู่สมบูรณ์ เพื่อลดความซับซ้อนในการวิเคราะห์ข้อมูลกราฟขนาดใหญ่

วิธีดำเนินการวิจัย

งานวิจัยนี้ศึกษาและสร้างแอปพลิเคชันสำหรับกราฟไม่กำหนดทิศทางที่ไม่มีเส้นหลายชั้นและลูป [6, 7] กำหนดให้ G เป็นกราฟ และ $V(G), E(G)$ แทนเซตของจุดและเซตของเส้นเชื่อมของกราฟ G ตามลำดับ [8] จะกล่าวว่ากราฟ G เป็น *กราฟถ่วงน้ำหนัก (weighted graph)* เมื่อแต่ละเส้น $e \in E(G)$ มีจำนวนจริงกำกับ โดยจำนวนจริงเหล่านี้เรียกว่า *น้ำหนักของเส้น* ซึ่งเขียนแทนด้วย $w(e)$ กราฟ G จะถูกกล่าวว่าเป็น*กราฟสองส่วน (Bipartite graph)* ถ้าสามารถแบ่งกั้นเซต $V(G)$ ออกเป็น V_1 และ V_2 โดยที่ $V_1 \cup V_2 = V(G)$ และ $V_1 \cap V_2 = \emptyset$ และเส้นแต่ละเส้นใน G เป็นเส้นที่เชื่อมจุดที่อยู่ใน V_1 และ V_2 และเรียก (V_1, V_2) ว่า *เซตแบ่งกั้น (partite set)* ของ G

ถ้า $S \subseteq V(G)$ *ย่านเซต* (*Neighbor set*) ของ S ใน G เขียนแทนด้วย $N(S)$ คือเซตของจุดใน $V(G) - S$ ที่ประชิดกับจุดใน S ให้ $M \subseteq E(G)$ จะเรียก M ว่า *เซตจับคู่ M (M-matching set)* ใน G ถ้าสองเส้นเชื่อมใด ๆ ของ M ไม่มีจุดปลายร่วมกัน และสำหรับเส้นเชื่อม $e \in M$ ถ้า u, v เป็นจุดปลายของ e แล้วจะกล่าวว่า u *จับคู่* (*Saturates*) กับ v โดย M เซตจับคู่ M ใน G จะเรียกว่า*เซตจับคู่สมบูรณ์ M (M-perfect matching set)* ถ้าทุกจุดยอดของ G เป็นจุดยอดของเส้นเชื่อมของ M และเซตจับคู่ M ใน G จะถูกเรียกว่า *เซตจับคู่ใหญ่สุด (maximum matching set)* ถ้า $M' \subset M$ สำหรับทุก ๆ เซตจับคู่ M' ใน G

ให้ M เป็นเซตจับคู่ จะเรียกวิถี P ว่า *วิถีสลับของ M (M-alternating path)* ถ้าเส้นเชื่อมใน P ประกอบด้วยเส้นเชื่อมที่อยู่ใน M และไม่อยู่ใน M สลับกัน และจะเรียกวิถี P ว่า *วิถีแต่งเติมของ M (M-augmenting path)* ถ้า P เป็นวิถีสลับที่จุดปลายทั้งสองไม่ถูกจับคู่ใน M

ทฤษฎีบท 1 [6, 9, 10] (Berge's Theorem) เซตจับคู่ M ในกราฟ G เป็นเซตจับคู่ใหญ่สุด ก็ต่อเมื่อ G ไม่มีวิถีแต่งเติมของ M

ทฤษฎีบท 2 [6, 9, 10] (Hall's Theorem) ให้ G เป็นกราฟสองส่วนที่มี (X, Y) เป็นเซตแบ่งกั้น แล้ว G มีการจับคู่ซึ่งทุกจุดใน X ถูกจับคู่ ก็ต่อเมื่อ $|N(S)| \geq |S|$ สำหรับทุก $S \subseteq X$

ทฤษฎีบทข้างต้นนี้เป็นทฤษฎีพื้นฐานของขั้นตอนวิธีในการหาการจับคู่ใหญ่สุดในกราฟสองส่วนซึ่งมีหลักการดังต่อไปนี้ [10]

ขั้นตอนวิธีการหาการจับคู่ใหญ่สุดในกราฟสองส่วน (Matching Algorithm)

กำหนดให้ G เป็นกราฟสองส่วนที่มี (X, Y) เป็นเซตแบ่งกั้น

ขั้นตอนที่ 1 : สร้างเซตการจับคู่ M ใด ๆ

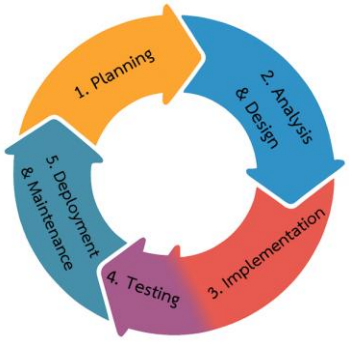
ขั้นตอนที่ 2 : ถ้าทุกจุดใน X ถูกจับคู่โดย M แล้วหยุด มิฉะนั้นแล้วให้ $u \in X$ เป็นจุดที่ไม่ถูกจับคู่โดย M และให้ $S = \{u\}$ และ $T = \emptyset$

ขั้นตอนที่ 3 : ถ้า $N(S) = T$ แล้วจะได้ว่า $|N(S)| < |S|$ เพราะว่า $|T| = |S| - 1$ ดังนั้นจึงหยุด เนื่องจากโดยทฤษฎีบทของฮาวว์จะได้ว่าไม่มีการจับคู่ซึ่งจุกทุดจุดใน X ถูกจับคู่ มิฉะนั้นแล้วให้ $y \in N(S) - T$

ขั้นตอนที่ 4 : ถ้า y เป็นจุดที่ถูกจับคู่โดย M แล้วให้ $yz \in M$ และแทนที่ S ด้วย $S \cup \{z\}$ และ T ด้วย $T \cup \{y\}$ แล้วกลับไปที่ขั้นตอนที่ 3 มิฉะนั้นให้ P เป็นวิถี $u - v$ ซึ่งเป็นวิถีแต่งเติมของ M แล้วจึงแทน M ด้วย $\widehat{M} = M \triangle E(P)$ (เมื่อ $M \triangle E(P) = (M - E(P)) \cup (E(P) - M))$ แล้วจึงกลับไปที่ขั้นตอนที่ 2

การวิเคราะห์และออกแบบระบบแอปพลิเคชันการจับคู่ในกราฟถ่วงน้ำหนักสำหรับปัญหาการจัดสรรงาน

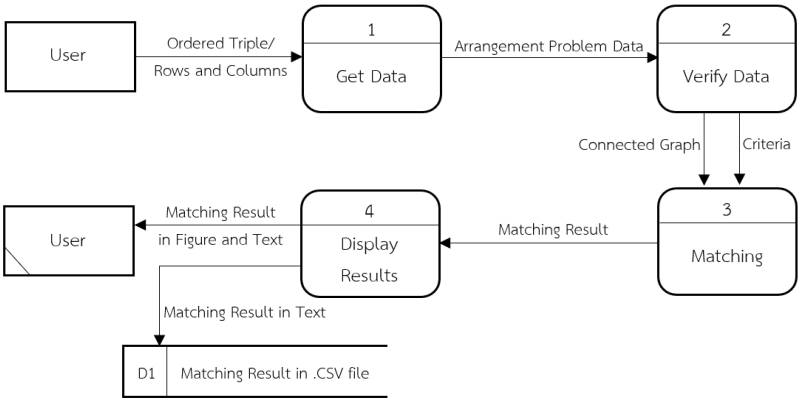
หลังจากที่กำหนดปัญหาแล้ว ขั้นตอนต่อไปคือการวิเคราะห์และออกแบบระบบคอมพิวเตอร์ที่ต้องการใช้เป็นเครื่องมือสำหรับแก้ปัญหาการจัดสรรงาน ซึ่งการพัฒนาระบบคอมพิวเตอร์จะถูกดำเนินการโดยใช้หลักการของวงจรชีวิตการพัฒนาซอฟต์แวร์ (Software Development Life Cycle: SDLC) ตามแบบจำลองบิกแบง (Big Bang Model) ซึ่งเหมาะต่อการพัฒนาระบบที่มีขนาดเล็ก และเป็นโครงการทดลอง โดยขั้นตอนของการพัฒนาระบบเพื่อแก้ปัญหาการจัดสรรงานนี้สามารถแบ่งออกได้เป็น 5 ขั้นตอน (ภาพที่ 1) ดังรายละเอียดต่อไปนี้



