การแก้ปัญหาการรับส่งเอกสารโดยวิธีการค้นหาย่านคำตอบข้างเคียงขนาดใหญ่

Solving the Messenger problem using Large Neighborhood Search

ธนาวัฒน์ สิทธิสันติกุล[1,*]

[1] สาขาวิชาวิศวกรรมขนส่ง ภาควิชาวิศวกรรมโยธา คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพมหานคร

**Corresponding author; E-mail address: Thanawat.Sit@Outlook.com*

บทคัดย่อ

งานวิจัยนี้จัดทำขึ้นเพื่อศึกษาปัญหาการจัดเส้นทางการรับส่งเอกสาร ซึ่งนำเสนอการนำเส้นทางการรับส่งสินค้าแต่ละงานมารวมกันเป็นปัญหาการรับและส่ง โดยสร้างแบบจำลองทางคณิตศาสตร์ที่พิจารณาถึงข้อจำกัดต่าง ๆ ได้แก่ ความสามารถในการบรรทุกของยานพาหนะ และกรอบเวลาในการให้บริการ ซึ่งมีเป้าหมายในการหาเส้นทางที่ดีที่สุดทั้งในแง่ของระยะทางที่สั้นที่สุด และการใช้ยานพาหนะจำนวนน้อยที่สุด ทั้งนี้ได้กำหนดขอบเขตของการศึกษา ซึ่งมีสมมติฐานและลักษณะของปัญหาที่สำคัญ ได้แก่ เส้นทางที่ดีที่สุดคือเส้นทางที่สั้นที่สุดโดยไม่ได้พิจารณาผลของการจราจร, พัสดุทุกชนิดมีขนาดเท่ากัน, ยานพาหนะทุกคันมีความสามารถในการบรรทุกเหมือนกัน และ ให้ความสำคัญกับเวลาการให้บริการ งานวิจัยนี้เลือกใช้วิธีการค้นหาย่านคำตอบขนาดใหญ่มาใช้ ซึ่งสามารถแก้ปัญหาได้อย่างรวดเร็ว และเหมาะกับปัญหาขนาดใหญ่มาใช้ในการแก้ปัญหาของการศึกษานี้ จากการศึกษาพบว่า การให้บริการแบบรวมงานสามารถลดต้นทุนการขนส่งรวมได้อย่างมีนัยสำคัญเมื่อเทียบกับการให้บริการแบบทั่วไป

คำสำคัญ: ปัญหาการรับส่งพัสดุ, วิธีการค้นหาคำตอบข้างเคียงขนาดใหญ่, เมตาฮิวริสติกส์, ปัญหาการรับส่งเอกสาร

Abstract

This research studies the vehicle routing problem of parcel shipping. Which, instead of using one trip per job, this study introduces a more efficient approach by combining several jobs and turning them into a series of pickup and delivery services. Large Neighborhood Search Metaheuristics is used to solve a mathematical model formulated based on the characteristics of problem such as vehicle capacity and service time windows, seeking the most efficient route and pickup and delivery order. For simplicity, the impact of road traffic is disregarded. The study shows that the total cost of this approach is significantly lower compared to the conventional method.

Keywords: Pickup and Delivery Problem, Large Neighborhood Search, Metaheuristics, Messenger Problem

1. บทนำ

ปัญหาการรับส่งเอกสารในงานวิจัยนี้มีลักษณะหลายประการที่คล้ายคลึงกับปัญหาการรับส่งสินค้า (Pickup and Delivery Problem; PDP) ซึ่งเป็นปัญหารูปแบบหนึ่งของปัญหาการจัดเส้นทางเดินรถ (Vehicle Routing Problem; VRP) ที่ได้รับการศึกษาและพัฒนาอย่างแพร่หลายในระยะหลัง

สำหรับปัญหาการจัดเส้นทางเดินรถนั้นเป็นรูปแบบทั่วไปของปัญหาที่เกี่ยวข้องกับการจัดเส้นทางการให้บริการขนส่งที่มีผลต่อความพึงพอใจของลูกค้าอย่างมีนัยสำคัญ เป็นแนวคิดหนึ่งที่พยายามใช้ทรัพยากรของธุรกิจให้ได้อย่างมีประสิทธิภาพสูงสุด โดยการลดต้นทุนส่วนเกินที่ไม่จำเป็น กล่าวคือเป็นการหาเส้นทางที่มีต้นทุนต่ำที่สุด โดยที่ยังคงสามารถให้บริการลูกค้าได้อย่างมีประสิทธิภาพ

จากที่กล่าวไว้ข้างต้น ปัญหาการรับส่งเอกสาร (The Messenger Problem) เป็นรูปแบบหนึ่งของปัญหาการจัดเส้นทางเดินรถ (Vehicle Routing Problem; VRP) ที่มีลักษณะเฉพาะที่แตกต่างจากปัญหาทั่วไป ได้แก่

1. Pairing and Priority Characteristic: เป็นลักษณะของลูกค้า ซึ่งต้องประกอบด้วยกิจกรรมการรับ (Pickup) เอกสาร ณ จุดต้นทาง และส่ง (Delivery) ไปยังจุดปลายทาง ทั้งนี้ กิจกรรมทั้งสองต้องเกิดขึ้นโดยใช้ยานพาหนะคันเดียวกันและเกิดขึ้นตามลำดับ

2. Unit Demand: สำหรับปัญหาในงานวิจัยนี้ กำหนดให้น้ำหนักบรรทุกของลูกค้าแต่ละรายเป็นหนึ่งหน่วยเท่ากัน

3. Time-Independent Travel Times: ยานพาหนะที่ใช้ในงานวิจัยนี้คือ จักรยานยนต์ ที่ได้รับผลกระทบจากผลจากสภาพการจราจรน้อยกว่ายานพาหนะประเภทอื่น ๆ อย่างมีนัยสำคัญ ดังนั้น จึงกำหนดให้ระยะเวลาที่ใช้ในการเดินทางแปรผันตรงกับระยะทางในการเดินทางเท่านั้น

4. Time Windows: เป็นกรอบเวลาการให้บริการของลูกค้าแต่ละราย เป็นการกำหนดเวลาที่การบริการต้องเริ่มต้นขึ้นภายในช่วงเวลาดังกล่าว

สำหรับแนวโน้มในการพัฒนาการแก้ปัญหาการจัดเส้นทางเดินรถในปัจจุบัน [1] งานวิจัยในระยะหลังนั้นมีเป้าหมายในการทำให้สามารถปรับใช้

จริงได้มากขึ้น โดยมีการเพิ่มลักษณะต่าง ๆ ที่มีความสมจริงมากยิ่งขึ้น เช่น ลักษณะลูกค้าแบบพลวัต, ยานพาหนะที่มีลักษณะต่างกัน เป็นต้น ทำให้ลักษณะของปัญหามีความซับซ้อนมากยิ่งขึ้น ทำให้วิธีการแก้ปัญหาโดยใช้ Exact Methods เริ่มไม่มีประสิทธิภาพมากพอกับความต้องการของผู้ให้บริการที่ต้องการความรวดเร็วมากขึ้น ดังนั้น วิธีการประมาณค่าในรูปแบบต่าง ๆ จึงได้รับการพัฒนามากยิ่งขึ้นกว่าในอดีต [2, 3]

โดยทั่วไป วิธีการแก้ปัญหาการจัดเส้นทางสามารถแบ่งได้เป็น 2 กลุ่มหลัก ได้แก่ 1. กระบวนการหาคำตอบที่ดีที่สุดอย่างแท้จริง หรือ Exact Methods และ 2. กระบวนการประมาณคำตอบที่ดีที่สุด หรือ Approximate Methods ซึ่งวิธีการแก้ปัญหาประเภทที่ 1. นั้น ให้ผลลัพธ์เป็นคำตอบที่ดีที่สุดอย่างแท้จริง (Optimal Solution) โดยข้อเสียที่สำคัญของวิธีประเภท 1. ได้แก่ ถึงแม้ว่าระบบให้ผลลัพธ์เป็นคำตอบที่มีคุณภาพสูง แต่จำเป็นต้องใช้เวลาในการประมวลผลเป็นเวลานาน ซึ่งไม่เหมาะสม ดังนั้น จากการสำรวจงานวิจัยส่วนใหญ่ที่เกี่ยวข้องโดย [1] พบว่า มักใช้วิธีการแก้ปัญหาในกลุ่มที่ 2. เป็นหลัก เนื่องจาก สามารถแก้ปัญหาได้อย่างรวดเร็ว และให้คำตอบที่มีคุณภาพ ดีพอ สำหรับการนำไปใช้จริง

