การใช้ไมโครคอมพิวเตอร์แก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์

โดย

นายมงคล พงษารัตน์

นายยุพราช ฉายสุริยะ

นายวิชญพงศ์ ศรีคชา

ปัญหาพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาสถิติประยุกต์ ภาควิชาสถิติประยุกต์

คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

ปีการศึกษา 2531

บทคัดย่อ

นายมงคล พงษารัตน์ วิทยาศาสตรบัณฑิต (สถิติประยุกต์)

นายยุพราช ฉายสุริยะ

นายวิชญพงศ์ ศรีคชา

สาขาวิชาสถิติประยุกต์

เรื่อง การใช้ไมโครคอมพิวเตอร์แก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์

การแก้ไขปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์นั้น เป็นการแก้ไขปัญหาเพื่อหาค่าคำตอบ โดยที่ค่าคำตอบนั้นจะเป็นค่าสูงสุดหรือต่ำสุดและค่าคำตอบนั้นจะต้องเป็นตัวเลขจำนวนเต็ม จุดมุ่งหมายที่ได้จัดทำโปรแกรมสำเร็จรูปนี้ขึ้นมา เพราะยังไม่มีผู้ใดจัดทำขึ้นมา ทางคณะผู้จัดทำได้พิจารณาถึงข้อได้เปรียบของวิธีนี้เห็นสมควรว่าน่าจะทำเป็นโปรแกรมคอมพิวเตอร์ขึ้น จึงได้ทำการศึกษาวิธีการ และทำการวางระบบงาน เพื่อทำการเขียนโปรแกรมนี้ขึ้นมา โดยภาษาที่ใช้ในการเขียนนั้นใช้ภาษาปาสคาลโดยใช้ Turbo PASCAL Version 5.0 โดยเขียนบนเครื่อง IBM PC/XT เนื่องจากว่าเป็นเครื่องคอมพิวเตอร์ที่มีใช้กันมาก ส่วนข้อได้เปรียบของวิธีบรานช์แอนด์บาวด์นี้ก็คือ ถ้าค่าคำตอบที่ได้นั้นยังไม่เป็นค่าที่ดีที่สุด (optimal) คือเป็นค่าต่ำสุดหรือสูงสุด ก็ยังมีค่าคำตอบอื่นที่สามารถนำไปใช้ก่อนได้ แต่ถ้าใช้วิธีอื่นถ้าค่าคำตอบที่ได้ยังไม่เป็นค่าที่ดีที่สุด ก็ไม่สามารถที่จะนำ คำตอบใดไปใช้ได้ และเมื่อใช้โปรแกรมนี้แก้ไขปัญหาโปรแกรมจำนวนเต็มทำให้ได้ผลลัพธ์รวดเร็วมาก และโปรแกรมนี้สามารถนำไปประยุกต์ใช้กับงานที่เกิดขึ้นจริงได้

กิตติกรรมประกาศ

ขอขอบคุณอาจารย์สมศรี บัณฑิตวิไล ที่ให้คำปรึกษาและช่วยแก้ไขปัญหาต่าง ๆ ขอขอบคุณพระเจ้าที่ประทานสมองมาให้ ขอบคุณน้อง ๆ และเพื่อน ๆ ที่ช่วยกันให้กำลังใจ

ปัญหาพิเศษนี้สำเร็จลงได้ด้วยความร่วมมือของทุกคน

ผู้จัดทำ

สารบัญ

หน้า

บทที่ 1

บทนำ

ความเป็นมาของปัญหา

เนื่องจากว่าในปัจจุบันนี้เครื่องไมโครคอมพิวเตอร์ (Microcomputer) มีราคาถูกลงมาก จึงทำให้หน่วยงานต่าง ๆ ทั้งเอกชนและรัฐ มีเครื่องคอมพิวเตอร์ใช้กันแทบทุกหน่วยงาน ดังนั้นจึงทำให้บทบาทของเครื่องคอมพิวเตอร์มีมากขึ้น เช่น ใช้เก็บประวัติของผู้ป่วยที่เข้ามารักษาในโรงพยาบาล ใช้แทนเครื่องพิมพ์ดีด ฯลฯ จึงทำให้การสร้างโปรแกรม การพัฒนาโปรแกรมต่าง ๆ มีมากขึ้น รวมทั้งโปรแกรมทางด้านสถิติได้เกิดขึ้นมามากมาย เช่น SPSS/PC+ Statgraphics ฯลฯ แต่ยังไม่มีโปรแกรมที่ใช้แก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์

การศึกษาในปีที่ 3 ของนักศึกษาภาควิชาสถิติประยุกต์ คณะครุศาสตร์อุตสาหกรรมและวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง ในภาคการศึกษาที่ 2 วิชาการวิจัยการดำเนินงาน 2 (Operations Research II) ได้ศึกษาวิธีการแก้ปัญหาโปรแกรมจำนวนเต็ม (Integer Programming) ในการแก้ปัญหาโปรแกรมจำนวนเต็มนั้นมีหลายวิธีที่สามารถแก้ปัญหาได้ แต่ในที่นี้จะกล่าวถึงเฉพาะการแก้ไขปัญหาจำนวนเต็ม โดยใช้วิธีบรานช์แอนด์บาวด์ (Branch and Bound Method) วิธีนี้เป็นวิธีที่มีประสิทธิภาพมาก และมีข้อดีกว่าวิธีอื่นคือถ้าค่าคำตอบที่ได้นั้นยังไม่เป็นค่าที่ดีที่สุด (optimal) คือเป็นค่าต่ำสุดหรือสูงสุด ก็ยังมีค่าคำตอบอื่นที่สามารถนำไปใช้ก่อนได้ แต่วิธีอื่นถ้าค่าคำตอบที่ได้ยังไม่เป็นค่าที่ดีที่สุด ก็ไม่สามารถที่จะนำคำตอบใดไปใช้ได้

เนื่องจากการแก้ไขปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์ เป็นวิธีที่มีประสิทธิภาพ และยังไม่มีผู้ใดทำการเขียนโปรแกรมคอมพิวเตอร์ที่ใช้วิธีนี้ในการแก้ปัญหาโปรแกรมจำนวนเต็ม คณะผู้จัดทำจึงได้เสนอเป็นปัญหาพิเศษขึ้น ในการใช้คอมพิวเตอร์เพื่อใช้แก้ไขปัญหา

ต่าง ๆ นั้น ปัญหาเหล่านั้นมิใช่ว่ามนุษย์ไม่สามารถที่จะแก้ไขปัญหานั้น การใช้คอมพิวเตอร์นั้นเป็นการใช้แก้ไขปัญหาที่มีจำนวนมากมาย ถ้าใช้มนุษย์ทำอาจจะเกิดผิดพลาดได้ จึงใช้คอมพิวเตอร์ช่วยในการแก้ปัญหา และใช้เวลาในการแก้ไขปัญหาน้อยมาก

<u>วัตถุประสงค์</u>

1. สร้างโปรแกรมสำเร็จรูปขึ้นเพื่อใช้ในการแก้ปัญหาการโปรแกรมจำนวนเต็ม โดยใช้วิธีบรานซ์แอนด์บาวด์ เนื่องจากยังไม่มีผู้ใดทำขึ้นมา
2. สามารถใช้ในการเรียนการสอนวิชาการวิจัยการดำเนินงานได้
3. สามารถใช้ในการแก้ปัญหาจริงที่เกิดขึ้นได้

