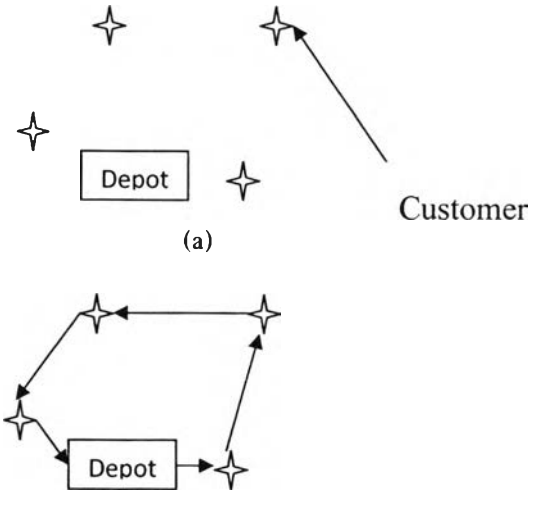
บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้องในอดีต

จากการศึกษางานวิจัยที่ผ่านมา พบว่ามีงานวิจัยจำนวนมากที่ทำการศึกษาปัญหาการจัดเส้นทางการเดินรถ (Vehicle Routing Problem) โดยทำการศึกษาในรูปแบบปัญหาและข้อจำกัดที่แตกต่างกันไป อีกทั้งวิธีในการแก้ปัญหาเพื่อหาผลเฉลยที่แตกต่างกันออกไป เนื่องมาจากปัญหาการจัดเส้นทางการเดินรถเป็นกลุ่มปัญหาที่รวบรวมปัญหาที่คล้ายคลึงกัน ซึ่งแต่ละปัญหามีลักษณะและรายละเอียดปลีกย่อยที่แตกต่างกันอยู่มาก ดังนั้นปัญหาการจัดเส้นทางการเดินรถจึงยังคงเป็นปัญหาที่ควรค่าแก่การศึกษาในปัจจุบัน

ในบทนี้จะกล่าวถึงปัญหาการเดินทางของพนักงานขาย รายละเอียดปัญหาการจัดเส้นทางการเดินรถ แบบจำลองแบบแบ่งเซ็ต การแก้ปัญหาจัดเส้นทางการเดินรถ งานวิจัยในอดีตเกี่ยวกับปัญหาการจัดเส้นทางการเดินรถ ทฤษฎีความน่าจะเป็น และ ทฤษฎีการหาเส้นทางที่สั้นที่สุด

2.1 ปัญหาการเดินทางของพนักงานขาย



รูปที่ 2.1 ปัญหาการเดินทางของพนักงานขาย

ปัญหาการเดินทางของพนักงานขาย (Travelling Salesman Problem: TSP) เป็นปัญหาที่ได้รับการวิจัยอย่างแพร่หลายตั้งแต่อดีตจนมาถึงปัจจุบัน โดยมีลักษณะของปัญหาคือ ต้องการหาเส้นทางการเดินทางที่มีค่าใช้จ่ายต่ำที่สุดในการเดินทางออกจากจุดเริ่มต้นหนึ่งจุดไปจุดปลายทางทุกๆ จุด จากนั้นกลับมายังจุดเริ่มต้น โดยจะต้องทราบค่าใช้จ่ายในการเดินทางระหว่างจุดสองจุดทั้งหมดเพื่อใช้ในการแก้ปัญหา รูปที่ 2.1 (a) แสดงตัวอย่างปัญหาการเดินทางของพนักงานขาย โดยมีจุดต้นทางคือศูนย์กระจายสินค้า (Depot) และต้องการเดินทางไปยังลูกค้า (Customer) ทั้งหมดด้วยค่าใช้จ่ายที่ต่ำที่สุด และรูปที่ 2.2 (b) แสดงตัวอย่างผลเฉลยของปัญหาการเดินทางของพนักงานขาย

แบบจำลองการแก้ปัญหาการเดินทางของพนักงานขาย [4] เป็นแบบจำลองที่ไม่มีความซับซ้อนมากนัก โดยแบบจำลองคณิตศาสตร์มาตรฐานของปัญหาการเดินทางของพนักงานขายมีดังนี้

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad 1 \le j \le n \qquad (2.1)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad 1 \le i \le n \qquad (2.2)$$

$$\sum_{i,j \in S_t} x_{ij} \le |S_t| - 1 \qquad (S_t \subset V; 2 \le |S_t| \le N-2) \qquad (2.3)$$

$$x_{ij} \in [0,1] \qquad \forall i, j \qquad (2.4)$$

โดยที่ n คือ จำนวนลูกค้าทั้งหมด

c_{ij} คือ ค่าใช้จ่ายในการเดินจากจุด i ไปยังจุด j

x_{ij} มีค่าเท่ากับ 1 เมื่อมีการเดินทางจากจุด i ไปยังจุด j และ มีค่าเท่ากับ 0 ในกรณีอื่นๆ

S_t คือ เซตของรอบการเดินทางที่ไม่สมบูรณ์

V คือ เซตของรอบการทำงานที่สมบูรณ์

สมการจุดประสงค์ (Objective Function) คือ ค่าใช้จ่ายรวมในการเดินทางต่ำที่สุด โดยมีเงื่อนไขคือจุดทุกจุดจะต้องได้รับการเข้าและออกหนึ่งครั้งตามสมการเงื่อนไขที่ 2.1 และ 2.2 ตามลำดับ และ เงื่อนไข 2.3 เพื่อป้องกันการเกิดรอบการเดินทางที่ไม่สมบูรณ์ เงื่อนไข 2.4 เพื่อเป็นการกำหนดตัวแปรให้เกิดได้ 2 กรณีคือเลือกกับไม่เลือก

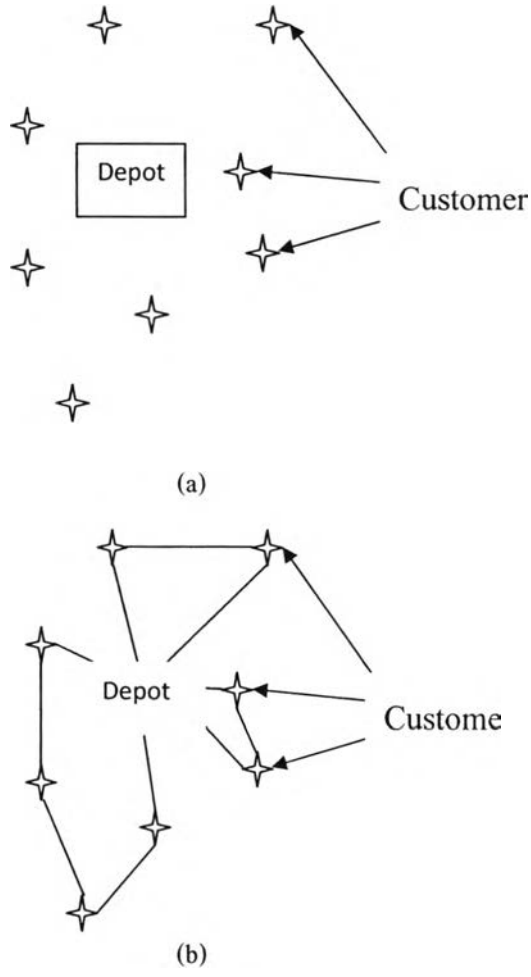
อย่างไรก็ตามเห็นได้ว่าปัญหาการเดินทางของพนักงานขายยังไม่ได้คำนึงถึงข้อจำกัดหลายประการที่เกิดขึ้นบ่อยครั้งในการปฏิบัติงานจริง เช่น ความจุของยานพาหนะ (Capacity) ปริมาณความต้องการของลูกค้า (Demand) และ จุดกระจายสินค้าอาจมีมากกว่าหนึ่งจุด เป็นต้น ในปัจจุบันจึงได้มีการศึกษาปัญหาการส่งสินค้าของพนักงานขายขนาดใหญ่ที่พัฒนาขึ้นและคำนึงถึงข้อจำกัดต่างๆ ที่กล่าวมาข้างต้น จนเป็นเหตุให้อาจมีความจำเป็นต้องใช้ยานพาหนะในการขนส่งมากกว่าหนึ่งคัน โดยปัญหาลักษณะนี้เรียกว่าปัญหาการจัดเส้นทางการเดินรถ ซึ่งจะกล่าวถึงรายละเอียดในหัวข้อถัดไป

2.2 ปัญหาการจัดเส้นทางการเดินรถ

ปัญหาการจัดเส้นทางการเดินรถ (Vehicle Routing Problem: VRP) คือ ปัญหาที่ประกอบไปด้วยเซตของเส้นทางการเดินรถ อาจจะมีจุดกระจายสินค้าหนึ่งจุด (Single Depot) หรือ จุดกระจายสินค้าหลายจุด (Multi Depot) และมีจุดประสงค์ของการแก้ปัญหาคือสามารถหาเส้นทางการเดินรถที่ขนส่งสินค้าไปยังลูกค้าทุกคนในปริมาณตามความต้องการของลูกค้า โดยมีค่าใช้จ่ายต่ำที่สุดเท่าที่จะสามารถเป็นไปได้โดยคำนึงถึงข้อจำกัดหลายประการที่เกิดขึ้นเนื่องจากสภาพการปฏิบัติงานจริง ยิ่งไปกว่านั้นจุดเริ่มต้นและจุดสิ้นสุดของเส้นทางการเดินรถจะต้องอยู่ที่จุดกระจายสินค้า

จากรูปที่ 2.2 (a) แสดงตัวอย่างข้อมูลนำเข้า (input) ของการแก้ปัญหาการจัดเส้นทางการเดินรถแบบจุดกระจายสินค้าหนึ่งจุด โดยจุดสี่เหลี่ยมตรงกลางคือจุดกระจายสินค้า และ จุดดาวคือตำแหน่งของลูกค้า ซึ่งผลลัพธ์จากการแก้ปัญหานี้อาจเป็นไปได้หลายรูปแบบ รูป 2.2 (b) คือรูปแบบหนึ่งของผลลัพธ์ที่สามารถเป็นไปได้ อย่างไรก็ดีในการแก้ปัญหาจริงนั้นจะต้องมีข้อมูลอื่นๆ มาประกอบในการแก้ปัญหาเพื่อที่จะทราบว่าผลลัพธ์ใดมีค่าใช้จ่ายในการเดินทางที่ต่ำที่สุด เช่น ความ

จุของยานพาหนะ เส้นทางที่สามารถเดินทางได้ และ ค่าใช้จ่ายในการเดินทางของแต่ละเส้นทาง เป็นต้น



รูปที่ 2.2 ปัญหาการจัดเส้นทางการเดินรถ

2.3 แบบจำลองการแบ่งเซต

ในงานวิจัยของ Hoffman และ Padberg กล่าวถึงแบบจำลอง Set Partitioning ซึ่งเป็นแบบจำลองที่มักถูกนำมาใช้ในการแก้ปัญหาการจัดเส้นทางการเดินรถ ปัญหาการจัดตารางเวลา (Scheduling problems) และปัญหาด้านตำแหน่งที่ตั้งที่เหมาะสม (Location problems) มีแบบจำลอง

ทั่วไปดังนี้

$$\text{Min} \sum_{j \in J} c_j x_j$$

Subject to:

$$\sum_{j \in J} a_{rj} x_j = 1 \qquad r \in R \qquad (2.5)$$

$$x_j \in \{1,0\} \qquad j \in J \qquad (2.6)$$

โดยที่ R คือ Set ของเงื่อนไขบังคับ (Constraints) ต่างๆ

J คือ Set ของสดมภ์ (Column) ต่างๆ

x_j คือตัวแปรในการตัดสินใจ ถ้าเส้นทาง j ถูกเลือกใช้จะมีค่าเท่ากับ 1 และมีค่าเท่ากับ 0 ในกรณีอื่นๆ