Large Neighborhood Search (LNS) เป็นวิธีการค้นหาคำตอบ ที่ดีที่สุดแบบหนึ่ง โดยจัดอยู่ในกลุ่ม Metaheuristic ซึ่งเป็นกลุ่มวิธีการประมาณคำตอบที่ดีที่สุด (Approximate Methods) รูปแบบหนึ่ง [4] สำหรับกระบวนการ LNS ประกอบด้วย 2 ขั้นตอนใหญ่ ได้แก่ กระบวนการกำจัด (Removing Method) และ กระบวนการสร้างเส้นทางใหม่ (Reinserting Method) ซึ่งจะเป็นการพัฒนาคำตอบแบบวนรอบซ้ำ โดยในแต่ละรอบนั้น ระบบจะทำการเลือกกำจัดลูกค้าบางส่วนออกจากเส้นทางผ่านกระบวนการกำจัด และ แทรกกลับไปใหม่ผ่านกระบวนการสร้างเส้นทาง [5]

1.1 วัตถุประสงค์

1. เพื่อสร้างแบบจำลองเพื่อแก้ปัญหาการจัดเส้นทาง (Vehicle Routing Problem) เพื่อใช้สำหรับปัญหาการขนส่งเอกสาร (Messenger Problem)
2. เพื่อทดสอบและพัฒนาแบบจำลองการแก้ปัญหาในข้อ 1 ให้สามารถหาคำตอบที่เหมาะสม (ดี) ได้อย่างมีประสิทธิภาพ โดยใช้ Li and Lim Benchmark [6] เป็นมาตรฐานวัดประสิทธิภาพ

1.2 ขอบเขตการวิจัย

1. ใช้กระบวนวิธีแก้ปัญหา Large Neighborhood Search
2. ข้อมูลทั้งหมดมีลักษณะเป็นแน่นอน (Deterministic)
3. แบบจำลองมีลักษณะเป็นแบบจุดเริ่มต้นเดียว (Single Depot)
4. ยานพาหนะทุกคันมีลักษณะเหมือนกัน (Homogeneous Fleet)
5. ลูกค้าทุกคนมีปริมาณเอกสารที่เข้าสู่บริการเท่ากัน คือ 1 หน่วย
6. ระยะเวลาที่ใช้ในการเดินทางแปรผันตรงกับระยะทางในการเดินทางเท่านั้น (Time-Independent Travel Times)
7. ลักษณะการรับส่งแบบ One-to-One และ สามารถรับหรือส่งพร้อมกันได้ (P-D)

2. ทฤษฎีที่เกี่ยวข้อง

2.1 ปัญหาการรับส่งสินค้า

ปัญหาการรับส่งสินค้า (Pickup and Delivery Problem; PDP) เป็นลักษณะหนึ่งของปัญหาการจัดเส้นทางเดินรถ (Vehicle Routing Problem; VRP) [7] ที่มีลักษณะในการให้บริการรับ (Pickup) คนหรือสินค้าที่จุดเริ่มต้น และทำการส่ง (Delivery) ไปยังจุดปลายทาง [6] ด้วยการกำหนดปัญหาในลักษณะนี้ทำให้เกิดเงื่อนไขเพิ่มเติมจากปัญหาการจัดเส้นทางเดินรถทั่วไป ได้แก่

1. Pairing Constraint: เป็นเงื่อนไขที่ระบุให้การรับ (Pickup) และการส่งสินค้า (Delivery) ของลูกค้าแต่ละรายต้องเกิดขึ้นด้วยยานพาหนะคันเดียวกัน
2. Priority Constraint: เป็นเงื่อนไขที่ระบุให้กิจกรรมการรับ (Pickup) ต้องเกิดขึ้นก่อนการส่งสินค้า (Delivery) เสมอสำหรับลูกค้าแต่ละราย

โดยคำตอบของปัญหาการรับส่งสินค้าต้องเป็นไปตาม 4 ลักษณะนี้ [7] ได้แก่

1. จำเป็นต้องรับ และ ส่งสินค้าแก่ลูกค้าทุกรายให้สำเร็จ
2. ไม่มีการเปลี่ยนถ่ายสินค้าที่ Node อื่น ๆ นอกจาก Depot เท่านั้น
3. ปริมาณสินค้าที่บรรทุกต้องไม่เกินความสามารถในการบรรทุกของรถ
4. ต้นทุนรวม (ของทุกเส้นทาง) ของคำตอบมีค่าต่ำที่สุด

2.2 ปัญหาการจัดเส้นทางเดินรถ

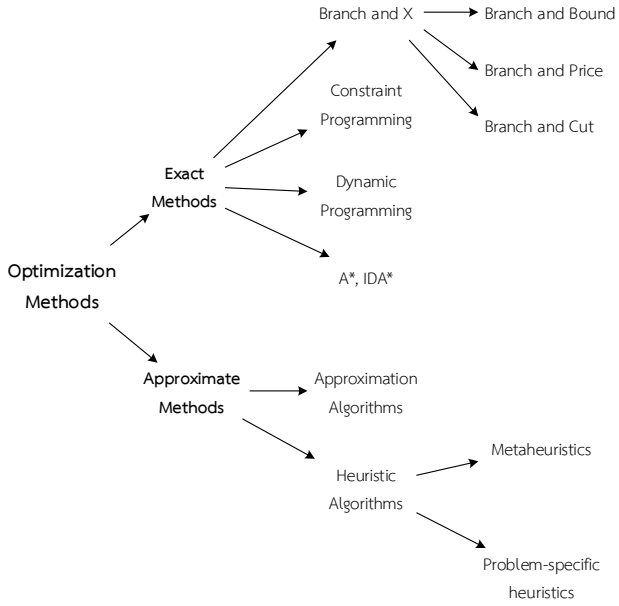
ปัญหาการจัดเส้นทางการเดินรถ (Vehicle Routing Problems; VRP) เป็นปัญหาเกี่ยวข้องทั้งกับการขนส่งบุคคล และ สินค้า [8] โดยเป็นปัญหารูปแบบหนึ่งของการวิจัยเชิงปฏิบัติการ (Operations Research) ที่ได้รับการพัฒนาอย่างต่อเนื่อง [9] และจัดให้อยู่วิธีการแก้ปัญหาการหาค่าที่เหมาะสมที่สุดเชิงการจัด (Combinatorial Optimization problem) ซึ่งเป็นปัญหาเอ็นพีแบบยาก (NP-Hard problem) [10] สำหรับเป้าหมายทั่วไปของการแก้ปัญหาการจัดเส้นทางเดินรถ มีดังนี้ [11]

1. เพื่อลดต้นทุนการขนส่งสินค้า และ ต้นทุนด้านยานพาหนะให้ต่ำที่สุด
2. เพื่อหาจำนวนรถ และ คนขับ ที่เหมาะสมที่สุดสำหรับการปฏิบัติการ
3. เพื่อหาระยะเวลาที่เหมาะสม (ดี) ที่สุดสำหรับการปฏิบัติการส่งสินค้า
4. เพื่อหาเส้นทางที่มีระยะทางสั้นที่สุด โดยที่ยังคงสามารถให้บริการลูกค้าทุกคนได้

โดยสรุป ปัญหาการจัดเส้นทางการเดินรถ เป็นการให้บริการกระจายสินค้าที่เหมาะสมเพื่อตอบสนองความต้องการของลูกค้าระหว่างจุดต่าง ๆ บนแผนที่ ในการเลือกพื้นที่การให้บริการขึ้นอยู่กับลักษณะการกระจายตัวของลูกค้า ซึ่ง VRP สามารถประยุกต์ใช้ได้อย่างหลากหลายในสายงานต่าง ๆ เช่น การขนส่งเชื้อเพลิงน้ำมัน, การจัดเที่ยวบินของสายการบินต่าง ๆ, การจัดส่งอาหาร และ เครื่องดื่มให้แก่ร้านอาหาร, การขนส่งเงินให้กับตู้ ATM และ ธนาคาร, การให้บริการรับส่งนักเรียน และ พนักงานจากบ้านถึงโรงเรียน หรือ ที่ทำงาน, การขนส่งสินค้าที่ลูกค้าซื้อสินค้าออนไลน์ และ การเก็บขนส่งขยะจากบ้านไปยังที่ทิ้งขยะส่วนกลาง เป็นต้น [11, 12]