<u>วิธีการดำเนินงาน</u>

1. ศึกษาหารูปแบบของปัญหาการโปรแกรมจำนวนเต็ม
2. หารูปแบบของสมการจากปัญหาการโปรแกรมจำนวนเต็ม
3. นำสมการที่ได้ไปประมวลผลโดยไมโครคอมพิวเตอร์โดย
   - ศึกษาความเป็นไปได้ในการใช้ไมโครคอมพิวเตอร์ ในการแก้ไขปัญหาการโปรแกรมจำนวนเต็ม โดยวิธีบรานซ์แอนด์บาวด์
   - กำหนดคุณสมบัติของโปรแกรม
   - กำหนดแบบและลักษณะของโปรแกรม
   - เขียนโปรแกรมและจัดทำคู่มือ
   - ทดสอบโปรแกรมจากข้อมูลจริง

<u>ประโยชน์ที่คาดว่าจะได้รับ</u>

1. สามารถนำไปประยุกต์ใช้กับงานจริงได้
2. ทำให้การแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานซ์แอนด์บาวด์ ทำได้รวดเร็วและลดความผิดพลาดจากการคำนวณด้วยมือ
3. สามารถนำไปใช้ในการเรียนการสอนวิชาการวิจัยการดำเนินงาน

บทที่ 2

ทฤษฎีที่ใช้ในการทำปัญหาพิเศษ

2.1 โปรแกรมเชิงเส้น (Linear Programming)

โปรแกรมเชิงเส้นเป็นตัวแบบเชิงปริมาณที่เกี่ยวข้องกับการจัดทรัพยากร (Resource) ที่มีจำกัดให้กับกิจกรรมต่าง ๆ (Activity) อย่างมีประสิทธิภาพทั้งนี้ต้องให้ได้ตามวัตถุประสงค์ที่ต้องการ ลักษณะที่สำคัญของโปรแกรมเชิงเส้นคือ ฟังก์ชันวัตถุประสงค์ (Objective Function) และฟังก์ชันข้อจำกัด (Constraints Function) ต้องมีลักษณะเป็นเส้นตรง

George Stigler และ George D.Dantzig จัดเป็นผู้คิดค้นโปรแกรมเชิงเส้นเป็นกลุ่มแรก George Stigler นำโปรแกรมเชิงเส้นมาศึกษาปัญหาการลดความอ้วน โดยพยายามจะหาส่วนผสมของอาหารที่มีคุณค่าด้านลดความอ้วนมากที่สุด และมีต้นทุนต่ำสุด George D. Dantzig เป็นผู้คิดค้นวิธีการซิมเพล็กซ์ (Simplex Method) เพื่อแก้ปัญหาโปรแกรมเชิงเส้นโดยเน้นหนักทางด้านวงการทหาร เช่น กำหนดเส้นเดินทางของเครื่องบินรบและการวางแผนกำลังทางทหาร

สำหรับทางด้านธุรกิจนั้น ผู้ตัดสินใจส่วนใหญ่จะต้องวางแผนให้ได้รับความสำเร็จตรงตามวัตถุประสงค์ที่ตั้งไว้ (ส่วนใหญ่ได้แก่การทำให้เกิดกำไรสูงสุด หรือทำให้ได้ต้นทุนต่ำสุด) ภายใต้ทรัพยากรที่มีอยู่อย่างจำกัด ทรัพยากรดังกล่าวรวมถึง คน (Man) เงิน (Money) วัสดุ (Material) เครื่องจักร (Machine) ฝ่ายจัดการ (Management) และการตลาด (Marketing) จะเห็นว่าโปรแกรมเชิงเส้นย่อมเป็นวิธีการที่เหมาะสมที่สุดในการแก้ปัญหาทางด้านธุรกิจ

ตัวอย่างปัญหาโปรแกรมเชิงเส้น :

(ก) การผลิตสินค้า สามารถผลิตให้ได้คุณภาพ (grade) ต่าง ๆ กัน ขึ้นอยู่กับส่วนผสมที่ใช้ ปัญหาคือจะผลิตแต่ละ grade เป็นจำนวนเท่าไร โดยใช้ส่วนผสมอย่างไรจึงจะกำไรสูงสุด

(ข) การควบคุมสภาวะอากาศเสียซึ่งเกิดจากโรงงานอุตสาหกรรมเช่นโรงงานถลุงเหล็ก ถ้าที่มาของสารที่จะทำให้อากาศเสียคือเตาเผา 2 ชนิด และทางโรงงานต้องการนำเครื่องมือมาใช้กับเตาทั้ง 2 เพื่อลดปริมาณสารที่ทำให้อากาศเสีย ปัญหาคือทางโรงงานต้องการสิ้นเปลืองค่าใช้จ่ายน้อยที่สุดในการใช้เครื่องมือเหล่านี้ โดยที่สามารถควบคุมปริมาณสารที่ทำให้อากาศเสียให้อยู่ในอัตราที่กฎหมายกำหนด

2.2 <u>ลักษณะทั่วไปของโปรแกรมเชิงเส้น</u>

ตัวแบบทั่วไปของโปรแกรมเชิงเส้น สำหรับการหาค่าสูงสุด (maximum) ก็คือต้องการหาค่า $x_1, x_2, \ldots, x_n$ ซึ่งทำให้ได้

$$\max \; Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

โดยมีข้อจำกัดที่ว่า

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leqslant b_1 \qquad \ldots(1)$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leqslant b_2 \qquad \ldots(2)$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leqslant b_m \qquad \ldots(m)$$

$$x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0 \qquad \ldots(*)$$

เรียก $Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$ ว่า ฟังก์ชันวัตถุประสงค์

เรียก (1) ถึง (m) ว่า ฟังก์ชันข้อจำกัด

เรียก (*) ว่า Non-negativity

เรียก x_j ว่า ตัวแปรตัดสิน (Decision Variable)

เรียก a_{ij}, b_i, c_j ว่า พารามิเตอร์ของรูปแบบ ซึ่งตัวเลขที่มีค่าคงที่

รูปแบบอื่น ๆ อาจเป็น min Z หรือฟังก์ชันข้อจำกัดเป็นแบบ $\geq$ หรือ $=$ หรือไม่มีเงื่อนไข (*) แต่ไม่ว่าจะเป็นรูปแบบใดก็สามารถจัดให้เป็นรูปแบบดังกล่าวข้างต้นได้ จึงอาจเรียกรูปแบบข้างต้นนี้ว่า แบบมาตรฐานของโปรแกรมเชิงเส้น (Standard form of Linear Programming)

บริเวณในข่าย (Feasible Region) ได้แก่บริเวณที่ประกอบด้วยจุด $(x_1, x_2, \ldots, x_n)$ ที่เป็นไปตามเงื่อนไขทั้งหมด

คำตอบในข่าย (Feasible Solution) หมายถึงค่า $x_1, x_2, \ldots, x_n$ ที่สอดคล้องตามเงื่อนไขทั้งหมด

คำตอบนอกข่าย (Infeasible Solution) หมายถึงค่า $x_1, x_2, \ldots, x_n$ ที่ไม่สอดคล้องตามเงื่อนไขอันใดอันหนึ่ง

