โปรแกรมโลจิสติกส์เบื้องต้น

อ.ดร.ภัทรพงษ์ ภาคภูมิ

15 มีนาคม พ.ศ. 2563

สารบัญ

0.1 ประมวลการสอน

ประจำภาคปลาย ปีการศึกษา 2560

1. คณะ วิศวกรรมศาสตร์ สาขาวิชา วิศวกรรมอุตสาหการ
2. รหัสวิชา 02206337 ชื่อวิชา โปรแกรมโลจิสติกส์เบื้องต้น (Introduction to Logistics Program)
3. จำนวน 3 หน่วยกิต (บรรยาย 3 ชม. ศึกษาด้วยตัวเอง 6 ชม.) นิสิตเข้าเรียน 3 ชั่วโมงต่อสัปดาห์และศึกษาด้วยตัวเองนอกห้องเรียนเป็นเวลา 6 ชั่วโมงต่อสัปดาห์
4. คำอธิบายรายวิชา
   แบบจำลองการตัดสินใจ วิธีการหาคำตอบที่ดีที่สุด วิธีฮิวริสติกส์ วิธีเมต้าฮิวริสติกส์ ขั้นตอนวิธีเชิงพันธุกรรม ปัญหาการจัดเส้นทางสำหรับยานพาหนะ ปัญหาการวางแผนจัดวางตู้คอนเทนเนอร์ภายในเรือสินค้า
   Decision model. Optimization solution method. Heuristic method. Meta-heuristic method. Genetic algorithm. Vehicle routing problem. Containership storage planning problem.
5. หัวข้อวิชา
   (a) Decision model
   (b) Linear Programming
   (c) Mixed Integer Linear Programming
   (d) Heuristic Algorithms
   (e) Metaheuristic Algorithms
   (f) Genetic Algorithm
   (g) Vehicle Routing Problems
   (h) Containership Storage Planning Problem
   (i) Other Selected Topics
6. มาตรฐานผลการเรียนรู้จากหลักสูตรสู่รายวิชา
   (a) คุณธรรม จริยธรรม
      i. มีวินัย ตรงต่อเวลา รับผิดชอบต่อตนเองและสังคม เคารพกฎระเบียบและข้อบังคับต่าง ๆ ขององค์กรและสังคม
      ประเมินจากการตรงต่อเวลาของนิสิตในการเข้าเรียน การส่งงานที่ได้รับมอบหมายตามกำหนดเวลาและการแต่งกายที่ถูกต้องตามระเบียบมหาวิทยาลัย
      สุ่มตรวจเช็คการเข้าเรียนและการแต่งกายด้วยความน่าจะเป็นหนึ่งในสอง 5%
   (b) ความรู้
      i. มีความรู้และความเข้าใจทางคณิตศาสตร์พื้นฐาน วิทยาศาสตร์พื้นฐาน วิศวกรรมพื้นฐาน และ เศรษฐศาสตร์ เพื่อการประยุกต์ใช้กับงานทางด้านวิศวกรรมศาสตร์ที่เกี่ยวข้อง และการสร้างนวัตกรรมทางเทคโนโลยี
      ประเมินผลสัมฤทธิ์ทางการเรียนและการปฏิบัติของนิสิต เช่น การทดสอบย่อย การสอบกลางภาค การสอบปลายภาค
      สอบย่อย 10% สอบกลางภาค 20% สอบปลายภาค 25%

ii. มีความรู้และความเข้าใจเกี่ยวกับหลักการที่สาคัญ ทั้งในเชิงทฤษฎีและปฏิบัติ ในเนื้อหาของสาขาวิชาเฉพาะด้านทางวิศวกรรม และทางด้านโลจิสติกส์ สามารถนาไปประยุกต์ใช้ในการวางแผนและแก้ปัญหาในกิจกรรมด้านโลจิสติกส์ได้
ประเมินจากการสอบภาคทฤษฎีและปฏิบัติในรายวิชาที่เกี่ยวข้อง
ประเมินจากรายงานของนิสิต และการนำเสนอหน้าชั้นเรียน
รายงานและการนำเสนอหน้าชั้นเรียน 5%

iii. สามารถบูรณาการความรู้ในสาขาวิชาที่ศึกษากับความรู้ในศาสตร์อื่น ๆ ที่เกี่ยวข้อง

iv. สามารถวิเคราะห์และแก้ไขปัญหา ด้วยวิธีการที่เหมาะสม รวมถึงการประยุกต์ใช้เครื่องมือที่เหมาะสม เช่น โปรแกรมคอมพิวเตอร์ เป็นต้น

v. สามารถใช้ความรู้และทักษะในสาขาวิชาของตน ในการประยุกต์แก้ไขปัญหาในงานจริงได้
เยี่ยมชมสถานประกอบการเพื่อศึกษาดูงานจริง 3%

(c) ทักษะทางปัญญา

i. มีความคิดอย่างมีวิจารณญาณที่ดี
ประเมินผลจากการแก้สถานการณ์ที่ เกิดขึ้นในระหว่างการเรียน การสอน เช่น การแก้ปัญหาโจทย์ การตอบซักถามคำถาม 5 %
ประเมินผลจากการทำรายงานและงานที่ได้รับมอบหมาย 15%

ii. สามารถรวบรวม ศึกษา วิเคราะห์ และ สรุปประเด็นปัญหาและความต้องการ

iii. สามารถคิด วิเคราะห์ และแก้ไขปัญหาด้านวิศวกรรมได้อย่างมีระบบ รวมถึงการใช้ข้อมูลประกอบการตัดสินใจในการทำงานได้อย่างมีประสิทธิภาพ

iv. สามารถสืบค้นข้อมูลและแสวงหาความรู้เพิ่มเติมได้ด้วยตนเอง เพื่อการเรียนรู้ตลอดชีวิต และทันต่อการเปลี่ยนแปลงทางองค์ความรู้และเทคโนโลยีใหม่ๆ

(d) ทักษะความสัมพันธ์ระหว่างบุคคลและความรับผิดชอบ

i. สามารถสื่อสารกับกลุ่มคนที่หลากหลาย และสามารถสนทนาทั้งภาษาไทยและภาษาต่างประเทศได้อย่างมี ประสิทธิภาพ สามารถใช้ความรู้ในสาขาวิชาชีพมาสื่อสารต่อสังคมได้ในประเด็นที่เหมาะสม
การแสดงความคิดเห็นจากการถาม ตอบในชั้นเรียน 2%

ii. สามารถเป็นผู้ริเริ่มแสดงประเด็นในการแก้ไขสถานการณ์เชิงสร้างสรรค์ทั้งส่วนตัวและส่วนรวม พร้อมทั้ง แสดงจุดยืนอย่างพอเหมาะทั้งของตนเองและของกลุ่ม รวมทั้งให้ความช่วยเหลือและอำนวยความสะดวก ในการแก้ไขปัญหาสถานการณ์ต่าง ๆ

iii. สามารถวางแผนและรับผิดชอบในการพัฒนาการเรียนรู้ทั้งของตนเอง และสอดคล้องกับทางวิชาชีพอย่างต่อเนื่อง

iv. รู้จักบทบาท หน้าที่ และมีความรับผิดชอบในการทำงานตามที่มอบหมาย ทั้งงานบุคคลและงานกลุ่ม สามารถปรับตัวและทำงานร่วมกับผู้อื่นทั้งในฐานะผู้นำและผู้ตามได้อย่างมีประสิทธิภาพ สามารถวางตัวได้อย่างเหมาะสมกับความรับผิดชอบ

(e) ทักษะการวิเคราะห์เชิงตัวเลข การสื่อสาร และการใช้เทคโนโลยีสารสนเทศ

i. มีทักษะในการใช้คอมพิวเตอร์ สำหรับการทำงานที่เกี่ยวข้องกับวิชาชีพได้เป็นอย่างดี
ความถูกต้องของคำตอบที่เกิดจากการ ใช้คอมพิวเตอร์ในการประมวลผล
สอบใช้คอมพิวเตอร์ 10%

ii. สามารถประยุกต์ใช้เทคโนโลยีสารสนเทศและการสื่อสาร ที่ทันสมัยได้อย่างเหมาะสมและมีประสิทธิภาพ

7. วิธีการสอน
   การบรรยาย อภิปราย ศึกษาค้นคว้าด้วยตนเอง การทำการบ้าน การทำงานกลุ่ม

8. อุปกรณ์สื่อการสอน Slides, whiteboard, computer

9. การวัดผลสัมฤทธิ์ในการเรียน

   (a) classroom and homework 35 %

      i. CC การเข้าเรียน 5 %
         ส่งเป็น selfie ให้รูปและเวลาชัดเจนและ identifiable กลับด้านให้อ่านได้ การส่งรูปผิดเป็นเจตนาทำทุจริตและส่งผลให้เกรดลดลง 1 ประจุในแต่ละครั้งที่ทำ

      ii. IN classroom individual assignment 15 %

      iii. GW classroom group work 15 %
         นิสิตที่ลาเรียนให้ส่งไฟล์แจ้งการลาเรียน โดยต้องแจ้งล่วงหน้าก่อนเวลา 3 วันยกเว้นเหตุสุดวิสัย คะแนน CC วันนั้นจะไม่นำมาคิด
         ส่วนงาน IN และ GW ต้องทำงานหลังจากลาภายใน 7 วัน โดยหาโจทย์ที่ใกล้เคียงกันมา present ให้อ.ในชั่วโมง office hour ที่นัดหมายไว้