2.3 วิธีการแก้ปัญหาการจัดเส้นทางเดินรถ

วิธีการแก้ปัญหาการจัดเส้นทางทั่วไปสามารถจำแนกออกได้เป็นสองกลุ่มใหญ่ ตามแนวคิดการหาคำตอบที่ดีที่สุด ได้แก่ 1. กระบวนการหาคำตอบที่ดีที่สุดอย่างแท้จริง หรือ Exact Methods และ 2. กระบวนการประมาณคำตอบที่ดีที่สุด หรือ Approximate Methods ซึ่งเป็นกลุ่มที่รวมถึง วิธีการแก้ปัญหาแบบฮิวริสติกส์ (Heuristics) และ เมตาฮิวริสติกส์ (Metaheuristics) [10, 13]



รูปที่ 1 การจำแนกวิธีการแก้ปัญหาการจัดเส้นทางเดินรถ [13]

2.3.1 Exact Methods

กระบวนการหาคำตอบที่ดีที่สุดอย่างแท้จริง (Exact Methods) สามารถแบ่งออกได้เป็นสามประเภท [10] คือ Lagrangian Relaxation-based methods, Column Generation และ Dynamic Programming ซึ่งจะพิจารณาตรวจสอบทุกคำตอบที่เป็นไปได้ (Feasible Solution) จนได้ผลลัพธ์เป็นคำตอบที่ดีที่สุด ซึ่งแนวคิดนี้เอง ทำให้กระบวนการกลุ่มนี้มีประสิทธิภาพต่ำ เนื่องจากใช้เวลาในการแก้ปัญหานาน โดย Exact methods เหมาะกับการแก้ปัญหาขนาดเล็ก ที่มีลูกค้าจำนวนไม่มาก (75 – 100 คน) รวมถึงมีเงื่อนไข (Constraints) ที่ไม่ซับซ้อนมากนัก [1]

2.3.2 Heuristics

Heuristics สามารถค้นหาคำตอบที่ดี (ใกล้เคียง) กับคำตอบที่ดีที่สุดในปัญหาที่มีขนาดใหญ่ได้ ซึ่งคำตอบดังกล่าวจะเป็นคำตอบที่ *เหมาะสม* หรือ *ดีพอ* ได้ โดยที่ใช้เวลาคำนวณน้อยลงอยู่ในระดับที่ *ยอมรับได้* [13] ซึ่งสามารถแบ่งได้เป็นสองกลุ่มหลัก ได้แก่ Heuristics ที่ใช้ในการสร้างเส้นทาง (Route Construction heuristics) และอีกกลุ่มหนึ่งคือ Heuristics ที่ใช้ในการพัฒนาเส้นทางให้ดียิ่งขึ้น (Route-improving heuristics) [2]

2.3.3 Metaheuristics

Metaheuristics เป็นวิธีการประมาณค่าอีกรูปแบบหนึ่ง ซึ่งเหมาะกับการแก้ปัญหาที่มีขนาดใหญ่ กล่าวคือ เป็นการให้แนวทางที่ชี้นำการสำรวจค้นหาในแต่ละรอบของการวนซ้ำ (Iteration) เพื่อให้ได้คำตอบที่ดีที่สุดอย่างชาญฉลาดจากคำตอบทั้งหมดที่เป็นไปได้ โดยใช้เครื่องมือต่าง ๆ เช่น Heuristics เป็นต้น [2] ซึ่งสามารถทำให้กระบวนการค้นหาสามารถออกจากจุดที่มีค่าที่ดีที่สุดเฉพาะที่ (Local Optimum) ได้ดียิ่งขึ้น [14]

Metaheuristics สามารถแบ่งออกได้เป็นสองรูปแบบ ตามลักษณะของการแก้ปัญหาในแต่ละรอบของการวนซ้ำ ได้แก่ *Metaheuristics generating a sequence of solutions* [14] ที่มีลักษณะในการหาคำตอบที่ดีที่สุดเพียงคำตอบเดียวในการวนซ้ำแต่ละครั้ง โดยมีเป้าหมายในการพัฒนาคำตอบให้ดีขึ้นในแต่ละรอบของการวนซ้ำ อาทิเช่น Simulated Annealing (SA), Tabu Search, Variable Neighborhood Search (VNS), Large Neighborhood Search (LNS) เป็นต้น อีกกลุ่มหนึ่ง ได้แก่ *Metaheuristics generating set of solutions* [15] มีลักษณะการสร้างคำตอบใหม่โดยหาคำตอบที่ดีที่สุดมากกว่าหนึ่งคำตอบในการวนซ้ำแต่ละครั้ง ซึ่งเป็นอาจเป็นการรวมคำตอบปัจจุบันที่เลือกจากเซตของคำตอบ หรือประชากรที่ระบบเรียนรู้และเก็บไว้ ดังเช่น Generic Algorithm (GA) , Memetic Algorithm (MA) และ Scatter Search (SS) เป็นต้น ซึ่งโดยทั่วไปเป็นการเลียนแบบพฤติกรรมต่าง ๆ ที่เกิดขึ้นในธรรมชาติ ดังเช่น Swarm Optimization (PSO) และ Ant Colony Optimization (ACO) เป็นต้น

2.4 Large Neighborhood Search Metaheuristics

Large Neighborhood Search (LNS) เริ่มพัฒนาขึ้นโดย [16] ซึ่งเป็นวิธีการค้นหาที่เหมาะสมกับการใช้แก้ปัญหาการจัดเส้นทางขนาดใหญ่ ซึ่งจัดให้อยู่ในกลุ่ม Very Large Scale Neighborhood Search (VLSN) [17] ซึ่ง VLSN algorithms มีพื้นฐานจากการค้นหาจุดที่มีค่าที่ดีที่สุดเฉพาะที่ (Local Optima) ที่มีคุณภาพสูง ซึ่งจะเข้าใกล้ค่าที่ดีที่สุดอย่างแท้จริง (Global Optimum) [4] โดยการค้นหา Neighborhood ขนาดใหญ่จำเป็นต้องใช้เวลาในการคำนวณสูง ดังนั้น จึงมีการใช้เทคนิคต่าง ๆ ในการคัดกรอง Neighborhood บางส่วนที่ไม่จำเป็นออกจากกระบวนการพิจารณา และ สามารถทำให้คำตอบมีคุณภาพดีขึ้น ซึ่ง LNS จะกำหนด Neighborhood ผ่านกระบวนการกำจัด (Remove) และ สร้างเส้นทางใหม่ (Reinsert) บางส่วนของคำตอบที่ดีที่สุดในปัจจุบัน (Incumbent Solution) โดยใช้เครื่องมือต่าง ๆ เช่น Heuristics เป็นต้น

สำหรับแนวคิดพื้นฐานของ Large Neighborhood Search นั้น [14] การวนซ้ำแต่ละครั้งนั้น LNS จะจำแนกตัวแปรบางส่วนจาก Incumbent Solution เพื่อนำตัวแปรเหล่านี้ออกมาแก้หาคำตอบ ซึ่งเรียกตัวแปรกลุ่มนี้ว่าเป็นการกำหนด Neighborhood ในขณะเดียวกันนั้น กำหนดให้ค่าของตัวแปรที่เหลือเป็นค่าคงที่ที่ตรงตาม Incumbent Solution เรียกตัวแปรกลุ่มนี้ว่า Fixed Variables สำหรับกระบวนการที่กล่าวมานี้ คือ กระบวนการกำจัด (Remove) นั่นเอง จากนั้น ระบบจะหาคำตอบของ Neighborhood ซึ่งเรียกว่า กระบวนการหาเส้นทางใหม่ (Reinsert) โดย

สรุปนั้น LNS จะดึง Node บางส่วนที่คาดว่าเป็นทางเลือกที่ไม่ดีจาก Incumbent Solution ออกมาผ่านกระบวนการกำจัด (Remove) เพื่อหาทางแทรก Node เหล่านั้นลงไปในจุดที่เหมาะสมยิ่งขึ้นนั่นเอง [4]

สำหรับประเด็นสำคัญของ LNS ที่ผลต่อประสิทธิภาพของ LNS และคุณภาพของคำตอบ ได้แก่

1. คุณภาพคำตอบแรกที่เข้าสู่ LNS มีความสำคัญมาก ซึ่งอาจทำให้ LNS ใช้เวลานานเมื่อคำตอบเริ่มต้นมีคุณภาพไม่ดี [5]
2. จำนวนของตัวแปรที่กำหนดให้เป็นค่าคงที่มีความสำคัญต่อระยะเวลาในการหาคำตอบ เนื่องจาก Neighborhood จะมีขนาดใหญ่ขึ้นแปรผกผันตามจำนวนตัวแปรกลุ่มนี้ ซึ่งโดยทั่วไปจะสุ่มค่าในแต่ละรอบของการวนซ้ำ
3. หลักการในการเลือกกำจัดตัวแปร หรือ กำหนด Neighborhood ผ่านกระบวนการกำจัด ซึ่งมีลักษณะตั้งแต่การสุ่มไปจนถึงหลักการเลือกที่ซับซ้อน ซึ่งจะให้ผลที่แตกต่างกัน
4. กระบวนการในการแก้หาคำตอบ หรือ การสร้างเส้นทางใหม่ (Reinsert) ควรให้คำตอบที่มีคุณภาพดี แต่ใช้เวลาในการคำนวณสั้น

