แบบจำลองปัญหาเส้นทางพาหนะที่มีความจุคันรถในการรับและส่งพัสดุในกรณีมีการตีกลับพัสดุ

ภัทรพล จำปาหอม*, วชิรวิชญ์ ชาภูมี, ศิวนาท ทนงจิตร, และมนสิการ จันทร์สร้าง
โรงเรียนมหิดลวิทยานุสรณ์ จังหวัดนครปฐม ประเทศไทย
*pattarapon.jam_g33@mwit.ac.th

บทคัดย่อ

หลังการแพร่ระบาดของ COVID-19 การซื้อขายออนไลน์เติบโตอย่างรวดเร็ว ธุรกิจขนส่งให้ความสำคัญกับการวางแผนเส้นทางซึ่งเกี่ยวข้องกับปัญหาเส้นทางยานพาหนะที่มีความจุจำกัดในการรับและส่งพัสดุ (CVRPPD) เพื่อใช้ในการลดต้นทุน อย่างไรก็ตาม งานวิจัยที่ผ่านมายังไม่มีการศึกษาการวางแผนเส้นทางในกรณีที่เกิดการตีกลับของพัสดุด้วยเหตุผล เช่น ที่อยู่ไม่ถูกต้อง, ไม่มีผู้รับปลายทาง หรือผู้รับปฏิเสธการรับพัสดุ เป็นต้น งานวิจัยนี้จึงพัฒนาแบบจำลองทางคณิตศาสตร์สำหรับการวางแผนเส้นทางการขนส่งที่มีข้อจำกัดด้านความจุ โดยรองรับทั้งการรับและส่งในกรณีที่พัสดุมีโอกาสในการตีกลับ ซึ่งคณะผู้จัดทำศึกษาแผนที่ของเมืองขนาดต่าง ๆ ในประเทศสหรัฐอเมริกา พบว่าความน่าจะเป็นของการตีกลับพัสดุมีความสัมพันธ์กับโอกาสที่จะเกิดการล้มเหลวในรูปแบบเส้นโค้งซิกมอยด์ (sigmoid curve) โดยโอกาสที่จะเกิดการล้มเหลวที่คำนวณได้มีผลลัพธ์ที่ใกล้เคียงกันในทุกวิธีการสร้างเส้นทาง และในแง่ของระยะทางและเวลาในการประมวลผล วิธี Saving Algorithm มีผลลัพธ์ใกล้เคียงวิธี Nearest Neighbor Heuristic แต่น้อยกว่าวิธี Simulated Annealing เป็นอย่างมาก และในแง่ของโอกาสที่จะเกิดการล้มเหลว วิธี Saving Algorithm ให้ผลลัพธ์ที่ต่ำกว่าวิธีการอื่นเล็กน้อย แบบจำลองทางคณิตศาสตร์นี้สามารถนำไปประยุกต์ใช้จริงในการวางแผนเส้นทาง ช่วยเพิ่มประสิทธิภาพและลดต้นทุนการขนส่ง อีกทั้งยังสร้างความมั่นใจให้ผู้ประกอบการในการขนส่งพัสดุในกรณีเกิดการตีกลับ

คำสำคัญ: ปัญหาเส้นทางยานพาหนะที่มีความจุจำกัดในการรับและส่งพัสดุ (CVRPPD), การจำลองการอบเหนียว (Simulated annealing), อัลกอริทึมประหยัด (Saving Algorithm), วิธีเพื่อนบ้านที่ใกล้เคียงที่สุด (Nearest Neighbor Heuristic), แบบจำลองทางคณิตศาสตร์ (Mathematical model); การตีกลับพัสดุ (Returned package)

1.บทนำ

ปัญหาเส้นทางการเดินรถ (Vehicle Routing Problem: VRP) เป็นประเด็นที่มีการศึกษามาหลายสิบปี (Dantzig & Ramser, 1959) โดยว่าด้วยการหาเส้นทางที่เหมาะสมที่สุดสำหรับวางแผนการเดินรถ อย่างไรก็ตาม ความแตกต่างในนิยาม

ของ “เส้นทางที่ดีที่สุด” รวมถึงปัจจัยเสริมต่าง ๆ ที่เพิ่มขึ้น ส่งผลให้แนวทางในการค้นหาเส้นทางที่เหมาะสมมีหลากหลายวิธี อีกทั้งในโลกแห่งความจริง การขนส่งมักประสบปัญหาหลากหลาย เช่น สภาพการจราจร อุบัติเหตุ และถนนชำรุด เป็นต้น

คณะผู้จัดทำให้ความสนใจในปัญหาการตีกลับพัสดุ หมายถึงการจัดส่งที่ล้มเหลว (failed delivery) ซึ่งพัสดุจะยังคงอยู่ในรถและต้องถูกนำกลับสู่ศูนย์กระจายพัสดุ หรือเข้าสู่กระบวนการคืนพัสดุในภายหลัง ปัญหานี้เกิดจากการเขียนที่อยู่ผิด, ไม่มีผู้รับปลายทาง หรือผู้รับปฏิเสธการรับจากสาเหตุต่างๆ ส่งผลให้พัสดุตกค้างในรถซึ่งมีความจุจำกัด และทำให้

แผนการเดินทางสำหรับปัญหาเส้นทางพาหนะที่มีความจุจำกัดในการรับ-ส่งพัสดุ (Capacitated Vehicle Route Problem for Pickup and Delivery, CVRPPD) เกิดความผิดพลาดระหว่างขนส่งได้ งานวิจัยนี้มุ่งศึกษาระยะทางของเส้นทาง และทดสอบความเป็นไปได้ของการเปลี่ยนเส้นทางจากแผนเดิมก่อนเริ่มเดินรถ เนื่องจากการตีกลับของพัสดุที่ส่งผลให้ไม่สามารถรับพัสดุเพิ่มได้ในบางจุด

นอกจากนี้ การตีกลับพัสดุจากการสั่งซื้อสินค้าทางออนไลน์เป็นปัญหาที่พบได้บ่อยในหลายประเทศ โดยจากรายงานโดย Loqate (2021) พบว่าในสหรัฐอเมริกา สหราชอาณาจักร และเยอรมนีในปี 2020 มีการจัดส่งล้มเหลวจากการซื้อออนไลน์คิดเป็น 8%, 6% และ 7% และคำนวณต้นทุนรวมต่อปีจากการตีกลับพัสดุในทั้ง 3 ประเทศคิดเป็นประมาณ 16 ล้านบาท[1] ทำให้เห็นว่าการตีกลับพัสดุส่งผลต่อทั้งต้นทุนและการจัดการเส้นทางขนส่ง ซึ่งเป็นความท้าทายสำคัญของธุรกิจขนส่งทั่วโลกในปัจจุบัน

โดยในการสร้างเส้นทางเพื่อแก้ปัญหาการเดินรถในปัจจุบันมีหลายวิธีถูกนำมาใช้ โดยในที่นี้จะกล่าวถึง 3 วิธีที่เป็นที่นิยมได้แก่ Nearest Neighbor Heuristic (NNH), Savings Algorithm และ Simulated Annealing (SA) โดยแต่ละวิธีมีกระบวนการและผลลัพธ์โดยคร่าว ดังนี้

NNH เป็นวิธีการที่เรียบง่าย โดยทำการเลือกสร้างเส้นทางไปยังจุดถัดไปที่ใกล้ที่สุดจากตำแหน่งปัจจุบัน หลังจากนั้นจึงทำซ้ำไปจนกว่าจะครบทุกจุด ข้อได้เปรียบของวิธีนี้คือความซับซ้อนที่น้อยและประมวลผลได้อย่างรวดเร็ว แต่อย่างไรก็ตาม จากการวิจัยของ Harahap (2023) พบว่า NNH มักจะให้ผลลัพธ์ด้านระยะทางที่ยาวกว่าวิธีอื่น ๆ

Savings Algorithm เป็นกระบวนการที่ใช้ในการสร้างเส้นทางการเดินรถโดยการจับคู่เพื่อสร้างเส้นทางใหม่จากการเปรียบเทียบระยะทางกับเส้นทางเดิม หากเส้นทางที่เกิดจากการจับคู่ทำให้ระยะทางรวมสั้นลงก็จะเลือกใช้เส้นทางที่เกิดขึ้นใหม่นั้น โดยผลลัพธ์ของวิธีนี้ของเส้นทางที่มีระยะทางที่สั้นโดยที่สามารถลดเวลาประมวลผลไปพร้อมกันได้ (Paessens, 1988)