   (b) exams and quizzes 65 %

      i. HQ สอบย่อย 10%

      ii. CQ สอบโดยใช้คอมพิวเตอร์ 10 %

      iii. MID สอบกลางภาค 20%

      iv. FIN การสอบไล่ 25 % (cumulative) **นิสิตที่ลาสอบ ต้องแจ้งล่วงหน้าก่อนเวลา 7 วันยกเว้นเหตุสุดวิสัย ต้องตามสอบภายใน 7 วัน โดยโจทย์ใหม่จะมีความยากขึ้นมาก**

10. การประเมินผลการเรียน อิงกลุ่มและอิงเกณฑ์ดังนี้

    (a) A 80+

    (b) B 70+

    (c) C 60+

    (d) D 50+

11. เอกสารอ่านประกอบ
    (Required) Introduction to Operations Research, Frederick S. Hillier, Gerald J. Lieberman, (10th Ed) 2015

12. ชั่วโมงเรียน จันทร์และพุธ เวลา 9:00-10:30 ห้อง 9202

13. **แลปคอมพิวเตอร์ พฤหัสบดี เวลา 16:00-18:00 ห้อง 8404**
    แลปคอมพิวเตอร์หลังสอบกลางภาค พุธ เวลา 9:00-10:30 ห้อง 8302

14. office hours จันทร์และพุธ เวลา 8-9 และ พุธ เวลา 15 -16 (by appointment only นัดเวลาเข้าพบหลังเรียนเท่านั้น) **คำถามหรือข้อสงสัยต่างๆ ที่เกี่ยวกับการเรียนการสอนให้ถามโดยตรงกับอ.ผู้สอนระหว่างเรียน หลังเรียนหรือที่ภาควิชาเท่านั้น**

15. ตารางเรียนโดยประมาณ

    (a) Decision model
    สัปดาห์ที่ 1-2

    (b) Linear Programming
    สัปดาห์ที่ 3

    (c) Mixed Integer Linear Programming
    สัปดาห์ที่ 4-7 (สอบกลางภาคสัปดาห์ถัดไป)

    (d) Heuristic Algorithms
    สัปดาห์ที่ 8-9

    (e) Metaheuristic Algorithms
    สัปดาห์ที่ 10

    (f) Genetic Algorithm
    สัปดาห์ที่ 11

    (g) Vehicle Routing Problems
    สัปดาห์ที่ 12

    (h) Containership Storage Planning Problem
    สัปดาห์ที่ 13

    (i) Other Selected Topics
    สัปดาห์ที่ 14-15

บทที่ 1

บทนำ

สำหรับปัญหาโลจิสติกส์ การหาคำตอบเชิงปริมาณด้วยหลักการทางคณิตศาสตร์ที่เป็นคำตอบที่ดีที่สุดหรือคำตอบที่เหมาะสมในระยะเวลาที่เหมาะสม เริ่มได้ด้วยการสร้างแบบจำลองทางคณิตศาสตร์ เพื่อจำลองสถานการณ์ ซึ่งสามารถแบ่งได้เป็นหลายแบบ ตามแต่ลักษณะของปัญหาและวัตถุประสงค์ของงานเช่น

1. linear model
2. mixed integer linear model
3. non-linear model
4. nonlinear mixed integer model

รูปที่ 1.1: motivation สำหรับปัญหาการวางแผนจัดวางตู้คอนเทนเนอร์ภายในเรือสินค้า

วิธีการหาคำตอบที่ดีที่สุดของแบบจำลองทางคณิตศาสตร์มีหลายวิธีเช่น

1. linear model
    (a) graphical method
    (b) simplex method
    (c) interior point method
2. integer linear model และ mixed integer linear model

   (a) branch-and-bound method
   (b) cutting plane algorithm
3. non-linear model
   (a) gradient descent algorithm
4. nonlinear mixed integer model
   (a)

วิธีการหาคำตอบด้วยวิธีดังต่อไปนี้ไม่สามารถแสดงได้ว่าคำตอบที่ได้เป็นวิธีการหาคำตอบที่ดีที่สุด แต่มีความนิยมเพราะในหลายกรณีสามารถหาคำตอบที่ดีได้อย่างรวดเร็ว

1. heuristic เรียกวิธีการหาคำตอบที่ใช้เฉพาะปัญหา เช่นวิธีการ greedy หรือ nearest neighbor สำหรับปัญหา การเดินทางของเซลแมน วิธีการ Clarke and Wright saving algorithm สำหรับปัญหา vehicle routing algorithm
2. meta-heuristic เรียกวิธีการหาคำตอบที่ใช้กับหลากหลายปัญหา โดยมีหลักการสำหรับการประยุกต์ใช้ ในหลายครั้งหลักการเหล่านี้เป็นการลองเลียนหลักธรรมชาติ เช่น genetic algorithm, similated annealing, และ ant colony optimization

โดยมีวิธี hybrid คือการผสมผสานของสองวิธีใหญ่ๆข้างต้น

ใบบทต่อไปนี้นิสิตจะได้สร้างแบบจำลองจากสถานการณ์จริงหรือสถานการณ์จำลอง ศึกษากระบวนการหาคำตอบกับตัวอย่างปัญหาขนาดย่อมๆ และเขียนโปรแกรมคอมพิวเตอร์เพื่อหาคำตอบของปัญหาทั้งขนาดเล็กและขนาดใหญ่

บทที่ 2

แบบจำลองการตัดสินใจ Decision model

ใบบทนี้นิสิตจะได้สร้างแบบจำลองจากสถานการณ์จริงหรือสถานการณ์จำลอง

เราจะใช้แบบจำลองการตัดสินใจควบคู่ไปกับการเขียนโปรแกรม GAMS (General Algebraic Modelling Systems) [4] ซึ่งมีลักษณะตามตัวอย่าง 2.0.1 โดยในขั้นต้นนี้จะเป็นโมเดลที่ไม่ซับซ้อน ตัวแปร ค่าพารามิเตอร์ สมการข้อจำกัด และอื่นๆ จะเขียนออกมาตรงไปตรงมาในขั้นตอนเดียว ซึ่งจะทำได้กับโมเดลที่มีขนาดเล็กและไม่ซับซ้อนเช่นในตัวอย่างดังต่อไปนี้

ตัวอย่าง 2.0.1. ตัวอย่าง GAMS code สำหรับปัญหาโจนส์ชาวไร่ Jones the Farmer.gms 2.1.2

รูปที่ 2.1: โค๊ดโปรแกรม GAMS ของปัญหาชาวไร่โจนส์ Jones the Farmer

```
positive variable x1 acres of corn
                  x2 acres of wheat;

variable z objective function;

equations  Income total income
           land limitation of land
           Mincorn lower limit of corn
           Labor limitation of labor;

Income..   30*x1 + 100*x2   =e=  z;
Land..        x1 +      x2  =l=  7;
Mincorn.. 10*x1             =g= 30;
Labor..     4*x1 +  10*x2   =l= 40;

model jones /all/;
solve jones using lp max z;
```

1. ให้ตัวแปรที่เป็นบวก(และศูนย์) เป็น positive variable เช่น x_1 และ x_2 จากนั้นตามด้วยความหมายของตัวแปรนั้นๆ (optional)

2. จบคำสั่ง(แต่งตั้งตัวแปร)ด้วย semicolon

```
positive variable x1 acres of corn
                  x2 acres of wheat;
variable z objective function;
```

3. ค่าสมการเป้าหมาย objective function z ต้องกำหนดให้เป็นตัวแปรฟรีเสมอ (free variable) ซึ่งเขียนสั้นๆได้ว่า variable

4. ถ้าเราแยกให้คำสั่งแต่งตั้งตัวแปร x_1 และ x_2 จะได้ดังนี้ (ต้องมี semicolon ทั้งสองแห่ง)

```
positive variable x1 acres of corn;
positive variable x2 acres of wheat;
```

5. การแต่งตั้งข้อจำกัด constraints ก็ทำได้ในลักษณะคล้ายกับตัวแปร

```
equations  Income total income
           land limitation of land
           Mincorn lower limit of corn
           Labor limitation of labor;
```

6. จากนั้นต้องเขียนบอกสมการ-อสมการของข้อจำกัดแต่ละตัวที่จะใช้ โดยจะต้องมีไข่ปลาสองฟองเสมอ และ $=e=$ แทนเครื่องหมาย $=$, $=l=$ แทนเครื่องหมาย $\leq$, และ $=g=$ แทนเครื่องหมาย $\geq$ ควรเว้นวรรคตามความเหมาะสม การจัดรูปที่ดีแบบจะช่วยให้อ่านได้ง่ายขึ้น

```
Income..   30*x1 + 100*x2   =e=  z;
Land..        x1 +      x2   =l=  7;
Mincorn.. 10*x1              =g= 30;
Labor..     4*x1 +  10*x2   =l= 40;
```