คำตอบที่เหมาะที่สุด (Optimal Solution) ได้แก่ คำตอบในข่ายที่ให้ค่าเหมาะที่สุด (สูงสุดหรือต่ำสุดแล้วแต่กรณี) ของฟังก์ชันวัตถุประสงค์

2.3 สมมติฐานของโปรแกรมเชิงเส้น

(ก) Proportionality

สำหรับกิจกรรมแต่ละอย่างจะเห็นว่าส่วนที่เปลี่ยนแปลงไปของ Z ที่เกิดจากกิจกรรม k ได้แก่ c_kx_k แต่ปริมาณของแต่ละทรัพยากร i ที่ถูกใช้ไปกับกิจกรรม k ได้แก่ $a_{ik}x_k$ กล่าวคือแต่ละค่าเป็นสัดส่วนโดยตรงกับปริมาณ x_k ของกิจกรรม k ที่ทำไปโดยไม่มีค่าใช้จ่ายเริ่มต้น (start-up charge) ของกิจกรรมแต่ละอย่าง

(ข) Additivity

สมมติฐานนี้ และสมมติฐานในข้อ (1) ตั้งขึ้นเพื่อให้ได้ฟังก์ชันวัตถุประสงค์และฟังก์ชันข้อจำกัดที่เป็นเส้นตรง กล่าวคือกิจกรรมแต่ละอย่างไม่ก้าวก่ายกัน ดังนั้นส่วนที่เปลี่ยนแปลงไปของ Z ที่เกิดจากกิจกรรมทั้งหมด จึงเป็นผลบวกของส่วนที่เปลี่ยนแปลงไปของ Z ที่เกิดจากกิจกรรมแต่ละอย่าง และปริมาณทรัพยากรที่ถูกใช้ไปทั้งหมดจะเท่ากับผลบวกของปริมาณทรัพยากรที่ถูกใช้ไปกับกิจกรรมแต่ละอย่าง

(ค) Divisibility

บางครั้งตัวแปรตัดสินจะมีความหมายจริง ๆ เฉพาะค่าที่เป็นจำนวนเต็มเท่านั้น แต่เนื่องจากคำตอบที่ได้จากการโปรแกรมเชิงเส้น บางครั้งอาจไม่มีค่าเป็นจำนวนเต็มก็ได้ จึงมีสมมติฐานนี้ซึ่งกล่าวว่า ปริมาณของกิจกรรมสามารถแบ่ง (หาร) เป็นส่วน ๆ ได้เพื่อให้ค่าที่ไม่เป็นจำนวนเต็มของตัวแปรตัดสินเป็นค่าที่ใช้ได้ ปัญหาการโปรแกรมเชิงเส้นที่ต้องการคำตอบที่เป็นจำนวนเต็มอาจอนุโลมให้ใช้การปัดเศษจากคำตอบซึ่งได้ในตอนแรกให้เป็นจำนวนเต็ม มิฉะนั้นก็ต้องใช้การแก้ปัญหาโดยวิธีของโปรแกรมจำนวนเต็ม (Integer Programming)

(ง) Certainty

สมมติฐานนี้กล่าวว่า พารามิเตอร์ของโปรแกรมเชิงเส้น อันได้แก่ a_{ij}, b_i, c_j เป็นค่าคงที่ซึ่งรู้ค่าแน่นอนในทางปฏิบัติ เนื่องจากปัญหาโปรแกรมเชิงเส้น เป็นการสร้างรูปแบบขึ้นเพื่อการดำเนินงานในอนาคต ค่าพารามิเตอร์ต่าง ๆ จึงได้จากการคาดการณ์ล่วงหน้า ไม่ใช่ค่าที่รู้แน่นอน ดังนั้นสิ่งสำคัญที่ควรทำหลังจากได้คำตอบแล้วก็คือ การวิเคราะห์ความไว (Sensitive Analysis) เพื่อดูว่าถ้าค่าของพารามิเตอร์เปลี่ยนไป คำตอบจะเปลี่ยนแปลงมากน้อยแค่ไหน

2.4 การแก้ปัญหาโปรแกรมเชิงเส้นโดยวิธีการซิมเพล็กซ์

เป็นวิธีหาคำตอบสำหรับโปรแกรมเชิงเส้นที่ใช้ทั่วไป ใช้ได้แม้ว่าปัญหาจะมีตัวแปรมากมาย

จะแสดงการใช้วิธีนี้ในการหาคำตอบจาก

$$\max Z = 3x_1 + 5x_2$$

$$\begin{aligned} \text{St.} \quad x_1 &\leqslant 4 && \ldots(1) \\ 2x_2 &\leqslant 12 && \ldots(2) \\ 3x_1 + 2x_2 &\leqslant 18 && \ldots(3) \\ x_1 \geqslant 0,\ x_2 &\geqslant 0 \end{aligned}$$

เปลี่ยนปัญหาเดิมให้อยู่ในรูปมาตรฐาน ดังนี้

$$\max Z = 3x_1 + 5x_2 + 0x_3 + 0x_4 + 0x_5$$

$$\begin{aligned} \text{St.} \quad x_1 + x_3 &= 4 \\ 2x_2 + x_4 &= 12 \\ 3x_1 + 2x_2 + x_5 &= 18 \\ x_1, x_2, x_3, x_4, x_5 &\geqslant 0 \end{aligned}$$

เรียก x_3, x_4, x_5 ที่เพิ่มขึ้นนี้ว่า ตัวแปรส่วนขาด (Slack Variable) อันได้แก่ตัวแปรซึ่งเพิ่มเข้าไปในเงื่อนไขแบบ $\leqslant$ เพื่อให้เงื่อนไขอยู่ในรูปสมการ จะเห็นว่าเราสามารถหาคำตอบสำหรับสมการทั้ง 3 ได้โดยให้ตัวแปร 2 ตัวเป็นค่าใด ๆ แล้วหาค่าตัวแปรที่เหลือ 3 ตัว วิธีการซิมเพล็กซ์จะเลือกค่าใด ๆ นั้นเป็น 0 นั่นคือให้ตัวแปร 2 ตัวในที่นี้มีค่าเป็น 0 แล้วหาค่าตัวแปรอื่น ๆ ที่เหลือ ตัวแปรที่เราเลือกค่าให้เป็น 0 เรียกว่า ตัวแปรที่ไม่เป็นมูลฐาน (Nonbasic variable) ส่วนตัวแปรที่เหลือเรียกว่าตัวแปรมูลฐาน (Basic variable) ถ้าค่าของตัวแปรมูลฐานทุกตัวไม่เป็นค่าที่น้อยกว่า 0 เรียกคำตอบที่ได้นั้นว่าคำตอบมูลฐานในข่าย (Basic feasible solution) และคำตอบเหมาะที่สุดก็จะได้จากคำตอบมูลฐานในข่าย

วิธีการซิมเพล็กซ์อย่างคร่าว ๆ ก็คือหาคำตอบมูลฐานในข่ายซึ่งทำให้ได้ค่า Z ที่ดีขึ้นเรื่อย ๆ โดยที่ในแต่ละครั้งจะเปลี่ยนตัวแปรที่ไม่เป็นมูลฐานตัวหนึ่งให้เป็นตัวแปรมูลฐาน เรียกตัวแปรมูลฐานตัวเข้า และเปลี่ยนตัวแปรมูลฐานอีกตัวหนึ่งซึ่งเรียกว่าตัวแปรมูลฐานตัวออก ให้เป็นตัวแปรที่ไม่เป็นมูลฐาน