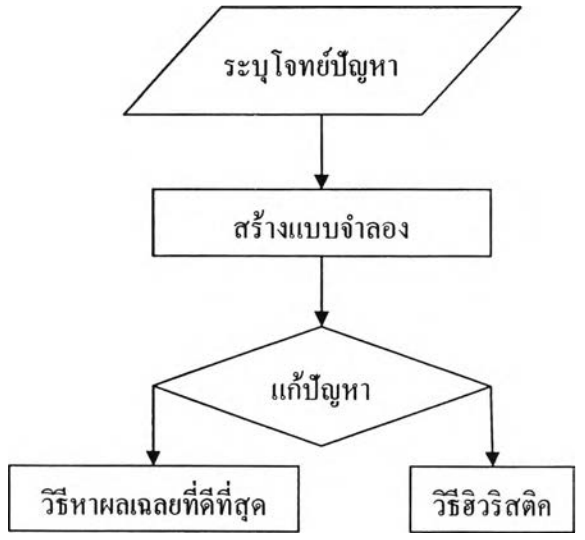
c_j คือ Cost ของ Column j

a_{rj} พารามิเตอร์ระบุเซ็ตของลูกค้าในเส้นทาง j มีค่าเท่ากับ 1 เมื่อลูกค้า r ได้รับการส่งสินค้าจากเส้นทาง j และมีค่าเท่ากับ 0

การแก้ปัญหาเพื่อหาผลเฉลยของแบบจำลองการแบ่งเซ็ตสามารถแก้ปัญหาได้ทั้ง 2 รูปแบบ คือวิธีการหาผลเฉลยที่ดีที่สุด และ วิธีฮิวริสติค โดยปรกติแบบจำลอง Set Partitioning จะมีขนาดใหญ่ และมีตัวแปรในการตัดสินใจจำนวนมาก การศึกษาปัญหาต่างๆ โดยการประยุกต์ใช้แบบจำลองประเภทนี้จึงมักสนใจในแง่ของการพัฒนาเทคนิคเพื่อลดตัวแปรตัดสินใจของแบบจำลองคณิตศาสตร์ลง หรือการปรับรูปแบบของแบบจำลองทางคณิตศาสตร์ให้เหมาะสมและมีจำนวน ตัวแปรตัดสินใจที่มีประสิทธิภาพโดยที่ผลของประยุกต์ใช้เทคนิคเหล่านี้ไม่ทำให้แบบจำลองมีค่าคำตอบที่ดีที่สุดด้อยไปกว่าเดิม ซึ่งเทคนิคการลดขนาดของแบบจำลองทางคณิตศาสตร์ต่างๆ นี้มีมากมาย เช่น การก่อกำเนิดสดมภ์ (Column Generation) เป็นต้น

2.4 การแก้ปัญหาการจัดเส้นทางการเดินรถ

จากรูปที่ 2.3 แสดงขั้นตอนการแก้ปัญหาการจัดเส้นทางการเดินรถ ซึ่งสามารถแบ่งเป็นขั้นตอนหลักๆ ได้ทั้งหมด 3 ขั้นตอนดังต่อไปนี้



รูปที่ 2.3 ขั้นตอนการแก้ปัญหาการจัดเส้นทางการเดินรถ

1. ระบุโจทย์ปัญหา คือ การศึกษาระบบการทำงานปัจจุบันและระบุว่าปัญหาที่แท้จริงคืออะไร รวมไปถึงระบุวัตถุประสงค์ในการแก้ปัญหา (Objective) ที่แน่ชัดอีกด้วย โดยส่วนมากวัตถุประสงค์ที่ต้องการในการแก้ปัญหาการจัดเส้นทางการเดินรถมีดังนี้

- เพื่อหาเส้นทางการเดินรถที่มีค่าใช้จ่ายในการขนส่งน้อยที่สุด (Minimize Cost) ซึ่งค่าใช้จ่ายในที่นี้อาจมีความหมายคลอบคลุมถึงสิ่งอื่นๆ นอกเหนือจากตัวเงิน เช่น เวลาการปฏิบัติงาน และ ความเชื่อมั่นของลูกค้าที่มีต่อบริษัท เป็นต้น
- เพื่อหาเส้นทางการเดินรถที่ใช้จำนวนยานพาหนะในการขนส่งน้อยที่สุด (Minimize Number of Truck)

2. สร้างแบบจำลอง คือ การสร้างแบบจำลองทางคณิตศาสตร์เพื่อใช้ในการแก้ปัญหา โดยแบบจำลองประกอบด้วย 2 ส่วนหลักๆ คือ วัตถุประสงค์ (Objective) และ ข้อจำกัด

(Constraint) ซึ่งข้อจำกัดในการแก้ปัญหานั้นสร้างเพื่อให้ผลเฉลยสอดคล้องกับการทำงานจริง เช่น กำหนดเวลาการขนส่งสินค้า ความจุของยานพาหนะที่ใช้ เป็นต้น

3. การแก้ปัญหาการจัดเส้นทางการเดินรถ (Vehicle Routing Problem: VRP) โดยการนำศาสตร์การวิจัยดำเนินงาน (Operations Research) มาประยุกต์ใช้ในการช่วยแก้ปัญหา

อัลกอริทึมในการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยศาสตร์การวิจัยดำเนินงานจำแนกออกเป็น 2 ประเภทหลักๆ คือ วิธีหาผลเฉลยที่ดีที่สุด (Exact) และ วิธีฮิวริสติค (Heuristic) ดังที่กล่าวมาแล้วข้างต้น ดังนั้นทฤษฎีที่เกี่ยวข้องในการจัดเส้นทางการเดินรถของงานวิจัยนี้แบ่งออกเป็น 2 ทฤษฎีหลักคือ ทฤษฎีการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีหาผลเฉลยที่ดีที่สุด และ ทฤษฎีการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีฮิวริสติค โดยรายละเอียดของทั้ง 2 ทฤษฎีจะกล่าวถึงในลำดับถัดไป

2.4.1 วิธีหาผลเฉลยที่ดีที่สุด (Exact algorithm)

วิธีหาผลเฉลยที่ดีที่สุด (Exact Method) เป็นแนวทางการแก้ปัญหาแบบจำลองคณิตศาสตร์ที่ได้ผลเฉลยที่มีคุณภาพสูงที่สุด อย่างไรก็ดีในเชิงปฏิบัตินั้นยังข้อจำกัดที่สำคัญคือปัญหาในความเป็นจริงนั้นซับซ้อนและมีปัจจัยซึ่งมีผลกระทบจำนวนมาก เป็นเหตุให้วิธีหาผลเฉลยที่ดีที่สุดนี้นั้นเหมาะกับปัญหาที่มีขนาดเล็กเท่านั้น โดยวิธีหาผลเฉลยที่ดีที่สุดที่ได้รับความนิยมนั้นมีด้วยกันหลายวิธี เช่น วิธีซิมเพล็กซ์, วิธีแตกกิ่ง และ วิธีกำเนิดสดมภ์ เป็นต้น

2.4.1.1 วิธีซิมเพล็กซ์

วิธีซิมเพล็กซ์ (Simplex Method) เป็นวิธีแก้ปัญหากำหนดการเชิงเส้น (Linear Programming Problem: LP) ด้วยหลักการทางพีชคณิตเพื่อหาผลเฉลยของปัญหา วิธีซิมเพล็กซ์จะเริ่มต้นด้วยปรับรูปแบบของปัญหาให้เข้าเป็นรูปแบบทาบูย์ (Tableau form) ซึ่งเป็นมาตรฐานของโปรแกรมเชิงเส้นที่สามารถแก้ปัญหาได้ จากนั้นทำการวนซ้ำ (Iteration) จนกระทั่งได้ค่าที่ดีที่สุด

รูปแบบทาบูย์ เป็นรูปแบบเมตริกซ์เงื่อนไขที่มีลักษณะเครื่องหมายบังคับให้เป็นเครื่องสมการเท่านั้น กล่าวคือเครื่องหมาย (Sense) ต้องเป็นเครื่องหมายเท่ากับเท่านั้นห้ามเป็นเครื่องหมายอื่น ดังนั้นหากสมการเงื่อนไขมีเครื่องหมายเป็นเครื่องหมายอื่นที่ไม่ใช่เท่ากับต้องเปลี่ยนให้อยู่ในเป็นเครื่องหมายเท่ากับก่อนจึงจะสามารถแก้ปัญหาได้ ยิ่งไปกว่านั้นต้องเป็นรูปแบบปัญหาการหาขีดจำกัดล่าง (Minimize Problem) หากเป็นปัญหาการหาขีดจำกัดบน (Maximize Problem) ก็จำเป็นต้องเปลี่ยนรูปแบบปัญหาให้เป็นปัญหาขีดจำกัดล่างตามรูปแบบมาตรฐานของ Tableau Form เช่นกัน

2.4.1.2 วิธีแตกกิ่ง

วิธีแตกกิ่ง (Branch & Bound) คือวิธีในการแก้ปัญหากำหนดการเชิงจำนวนเต็ม (Integer Programming: IP) โดยการแบ่งหรือแตกกิ่ง (Dividing or Branching) และ การหยุด (Conquering) โดยเริ่มต้นจากปัญหาที่มีขนาดใหญ่และแบ่งเป็นปัญหาย่อยๆ (Sub problem) จากนั้นพิจารณาขอบเขต (Bounding) ของคำตอบสำหรับปัญหาย่อยและพิจารณาตัดปัญหาที่ไม่สามารถให้คำตอบที่ดีที่สุดในขณะนั้นได้ และทำซ้ำกับทุกปัญหาย่อยๆจนกระทั่งพบปัญหาย่อยที่ให้ผลเฉลยที่ดีที่สุด ซึ่งวิธี Branch & Bound นั้นสามารถนำไปแก้ปัญหาได้หลายรูปแบบ เช่น Integer Programming (IP) Mixed Integer Programming(MIP) และ Binary Integer Programming (BP) เป็นต้น

2.4.1.3 วิธีกำเนิดสดมภ์

จากงานวิจัยของ Barnhart et al. (1998) สรุปไว้ว่าวิธีกำเนิดสดมภ์ (Column Generation) เป็นเทคนิคเพื่อหาคำตอบในปัญหาการหาผลเฉลยที่ดีที่สุดขนาดใหญ่ (Large-Scale Optimization Problem) โดยวิธีการนี้จะเริ่มต้นด้วยการสร้างเซตของแบบจำลองซึ่งเป็นสับเซตของแบบจำลองที่เป็นไปได้ทั้งหมด (Master Problem: MP) ของปัญหาเรียกว่า Restricted Master Problem (RMP) จากนั้นหาค่าคำตอบของ RMP เพื่อวิเคราะห์การเปลี่ยนแปลงของค่าคำตอบที่ดีที่สุด (Objective Function Value) เมื่อค่ากำหนดของสมการเงื่อนไข (Right hand side) เปลี่ยนแปลง 1 หน่วย (Shadow price) จากนั้นนำมาหาค่าใช้จ่ายที่ลดลงได้ (Reduced cost) ตามสมการความสัมพันธ์ดังนี้

$$\bar{c}_a = c_a - (y^*)^T a \qquad (2.7)$$

โดยที่ $\bar{c}_a$ Reduced cost ของตัวแปร

c_a ค่าสัมประสิทธิ์ของตัวแปร

y^* Matrix ของค่า Shadow price

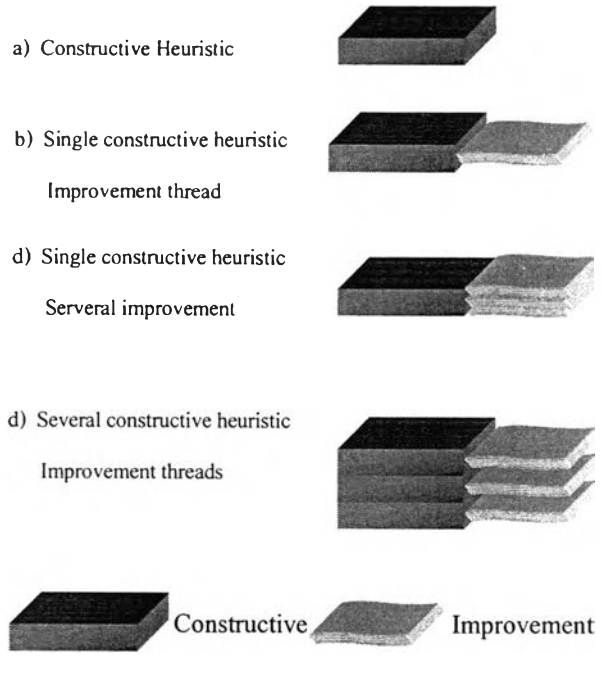
a Matrix ของ element ของตัวแปรใน Master Problem