ภาพที่ 1 วงจรชีวิตการพัฒนาซอฟต์แวร์ของแอปพลิเคชันการจับคู่ในกราฟถ่วงน้ำหนัก

ขั้นตอนที่ 1 การวางแผนงาน (Planning) : ขั้นตอนนี้ถูกนำมาใช้เพื่อกำหนดวัตถุประสงค์ ขอบเขตการทำงาน และข้อจำกัดของระบบ รวมถึงทิศทางของการพัฒนาในขั้นตอนต่าง ๆ

ขั้นตอนที่ 2 การวิเคราะห์และออกแบบระบบ (System Analysis and Design): การวิเคราะห์และออกแบบการทำงานของระบบนี้ได้ถูกออกแบบดังขั้นตอนที่แสดงไว้ในภาพที่ 2 โดยแบ่งกระบวนการทำงานออกเป็น 4 ขั้นตอน ได้แก่ การรับข้อมูล (Get Data) การตรวจสอบข้อมูล (Verify Data) การจับคู่ (Matching) และการแสดงผล (Display Results) ตามลำดับ เนื่องด้วยระบบที่พัฒนาเป็นระบบขนาดเล็กจึงไม่มีการนำระบบการจัดการฐานข้อมูล (Database Management System: DBMS) เข้ามาใช้จัดเก็บข้อมูล สำหรับข้อมูลที่รับเข้ามาเพื่อทดสอบและประมวลผลนั้นได้มาจากผู้ใช้งานโดยตรง เนื่องจากมีจำนวนข้อมูลที่ไม่มากโดยเป็นการจำลองป้อนข้อมูลในลักษณะต่าง ๆ และผลลัพธ์ที่ได้จะถูกแสดงผลการจับคู่ออกทางหน้าจอ และจัดเก็บในรูปแบบของไฟล์เก็บข้อมูลรูปแบบตาราง (Comma Separated Value: CSV)



ภาพที่ 2 แผนภาพกระแสข้อมูลระดับที่ 1 ของการพัฒนาแอปพลิเคชันการจับคู่ในกราฟถ่วงน้ำหนัก

ขั้นตอนที่ 3 การพัฒนาระบบ (Implementation): สำหรับขั้นตอนของการพัฒนาระบบได้ถูกจัดทำตามแผนภาพกระแสข้อมูล (Data Flow Diagram: DFD) ที่ได้ออกแบบไว้ในขั้นตอนของการวิเคราะห์และออกแบบ (ภาพที่ 2) ภายใต้การเขียนโปรแกรมภาษาโปรแกรมไพทอน (Python Programming Language) เวอร์ชัน 3.12.4 โดยรายละเอียดในแต่ละกระบวนการของการประมวลผล (Process) ได้ถูกแสดงไว้ดังตารางที่ 1

ตารางที่ 1 กระบวนการในแต่ละโมดูล

ชื่อกระบวนการ	ข้อมูลเข้า	ข้อมูลออก	คำอธิบาย
1. Get Data	เซทของสามค่าที่มีลำดับ (Ordered Triple) หรือจำนวนแถวและหลัก	ข้อมูลสำหรับปัญหาการจัดเรียง (Arrangement Problem Data)	รับค่าเริ่มต้นเพื่อกำหนดโครงสร้างพื้นฐานของปัญหา
2. Verify Data	ข้อมูลสำหรับปัญหาการจัดเรียง	ข้อมูลกราฟเชื่อมโยง (Connected Graph Data)	ตรวจสอบว่าจุดยอดทุกจุดมีการเชื่อมต่อกันตามเงื่อนไขหรือไม่
3. Matching	ข้อมูลกราฟเชื่อมโยง และหลักเกณฑ์ (Criteria) ของวิธีการที่ต้องการจับคู่	ผลการจับคู่ (Matching Result)	ประมวลผลตามเงื่อนไขการจับคู่ทั้งสี่แบบ
4. Display Results	ผลการจับคู่	รูปภาพและข้อความแสดงผลการจับคู่ และไฟล์ข้อมูลตารางที่จัดเก็บผลการจับคู่	แสดงผลการจับคู่

ขั้นตอนที่ 4 การทดสอบระบบ (Testing): ในขั้นตอนการทดสอบระบบนี้ได้มีการทดสอบการทำงานในด้านต่าง ๆ เพื่อตรวจสอบความถูกต้องในแต่ละโมดูลโดยการใช้กรณีทดสอบ (Test Case) ซึ่งจะช่วยตรวจสอบความถูกต้องว่าระบบสามารถทำงานและให้คำตอบเป็นไปตามที่ต้องการหรือไม่ นอกจากนี้ยังมีการวิเคราะห์ถึงเวลากระทำการ (Execution Time) เพื่อทดสอบประสิทธิภาพของโมเดลว่า ใช้เวลาการทำงานเป็นอย่างไรเมื่อมีการสร้างจุดยอดจำนวนมาก ซึ่งขั้นตอนของการทดสอบโดยใช้กรณีทดสอบนี้จะถูกทำไปพร้อมกับขั้นตอนการพัฒนาระบบ เพื่อปรับปรุงระบบให้สามารถทำงานได้เป็นไปตามที่คาดหวังและได้ผลลัพธ์ที่ถูกต้องแม่นยำ

ขั้นตอนที่ 5: การติดตั้งและบำรุงรักษา (Deployment and Maintenance): เนื่องจากระบบที่ถูกพัฒนาขึ้นนี้จัดเป็นต้นแบบ (Prototype) จึงไม่มีการแบ่งขั้นตอนการติดตั้งและการบำรุงรักษาอย่างชัดเจน

การพัฒนาระบบนี้เป็นการพัฒนาแบบระบบอิสระโดยใช้ภาษาโปรแกรมไพทอนที่มีไลบรารีสนับสนุนการจัดการเกี่ยวกับกราฟ และข้อมูล อาทิ Networkx Pandas ซึ่งรวมถึง Tkinter ที่สนับสนุนการสร้างส่วนต่อประสานผู้ใช้งาน (User Interface: UI) โดยสถาปัตยกรรมระบบ (System Architecture) ของระบบที่สร้างขึ้นนี้เป็นสถาปัตยกรรมแบบแบ่งชั้น (Layered Architecture) ที่ง่ายต่อการจัดการ ซึ่งสามารถแบ่งออกได้ 3 ชั้น ได้แก่ ชั้นนำเสนอ ชั้นตรรกะ และชั้นข้อมูล ดังรายละเอียดต่อไปนี้

1. ชั้นนำเสนอ (Presentation Layer): ส่วนต่อประสานผู้ใช้ มีหน้าที่รับข้อมูลและแสดงผลแก่ผู้ใช้งาน
2. ชั้นตรรกะ (Logic Layer) หรือชั้นธุรกิจ (Business Layer): ส่วนการประมวลผลในกระบวนการทั้ง 4 ของแผนภาพกระแสข้อมูล
3. ชั้นข้อมูล (Data Layer): การจัดเก็บข้อมูลเป็นการจัดเก็บโดยใช้ทรัพยากรของเครื่องเนื่องจากเป็นระบบอิสระ โดยอยู่ในลักษณะของไฟล์ข้อมูลแบบตาราง (.CSV)

เนื่องจากขั้นตอนวิธีการหาการจับคู่ใหญ่สุดในกราฟสองส่วนข้างต้นนี้มีชื่อว่า*ขั้นตอนวิธีฮังกาเรียน* (Hungarian Algorithm) [10] จึงสามารสร้างรหัสเทียมของขั้นตอนวิธีฮังกาเรียนได้ดังต่อไปนี้

```
Function Hungarian_algorithm (Graph G, Set X, Set Y, Matching M)
    Mat_M = M
    LOOP
        IF ALL x in X are M-saturated THEN:
            STOP: M saturates X
            RETURN Mat_M
        END IF
        LET u is an M-saturated vertex in X
        S = {u}, T = ∅
        LOOP
            IF N(S) = T THEN:
                STOP: |N(S)| < |S|
                RETURN Mat_M
            END IF
            LET y is a vertex in N(S) \ T
            IF y is M-saturated THEN:
                LET yz is in M
                S = S ∪ {z}
                T = T ∪ {y}
                CONTINUE LOOP
            ELSE:
                Mat_M = Mat_M Δ E(P)
                M = Mat_M
                BREAK LOOP
            END IF
        END LOOP
    END LOOP
END FUNCTION
```