7. เราให้ชื่อโมเดลเป็นชื่อได้ก็ได้ที่ไม่ซ้ำกับคำที่ GAMS สงวนไว้ - reserved (คำที่สงวนไว้เช่น positive, variable, equation, model, etc.) และตามด้วยชื่อของข้อจำกัดทั้งหมดที่เราต้องการในโมเดลข้างใน slash ถ้าเราเอาทุกข้อจำกัดที่มีทั้งหมดตั้งแต่ต้นเราสามารถใช้คำว่า all แทนได้

```
model jones /all/;
```

8. solve statement แสดงถึงรูปแบบโมเดลที่เหมาะสม(ในที่นี้คือกำหนดการเชิงเส้น linear program:lp) และทิศทาง min หรือ max ของปัญหา (sense) ตามรูปแบบดังนี้

```
solve jones using lp max z;
```

```
solve jones max z using lp;
```

9. หรือดังนี้

ในตัวอย่างถัดไป การเขียนโมเดลใน GAMS จะแยกค่าพารามิเตอร์และ index ออกจากตัวโมเดลหลักเพื่อ ง่ายต่อการปรับค่าต่างๆนั้นในภายหลัง หรือเมื่อโมเดลมีความซับซ้อนขึ้น

ตัวอย่าง 2.0.2. ตัวอย่าง GAMS code สำหรับปัญหา transport.gms [4]

1. Sets เป็นการให้ชื่อของเซ็ต และให้สมาชิก ในตัวอย่างนี้คือเซ็ต i และเซ็ต j โดยเซ็ต i มีสมาชิกสองตัวคือ seattle กับ san-diego สังเกตว่าชื่อของสมาชิกแต่ละตัวจะต้องเขียนติดเป็นคำเดียวกัน

```
Sets
     i   'canning plants'   / seattle, san-diego /
     j   'markets'          / new-york, chicago, topeka / ;
```

2. การกำหนดพารามิเตอร์และให้ค่าของพารามิเตอร์ในหนึ่งมิติ

```
Parameters

     a(i)   'capacity of plant i in cases'
       /     seattle      350
             san-diego    600   /

     b(j)   'demand at market j in cases'
       /     new-york     325
             chicago      300
             topeka       275   / ;
```

3. เราสามารถใช้ table ในการกำหนดพารามิเตอร์และให้ค่าของพารามิเตอร์ในสองมิติ

```
Table d(i,j)   'distance in thousands of miles'
                      new-york       chicago       topeka
    seattle             2.5            1.7           1.8
    san-diego           2.5            1.8           1.4   ;
```

4. scalar ใช้เพื่อกำหนดพารามิเตอร์ศูนย์มิติ เราสามารถให้ค่าพารามิเตอร์ในรูปของสมการได้

5. การกำหนดตัวแปร และการกำหนดตัวแปรที่มีค่าบวกหรือศูนย์

```
Scalar f  'freight in dollars per case per thousand miles'  /90/ ;

Parameter c(i,j)  'transport cost in thousands of dollars per case' ;

          c(i,j) = f * d(i,j) / 1000 ;

Variables
     x(i,j)  'shipment quantities in cases'
     z       'total transportation costs in thousands of dollars' ;

Positive Variable x ;
```

รูปที่ 2.2: ส่วนแรกของโค้ดโปรแกรม GAMS ของปัญหา transport.gms

```
Sets
     i   'canning plants'   / seattle, san-diego /
     j   'markets'          / new-york, chicago, topeka / ;

Parameters
     a(i)  'capacity of plant i in cases'
       /    seattle     350
            san-diego   600  /
     b(j)  'demand at market j in cases'
       /    new-york    325
            chicago     300
            topeka      275  / ;

Table d(i,j)  'distance in thousands of miles'
                   new-york       chicago      topeka
    seattle           2.5           1.7          1.8
    san-diego         2.5           1.8          1.4  ;

Scalar f  'freight in dollars per case per thousand miles'  /90/ ;

Parameter c(i,j)  'transport cost in thousands of dollars per case' ;
          c(i,j) = f * d(i,j) / 1000 ;
```

รูปที่ 2.3: ส่วนที่สองของโค้ดโปรแกรม GAMS ของปัญหา transport.gms

```
Variables
     x(i,j)  'shipment quantities in cases'
     z       'total transportation costs in thousands of dollars' ;

Positive Variable x ;

Equations
     cost         'define objective function'
     supply(i)    'observe supply limit at plant i'
     demand(j)    'satisfy demand at market j' ;

cost ..        z  =e=  sum((i,j), c(i,j)*x(i,j)) ;
supply(i) ..   sum(j, x(i,j))  =l=  a(i) ;
demand(j) ..   sum(i, x(i,j))  =g=  b(j) ;

Model transport /all/ ;
Solve transport using lp minimizing z ;

Display x.l, x.m ;
```

2.1 linear model

แบบจำลองเชิงเส้นมีคุณสมบัติ 4 ประการ [1]

แบบฝึกหัด 2.1.1. [14] บริษัท Giapetto’s Woodcarving ผลิตของเล่นไม้สองอย่างคือหุ่นทหารและรถไฟ หุ่นทหารขายในราคา 27 ดอลลาร์ ต่อชิ้นและใช้วัตถุดิบ 10 ดอลลาร์ การผลิตหุ่นแต่ละตัวเพิ่มต้นทุนแปรผันและ overhead 14 ดอลลาร์ รถไฟขายในราคา 21 ดอลลาร์ต่อชิ้นและใช้วัตถุดิบ 9 ดอลลาร์ การรถไฟแต่ละอันเพิ่มต้นทุนแปรผันและ overhead 10 ดอลลาร์ การผลิตหุ่นทหารและรถไฟใช้แรงงานสองประเภทคือช่างไม้และช่างตกแต่ง หุ่นทหารแต่ละตัวใช้การตกแต่ง 2

ชั่วโมง และแรงงานช่างไม้ 1 ชั่วโมง รถไฟแต่ละตัวใช้การตกแต่ง 1 ชั่วโมง และแรงงานช่างไม้ 1 ชั่วโมง ในแต่ละสัปดาห์ บริษัท Giapetto's Woodcarving สามารถหาวัตถุดิบได้อย่างเพียงพอ แต่มีแรงงานตกแต่ง 100 ชั่วโมงและแรงงานช่างไม้ 80 ชั่วโมง รถไฟมี demand ไม่จำกัดแต่หุ่นทหารขายได้ไม่เกิน 40 ตัวต่อสัปดาห์ บริษัท Giapetto's Woodcarving ต้องการได้กำไรต่อสัปดาห์ที่สูงที่สุด (รายได้-ต้นทุน) จงเขียนโมเดลทางคณิตศาสตร์เป็นกำหนดการเชิงเส้นจำลองสถานการณ์ที่ บริษัท Giapetto's Woodcarving จะใช้หากำไรที่สูงสุด

วิธีทำ

เราสามารถกำหนดให้ x_1 แทนจำนวนการผลิตหุ่นทหาร x_2 แทนจำนวนการผลิตหุ่นรถไฟ ข้อมูลรวมกันทั้งหมดจะได้แบบจำลองทางคณิตศาสตร์ ดังนี้

$$
\begin{array}{llrll}
max & z = & 3x_1 + 2x_2 & & \text{(ต้องการกำไรสูงที่สุด)} \quad (2.1a)\\
s.t. & & 2x_1 + x_2 \leq & 100 & \text{(งานตกแต่ง)} \quad (2.1b)\\
 & & x_1 + x_2 \leq & 80 & \text{(งานไม้)} \quad (2.1c)\\
 & & x_1 \leq & 40 & \text{(demand ของหุ่นทหาร)} \quad (2.1d)\\
 & & x_1, x_2 \geq & 0 &
\end{array}
$$

รูปที่ 2.4: โค๊ดโปรแกรม GAMS ของปัญหา giapetto.gms

```
giapetto.gms
positive variable x1 soldier model
                  x2 train model;
variable profit objective function;
equation maxprofit maximize total profit
         finishing finishing work
         wood woodworking work
         soldierDem demand of soldier model;
maxprofit..  3*x1 + 2*x2 =e= profit;
finishing..  2*x1 + x2   =l= 100;
wood..       x1   + x2   =l= 80;
soldierDem.. x1          =l= 40;
model giapetto /all/;
solve giapetto using lp max profit;
```