คำตอบมูลฐานในข่ายอันหนึ่งที่ได้จากคำตอบมูลฐานอีกอันหนึ่ง โดยการเปลี่ยนตัวแปรมูลฐานกับตัวแปรที่ไม่เป็นมูลฐานเพียงครั้งเดียว เรียกว่าคำตอบมูลฐานในข่ายที่อยู่ข้างเคียงหรือติดกัน (adjacent) และคำตอบมูลฐานในข่ายจะเป็นคำตอบเหมาะที่สุดต่อเมื่อไม่มีคำตอบมูลฐานในข่ายที่อยู่ข้างเคียงซึ่งให้ค่า Z ที่ดีกว่า

เพื่อความสะดวกในการหาคำตอบด้วยมือโดยใช้วิธีการซิมเพล็กซ์ จะเขียนฟังก์ชันวัตถุประสงค์เป็น $Z - 3x_1 - 5x_2 = 0$ และเขียนเป็นตารางซิมเพล็กซ์ (Simplex Tableau) ซึ่งลงเฉพาะค่าสัมประสิทธิของตัวแปร ค่าคงที่ทางขวามือและตัวแปรมูลฐาน โดยถือว่า Z เป็นตัวแปรมูลฐานถาวร และ $Z - 3x_1 - 5x_2 = 0$ เป็นเงื่อนไขอีกอันหนึ่ง

ตัวแปรมูลฐาน	RHS	x_1	x_2	x_3	x_4	x_5
Z	0	-3	-5	0	0	0
x_3	4	1	0	1	0	0
x_4	12	0	2	0	1	0
x_5	18	3	2	0	0	1

วิธีการซิมเพล็กซ์แบ่งเป็นขั้นตอนดังนี้

<u>ขั้นที่ 1</u> : หาคำตอบมูลฐานในข่ายอันแรกโดยใช้ตัวแปรส่วนขาดเป็นตัวแปรมูลฐาน ในที่นี้คือ x_3, x_4, x_5 และโดยที่แต่ละสมการมีตัวแปรมูลฐานเพียงตัวเดียว (โดยมีสัมประสิทธิ์เป็น 1 ด้วย) ค่าคงที่ทางซ้ายมือของแต่ละสมการจึงเป็นค่าของตัวแปรมูลฐานแต่ละตัวนั้นในที่นี้จึงได้ (0, 0, 4, 12, 18) เป็นคำตอบมูลฐานในข่ายอันแรกโดยได้ Z = 0

<u>ขั้นที่ 2</u> : ดูว่าคำตอบที่ได้จากขั้นที่ 1 เป็นคำตอบเหมาะที่สุดหรือยัง โดยดูจากสัมประสิทธิ์ในสมการแรก ถ้าไม่มีค่าที่เป็นจำนวนลบอยู่เลย คำตอบที่ได้ในขั้นที่ 1 ก็จะเป็นคำตอบที่เหมาะที่สุด แต่ถ้ามีค่าที่เป็นจำนวนลบอยู่คำตอบที่ได้ก็ยังไม่ใช่คำตอบที่เหมาะที่สุด ต้องหาคำตอบใหม่โดยการเปลี่ยนตัวแปรมูลฐานกับตัวแปรที่ไม่เป็นมูลฐานตามขั้นที่ 3 ต่อไป

ในที่นี้สัมประสิทธิ์ในสมการแรกของ x_1 เป็น -3 และของ x_2 เป็น -5 . คำตอบ (0, 0, 4, 12, 18) จึงไม่ใช่คำตอบที่เหมาะที่สุด

<u>ขั้นที่ 3</u> : เปลี่ยนตัวแปรมูลฐานกับตัวแปรที่ไม่เป็นมูลฐานดังนี้

(3.1) เลือกตัวแปรมูลฐานตัวเข้าโดยดูสัมประสิทธิ์ในสมการแรก ตัวแปรที่มีสัมประสิทธิ์เป็นลบมากกว่าเพื่อน แสดงว่าเป็นตัวที่เพิ่มค่า Z ได้มากกว่าตัวอื่น จึงเลือกตัวแปรนั้นเป็นตัวแปรมูลฐานเข้า

ในที่นี้ได้ x_2 เป็นตัวแปรมูลฐานตัวเข้า เรียกแถวตั้งซึ่งอยู่ใต้สัมประสิทธิ์ของตัวแปรมูลฐานตัวเข้านี้ว่าแถวตั้งหลัก (Pivot Column)

(3.2) เลือกตัวแปรมูลฐานตัวออกโดยใช้การเปรียบเทียบอัตราส่วน (Ratio test) กล่าวคือเปรียบเทียบอัตราส่วนที่ได้จากการหารค่าคงที่ทางซ้ายมือของแต่ละสมการที่มีสัมประสิทธิ์ในแถวตั้งหลักเป็นจำนวนบวกด้วยสัมประสิทธิ์นั้น เรียกแถวนอนที่ได้อัตราส่วนน้อยที่สุดว่าแถวนอนหลัก (Pivot Row) ตัวแปรมูลฐานเดิมในแถวนี้จะเป็นตัวแปรมูลฐานตัวออก เนื่องจากเป็นตัวแปรที่จะมีค่าเป็น 0 ก่อนใคร (และจะเป็นจำนวนลบในที่สุด ซึ่งจะขัดกับเงื่อนไข non-negative) เมื่อตัวแปรมูลฐานตัวเข้ามีค่าเพิ่มขึ้น

ในที่นี้สมการที่ 3 และ 4 เท่านั้นที่มีสัมประสิทธิ์ในแถวตั้งหลักเป็นจำนวนบวก จึงได้อัตราส่วนที่นำมาเปรียบเทียบคือ 12/2 สำหรับสมการที่ 3 และ 18/2 สำหรับสมการที่ 4 และเนื่องจาก $12/2 < 18/2$ จึงได้สมการที่ 3 เป็นแถวนอนหลักและ x_4 เป็นตัวแปรมูลฐานตัวออก เรียกสัมประสิทธิ์ที่อยู่ทั้งในแถวนอนหลักและแถวตั้งหลักว่าตัวเลขหลัก (Pivot number) ในที่นี้คือสัมประสิทธิ์ 2 ในสมการที่ 3 ซึ่งวงไว้

ขั้นที่ 4 : หาคำตอบมูลฐานในข่ายอันใหม่

(4.1) หารแถวนอนหลักด้วยตัวเลขหลัก (จะได้สัมประสิทธิ์ของตัวแปรมูลฐานตัวเข้าเป็น 1)

(4.2) เปลี่ยนสมการอื่น ๆ ใหม่โดยทำสัมประสิทธิ์ของตัวแปรมูลฐานตัวเข้าในสมการนั้น ๆ ให้เป็น 0

ตารางต่อไปนี้แสดงค่าต่าง ๆ ที่ได้จากเริ่มต้นและเมื่อใช้วิธีการซิมเพล็กซ์ครั้งที่ 1 แล้ว

ครั้งที่ 0

ตัวแปรมูลฐาน	RHS	x_1	x_2 (เข้า)	x_3	x_4	x_5	อัตราส่วน
Z	0	-3	-5	0	0	0	
x_3	4	1	0	1	0	0	
ออก ← x_4	12	0	2	0	1	0	12/2 น้อยที่สุด
x_5	18	3	2	0	0	1	18/2