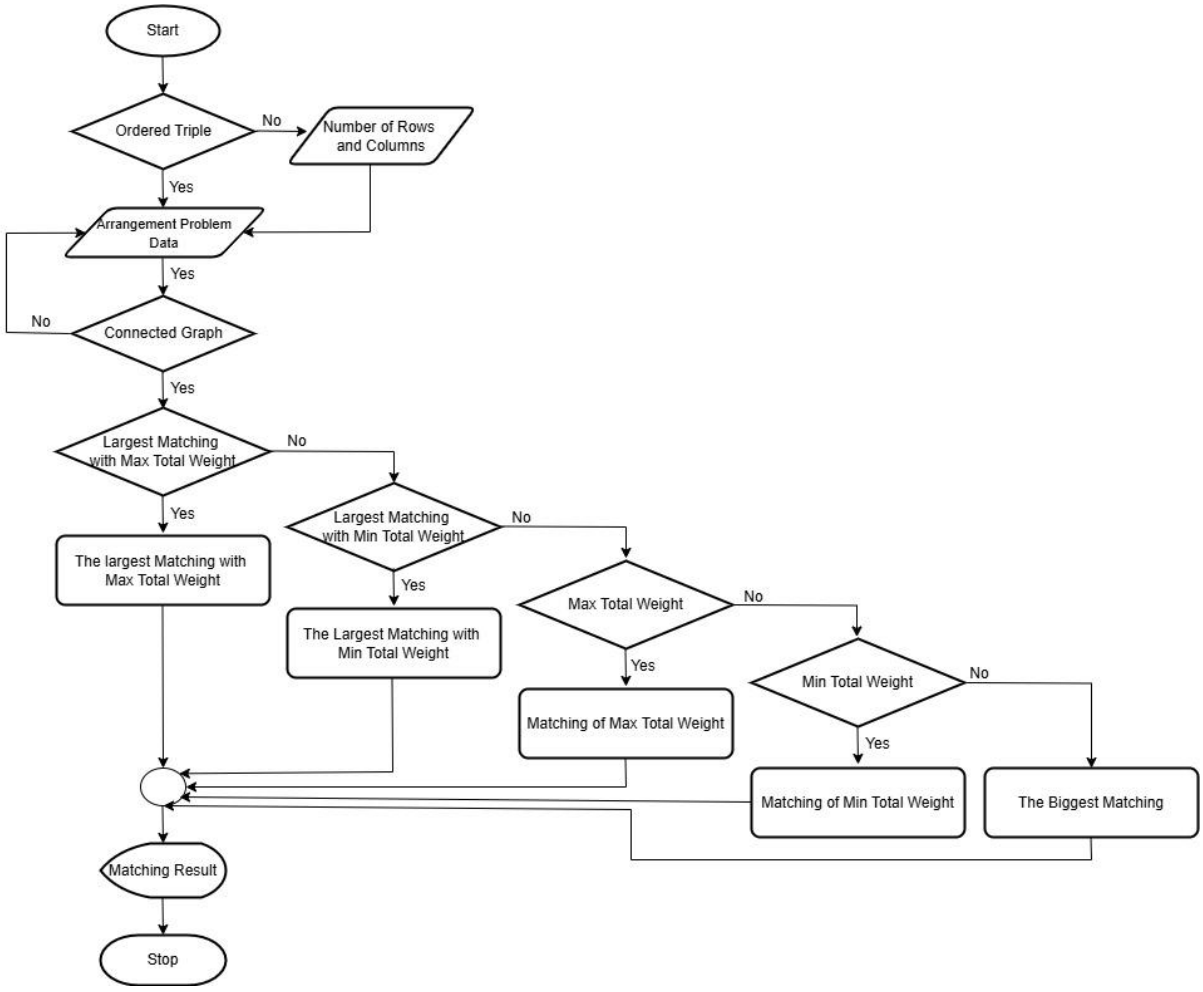
สำหรับหัวข้อถัดไปจะกล่าวถึงการพัฒนาแอปพลิเคชันเพื่อการจับคู่กราฟ [11] โดยอ้างอิงการออกแบบระบบที่กล่าวไว้ในข้างต้น

ผลการวิจัยและอภิปรายผล

1. การพัฒนาแอปพลิเคชันเพื่อการจับคู่กราฟ

การพัฒนาระบบเพื่อหาการจับคู่ในกราฟถ่วงน้ำหนักนี้ ผู้วิจัยได้มีการศึกษางานวิจัยที่เกี่ยวข้องเพื่อใช้เป็นองค์ความรู้พื้นฐานประกอบการพัฒนาระบบเพื่อให้ได้ชิ้นงานที่ให้มีความเหมาะสมและมีประสิทธิภาพสูงสุดดังตัวอย่างงานวิจัยต่อไปนี้ Akshitha, S. และคณะ [12] ได้นำเอาขั้นตอนวิธีของฮังกาเรียนเข้ามาใช้เพื่อแก้ปัญหาการเดินทางของพนักงาน (Traveling Salesman Problem) ในขณะที่ Wang, Y.และคณะ [13] ก็ได้นำเอาหลักการของฮังกาเรียนมาช่วยในการจัดสรรงานให้หุ่นยนต์ เช่นเดียวกับ Othman, F. และคณะ [14] ได้ทำการจับคู่โครงสร้างโปรตีนโดยใช้กราฟสองส่วน และ Rong, H. และ

คณะ [15] ได้นำกราฟสองส่วนเข้ามาช่วยในการจับคู่ภาพซึ่งการใช้ขั้นตอนวิธีการจับคู่จุดคุณลักษณะที่ดีที่สุดของกราฟแบบสองส่วนนั้นทำให้ได้ผลลัพธ์ที่มีความแม่นยำมากขึ้นเมื่อเทียบกับวิธีการดั้งเดิม โดยเมื่อได้ศึกษางานวิจัยที่ผ่านมาในอดีตประกอบกับการออกแบบระบบที่ได้วางไว้ การออกแบบผังงาน [16, 17] ก็ถือว่าเป็นองค์ประกอบที่สำคัญและจำเป็น เนื่องจากจะช่วยให้ผู้ที่พัฒนาระบบสามารถมองเห็น และเข้าใจหลักการทำงานในทุกส่วนของโปรแกรมได้อย่างถูกต้อง อีกทั้งยังทำให้เห็นถึงการรับค่าข้อมูลที่จำเป็น และการส่งข้อมูลออกเพื่อแสดงผล โดยภาพที่ 3 แสดงถึงลำดับขั้นตอนการทำงานผ่านแผนภาพผังงาน ซึ่งสามารถอธิบายขั้นตอนได้ว่า หลังจากที่เข้าสู่ระบบผู้ใช้ต้องทำการเลือกประเภทการนำเข้าข้อมูล (ได้แก่ การนำเข้าแบบสามสิ่งอันดับ (Ordered Triple) หรือการนำเข้าในรูปแบบตาราง) กรณีที่ผู้ใช้เลือกการนำเข้าข้อมูลในลักษณะของสามสิ่งอันดับ ค่าที่ใส่เข้ามาในระบบจะต้องประกอบไปด้วยค่า 3 ค่า ดังนี้ ชื่อจุดที่ 1 ชื่อจุดที่ 2 และค่าน้ำหนักของเส้นเชื่อมระหว่างจุดที่ 1 และจุดที่ 2 โดยจุด 1 และ 2 จะถูกกำหนดให้เป็นจุดของกราฟ ในขณะที่ค่าน้ำหนักจะเป็นสิ่งที่แสดงถึงความสำคัญของเส้นเชื่อมระหว่างจุดทั้งสอง แต่หากในขั้นตอนแรก ผู้ใช้เลือกการนำเข้าข้อมูลในรูปแบบข องตาราง ระบบจะแจ้งให้ผู้ใช้ระบุจำนวนแถวและหลักของตารางข้อมูลที่ต้องการ จากนั้นให้ทำการตั้งชื่อแถวและหลัก พร้อมใส่ค่าถ่วงน้ำหนักที่ต้องการลงในระบบตามลำดับ



ภาพที่ 3 ผังงานระบบการจับคู่กราฟถ่วงน้ำหนัก

เมื่อผู้ใช้นำเข้าข้อมูลกราฟสู่ระบบแล้ว ขั้นตอนถัดไปคือเลือกฟังก์ชันการทำงาน อันประกอบไปด้วย การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมมากที่สุด การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมน้อยที่สุด การจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด การจับคู่ที่มีค่า

น้ำหนักรวมน้อยที่สุด และการจับคู่ใหญ่สุด จากนั้นระบบจะมีการคำนวณเพื่อหาคำตอบจากฟังก์ชันการทำงานที่เลือกในลำดับต่อไป โดยผลลัพธ์ที่ได้จากการประมวลผลจะถูกแสดงออกทางหน้าจอ