3. วิธีการแก้ปัญหา

3.1 แบบจำลองการแก้ปัญหาการรับส่งเอกสาร

สำหรับโครงข่ายของปัญหาการรับส่งเอกสาร ประกอบไปด้วย

1. N เป็นเซตของ Node ทั้งหมดภายในแบบจำลอง ประกอบด้วย
   - ลูกค้าที่รอรับบริการ n ราย จุดให้บริการรับเอกสาร คือ $P = \{1, 2, \dots, n\}$
   - จุดให้บริการส่งเอกสาร คือ $D = \{n+1, n+2, \dots, 2n\}$
   - จุดให้บริการ (Depot) $Dp = \{0, 2n+1\}$
2. E เป็นเซตของ Arc ที่ $E = \{(i,j): i,j \in N\}$
3. เซตของยานพาหนะ $K = \{1, 2, \dots, k\}$ ที่ใช้ในการให้บริการ ซึ่งยานพาหนะทุกคันมีความสามารถในการบรรทุกเท่ากัน คือ Q

สำหรับลูกค้า i แต่ละรายนั้น มีข้อมูลต่าง ๆ ที่สำคัญต่อการพิจารณาการแก้ปัญหา ดังต่อไปนี้

1. จุดรับสินค้า (x_i, y_i) และ จุดส่งสินค้า (x_{i+n}, y_{i+n})
2. ปริมาณบรรทุกของสินค้า (sd_i) ซึ่งต้องมีค่าไม่เกินไปกว่าความสามารถในการบรรทุก กล่าวคือ $sd_i \leq Q$ ซึ่งปัญหาการรับส่งเอกสารนั้นกำหนดให้มีค่า 1 หน่วย
3. กรอบเวลาที่กำหนดของลูกค้า (Time Window; $[e_i, l_i]$) ซึ่งประกอบไปด้วย เวลาที่เร็วที่สุดที่สามารถให้บริการได้ (Earliest Arrival time, e_i) และ เวลาที่ช้าที่สุดที่สามารถเริ่มให้บริการได้ (Latest Arrival time, l_i) ซึ่งมีคุณสมบัติ คือ $e_i \leq l_i$
4. ระยะเวลาที่ต้องใช้ในการบริการแก่ลูกค้า (Service time, st_i)

โดยสรุป ข้อมูลลูกค้าแต่รายสามารถเขียนเป็นเซตได้ $i = \{st_i, sd_i, e_i, l_i, x_i, y_i, x_{i+n}, y_{i+n}\}$ ซึ่งข้อมูลต่าง ๆ ที่กล่าวถึงมาข้างต้นจะนำไปเป็นส่วนหนึ่งของการพิจารณาแก้ปัญหาการจัดเส้นทางผ่านแบบจำลองทางคณิตศาสตร์ (Mathematical Formulation Model) ดังนี้

แบบจำลองการแก้ปัญหาการรับส่งเอกสารแบบมีกรอบเวลาที่ใช้ในงานวิจัยนี้ ประกอบไปด้วย ตัวแปรตัดสินใจ (Decision Variables) 2 ชนิด ซึ่งมีลักษณะ ดังนี้

- x_{ij}^k จะมีค่าเป็น 1 ก็ต่อเมื่อ ยานพาหนะคันที่ k เดินทางจาก Node i ไปยัง j กล่าวคือ เดินทางบน $Arc\ (i,j)$ นั่นเอง และ มีค่าเป็น 0 เมื่อไม่ตรงตามเงื่อนไขดังกล่าว
- y^k จะมีค่าเป็น 1 ก็ต่อเมื่อ มีการใช้ยานพาหนะคันที่ k ในการบริการ และ มีค่าเป็น 0 เมื่อไม่ตรงตามเงื่อนไขดังกล่าว

แบบจำลองทางคณิตศาสตร์ที่ใช้ในการพิจารณาหาเส้นทางที่ดีที่สุดสำหรับงานวิจัยนี้ สามารถเขียนเป็น Formulation ได้ดังนี้

$$\text{Minimize} \quad Z = \sum_{k\in K}\sum_{i\in N}\sum_{j\in N} y^k x_{i,j}^k c_{i,j} + \sum_{k\in K} y^k c^k \tag{1}$$

Subject to

$$\sum_{k\in K}\sum_{i\in N} x_{i,j}^k = \sum_{k\in K}\sum_{i\in N} x_{j,i}^k = 1 \qquad \forall j \in U \cup A \tag{2}$$

$$\sum_{k\in K}\sum_{i\in N} x_{i,j}^k = \sum_{k\in K} y^k \quad j = 2n+1 \tag{3}$$

$$\sum_{j\in N} x_{i,j}^k - \sum_{j\in N} x_{n+1,j}^k = 0 \qquad \forall j \in P, \forall k \in K \tag{4}$$

$$\sum_{k\in K}\sum_{j\in N} x_{i,j}^k = 0 \qquad i = 2n+1 \tag{5}$$

$$\sum_{k\in K}\sum_{i\in N} x_{i,j}^k = \sum_{k\in K} y^k \quad i = 0 \tag{6}$$

$$t_i^k + st_i + tt_{i,n+i} \leq t_{n+i}^k \qquad \forall j \in P, \forall k \in K \tag{7}$$

$$x_{ij}^k (t_i^k + st_i + tt_{i,j}) \leq t_j^k \qquad \forall i,j \in N, \forall k \in K \tag{8}$$

$$t_{2n+1}^k - t_0^k = T^k \qquad \forall k \in K \tag{9}$$

$$e_i \leq t_i^k \leq l_i \qquad \forall k \in K \tag{10}$$

$$y^k lc^k + \sum_{i\in N} sd_i \sum_{j\in N} y^k x_{i,j}^k \leq Q \qquad \forall k \in K \tag{11}$$

$$x_{i,j}^k \in \{0,1\},\ y^k \in \{0,1\} \qquad \forall i,j \in N, \forall k \in K \tag{12}$$

สำหรับสมการที่ 1 เป็นสมการวัตถุประสงค์ซึ่งมีเป้าหมายคือการหาต้นทุนที่ต่ำที่สุดสำหรับเส้นทางที่สามารถให้บริการลูกค้าได้อย่างครบถ้วน โดยมีเงื่อนไขตามสมการที่ 2 ถึง 14 ซึ่งอธิบายได้ดังนี้

- สมการที่ 2 บังคับให้ทุก Node ของลูกค้าได้รับการบริการเพียงครั้งเดียว และภายในเส้นทาง Node กลุ่มนี้จะมี Node ลำดับก่อนหน้าและ ต่อไปเสมอ
- สมการที่ 3 บังคับให้ยานพาหนะทุกคันต้องกลับไปยัง Depot เมื่อสิ้นสุดการให้บริการทั้งหมด
- สมการที่ 4 บังคับให้ Pickup และ Delivery Node ของลูกค้ารายเดียวกันได้รับการบริการจากยานพาหนะคันเดียวกัน
- สมการที่ 5 ระบุว่าหลังการกลับสู่ Depot แล้วจะเป็นการสิ้นสุดเส้นทางนั้น ๆ กล่าวคือ Depot Node จะไม่มี Node ลำดับต่อไป
- สมการที่ 6 บังคับให้ยานพาหนะทุกคันต้องเริ่มต้นที่จุดให้บริการ (Depot)
- สมการที่ 7 ระบุให้ยานพาหนะต้องรับสินค้า (Pickup) ก่อนการส่งสินค้า (Delivery)
- สมการที่ 8 ซึ่งระบุให้ต้องเสร็จการให้บริการที่ Node ก่อนหน้าก่อนถึงจะเริ่มให้บริการที่ Node ลำดับถัดไปได้