ครั้งที่ 1

ตัวแปรมูลฐาน	RHS	x_1	x_2	x_3	x_4	x_5	อัตราส่วน
Z	30	-3	0	0	5/2	0	
x_3	4	1	0	1	0	0	
x_2	6	0	1	0	1/2	0	
x_5	6	3	0	0	-1	1	

เนื่องจากแต่ละสมการมีตัวแปรมูลฐานเพียงตัวเดียวดังกล่าวแล้ว จึงอ่านจากตารางครั้งที่ 1 ได้ (0, 6, 4, 0, 6) เป็นคำตอบมูลฐานในข่ายอันใหม่โดยมี Z = 30 และโดยที่สัมประสิทธิ์ในสมการแรกยังมีจำนวนที่เป็นลบคำตอบที่ได้ใหม่นี้จึงยังไม่ใช่คำตอบที่เหมาะที่สุด ใช้วิธีซิมเพล็กซ์อีกครั้งจะได้ x_1 เป็นตัวแปรมูลฐานตัวเข้าและ x_5 เป็นตัวแปรมูลฐานตัวออก และได้ตารางสำหรับครั้งที่ 1 และ 2 ดังนี้

<u>ครั้งที่ 1</u>

	ตัวแปรมูลฐาน	RHS	x_1 (เข้า ↓)	x_2	x_3	x_4	x_5	อัตราส่วน	
	Z	30	-3	0	0	5/2	0		
	x_3	4	1	0	1	0	0	4/1	
	x_2	6	0	1	0	1/2	0		
ออก ←	x_5	6	3	0	0	-1	1	6/3	น้อยที่สุด

<u>ครั้งที่ 2</u>

ตัวแปรมูลฐาน	RHS	x_1	x_2	x_3	x_4	x_5	อัตราส่วน
Z	36	0	0	0	3/2	1	
x_3	2	0	0	1	1/3	-1/3	
x_2	6	0	1	0	1/2	0	
x_1	2	1	0	0	-1/3	1/3	

จากตารางสำหรับครั้งที่ 2 ได้ (2, 6, 2, 0, 0) เป็นคำตอบมูลฐานในข่ายใหม่โดยมี Z = 36 และโดยที่สัมประสิทธิ์ในสมการแรกไม่มีค่าเป็นจำนวนลบแล้ว คำตอบที่ได้นี้จึงเป็นคำตอบเหมาะที่สุด

2.5 โปรแกรมจำนวนเต็ม (Integer Programming)

การโปรแกรมจำนวนเต็ม เป็นวิธีการเชิงปริมาณเฉพาะที่นำมาใช้ช่วยแก้ปัญหาของโปรแกรมเชิงเส้น จากหัวข้อก่อน ๆ นั้นทำให้เราทราบว่าตัวแปรมูลฐานที่มีค่าดีที่สุดสำหรับปัญหาเชิงเส้นสามารถมีค่าเป็นจำนวนเต็ม (Integer) ก็ได้หรือเป็นเศษ (Fractional values) ก็ได้ แต่ในความเป็นจริงแล้ว ปัญหาบางปัญหานั้นค่าเศษเกือบจะไม่มีประโยชน์และไม่มีความหมายเลย ตัวอย่างเช่น รถยนต์ 0.75 คันหรือรองเท้า 0.5 คู่

เมื่อเกิดปัญหาดังกล่าว ผู้ทำการตัดสินใจมักจะนิยมใช้วิธีการทำให้คำตอบที่ได้มาจากโปรแกรมเชิงเส้นเป็นจำนวนเต็มโดยการปัดเศษ เพราะวิธีดังกล่าวจะประหยัดทั้งต้นทุนและเวลา แต่การกระทำดังกล่าวมักจะมีปัญหา เพราะว่าปัญหาเกือบทุกปัญหาต้องมีข้อจำกัด (Constraints) ควบคุมไว้ การทำตัวเลขให้เป็นจำนวนเต็มโดยการปัดเศษอาจจะทำให้คำตอบที่ได้นั้นไม่เป็นไปตามข้อจำกัดต่าง ๆ ที่กำกับเอาไว้ได้ นอกจากนี้ผลลัพธ์จากการปัดเศษอาจจะไม่ใช่คำตอบที่ดีที่สุดก็ได้ การโปรแกรมจำนวนเต็มจึงเป็นวิธีการเชิงปริมาณวิธีหนึ่งที่จะเข้ามาช่วยแก้ปัญหาในการหาค่าจำนวนเต็มให้กับผู้ทำการตัดสินใจได้

ปัญหาการโปรแกรมจำนวนเต็มมีอยู่ด้วยกัน 3 ประเภทใหญ่ ๆ คือ

(1) All-Integer Programming เป็นปัญหาโปรแกรมเชิงเส้นที่มีข้อจำกัดว่าตัวแปรทุกตัว ณ ผลลัพธ์ที่ดีที่สุดจะต้องเป็นจำนวนเต็ม

(2) Mixed-Integer Programming เป็นปัญหาโปรแกรมเชิงเส้นที่ต้องการเพียงแต่ว่าตัวแปรบางตัวเท่านั้นที่ต้องเป็นจำนวนเต็ม

(3) Zero-One Programming เป็นปัญหาโปรแกรมเชิงเส้นที่ตัวแปรตัดสินมีค่าได้แต่เพียง 0 หรือ 1 เท่านั้น

2.6 ตัวแบบของโปรแกรมจำนวนเต็ม

ตัวแบบของโปรแกรมจำนวนเต็มโดยทั่วไปมีลักษณะดังนี้

$$\max \quad Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

โดยมีข้อจำกัดที่ว่า

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leqslant b_1 \quad \ldots(1)$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leqslant b_2 \quad \ldots(2)$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leqslant b_m \quad \ldots(m)$$

$$x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0,\ \text{integer}$$

ทั้งนี้ตัวแปรทั้งหมดมีความหมายเช่นเดียวกับโปรแกรมเชิงเส้น ข้อสมมติต่าง ๆ ก็คล้ายคลึงกัน เพียงแต่ว่ามีข้อสมมติเพิ่มขึ้นมาอีกหนึ่งข้อ คือ ตัวแปรทั้งหมดหรือตัวแปรบางตัวต้องเป็นจำนวนเต็ม

วิธีการแก้ปัญหาโปรแกรมจำนวนเต็มนั้นมีหลายวิธี ในที่นี้จะกล่าวถึงวิธีบรานช์แอนด์บาวด์เท่านั้น

2.7 วิธีการบรานช์แอนด์บาวด์ (Branch and Bound Method)