■

แบบฝึกหัด 2.1.2. [14] โจนาชาวไร่ต้องการหาปริมาณพื้นที่ที่เหมาะสม ที่จะปลูกข้าวโพดและธัญพืชสำหรับปีการเพาะปลูกนี้ พื้นที่ปลูกธัญพืช 1 เอเคอร์ ให้ผลผลิต 25 บุชเชลและใช้แรงงาน 10 ชั่วโมงต่อสัปดาห์ พื้นที่ปลูกข้าวโพด 1 เอเคอร์ ให้ผลผลิต 10 บุชเชลและใช้แรงงาน 4 ชั่วโมงต่อสัปดาห์ ธัญพืช 1 บุชเชลขายได้ในราคา 4 ดอลลาร์ ข้าวโพด 1 บุชเชลขายได้ในราคา 3 ดอลลาร์ พื้นที่เพาะปลูกมีอยู่ทั้งสิ้นอยู่ 7 เอเคอร์และมีแรงงานอยู่ 40 ชั่วโมงต่อสัปดาห์ รัฐบาลกำหนดให้ชาวสวนแต่ละรายต้องมีผลผลิตข้าวโพดอย่างน้อย 30 บุชเชลในฤดูกาลนี้ ให้ x_1 เป็นปริมาณพื้นที่เพาะปลูกของข้าวโพด และ x_2 เป็นปริมาณพื้นที่เพาะปลูกของธัญพืช ใช้ตัวแปรตัดสินใจนี้ในการสร้างกำนหดการเชิงเส้นที่จะหา คำตอบให้กับโจนาเพื่อที่จะหารายได้ที่สูงที่สุดจากข้าวโพดและธัญพืชนี้

วิธีทำ เราสามารถนำข้อมูลข้างต้นมาเขียนลงมนตารางได้ดังนี้

จากตาราง 2.1 เราจะได้กำหนดการเชิงเส้นดังนี้

ตารางที่ 2.1: ข้อมูลปัญหาโจนาชาวไร่

หน่วย	ปริมาณปลูก (เอเคอร์)	ผลผลิต (บุชเชล)	แรงงาน (ชั่วโมง)	รายได้ต่อบุชเชอร์ (ดอลลาร์)
ข้าวโพด	x_1	$10x_1$	$4x_1$	3
ธัญพืช	x_2	$25x_2$	$10x_2$	4
ข้อจำกัด/ความต้องการ	≤ 7	$10x_1 \geq 30$	≤ 40	max

$$max \quad z = cx \tag{2.3a}$$

$$s.t. \quad Ax \leq b \tag{2.3b}$$

$$x \geq \quad 0, \text{ and } x_i \text{ integer for some i} \tag{2.3c}$$

รูปที่ 2.5: แบบจำลองกำหนดการเชิงเส้นผสมจำนวนเต็ม

$$max \quad z = \quad 30x_1 + 100x_2 \quad \text{(ต้องการรายได้สูงที่สุด)} \tag{2.2a}$$

$$s.t. \quad x_1 + x_2 \leq \quad 7 \quad \text{(ข้อจำกัดทางพื้นที่)} \tag{2.2b}$$

$$10x_1 \geq \quad 30 \quad \text{(ผลผลิตข้าวโพดขั้นต่ำ)} \tag{2.2c}$$

$$4x_1 + 10x_2 \leq \quad 40 \quad \text{(ข้อจำกัดทางแรงงาน)} \tag{2.2d}$$

$$x_1, x_2 \geq \quad 0 \tag{2.2e}$$

■

2.2 ปัญหากำหนดการเชิงเส้นผสมจำนวนเต็ม

ปัญหากำหนดการเชิงเส้นผสมจำนวนเต็ม (mixed integer linear model) สามารถจัดให้อยู่ในรูปแบบโดยทั่วไปได้ดังรูปที่ 2.3 ต่อไปนี้

ตัวแปรตัดสินใจที่เป็นจำนวนเต็มได้แก่ปริมาณพนักงาน จำนวนชิ้นงานที่ต้องผลิต (ในบางกรณีจำนวนนี้ให้เป็นจำนวน-จริงได้)

2.2.1 job scheduling problem (JSP) [10] [3]

การจัดตารางเวลางานเป็นกระบวนการที่มนุษย์ทำในทุกๆวัน การวางแผนงานในชีวิตแต่ละวันมักจะทำโดยใช้ประสบการณ์ในการวางแผน ในเชิงทฤษฎีทางวิทยาศาสตร์ การจัดตารางเวลางานทำโดยการแบ่งงานออกเป็นชิ้นๆ โดยต้องใช้ต้นทุนในการทำงานแต่ละชิ้นในช่วงเวลาต่างๆ การจัดตารางเวลางานเพื่อเพิ่มประสิทธิภาพจึงเป็นสิ่งสำคัญต่อธุรกิจ เช่นในกระบวนการผลิต หรือในระบบการบริการ

การจัดตารางเวลางานที่เรียกว่า job shop scheduling คือการหาลำดับของกระบวนการสำหรับงาน n งาน บนเครื่อง-จักร m เครื่อง โดยมีเป้าหมายตามสมการวัตถุประสงค์ เช่นเพื่อให้การผลิตมีต้นทุนต่ำสุด ตัวอย่างของการจัดตารางเวลางาน

เช่น การวางแผนการผลิตและควบคุม การวางแผนและการจัดตารางเวลางานในห่วงโซ่อุปทาน และการจัดตารางเวลางานของรถไฟรางเดี่ยวเป็นต้น

โครงสร้างของปัญหา [8] ในที่นี้จะศึกษาปัญหาที่มีโครงสร้างในลักษณะดังต่อไปนี้
ให้เซ็ตของงานเป็น $J = \{1, 2, \ldots, n\}$ เซ็ตของเครื่องจักร $M = \{1, 2, \ldots, m\}$ แต่ละงานมีกระบวนการซึ่งต้องทำบนเครื่องจักรตามที่กำหนดมาแล้ว เช่นกระบวนการที่ i ของงาน j แทนโดย o_{ij} ทำบนเครื่องจักร $u_{ij} \in M$ ลำดับขั้นตอนกระบวนการของแต่ละงานเป็นไปตามที่ได้ถูกกำหนดไว้แล้ว โดยมีวัตถุประสงค์เพื่อจัดตารางเวลางาน n งานลงบนเครื่องจักร m เครื่อง และในที่นี้จะมีข้อพิจารณาดังต่อไปนี้

- กระบวนการที่ i ของงาน j ใช้เวลา $p_{ij} > 0$ หน่วย
- ทำแต่ละกระบวนการเพียง 1 ครั้งเท่านั้น
- ไม่มีการหยุดพักในหระหว่างดำเนินการกระบวนการ
- กระบวนการสองอันใดๆไม่สามารถทำซ้อนกันได้
- เวลาสำหรับการ setup ไม่ขึ้นอยู่กับเครื่องจักรหรือลำดับของงาน
- เครื่องจักรทุกเครื่องสามารถทำงานได้ตลอดเวลา

ให้นิสิตศึกษาตังอย่างจากข้อมูลดังต่อไปนี้ตามรูปที่ 2.6 ในขอบเขตสมมติฐานที่ให้

รูปที่ 2.6: ข้อมูลของปัญหา ft06 [10]

	o_{1j}		o_{2j}		o_{3j}		o_{4j}		o_{5j}		o_{6j}				
Job j	μ_{1j}	p_{1j}	μ_{2j}	p_{2j}	μ_{3j}	p_{3j}	μ_{4j}	p_{4j}	μ_{5j}	p_{5j}	μ_{6j}	p_{6j}	w_j	r_j	d_j
1	3	1	1	3	2	6	4	7	6	3	5	6	4	0	33
2	2	8	3	5	5	10	6	10	1	10	4	4	2	0	61
3	3	5	4	4	6	8	1	9	2	1	5	7	2	0	44
4	2	5	1	5	3	5	4	3	5	8	6	9	2	0	45
5	3	9	2	3	5	5	6	4	1	3	4	1	2	0	32
6	2	3	4	3	6	9	1	10	5	4	3	1	1	0	39

แต่ละงาน job j มีเวลาที่ต้องส่งงาน due date $d_j \geq 0$ มีน้ำหนักของงานเป็น w_j เพื่อบ่งบอกความสำคัญสัมพัทธ์ มีเวลาที่เริ่มต้นงานได้ relase date $r_j \geq 0$ ให้สมการวัตถุประสงค์เป็นผลรวมถ่วงน้ำหนักของงานที่ล่าช้า tardiness t_j = max $\{0, c_j - d_j\}$ เมื่อ c_j คือเวลาที่งาน j เสร็จ ปัญหานี้เรียกว่า Job Shop Scheduling Problem with Total Weighted Tardiness Objective (JSPTWT)

แบบจำลองทางคณิตศาสตร์สามารถเขียนได้ดังนี้

$$
\begin{aligned}
min \quad & \sum_{j=1}^{n} w_j t_j && && (2.4a)\\
s.t. \quad & s_{ij} + p_{ij} \le s_{i+1,j} && \forall j \in J, \forall i \in M \setminus \{m\} && (2.4b)\\
& s_{ij} + p_{ij} \le s_{kl} \vee s_{kl} + p_{kl} \le s_{ij} && \forall j, l \in J, \forall i, k \in M \setminus \{m\} && (2.4c)\\
& && \text{with } j \ne l, \text{ and } u_{ij} = u_{kl} && \\
& t_j \ge s_{mj} + p_{mj} - d_j && \forall j \in J && (2.4d)\\
& t_j \ge 0 && \forall j \in J && (2.4e)\\
& s_{1j} \ge r_j && \forall j \in J && (2.4f)
\end{aligned}
$$

แบบฝึกหัด 2.2.1. ให้ความหมายของสมการ (2.4b) (2.4c) (2.4d) และ (2.4e)

แบบฝึกหัด 2.2.2. เขียน (2.4c) ใหม่ในรูปแบบ linear constraints (hint: apply the M-method)

แบบฝึกหัด 2.2.3. เขียน GAMS code ของ job scheduling problem (2.4)

การหา feasible solution สามารถหาได้โดยกฎ FCFS ดังวิธีตามรูปที่ 2.7

รูปที่ 2.7: การหา feasible solution โดยกฎ FCFS

A feasible schedule is derived with the help of the First Come First Served rule (FCFS). The priority rule sorts the competing operations based on their arrival times: the earliest operation is processed first. If there is more than one operation with the same arrival time, then the operation with the lower job index j is prioritized. In this way, operation orders, and thus, job orders on the machines are determined as shown in Tab. 2.4.