- สมการที่ 9 ระบุให้ยานพาหนะปฏิบัติการอยู่ภายในระยะเวลาสูงสุดสำหรับการบริการต่อวันของยานพาหนะ
- สมการที่ 10 เป็นกรอบเวลาของลูกค้าที่ยานพาหนะต้องปฏิบัติตาม
- สมการที่ 11 เป็นการอธิบายเงื่อนไขของการบรรทุก ซึ่งต้องไม่เกินความสามารถในการบรรทุก
- กลุ่มนิพจน์ 12 เป็นการกำหนดค่าตัวแปร และ เซตต่าง ๆ ที่เกี่ยวข้องในแบบจำลอง

3.2 กระบวนการแก้ปัญหา (Solving Algorithm)

สำหรับการหาคำตอบแรก (Initial Solution) ก่อนเข้าสู่กระบวนการ Large Neighborhood Search นั้น งานวิจัยนี้เลือกใช้ Greedy Constructive Heuristics เนื่องจากเป็น Heuristics ที่มีความรวดเร็วและใช้เวลาในการแก้ปัญหาสั้น

ภาพรวมของกระบวนการ Large Neighborhood Search (LNS) แสดงดังในรูปที่ 2 และ Algorithm สามารถเขียนโดยทั่วไปได้ดังใน Algorithm 1 โดยลักษณะของการแก้ปัญหาแบบ LNS นั้นเป็นการค้นหาเพื่อให้ได้คำตอบที่ดีขึ้นในแต่ละรอบของการวนซ้ำ ซึ่งประกอบด้วยกระบวนการกำจัด (Removing Method) ซึ่งเลือกแบ่งตัวแปรตัดสินใจออกเป็นสองกลุ่ม คือ Fixed Variables ที่จะบังคับให้มีค่าตามคำตอบปัจจุบัน และ Neighborhood ที่จะมีลักษณะเป็นตัวแปรที่สามารถเปลี่ยนแปลงค่าได้ กล่าวคือ เป็นปัญหาการจัดเส้นทางสำหรับ Node บางส่วนจากทั้งหมดในระบบ หลังจากนั้น Neighborhood จะได้รับการแทรกเข้าสู่คำตอบในตำแหน่งใหม่ ที่คาดว่าเป็นทางเลือกที่ดีขึ้น ผ่านกระบวนการหาเส้นทางใหม่ (Reinserting Method) และทำการตรวจสอบคุณภาพของคำตอบที่หาได้จากการวนซ้ำในรอบนี้ และกระบวนการ LNS จะสิ้นสุดลงก็ต่อเมื่อเป็นไปตามเงื่อนไขที่กำหนด และมีสัญลักษณ์เพิ่มเติมดังต่อไปนี้

- เซตของคำตอบปัจจุบัน (Current Solution) – $\boldsymbol{\pi}$
- เซตของ Node ที่จะได้รับการกำจัด (Node Set that will be removed) – $\boldsymbol{\varphi}'$ ซึ่งกำหนดให้เป็นร้อยละ 20 ของจำนวน Nodes ทั้งหมด
- เซตของ Node ที่ได้รับการกำจัดแล้ว (Removed Node Set) – $\boldsymbol{\varphi}$
- เซตของ Node ที่เหลือจากกระบวนการกำจัด (Residual Solution) – $\boldsymbol{\pi}' = \boldsymbol{\pi} - \boldsymbol{\varphi}$

3.2.1 กระบวนการกำจัด (Removing Method)

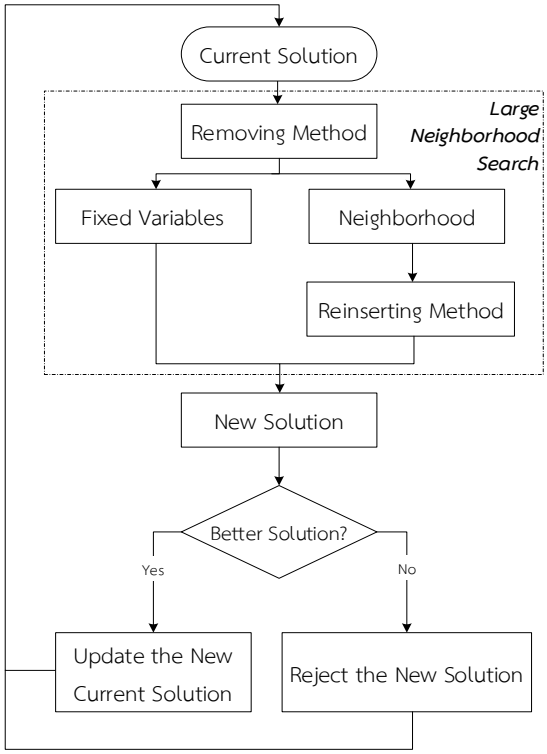
สำหรับกระบวนการนี้ เป็นการเลือกกำจัด Node หรือ ลูกค้าที่คาดว่าเป็นทางเลือกที่ไม่ดีในคำตอบปัจจุบันออกอย่างเป็นระบบ กระบวนการกำจัดมีหลายกระบวนการที่ใช้กันอย่างแพร่หลาย ซึ่งกระบวนการที่นำมาใช้ในงานวิจัยนี้ มีดังนี้

- *Shaw Removal Heurisitics [16]*

Shaw Removal Heuristic เป็นการกำจัด Node ออกโดยใช้ฟังก์ชันความเกี่ยวเนื่องกันของ Node (Relatedness Function) โดยเซตของ Node ที่จะได้รับการกำจัดออกจากคำตอบปัจจุบันโดยวิธี Shaw Removal Heuristic จะแทนด้วย $\boldsymbol{\varphi'}_R$

Algorithm 1 – Large Neighborhood Search (LNS) [18, 19]

		Description
1	**Input:** Current Solution $\boldsymbol{\pi}$	
2	$\boldsymbol{\pi}^b = \boldsymbol{\pi}$;	Current Global Best Solution
3	**repeat**	
4	$\quad\boldsymbol{\pi}^t$ = reinsert(remove($\boldsymbol{\pi}$));	การกำจัดและหาเส้นทางใหม่
5	$\quad$**if** accept($\boldsymbol{\pi}^t$, $\boldsymbol{\pi}$) **then**	กระบวนการตรวจสอบคุณภาพ
6	$\quad\quad\boldsymbol{\pi} = \boldsymbol{\pi}^t$	ของคำตอบภายในรอบการวนซ้ำ
7	$\quad$**end if**	
8	$\quad$**if** $c(\boldsymbol{\pi}^t) < c(\boldsymbol{\pi}^b)$ **then**	กระบวนการตรวจสอบคุณภาพ
9	$\quad\quad\boldsymbol{\pi}^b = \boldsymbol{\pi}^t$	ของคำตอบเทียบกับ *Current*
10	$\quad$**end if**	*Global Best Solution*
11	**until stop criterion is met**	
12	**Return** $\boldsymbol{\pi}$	



รูปที่ 2 ภาพรวมของกระบวนการแก้ปัญหาวิธี Large Neighborhood Search

$R(c, \varphi'_R)$ เป็นค่า Relatedness ระหว่าง Node $c \in \pi'$ และ เซตที่ถูกกำจัด (φ'_R) สำหรับ $R(c, \varphi'_R)$ สามารถคำนวณได้ดังสมการที่ 13

$$R(c, \varphi'_R) = \frac{\sum_{i \in \varphi'_R} r(c,i)}{|\varphi'_R|}, c \in \pi' \qquad (13)$$

โดยที่ $r(i,j)$ เป็นฟังก์ชันของความเกี่ยวเนื่องกันของ Node ซึ่งสามารถแบ่งออกได้เป็น 2 ลักษณะ คือ

- *ความเกี่ยวเนื่องเชิงระยะทาง (Relatedness based on distance)*
  ฟังก์ชันความเกี่ยวเนื่องเชิงระยะทางคำนวณได้ตามสมการที่ 14

$$r(i,j) = d_{ij} \quad (14)$$

  เมื่อ d_{ij} แทนระยะทางระหว่าง Node *i* ไปยัง Node *j*

- *ความเกี่ยวเนื่องเชิงเวลาเริ่มให้บริการ (Relatedness based on arrival times)*
  ฟังก์ชันความเกี่ยวเนื่องเชิงเวลาเริ่มให้บริการ เป็นไปตามสมการที่ 15

$$r(i,j) = \left|t_i^k - t_j^k\right| \quad (15)$$

เมื่อ t_i^k แทนเวลาที่รถคันที่ *k* ต้องไปถึงยัง Node *i* และ t_i^k แทนเวลาที่รถคันที่ *k* ต้องไปถึงยัง Node *j*