วิธีการบรานช์แอนด์บาวด์ถูกคิดค้นขึ้นโดย A.H. Land, A.G. Doig และ J.D.C Little ทั้งนี้การแก้ปัญหาโดยวิธีนี้จะเริ่มตั้งแต่การแก้ปัญหาโดยวิธีการของโปรแกรมเชิงเส้นธรรมดาโดยไม่ต้องคำนึงถึงข้อจำกัดที่ว่าตัวแปรต้องมีลักษณะเป็นจำนวนเต็ม การ Branching จะเริ่มขึ้นเมื่อตราบใดที่ผลลัพธ์ที่ได้มาไม่เป็นไปตามข้อกำหนดที่ว่าตัวแปรผลลัพธ์ต้องมีลักษณะเป็นจำนวนเต็ม ตัวแปรใดที่ไม่เป็นจำนวนเต็มจะถูกแบ่งออกเป็น 2 ส่วนทันทีเพื่อค้นหาจำนวนเต็มให้ได้ จะเห็นว่าจะเกิดปัญหาโปรแกรมเชิงเส้น 2 ปัญหาย่อย หลังจากที่แก้ปัญหาย่อยทั้ง 2 แล้ว ผลปรากฏว่าตัวแปรที่ต้องตัดสินใจยังไม่มีค่าที่เป็นจำนวนเต็มตามที่ต้องการอีก จะต้องแยกกิ่งก้านออกไปเรื่อย ๆ จนกว่าจะได้คำตอบที่เป็นจำนวนเต็ม

การ branch นั้นจะต้องหา upper bound และ lower bound เพื่อใช้ในการพิจารณาว่าจะ branch ไปทาง upper bound หรือ lower bound โดยที่

$$U_i = (-1 + f_{ro})(Y_{ok}) / Y_{rk} \quad , Y_{rk} < 0$$

$$D_i = (f_{ro})(Y_{ok}) / Y_{rk} \quad , Y_{rk} > 0$$

- U_i คือ uppet bound
- D_i คือ lower bound
- f_{ro} คือ เศษของคำตอบ X_i แถว r
- Y_{ok} คือ สัมประสิทธิ์ของสมการ Z หลัก k
- Y_{rk} คือ สัมประสิทธิ์ของสมการ Z หลัก k แถว r

เมื่อได้ค่า U_i, D_i แล้วจะเลือกค่าที่น้อยที่สุด เพื่อพิจารณาว่าจะ branch ไปทางใด แล้วจึงหาสมการเพิ่มคือ

- เมื่อ branch ไปทาง upper bound

$$s_i = (-1 + f_{ro}) - \sum Y_{ij}X_j$$

- เมื่อ branch ไปทาง lower bound

$$s_i = (-f_{ro}) - \sum Y_{ij}X_j$$

เมื่อได้สมการเพิ่มแล้วจะใส่เข้าไปในตารางสุดท้าย แล้วใช้วิธีซิมเพล็กซ์เพื่อหาคำตอบ ตรวจดูว่าค่าคำตอบเป็นจำนวนเต็มหมดหรือยัง ถ้ายังก็จะหา upper bound และ lower bound และทำขั้นตอนซ้ำไปเรื่อย ๆ จนกว่าจะได้ค่าคำตอบที่ดีที่สุด

ตัวอย่างของการแก้ปัญหาโดยใช้วิธีบรานช์แอนด์บาวด์

$$\max \quad Z = 2X_1 + X_2$$

$$\begin{aligned} \text{St.} \quad X_1 + X_2 &\leqslant 5 \\ -X_1 + X_2 &\leqslant 0 \\ 6X_1 + 2X_2 &\leqslant 21 \\ X_i &\geqslant 0, \text{ integer} \end{aligned}$$

วิธีการแก้ปัญหาโปรแกรมจำนวนเต็มจะทำตามขั้นตอนดังนี้

แก้ปัญหาโดยไม่ต้องคำนึงถึงข้อจำกัดที่ว่าตัวแปรต้องเป็นเลขจำนวนเต็ม และใช้วิธีการโปรแกรมเชิงเส้นเข้าช่วย

หาคำตอบจากปัญหาได้โดยใช้วิธีซิมเพล็ก โดยที่ตารางสุดท้ายคือ

ตัวแปรมูลฐาน	RHS	X_1	X_2	X_3	X_4	X_5
Z	31/4	0	0	1/2	0	1/4
X_2	9/4	0	1	3/2	0	-1/4
X_4	1/2	0	0	-2	1	1/2
X_1	11/4	1	0	-1/2	0	1/4

เริ่มทำการ Branching โดยหา upper bound และ lower bound

$$U_1 = (-1 + (1/4))(1/4) \ / \ (-1/4) = 3/4$$

$$U_2 = (-1 + (1/2))(1/2) \ / \ (-2) \quad = 1/8$$

$$U_3 = (-1 + (3/4))(1/2) \ / \ (-1/2) = 1/4$$

$$D_1 = (1/4)(1/2) \,/\, (3/2) = 1/12$$

$$D_2 = (1/2)(1/4) \,/\, (1/2) = 1/4$$

$$D_3 = (3/4)(1/4) \,/\, (1/4) = 3/4$$

หาค่า U_1, D_1 ที่มีค่าน้อยที่สุด จะได้ D_1 เป็นค่าที่น้อยที่สุด เพราะฉะนั้นจะ branch X_2 ไปทาง lower bound และสมการเพิ่มคือ

$$s_1 = (-1/4) + (3/2)X_3 - (1/4)X_5$$

จะได้ตารางใหม่คือ

ตัวแปรมูลฐาน	RHS	X_1	X_2	X_3	X_4	X_5	s_1
Z	31/4	0	0	1/2	0	1/4	0
X_2	9/4	0	1	3/2	0	-1/4	0
X_4	1/2	0	0	-2	1	1/2	0
X_1	11/4	1	0	-1/2	0	1/4	0
s_1	-1/4	0	0	-3/2	0	1/4	1

เมื่อได้ตารางใหม่แล้วใช้วิธีการซิมเพล็กซ์จะได้

ตัวแปรมูลฐาน	RHS	X_1	X_2	X_3	X_4	X_5	s_1
Z	23/3	0	0	0	0	1/3	1/3
X_2	2	0	1	0	0	0	1
X_4	5/6	0	0	0	1	1/6	-4/3
X_1	17/6	1	0	0	0	1/6	-1/3
X_3	1/6	0	0	1	0	-1/6	-2/3

ดูค่าคำตอบที่ได้ว่าเป็นจำนวนเต็มหรือยัง ถ้ายังไม่เป็นจำนวนเต็มก็ต้องทำต่อไปโดยใช้วิธีเดิม

หา upper bound และ lower bound

$$U_1 = (-1 + (5/6))(1/3) / (-4/3) = 1/24$$

$$U_2 = (-1 + (5/6))(1/3) / (-1/3) = 1/6$$

$$U_3 = \min\{ (-1 + (1/6))(1/3) / (-1/6),$$
$$(-1 + (1/6))(1/3) / (-2/3)\} = 5/12$$

$$D_1 = (5/6)(1/3) / (1/6) = 5/3$$

$$D_2 = (5/6)(1/3) / (1/6) = 5/3$$

จะเห็นว่า U_1 มีค่าน้อยที่สุด แต่ X_4 เป็น slack จึงมีความสำคัญน้อยกว่าตัวแปรตัดสิน X_1 เพราะฉะนั้นจะ branch X_1 ไปทาง upper bound และสมการเพิ่มคือ