Simulated Annealing จะทำการปรับปรุงเส้นทางที่มีอยู่แล้วผ่านขั้นตอนการสุ่มโดยพิจารณาถึงค่าอุณหภูมิ (Temperature) โดยในช่วงแรกของการสุ่มที่อุณหภูมิยังคงสูง แม้เส้นทางใหม่จะทำให้ระยะทางรวมเพิ่มขึ้นเล็กน้อยวิธีการนี้ก็ยังคงมีโอกาสเลือกนำเส้นทางนั้นไปสุ่มต่อ แต่เมื่ออุณหภูมิลดลงหลังการสุ่มหลายครั้ง วิธีการจะเลือกรับเฉพาะเส้นทางที่ระยะทางรวมสั้นลงเท่านั้น Wei (2018) พบว่า Simulated Annealing ให้ผลลัพธ์ที่ดีกว่า Tabu Search และ Genetic Algorithm เมื่อใช้ในการแก้ปัญหาเส้นทางพาหนะที่มีความจุจำกัด (Capacitated Vehicle Routing Problem หรือ CVRP)

ทั้งสามวิธีดังกล่าวมีลักษณะเด่นและข้อจำกัดที่แตกต่างกัน การเปรียบเทียบทั้งสามวิธีนี้จะช่วยให้เห็นว่าการเลือกใช้วิธีใดในแต่ละสถานการณ์สามารถช่วยปรับปรุงเส้นทางการเดินรถได้ดีที่สุด

[1] Loqate (2021) ระบุว่าค่าใช้จ่ายนี้ เกิดขึ้นในสหรัฐฯ $193,730, สหราชอาณาจักร £68,084 และเยอรมนี €144,354 เมื่อแปลงตามอัตราแลกเปลี่ยนปี 2020 รวมคิดเป็นราว 16 ล้านบาทต่อปี

2. วิธีทำการทดลอง

1 นิยาม CVRPPDRP Model

การที่จะแก้หรือทำความเข้าใจในปัญหาที่กล่าวมาจำเป็นต้องมีความเข้าใจในลักษณะเฉพาะของปัญหา ซึ่งคณะผู้จัดทำได้ศึกษาแบบจำลองทางคณิตศาสตร์ของ CVRPPD หรือปัญหาเส้นทางการขนส่งของรถที่มีความจุพัสดุจำกัด ซึ่งมีภารกิจคือ ส่งพัสดุ และรับพัสดุ ของ Kui-Ting Chen และคณะ (2015) และนำมาปรับปรุงและเพิ่มตัวแปรและเงื่อนไขสำหรับการตีกลับพัสดุ จึงได้แบบจำลองทางคณิตศาสตร์ที่ตรงตามวัตถุประสงค์ของงานวิจัย นั่นคือ CVRPPDRP (Capacitated Vehicle Routing Problem with Pickup and Delivery for Returned Packages) ดังนี้

กราฟ (Graph)

นิยามกราฟ $G = (V, E)$ โดยที่

- $V = \{0,1,2,\dots,n\}$: เซตของทุกจุด (Vertex) ซึ่งคือจุดตัดถนน เป็นตัวแทนของบ้านลูกค้าบนเส้นถนนนั้น ๆ เมื่อ **0** คือจุดกระจายพัสดุ
- $E = \left\{ \{i,j\} \middle| \begin{matrix} i,j \in V, i \sim j \\ \text{ถ้าระหว่าง } i \text{ กับ } j \text{ มีถนนเชื่อมกัน} \end{matrix} \right\}$: เซตของเส้น (edge) ซึ่งคือถนน

เซต (Set)

- $P \subseteq V - \{0\}$: เซตของจุดรับพัสดุ (Pickup node)
- $D \subseteq V - \{0\}$: เซตของจุดส่งพัสดุ (Delivery node)
- $P \cap D = \emptyset$: ไม่มีสมาชิกซ้ำในเซตของจุดรับและจุดส่งพัสดุ
- $SB \subseteq D$: เซตของจุดส่งของที่มีโอกาสที่จะถูกตีกลับพัสดุ (จุดตีกลับ)

พารามิเตอร์ (Parameters)

- c_{ij} : ระยะทางของถนนจากจุด i ไปถึงจุด j
- q_i : จำนวนพัสดุในรถขนส่งที่เปลี่ยนแปลงที่จุด i (เมื่อส่งพัสดุจะมีค่าเป็นลบ และรับพัสดุจะมีค่าเป็นบวก)
- Q : จำนวนพัสดุที่รถขนส่งบรรจุได้สูงสุด
- p : ความน่าจะเป็นของการเกิดการตีกลับพัสดุ

ตัวแปรตัดสินใจ (Decision Variables)

- x_{ij} : ตัวแปรไบนารี่ (Binary variable) มีค่าเท่ากับ 1 เมื่อรถอยู่ในเส้นทางจากจุด i ไปถึงจุด j
- u_i : จำนวนพัสดุในรถขนส่งหลังผ่านจุด i
- f_i : ตัวแปรไบนารี่ (Binary variable) จะมีค่าเท่ากับ 1 เมื่อเกิดการล้มเหลวเนื่องจากรถไม่สามารถรับพัสดุเพราะความจุเต็มคันรถ
- r_i : ตัวแปรไบนารี่ (Binary variable) จะมีค่าเท่ากับ 1 เมื่อจุด i คือจุดตีกลับ ($i \in SB$)

ฟังก์ชันวัตถุประสงค์ (Objective Function)

$$Minimize \quad \sum_{i \in V} \sum_{j \in V} (c_{ij} \cdot x_{ij}) \tag{1}$$

$$Minimize \sum_{i \in SB} (p \cdot f_i) \tag{2}$$

สมการ (1) คือฟังก์ชันจุดประสงค์ที่ต้องการค่าน้อยที่สุดของระยะทางรวม สมการ (2) ฟังก์ชันวัตถุประสงค์ที่กล่าวถึงโอกาสที่จะล้มเหลวในการขนส่งเมื่อมีโอกาสในการตีกลับเป็น p

เงื่อนไข (Constraints)

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in P \tag{3}$$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in D \tag{4}$$

$$\sum_{j \in V} x_{0j} = 1 \tag{5}$$

$$\sum_{i \in V} x_{i0} = 1 \quad (6)$$

$$\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} \quad \forall i \in V \quad (7)$$

$$0 \leq u_i \leq Q \quad \forall i \in V \quad (8)$$

$$u_i + q_j \leq u_j + Q \cdot (1 - x_{ij}) \quad \forall i, j \in V \quad (9)$$

$$u_j \geq u_i + q_j(x_{ij} - r_j) - Q \cdot (1 - x_{ij}) \quad \forall i, j \in V \quad (10)$$

$$p = P(i \in SB), \quad \forall i \in D \quad (11)$$

$$Q \geq u_i \cdot (1 - f_i), \quad \forall i \in P \quad (12)$$

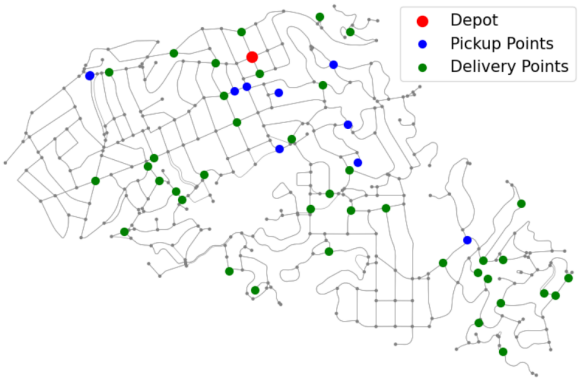
สมการ (3), (4) รถสามารถไปที่จุดรับของและส่งของได้ 1 ครั้ง สมการ (5), (6) รถจะต้องเริ่มที่จุดกระจายพัสดุและสิ้นสุดการเดินทางที่จุดกระจายพัสดุ สมการ (7) ใช้ครอบคลุมความต่อเนื่องของเส้นทาง (ถ้ารถมาถึงจุดส่งของแล้วต้องออกจากจุดนั้น ห้ามอยู่จุดเดิม) สมการ (8) จำนวนพัสดุภายในรถต้องมีค่าไม่เกินค่าบรรจุสูงสุดของรถ และมีค่าต่ำสุดเท่ากับ 0 สมการ (9) เพื่อหลีกเลี่ยงเส้นทางย่อย (subtours) ซึ่งเป็นเส้นทางที่ไม่มีลูกค้าอยู่ในเส้นทาง หรือเส้นทางที่ไม่เชื่อมโยงกับจุดกระจายพัสดุซึ่งเป็นจุดเริ่มต้นหรือจุดสิ้นสุดของเส้นทางด้วยการเดินทางไม่สามารถวนกลับ ไปที่จุดที่เคยไปมาแล้ว (Miller-Tucker-Zemlin constraints) (Desrochers, M. & Laporte, G., 1991) สมการ (10) ทำหน้าที่ควบคุมจำนวนพัสดุในรถทั้งระหว่างการสร้างเส้นทาง และตรวจสอบความถูกต้องภายหลังจากที่มีการจำลองจุดที่เกิดการตีกลับ สมการ (11) แสดงความน่าจะเป็น p คือความน่าจะเป็นในการตีกลับพัสดุของจุดในเซตจุดส่งพัสดุ (D) สมการ (12) ปริมาณบรรทุกที่จุด i ที่เป็นสมาชิกของจุดรับพัสดุ (P) ต้องไม่เกินความจุ Q ถ้าหากเกินจะทำให้มีการระบุการล้มเหลว ($f_i = 1$)