ดังที่กล่าวไปแล้วในข้างต้น แอปพลิเคชันเพื่อการจับคู่ในกราฟถ่วงน้ำหนักในงานนี้ได้ถูกพัฒนา โดยอิงหลักการของขั้นตอนวิธีของฮังกาเรียน ภาพที่ 4 จึงเป็นการแสดงตัวอย่างโค้ดที่ถูกเขียนขึ้นเพื่อแก้ปัญหาการจับคู่ใหญ่สุดโดยใช้ขั้นตอนวิธีของฮังกาเรียน [18] ในงานวิจัยนี้ ในขณะที่ภาพที่ 5 แสดงตัวอย่างโค้ดที่พัฒนาขึ้นเพื่อการแก้ปัญหาการจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด ซึ่งหลักการทำงานของฟังก์ชัน sort_vertext นี้มีจุดประสงค์ในการจัดการข้อมูลการจับคู่ (ได้แก่ตัวแปร matching) โดยอิงตามเส้นเชื่อมในกราฟ (ตัวแปร edges) โดยตัวแปร choice ที่รับมาจะถูกใช้กำหนดตรรกะในการประมวลผล ได้แก่ “การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด” หรือกรณีอื่นที่ไม่ใช่การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด ในกรณีที่ตัวแปร choice เป็นการจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด ค่าน้ำหนักของเส้นเชื่อมจะถูกตัดทิ้งออกไป แล้วจึงนำค่าของจุดยอด 1 และ 2 เพิ่มลงในตัวแปร sort_matching แต่หากค่าของ choice ไม่ใช่การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด ระบบจะตรวจสอบว่าการจับคู่แต่ละคู่ว่ามีเส้นเชื่อมที่ถูกต้องหรือไม่ ถ้าใช่จะจัดเรียงทิศทาง/รูปแบบให้สอดคล้องกัน แต่หากไม่จะยังคงค่าเดิมไว้

```
def Hungarian_algor(G, X, Y, M):
    unsaturated_X = [u for u in X if u not in M or M.get(u) not in Y]
    if not unsaturated_X:
        return False, M
    for start_node in unsaturated_X:
        S = {start_node}; T = set()
        parent = {start_node: None}
        queue = deque([start_node])
        found_path = False
        while queue:
            u = queue.popleft()
            for v in G.get(u, []):
                if v not in T:
                    if v not in M or M.get(v) not in X:
                        path = []; curr = u
                        while curr is not None:
                            if parent[curr] is not None:
                                path.append((parent[curr], curr))
                            if curr in M and M[curr] is not None:
                                if curr == u:
                                    path.append((curr, v))
                                else:
                                    pass
                            curr = parent.get(parent.get(curr))
                        path_list = [v]
                        curr_node = v
                        while curr_node is not None:
                            prev = parent.get(curr_node)
                            if prev is not None:
                                path_list.append(prev)
                            curr_node = prev

                        path_list.reverse()
                        M_new = M.copy()
                        for i in range(len(path_list) - 1):
                            node1, node2 = path_list[i], path_list[i + 1]
                            if node1 in M_new and M_new[node1] == node2:
                                del M_new[node1]
                                if node2 in M_new and M_new[node2] == node1:
                                    del M_new[node2]
                            else:
                                M_new[node1] = node2
                                M_new[node2] = node1
                        return True, M_new

                    else:
                        T.add(v)
                        z = M[v]
                        if z not in S:
                            S.add(z)
                            parent[v] = u
                            parent[z] = v
                            queue.append(z)
    return False, M
```

ภาพที่ 4 โค้ดขั้นตอนวิธีฮังกาเรียน

```python
def sort_vertex(matching, edges):
    choice = choice_matching.get()
    vertex_data = [(edge[0], edge[1]) for edge in edges]
    sort_matching = []
    if choice == "การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด":
        for vertex in matching:
            if len(vertex) == 3:
                node1, node2 = vertex[:2]
                sort_matching.append((node1, node2))
            else:
                sort_matching.append(vertex)
        return sort_matching
    else:
        for vertex in matching:
            node1, node2 = vertex
            if (node1, node2) in vertex_data:
                sort_matching.append((node1,node2))
            elif (node2,node1) in vertex_data:
                sort_matching.append((node2,node1))
            else:
                sort_matching.append((vertex))
        return sort_matching
```

ภาพที่ 5 ตัวอย่างโค้ดเพื่อแก้ปัญหาการจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด

จากภาพที่ 4 ตัวแปร G ในฟังก์ชัน Hungarian_algor เป็นการสื่อถึงกราฟโดยที่ต้องการนำมาแก้ปัญหาจับคู่ โดยกำหนดเป็นโครงสร้างข้อมูลชนิดดิกชันารี ในขณะที่ X และ Y แทนเซตของจุดในฝั่งซ้ายและขวาตามลำดับ ส่วน M แทนการจับคู่ปัจจุบัน ซึ่งการค้นหาวิถีแต่งเติมของ M ในกราฟสองส่วนถูกทำโดยการค้นหาในแนวกว้าง (Breadth-first Search: BFS) ซึ่งเป็นขั้นตอนวิธีหนึ่งในการท่องกราฟ [19, 20] โดยผลลัพธ์ที่ได้จากการทำงานของขั้นตอนวิธีฮังกาเรียนนี้จะแบ่งออกได้เป็น 2 กรณีคือ 1) ผลลัพธ์เป็นจริงเมื่อพบวิถีแต่งเติมพร้อมการจับคู่ใหม่ และ 2) ผลลัพธ์เป็นเท็จหากไม่พบวิถีแต่งเติม

ในขั้นตอนของการพัฒนาระบบนี้ นอกเหนือจากการเขียนโปรแกรมเพื่อสร้างระบบแล้ว การทดสอบความถูกต้องก็เป็นสิ่งที่ต้องกระทำควบคู่กัน ตารางที่ 2-5 แสดงกรณีทดสอบสำหรับกระบวนการทำงานทั้ง 4 เพื่อตรวจสอบความถูกต้องและวัดประสิทธิภาพในการทำงานในด้านต่าง ๆ ดังนี้

ตารางที่ 2 กรณีทดสอบกระบวนการ Get Data

รหัสกรณีทดสอบ	รายละเอียดการทดสอบ	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง	สถานะ
001	การรับข้อมูลแบบเซทของสามค่าที่มีลำดับ	ชุดข้อมูล 3 ตัว เช่น (A, B, 2)	ระบบสามารถนำเข้าข้อมูลเพื่อดำเนินการต่อในกระบวนการที่ 2	ผ่าน
002	การรับข้อมูลจำนวนแถวและหลัก	ข้อมูลในลักษณะตาราง	ระบบสามารถนำเข้าข้อมูลเพื่อดำเนินการต่อในกระบวนการที่ 2	ผ่าน
003	การตรวจสอบหากรับข้อมูลไม่ตรงตามรูปแบบ	ข้อมูลลักษณะอื่นที่ไม่ใช่เซทของสามค่าที่มีลำดับ หรือจำนวนแถวและหลัก	ระบบแจ้งเตือนเมื่อรับข้อมูลไม่ตรงรูปแบบ	ผ่าน

ตารางที่ 3 กรณีทดสอบกระบวนการ Verify Data

รหัสกรณีทดสอบ	รายละเอียดการทดสอบ	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง	สถานะ
004	การสร้างกราฟเชื่อมโยง	ข้อมูลปัญหาการจัดเรียงจากขั้นตอนที่ 1	กราฟที่มีจุดยอดและเส้นเชื่อมครบถ้วนตามข้อมูลนำเข้า	ผ่าน
005	กรณีไม่สามารถสร้างกราฟเชื่อมโยง	ข้อมูลปัญหาการจัดเรียงจากขั้นตอนที่ 1 ที่มีจุดเชื่อมต่อไม่ครบ	ระบบแจ้งเตือนว่าไม่สามารถสร้างกราฟเชื่อมโยงได้	ผ่าน

ตารางที่ 4 กรณีทดสอบกระบวนการ Matching

รหัสกรณีทดสอบ	รายละเอียดการทดสอบ	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง	สถานะ
006	การจับคู่ตามเกณฑ์	กราฟเชื่อมโยง และเกณฑ์การจับคู่	ผลลัพธ์การจับคู่ที่ถูกต้องตามเงื่อนไขร้อยละ 100	ผ่าน
007	การจับคู่ที่ไม่เป็นไปตามเกณฑ์	กราฟเชื่อมโยงที่ขัดกับเกณฑ์การจับคู่	ระบบแจ้งว่าไม่สามารถจับคู่ได้	ผ่าน

ตารางที่ 5 กรณีทดสอบกระบวนการ Display Results

รหัสกรณีทดสอบ	รายละเอียดการทดสอบ	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง	สถานะ
006	การแสดงผลการจับคู่ในรูปแบบกราฟและข้อความทางหน้าจอ	ผลการจับคู่	ผลการจับคู่ในรูปแบบกราฟและข้อความ	ผ่าน
007	การบันทึกผลการจับคู่ในรูปแบบไฟล์ .CSV	ผลการจับคู่	ผลการจับคู่ในรูปแบบไฟล์	ผ่าน
008	ความสอดคล้องของการแสดงผลทางหน้าจอและไฟล์ .CSV	ผลการจับคู่	การแสดงผลทางหน้าจอและไฟล์ .CSV ต้องตรงกัน	ผ่าน