$$s_2 = (-1/6) - (1/6)X_5 + (1/3)s_1$$

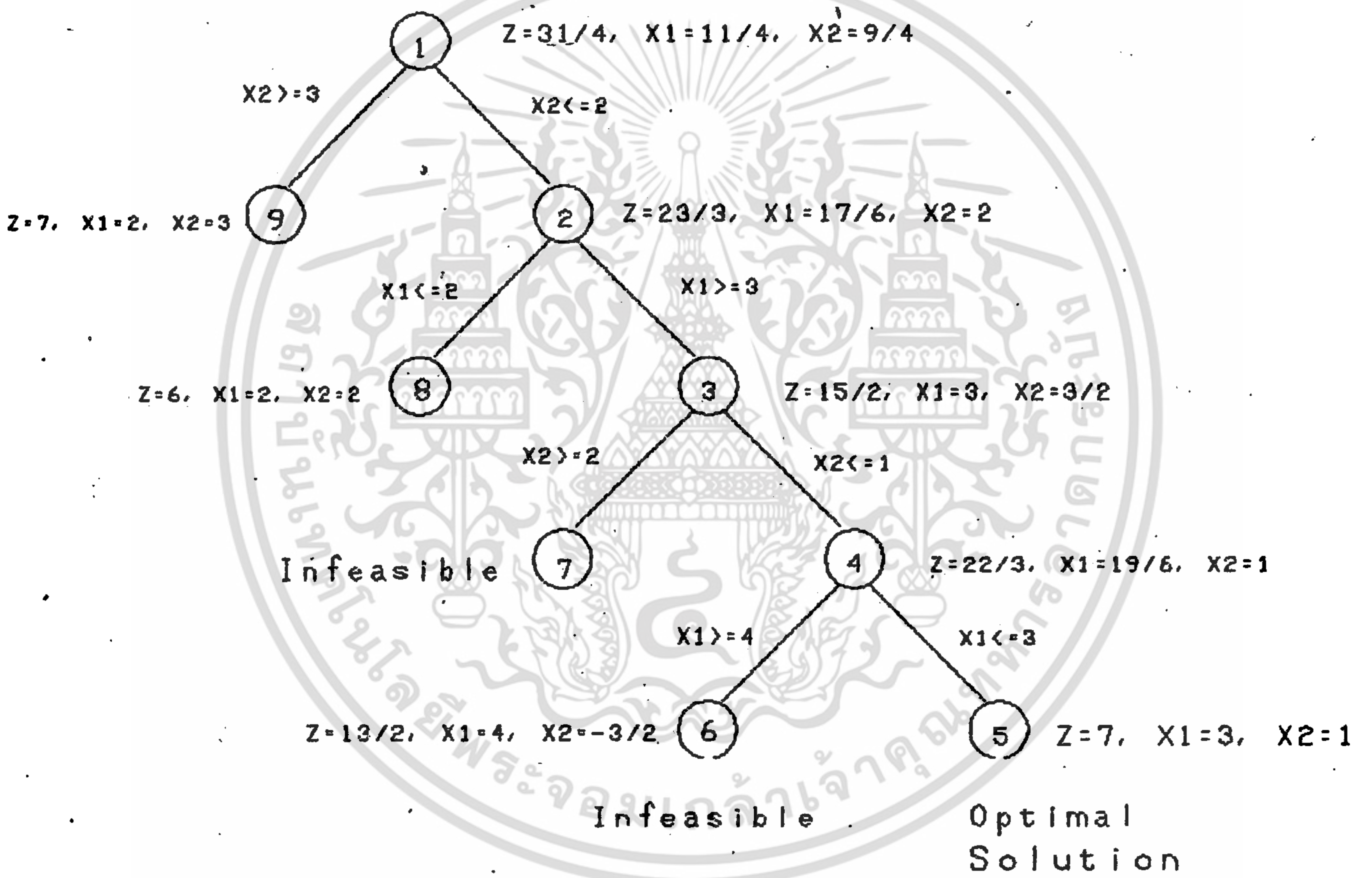
จะได้ตารางใหม่คือ

ตัวแปรมูลฐาน	RHS	X_1	X_2	X_3	X_4	X_5	s_1	s_2
Z	23/3	0	0	0	0	1/3	1/3	0
X_2	2	0	1	0	0	0	1	0
X_4	5/6	0	0	0	1	1/6	-4/3	0
X_1	17/6	1	0	0	0	1/6	-1/3	0
X_3	1/6	0	0	1	0	-1/6	-2/3	0
s_2	-1/6	0	0	0	0	-1/6	1/3	1

เมื่อใช้วิธีซิมเพล็กซ์จะได้ตารางใหม่คือ

ตัวแปรมูลฐาน	RHS	X_1	X_2	X_3	X_4	X_5	s_1	s_2
Z	15/2	0	0	0	0	1/2	0	1
X_2	3/2	0	1	0	0	1/2	0	3
X_4	3/2	0	0	0	1	-1/2	0	-4
X_1	3	1	0	0	0	0	0	-1
X_3	1/2	0	0	1	0	-1/2	0	-2
s_2	-1/2	0	0	0	0	-1/2	1	3

เมื่อได้ตารางสุดท้ายแล้ว พิจารณาดูว่าค่าคำตอบเป็นจำนวนเต็มหรือยัง ถ้ายังก็ทำวิธีการเดิมต่อไปจนกระทั่งได้ค่าคำตอบที่ดีที่สุด ในรูปที่ 1 นั้นเป็น Tree Diagram แสดงค่าคำตอบใน branch ต่าง ๆ สำหรับค่าคำตอบที่ดีที่สุดในข้อนี้คือ $Z = 7$, $X_1 = 1$, $X_2 = 1$, $X_3 = 1$



รูปที่ 1 Tree Diagram

บทที่ 3

ขั้นตอนการสร้างและการทำงานของโปรแกรม

3.1 ขั้นตอนการสร้างโปรแกรม

ในการสร้างโปรแกรมการแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์นี้ จะแบ่งออกเป็น 3 ขั้นตอนคือ

1. กำหนดคุณสมบัติของระบบงาน
2. เขียนโปรแกรม
3. เขียนคู่มือการใช้โปรแกรม

โดยที่โปรแกรมการแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์นี้ แบ่งการทำงานของโปรแกรมออกเป็น 3 ขั้นตอนคือ

1. ขั้นตอนนำข้อมูลเข้า
2. ขั้นตอนการประมวลผล
3. ขั้นตอนแสดงผลลัพธ์

3.1.1 การกำหนดคุณสมบัติของโปรแกรม

ก่อนที่จะกำหนดคุณสมบัติของโปรแกรมนี้ได้ จะต้องศึกษาถึงความเป็นไปได้ของวิธีการที่ใช้นั้นสามารถนำมาเขียนเป็นโปรแกรมคอมพิวเตอร์ได้หรือไม่ และควรใช้ภาษาคอมพิวเตอร์ภาษาอะไรในการเขียนโปรแกรม โดยที่ภาษานั้นสามารถสนับสนุนแนวความคิดที่จะใช้แก้ปัญหานี้

จากการศึกษาความเป็นไปได้ของโปรแกรมนั้นสามารถเขียนเป็นโปรแกรมคอมพิวเตอร์ได้ และภาษาที่เหมาะสมในการใช้เขียนแก้ปัญหานี้คือภาษาปาสคาล (PASCAL) เพราะว่าภาษาปาสคาลนี้มีการสนับสนุนในเรื่องเกี่ยวกับการใช้ pointer และเป็นภาษาที่เป็นที่นิยมใช้ในการเขียนโปรแกรมมาก วิธีบรานช์แอนด์บาวด์นั้นในการแก้ปัญหาจะมีลักษณะเป็น Binary Tree ถ้าใช้ภาษาปาสคาลและใช้ pointer จะทำให้เขียนโปรแกรมได้ง่ายขึ้น