หากมีตัวแปรใดที่สามารถให้ผลเฉลยที่ดีกว่าเดิมได้ (กล่าวคือมีค่าของค่าใช้จ่ายที่ลดลงได้เป็นลบ สำหรับกรณีของปัญหาค่าต่ำที่สุด) จะเพิ่มตัวแปรนั้นเข้าไปใน RMP และทำการหาค่าผลเฉลยใหม่อีกครั้ง จนกระทั่งไม่มีตัวแปรใดมีค่าของค่าใช้จ่ายที่ลดลงได้เป็นลบ กล่าวคือไม่มีตัวแปรใดที่สามารถให้ผลเฉลยที่ดีกว่าเดิมได้ จึงได้ผลเฉลยที่ดีที่สุด ยิ่งไปกว่านั้นวิธีในการเลือกตัวแปรมาเพิ่มใน RMP ในแต่ละรอบการทำงานนั้นส่งผลกระทบโดยตรงต่อความรวดเร็วในการแก้ปัญหา โดยจะเรียกปัญหาในการเลือกตัวแปรดังกล่าวว่าปัญหารอง (Sub Problem) วิธีการแก้ปัญหารองนั้นสามารถทำได้หลายรูปแบบขึ้นอยู่กับผู้ออกแบบอัลกอริทึม อย่างไรก็ตามวิธีกำเนิดสดมภ์อาจจะมีหรือไม่มีปัญหารองก็ได้ กล่าวคือหากไม่มีปัญหารองจะเลือกตัวแปรที่มีประสิทธิภาพโดยการเปรียบเทียบค่า Reduced Cost ของตัวแปรทั้งหมด จากที่กล่าวมาทั้งหมดเห็นได้ว่าวิธีกำเนิดสดมภ์สามารถช่วยลดขนาดของปัญหาในการหาผลเฉลยที่ดีที่สุดลงได้อย่างมาก ยิ่งไปกว่านั้นยังสามารถช่วยลดความยุ่งยากของการแก้ปัญหาได้ทั้ง IP และ LP อีกด้วย

จากการศึกษาการแก้ปัญหา LP ขนาดใหญ่ด้วยเทคนิคการกำเนิดสดมภ์ของ Wayne (2004) พบว่าเทคนิคการกำเนิดสดมภ์สามารถช่วยลดขนาดและความซับซ้อนของปัญหา LP ขนาดใหญ่ลงได้อย่างมาก เมื่อเทียบกับการแก้ปัญหาด้วยวิธี Simplex

2.4.2 วิธีฮิวริสติค

วิธีฮิวริสติค (Heuristic Method) เป็นแนวทางที่พยายามลดความซับซ้อนของปัญหาด้วยหลักการคิดของผู้ที่พัฒนาวิธีเพื่อประมาณหาค่าผลเฉลยที่มีคุณภาพในระดับที่สามารถยอมรับได้ ดังนั้นวิธีฮิวริสติคจึงมีรูปแบบในการแก้ปัญหาที่ค่อนข้างยืนหยุ่นอย่างมาก ส่งผลให้ในปัญหาหนึ่งๆ อาจมีวิธีในการแก้ปัญหาแบบฮิวริสติคที่แตกต่างกันมากมายหลายวิธี และ แม้ว่าวิธีฮิวริสติคจะได้ผลเฉลยที่ไม่ใช่ผลเฉลยที่ดีที่สุดแต่วิธีนี้ก็มีจุดเด่นอยู่ที่ความรวดเร็วในการคำนวณผลเฉลย



รูปที่ 2.4 ภาพกราฟิคแสดง Constructive และ Improvement

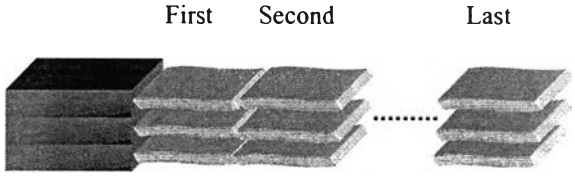
จากงานวิจัยของ Gendreau ทำการจำแนกวิธีการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีฮิวริสติคออกเป็น 4 ประเภทดังแสดงในรูปที่ 2.4 มีรายละเอียดดังนี้

Constructive Heuristics คือวิธีฮิวริสติคที่ได้ผลเฉลยจากการแก้ปัญหาเพียงรอบเดียว วิธีฮิวริสติคกลุ่มนี้มักจะใช้เวลาในการแก้ปัญหาที่น้อย แต่จะได้คุณภาพของผลเฉลยที่ดีเท่าไร เช่น วิธีประหยัด

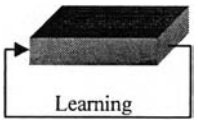
(Saving Method) จากงานวิจัยของ Clark และ Wright (1964) และ วิธีกวาด (Sweep Method) จากงานวิจัยของ Gillett และ Miller (1974) เป็นต้น

Improvement Heuristics คือวิธีฮิวริสติคนำผลเฉลยเบื้องต้นมาทำการพัฒนาคุณภาพของผลเฉลย โดยมากผลเฉลยเบื้องต้นจะได้มาจากวิธี Construction Heuristics ดังแสดงตัวอย่างกราฟฟิคเพื่อความเข้าในภาพที่ 2.4 ตัวอย่างวิธีฮิวริสติคในกลุ่มนี้จากการศึกษางานวิจัยในอดีต เช่น r-opt exchanges (1965) เป็นต้น

Population Mechanism คือวิธีฮิวริสติคที่ทำการวนรอบเพื่อหาผลเฉลยที่ดีขึ้นเรื่อยๆ โดยพัฒนาจากผลเฉลยเดิมในรอบก่อนหน้า แสดงตัวอย่างกราฟฟิคเพื่อความเข้าใจในภาพที่ 2.5 วิธีฮิวริสติคในกลุ่มนี้มักจะเป็นวิธีที่มีแนวคิดพื้นฐานแบบ Genetic Algorithm เช่น วิธีการแก้ปัญหาการจัดเส้นทางการเดินรถในงานวิจัยของ Reeves (2003) เป็นต้น



รูปที่ 2.5 ภาพกราฟฟิคแสดง Population Mechanism Heuristic



รูปที่ 2.6 ภาพกราฟฟิคแสดง Learning Mechanism Heuristic

Learning Mechanism คือวิธีฮิวริสติคที่มีลักษณะคล้ายคลึงกับ Population Mechanism ซึ่งจะทำการแก้ปัญหาเบื้องต้น จากนั้นทำการวนรอบแก้ปัญหาใหม่ทั้งหมดโดยเรียนรู้จากการแก้ปัญหาในรอบก่อนหน้าเพื่อที่จะหาผลเฉลยที่ดีขึ้นกว่าเดิมในทุกๆ รอบ และหยุดการวนรอบเมื่อได้ผลเฉลยที่ต้องการ แสดงตัวอย่างกราฟฟิคเพื่อความเข้าใจในภาพที่ 2.6 วิธีฮิวริสติคในกลุ่มนี้จากงานวิจัยในอดีตมีไม่มากนัก เช่น Ant Algorithm Heuristic ในงานวิจัยของ Reimann et al. (2004) เป็นต้น

2.4.2.1 วิธีประหยัด (Savings Method)

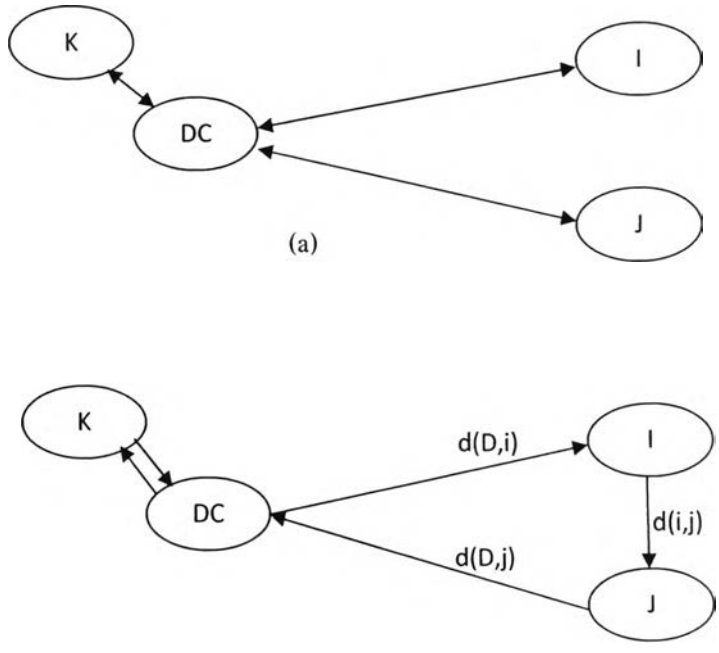
งานวิจัยของ Clarke และ Wright (1963) ได้พัฒนาวิธีการวิธีประหยัด (Savings Method) นี้ขึ้นเพื่อใช้ในการแก้ปัญหาการจัดเส้นทางการเดินรถที่มีขนาดของปัญหาใหญ่ คำตอบที่ได้มีค่าแตกต่างโดยเฉลี่ยจากคำตอบที่ดีที่สุดประมาณ 2 เปอร์เซ็นต์ จากรูปที่ 2.7(a) แสดงตัวอย่างการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีประหยัด เริ่มต้นจากการกำหนดให้รถวิ่งขนส่งระหว่างศูนย์กระจายสินค้า (Depot Center: DC) และ ลูกค้าทั้ง 3 ราย คือ i, j และ k แบบไปกลับจุดต่อจุด จากนั้นคำนวณค่าใช้จ่าย (Cost) ที่สามารถประหยัดเมื่อทำการรวมลูกค้าให้วิ่งไปด้วยรถคันเดียวกัน ดังสมการที่ 2.8 เพื่อความง่ายต่อการเข้าใจกำหนดให้ค่าใช้จ่ายค่าใช้จ่ายคือระยะทาง) เมื่อทำการคำนวณระยะทางที่ประหยัดได้สำหรับทุกคู่แล้วจึงเริ่มทำการรวมลูกค้าแต่ละคู่ให้อยู่ในเส้นทางเดินรถ (Route) เดียวกันโดยเริ่มรวมจากคู่ที่มาสามารถประหยัดได้มากที่สุดเรียงลำดับไปจนถึงน้อยที่สุด อย่างไรก็ดีต้องคำนึงถึงความจุของยานพาหนะอีกด้วย ได้ผลลัพธ์เส้นทางการเดินรถดังรูปที่ 2.7(b) เห็นได้ว่าผลลัพธ์ที่ได้ทำการรวมลูกค้าราย i และ j ไว้ในเส้นทางการเดินรถเดียวกัน ซึ่งนอกจากจะสามารถลดระยะทางในการเดินทางแล้วยังสามารถช่วยลดจำนวนยานพาหนะที่ใช้ในขนส่งอีกด้วย อย่างไรก็ดีต้องคำนึงถึงความจุของยานพาหนะว่าสามารถรองรับสินค้าตามความต้องการของลูกค้าทั้ง 2 ราย ได้หรือไม่อีกด้วย

$$S(i,j) = d(D,i) + d(D,j) - d(i,j) \qquad (2.8)$$

โดยที่ $S(i,j)$ คือ ระยะทางที่สามารถประหยัดได้ในการรวมลูกค้าคนที่ i และ j

$D(i,j)$ คือ ระยะทางจากจุด i ไปยังจุด j

D คือ จุดกระจายสินค้า (Distribution Center: DC)