จากตารางที่ 4 จะเห็นได้ว่าแอปพลิเคชันการจับคู่ในกราฟถ่วงน้ำหนักสามารถจับคู่กราฟภายใต้เงื่อนไขต่าง ๆ ได้ถูกต้องร้อยละ 100 แต่หากพิจารณาถึงเวลากระทำการจะพบว่าการสร้างกราฟโดยการใช้วิธีการคำนวณเพื่อหาตำแหน่งของจุดยอดโดยการใช้ขั้นตอนวิธี Fruchterman-Reingold (จากการใช้คำสั่ง Networkx.spring_layout) ในการพัฒนาแอปพลิเคชัน) มีความซับซ้อนด้านเวลา (Time Complexity) เท่ากับ $O(|V|^2 + |E|)$ ต่อการวนซ้ำหนึ่งรอบ ดังนั้นหากต้องการสร้างกราฟที่มีจุดยอดจำนวนไม่มากจะสามารถทำงานได้อย่างรวดเร็ว แต่ถ้าต้องการสร้างกราฟมีจุดยอดจำนวนมากจะส่งผลต่อเวลาในการประมวลผลอย่างหลีกเลี่ยงไม่ได้

2. ผลลัพธ์การจับคู่โดยใช้แอปพลิเคชันเพื่อการจับคู่ในกราฟถ่วงน้ำหนัก

จากผังงานการพัฒนาระบบและขั้นตอนวิธีของการจับคู่ในหัวข้อที่ผ่านมา การพัฒนาแอปพลิเคชันเพื่อจับคู่กราฟนี้ได้จัดวางวัตถุประสงค์ของการจับคู่ไว้ 5 รูปแบบ ได้แก่ (1) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมมากที่สุด (2) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมน้อยที่สุด (3) การจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด (4) การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด และ (5) การจับคู่ใหญ่สุดโดยไม่คำนึงถึงค่าน้ำหนักรวม โดยการจับคู่แต่ละประเภทมีความเหมาะสมกับลักษณะของปัญหาที่แตกต่างกันไป ดังนี้

(1) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมมากที่สุด เป็นการจับคู่ที่เน้นให้เซตการจับคู่มีขนาดใหญ่ที่สุด โดยที่ค่าน้ำหนักรวมต้องมากที่สุดเช่นกัน ซึ่งการจับคู่แบบนี้เหมาะกับปัญหาที่ต้องการหาการจับคู่ที่ใหญ่ที่สุดและมีคำตอบหรือผลกำไรสูงที่สุด

(2) การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมน้อยที่สุด เป็นการจับคู่ที่คล้ายกับรูปแบบที่ (1) แต่แตกต่างตรงที่การจับคู่ในรูปแบบนี้ต้องการให้ค่าน้ำหนักรวมมีค่าน้อยที่สุด ซึ่งเหมาะสมกับปัญหาที่เน้นการหาต้นทุนต่ำที่สุด

(3) การจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด เป็นการจับคู่ที่มุ่งเน้นให้ผลรวมของค่าน้ำหนักทั้งหมดมีค่ามากที่สุด โดยไม่จำเป็นต้องได้เซตการจับคู่ที่มีขนาดใหญ่ที่สุด วิธีนี้เหมาะกับปัญหาที่ต้องการผลตอบแทนหรือประสิทธิภาพโดยรวมสูงสุด เช่น การจัดสรรงานให้ตรงกับความถนัดของแต่ละบุคคล แม้อาจมีบางคนที่ไม่ได้รับการจัดสรรงานก็ตาม

(4) การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด เป็นการจับคู่ที่มุ่งลดผลรวมของค่าน้ำหนักให้มีค่าน้อยที่สุด โดยไม่จำเป็นต้องได้เซตการจับคู่ที่ใหญ่ที่สุด วิธีนี้เหมาะกับปัญหาที่เน้นการลดต้นทุนหรือความสูญเสียให้น้อยที่สุด แม้อาจมีบางองค์ประกอบที่ไม่ได้รับการจับคู่ก็ตาม

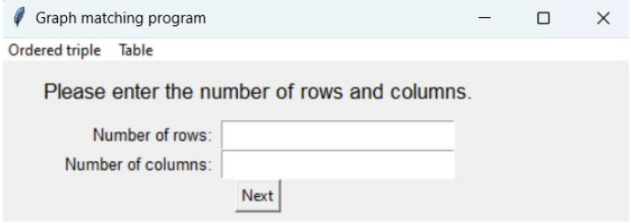
(5) การจับคู่ใหญ่สุดโดยไม่คำนึงถึงค่าน้ำหนักรวม เป็นการจับคู่ที่มุ่งเน้นให้เซตการจับคู่มีขนาดใหญ่ที่สุดเท่าที่เป็นไปได้ โดยไม่พิจารณาค่าน้ำหนักของเส้นเชื่อม

ซึ่งตัวอย่างการใช้งานแอปพลิเคชันและผลการจับคู่ได้ถูกแสดงและอธิบายไว้ดังรายละเอียดต่อไปนี้

4.1 การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมมากที่สุด

เมื่อผู้ใช้งานได้เข้าสู่ระบบ ก่อนที่จะดำเนินการหาการจับคู่ในกราฟ ผู้ใช้ต้องเลือกวิธีการนำเข้าข้อมูลดังที่ได้อธิบายไปในก่อนหน้านี้ ดังภาพที่ 6

(a)



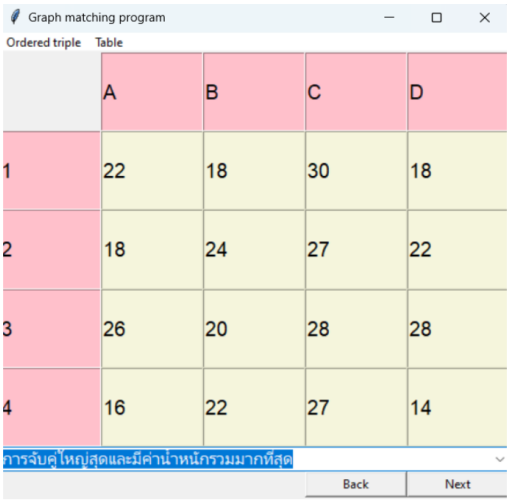
(b)

ภาพที่ 6 การรับข้อมูลเข้า (a) เมื่อต้องการรับค่าแบบสามสิ่งอันดับ และ (b) การรับค่าเข้าแบบตาราง

กรณีที่เป็นการรับค่าเข้าในรูปแบบตาราง หลังจากที่ระบุจำนวนแถวและหลักแล้ว ระบบจะทำการสร้างตารางที่มีมิติเท่ากับที่กำหนด จากนั้นผู้ใช้กรอกชื่อแถว ชื่อหลัก พร้อมค่าน้ำหนักลงในตารางที่ปรากฏขึ้นมา ดังเช่น ปัญหาการจัดพนักงานขายให้เหมาะสมกับสินค้าประจำเคาน์เตอร์เพื่อให้ได้กำไรมากที่สุด ขั้นแรกผู้ใช้ต้องทำการกรอกข้อมูลดังตารางที่ 6 ลงในระบบ โดยภาพที่ 6 แสดงการนำเข้าข้อมูลในรูปแบบของตาราง และการเลือกวิธีการจับคู่แบบจับคู่ใหญ่สุดและมีค่าน้ำหนักรวมมากที่สุด

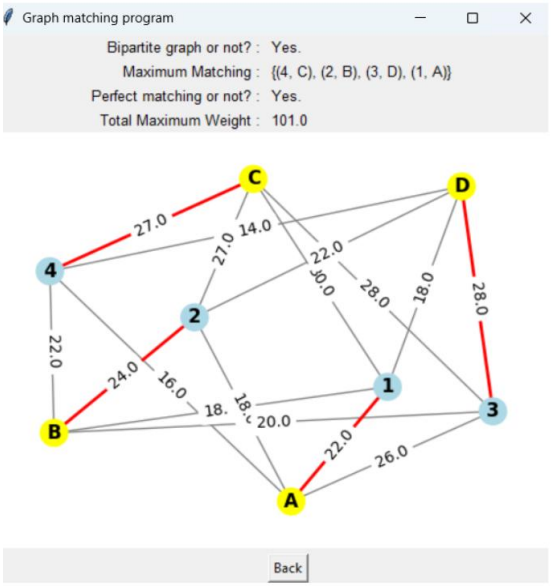
ตารางที่ 6 กำไรที่พนักงานขายสามารถทำได้ในแต่ละเคาน์เตอร์สินค้า

หมายเลขเคาน์เตอร์	พนักงานขาย			
	A	B	C	D
1	22	18	30	18
2	18	24	27	22
3	26	20	28	28
4	16	22	27	14