คำตอบที่ดีที่สุดของปัญหา (2.4) สามารถหาได้โดยการใช้ซอฟแวร์ GAMS และ optimization software อื่นๆ คำตอบที่ดีได้เป็นไปตามรูปที่ 2.10 ดังต่อไปนี้

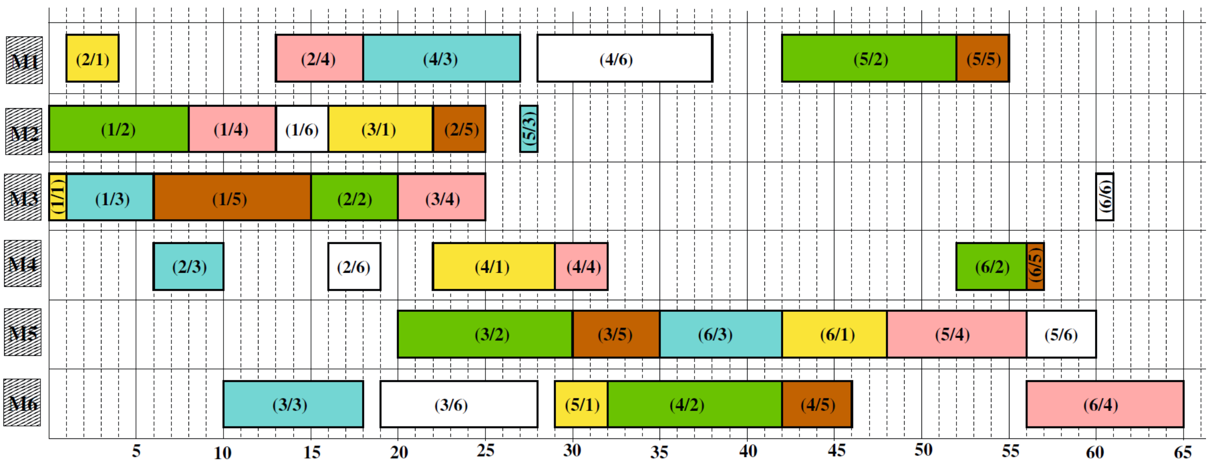
คำตอบที่ดีที่สุดของปัญหา (2.4) ตามสมการวัตถุประสงค์อื่นๆเช่น minimum makespan เป็นไปตามรูปที่ 2.11 และตามวัตถุประสงค์ FCFS เป็นไปตามรูปที่ 2.9 ดังต่อไปนี้

แบบฝึกหัด 2.2.4. อภิปรายคำตอบตามรูปทั้งสาม JSPTWT02 JSPTWT03 JSPTWT04 ของปัญหา ft06 (2.4) ในกลุ่ม

รูปที่ 2.8: feasible solution โดยกฎ FCFS

Mach.	Job order
1	$1 \to 4 \to 3 \to 6 \to 2 \to 5$
2	$2 \to 4 \to 6 \to 1 \to 5 \to 3$
3	$1 \to 3 \to 5 \to 2 \to 4 \to 6$
4	$3 \to 6 \to 1 \to 4 \to 2 \to 5$
5	$2 \to 5 \to 3 \to 1 \to 4 \to 6$
6	$3 \to 6 \to 1 \to 2 \to 5 \to 4$

รูปที่ 2.9: คำตอบของปัญหา ft06 (2.4) ตามตถุประสงค์ FCFS [10]

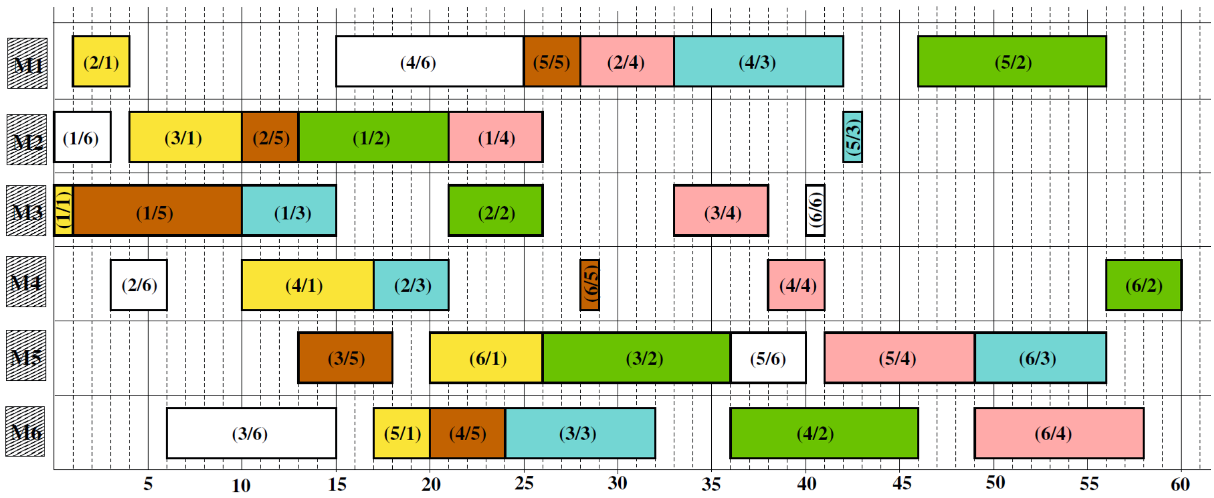


2.2.2 knapsack problem

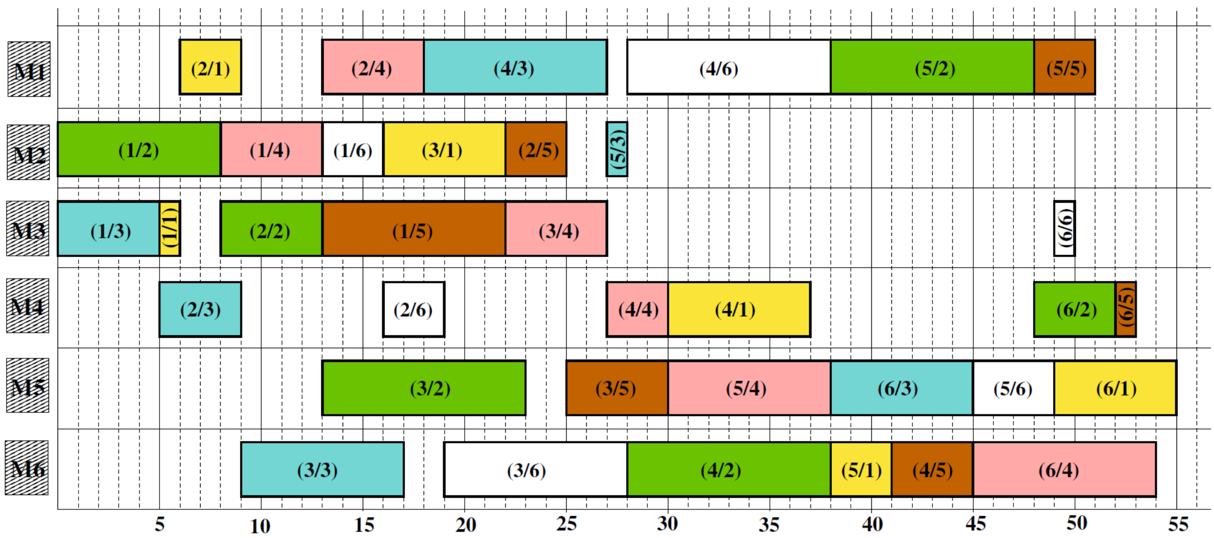
ปัญหานี้ได้แรงบันดาลใจมาจากการเลือกของใส่กระเป๋าเดินทางก่อนเดินทางข้ามทะเลทราย โดยมีข้อจำกัดคือน้ำหนักบรรทุกและให้ได้มูลค่าโดยรวมสูงสุด ให้น้ำหนักของ i เป็น a_i และมูลค่าเป็น c_i น้ำหนักสูงสุดที่รับได้เป็น b จะได้แบบจำลองทางคณิตศาสตร์ดังรูปที่ 2.12