รูปที่ 2.7 การลดระยะทางในการเดินทางโดยการรวมเส้นทาง

ขั้นตอนการทำ Saving Method นั้นเห็นได้ชัดว่าเป็นการแก้ปัญหาอย่างตะกละ (Greedy Algorithm) กล่าวคือ เลือกรวมลูกค้าที่ประหยัดได้มากที่สุดโดยไม่นึกผลกระทบที่เกิดขึ้นภายหลัง ซึ่งอาจจะให้ผลเฉลยที่ดีกว่า ยิ่งไปกว่านั้นความผิดพลาดดังกล่าวยังส่งผลกระทบมากขึ้นเมื่อปัญหามีขนาดใหญ่เนื่องจากปัญหาที่มีขนาดใหญ่จะมีทางเลือกที่ไม่ได้คำนึงถึงมากขึ้นตามไปด้วย

จากวิธีการทั้งหมดของ Saving Method นั้นจะได้เส้นทางการเดินรถ (Route) ก่อนที่จะได้กลุ่มของลูกค้าที่ได้รับส่งสินค้าด้วยยานพาหนะคันเดียวกัน (Cluster) ดังนั้นวิธีนี้ถือเป็นวิธีแก้ปัญหาแบบแบ่งเส้นทางการเดินรถก่อนการแบ่งกลุ่ม (Route First Cluster Second)

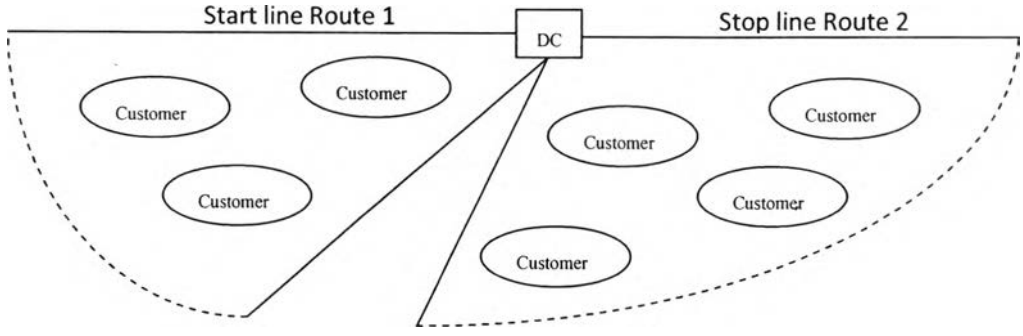
2.4.2.2 วิธีแทรก (Insertion Heuristic)

จากงานวิจัยของ Dantzig และ Ramser (1959) นำเสนอวิธีแทรก (Insertion Heuristic) ซึ่งเป็นวิธีในการแก้ปัญหาที่มีอัลกอริทึมที่ง่ายและตรงไปตรงมา อัลกอริทึมมีความคล้ายคลึงกับ

Saving Method โดยอัลกอริทึมนี้คือการพยายามแทรกลูกค้าใหม่เข้าไประหว่างลูกค้าในเส้นทางการเดินทาง ซึ่งการแทรกนี้จะทำการเลือกแทรกลูกค้าที่มีค่าใช้จ่ายที่เพิ่มขึ้นในการแทรกน้อยที่สุดก่อน อย่างไรก็ดีการแทรกต้องคำนึงถึงข้อจำกัดหลักอื่นให้สามารถเป็นไปได้ด้วย เช่น ความจุยานพาหนะ และ กำหนดการขนส่งสินค้า เป็นต้น เมื่อเส้นทางการเดินรถในปัจจุบันไม่สามารถเพิ่มลูกค้าในเส้นทางการเดินรถได้อีก อัลกอริทึมจะทำการสร้างเส้นทางการเดินรถใหม่และกระทำเช่นเดิมจนกระทั่งส่งสินค้าไปยังลูกค้าครบทุกราย

จากวิธีการแก้ปัญหาด้วยวิธีแทรกทั้งหมดข้างต้นเห็นได้ว่าวิธีแทรกเป็นรูปแบบการแก้ปัญหาอย่างตะกละ และได้เส้นทางการเดินรถก่อนที่จะได้กลุ่มของลูกค้าที่ได้รับส่งสินค้าด้วยยานพาหนะคันเดียวกันเช่นเดียวกับวิธีประหยัด

2.4.2.3 วิธีกวาด (Sweep Method)



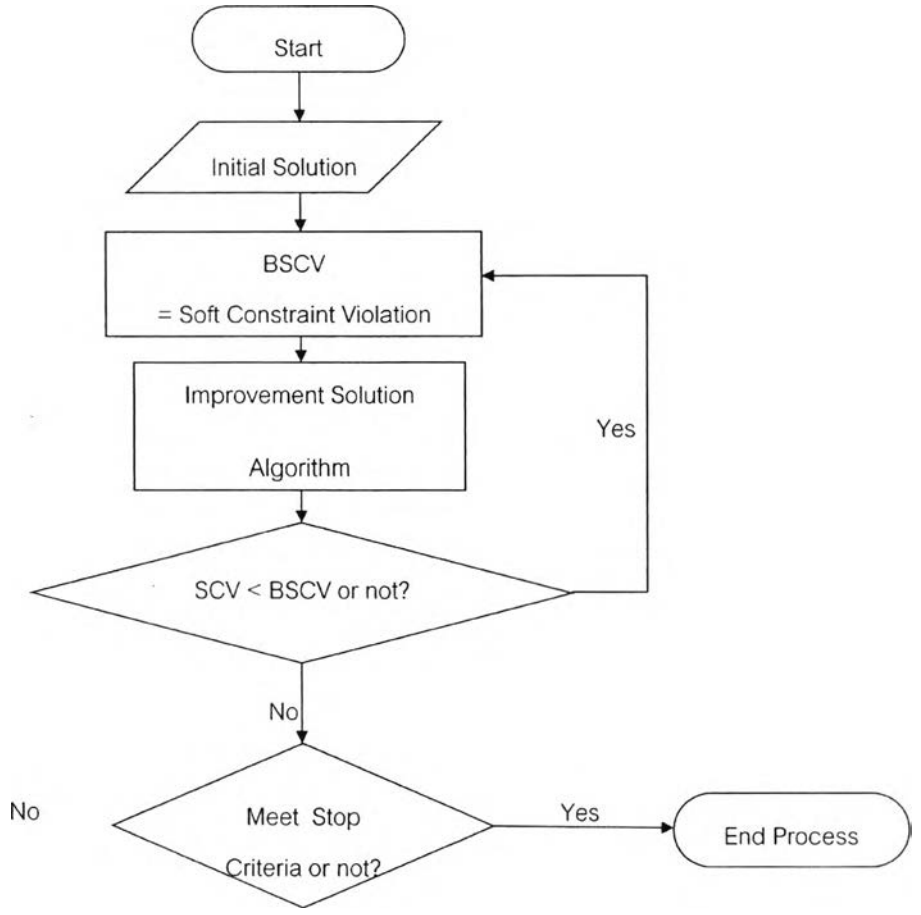
รูปที่ 2.8 จัดเส้นทางการเดินรถด้วยวิธีกวาด

วิธีกวาด (Sweep Method) เป็นวิธีการที่สะดวกและใช้เวลาในการแก้ปัญหาน้อย เหมาะสำหรับการแก้ปัญหาขนาดใหญ่ คำตอบที่ได้มีค่าแตกต่างกับคำตอบที่ดีที่สุดโดยเฉลี่ยประมาณ 10 เปอร์เซ็นต์ วิธีการนี้เริ่มต้นจากการกำหนดพื้นที่ความรับผิดชอบของศูนย์กระจายสินค้าแต่ละแห่ง แล้วกำหนดเส้นตรงออกจากศูนย์กระจายสินค้าให้หมุนไปในทิศตามเข็มหรือทวนเข็มนาฬิกา เมื่อเส้นตรงนี้ตัดกับลูกค้ารายก็ให้รวมเข้าอยู่ในเส้นทางเดินรถ หมุนเส้นตรงนี้ต่อไปเรื่อยๆ จนกว่าจะไม่สามารถรวมลูกค้ารายใหม่เข้ามาในเส้นทางเดินรถได้ เนื่องจากข้อจำกัดต่างๆ เช่น ปริมาณสินค้าเกินความจุของรถ เป็นต้น เริ่มต้นจัดสายรถใหม่จะลูกค้าที่เหลืออยู่จนกระทั่งลูกค้าทุกรายมีสายรถ

ครบดังรูปที่ 2.8 จากนั้นเพื่อให้ได้เส้นทางที่สั้นที่สุดควรใช้เทคนิคต่างๆในการเดินรถภายใน เช่น หลักการปัญหาการเดินทางของพนักงานขาย (Traveling Salesman Problem: TSP) มาประยุกต์ใช้เพื่อช่วยเพิ่มประสิทธิภาพของผลเฉลย แต่อย่างไรก็ดีข้อเสียของวิธีนี้ คือ การควบคุมปัจจัยภายนอกอื่นๆ เช่น เวลาในการเดินทางทั้งหมด และ ช่วงเวลากำหนดการส่งสินค้า (Time Windows) นั้นทำได้ยาก ดังนั้น วิธีกวาดจึงไม่เหมาะสมที่จะถูกเลือกนำมาใช้ในงานวิจัยนี้เนื่องจากงานวิจัยนี้เป็นการจัดเส้นทางการเดินรถที่คำนึงถึงช่วงกำหนดการส่งสินค้า

จากวิธีการแก้ปัญหาด้วยวิธีกวาดทั้งหมดข้างต้นเห็นได้ว่าเป็นวิธีการแก้ปัญหาแบบที่จะได้กลุ่มของลูกค้าที่ได้รับการขนส่งสินค้าด้วยยานพาหนะคันเดียวกันก่อนที่จุดได้เส้นทางการเดินรถ (Cluster First Route Second) ซึ่งแตกต่างจากวิธีประหยัด และ วิธีแทรก อย่างไรก็ตามวิธีกวาดนั้นยังคงเป็นวิธีการแก้ปัญหาอย่างตะกละ

2.4.2.4 วิธีการค้นหาเฉพาะแห่ง (Neighbourhood Search)



รูปที่ 2.9 ขั้นตอนการแก้ปัญหาด้วยวิธีการค้นหาเฉพาะแห่ง

Shaw (1998) ทำการศึกษาวิธีการค้นหาเฉพาะแห่ง (Neighbourhood Search: NS) หรืออีกชื่อหนึ่งที่เป็นที่รู้จักคือ local Search คือวิธีฮิวริสติคอีกวิธีหนึ่งในการแก้ปัญหาของศาสตร์การวิจัยดำเนินงานที่มีการวิจัยซึ่งได้รับความนิยมอย่างแพร่หลายในปัจจุบัน เนื่องมาจากมีอัลกอริทึมในการแก้ปัญหาที่ยืดหยุ่นและเข้าใจง่าย อัลกอริทึมจะเริ่มต้นด้วยเส้นทางการเดินรถเบื้องต้น (Initial Solution) จากนั้นทำการวนรอบเพื่อหาเส้นทางการเดินรถที่ดีขึ้นจนกระทั่งได้เส้นทางการเดินรถที่ต้องการจึงหยุดการวนรอบ อีกทั้งอัลกอริทึมจะมีการคำนึงถึงข้อจำกัดต่างๆ อีกด้วย