ภาพที่ 7 การเลือกฟังก์ชันการจับคู่ใหญ่สุดและมีค่าน้ำหนักรวมมากที่สุด

จากภาพที่ 7 เมื่อกดปุ่ม “Next” ระบบจะทำการประมวลผลตามขั้นตอนวิธีที่กำหนด และแสดงผลลัพธ์ของการจับคู่ที่เลือกดังภาพที่ 8 ซึ่งแสดงให้เห็นว่า พนักงาน A ถูกจัดให้ประจำเคาน์เตอร์ 1 พนักงาน B ถูกจัดให้ประจำเคาน์เตอร์ 2 พนักงาน C ถูกจัดให้ประจำเคาน์เตอร์ 4 และพนักงาน D ถูกจัดให้ประจำเคาน์เตอร์ 3 และมีผลรวมค่าน้ำหนักมากที่สุดคือ 101



ภาพที่ 8 ผลการจับคู่ใหญ่สุดและมีค่าน้ำหนักรวมมากที่สุดของข้อมูลทดสอบตารางที่ 6

4.2 การจับคู่ใหญ่สุดที่มีค่าน้ำหนักรวมน้อยที่สุด

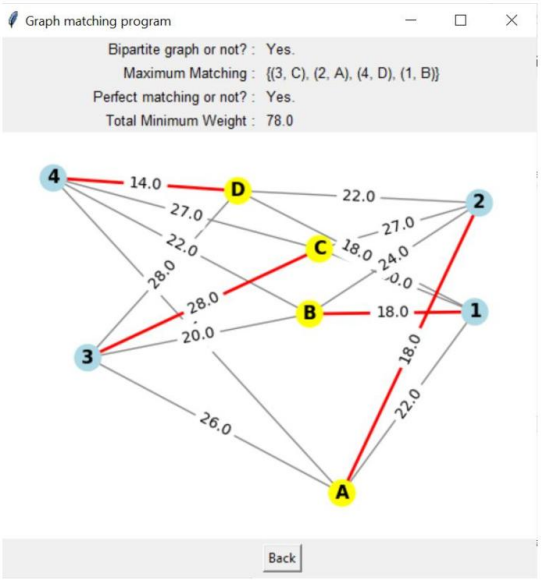
ในส่วนนี้จะเป็นการแสดงผลลัพธ์ที่ได้จากการใช้แอปพลิเคชันเพื่อทำการหาการจับคู่ในกราฟซึ่งเป็นการจับคู่ใหญ่สุดแต่มีค่าน้ำหนักรวมน้อยที่สุด โดยข้อมูลที่นำมาทำการทดลองคือ

บริษัท Anon ต้องการจัดพนักงาน A, B, C และ D ไปประจำ 4 สาขา โดยที่พนักงานแต่ละคนมีค่าใช้จ่ายเงินเดือนที่แตกต่างกันในแต่ละสาขา ดังแสดงในตารางที่ 7

ตารางที่ 7 ตารางแสดงค่าใช้จ่ายเงินเดือนของพนักงานแต่ละคนในแต่ละสาขาใน 1 ปี (หน่วย: แสนบาท)

สาขา	พนักงาน			
	A	B	C	D
1	22	18	30	18
2	18	24	27	22
3	26	20	28	28
4	16	22	27	14

ผลลัพธ์ที่ได้จากการทำงานของระบบแสดงได้ดังภาพที่ 9 ซึ่งสามารถสรุปได้ว่า พนักงาน A ถูกจัดให้ประจำเคาน์เตอร์ 2 พนักงาน B ถูกจัดให้ประจำเคาน์เตอร์ 1 พนักงาน C ถูกจัดให้ประจำเคาน์เตอร์ 3 และพนักงาน D ถูกจัดให้ประจำเคาน์เตอร์ 4 และมีผลรวมค่าน้ำหนักน้อยที่สุดคือ 78 นั่นคือบริษัทจะมีค่าใช้จ่ายเงินเดือนของพนักงานน้อยที่สุด 78 แสนบาท



ภาพที่ 9 ผลการจับคู่ใหญ่สุดและมีค่าน้ำหนักรวมน้อยที่สุดของข้อมูลทดสอบตารางที่ 7

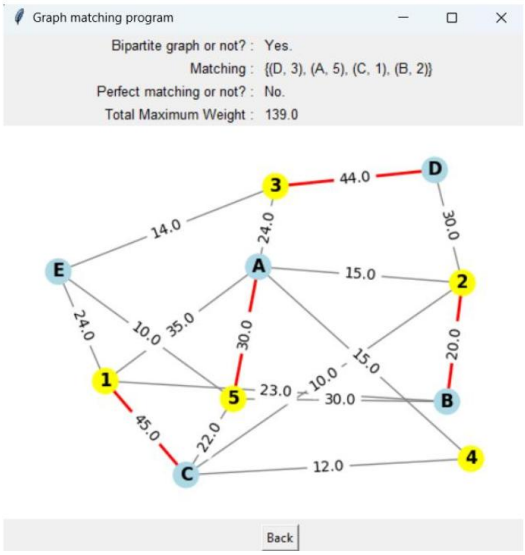
4.3 การจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด

การจับคู่ในกราฟลักษณะนี้จะเน้นที่การได้ค่าน้ำหนักรวมมากที่สุด โดยไม่คำนึงถึงขนาดของการจับคู่ในกราฟ โดยข้อมูลที่นำมาทำการทดลองเพื่อหาการจับคู่ที่มีค่าน้ำหนักรวมมากที่สุดได้แสดงไว้ในตารางที่ 8

ตารางที่ 8 ความสามารถในการทำอาหาร (กล่อง) ของเชฟต่อการทำอาหาร 1 ครั้ง

เชฟ	ชนิดของอาหาร				
	1	2	3	4	5
A	35	15	24	15	30
B	23	20	-	-	30
C	45	10	-	12	22
D	-	30	44	-	-
E	24	-	14	-	10

ข้อมูลในตารางที่ 8 แสดงความสามารถของเชฟแต่ละคนในการทำอาหารกล่องในระยะเวลาที่กำหนด ซึ่งเชฟแต่ละคนจะมีความถนัดในการทำอาหารแต่ละประเภทแตกต่างกัน ดังนั้นหากต้องการจัดสรรงานให้เชฟแต่ละคนประกอบอาหารคนละ 1 ชนิด จะสามารถแบ่งงานได้อย่างไร เพื่อให้ได้ปริมาณอาหารมากที่สุด เพื่อแก้ปัญหาดังกล่าว ข้อมูลในตารางที่ 8 จะถูกนำเข้าสู่โปรแกรมประยุกต์ และทำการเลือกฟังก์ชันการแก้ปัญหาการหาการจับคู่ที่มีค่าน้ำหนักรวมมากที่สุด โดยผลลัพธ์เป็นไปดังภาพที่ 10 ซึ่งสามารถสรุปได้ว่า เชฟ A ได้รับเลือกให้ทำอาหารประเภทที่ 5 เชฟ B ได้รับเลือกให้ทำอาหารประเภทที่ 2 เชฟ C ได้รับเลือกให้ทำอาหารประเภทที่ 1 และเชฟ D ได้รับเลือกให้ทำอาหารประเภทที่ 3 และจำนวนอาหารที่สามารถทำได้มากที่สุดคือ 139 กล่อง ในขณะที่เชฟ E ไม่ได้รับเลือกให้ทำอาหารประเภทใดเลย เนื่องจากอาหารประเภทที่ 1 3 และ 5 ถูกจัดสรรให้เชฟ C D และ A แล้ว ตามลำดับ



ภาพที่ 10 ผลการจับคู่ที่มีค่าน้ำหนักรวมมากที่สุดของข้อมูลทดสอบตารางที่ 8

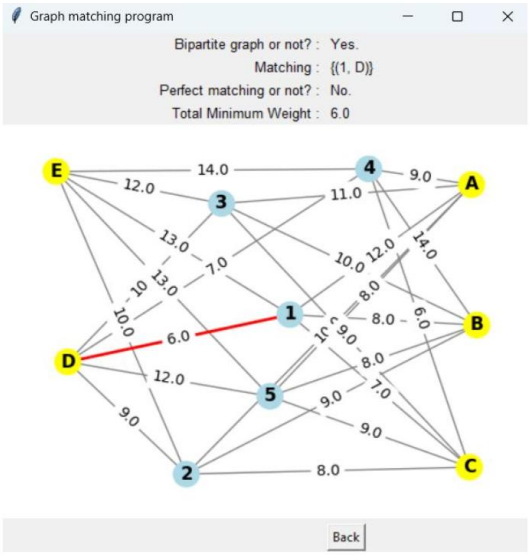
4.4 การจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุด

สำหรับข้อมูลในการทดสอบปัญหาการหาการจับคู่กราฟถ่วงน้ำหนัก ที่มีค่าน้ำหนักรวมน้อยที่สุด จะยกตัวอย่างปัญหาเกี่ยวกับการที่ผู้ประกอบการรายหนึ่งต้องการจ้างให้โรงงานแห่งที่ 1-5 ทำการผลิตสินค้าเพื่อนำออกจำหน่าย โดยตารางที่ 9 แสดงต้นทุนการผลิตต่อหน่วย (บาท) ของสินค้าแต่ละประเภท แต่เนื่องด้วยข้อจำกัดด้านเงินทุน ผู้ประกอบการจึงต้องการทราบว่าควรจ้างบริษัทใดผลิตสินค้าเพียงหนึ่งชนิดเพื่อให้ได้ปริมาณมากที่สุด โดยใช้ต้นทุนการผลิตต่ำที่สุด

ตารางที่ 9 ต้นทุนการผลิตสินค้าแต่ละประเภทต่อหน่วยของโรงงานแห่งที่ 1-5

โรงงาน	ต้นทุนการผลิตแต่ละประเภทต่อหน่วย (บาท)				
	A	B	C	D	E
1	12	8	7	6	13
2	10	9	8	9	10
3	11	10	9	10	12
4	9	14	6	7	14
5	8	8	9	12	13

ในการแก้ปัญหานี้ ผู้ใช้ได้ทำการเลือกฟังก์ชันการทำงานแบบการจับคู่กราฟถ่วงน้ำหนักที่มีค่าน้ำหนักรวมน้อยที่สุด เพราะไม่ต้องการให้ความสำคัญกับการจับคู่ใหญ่สุด ซึ่งจะช่วยคัดเลือกคำตอบที่เหมาะสมที่สุดในปัญหาการจ้างโรงงานเพื่อผลิตสินค้าเพียงหนึ่งชนิด โดยต้องการปริมาณสินค้ามากที่สุด และต้นทุนในการผลิตต่ำที่สุด โดยผลลัพธ์เป็นดังภาพที่ 11 ซึ่งแสดงให้เห็นว่ามีเพียงโรงงานที่ 1 เพียงโรงงานเดียวที่ได้รับการจัดสรรให้ผลิตงานประเภท D ซึ่งปัญหาการหาการจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุดนี้จะเน้นการบริหารงานเพื่อให้ได้ต้นทุนต่ำที่สุด ดังนั้นจึงมีองค์ประกอบบางองค์ประกอบ (ได้แก่โรงงานที่ 2-5) ไม่ได้รับการจับคู่ในการแก้ปัญหาประเภทนี้



ภาพที่ 11 ผลการจับคู่ที่มีค่าน้ำหนักรวมน้อยที่สุดของข้อมูลทดสอบตารางที่ 9

4.5 การจับคู่ใหญ่สุดโดยไม่คำนึงถึงค่าน้ำหนักรวม

เป็นการหาการจับคู่ที่ต้องการให้เซตการจับคู่มีขนาดใหญ่ที่สุดโดยไม่คำนึงถึงค่าน้ำหนักรวมว่าจะมีค่ามากที่สุดหรือน้อยที่สุด ตัวอย่างเช่นปัญหาการจัดตารางห้องประชุม หากต้องการจัดการประชุมเพื่อประชุมกลุ่มย่อยกลุ่มละ 1 ชั่วโมง ในช่วงเวลา 8.00-12.00 น. ภายใต้เงื่อนไขจำนวนห้องประชุมมีไม่จำกัด และแต่ละกลุ่มประกอบด้วยสมาชิกดังต่อไปนี้

กลุ่มที่ 1: A, B, P

กลุ่มที่ 2: A, J, H, S, T

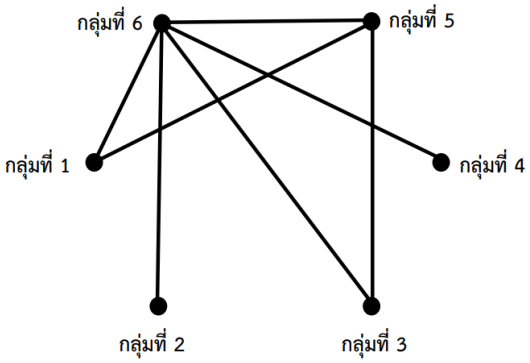
กลุ่มที่ 3: G, H, L, P

กลุ่มที่ 4: B, P, L, M, T

กลุ่มที่ 5: R, J, S, M

กลุ่มที่ 6: E, N, Z

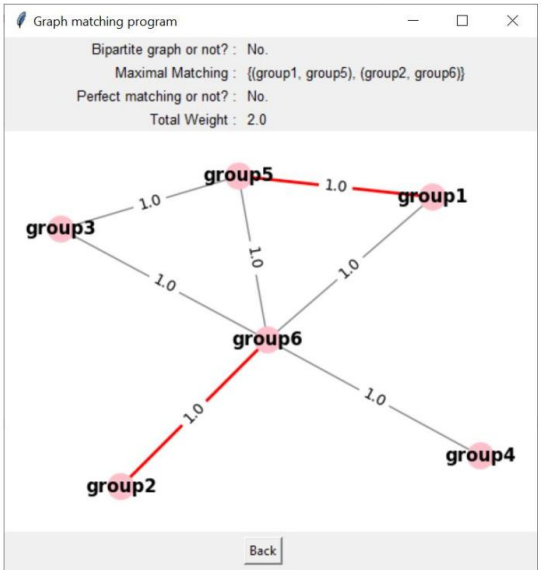
จากข้อมูลจะเห็นว่ามีสมาชิกบางคนสังกัดมากกว่า 1 กลุ่มย่อย เช่น A เป็นสมาชิกของกลุ่มที่ 1 และ 2 จึงทำให้กลุ่มที่ 1 และ 2 ไม่สามารถจัดประชุมในเวลาเดียวกันได้ มิฉะนั้นแล้ว A จะไม่สามารถเข้าประชุมได้ครบ ดังนั้นจะจัดการประชุมอย่างไรเพื่อให้สมาชิกทุกคนได้เข้าร่วมการประชุมในกลุ่มที่ตนเองสังกัดอย่างครบถ้วน เพื่อแก้ปัญหานี้ จึงจำเป็นต้องแปลงปัญหาและข้อมูลที่กำหนดให้อยู่ในรูปของกราฟ (ภาพที่ 12) เมื่อกำหนดให้จุดยอดในกราฟแทนกลุ่มการประชุม เส้นเชื่อมระหว่าง 2 จุดใด ๆ แทนการที่ไม่มีสมาชิกคนใดที่เป็นสมาชิกใน 2 กลุ่มนั้น ๆ และกำหนดค่าน้ำหนักของแต่ละเส้นเป็น 1 เนื่องจากการประชุมของทุกกลุ่มมีความสำคัญเท่ากัน



ภาพที่ 12 กราฟปัญหาการจัดตารางประชุม

เมื่อทำการนำเข้าข้อมูลกราฟในภาพที่ 12 เข้าสู่ระบบแล้ว จึงเลือกการจับคู่แบบใหญ่สุด ซึ่งผลลัพธ์ที่ได้จะพบว่า กลุ่มที่ 1 และ 5 สามารถจัดประชุมพร้อมกันได้ ในขณะที่กลุ่มที่ 2 สามารถจัดประชุมพร้อมกับกลุ่มที่ 6 และกลุ่มที่ 3 และ 4 ต้องจัดการประชุมแบบเดี่ยว (ดังแสดงในภาพที่ 13) ทำให้สามารถจัดตารางการใช้ห้องประชุมได้ดังนี้

เวลา 8.00 – 9.00 น. กลุ่ม 1 และ 5
เวลา 9.00 – 10.00 น. กลุ่ม 2 และ 6
เวลา 10.00 – 11.00 น. กลุ่ม 3
เวลา 11.00 – 12.00 น. กลุ่ม 4



ภาพที่ 13 ผลการจับคู่ใหญ่สุด

จากการดำเนินการพัฒนาระบบและตรวจสอบผลลัพธ์การทำงานพบว่าเป็นระบบที่สามารถช่วยผู้ใช้งานเพื่อการจับคู่ในกราฟถ่วงน้ำหนักสำหรับแก้ปัญหาการจัดสรรงานได้อย่างมีประสิทธิภาพ แต่ด้วยข้อจำกัดในด้านที่เป็นระบบอิสระ ยังไม่รองรับการทำงานบนเว็บไซต์ รวมถึงระยะเวลาที่ใช้ในการสร้างกราฟเมื่อมีจุดยอดเป็นจำนวนมาก และการทดลองจับคู่บนชุดข้อมูล (Dataset) แบบต่าง ๆ ทำให้การปรับปรุงประสิทธิภาพของระบบเพื่อให้ทำงานได้อย่างถูกต้องแม่นยำ ใช้ได้กับชุดข้อมูลที่มีความหลากหลาย และสามารถประมวลผลได้ในระยะเวลาอันรวดเร็วจึงเป็นเป้าหมายที่สำคัญและท้าทายต่องานวิจัยนี้ต่อไปในอนาคต