หลังจากการสร้างเส้นทางเสร็จสิ้นแล้ว จะประยุกต์ใช้สมการ สมการ (2), (11) และ (12) เพื่อหาโอกาสการล้มเหลว เนื่องจากงานวิจัยนี้สนใจในการศึกษาเส้นทางที่สั้นที่สุด และมีผลพลอยได้คือมีโอกาสที่จะล้มเหลวในการขนส่งน้อยที่สุด

2 กำหนดข้อมูลเมืองและจุดที่ใช้ศึกษา

จากข้อมูลของ National Household Travel Survey, USA แสดงให้เห็นว่า โดยปกติแต่ละครัวเรือนจะสั่งพัสดุออนไลน์สัปดาห์ละ 1 ครั้ง (Cokyasar, T., 2022) ในงานนี้จะให้จุดตัดถนนเป็นตัวแทนของบ้านลูกค้าบนเส้นถนนนั้น ๆ ดังนั้นการพิจารณาการขนส่งใน 1 วัน เป็นการสุ่มจุดตัดถนนที่จะมีการส่งหรือรับพัสดุนั้นมีเพียงแค่ 1/7 ของจุดทั้งหมด จากนั้นใช้ OpenStreetMap ในการนำเข้าข้อมูลเมืองต่าง ๆ มาเปลี่ยนเป็น Edge-weighted graph โดยกำหนดให้จุด (node) คือ จุดตัดถนนและ เส้น (edge) คือ เส้นถนนเชื่อมระหว่างจุดตัดถนน และกำหนดให้น้ำหนักของเส้น (weighted edge) คือ ระยะทางบนถนนที่ใช้เดินทางระหว่างจุดตัดถนน

ต่อมาสุ่มจุดกระจายพัสดุ (depot) มา 1 จุด จากนั้นสุ่มเลือกจุดที่เหลือด้วยจำนวนจุดทั้งหมดหารด้วย 7 ได้เป็นจุดลูกค้า และให้ 1/5 ของจำนวนจุดลูกค้าทั้งหมด เป็นจุดที่ต้องไปรับพัสดุ (P) และจุดที่เหลือเป็นจุดที่ต้องไปส่งพัสดุ (D) ได้แผนที่ตัวอย่างดังภาพที่ 1



ภาพที่ 1 แผนที่ตัวอย่างของเมือง Piedmont, California, USA สร้างด้วย OSMnx

จากนั้นกำหนดความต้องการของจุด นั่นคือจำนวนพัสดุในรถขนส่งที่เปลี่ยนแปลงที่จุด i (q_i) โดยจุดส่งของ (delivery node) มีค่าเป็น -1 และจุดรับของเป็นค่าสุ่มตั้งแต่ 1-4 และกำหนดความจุคันรถ (Q) เป็นค่าที่มากที่สุดระหว่างผลรวมของพัสดุที่ถูกส่งเพื่อให้พอดีกับพัสดุที่มีในรถและพร้อมถูกส่ง และผลรวมของพัสดุที่ถูกรับ เพื่อให้พอดีกับความต้องการรับพัสดุบนรถ เนื่องจากในทุกเช้าของวันในการขนส่งจะมีข้อมูลของจำนวนพัสดุที่ต้องไปส่งและรับ ซึ่งการเลือกรถที่มีความจุเหมาะสมกับจำนวนพัสดุจะทำให้การขนส่งมีประสิทธิภาพมากขึ้น นอกจากนี้ จะกำหนดโอกาสเกิดการตีกลับของพัสดุ เพื่อให้สอดคล้องกับในความเป็นจริงที่อาจเกิดการส่งพัสดุผิดที่ หรือปัญหาอื่น ๆ ซึ่งส่งผลให้เกิดการตีกลับของพัสดุขึ้นได้

3 หาเส้นทางจากวิธีการต่าง ๆ

เส้นทางขนส่งจะมีจุดกระจายพัสดุ (depot) เป็นจุดเริ่มต้นและจุดสิ้นสุดการขนส่ง คณะผู้จัดทำได้เลือกวิธีการหาเส้นทาง ดังนี้

3.1 Nearest Neighbor Heuristic (NNH)

การสร้างเส้นทางด้วย Nearest Neighbor Heuristic (NNH) จะได้เส้นทางที่เริ่มต้นที่จุดกระจายพัสดุ ไปยังจุดที่ใกล้ที่สุดจากจุดเริ่มต้น ทำซ้ำจนกว่าจะสิ้นสุดที่จุดสุดท้ายของเซตจุดทั้งหมด (V) และเพื่อให้ได้เส้นทางปิด (Closed path) ตามเงื่อนไขของแบบจำลองทางคณิตศาสตร์ ด้วยการตั้งค่าให้ NNH หาเส้นทางโดยให้จุดสุดท้ายกลับไปยังจุดกระจายพัสดุ

3.2 Saving Algorithm

จาก Clarke, G. and Wright, J.R. (1964) และน้ำฝน พาพันธ์ และ ภัทรานิษฐ์ แก้วประดิษฐ (2019) ได้อธิบาย Saving algorithm ตามขั้นตอนดังนี้

1. **คำนวณค่าประหยัด** เริ่มต้นจากการสร้างเมทริกซ์ระยะทาง (c_{ij}) ระหว่างทุกจุดโดยหาจาก nx.shortest_path_length ซึ่งเป็นฟังก์ชันใน NetworkX ที่ใช้ Dijkstra algorithm ในการหาระยะทางแล้วทำการสร้างเส้นทางจากจุดกระจายพัสดุไปยังจุดใด ๆ ทุกจุดในกราฟหลังจากนั้นลากเส้นเชื่อมกลับจากจุดนั้น ๆ กลับมายังจุดกระจายพัสดุ แล้วคำนวนค่าประหยัดของลูกค้า i และ j คือต้นทุนที่สามารถประหยัดได้จากการเชื่อมโยง (i, j) ทำให้เกิดเส้นทาง (0, i, j,0) ดังภาพที่ 2 (ข) แทนการเสียต้นทุนในการเดินทางสองเส้นทางคือ (0, i,0) และ (0, j,0) ดังภาพที่ 2 (ก) จากนั้นคำนวณค่าประหยัดด้วยสมการ (13)

$$S_{ij} = d_{0i} + d_{0j} - d_{ij} \quad (13)$$

โดย 0 แทนจุดกระจายพัสดุ, S_{ij} คือค่าประหยัดระหว่างลูกค้า i และ j, d_{ij} คือระยะทางระหว่างลูกค้า i และ j

2. **เรียงลำดับค่า S_{ij}** จากมากไปน้อย
3. **แทรกลูกค้ารายใหม่เข้าเส้นทางเดิม** หากค่าระยะประหยัด (Saving) มีเครื่องหมายเป็นบวก นั่นคือระยะทางรวมสั้นกว่าระยะทางใหม่ ให้ทำการรวม 2 เส้นทางนั้นเป็นเส้นทางใหม่ เราจะทำการรวมลูกค้า i และ ลูกค้า j ให้อยู่ในเส้นทางเดียวกันคือเราจะได้เส้นทางในการขนส่งพัสดุ (0, i, j,0) ดังภาพที่ 2 (ข)
4. **ทำการแทรกลูกค้า**เข้าไปเรื่อย ๆ และทำซ้ำจนกว่าจะเหลือเส้นทางเพียงเส้นทางเดียว

(ก) (ข)

ภาพที่ 2 (ก) ส่งพัสดุไปกลับ ทุกจุด (ข) รวมจุดส่งพัสดุเข้าด้วยกัน

3.3 Simulated Annealing (SA)

Simulated Annealing (SA) ถูกนำมาใช้ในการหาทางเดินที่ดีที่สุดจากจุดเริ่มต้น (depot) ไปยังจุดรับและส่งพัสดุ (pickup and delivery nodes) เริ่มต้นจากการสร้างเส้นทางเริ่มต้นแบบสุ่ม แล้วทำการปรับเส้นทางโดยการสลับตำแหน่งของสองจุด (Two-opt swap) และคำนวณระยะทางใหม่ หากเส้นทางใหม่ดีขึ้น (ระยะทางสั้นกว่า) ก็จะยอมรับเส้นทางนั้น แต่ถ้าระยะทางแย่ลง ก็จะยอมรับด้วยความน่าจะเป็นที่ลดลงตามการลดอุณหภูมิ (cooling rate) การลดอุณหภูมิจะช่วยให้โอกาสในการยอมรับเส้นทางที่แย่ลงนั้นน้อยลง ซึ่งช่วยหลีกเลี่ยงการติดอยู่ที่ค่าที่ไม่ดีและหาทางที่ดีที่สุดจนเจอได้ โดยในการทดลองนี้ใช้ค่าอุณหภูมิที่ 10,000, อัตราการเย็นตัวเป็น 0.995 และอุณหภูมิต่ำสุดเป็น 10^{-3}