คุณสมบัติของโปรแกรมนั้นจะแบ่งออกเป็น 3 ขั้นคือ

1. ขั้นตอนการนำข้อมูลเข้า การนำข้อมูลเข้าเพื่อใช้ในการคำนวณนั้นจะแบ่งออกเป็น 2 ลักษณะคือ

1. นำข้อมูลเข้าทางแป้นพิมพ์ เป็นการนำข้อมูลเข้าทางหน้าจอภาพ

2. นำข้อมูลเข้าจากแผ่นบันทึกข้อมูล (Diskette) โดยที่ข้อมูลที่จะนำเข้ามานั้นจะต้องป้อนเข้าจากแป้นพิมพ์ก่อน แล้วจึงบันทึกลงบนแผ่นบันทึกข้อมูล เมื่อต้องการนำมาใช้ก็สั่งให้ load เข้ามา

2. ขั้นตอนการประมวลผล การประมวลผลนั้นจะแบ่งออกเป็น 2 ช่วง คือในช่วงแรกจะเป็นการหาค่าโปรแกรมเชิงเส้นโดยวิธีการซิมเพล็กซ์ ในช่วงที่ 2 เป็นการทำบรานช์แอนด์บาวด์ จะใช้ข้อมูลจากวิธีการซิมเพล็กซ์

3. ขั้นตอนแสดงผลลัพธ์ ในช่วงนี้จะแสดงผลลัพธ์ที่ได้บนหน้าจอ หรือออกทางเครื่องพิมพ์

3.1.2 การเขียนโปรแกรม

โปรแกรมแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์นี้ เขียนด้วยภาษาปาสคาลซึ่งเป็นของบริษัท Borland International จำกัด มีชื่อว่า Turbo PASCAL ในการเขียนโปรแกรมนี้เป็น Version 5.0 ซึ่งเป็นรุ่นล่าสุดในขณะนี้ สำหรับเครื่องคอมพิวเตอร์ที่ใช้ในการทำงานควรมี ลักษณะดังนี้เป็นอย่างน้อย

- เป็นเครื่องคอมพิวเตอร์ IBM PC, AT, XT หรือเครื่องเลียนแบบ (Compatible)
- มีหน่วยความจำไม่ต่ำกว่า 384 กิโลไบท์
- มี Disk Drive อย่างน้อย 1 ตัว
- จอภาพโมโนโครม (อาจมีเครื่องพิมพ์ด้วย)

การเขียนโปรแกรมนั้นจะเขียนโปรแกรมในลักษณะเป็น module คือแบ่งเป็นชุดคำสั่งย่อย ๆ ทำให้สามารถตรวจสอบโปรแกรมได้ง่ายว่ามีที่ผิดที่ใดบ้าง เพราะถ้าเกิดข้อผิดพลาดก็จะทราบว่าเกิดขึ้นที่ module ใดแล้ว ทำให้การแก้ไขนั้นทำได้ง่ายขึ้น

3.1.3 การเขียนคู่มือ

ในการเขียนโปรแกรมที่ดีนั้นควรทำคู่มือการใช้โปรแกรมนั้นด้วย โดยที่คู่มือนั้นควรมีลักษณะที่อ่านเข้าใจง่าย และอธิบายขั้นตอนการใช้โปรแกรมอย่างละเอียดโดยได้จัดทำเป็นหนังสือคู่มืออีก 1 เล่ม คือคู่มือการใช้โปรแกรมการแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์

3.2 <u>การทำงานของโปรแกรม</u>

การทำงานของโปรแกรมจะแบ่งออกเป็น 3 ช่วงดังนี้

1. ช่วงการนำข้อมูลเข้าระบบ จะเป็นการทำงานเป็น 4 อย่างคือ

   1.1 ใส่ข้อมูลเข้าทางแป้นพิมพ์

   1.2 ใส่ข้อมูลเข้าทางแผ่นบันทึกข้อมูล (Load data)

   1.3 บันทึกข้อมูลลงบนแผ่นบันทึกข้อมูล (Save data)

   1.4 แสดงข้อมูลที่อยู่ในระบบ (List data)

โดยที่จะทำเป็น Menu ให้เลือกว่าจะทำอะไรก็ให้เลือกหัวข้อนั้น

2. ช่วงการประมวลผล หลังจากที่ได้รับข้อมูลแล้วจะทำการประมวลผลต่อไป โดยจะใช้วิธีการซิมเพล็กซ์หาค่าคำตอบก่อน เมื่อได้ค่าคำตอบแล้วถ้าค่าคำตอบที่ได้ไม่เป็นจำนวนเต็ม จะทำการ branch และ bound จนกว่าจะได้ค่าคำตอบที่สูงสุดหรือต่ำสุด

3. ช่วงการแสดงผลลัพธ์ หลังจากที่ได้ประมวลผลหาค่าคำตอบได้แล้ว จะแสดงผลลัพธ์ที่ได้คือ ค่าสูงสุดหรือต่ำสุด และค่าของตัวแปรตัดสิน โดยที่สามารถแสดงผลทางจอภาพและทางเครื่องพิมพ์

<u>หมายเหตุ</u> คู่มือการใช้โปรแกรมและ Listing ของโปรแกรมนี้ได้ทำแยกเป็นอีก 1 เล่ม

บทที่ 4

สรุปผลและข้อเสนอแนะ

สรุปผล

ผลจากการใช้โปรแกรมการแก้ปัญหาโปรแกรมจำนวนเต็มโดยวิธีบรานช์แอนด์บาวด์นั้น พอที่จะสรุปเป็นข้อ ๆ ได้ดังนี้

1. ทำให้การแก้ไขปัญหาโปรแกรมจำนวนเต็มนั้นทำได้รวดเร็วขึ้น
2. ผลที่ได้จากการคำนวณมีความผิดพลาดน้อย
3. สามารถนำไปประยุกต์ใช้กับงานจริงได้
4. สามารถนำไปประกอบในการศึกษาวิชาการวิจัยการดำเนินงานได้

ข้อเสนอแนะ

โปรแกรมนี้สามารถนำไปใช้กับโปรแกรมจำนวนเต็มได้ไม่ทุกปัญหา จึงอยากให้ผู้ที่มีความสามารถและมีความสนใจในปัญหานี้ สามารถนำไปพัฒนาต่อไปได้เลย ถ้ามีข้อเสนอแนะเกี่ยวกับโปรแกรมหรือมีปัญหากรุณาติดต่อกับทางภาควิชาฯ

บรรณานุกรม

1. สมภพ เจริญกุล, การตัดสินใจทางการตลาดโดยอาศัยเทคนิคเชิงปริมาณ.
กรุงเทพฯ : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2527.

2. บุญเลิศ เอี่ยมทัศนา, เรียนรู้ภาษาปาสคาล.
กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2529.

3. Hamdy A.Taha, An Introduction Operation Research.
New York : Macmillan Publishing Company, 1987.

4. Linus Schrage, Linear, Integer, and Quadratic Programming with LINDO. California : The Scientific Press, 1984.

5. Borland, TURBO PASCAL 4.0 Owner's Handbook.
California : Borland International, 1987.

6. G. Michael Schneider and Steven C. Bruell,
ADVANCED PROGRAMMING AND PROBLEM SOLVING WITH PASCAL.
Singapore : John Wiley & Sons Inc., 1987.