สรุปผล

งานวิจัยนี้ได้นำเสนอการพัฒนาแอปพลิเคชันสำหรับการหาการจับคู่ในกราฟถ่วงน้ำหนัก ซึ่งมีจุดแข็งที่สำคัญคือ แอปพลิเคชันนี้สามารถรองรับการรับเข้าของข้อมูลได้ 2 รูปแบบ และสามารถหาการจับคู่ได้หลากหลายรูปแบบถึง 5 ลักษณะ ที่ช่วยเพิ่มความยืดหยุ่นในการประยุกต์ใช้งานจริงมากกว่าซอฟต์แวร์หรือโค้ดที่มีอยู่ทั่วไปที่มักจำกัดเพียงกรณีใดกรณีหนึ่ง ผลลัพธ์จากการพัฒนานี้จึงไม่เพียงแต่ช่วยลดต้นทุนหรือเพิ่มผลประโยชน์ในปัญหาการจัดสรรงาน แต่ยังสามารถนำไปต่อยอดในสาขาอื่น ๆ ที่เกี่ยวข้องกับการจัดสรรทรัพยากร การขนส่ง และการวางแผนตารางงานได้อย่างกว้างขวาง โดยแสดงให้เห็นถึงการบูรณาการระหว่างทฤษฎีทางคณิตศาสตร์และเทคโนโลยีคอมพิวเตอร์เพื่อสนับสนุนการตัดสินใจเชิงกลยุทธ์อย่างมีประสิทธิภาพ

กิตติกรรมประกาศ

คณะผู้วิจัยขอขอบคุณคณะศิลปศาสตร์และวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน ที่เอื้อเฟื้อทรัพยากรและสิ่งอำนวยความสะดวกที่จำเป็นต่อการทำวิจัย จนทำให้งานวิจัยฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี

เอกสารอ้างอิง

1. Shahriar Tanvir Alam, Eshfar Sagor, Tanjeel Ahmed, Tabassum Haque, Md Shoaib Mahmud, Salman Ibrahim, Ononya Shahjahan, Mubtasim Rubaet. Assessment of Assignment Problem using Hungarian Method. Proceedings of the First Australian International Conference on Industrial Engineering and Operations Management; 20-21 December 2022; Sydney, Australia. 2328-2338. https://ieomsociety.org/proceedings/2022australia/498.pdf
2. Jayasree T G , Malavika V. Graph Theory for Optimum Assignment. International Journal of Scientific Research in Science and Technology. 2025; 12(16): 337-343. https://ijsrst.com/home/issue/view/article.php?id=IJSRST25121640
3. Fatemeh Rajabi-Alni, Alireza Bagheri. Computing a Many-to-Many Matching with Demands and Capacities between Two Sets Using the Hungarian Algorithm. Journal of Mathematics. 2023; Volume 2023, Article ID 7761902: 1-6. https://doi.org/10.1155/2023/7761902
4. M. Usha Devi, A. Marimuthu, S. Santhana Megala. Bipartite Graph Matching in Donor-Recipient Using Enhanced Hopcroft Karp Algorithm for Liver transplantation. Advances and Applications in Mathematical Sciences. 2022; 21(9): 5291-5298. https://www.mililink.com/upload/article/956498328aams_vol_219_july_2022_a41_p5291-5298_m._usha_devi_et_al.pdf
5. Gabriel Cristian Dragomir-Loga, Marius Pop. Edmonds' Blossom Algorithm Analysis in a Medical Emergency Management System. The 9th IEEE International Conference on E-Health and Bioengineering - EHB 2021 Grigore T. Popa University of Medicine and Pharmacy; 18-19 November 2021; Web Conference, Romania. 1-4. https://ieeexplore.ieee.org/document/9657586
6. นวรัตน์ อนันต์ชื่น. ทฤษฎีกราฟ 1 .พิมพ์ครั้งที่ 1. นครปฐม: ภาควิชาคณิตศาสตร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร; 2540. 5-7, 113-118. https://lib.rmutp.ac.th/bibitem?bibid=b00033744

7. Chartrand, G. and Lesniak, L. Graphs & Digraphs. 4th ed. Florida: Chapman & Hall/CRC; 2005. 28-30. https://books.google.co.th/books/about/Graphs_Digraphs_Fourth_Edition.html?id=LgZKLPRH2D4C&redir_esc=y
8. วรานุช แขมมณี. ทฤษฎีกราฟเบื้องต้น. พิมพ์ครั้งที่ 1. สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย; 2559. 2-6, 120. https://www.chulabook.com/testprep/25508?srsltid=AfmBOoq4WHZqXg8sjDXafxZywmtLfRs1QrJqnoZBuAAcf2t2OETPgZgC
9. West, D.B. Introduction to Graph Theory. 2nd ed. New Jersey: Prentice Hall; 2001. 107-111. https://books.google.co.th/books/about/Introduction_to_Graph_Theory.html?id=TuvuAAAAMAAJ&redir_esc=y
10. Bondy, AJ and Murty, USR. Graph Theory with Applications. 5th ed. U.S.A: Elsevier Science Publishing Co., Inc.; 1982. 70-90. https://www.iro.umontreal.ca/~hahn/IFT3545/GTWA.pdf
11. วิษณุ ช้างเนียม. โครงสร้างข้อมูลและอัลกอริทึม + การประยุกต์ใช้ AI (DATA STRUCTURE & ALGORITHM + AI). พิมพ์ครั้งที่ 3 นนทบุรี: บริษัท ไอดีซี พรีเมียร์ จำกัด; 2568. 305-307. https://www.chulabook.com/computer/219458?srsltid=AfmBOopXXwwLjJGsVu32jHSVbHkWBe-arLZFB_adImKtsZnlPnrcgBNc
12. Akshitha, S., Ananda Kumar, K. S., Nethritha Meda, M., Sowmva, R. and Suman Pawar, R. Implementation of Hungarian Algorithm to obtain Optimal Solution for Travelling Salesman Problem. In: 3rd IEEE. https://ieeexplore.ieee.org/document/9012439
13. Wang, Y., Wu, Y. -X. and Zhang, H. An Improved Multi-Robot Task Allocation Algorithm Based on the Hungarian Algorithm. In: *44th Chinese Control Conference (CCC)*; 28-30 July 2025;Chongqing, China. 4903-4908. https://ieeexplore.ieee.org/document/11178894
14. Othman, F., Abdullah, R. and Salam, R. A. Bipartite Graph for Protein Structure Matching, In: *Second Asia International Conference on Modelling & Simulation (AMS)*; 13-15 May 2008; Kuala Lumpur, Malaysia. IEEE publisher; 928-933. https://ieeexplore.ieee.org/document/4530600
15. Rong, H. -g., Li, Y. -j. and Zhang, X. -s. An Image Matching Algorithm Based on Bipartite Graph, In: *7th* International Conference on Information Science, Computer Technology and Transportation; 27-29 May 2022; Xishuangbanna, China. 1-4. https://ieeexplore.ieee.org/abstract/document/10071784
16. A. B. Chaudhuri. Flowchart and Algorithm Basics The Art of Programming. 2nd ed. Virginia: Mercury Learning and Information.; 2020. 1-17. https://books.google.co.th/books/about/Flowchart_and_Algorithm_Basics.html?id=pUOQzQEACAAJ&redir_esc=y
17. ศิริพร อ่วมมีเพียร. วิชาการเขียนโปรแกรมคอมพิวเตอร์เบื้องต้นรหัสวิชา 20901-1002. พิมพ์ครั้งที่ 1. กรุงเทพฯ : วังอักษร, 2563. 21-38. https://search-library.spu.ac.th/bib/189590
18. Shopov, V. and Markova, V. Application of Hungarian Algorithm for Assignment Problem. In: 2021 International Conference on Information Technologies (InfoTech); 16-17 Sep 2021; Varna, Bulgaria. Piscataway: Institute of Electrical and Electronics Engineers, Inc.; 2021. 1–4. https://ieeexplore.ieee.org/abstract/document/9548600

19. Deo, N. Graph Theory with Applications to Engineering and Computer Science. 2nd ed. New York: Dover Publications, Inc.; 2016. 496. https://books.google.co.th/books/about/Graph_Theory_with_Applications_to_Engine.html?id=uk1KDAAAQBAJ&redir_esc=y
20. Denise, G. and Matthias, B. The Practitioner's Guide to Graph Data. 1sted. California: O'Reilly Media.; 2020. 420. https://books.google.co.th/books/about/The_Practitioner_ s_Guide_to_Graph_Data.html?id=n6nYDwAAQBAJ&redir_esc=y