4 การวิเคราะห์เส้นทาง

Algorithm 1: Failure Simulation

1. **Define FUNCTION** ‘simulate_failures(route, q_i, return_prob, vehicle_capacity, seed_value)’:
    2. **Set** random seed to seed_value
    3. Initialize load = number of delivery nodes
    4. Initialize failures = 0
    5. **For** each node **in** route:
        6. **IF** node is in pickup_nodes **THEN**:
            7. Retrieve demand from q_i[node]
            8. Increment load by demand
        9. **ELSE IF** node is in delivery_nodes **THEN**:
            10. Retrieve delivery_amount from q_i[node]
            11. Increment load by delivery_amount
            12. **IF** random value < return_prob **THEN**:
                13. Increment load by 1 (simulate product return)
        14. **IF** load > vehicle_capacity OR load < 0 **THEN**:
            15. Increment failures by 1
            16. **BREAK** the loop
    17. **RETURN** failures
18. Initialize total_failures = 0
19. Initialize total_tests = 0
20. **For** seed_value = 1 **to** 10001:
    21. Initialize failures_per_seed = 0
        22. **Call** ‘simulate_failures(route_full, q_i, return_prob, vehicle_capacity, seed_value)’
        23. Increment failures_per_seed by returned failures
    24. Increment total_failures by failures_per_seed
    25. Increment total_tests by num_tests_per_seed
26. **Calculate** per_failure_rate = (total_failures / total_tests) * 100
27. **Output** per_failure_rate

คำอธิบาย

route: ลำดับของจุดตามเส้นทาง, q_i: ความต้องการของจุด, return_prob: ความน่าจะเป็นของการตีกลับพัสดุ, vehicle_capacity: ความจุสูงสุดของรถ (Q), seed_value: รหัสการสุ่มสำหรับการใช้ข้อมูลของการสุ่มซ้ำ, total_failures: จำนวนครั้งที่ล้มเหลวของทุกรหัสการสุ่ม, total_tests: จำนวนครั้งที่ทดสอบ, per_failure_rate: เปอร์เซ็นต์การล้มเหลวในการขนส่ง, pickup_nodes: จุดรับพัสดุ (Pickup node), delivery_nodes: จุดส่งพัสดุ (Delivery node), route_full: เส้นทางแบบเต็มที่ใช้ในการทดสอบ

นำข้อมูลที่ได้จากแต่ละเส้นทาง มาวิเคราะห์โดยใช้การทดสอบแต่ละเส้นทางในกรณีที่มีการตีกลับพัสดุ ณ จุดต่าง ๆ โดยกำหนดโอกาสในการตีกลับพัสดุเป็น 0.3 และทุกจุดส่งพัสดุมีโอกาสตีกลับพัสดุเท่ากัน (ค่า 0.3 นี้เป็นเพียงค่าที่คณะผู้จัดทำเลือกมาใช้ สำหรับการนำไปใช้ต่อ ควรใช้ค่าที่มีความเป็นจริง) เมื่อเกิดกรณีที่ไม่สามารถส่งพัสดุได้ตามแผนเส้นทางเดิมเนื่องจากเกิดการตีกลับจนไม่สามารถรับพัสดุเพิ่มได้ จะถือว่าล้มเหลวทันที เมื่อทดสอบเสร็จสิ้นจะทำการบันทึกค่าต่าง ๆ ที่ได้จากการทดสอบ คือ ระยะทาง และโอกาสที่จะล้มเหลวในการขนส่ง (%) ของทั้งสองเส้นทาง แล้วนำมาเปรียบเทียบกัน ตามอัลกอริทึม 1

อัลกอริทึมนี้สร้างขึ้นจากสมการ (11) และ (12) ให้เป็นฟังก์ชัน simulated_failures โดยเริ่มจากการ

กำหนดความจุเริ่มต้นของรถ คือ ผลรวมของพัสดุที่ต้องส่ง และกำหนดตัวแปร failures เป็น 0 จากนั้นในทุก ๆ จุดตามลำดับของ route หากจุดนั้น ๆ เป็นจุดรับพัสดุ (Pickup node) จะทำการเพิ่มพัสดุในรถ (load) ตามความต้องการในการส่งของจุดนั้น ๆ (q_i) แต่ถ้าหากจุดนั้นเป็นจุดส่งพัสดุ (Delivery node) จะทำการสุ่มค่าจำนวนจริงตั้งแต่ 0 ถึง 1 โดยถ้าค่าที่สุ่มมามีค่าน้อยกว่าโอกาสที่จะเกิดการตีกลับ จะถือว่าเกิดการตีกลับขึ้นซึ่งจะทำให้จำนวนพัสดุในรถไม่เปลี่ยนแปลง แต่ถ้าไม่เกิดการตีกลับขึ้นจะลดจำนวนพัสดุในรถลง 1 ชิ้น หลังจากนั้นจึงทำการตรวจสอบจำนวนพัสดุในรถหากจำนวนพัสดุไม่เกินความจุของรถ ทำการวนซ้ำกระบวนการกับจุดต่อไปบนลำดับ (route) จนครบทุกจุดแล้วส่งออกค่า failures คือ 0 และหากจำนวนพัสดุเกินความจุของรถ หรือมีค่าต่ำกว่า 0 การวนซ้ำจะสิ้นสุดทันทีและส่งออกค่า failures คือ 1 จากนั้นทดสอบเส้นทาง 10,000 ครั้ง ด้วย seed ที่แตกต่างกัน โดยในทุก seed เกิดการทดสอบ 1 ครั้งและค่า failures จะถูกรวมกันและหารด้วยจำนวนครั้งในการทดสอบ จนได้ผลออกมาเป็นโอกาสที่จะล้มเหลวเนื่องจากการตีกลับของพัสดุ (%)

3. ผลและอภิปรายผลการทดลอง

1 ข้อมูลเมืองตัวอย่าง

ใช้ไลบารี OSMnx ในการนำเข้าข้อมูลเมืองจาก OpenStreetMap เพื่อการศึกษาแบบจำลองในการหาเส้นทางที่เหมาะสมที่สุดในการขนส่งโดยพิจารณากรณีที่เกิดการตีกลับ โดยมีตัวแปรเป็นขนาดของเมืองตัวอย่างเพื่อนำมาเปรียบเทียบถึงประสิทธิภาพของแบบจำลองที่อาจแตกต่างกัน เมื่อเปลี่ยนเมืองตัวอย่างโดยอิงขนาดเมืองจากจำนวนจุดตัดของถนนบนเมือง (node)

จากตารางที่ 1 ศึกษาเมืองในสหรัฐอเมริกาซึ่งมีขนาดเมืองตั้งแต่ 6.80 ถึง 29.64 ตารางกิโลเมตร รวมทั้งสิ้น 23 เมือง

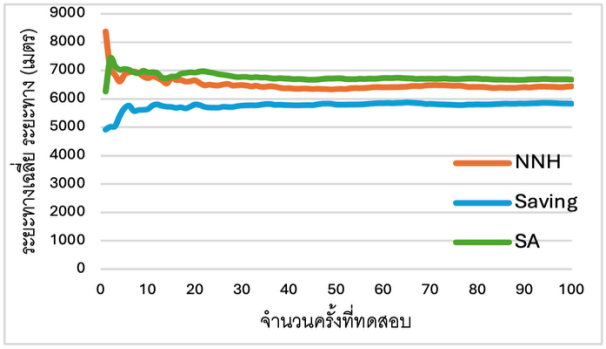
2 การทดสอบซ้ำเพื่อหาค่าเฉลี่ย

หลังจากทำการวิเคราะห์ข้อมูลและได้ผลการทดสอบทั้ง 3 ค่าจากแต่ละเมืองแล้ว เพื่อป้องกันไม่ให้ผลการทดลองคลาดเคลื่อนจึงต้องทำการทดสอบซ้ำซึ่งเป็นการเก็บข้อมูลจากเส้นทางที่ต่างกันจากการสุ่ม เพื่อนำค่าเฉลี่ยที่ได้ไปบันทึกผลเพื่อขั้นตอนต่อไป ในด้านจำนวนครั้งที่ทดสอบซ้ำนั้น ผู้จัดทำได้เลือกใช้เมือง Crested Butte, Colorado ซึ่งมีจำนวนจุดตัดถนนทั้งหมด 98 จุด เป็นเมืองสำหรับการทดสอบเพื่อหาว่า