จากรูปที่ 2.9 แสดงขั้นตอนการแก้ปัญหาด้วยวิธีการค้นหาเฉพาะแห่ง โดยอัลกอริทึมเริ่มต้นด้วยการกำหนดข้อจำกัดของปัญหาโดยแบ่งข้อจำกัด (Constraint) ออกเป็น 2 กลุ่มดังนี้

1. ข้อจำกัดหลัก (Hard Constraint) คือ ข้อจำกัดที่ห้ามฝ่าฝืนโดนเด็ดขาด เช่น ความจุของยานพาหนะ และ ช่วงเวลาห้ามวิ่ง เป็นต้น
2. ข้อจำกัดรอง (Soft Constraint) คือ ข้อกัดที่สามารถฝ่าฝืนได้แต่ทางเลือกที่ดีที่สุดของผลเฉลยจะเลือกทางเลือกที่ฝ่าฝืนข้อจำกัดรองน้อยที่สุด เช่น ระยะทางรวมในการขนส่งสินค้า และ จำนวนยานพาหนะที่ใช้ในการขนส่ง เป็นต้น

จากนั้น สร้างผลเฉลยเบื้องต้น (Initial Solution) โดยผลเฉลยเบื้องต้นนั้นควรได้มาด้วยวิธีการที่ง่าย รวดเร็ว และ ไม่ขัดต่อข้อจำกัดหลัก ขั้นตอนต่อไปคือการวนรอบซึ่งจะเริ่มต้นด้วยการเก็บค่าการฝ่าฝืนข้อจำกัดรอง (Soft Constraint Violation) ของผลเฉลยเบื้องต้นเป็นค่าการฝ่าฝืนของปัญหา (Best Soft Constraint Violation: BSCV) จากนั้นทำการพัฒนาเส้นทางการเดินรถ (Improvement Solution Algorithm) โดยมีข้อห้ามไม่ให้ค้นหาซ้ำกับเซตของคำตอบที่มีอยู่แล้ว หรือที่เรียกว่ารายการต้องห้าม (Neighbourhood Search List) ซึ่งวิธีการห้ามดังกล่าวจะเป็นการห้ามเพื่อที่จะช่วยให้ไม่ต้องไปหาผลเฉลยเดิม หรือหลงในวัฏจักร (cyclic) ซึ่งจะส่งผลให้ไม่สามารถหาคำตอบที่ดีขึ้นได้ จากนั้นทำการหาค่าการฝ่าฝืนข้อจำกัดรองของเส้นทางการเดินรถใหม่ หากมีค่าการฝ่าฝืนข้อจำกัดรองที่น้อยกว่าค่าการฝ่าฝืนของปัญหาและไม่ฝ่าฝืนข้อจำกัดหลัก ก็จะถูกจัดเก็บค่าการฝ่าฝืนนั้นเป็นค่าการฝ่าฝืนของปัญหาและเก็บผลเฉลยเป็นผลเฉลยของปัญหา แต่หากมีค่าการฝ่าฝืนค่าจำกัดรองมากกว่าค่าการฝ่าฝืนของปัญหาจะไม่ถูกจัดเก็บ จากนั้นทำการวนรอบกระทำซ้ำแบบเดิมไปเรื่อยจนผลเฉลยมีคุณภาพระดับที่ต้องการจึงหยุดการวนรอบ อย่างไรก็ดีตัวชี้วัดในการหยุดการวนรอบของ Neighbourhood Search มีหลากหลายดังจะกล่าวถึงในลำดับถัดไป

งานวิจัยของเกรียงศักดิ์และณกร (2007) วิธีในการสลับเปลี่ยนแปลงเส้นทางการเดินรถเพื่อหาเส้นทางการเดินรถที่ดีขึ้นในแต่ละรอบ (Improvement Solution Algorithm) มีอยู่ด้วยกันมากมายหลายวิธีเนื่องจากเป็นวิธีฮิวริสติค ดังนั้นวิธีการขึ้นอยู่กับหลักการคิดของผู้พัฒนาและลักษณะของปัญหา สามารถแบ่งได้เป็น 2 กลุ่มตามระดับของการแก้ปัญหาคือ สลับเปลี่ยนภายในยานพาหนะ (Internal Trial Swap) และ สลับเปลี่ยนภายนอกยานพาหนะ (External Trial Swap)

	Truck 1					
	1	2	3	4	5	6
Customer 1	1					
Customer 2			1			
Customer 3					1	
Customer 4		1				
Customer 5				1		
Customer 6						1

(a)

	Truck 1					
	1	2	3	4	5	6
Customer 1	1					
Customer 2			1			
Customer 3					1	
Customer 4		1				
Customer 5				1		
Customer 6						1

(b)

รูปที่ 2.10 ตัวอย่างการสลับเปลี่ยนภายในยานพาหนะ

	Truck 1					
	1	2	3	4	5	6
Customer 1	1					
Customer 2		1				
Customer 3					1	
Customer 4			1			
Customer 5				1		
Customer 6						1

(c)

รูปที่ 2.10 ตัวอย่างการสลับเปลี่ยนภายในยานพาหนะ

รูปที่ 2.10 แสดงตัวอย่างการทำสลับเปลี่ยนลำดับภายในยานพาหนะคันเดียวกัน (Internal Trial Swap) โดยการสลับภายในยานพาหนะ คือ การสลับลำดับการขนส่งสินค้าภายในยานพาหนะคันเดียวกัน จากรูปที่ 2.10 (a) เมื่อทำการสุ่มเลือกคอลัมน์ที่ 3 คือลำดับที่ 3 ของยานพาหนะคันที่ 1 จากนั้นทำการเลือกลำดับก่อนหน้ามาเพื่อเตรียมสลับลำดับ ดังรูปที่ 2.10 (b) จากนั้น ในรูปที่ 2.10 (c) หลังจากทำการสลับแล้วจะเห็นได้ว่าลำดับการขนส่งของยานพาหนะคันที่ 1 จาก 1,4,2,5,3,6 จะเปลี่ยนเป็น 1,2,4,5,3,6 อย่างไรก็ตามการสลับเปลี่ยนภายในยานพาหนะนั้นเป็นประเด็นที่ไม่มีงานวิจัยที่เกี่ยวข้องมากนัก เนื่องจากเป็นปัญหาขนาดเล็ก มีลูกค้าในยานพาหนะเพียงไม่กี่ราย ดังนั้นการสลับเปลี่ยนภายในจึงไม่ได้รับประโยชน์จากการใช้วิธีค้นหาเฉพาะแห่งซึ่งเป็นวิธีฮิวริสติคมากเท่าใด เนื่องสามารถใช้วิธีหาผลเฉลยที่ดีที่สุดในการแก้ปัญหาโดยใช้เวลาในการแก้ปัญหาที่ไม่แตกต่างกันมากนัก

การสลับเปลี่ยนภายนอกยานพาหนะคือ การสลับเปลี่ยนลูกค้าที่จะต้องไปส่งระหว่างยานพาหนะคนละคัน ซึ่งโดยส่วนใหญ่จะกระทำเมื่อ ไม่สามารถทำการสลับเปลี่ยนภายในยานพาหนะแล้วไม่สามารถหาผลเฉลยที่ดีกว่าปัจจุบันได้อีกแล้ว โดยมีรูปแบบการสลับเช่นเดียวกันกับ การสลับเปลี่ยนภายในยานพาหนะ

จากงานวิจัยของ Ahaja และ Orline (2000) สรุปไว้ว่าเทคนิคในการสลับเปลี่ยนเส้นทางการเดินรถนั้นมีอยู่ด้วยกันหลากหลาย และ สามารถจำแนกออกเป็น 3 ประเภทหลักๆ ดังนี้

1. ประเภททำลายเส้นทางระหว่างสองจุดใดๆ บนเส้นทางการเดินรถและนำมาสร้างใหม่ในลักาณที่แตกต่าง (Adding and Delete Arcs)
2. ประเภทสลับเปลี่ยนลำดับและลูกค้าระหว่างยานพาหนะ (Compounded swap)
3. ประเภทสลับเปลี่ยนลูกค้าแบบหมุนวน (Cyclical shift)

ตัวชี้วัดในการหยุดการวนรอบของวิธี Neighbourhood Search [7] มีอยู่ด้วยกันหลายตัวชี้วัดขึ้นอยู่กับความต้องการของผู้สร้างอัลกอริทึม อย่างไรก็ดีสามารถสรุปตัวชี้วัดที่นิยมใช้ทั่วไปได้ 3 ตัว คือ ระยะเวลาในการคำนวณ จำนวนรอบ และ จำนวนรอบที่ไม่สามารถหาผลเฉลยที่ดีกว่าเดิมได้ ซึ่งตัวชี้วัดที่ได้รับความนิยมมากที่สุดคือ จำนวนรอบที่ไม่สามารถหาผลเฉลยที่ดีกว่าเดิมได้ เช่น หากกำหนดจำนวนรอบเป็น 3 หมายความว่าหากไม่สามารถหาผลเฉลยที่ดีกว่าเดิมได้ภายใน 3 รอบการทำงาน โปรแกรมจะหยุดการคำนวณและถือว่าผลเฉลยดังกล่าวเป็นผลเฉลยของปัญหา

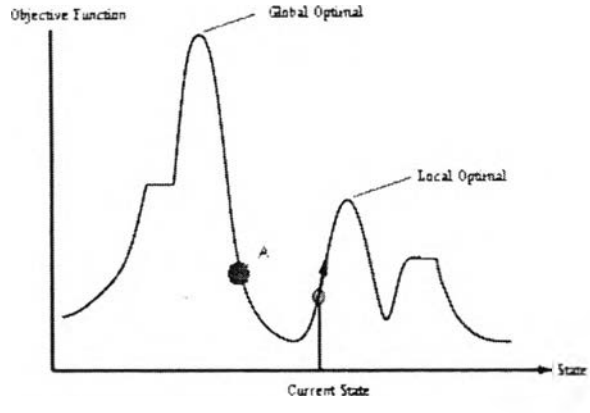
2.4.2.4 วิธีการค้นหาเฉพาะแห่งขนาดใหญ่ (Large Scale Neighbourhood Search)

ในการแก้ปัญหาขนาดใหญ่ด้วยวิธีการค้นหาเฉพาะแห่งนั้นจะประสบปัญหาการไม่สามารถผลเฉลยที่ดีที่สุด (Global Optimal) เนื่องมาจากการกำหนดผลเฉลยเบื้องต้นที่ตายตัวทำให้การค้นหาผลเฉลยนั้นทำได้ในขอบเขตที่แคบ ในเชิงวิชาการนั้นปัญหาดังกล่าวถูกเรียกว่าปัญหาดีที่สุดเฉพาะที่ (Local Optima) จึงจำเป็นที่จะต้องพัฒนาวิธีการในการแก้ปัญหาเพื่อใช้ในการแก้ปัญหาขนาดใหญ่ ซึ่งวิธีการที่พัฒนาขึ้นดังกล่าว คือ วิธีการค้นหาเฉพาะแห่งขนาดใหญ่ (Large Scale Neighbourhood Search: LNS) ดังจะกล่าวถึงรายละเอียดในลำดับถัดไป

จากงานวิจัยของ Ahuja และ Orlin (2000) สรุปไว้ว่าวิธีการค้นหาเฉพาะแห่งขนาดใหญ่ (Large Scale Neighbourhood Search: LNS) คือวิธีที่พัฒนามาจากวิธีการค้นหาเฉพาะแห่ง เพื่อใช้ในการแก้ปัญหาขนาดใหญ่ โดยวิธีการในการแก้ปัญหาจะคล้ายคลึงกับวิธีการค้นหาเฉพาะแห่ง แต่จะมีระยะของการค้นหาผลเฉลยข้างเคียง (Neighbourhood solution) ที่กว้างขึ้น กล่าวคือ เส้นทางการเดินรถในแต่รอบของการวนรอบจะมีลักาณะที่แตกต่างกันอย่างมาก เช่น ทำการวนรอบโดย