Shaw Removal Heuristic (SRH) สามารถสรุปได้ตาม Algorithm 2

Algorithm 2 – Shaw Removal Heuristic (SRH) [18, 19]

Input: Residual Solution π',
Expected size of a removed set γ
Let φ be a set of removed customers
Let **L** be a sorted set of customers and L_i be the i-th customer of **L;**
if φ = {} **execute**
    r ⟵ A customer selected randomly from π;
    φ ⟵ {r};
    π' ⟵ π \ {r};
end
while $|\varphi| < \gamma$ **execute**
    r ⟵ A customer selected randomly from φ;
    L ⟵ Customers of π;
    Sort L such that i < j ⇒ *r*(r, L_i) < *r*(r, L_j)
    Choose a random number *n* from [0,1];
    i ⟵ *floor*(*n*|L|);
    φ ⟵ φ ∪ {L_i};
    π' ⟵ π \ {L_i};
end
Return π' and φ;

3.2.2 กระบวนการหาเส้นทางใหม่ (Reinserting Method)

กระบวนการหาเส้นทางใหม่ มีความสำคัญต่อประสิทธิภาพของโปรแกรมโดยตรง เนื่องจากมีผลต่อประสิทธิภาพของการหาค่าที่เหมาะสมของแต่ละ Neighborhood ในแต่ละรอบของการวนซ้ำ (Iteration) และเช่นเดียวกันกับ Removing Method กระบวนการสร้างเส้นทางใหม่ก็มีหลายวิธีการที่ใช้กันอย่างแพร่หลาย ซึ่งงานวิจัยนี้เลือกใช้ Greedy Insertion Heuristics โดยพิจารณา Insertion Cost ($\Delta c_{i,k}$) เป็นต้นทุนที่เปลี่ยนแปลงไปเมื่อแทรก Node i ลงในเส้นทาง k ที่จุดทางเลือกที่ดีที่สุด โดยที่ i **∈** φ" โดยการแทรกที่เป็นไปไม่ได้นั้น Insertion Cost จะมีค่าเป็น ∞ สำหรับ Insertion Cost นั้นเป็นพารามิเตอร์ที่สำคัญสำหรับการพิจารณาแทรก Node ต่าง ๆ ที่ได้รับการกำจัดมากลับลงไปในคำตอบใหม่ โดยพิจารณาแทรก Node i ใด ๆ ลงในเส้นทาง k ใด ๆ ที่มี Insertion Cost ต่ำที่สุด ซึ่งมีค่าคือ $c(i,s) = min_{k \in R}(\Delta c_{i,k})$ เมื่อ R เป็นเซตของเส้นทางทั้งหมดภายในคำตอบปัจจุบัน (π) และ $c(i,s)$ เป็นต้นทุนของการแทรก Node i ณ จุดที่ดีที่สุดภายใต้เส้นทางที่ดีที่สุดในคำตอบปัจจุบัน

3.2.3 การยอมรับคำตอบใหม่ และ การสิ้นสุดกระบวนการค้นหาคำตอบ (Acceptance and Stopping Criterion)

สำหรับกรณีที่คำตอบที่ได้จากการหาเส้นทางในรอบการวนซ้ำปัจจุบันที่มีคุณภาพดีกว่าเดิมนั้น ระบบจะรับคำตอบนั้นเข้าเป็นคำตอบปัจจุบัน (π) ทันที แต่สำหรับคำตอบที่คุณภาพไม่ดีเท่ากับคำตอบปัจจุบันนั้นระบบจะปฏิเสธและยังคงใช้คำตอบที่ดีที่สุดจากการวนซ้ำรอบก่อนหน้า (π^b) ต่อไป

การสิ้นสุดของกระบวนการหาจัดเส้นทางด้วยวิธี LNS เกิดขึ้นได้ด้วยเงื่อนไขสองประการ ได้แก่ 1. จำนวนรอบของการวนซ้ำถึงจำนวนสูงสุดที่กำหนดไว้ ซึ่งกำหนดให้เป็น 30 รอบ และ 2. เวลาที่ใช้ในการประมวลผลถึงเวลาที่นานที่สุดที่กำหนดไว้ ซึ่งในกรณีนี้ไม่มีการกำหนด เนื่องจาก งานวิจัยนี้จะทำการทดสอบประสิทธิภาพด้าน*ระยะเวลารวมที่ใช้ในการคำนวณ (Computation Time)*

3.3 กระบวนการทดสอบและพัฒนาโปรแกรม

การพัฒนาโปรแกรมคอมพิวเตอร์เพื่อใช้ในการแก้ปัญหาการรับส่งเอกสารในงานวิจัยนี้ใช้โปรแกรม MATLAB ซึ่งใช้กันอย่างแพร่หลายและเหมาะแก่การแก้ปัญหา การทดสอบประสิทธิภาพของโปรแกรมที่พัฒนาขึ้นนั้นใช้ Li and Lim Benchmark [6] ซึ่งมีความเหมาะสมต่อการทดสอบโปรแกรมเพื่อแก้ปัญหาในงานวิจัยนี้ โดยมีการเผยแพร่ชุดข้อมูลไว้ที่ https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmark/

Li and Lim Benchmark เป็นชุดข้อมูลที่ใช้สำหรับการประเมินคุณภาพของกระบวนการแก้ ปัญหาการรับส่งสินค้าแบบมีกรอบเวลา (Pickup and Delivery Problem with Time Windows) ซึ่งมีลักษณะสอดคล้องกับปัญหาในงานวิจัยนี้ สำหรับชุดข้อมูลนั้นมีจำนวนลูกค้าตั้งแต่ 100, 200, 400, 600, 800 และ 1000 ราย ซึ่งในแต่ละกรณีนั้นมีชุดข้อมูลในช่วงระหว่าง 56 – 60 Instances ชุดข้อมูลทั้งหมดของ Li and Lim นั้นระบุผลลัพธ์ที่ดีที่สุดที่ค้นพบ (Best-known Results) ซึ่งทดสอบโดยมีเป้าหมายในการลดต้นทุน 2 ส่วนให้ต่ำที่สุด ได้แก่ 1. จำนวนรถที่ใช้ในระบบ (Minimum Number of Vehicles) และ 2. ระยะทางรวมในการให้บริการ (Minimum Total Distance)

กรอบของปัญหาภายในชุดข้อมูล Li and Lim Benchmark นั้นมีความคล้ายคลึงกับปัญหาในงานวิจัยนี้ ซึ่งมีความแตกต่างอยู่เพียงเล็กน้อย คือ กิจกรรมที่เกิดขึ้นบน Node (Visit) ซึ่งสรุปได้ตามตารางที่ 1 จึงสามารถสรุปได้ว่า Li and Lim Benchmark มีความเหมาะสมในการใช้ทดสอบประสิทธิภาพของการแก้ปัญหาการรับส่งเอกสารในงานวิจัยนี้

ตารางที่ 1 การเปรียบเทียบลักษณะของกรอบปัญหาในงานวิจัยนี้ และ Li and Lim Benchmark

ลักษณะของปัญหา	งานวิจัยนี้	Li and Lim Benchmark
จำนวนจุดให้บริการ (Depot)	จุดให้บริการเดียว	จุดให้บริการเดียว
ปัญหาแบบมีกรอบเวลา	มีกรอบเวลาการให้บริการ	มีกรอบเวลาการให้บริการ
ลักษณะของยานพาหนะ	Homogeneous	Homogeneous
ลักษณะโครงข่าย (*Structure*)	One-to-One	One-to-One
กิจกรรมบน Node (*Visit*)	P-D	P/D
Information Evolution	Static	Static
Information Quality	Deterministic	Deterministic

3.4 การประเมินประสิทธิภาพของโปรแกรม

- *ความแตกต่างของผลลัพธ์จากโปรแกรมและผลลัพธ์ที่ดีที่สุดของ Benchmark* ซึ่งเป็นการวัดประสิทธิภาพเชิงคุณภาพของผลลัพธ์ในด้านการลดต้นทุนให้ต่ำที่สุด ซึ่งเป็นวัตถุประสงค์หลักของการแก้ปัญหาการรับส่งสินค้า และในงานวิจัยนี้
- *ระยะเวลารวมที่ใช้ในการคำนวณ (Computation Time)* เป็นวัตถุประสงค์หนึ่งในงานวิจัยนี้ ที่ต้องการทำให้กระบวนการแก้ปัญหาต้องการความรวดเร็วมากยิ่งขึ้น

ตารางที่ 2 จำนวนและประเภท Instances ที่นำมาทดสอบ

No. of Nodes	Type	Tested Instances
100	Clustered	9
	Random	10
200	Clustered	10
	Random	10
400	Clustered	5
	Random	5
1000	Clustered	2
	Random	2