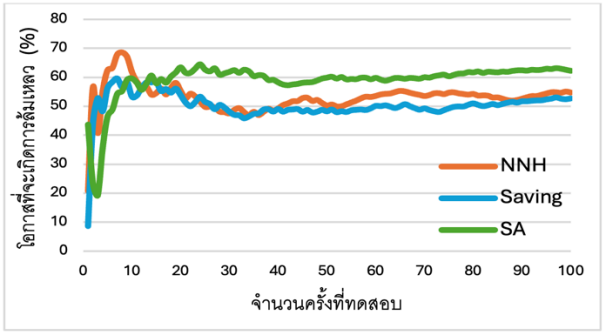
ชื่อเมืองตัวอย่าง	จำนวนจุดตัดถนนทั้งหมด	จำนวนจุดลูกค้า	เส้นถนน
Crested Butte, CO	98	14	292
Leavenworth, WA	113	16	319
Calistoga, CA	221	31	558
Moab, UT	239	34	620
Sausalito, CA	258	36	635
Marfa, TX	272	38	888
Carmel-by-the-Sea, CA	283	40	881
Jackson, WY	334	47	916
Whitefish, MT	319	45	867
Piedmont, CA	352	50	944
Breckenridge, CO	373	53	875
Healdsburg, CA	455	65	1,146
Bar Harbor, ME	603	86	1,430
Park City, UT	606	86	1463
Taos, NM	620	88	1,500
Boerne, TX	718	102	1736
Ouray, CO	749	107	1775
Big Bear Lake, CA	866	123	2220
Ashland, OR	927	132	2497
Boulder City, NV	976	139	2358
Truckee, CA	1027	146	2420
Sedona, AZ	1032	147	2385
Los Gatos, CA	1082	154	2441

ตารางที่ 1 แสดงข้อมูลเมืองที่ใช้ศึกษา

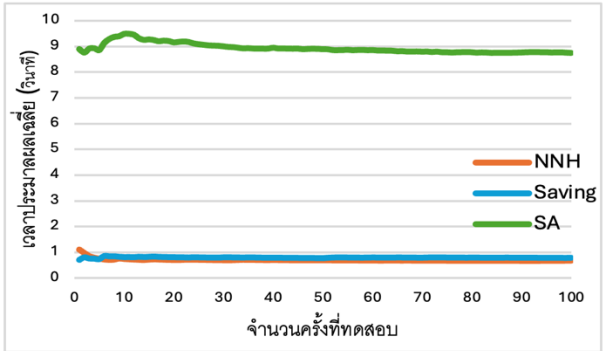
จำเป็นต้องทำการทดสอบซ้ำกี่ครั้งเพื่อให้ค่าเฉลี่ยที่ได้มีความคลาดเคลื่อนน้อยที่สุด โดยหลังจากเพิ่มจำนวนการทดสอบตั้งแต่ 1 ครั้งไปจนถึง 100 ครั้ง และหาค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐานจากข้อมูลที่เพิ่มขึ้นทีละรอบ



ภาพที่ 3 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับค่าเฉลี่ยของระยะทางรวมตลอดทั้งเส้นทางการขนส่งของเมือง Crested Butte, Colorado

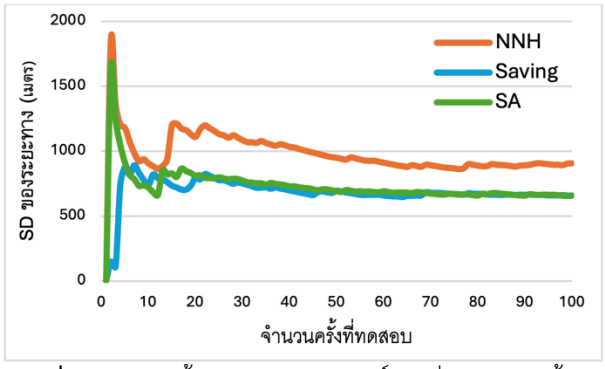


ภาพที่ 4 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับค่าเฉลี่ยของโอกาสที่จะเกิดการล้มเหลวของเมือง Crested Butte, Colorado

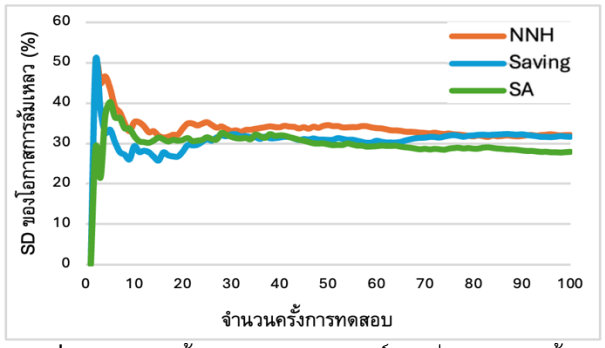


ภาพที่ 5 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับค่าเฉลี่ยของระยะเวลาที่ใช้ในการประมวลผล

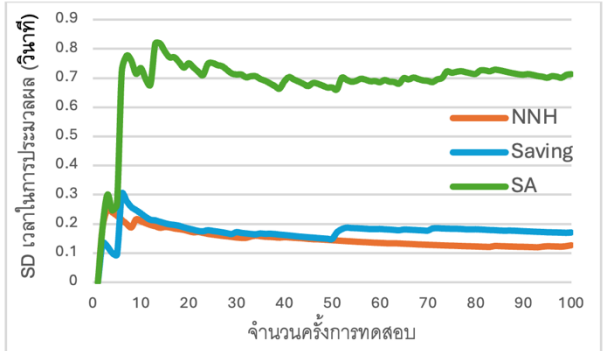
จากผลการทดสอบซ้ำจำนวน 100 ครั้ง พบว่าค่าผลลัพธ์จากทั้งสี่ประเภทของกราฟ ได้แก่ ค่าเฉลี่ยของระยะทางรวม ค่าเฉลี่ยของโอกาสเกิดการล้มเหลว ค่าเฉลี่ยของระยะเวลาที่ใช้ในการประมวลผล และส่วน



ภาพที่ 6 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับส่วนเบี่ยงเบนมาตรฐานของระยะทางรวมตลอดทั้งเส้นทางการขนส่งของเมือง Crested Butte.



ภาพที่ 7 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับส่วนเบี่ยงเบนมาตรฐานของโอกาสที่จะเกิดการล้มเหลวของเมือง Crested Butte.



ภาพที่ 8 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างจำนวนครั้งในการทดสอบซ้ำกับส่วนเบี่ยงเบนมาตรฐานของระยะเวลาที่ใช้ในการประมวลผลของเมือง Crested Butte, Colorado

เบี่ยงเบนมาตรฐานของแต่ละตัวแปร มีแนวโน้มเข้าสู่ภาวะคงที่หลังจากจำนวนการทดสอบซ้ำเพิ่มขึ้น แม้ในช่วงเริ่มต้น โดยเฉพาะ 10–20 ครั้งแรก ค่าของบางอัลกอริทึม เช่น SA และ NNH ในบางตัวแปร จะมีความผันผวนสูงกว่าวิธี Saving ซึ่งสะท้อนถึงความไวต่อการเปลี่ยนแปลงในช่วงเริ่มต้นและความไม่เสถียรของตัวแปรในระยะต้น แต่หลังจากประมาณ 50 ครั้งของการทดสอบ ค่าผลลัพธ์ของทุกอัลกอริทึมในทุกตัวแปรเริ่มลดการแปรผันและเข้าสู่ช่วงที่มีความเสถียรสูงขึ้นอย่างชัดเจน

นอกจากนี้ กราฟส่วนเบี่ยงเบนมาตรฐานแสดงให้เห็นว่าทั้งสามอัลกอริทึมมีแนวโน้มลดความผันผวนของผลลัพธ์ลงเมื่อจำนวนการทำซ้ำเพิ่มขึ้น โดยอัลกอริทึม Saving มีค่าความแปรผันต่ำสุดตั้งแต่ช่วงต้น ขณะที่ SA และ NNH แม้จะเริ่มต้นด้วยค่าผันผวนสูงกว่า แต่ก็สามารถเข้าสู่ความเสถียรได้ภายหลัง ผลดังกล่าวชี้ว่าเมื่อถึงระดับการทำซ้ำประมาณ 50 ครั้ง แบบจำลองทั้งสามสามารถให้ค่าที่มีความคงที่และเชื่อถือได้ ดังนั้น การเพิ่มจำนวนการทดสอบซ้ำเกินกว่า 50 ครั้งอาจไม่ก่อให้เกิดการเปลี่ยนแปลงที่มีนัยสำคัญต่อผลลัพธ์ และสามารถใช้ 50 ครั้งเป็นเกณฑ์ที่เหมาะสมเพื่อลดเวลาและทรัพยากรในการทดสอบโดยไม่กระทบต่อความถูกต้องของข้อมูล