เปลี่ยนวิธีหาผลเฉลยเบื้องต้น (Initial Solution) วิธีการนี้สามารถช่วยแก้ปัญหาดีที่สุดเฉพาะพื้นที่ (Local Optima) ในการแก้ปัญหาขนาดใหญ่ลงได้ ดังจะกล่าวถึงรายละเอียดในลำดับถัดไป

รูปที่ 2.11 แสดงการเกิดปัญหาดีที่สุดเฉพาะแห่ง (Local Optima) โดยผลเฉลยเบื้องต้นปัจจุบันอยู่ที่ตำแหน่ง Current State เห็นได้ว่าเมื่อทำการวนรอบด้วยวิธีค้นหาเฉพาะแห่ง (NS) ซึ่งมีระยะของการค้นหาผลเฉลยข้างเคียงที่แคบ ผลเฉลยจะค่อยๆ มีแนวโน้มไปในทิศทางเข้าหาผลเฉลยดีที่สุดเฉพาะแห่ง (Local Optimal) และในที่สุด จะได้ผลเฉลยสุดท้ายเป็นผลเฉลยดีที่สุดเฉพาะแห่งเท่านั้น ซึ่งจะเห็นได้ว่ายังมีผลเฉลยอื่นๆ ที่ให้ผลเฉลยที่ดีกว่าผลเฉลยดีที่สุดเฉพาะแห่ง แต่หากทำการแก้ปัญหาด้วยวิธีค้นหาเฉพาะแห่งขนาดใหญ่ (LNS) ซึ่งมีระยะของการหาผลเฉลยข้างเคียงที่กว้าง แล้วพบว่าผลเฉลยสามารถกระโดดข้ามจากจุด Current State ไปยังจุด A ได้ และเมื่อวนรอบเพื่อหาผลเฉลยที่ดีที่สุดแล้ว ผลเฉลยจะมีแนวโน้มไปทางผลเฉลยที่ดีที่สุด (Global Optimal) จนได้ผลเฉลยสุดท้ายเป็นผลเฉลยที่ดีที่สุด สังเกตได้ว่า แม้ตำแหน่งของผลเฉลยเบื้องต้นที่เปลี่ยนแปลงแตกต่างกันไม่เยอะ ก็สามารถให้ผลเฉลยสุดท้ายที่แตกต่างกันมากได้ เนื่องมาจากวิธีการค้นหาเฉพาะแห่ง เป็นวิธีที่วนรอบเพื่อไปในทิศทางที่ผลเฉลยมีแนวโน้มที่ดีขึ้นโดยไม่ได้สนใจว่าผลเฉลยที่ดีที่สุดอยู่ในทิศทางใด แต่วิธีการค้นหาเฉพาะแห่งขนาดใหญ่แก้ปัญหาดังกล่าวได้ด้วยการขยายระยะการค้นหาผลเฉลยข้างเคียง จึงสรุปได้ว่าในการแก้ปัญหาขนาดใหญ่นั้นวิธีการค้นหาเฉพาะแห่งขนาดใหญ่เหมาะสมกว่าวิธีการค้นหาเฉพาะแห่ง ซึ่งปัญหาการจัดเส้นทางการเดินรถก็จัดเป็นปัญหาขนาดใหญ่เช่นเดียวกัน อย่างไรก็ดีวิธีแก้ปัญหาด้วยการเปลี่ยนผลเฉลยเบื้องต้นก็ยังคงมีปัญหา เนื่องจากการขยายขนาดการค้นหาผลเฉลยข้างเคียงนั้นส่งผลกระทบต่อระยะเวลาที่ใช้ในการหาผลเฉลย กล่าวคือ ยิ่งระยะการค้นหาผลเฉลยข้างเคียงกว้างเท่าไหร่ ก็จะยิ่งใช้เวลาในการหาผลเฉลยมากขึ้นเท่านั้น



รูปที่ 2.11 ปัญหาดีที่สุดเฉพาะแห่ง

เพื่อที่จะแก้ปัญหาดีที่สุดเฉพาะแห่ง (Local Optima) ในบางครั้งยังสามารถใช้วิธี Trial Swap ภายนอกเพื่อแก้ปัญหาดีที่สุดเฉพาะแห่งได้อีกด้วย โดยจะทำการสลับเปลี่ยนภายนอกแม้ว่าผลเฉลยหลังจากการสลับเปลี่ยนจะไม่ให้ค่าการฝ่าฝืนข้อจำกัดรองที่ต่ำกว่าก็ตาม การกระทำดังกล่าวเปรียบเสมือนการเปลี่ยนผลเฉลยเบื้องต้น อย่างไรก็ตามข้อจำกัดหลักยังคงเป็นข้อจำกัดที่ห้ามฝ่าฝืนเสมอไม่ว่าในกรณีใดๆ

2.5 งานวิจัยในอดีตเกี่ยวกับปัญหาการจัดเส้นทางการเดินรถ

Pepin (2006) ทำการศึกษาเปรียบเทียบประสิทธิภาพในการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีฮิวริสติคที่แตกต่างกัน 5 วิธี พบว่าแม้ว่าวิธีกำเนิดสดมภ์จะให้ผลเฉลยที่มีคุณภาพมากที่สุดแต่ใช้เวลาในการแก้ปัญหาที่มาก ในขณะที่วิธีการค้นหาเฉพาะแห่งขนาดใหญ่มีคุณภาพผลเฉลยที่แย่กว่าเล็กน้อยแต่กลับใช้เวลาในการแก้ปัญหาที่น้อยกว่ามาก ดังนั้นจึงสรุปได้ว่าวิธีการค้นหาเฉพาะแห่งเป็นวิธีฮิวริสติคที่เหมาะสมในการแก้ปัญหาการจัดเส้นทางการเดินรถ

Shaw (1998) ทำการศึกษาการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธี Large Scale Neighborhood Search (LNS) บนข้อจำกัดทางด้านช่วงเวลากำหนดส่งสินค้า (Time Windows) และความจุ (Capacity) ขั้นตอนในงานวิจัยนี้ใช้หลักการคัดออก (Removal) และ แทรกกลับ (Re-insertion) โดยทำการทดสอบวิธีในการแทรกกลับหลายวิธี ได้ผลสรุปว่าวิธีการในการแทรกกลับที่แตกต่างกันส่งผลกระทบต่อคุณภาพของผลเฉลยและระยะเวลาในการประมวลผลอย่างมาก ดังนั้น

สิ่งที่จำเป็นต้องคำนึงถึงในการใช้วิธีการนี้คือ วิธีการในการคัดออกและแทรกกลับที่มีประสิทธิภาพและเหมาะสมกับปัญหา

Ergu และ Savelsbergh (2004) ทำการแก้ปัญหา Time-constrained lane covering (TCLC) ด้วยวิธีฮิวริสติคโดยจะทำการกำหนดให้ตำแหน่งเริ่มต้น (Starting Location) และระยะเวลาการเดินทาง (duration of tour) เป็นค่าคงที่ในการแก้ปัญหาและมีสมการเป้าหมาย (Objective Function) คือ Minimize Duration of tour โดยใช้ข้อจำกัดคือ Time windows ซึ่งขั้นตอนของงานวิจัยเริ่มต้นด้วยการกำหนดรอบการขนส่ง (Cycle) ที่เป็นไปได้ หรือ Cycle Generation ด้วยข้อจำกัด 2 ประการ คือ ระยะทางสูงสุดต่อรอบการขนส่ง และ จำนวนจุดขนส่งสูงสุดต่อรอบการขนส่ง จากนั้นจำกัดข้อจำกัดเกี่ยวกับกำหนดการขนส่งด้วย Time Windows Algorithm ซึ่งมีหลักการคือ วนรอบเปลี่ยนเลือกจุดที่มีระยะเวลาการรอ (Waiting Time) สูงที่สุดมาเป็นจุดเริ่มต้นเสมอ (Origin) สุดท้ายจากงานวิจัยนี้สรุปได้ว่าวิธีการนี้จะให้ผลเฉลยที่ดีเทียบเคียงกับวิธีหาผลเฉลยที่ดีที่สุดแต่ไม่สอดคล้องกับความเป็นจริงเนื่องจากเลือกจุดที่มีระยะเวลารอคอยสูงสุดเป็นจุดเริ่มต้นเสมอซึ่งในความเป็นจริงลูกค้าที่มีกำหนดการขนส่งสินค้าช่วงเย็นจะไม่สามารถเป็นจุดเริ่มต้นที่ดีได้เลย แต่ในงานวิจัยนี้สามารถเป็นได้เนื่องจากกำหนดช่วงเวลาการขนส่งสินค้าเป็นค่าคงที่ ยิ่งไปกว่านั้นจากงานวิจัยนี้สามารถสรุปเพิ่มเติมได้ว่าหากเพิ่มระยะทางสูงสุดต่อรอบการส่งสูงสุด หรือ จำนวนจุดขนส่งสูงสุดต่อรอบการขนส่งสามารถช่วยเพิ่มคุณภาพของผลเฉลยได้แต่จะทำให้ปัญหามีขนาดใหญ่ขึ้นอย่างมาก กล่าวคือ การหย่อนข้อจำกัดการทำ Cycle Generation สามารถช่วยเพิ่มคุณภาพของผลเฉลยได้แต่จะส่งผลกระทบให้ขนาดปัญหาใหญ่ขึ้นตามไปด้วย ดังนั้นการเลือกข้อจำกัดการสร้างรอบการขนส่งเป็นสิ่งที่สำคัญอย่างยิ่งในการแก้ปัญหาการจัดเส้นทางการเดินรถ

Bent และ Hentenryck (2006) เสนอวิธีที่ผสมผสานระหว่างวิธี 2 วิธีที่ได้รับความนิยมในการจัดเส้นทางการเดินรถไว้ด้วยกัน 2 ขั้นตอน โดยในขั้นตอนแรกใช้วิธี Simulated annealing (SA) ในการลดจำนวนเส้นทางเดินรถ (Route) เพื่อช่วยลดขนาดของปัญหาและเพิ่มความถูกต้องของผลเฉลยเมื่อใช้วิธีฮิวริสติค และในขั้นตอนที่ 2 ใช้วิธี LNS ในการหาเส้นทางการเดินรถที่มีค่าใช้จ่ายต่ำที่สุด ซึ่งอัลกอริทึมที่พัฒนาขึ้นนี้สามารถหาคำตอบของปัญหา VRP ที่จำนวนลูกค้ามากถึง 600 แห่งภายในเวลาอันรวดเร็วและได้ผลเฉลยที่มีประสิทธิภาพสูงกว่าวิธีฮิวริสติคอื่น ดังนั้นงานวิจัยนี้เป็นผลลัพธ์ที่สามารถสนับสนุนความคิดที่ว่าการรวมข้อดีของหลายวิธีเพื่อพัฒนาวิธีการแก้ปัญหาใหม่สามารถช่วยเพิ่มประสิทธิภาพของวิธีการในการแก้ปัญหาได้

เกรียงศักดิ์ และ ณกร (2007) ทำการศึกษาวิธีประยุกต์ระหว่างวิธี Column Generation (CG) และ Large Scale Neighbourhood Search เพื่อแก้ปัญหา Pickup and Delivery Problem (PDP) โดยใช้ CG ในการจำกัดขนาดปัญหาและกำหนดสร้างปัญหาย่อย โดยโครงสร้างของ CG เป็นรูปแบบที่ออกแบบมาเฉพาะรูปแบบของปัญหาหรือเงื่อนไขของปัญหาภายใต้ Branch and Bound Tree และ