4. ผลการวิจัย

ในส่วนนี้ เป็นรายงานผลการคำนวณจากโปรแกรมคอมพิวเตอร์ที่พัฒนาโดยใช้ Large Neighborhood Search Metaheuristics ที่กล่าวถึงไว้ในหัวข้อที่ 3 ซึ่งสามารถจำแนกได้ตามตารางที่ 2

โดยผลการคำนวณที่แสดงในตารางที่ 3 ถึง 6 นั้น เป็นผลลัพธ์ที่ดีที่สุดที่พบในการศึกษานี้ (LNS Approach) เปรียบเทียบกับผลลัพธ์ที่ดีที่สุดที่ค้นพบ (Best-known Results) โดยงานวิจัยต่าง ๆ ที่ได้รับการเผยแพร่ไว้ที่ https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmark/ ซึ่งในแต่ละผลการคำนวณนั้นประกอบด้วยตัวแทนของต้นทุน 2 ส่วน ได้แก่ จำนวนยานพาหนะที่ใช้ (Vehicle) และ ระยะทางรวมทั้งหมดในระบบ (Distance)

4.1 100 Nodes

ตารางที่ 3 ผลการคำนวณ Li and Lim Benchmark ประเภท 100 Nodes

Instance	Best-known Results		LNS Approach	
	Vehicles	*Distance*	*Vehicles*	*Distance*
lc101	10	828.94	12	896.55
lc102	10	828.94	11	905.82
lc103	9	1,035.35	11	1,102.87
lc104	9	860.01	10	969.28
lc105	10	828.94	11	922.94
lc106	10	828.94	11	922.58
lc107	10	828.94	12	913.18
lc108	10	826.44	11	893.73
lc109	9	1,000.6	11	1,095.96
lr101	19	1,650.8	21	1,939.69
lr102	17	1,487.57	20	1,645.25
lr103	13	1,292.68	15	1,540.87
lr104	9	1,013.39	11	1,189.72
lr105	14	1,377.11	16	1,575.41
lr106	12	1,252.62	14	1,459.30
lr107	10	1,111.31	11	1,333.57
lr108	9	968.97	10	1,112.38
lr109	11	1,208.96	14	1,413.27
lr110	10	1,159.35	11	1,362.24

4.2 200 Nodes

ตารางที่ 4 ผลการคำนวณ Li and Lim Benchmark ประเภท 200 Nodes

Instance	Best-known Results		LNS Approach	
	Vehicles	*Distance*	*Vehicles*	*Distance*
lc1_2_1	20	2,704.57	23	2,988.55
lc1_2_2	19	2,764.56	21	3,166.80
lc1_2_3	17	3,127.78	19	3,677.64
lc1_2_4	17	2,693.41	19	2,983.49
lc1_2_5	20	2,702.05	23	3,011.70
lc1_2_6	20	2,701.04	22	3,054.34
lc1_2_7	20	2,701.04	24	3,069.19
lc1_2_8	19	3,354.27	21	3,866.80
lc1_2_9	18	2,724.24	21	3,065.31
lc1_2_10	17	2,942.13	19	3,282.83
lr1_2_1	20	4,819.12	22	5,547.77
lr1_2_2	17	4,621.21	20	5,513.10
lr1_2_3	14	4,402.38	16	5,073.74
lr1_2_4	10	3,027.06	12	3,587.07
lr1_2_5	16	4,760.18	18	5,691.27
lr1_2_6	13	4,800.94	15	5,677.11
lr1_2_7	12	3,543.36	14	4,079.12
lr1_2_8	9	2,759.32	12	3,257.10
lr1_2_9	13	5,050.75	15	5,974.03
lr1_2_10	11	3,664.08	12	4,271.95

4.3 400 Nodes

ตารางที่ 5 ผลการคำนวณ Li and Lim Benchmark ประเภท 400 Nodes

Instance	Best-known Results		LNS Approach	
	Vehicles	*Distance*	*Vehicles*	*Distance*
lc1_4_1	40	7,152.06	45	8,382.93
lc1_4_2	38	8,007.79	42	9,373.12
lc1_4_3	32	8,678.23	38	10,607.40
lc1_4_4	30	6,451.68	33	7,607.18
lc1_4_5	40	7,150.00	46	8,850.99
lr1_4_1	40	10,639.75	44	12,942.19
lr1_4_2	30	11,009.51	35	14,083.37
lr1_4_3	22	9,251.13	28	11,795.19
lr1_4_4	15	7,037.17	19	8,950.58
lr1_4_5	28	11,374.06	34	13,735.31

4.4 1000 Nodes

ตารางที่ 6 ผลการคำนวณ Li and Lim Benchmark ประเภท 1000 Nodes

Instance	Best-known Results		LNS Approach	
	Vehicles	*Distance*	*Vehicles*	*Distance*
lc1_10_1	100	42,488.66	108	53,365.76
lc1_10_2	94	44,548.51	110	56,059.84
lr1_10_1	100	56,744.91	112	74,920.30
lr1_10_2	80	49,349.84	91	64,292.97

5. บทสรุปและข้อเสนอแนะ

5.1 สรุปผลการประเมินประสิทธิภาพของโปรแกรม

ตารางที่ 7 สรุปประสิทธิภาพของโปรแกรม

No. of Nodes	Time (Seconds)	Problem Type	*(%) Δ Distance*	*(%) Δ Vehicles*
100	198.770	Clustered	15.633	15.062
		Random	15.933	17.371
200	513.146	Clustered	13.095	13.301
		Random	17.401	16.429
400	1,210.001	Clustered	19.638	13.355
		Random	25.002	20.407
1000	2,896.335	Clustered	25.720	12.511
		Random	31.155	12.875

จากตารางที่ 7 พบว่า ความแตกต่างของผลลัพธ์จากโปรแกรมและผลลัพธ์ที่ดีที่สุดของ Benchmark ในด้านระยะทางรวมของคำตอบ ((%) Δ Distance) มีเพิ่มขึ้นตามขนาดของปัญหา กล่าวคือ จำนวน Nodes ในโครงข่ายเพิ่มขึ้น และจากที่กล่าวมาในหัวข้อที่ 2 คุณภาพของคำตอบแรก (Initial Solution) ที่เข้าสู่ LNS มีความสำคัญมาก [5] ซึ่งงานวิจัยนี้ใช้ Greedy Heuristics ในการค้นหาคำตอบแรก ที่ได้รับคำตอบที่ค่อนข้างหยาบ หรือ คุณภาพไม่ดี หากแต่มีความรวดเร็วในการคำนวณ ดังแสดงในตารางที่ 8 ที่เวลาในการคำนวณในกระบวนการ Greedy Heuristics นั้นเป็นเพียงน้อยกว่า 1 % ของเวลาทั้งหมด แต่ให้ผลลัพธ์ที่คุณภาพแย่กว่าคำตอบสุดท้ายถึงมากกว่า 50 %

ตารางที่ 8 การเปรียบเทียบคำตอบแรกและผลลัพธ์จากการคำนวณ

Instance	Solution	Vehicles	Distance	Time
lc101	Greedy	24	1,830.32	2.278
	LNS	12	896.55	197.145
lc1_2_1	Greedy	50	6,977.24	5.716
	LNS	23	2,988.55	497.184
lc1_4_1	Greedy	102	19,586.77	11.072
	LNS	45	8,382.93	1,172.198
lc1_10_1	Greedy	257	122,584.57	54.266
	LNS	108	53,365.76	2,841.390

5.2 ข้อเสนอแนะ

เนื่องจากผู้วิจัยคาดว่างานวิจัยประเภทนี้มีประโยชน์และยังต้องได้รับการพัฒนาอีกมาก ผู้วิจัยมีข้อเสนอแนะเพื่อเป็นประโยชน์แก่ผู้ที่จะนำไปพัฒนาแบบจำลองและกระบวนการแก้ปัญหาให้มีประสิทธิภาพยิ่งขึ้น ดังนี้

5.2.1 กระบวนการหาคำตอบแรก (Initial Solution)