ชื่อเมืองตัวอย่าง	จำนวนจุดตัดถนน	ระยะทางรวมตลอดทั้งเส้นทางการขนส่ง (เมตร)		
		NNH	Saving	SA
Crested Butte, CO	98	6594.26	5975.48	5857.30
Leavenworth, WA	113	7934.05	7073.88	6640.79
Calistoga, CA	221	20340.68	18356.76	18305.49
Moab, UT	239	25300.93	22011.64	21580.02
Sausalito, CA	258	21316.74	18438.08	19143.54
Marfa, TX	272	15566.37	13955.38	13552.38
Carmel-by-the-Sea, CA	283	14647.64	12624.75	12563.87
Jackson, WY	334	33922.35	28705.02	29893.32
Whitefish, MT	319	32454.29	28564.28	28873.43
Piedmont, CA	352	25705.40	21772.30	23311.06
Breckenridge, CO	373	66256.69	59060.36	67362.94
Healdsburg, CA	455	41012.65	35388.94	43241.81
Bar Harbor, ME	603	127145.38	109821.84	149619.99
Park City, UT	606	88204.72	77903.88	105835.20
Taos, NM	620	63072.84	52873.42	73279.52
Boerne, TX	718	84543.07	73762.75	129359.45
Ouray, CO	749	409919.77	344498.56	512178.44
Big Bear Lake, CA	866	82151.38	68491.63	123,928.89
Ashland, OR	927	71781.17	62782.27	122488.43
Boulder City, NV	976	101276.22	85590.53	171532.38
Truckee, CA	1027	181676.95	96398.51	398599.84
Sedona, AZ	1032	111506.39	162393.00	192711.14
Los Gatos, CA	1082	118229.50	101999.99	208874.46

ตารางที่ 2 แสดงข้อมูลระยะทางของเส้นทางในแต่ละเมืองที่สร้างขึ้นโดยวิธีการสร้างเส้นทางที่แตกต่างกัน

3 ผลการทดลอง

ให้ข้อมูลของจำนวนจุดตัดถนนทั้งหมด แทนจุดของบ้านที่อยู่บนถนนระหว่างจุดตัดถนนนั้น ๆ เพื่อลดความซับซ้อนในการวิเคราะห์ แล้วจึงกำหนดข้อมูลเมืองและหาเส้นทาง

ตารางที่ 2 แสดงข้อมูลระยะทางโดยพบว่าระยะทางที่ถูกสร้างขึ้นในแต่ละวิธีแปรผันตรงกับจำนวนจุดตัดถนนทั้งหมด และพบว่ามีค่าผิดปกติ (Outlier) คือเมือง Ouray, Colorado และ Truckee, California ที่มีค่าสูงเกินปกติ

4 วิเคราะห์ผลการทดลอง

จากแผนภูมิและตารางข้างต้น ทำให้เห็นว่าวิธีการสร้างเส้นทางแต่ละรูปแบบจะให้ผลลัพธ์ต่างกันอย่างเห็นได้ชัดโดยแต่ละวิธีการสร้างจะมีลักษณะที่สังเกตได้ดังนี้

4.1 Nearest Neighbor Heuristic

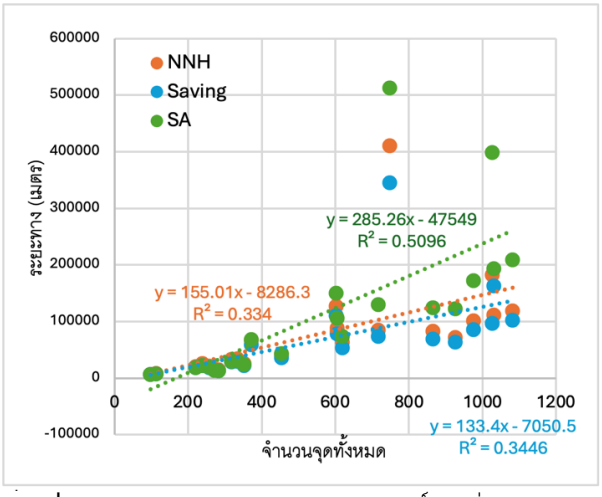
จากภาพที่ 10 ระยะทางที่ได้มีความสัมพันธ์แปรผันตรงกับจำนวนจุดทั้งหมดโดย NNH มีความชันอยู่ที่ 110.37 เมตรต่อจุด และในด้านโอกาสที่จะเกิดการล้มเหลวนั้นมีการกระจายที่สูงแต่แนวโน้มมีค่าคงที่อยู่ในช่วงร้อยละ

ชื่อเมืองตัวอย่าง	จำนวนจุดตัดถนน	โอกาสที่จะเกิดการล้มเหลว (%)		
		NNH	Saving	SA
Crested Butte, CO	98	51.01	52.23	59.27
Leavenworth, WA	113	70.73	57.36	63.68
Calistoga, CA	221	74.07	64.55	71.11
Moab, UT	239	60.27	67.88	63.26
Sausalito, CA	258	67.47	69.66	62.93
Marfa, TX	272	66.66	58.40	63.44
Carmel-by-the-Sea, CA	283	63.15	58.89	66.19
Jackson, WY	334	71.26	74.44	71.75
Whitefish, MT	319	61.10	54.70	69.30
Piedmont, CA	352	65.11	62.33	75.44
Breckenridge, CO	373	70.64	60.27	69.32
Healdsburg, CA	455	74.29	78.79	67.26
Bar Harbor, ME	603	74.41	54.25	65.60
Park City, UT	606	73.48	63.07	69.60
Taos, NM	620	69.91	62.84	62.92
Boerne, TX	718	68.06	74.57	68.73
Ouray, CO	749	75.97	74.90	77.48
Big Bear Lake, CA	866	73.86	65.48	61.22
Ashland, OR	927	64.75	68.97	67.50
Boulder City, NV	976	67.33	72.95	71.55
Truckee, CA	1027	64.30	63.54	65.83
Sedona, AZ	1032	73.25	70.23	70.34
Los Gatos, CA	1082	69.73	76.65	70.14

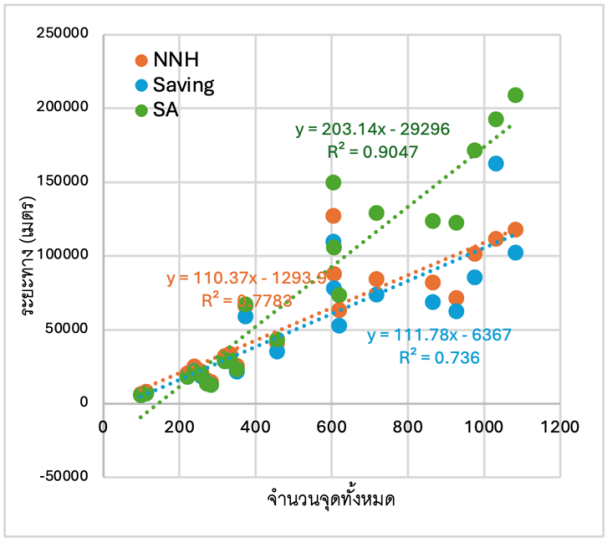
ตารางที่ 3 แสดงข้อมูลโอกาสที่จะเกิดการล้มเหลวของเส้นทางในแต่ละเมืองที่สร้างขึ้นโดยวิธีการสร้างเส้นทางที่แตกต่างกันในกรณีโอกาสเกิดการตีกลับที่ร้อยละ 30

ชื่อเมืองตัวอย่าง	จำนวนจุดตัดถนน	ระยะเวลาที่ใช้ในการประมวลผล (วินาที)		
		NNH	Saving	SA
Crested Butte, CO	98	1.15	0.58	8.88
Leavenworth, WA	113	1.33	1.09	10.23
Calistoga, CA	221	1.44	1.13	25.49
Moab, UT	239	1.86	1.51	28.76
Sausalito, CA	258	2.04	1.81	30.81
Marfa, TX	272	1.66	1.59	42.69
Carmel-by-the-Sea, CA	283	2.82	2.49	45.74
Jackson, WY	334	2.8	2.62	63.17
Whitefish, MT	319	4.85	4.04	53.04
Piedmont, CA	352	4.08	3.86	59.34
Breckenridge, CO	373	3.72	5.13	77.53
Healdsburg, CA	455	5.00	6.84	102.51
Bar Harbor, ME	603	8.92	13.85	146.92
Park City, UT	606	10.40	14.61	177.53
Taos, NM	620	10.11	14.77	180.06
Boerne, TX	718	14.60	16.55	254.13
Ouray, CO	749	18.84	25.03	244.42
Big Bear Lake, CA	866	27.77	38.18	362.97
Ashland, OR	927	34.22	48.33	513.94
Boulder City, NV	976	28.63	49.14	564.07
Truckee, CA	1027	34.81	56.57	577.27
Sedona, AZ	1032	38.64	57.46	517.80
Los Gatos, CA	1082	46.04	65.72	603.01