แบบฝึกหัด 2.2.5. แก้ปัญหา linear relaxation ของปัญหา knapsack ในรูปที่ 2.13

รูปที่ 2.10: คำตอบที่ดีที่สุดของปัญหา ft06 (2.4) ตามสมการวัตถุประสงค์ TWT [10]



รูปที่ 2.11: คำตอบที่ดีที่สุดของปัญหา ft06 (2.4) ตามสมการวัตถุประสงค์ minimum makespan [10]



ตัวอย่าง 2.2.1. ในการเตรียมตัวสำหรับการเดินป่า Josie Camper เลือกของระหว่างสี่ชิ้น โดยมีน้ำหนักและ benefit ตามตารางที่ 2.2 ดังต่อไปนี้

ตารางที่ 2.2: weights and benefits of items for Josie Camper knapsack

item	น้ำหนัก (lbs)	benefit
1	5	16
2	7	22
3	4	12
4	3	8

ถ้ากระเป๋า knapsack ของ Josie Camper รองรับได้ทั้งหมด 14 lbs จงแก้ปัญหานี้ เพื่อให้ได้ benefit สูงที่สุด

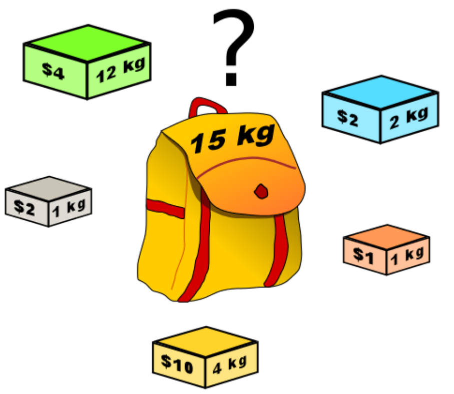
วิธีทำ ■

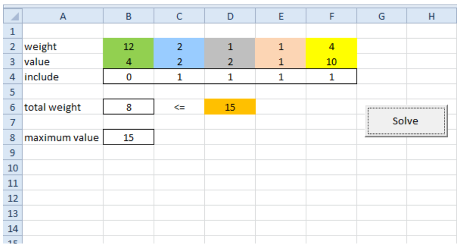
แบบฝึกหัด 2.2.6. NASA พิจารณาเลือกวัตถุสามชิ้นเพื่อนำไปในอวกาศพร้อมกับยาน โดยมีน้ำหนักและ benefit ตามตาราง

$$max \quad z = cx \tag{2.5a}$$
$$s.t. \quad ax \leq b \tag{2.5b}$$
$$x_j \in \{0, 1\} \tag{2.5c}$$

รูปที่ 2.12: แบบจำลองทางคณิตศาสตร์ของปัญหา knapsack

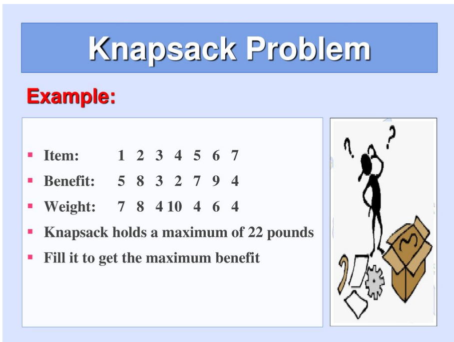
รูปที่ 2.13: ตัวอย่างปัญหา knapsack 1: motivation และการแก้ด้วย Excel solver





	A	B	C	D	E	F	G	H
1								
2	weight	12	2	1	1	4		
3	value	4	2	2	1	10		
4	include	0	1	1	1	1		
5								
6	total weight	8	<=	15			Solve	
7								
8	maximum value	15						
9								
10								
11								
12								
13								
14								

รูปที่ 2.14: ตัวอย่างปัญหา knapsack 2



$$max \quad 5x_1 + 8x_2 + 3x_3 + 2x_4 + 7x_5 + 9x_6 + 4x_7 \tag{2.6a}$$
$$s.t. \quad 7x_1 + 8x_2 + 4x_3 + 10x_4 + 4x_5 + 6x_6 + 4x_7 \leq 22 \tag{2.6b}$$
$$x_j \in \{0, 1\} \tag{2.6c}$$

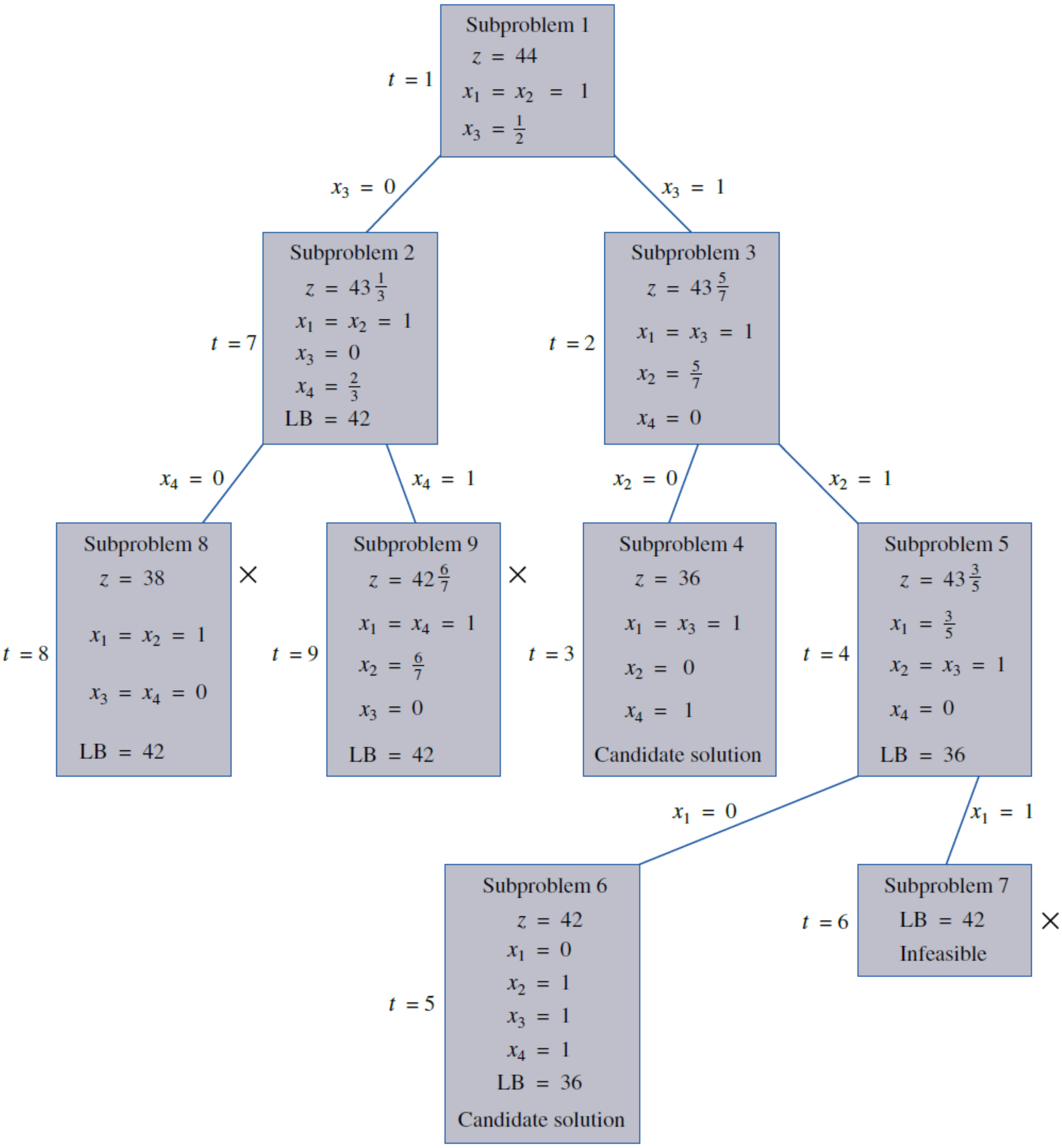
รูปที่ 2.15: แบบจำลองทางคณิตศาสตร์ของปัญหา knapsack ในรูปที่ 2.13

ที่ 2.3 ดังต่อไปนี้

ถ้าสามารถเลือกวัตถุได้หลายชิ้น และ space shuttle รองรับได้ทั้งหมด 26 lbs จงแก้ปัญหานี้ เพื่อให้ได้ benefit สูงที่สุด

แบบฝึกหัด 2.2.7. ในการย้ายหอพักเข้ามหาลัย นิสิตใหม่ต้องเลือกขนย้ายเฟอร์นิเจอร์ระหว่างห้าชิ้น โดยมีปริมาตรและมูลค่า ตามตารางที่ 2.4 ดังต่อไปนี้