ใช้วิธี Local Search ซึ่งมีโครงสร้างที่ง่ายและยืดหยุ่นมาประยุกต์ใช้ในการหาผลเฉลยของปัญหาย่อย

เกรียงศักดิ์ และ ณกร ใช้ข้อจำกัดหลัก (Hard Constraint) ทั้งหมด 5 อย่างด้วยกันคือ

1) งานรับหรือส่งสินค้าแต่ละงานต้องถูกรองรับด้วยรถเพียงคันเดียว
2) การรับและส่งสินค้าชิ้นเดียวกันต้องเป็นกิจกรรมที่ปฏิบัติโดยรถคันเดียวกัน
3) การรับสินค้าเป็นกิจกรรมที่ต้องกระทำก่อนการส่งสินค้าชิ้นนั้น
4) ปริมาณสินค้าภายในยานพาหนะเมื่อมาถึงจุดรับ/ส่งสินค้าลำดับที่ i จะต้องมีปริมาณเท่ากับผลรวมปริมาณสินค้าเมื่อมาถึงจุดที่ i-1 กับ ปริมาณสินค้าที่ให้บริการที่จุด i-1
5) ปริมาณสินค้าที่บรรทุกบนยานพนะต้องไม่เกินความจุที่สามารถรองรับได้

นอกจากนี้เกรียงศักดิ์ และ ณกร กำหนดข้อจำกัดรอง (Soft Constraint) ทั้งหมด 2 ข้อดังต่อไปนี้

1) จำนวนยานพาหนะที่ใช้
2) ค่าใช้รวมในการเดินทางเพื่อการขนส่ง

สังเกตได้ว่าข้อจำกัดรองสามารถมีมากกว่า 1 ข้อจำกัดได้ ทั้งที่การเปรียบเทียบแต่ละเส้นทางการเดินรถนั้นต้องเปรียบเทียบกันบนมาตรฐาน (Standard) เดียวกัน โดยใช้ตัวแปรน้ำหนัก (Weight Factor) ของแต่ละข้อจำกัดลงไป กล่าวคือหากการเพิ่มขึ้นของจำนวนรถหนึ่งคันส่งผลกระทบเทียบเท่าระยะทางที่เพิ่มขึ้น 10 กิโลเมตร ค่าของตัวแปรน้ำหนักค่าเท่ากับ 10 กิโลเมตรต่อคัน เป็นต้น

ยศศิริ (2007) ศึกษาวิธีการแก้ปัญหาการจัดเส้นทางการเดินรถแบบเต็มคันรถ กล่าวคือไม่มีการรับสินค้าเพิ่มเติมระหว่างทางโดยการควบรวมรอบการขนส่งเพื่อให้เกิดเส้นทางการขนส่งอย่างต่อเนื่อง ยิ่งไปกว่านั้นยังมีการเพิ่มเงื่อนไขตามสภาพความเป็นจริงของปัญหา เช่น ลักษณะสินค้า ฝูงรถ เวลาการนำสินค้าขึ้น-ลง ช่วงเวลาห้ามวิ่ง และกรอบเวลา เป็นต้น พบว่าการกำหนดเงื่อนไขสามารถลดขนาดของปัญหาลงได้อย่างมากซึ่งเป็นข้อดีของแบบจำลองแบบเส้นทาง (Path Base) อย่างไรก็ดีการกำหนดเงื่อนไขเป็นสิ่งที่พึงระวังอย่างยิ่ง เนื่องจากการจำกัดเงื่อนไขที่มากเกินไปอาจทำให้จำกัดขอบเขตในการค้นหาผลเฉลยซึ่งส่งผลให้ประสิทธิภาพของโปรแกรมถูกจำกัดจนได้ผลเฉลยที่มีประสิทธิภาพน้อยกว่าที่ควรจะเป็น และจากงานวิจัยนี้สรุปได้ว่า การแก้ปัญหาด้วยแบบจำลองขนาดเต็มและการแก้ปัญหาด้วยเทคนิคการกำเนิดสดมภ์ให้ค่าผลเฉลยที่ดีที่สุดมีค่า

ใกล้เคียงกันอย่างมาก แต่ การแก้ปัญหาด้วยเทคนิคการกำเนิดสดมภ์ใช้เวลาในการแก้ปัญหาน้อยกว่าอย่างมาก นอกจากนี้ ยศศิริ อดุลยศักดิ์ ยังเสนอแนวคิดในการพัฒนาวิธีการกำเนิดสดมภ์โดยการพัฒนาขั้นตอนในการหาเส้นทางที่สามารถลดค่าใช้จ่ายได้มากที่สุดหรือการหาเส้นทางที่มีค่าใช้จ่ายต่ำสุด (Shortest Path Algorithm) มาประยุกต์ร่วม ซึ่งจะทำการทดสอบการพัฒนาวิธีกำเนิดสดมภ์ด้วยเทคนิคการหาเส้นทางที่สั้นที่สุดดังกล่าวในงานวิจัยนี้

งานวิจัยของ Liu (2003) ศึกษาวิธีการแก้ปัญหาเส้นทางการเดินด้วยวิธีผสมผสานระหว่างวิธีใช้จุดกระจายสินค้า (Hub-and-spoke) และ ขนส่งโดนตรง (Direct Shipment) โดยใช้ค่าใช้จ่ายเป็นตัวชี้วัดในการช่วยแก้ปัญหา สรุปได้ว่าวิธีผสมผสานนี้มีประสิทธิภาพในการแก้ปัญหาที่ดีกว่าวิธีทั้ง 2 ข้างต้น เห็นได้ว่าในบางครั้งหากนำวิธีพื้นฐานที่ไม่มีความซับซ้อนมาผสมผสานใช้ในการแก้ปัญหาก็อาจทำให้เกิดวิธีการแก้ปัญหาที่มีประสิทธิภาพได้

งานวิจัยของ Ching (2004) ศึกษาการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยวิธีการค้นหาเฉพาะแห่ง โดยการจำแนกปัญหาออกเป็น 2 กลุ่มคือ สินค้าเต็มคันรถ และ สินค้าไม่เต็มคันรถ ก่อนจะทำการแก้ปัญหาด้ววิธีการวนรอบเพื่อหาผลเฉลย สรุปได้ว่าวิธีการนี้เป็นวิธีการที่สามารถแก้ปัญหาได้อย่างรวดเร็ว แต่คุณภาพของผลเฉลยจะค่อนข้างแตกต่างกับผลเฉลยที่ดีที่สุด

อัลกอริทึมของวิธีการค้นหาเฉพาะแห่งขนาดใหญ่ที่ได้รับการยอมรับว่ามีประสิทธิภาพโดยรวมดีที่สุดในปัจจุบันคือ งานวิจัยของ Pisinger และ Ropke (2007) ซึ่งเริ่มต้นทำการวนรอบเพื่อหาผลเฉลยด้วยการทำลายเส้นทางการเดินรถเดิม (Destruction) โดยวิธีในการทำลายเส้นทางการเดินรถมีด้วยกัน 2 วิธีคือ

1. Neighbourhood Operators คือ การเลือกทำลายด้วยตัวชี้วัดอย่างใดอย่างหนึ่ง เช่น เลือกทำลายจุดที่จะสามารถลดค่าใช้จ่ายได้มากที่สุดเมื่อออกไปจากเส้นทางการเดินรถนั้นๆ เป็นต้น
2. Roulette – wheel คือ การเลือกใช้ตัวชี้วัดหลายตัว แต่ละรอบของการวนรอบจะใช้ตัวชี้วัดที่แตกต่างกัน โดยใช้หลักความน่าจะเป็นและสถิติเข้าช่วยในการเลือกตัวชี้วัดที่จะนำมาใช้ในแต่ละรอบ

หลังจากทำการทำลายโดยนำจุดบางจุดออกจากเส้นทางการเดินรถแล้วอัลกอริทึมจะทำการสร้างเส้นทางการเดินรถใหม่ (Reconstruction) ด้วยวิธีที่มีลักษณะคล้ายคลึงกับการทำลายเส้นทาง

การเดินรถ ทำเช่นนี้จนได้เฉลยที่ต้องการจึงทำการหยุดการวนรอบ จากงานวิจัยสรุปได้ว่าอัลกอริทึมนี้สามารถหาผลเฉลยที่ใกล้เคียงกับผลเฉลยที่ดีที่สุดในระยะเวลาสั้นในการแก้ปัญหาการจัดเส้นทางการเดินรถขนาดใหญ่

งานวิจัยของ Desaulniers (2007) นำเสนอวิธีผสม (Hybrid Method) โดยนำวิธีการค้นหาเฉพาะแห่งของ Pisinger และ Ropke (2007) มาพัฒนาโดยการใช้วิธีกำเนิดสดมภ์ซึ่งเป็นวิธีการค้นหาผลเฉลยที่ดีที่สุดมาประยุกต์ใช้ในขั้นตอนการสร้างเส้นทางการเดินรถใหม่ (Reconstruction) สรุปได้ว่าวิธีที่พัฒนาขึ้นให้ผลเฉลยที่ดีกว่าวิธีค้นหาเฉพาะแห่งของ Pisinger และ Ropke เมื่อใช้กับปัญหาที่มีลูกค้าไม่เกิน 200 ราย อย่างไรก็ดีวิธีที่พัฒนาขึ้นใช้เวลาในการแก้ปัญหาที่สูงกว่าอย่างมากเมื่อเปรียบเทียบกัน

จากการศึกษางานวิจัยที่ผ่านมาในอดีตพบว่าการแก้ปัญหาการจัดเส้นทางการเดินรถด้วยเทคนิคต่างๆ ทั้งการหาผลเฉลยด้วยวิธีหาผลเฉลยที่ดีที่สุด วิธีฮิวริสติค และ วิธีผสม (Hybrid Method) นั้นต่างมีจุดเด่นและจุดด้อยที่แตกต่างกันออกไป ยิ่งไปกว่านั้นรูปแบบการแก้ปัญหายังแตกต่างกันไปตามลักษณะเฉพาะของแต่ละปัญหา เพื่อที่จะสามารถหาผลเฉลยที่มีคุณภาพในระยะเวลาการทำงานที่สามารถยอมรับได้ ในบทต่อไปจะกล่าวถึงขั้นตอน รายละเอียด และลักษณะเฉพาะของปัญหาในการแก้ปัญหาการจัดเส้นทางแบบเต็มคันของงานวิจัยนี้

2.6 ทฤษฎีความน่าจะเป็น

ความน่าจะเป็น (Probability) คือค่าที่แสดงให้ทราบว่าเหตุการณ์หนึ่งๆ มีโอกาสเกิดขึ้นมากหรือน้อยเพียงใด ซึ่งเป็นค่าที่สามารถใช้ประโยชน์ในการทดลองที่ไม่สามารถทำนายหรือบอกผลของเหตุการณ์ได้ล่วงหน้า หรือที่เรียกอีกอย่างหนึ่งว่าการทดลองเชิงสุ่ม (Random Experiment) ถ้าทำการทดลอง N ครั้ง และเกิดเหตุการณ์ E ขึ้น n ครั้ง ความน่าจะเป็นของเหตุการณ์ E แทนด้วยสัญลักษณ์ P(E) สามารถหาได้จากสมการความสัมพันธ์ดังนี้

$$P(E) = \lim_{n \to \infty} \frac{n}{N} \qquad (2.9)$$

หากทำการทดลอง N ครั้งและเหตุการณ์สนใจเกิดขึ้นทั้งหมด n ครั้ง ค่าความน่าจะเป็นของการเกิดเหตุการณ์จะสามารถหาได้ตามสมการที่ 6 เช่น การทดลองโยนเหรียญ 10 ครั้ง โดยเหตุการณ์ที่สนใจคือโยนเหรียญออกหัว หากโยนเหรียญได้หัวทั้งหมด 7 ครั้ง ค่าความน่าจะเป็นของ