จากที่แสดงในหัวข้อที่ 5.1 คุณภาพของ Initial Solution อาจมีผลต่อผลลัพธ์สุดท้ายของกระบวนการ LNS ซึ่งควรได้รับการศึกษาในหัวข้อดังกล่าวมากขึ้น โดยการเปลี่ยนวิธีการจาก Greedy Heuristics เป็นวิธีอื่น อาทิเช่น Local Search, Simulated Annealing เป็นต้น เพื่อตรวจสอบว่าเมื่อคุณภาพของคำตอบแรกดีขึ้นจะมีผลดีต่อผลลัพธ์สุดท้ายหรือไม่ อย่างไร ซึ่ง Greedy Heuristics พิสูจน์ให้เห็นแล้วว่า ถึงแม้จะเป็นวิธีแก้ปัญหาที่รวดเร็ว แต่คุณภาพของคำตอบยังเป็นข้อเสียที่มีความสำคัญต่อผลลัพธ์สุดท้าย

5.2.2 กระบวนการ Large Neighborhood Search

— *กระบวนการกำจัด และ การหาเส้นทางใหม่ (Removing and Reinserting Heuristics)*

จากงานวิจัยต่าง ๆ [4] พบว่ายังมีกระบวนการกำจัดและการหาเส้นทางใหม่อื่น ๆ เช่น Worst Removal Heuristic, Regret-*k* Insertion Heuristics เป็นต้น ที่ได้รับการพัฒนา รวมถึงพิสูจน์ว่ามีประสิทธิภาพและให้คุณภาพของคำตอบที่ดี ดังนั้น การใช้กระบวนการกำจัดและการหาเส้นทางใหม่ที่หลากหลายในกระบวนการ LNS จึงอาจจะให้คำตอบที่มีคุณภาพดีขึ้น

— *จำนวนรอบในการวนซ้ำทั้งหมด (Number of Iterations)*

ในงานวิจัยนี้กำหนดให้มีจำนวนรอบของการวนซ้ำในกระบวนการ LNS ทั้งหมด 30 รอบ ซึ่งจำนวนรอบในการวนซ้ำมีผลโดยตรงต่อความรวดเร็วในการแก้ปัญหาในงานวิจัยนี้ ดังนั้น การหาความสัมพันธ์ระหว่างจำนวนรอบของการวนซ้ำกับประสิทธิภาพที่ดีขึ้น ทั้งในด้านการลดต้นทุนรวมของการให้บริการ และ ระยะเวลาในการแก้ปัญหา จะทำให้ทราบได้ว่า *จำนวนรอบในการวนซ้ำที่เหมาะสมที่สุด (Optimal Number of Iterations)* ซึ่งการเพิ่มจำนวนรอบของการวนซ้ำอาจไม่ทำให้คุณภาพของคำตอบดีขึ้นเทียบกับระยะเวลาในการแก้ปัญหาที่เพิ่มขึ้น

— *จำนวน Nodes ที่เหมาะสมในการกำจัดออกในขั้นตอนการกำจัด (Number of Removing Requests)*

ในงานวิจัยนี้มีกำหนดให้จำนวน Nodes ที่กำจัดออกในขั้นตอนการกำจัดเป็นร้อยละ 20 ของจำนวน Nodes ทั้งหมด โดยพารามิเตอร์นี้เป็นการกำหนดขนาดของปัญหาในขั้นตอนการสร้างเส้นทางใหม่ (Reinserting Method) ซึ่งมีผลโดยตรงต่อระยะเวลาในการแก้ปัญหาในแต่ละรอบของการวนซ้ำ ซึ่งหากกำจัด Nodes ออกจำนวนมากจะทำให้ขั้นตอนการสร้างเส้นทางใหม่มีความซับซ้อนและใช้เวลานาน เนื่องจาก ปัญหามีขนาดใหญ่ ในทางตรงกันข้าม การกำจัด Nodes ออกน้อยเกินไป อาจทำให้การพัฒนาคุณภาพของคำตอบในแต่ละรอบของการวนซ้ำไม่มีประสิทธิภาพเท่าที่ควร กล่าวคือ Neighborhood มีขนาดเล็กเกินไป อาจทำให้เกิด Local Optima ได้ ดังนั้น การหา*จำนวน Nodes ที่เหมาะสมในการกำจัดออกในขั้นตอนการกำจัด (Optimal Number of Removing Requests)* จึงมีความสำคัญต่อประสิทธิภาพในการแก้ปัญหาทั้งในด้านคุณภาพของคำตอบ และ ระยะเวลาในการแก้ปัญหาด้วยกระบวนการ Large Neighborhood Search

กิตติกรรมประกาศ

ขอขอบพระคุณ รองศาสตราจารย์ ดร. มาโนช โลหเตปานนท์ เป็นอย่างสูง ที่กรุณาให้คำแนะนำ ตรวจสอบ และแก้ไขข้อบกพร่องทุกขั้นตอนของงานวิจัยนี้ และให้ความช่วยเหลือตลอดการศึกษาทำให้งานวิจัยนี้สำเร็จลุล่วงไปได้ด้วยดี

ขอขอบพระคุณ คุณพ่อ คุณแม่ และทุก ๆ ท่านที่ไม่ได้กล่าวมาในที่นี้ ซึ่งช่วยเหลือและเป็นกำลังใจในการทำโครงงานนี้ตลอดมา

เอกสารอ้างอิง

[1] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, and A. A. Juan, "Rich vehicle routing problem: Survey," *ACM Computing Surveys (CSUR),* vol. 47, no. 2, p. 32, 2015.

[2] G. Berbeglia, J.-F. Cordeau, and G. Laporte, "Dynamic pickup and delivery problems," *European Journal of Operational Research,* vol. 202, no. 1, pp. 8-15, 2010/04/01/ 2010.

[3] A. Larsen, O. B. G. Madsen, and M. M. Solomon, "Recent Developments in Dynamic Vehicle Routing Systems," in *The Vehicle Routing Problem: Latest Advances and New Challenges*, B. Golden, S. Raghavan, and E. Wasil, Eds. Boston, MA: Springer US, 2008, pp. 199-218.

[4] D. Pisinger and S. Ropke, "Large neighborhood search," in *Handbook of metaheuristics*: Springer, 2010, pp. 399-419.

[5] L. Hong, "An improved LNS algorithm for real-time vehicle routing problem with time windows," *Computers & Operations Research,* vol. 39, no. 2, pp. 151-163, 2012/02/01/ 2012.

[6] H. Li and A. Lim, "A metaheuristic for the pickup and delivery problem with time windows," *International Journal on Artificial Intelligence Tools,* vol. 12, no. 02, pp. 173-186, 2003.

[7] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, "Static pickup and delivery problems: a classification scheme and survey," *TOP,* journal article vol. 15, no. 1, pp. 1-31, July 01 2007.

[8] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *European Journal of Operational Research,* vol. 225, no. 1, pp. 1-11, 2013.

[9] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & Industrial Engineering,* vol. 99, pp. 300-313, 2016/09/01/ 2016.

[10] N. A. El-Sherbeny, "Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods," *Journal of King Saud University - Science,* vol. 22, no. 3, pp. 123-131, 2010/07/01/ 2010.

[11] P. Kirci, "An optimization algorithm for a capacitated vehicle routing problem with time windows," *Sadhana,* Article vol. 41, no. 5, pp. 519-529, 2016.

[12] M. Drexl, "Applications of the vehicle routing problem with trailers and transshipments," *European Journal of Operational Research,* vol. 227, no. 2, pp. 275-283, 2013.

[13] E.-G. Talbi, *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

[14] N. Labadie, C. Prins, C. Prodhon, N. Labadie, C. Prins, and C. Prodhon, "Metaheuristics Generating a Sequence of Solutions," in *Metaheuristics for Vehicle Routing Problems*: John Wiley & Sons, Inc., 2016, pp. 39-75.

[15] N. Labadie, C. Prins, C. Prodhon, N. Labadie, C. Prins, and C. Prodhon, "Metaheuristics Based on a Set of Solutions," in *Metaheuristics for Vehicle Routing Problems*: John Wiley & Sons, Inc., 2016, pp. 77-107.

[16] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *International Conference on Principles and Practice of Constraint Programming*, 1998, pp. 417-431: Springer.

[17] R. K. Ahuja, Ö. Ergun, J. B. Orlin, and A. P. Punnen, "A survey of very large-scale neighborhood search techniques," *Discrete Applied Mathematics,* vol. 123, no. 1, pp. 75-102, 2002/11/15/ 2002.

[18] G. Mattos Ribeiro and G. Laporte, "An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem," *Computers & Operations Research,* vol. 39, no. 3, pp. 728-735, 2012/03/01/ 2012.

[19] S. Ropke and D. Pisinger, "A unified heuristic for a large class of Vehicle Routing Problems with Backhauls," *European Journal of Operational Research,* vol. 171, no. 3, pp. 750-775, 2006/06/16/ 2006.