รูปที่ 2.16: branch-and-bound method สำหรับปัญหาตามตารางที่ 2.2



ตารางที่ 2.3: weights and benefits of items for NASA space flight

item	น้ำหนัก (lbs)	benefit
1	10	3
2	15	4
3	17	5

ถ้ารถขนของสามารถรองรับได้ทั้งหมด 1100 cubic ft นิสิตใหม่ควรขนย้ายสิ่งใดบ้าง

แบบฝึกหัด 2.2.8. ในการลงทุนโครงการระหว่าง 4 โครงการ โดยมี NPV และมูลค่า cash outflow ที่เวลาที่ 0 ในหน่วยล้านดอลลาร์ ตามตารางที่ 2.5 ดังต่อไปนี้

ตารางที่ 2.4: ปริมาตรและมูลค่าของเฟอร์นิเจอร์แต่ละชิ้น

item	มูลค่า ($)	ปริมาตร (cubic ft.)
เตียงนอน	60	800
โต๊ะกินข้าว	48	600
เครื่องเสียง	14	300
โซฟา	31	400
ทีวี	10	200

ตารางที่ 2.5: มูลค่า cash outflow และ NPV ของโครงการลงทุน

โครงการ	cash outflow at time 0 ($)	NPV ($)
1	3	5
2	5	8
3	2	3
4	4	7

ถ้ามีเงินลงทุนทั้งหมด 6 ล้านดอลลาร์ที่เวลาที่ 0 หาการลงทุนเพื่อให้ได้ NPV สูงที่สุด

2.2.3 Lot-Sizing problem

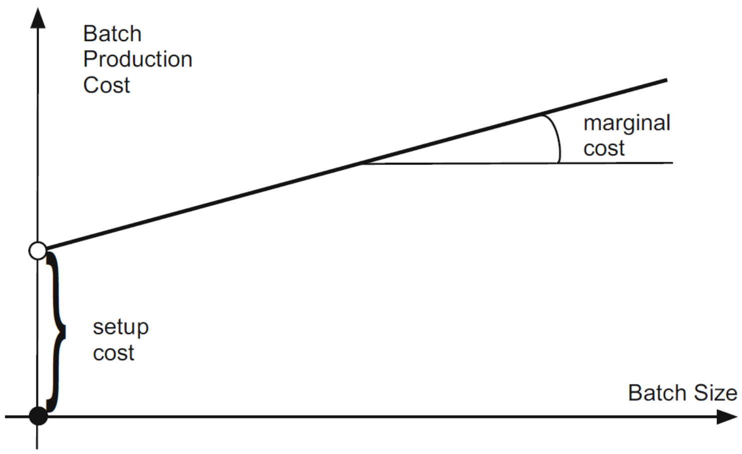
[13] ในปัจจุบันบริษัทขนาดกลางและขนาดใหญ่หลายแห่งได้ใช้คอมพิวเตอร์ในการวางแผนการผลิต เพื่อการติดตามการจัดซื้อและติดตามข้อมูลการผลิตของแต่ละผลิตภัณฑ์ในเวลาจริง และเพื่อการวางแผนการกระจายสินค้า ซึ่งสิ่งเหล่านี้มีความสำคัญยิ่ง แต่กระนั้นผลลัพธ์ที่สุดยอดกว่าสามารถได้จากการนำเครื่องมือเหล่านี้เพื่อการสร้างระบบที่หาการวางแผนที่ให้ผลประโยชน์ที่ดีที่สุด ตัวอย่างเช่นบริษัท Kellogg ได้พัฒนาระบบเพื่อการวางแผนการผลิตและการกระจายสินค้าที่ให้ผลประโยชน์ที่ดีที่สุด สำหรับ cereal และธุรกิจอาหารสะดวกบริโภค การวางแผนระบบนี้เป็นการวางแผนการจัดการในระยะสั้น เพื่อหาการไหลของผลิตภัณฑ์ที่ดีที่สุดและเพื่อการวางแผนระยะกลางในการบริหารต้นทุน การใช้ระบบนี้ทำให้บริษัท Kellogg ลดต้นทุนการจัดการระบบปีละ 4 ล้านดอลลาร์ และลดงบประมาณจากการรวบรวมระบบเข้าศูนย์กลางได้ปีละ 40 ล้านดอลลาร์

ระบบนี้ของบริษัท Kellogg ใช้ชื่อว่า Kellogg Planning System (KPS) ได้ใช้โปรแกรมคณิตศาสต์ที่เป็นระบบเชิงเส้นเท่านั้น ไม่ได้คำนึงถึงเวลาการเตรียมการผลิตและการห่อกล่องผลิตภัณฑ์ หรือ set-up times ในการพัฒนาระบบในขั้นที่สูงขึ้นใช้ระบบสมการเชิงเส้นผสมจำนวนเต็ม หรือ mixed integer linear program เพื่อพิจารณา set-up times รวมเข้ากับระบบ สิ่งที่ได้จะเป็นระบบขนาดใหญ่ที่มีความซับซ้อนและยากในการหาคำตอบที่ดีที่สุด หรือแม้กระทั่งคำตอบที่ใกล้เคียงกับคำตอบที่ดีที่สุด การปรับสูตรสมการใหม่อาจมีผลต่อการหาคำตอบที่รวดเร็วขึ้น

ตัวอย่างปัญหาการผลิตเบื้องต้น ในการผลิตจักรยานแข่งคุณภาพสูงต้องใช้วัตถุดิบและเครื่องจักรพิเศษ เนื่องการ demand ที่เป็นไปตามฤดูกาล economy of scale และค่า set-up cost ที่สูง โรงงานจะผลิตอย่างมากเดือนละ 1 batch เท่านั้น

ให้ปัญหาตัวอย่างนี้มี set-up cost 5000 ยูโร และต้นทุนต่อหน่วยคันละ 100 ยูโร ผลิต 1 คันจึงมีต้นทุนรวม 5100 ยูโร และ 10 คันมีต้นทุนเป็น 6000 ยูโร ตามลักษณะของรูปที่ 2.17

รูปที่ 2.17: ลักษณะของ fixed-charge problem



การพยากรณ์ปริมาณความต้องการของผลิตภัณฑ์ในปีนี้เป็นไปตามตารางรูปที่ 2.18

รูปที่ 2.18: พยากรณ์ปริมาณความต้องการของผลิตภัณฑ์

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
400	400	800	800	1200	1200	1200	1200	800	800	400	400

โดยให้สต๊อกเริ่มต้นเป็นจักรยาน 200 คันของต้นปีปัจจุบัน ค่าเก็บจักรยานข้ามเดือนเฉลี่ยคันละ 5 ยูโรซึ่งรวมถึง ค่าอาคารและค่าเก็บรักษา พื้นที่จัดเก็บมีเพียงพอกับจำนวน การวางแผนการผลิตต้องให้เป็นไปตามความต้องการผลิตภัณฑ์ของลูกค้า โดยจะวางแผนถึงเดือนสิงหาคม

บางคำตอบของปัญหา เนื่องจาก economy of scale ถ้าโรงงานจะเลือกผลิตทั้งหมดในเดือนแรก (สำหรับปริมาณความต้องการผลิตภัณฑ์จนถึงเดือนสิงหาคมคือ 7200 คัน) จะต้องผลิต 7000 คันในเดือนแรก ให้ผลลัพธ์เป็นไปตามรูปที่ 2.19

รูปที่ 2.19: คำตอบของปัญหาที่เลือกผลิตครั้งเดียวในเดือนแรก

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Total
Demand	400	400	800	800	1,200	1,200	1,200	1,200	7,200
Production	7,000	0	0	0	0	0	0	0	7,000
Unit cost	700,000	0	0	0	0	0	0	0	700,000
Set-up cost	5,000	0	0	0	0	0	0	0	5,000
End inventory	6,800	6,400	5,600	4,800	3,600	2,400	1,200	0	
Inv. cost	34,000	32,000	28,000	24,000	18,000	12,000	6,000	0	154,000

ต้นทุนการผลิตที่เลือกตามแบบรูปที่ 2.19 คือเลือกต้นทุนการผลิตที่ต่ำที่สุด โดยจะมีต้นทุนรวมเป็น 859,000 ยูโร

สำหรับการผลิตที่ให้ต้นทุนการเก็บรักษาต่ำสุดคือเลือกที่จะผลิตทุกเดือนตามความต้องการในแต่ละเดือน จะได้คำตอบตามรูปที่ 2.20 โดยจะมีต้นทุนรวมเป็น 740,000 ยูโร ค่าเก็บรักษาจะคิดจากค่าเฉลี่ยของสต๊อคสองเดือนติดกัน ส่งผลสุดท้ายเป็นค่าเก็บ 50% ในสต๊อคเริ่มต้นและเดือนสุดท้าย บวก 100% ในสต๊อคของเดือนที่เหลือ

รูปที่ 2.20: คำตอบของปัญหาที่เลือกผลิตทุกเดือน

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Total
Demand	400	400	800	800	1,200	1,200	1,200	1,200	7,200
Production	200	400	800	800	1,200	1,200	1,200	1,200	7,000
Unit cost	20,000	40,000	80,000	80,000	120,000	120,000	120,000	120,000	700,000
Set-up cost	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	40,000
End Invent.	0	0	0	0	0	0	0	0	
Invent. cost	0	0	0	0	0	0	0	0	0