ตารางที่ 4 แสดงข้อมูลเวลาที่ใช้ในการประมวลผลของเส้นทางในแต่ละเมืองที่สร้างขึ้นโดยวิธีการสร้างเส้นทางที่แตกต่างกัน



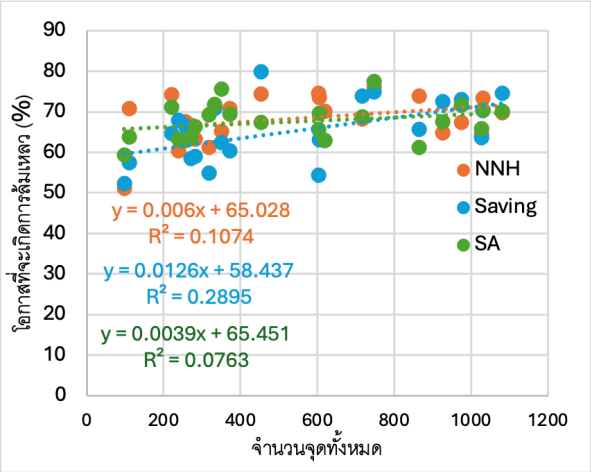
ภาพที่ 9 แผนภูมิกระจายแสดงความสัมพันธ์ระหว่างจำนวนจุดตัดถนนทั้งหมดกับระยะทางรวมตลอดทั้งเส้นทางการขนส่งพร้อมเส้นแนวโน้มแบบเชิงเส้น



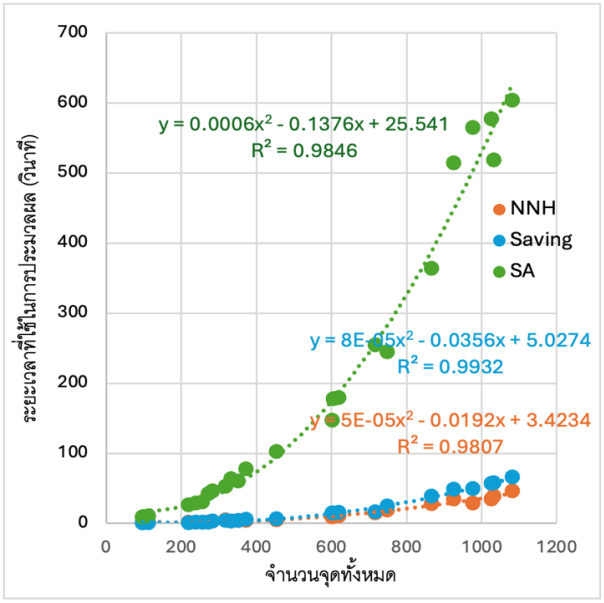
ภาพที่ 10 แผนภูมิกระจายแสดงความสัมพันธ์ระหว่างจำนวนจุดตัดถนนทั้งหมดกับระยะทางรวมตลอดทั้งเส้นทางการขนส่งพร้อมเส้นแนวโน้มแบบเชิงเส้นในกรณีที่ไม่นำค่าผิดปกติ (outlier) มาใช้

4.2 Saving Algorithm

วิธีนี้แม้จะมีผลลัพธ์ที่ค่อนข้างใกล้เคียงกับ NNH แต่เมื่อพิจารณาอย่างละเอียดจะพบว่าระยะทางที่ได้จะสั้นกว่า โดยเฉพาะอย่างยิ่งในเมืองขนาดใหญ่ และเมื่อพิจารณาโอกาสที่จะเกิดการล้มเหลวจะพบว่ามีผลลัพธ์โดยรวมดีกว่าวิธีการอื่น ๆ โดยจะเห็นความต่างได้อย่างชัดเจนในเมืองขนาดเล็กตามภาพที่ 11



ภาพที่ 11 แผนภูมิกระจายแสดงความสัมพันธ์ระหว่างจำนวนจุดตัดถนนทั้งหมดกับโอกาสที่จะเกิดการล้มเหลวพร้อมเส้นแนวโน้มแบบเชิงเส้น



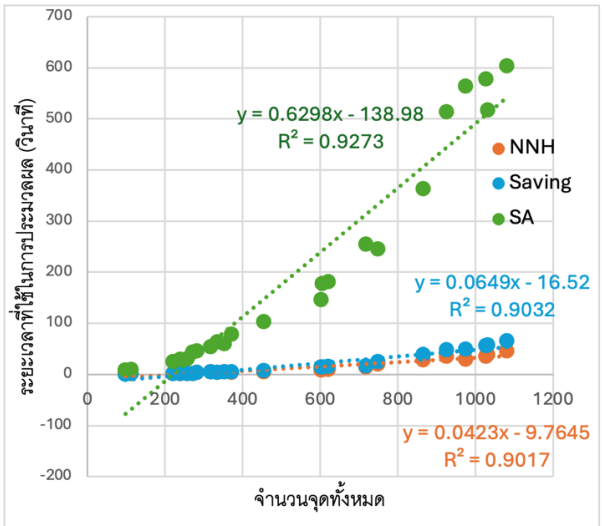
ภาพที่ 12 แผนภูมิกระจายแสดงความสัมพันธ์ระหว่างจำนวนจุดตัดถนนทั้งหมดกับเวลาที่ใช้ในการประมวลผลพร้อมเส้นแนวโน้มแบบพหุนาม

4.3 Simulated Annealing

ผลที่ได้คือระยะทางที่สั้นสำหรับเมืองเล็กแต่หากเป็นเมืองใหญ่แล้วจะเห็นได้จากเส้นแนวโน้มว่าระยะทางเพิ่มขึ้นอย่างรวดเร็วในส่วนของโอกาสล้มเหลวนั้นจะมีการกระจายตัวที่มากแต่ก็คงที่อยู่ในช่วงร้อยละ 60 ถึง 80 เหมือนกับ NNH และสุดท้าย SA เป็นวิธีที่ใช้เวลาประมวลผลนานเป็นอย่างมาก ดังภาพที่ 12 และ 13

4.4 ความน่าจะเป็นของการตีกลับพัสดุ

จากภาพที่ 14-16 ได้แสดงผลในการศึกษาค่าความน่าจะเป็นของการตีกลับพัสดุที่แตกต่างกัน โดยที่ p = 0 โอกาสการล้มเหลวเป็น 0 แสดงให้เห็นว่าหากไม่มีการตีกลับ ระบบจะไม่มีการล้มเหลวและที่ p = 0.05-0.35 มีการเติบโตของโอกาสการล้มเหลวที่สูงขึ้นและค่าที่ปลายของช่วงนี้ มีการเติบโตของโอกาสการล้มเหลวน้อยลง ก่อนเข้าใกล้การล้มเหลว 100% ที่ p = 0.40-0.50



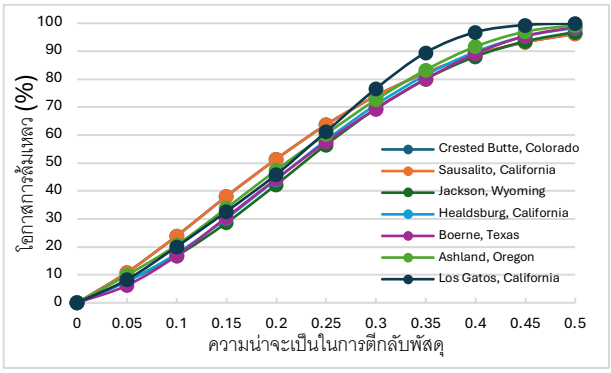
ภาพที่ 13 แผนภูมิกระจายแสดงความสัมพันธ์ระหว่างจำนวนจุดตัดถนนทั้งหมดกับเวลาที่ใช้ในการประมวลผลพร้อมเส้นแนวโน้มแบบเชิงเส้น