การโยนเหรียญได้หัวมีค่าเท่ากับ 0.7 เป็นต้น และสามารถประมาณค่าความน่าจะเป็นได้โดยไม่ต้องคำนึงถึงลิมิตได้เมื่อ N มีค่ามากๆ ดังนั้นเมื่อ N มีค่ามากๆ สามารถหาค่าความน่าจะเป็นได้ดังนี้

$$P(E) = \frac{n}{N} \qquad (2.10)$$

การกระทำของเหตุการณ์ (Operation of events) ในทฤษฎีของความน่าจะเป็นที่สำคัญมีด้วยกัน 4 รูปแบบดังนี้

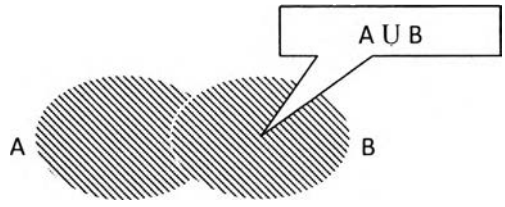
1) ยูเนียนของเหตุการณ์ (Union of Event) ดังแสดงตัวอย่างในรูปที่ 2.9 โดยวงกลม 2 วง คือ เหตุการณ์ใดๆ A และ B ยูเนียนของ A กับ B จะถูกเขียนแทนด้วยสัญลักษณ์ $A \cup B$ หมายถึงเหตุการณ์ที่ประกอบไปด้วยสมาชิกของเหตุการณ์ A หรือสมาชิกของเหตุการณ์ B หรือ สมาชิกที่อยู่ทั้งในเหตุการณ์ A และ B ซึ่งก็คือพื้นที่แรเงาทั้งหมดนั่นเอง

2) อินเตอร์เซ็คชั่นของเหตุการณ์ (Intersection of Event) ดังแสดงตัวอย่างในรูปที่ 2.10 โดยวงกลม 2 วง คือเหตุการณ์ใดๆ A และ B อินเตอร์เซ็คชั่นของ A กับ B จะถูกเขียนแทนด้วยสัญลักษณ์ $A \cap B$ หมายถึง สมาชิกที่อยู่ทั้งในเหตุการณ์ A และ B ซึ่งก็คือพื้นที่แรเงาทั้งหมดนั่นเอง

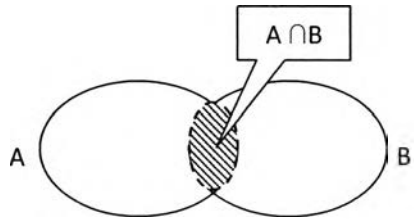
3) คอมพลีเมนต์ของเหตุการณ์ (Complement of Event) ดังแสดงตัวอย่างในรูปที่ 2.11 โดยวงกลม 2 วง คือเหตุการณ์ใดๆ A และ B คอมพลีเมนต์ของ A จะถูกเขียนแทนด้วยสัญลักษณ์ A' หมายถึง สมาชิกที่อยู่นอกเหตุการณ์ A ซึ่งก็คือพื้นที่แรเงาทั้งหมดนั่นเอง

4) คอนดิชั่นของเหตุการณ์ (Condition of Event) หาก A และ B คือเหตุการณ์ใดๆ A คอนดิชั่น B จะถูกเขียนแทนด้วยสัญลักษณ์ A/B หมายถึงเหตุการณ์ A โดยกำหนดให้เกิดเหตุการณ์ B ไปแล้ว ดังนั้น P(A/B) หมายถึง ความน่าจะเป็นที่จะเกิดเหตุการณ์ A เมื่อเกิดเหตุการณ์ B ไปแล้ว มีสมการความสัมพันธ์ดังนี้

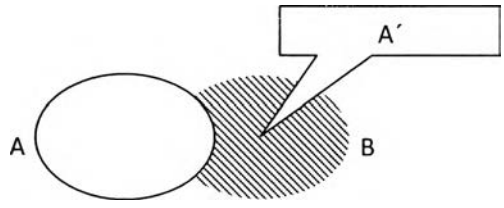
$$P(A/B) = \frac{P(A \cap B)}{P(B)} \qquad (2.11)$$



รูปที่ 2.12 ยูเนียนของเหตุการณ์



รูปที่ 2.13 อินเตอร์เซ็คชั่นของเหตุการณ์



รูปที่ 2.14 คอมพลีเมนต์ของเหตุการณ์

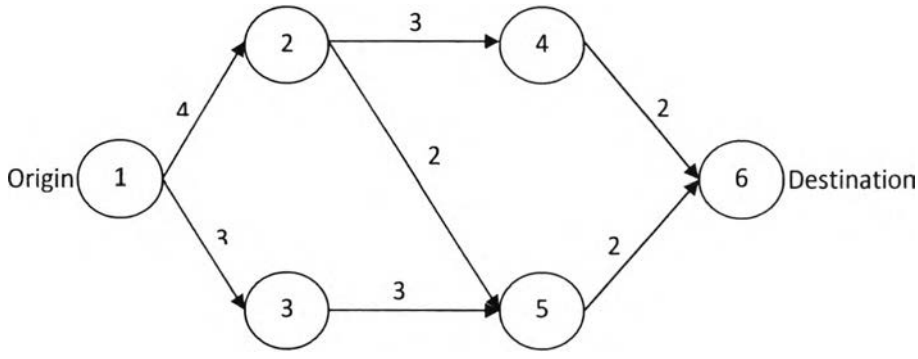
2.7 การหาเส้นทางที่สั้นที่สุด (Shortest Path Algorithm)

การหาเส้นทางที่สั้นที่สุด (Shortest Path Algorithm) คือวิธีในการหาเส้นทางที่มีความยากลำบาก (Impedance) เช่น ระยะทาง และ เวลาการเดินทาง น้อยที่สุดในการเดินทางจากจุดต้นทางไปยังจุดปลายทาง ซึ่งวิธีการในการหาเส้นทางที่สั้นที่สุดนั้นสามารถทำได้มากมายหลายวิธี แต่วิธีที่ได้รับความนิยมและมีการวิจัยเกี่ยวกับประสิทธิภาพในการทำงานอย่างแพร่หลายคือ Dijkstra Algorithm ซึ่งเป็นวิธีการหาผลเฉลยที่ดีที่สุด และ A-Star Algorithm ที่เป็นวิธีฮิวริสติค รายละเอียดของวิธีจะกล่าวถึงในลำดับถัดไป

2.7.1 ไดรช์ตร้า (Dijkstra Algorithm)

อัลกอริทึมหนึ่งที่แพร่หลาย เสถียรภาพสูง และมีประสิทธิภาพในการหาเส้นทางที่สั้นที่สุด คืออัลกอริธึมของ Dijkstra (1963) วิธีการนี้จะเริ่มต้นด้วยการกำหนดระยะทางไปยังจุดปลายทาง (Destination) สำหรับทุก node โดยกำหนดค่าความลำบากในการไปยังจุดต้นทาง (Origin) ให้ค่ามีเป็นศูนย์นอกจากนั้นให้มีค่าเป็น ∞ จากนั้นทำการวนรอบในการหาเส้นทางออกจากจุดต้นทางโดยเลือกเก็บค่าความลำบากที่น้อยที่สุดในการเดินทางไปยังแต่ละจุดไว้เป็นผลเฉลย จนกระทั่งได้ค่าความลำบากที่น้อยที่สุดในการเดินทางไปจุดปลายทาง อย่างไรก็ดี Dijkstra Algorithm มีข้อด้อยที่สำคัญมากประการหนึ่งคือ ความลำบากในการเดินทางของแต่ละ Arc ต้องมีค่าเป็นจำนวนจริงบวก กล่าวคือต้องมีค่ามากกว่าหรือเท่ากับศูนย์ อีกทั้งวิธีการนี้ยากแก่การเข้าใจ ผู้วิจัยจึงทำการสร้างปัญหาตัวอย่างดังรูปที่ 2.12

<u>ตัวอย่างการแก้ปัญหา Dijkstra Algorithm</u>



รูปที่ 2.15 ตัวอย่างการแก้ปัญหาด้วย Dijkstra Algorithm

ขั้นตอนที่ 1: กำหนดเซตผลเฉลยเบื้องต้นของทั้ง 6 nodes [0, ∞, ∞, ∞, ∞, ∞]

ขั้นตอนที่ 2: รอบที่ 1 หาระยะทางสั้นที่สุดที่ออกจากจุดเริ่มต้น (node1)

จากนั้นนำมาใส่ในเซตของผลเฉลยได้เลยเนื่องจาก 4 และ 3 มีค่าน้อยกว่า∞

ได้ผลลัพธ์ดังนี้ [0, 4, 3, ∞, ∞, ∞]

ขั้นตอนที่ 3: กระทำซ้ำขั้นตอนที่ 2 จะกระทั่งได้ความลำบากที่น้อยที่สุดในการเดินทางไปจุดปลายทาง (node6) ได้ผลเฉลยดังนี้ [0, 4, 3, 7, 6, 8] เป็นอันสิ้นสุดกระบวนการทำงาน สรุปได้ว่าความลำบากในการเดินทางจาก node 1 ไปยัง node 6 ที่น้อยที่สุดคือ 8

อย่างไรก็ดีไดรช์ตร้าเป็นวิธีหาผลเฉลยที่ดีที่สุด ดังนั้นจึงมีข้อดีและข้อเสียเช่นเดียวกันกับวิธีหาผลเฉลยที่ดีที่สุดทั่วไปนั่นก็คือ ได้ผลเฉลยที่ดีที่สุดแต่ใช้เวลาในการประมวลผลที่นาน รวมไปถึงขนาดปัญหาที่ใหญ่และซับซ้อน ดังนั้นในปัจจุบันมีการพัฒนาวิธีการหาเส้นทางที่สั้นที่สุดแบบฮิวริสติคที่ใช้ชื่อว่า A-Star Algorithm ดังที่จะกล่าวในหัวข้อถัดไป

2.7.2 เอสตาร์ (A-star Algorithm)

เอสตาร์ (A-star Algorithm) หรือ A* Algorithm คือวิธีในการหาเส้นทางที่สั้นที่สุดแบบฮิวริสติค จึงมีวิธีในการค้นหาเส้นทางที่สั้นที่สุดที่แตกต่างกันออกไปตามแต่ความคิดของผู้พัฒนา มีสมการพื้นฐานดังสมการที่ 2.11

$$f'(n) = g(n)+h'(n) \qquad (2.11)$$

โดยที่ g(n) คือระยะทางทั้งหมดจากจุดเริ่มต้นถึงตำแหน่งปัจจุบัน(n)

$h'(n)$ คือระยะทางโดยประมาณจากตำแหน่งปัจจุบัน(n)ไปยังจุดปลายทาง

$f'(n)$ คือระยะทางโดยประมาณจากจุดเริ่มต้นไปยังจุดปลายทาง

วิธีเอสตาร์จะทำการวนรอบเพื่อหาเส้นทางการเดินทางออกจากจุดเริ่มต้น โดยจะเลือกเดินทางไปยังจุดที่มีค่าระยะทางโดยประมาณ (f'(n)) น้อยที่สุดเรื่อยๆ จนถึงจุดปลายทาง อย่างไรก็ดีค่าระยะทางโดยประมาณจะถูกต้องมากหรือน้อยขึ้นอยู่กับฟังชั่นการประมาณค่าระยะทางจาก

ตำแหน่งปัจจุบันไปยังจุดปลายทาง ดังนั้นประสิทธิภาพการแก้ปัญหาจึงขึ้นอยู่กับผู้ออกแบบวิธีการแก้ปัญหา ว่าจะสามารถออกฟังชั่นการประมาณค่าได้ใกล้เคียงค่าจริงมากเพียงใด