4. สรุปผลการทดลอง

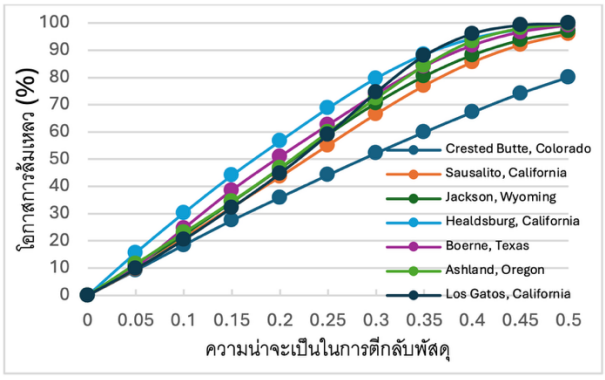
จากการศึกษาพบว่า แบบจำลองคณิตศาสตร์ที่สร้างขึ้นมาสามารถนำไปใช้เพื่อแสดงถึงเส้นทางการเดินรถที่เหมาะสมที่สุดสำหรับการขนส่งในกรณีที่มีโอกาสเกิดการตีกลับของพัสดุได้ โดยที่เป็นเช่นนี้เพราะหากแบ่งประเภทตามงานวิจัยของ Liu F. และคณะ (2023) NNH และ Saving Algorithm เป็นวิธีการประเภทเดียวกันนั่นคือ ฮิวริสติกส์การสร้าง (constructive heuristics) ซึ่งเป็นการสร้างเส้นทางขึ้นมาทำให้ระยะทางที่ได้มีแนวโน้มใกล้เคียงกัน ในขณะที่ SA เป็นวิธีอีกประเภทนั่นคือ เมตาฮิวริสติกส์ (metaheuristics) โดย SA จะเป็นเพียงการปรับเส้นทางเดิมซึ่งเริ่มต้นจากการสุ่มให้ดีขึ้นเรื่อย ๆ ทำให้ในเมืองขนาดใหญ่ซึ่งมีจุดจำนวนมาก เป็นไปได้ยากที่จะสุ่มเจอรูปแบบจากการสุ่มที่จะทำให้ระยะทางสั้นลง แต่อย่างไรก็ดี เมื่อใช้ SA กับเมืองที่มีจุดน้อยจะเห็นได้ว่าสามารถที่จะสุ่มเจอรูปแบบที่ได้ระยะทางสั้นลงได้จนทำให้ได้ผลลัพธ์ดีกว่า NNH และ Saving Algorithm ในแง่โอกาสที่จะเกิดการล้มเหลวพบว่ามีแนวโน้มใกล้เคียงกันในทุกวิธีการสร้างเส้นทาง อย่างไรก็ดีในเมืองขนาดเล็กจะพบว่า Saving Algorithm จะมีโอกาสที่จะเกิดการล้มเหลวต่ำกว่าวิธีการอื่น ๆ เล็กน้อย และส่วนของเวลาการประมวลผล NNH ซึ่งเป็นวิธีที่เรียบง่ายที่สุดก็จะใช้เวลาน้อยที่สุด ซึ่ง Saving Algorithm จะใช้เวลาที่ใกล้เคียงกัน และ SA นั้นจะมีแนวโน้มการใช้เวลามากกว่าเป็นอย่างมาก

ในการศึกษาความน่าจะเป็นของการตีกลับพัสดุที่ต่างกัน ได้ความสัมพันธ์ของความน่าจะเป็นของการตีกลับพัสดุและโอกาสที่จะเกิดการล้มเหลวเป็นความสัมพันธ์เชิงโค้ง (nonlinear) และใกล้เคียง sigmoid curve ที่จะเข้าสู่ค่าสูงสุดที่ p = 0.40-0.50

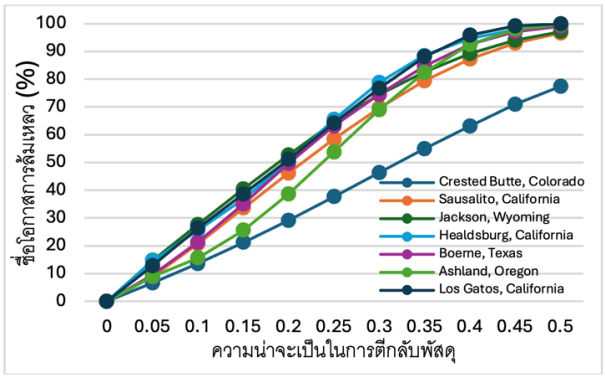
ในงานวิจัยนี้ ไม่มีการแก้ปัญหาโดยใช้วิธี Multi-objective optimization โดยตรง เส้นทางที่ได้มาจากอัลกอริทึมแต่ละแบบจะถูกประเมินภายหลังด้วยฟังก์ชันความล้มเหลว (failure rate) ผ่านการจำลอง ดังนั้น การลดสมการ (2) เป็นผลทางอ้อมจากโครงสร้าง



ภาพที่ 14 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างความน่าจะเป็นในการตีกลับพัสดุกับโอกาสการล้มเหลวด้วยการสร้างเส้นทางแบบ NNH



ภาพที่ 15 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างความน่าจะเป็นในการตีกลับพัสดุกับโอกาสการล้มเหลวด้วยการสร้างเส้นทางแบบ Saving



ภาพที่ 16 แผนภูมิเส้นแสดงความสัมพันธ์ระหว่างความน่าจะเป็นในการตีกลับพัสดุกับโอกาสการล้มเหลวด้วยการสร้างเส้นทางแบบ SA

เส้นทาง ไม่ได้ถูกนำมา optimize ระหว่างสร้างเส้นทาง สุดท้ายแล้วในแง่ของการประยุกต์ใช้นั้น ผู้ใช้จำเป็นต้องให้นิยามว่าเส้นทางที่เหมาะสมที่สุดพิจารณาจากระยะทาง โอกาสที่จะล้มเหลว หรือระยะเวลาที่ใช้ในการสร้าง

5. อ้างอิง

Boeing, G. (2024). Modeling and Analyzing Urban Networks and Amenities with OSMnx. Working paper. https://geoffboeing.com/publica tions/osmnx-paper/

Chen, K.-T., Dai, Y., Fan, K., & Baba, T. (2015). A particle swarm optimization with adaptive multi-swarm strategy for capacitated vehicle routing problem. *2015 1st International Conference on Industrial Networks and Intelligent Systems (INISCom)*, 79–83. https://doi.org/10.4108/icst.iniscom.2015.258972

Clarke, G. and Wright, J.R. Scheduling of Vehicle Routing Problem from a Central Depot to a Number of Delivery Points, *Operation Research*. (1964). 12:568-581.

Cokyasar, T., Subramanyam, A., Larson, J., Stinson, M., & Sahin, O. (2022). Time-Constrained capacitated vehicle routing problem in urban E-Commerce delivery. *Transportation Research Record Journal of the Transportation Research Board*, 2677(2), 190–203. https://doi.org/10.1177/03611981221124592

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80

Desrochers, M., & Laporte, G. (1991). Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10(1), 27–36. https://doi.org/10.1016/01676377(91)90083-2

Federal Highway Administration. National Household Travel Survey, 2017. Available at https://nhts.ornl.gov, accessed on Oct. 28, 2021.

Harahap, R. F., & Sawaluddin, N. (2023). Study vehicle routing problem using Nearest Neighbor Algorithm. *Journal of Physics Conference Series*, *2421*(1), 012027. https://doi.org/10.1088/1742-6596/2421/1/012027

Liu, F. et al. (2023, March 1). Heuristics for Vehicle Routing Problem: A Survey and Recent Advances. arXiv.org. https://arxiv.org/abs/2303.04147

Loqate. (2021). *Fixing failed deliveries: Why accurate address data is key*. https://info.loqate.com/hubfs/Loqate_Fixing%20failed%20deliveries.pdf

OpenStreetMap. (n.d.). OpenStreetMap. https://www.openstreetmap.org/about

Paessens, H. (1988). The savings algorithm for the vehicle routing problem. *European Journal of Operational Research*, 34(3), 336–344. https://doi.org/10.1016/0377-2217(88)90154-3

Van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). Simulated annealing. *In Springer eBooks* (pp. 7–15). https://doi.org/10.1007/978-94-015-7744-1_2

Wei, L., Zhang, Z., Zhang, D., & Leung, S. C. (2017). A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research*, 265(3), 843–859. https://doi.org/10.1016/j.ejor.2017.08.035

น้ำฝน พาพันธ์, & ภัทราริษฐ์ แก้วประดิษฐ. (2563). การจัดเส้นทางการขนส่งสินค้าโดยวิธีอัลกอริธึมแบบประหยัด กรณีศึกษา: โรงงานเม็ดพลาสติก [โครงงานวิศวกรรมศาสตรบัณฑิต]. *มหาวิทยาลัยธุรกิจบัณฑิตย์.*

กิตติกรรมประกาศ

ในงานวิจัยฉบับนี้ สำเร็จลุล่วงได้อย่างสมบูรณ์ด้วยความกรุณาอย่างยิ่งจากอาจารย์ ดร.มนสิการ จันทร์สร้าง ที่ได้สละเวลาอันมีค่าแก่คณะผู้วิจัย เพื่อให้คำปรึกษา ความรู้ ตลอดจนตรวจทานแก้ไขข้อบกพร่องต่าง ๆ ด้วยความเอาใจใส่อย่างยิ่ง จนงานวิจัยฉบับนี้เสร็จสมบูรณ์ลุล่วงได้ด้วยดี คณะผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้ และสุดท้ายนี้คณะผู้วิจัยหวังว่างานวิจัยฉบับนี้คงเป็นประโยชน์สำหรับหน่วยงานที่เกี่ยวข้อง และผู้ที่สนใจศึกษาต่อไป