คำตอบ 740,000 ยูโรลดต่ำลงจาก 859,000 ยูโรอย่างมาก แต่จะมีคำตอบที่ต่ำกว่านี้หรือไม่สามารถ หาได้จากการลองผิดลองถูก แต่คำตอบที่ดีที่สุดที่มีจะเป็นคำตอบที่ดีที่สุดของปัญหาหรือไม่จะต้องใช้วิธีการตรวจสอบที่มีระบบกว่าการลองผิดลองถูกนี้ วิธีนั้นคือการสร้างแบบจำลองทางคณิตศาสตร์ ซึ่งจะได้ตามรูป 2.21 ดังต่อไปนี้

การสร้างแบบจำลองทางคณิตศาสตร์นี้ทำให้ได้แบบจำลองทางคณิตศาสตร์เชิงเส้นผสมจำนวนเต็ม ซึ่งสามารถใช้กระบวนการแก้ปัญหานี้ได้โดยวิธี branch-and-bound และ/หรือ branch-and-cut ซึ่งจะได้กล่าวถึงในบทถัดๆไป

$$mincost := \sum_{t=1}^{NT}(px_t + qy_t) + \sum_{t=1}^{NT-1} hs_t + \frac{h}{2}s_{NT} \tag{2.7a}$$

$$s.t. \quad s_{t-1} + x_t = d_t + s_t \text{ for all } t = 1, \ldots, NT \tag{2.7b}$$

$$s_0 = s_{ini}, s_{NT} = 0$$

$$x_t \leq (\sum_{k=1}^{NT} d_k)y_t \text{ for all } t = 1, \ldots, NT \tag{2.7c}$$

$$x_t, s_t \geq 0, y_t \in \{0, 1\} \text{ for all } t = 1, \ldots, NT \tag{2.7d}$$

รูปที่ 2.21: แบบจำลองทางคณิตศาสตร์ของปัญหาเบื้องต้น

รูปที่ 2.22: คำตอบที่ดีที่สุดของปัญหา

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Total
Demand	400	400	800	800	1,200	1,200	1,200	1,200	7,200
Production	600	0	1,600	0	1,200	1,200	1,200	1,200	7,000
unit cost	60,000	0	160,000	0	120,000	120,000	120,000	120,000	700,000
set-up cost	5,000	0	5,000	0	5,000	5,000	5,000	5,000	30,000
End Inventory	400	0	800	0	0	0	0	0	
Inv. cost	2,000	0	4,000	0	0	0	0	0	6,000

กระบวนการนี้จะสามารถให้คำตอบที่ดีที่สุด ซึ่งเป็นไปตามรูป 2.22 ดังต่อไปนี้

คำตอบที่ดีที่สุดนี้ให้ค่าต้นทุนเป็น 736,000 ยูโร ซึ่งน้อยกว่าคำตอบเดิมที่มีคือ 740,000 ยูโร ไม่มากนัก

ปัญหา Multi-item Capacitated Lot-Sizing (MICLS)

ปัญหาการผลิต Multi-item Capacitated Lot-Sizing มี setup cost เป็น fixed-charge problem ผลิตภัณฑ์ NI ชนิด ใน NT ช่วงเวลา โดยมีต้นทุนการผลิต, holding cost, และ setup cost

กำหนดให้ indexes เป็นดังนี้

$i = 1, \ldots, NI$ เป็น products ทั้งหมด

$t = 1, \ldots, NT$ เป็นช่วงเวลาทั้งหมด

โดยมีตัวแปรตัดสินใจเป็น

x_t^i คือปริมาณการผลิตของ product i ในช่วงเวลา t

s_t^i คือปริมาณ invetory ของ product i ในท้ายช่วงเวลา t

y_t^i คือตัวแปรตัดสินใจผลิตหรือไม่ผลิต product i ในช่วงเวลา t

และพารามิเตอร์เป็น

p_t^i คือต้นทุนการผลิตต่อหน่วยของ product i ในช่วงเวลา t

h_t^i คือต้นทุน holding cost ต่อหน่วยของ product i ในท้ายช่วงเวลา t

q_t^i คือ setup cost of product i ในช่วงเวลา t

d_t^i คือ demand ของ product i ในช่วงเวลา t

จะได้โมเดลทางคณิตศาสตร์เป็นดังนี้

$$min \quad \sum_i \sum_t (p_t^i x_t^i + q_t^i y_t^i + h_t^i s_t^i) \tag{2.8a}$$

$$s.t. \quad s_{t-1}^i + x_t^i = d_t^i + s_t^i \quad \forall i, t \tag{2.8b}$$

$$x_t^i \leq M_t^i y_t^i \quad \forall i, t \tag{2.8c}$$

$$\sum_i \alpha^{ik} x_t^i + \sum_i \beta^{ik} y_t^i \leq L_t^k \quad \forall t, k \tag{2.8d}$$

$$x_t^i \geq 0, s_t^i \geq 0, y_t^i \in \{0, 1\} \tag{2.8e}$$

แบบฝึกหัด 2.2.9. เขียน GAMS code ของ Uncapacitated Lot-Sizing problem

2.2.4 network flows

ในทุกวันของชีวิต เครือข่าย networks มีอยู่ทุกที่ทุกหนแห่งเช่น สายไฟส่งไฟฟ้า เครือข่ายโทรศัพท์ส่งสัญญาณ ระบบทางหลวงเชื่อมโยงการจราจร สายรถไฟที่มีจุดเชื่อมต่อตามชุมสาย การบริการทางการบิน การผลิตและเครือข่ายการกระจายสินค้า ในตัวอย่างทั้งหมดนี้มีวัตถุประสงค์เพื่อการเคลื่อนย้ายวัตถุ (ตัวตน entity) จากจุดหนึ่งไปยังอีกจุดหนึ่งในรูปแบบที่มีประสิทธิภาพสูงสุด เนื้อหาในบทนี้จะเน้นการสร้างแบบจำลองทางคณิตศาสตร์และเรียนรู้วิธีการหลากหลาย เพื่อหาคำตอบของแบบจำลองนั้น ในที่นี่ปัญหาที่ต้องการคำตอบคือสามปัญหาพื้นฐานดังต่อไปนี้

1. ปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด Minimum cost flow problem (MCFP) ปัญหานี้เป็นปัญหาพื้นฐานที่สุดของปัญหาเครือข่าย คือการหาเส้นทางการส่งของผ่านเครือข่ายให้ได้ตามปริมาณที่ต้องการ demand ในแต่ละ node โดยให้มีต้นทุนต่ำสุด ตัวอย่างของปัญหานี้คือการกระจายสินค้าจากโรงงานไปยังโกดังหรือจากโกดังไปยังผู้ขายปลีก การส่งผ่านวัตถุดิบหรือชิ้นส่วนในระหว่างการผลิตไปตามสถานีการผลิตต่างๆในระบบการผลิต การเดินทางของรถยนต์ไปตามระบบการขนส่งในเมือง การส่งผ่านสัญญาณโทรศัพท์ไปตามสถานีเชื่อมต่อสัญญาณ

   สัญกรณ์สำหรับปัณหานี้ให้ $G = (N, A)$ เป็นเครือข่ายที่มีทิศทาง N มี n nodes และ A มี m arcs ทางเชื่อมต่อที่มีทิศทาง แต่ละ arc $(i, j) \in A$ มีต้นทุน c_{ij} การขนส่งต่อหน่วย และ capacity u_{ij} ขอบเขตบนในการส่งผ่าน และ l_{ij} ขอบเขตล่างในการส่งผ่าน ให้ $b(i)$ แทน demand และ supply ของแต่ละ node ใน node i ที่ $b(i) > 0$ จะเป็น supply node ส่วน node i ที่ $b(i) < 0$ จะเป็น demand node ส่วน node i ที่ $b(i) = 0$ จะเป็น transshipment node ให้ x_{ij} แทนปริมาณการส่งผ่านบน arc $(i, j) \in A$ จะได้แบบจำลองทางคณิตศาสตร์ดังต่อไปนี้

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.9a}$$

$$\text{subject to} \quad \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = b(i) \tag{2.9b}$$

$$l_{ij} \le x_{ij} \le u_{ij} \quad \text{for all } (i,j) \in A \tag{2.9c}$$

รูปที่ 2.23: แบบจำลองทางคณิตศาสตร์ของปัญหา MCFP

   โดยที่ $\sum_{i=1}^{n} b(i) = 0$

   แบบฝึกหัด 2.2.10. จงให้เหตุผลของข้อบังคับ $\sum_{i=1}^{n} b(i) = 0$

   ในรูปของเมทริกซ์ (2.9) จะเขียนได้ดังนี้

$$\min \quad cx \tag{2.10a}$$

$$\text{subject to} \quad \mathcal{N}x = b, \tag{2.10b}$$

$$l \le x \le u \tag{2.10c}$$

รูปที่ 2.24: แบบจำลองทางคณิตศาสตร์ของปัญหา MCFP ในรูปแบบเมทริกซ์

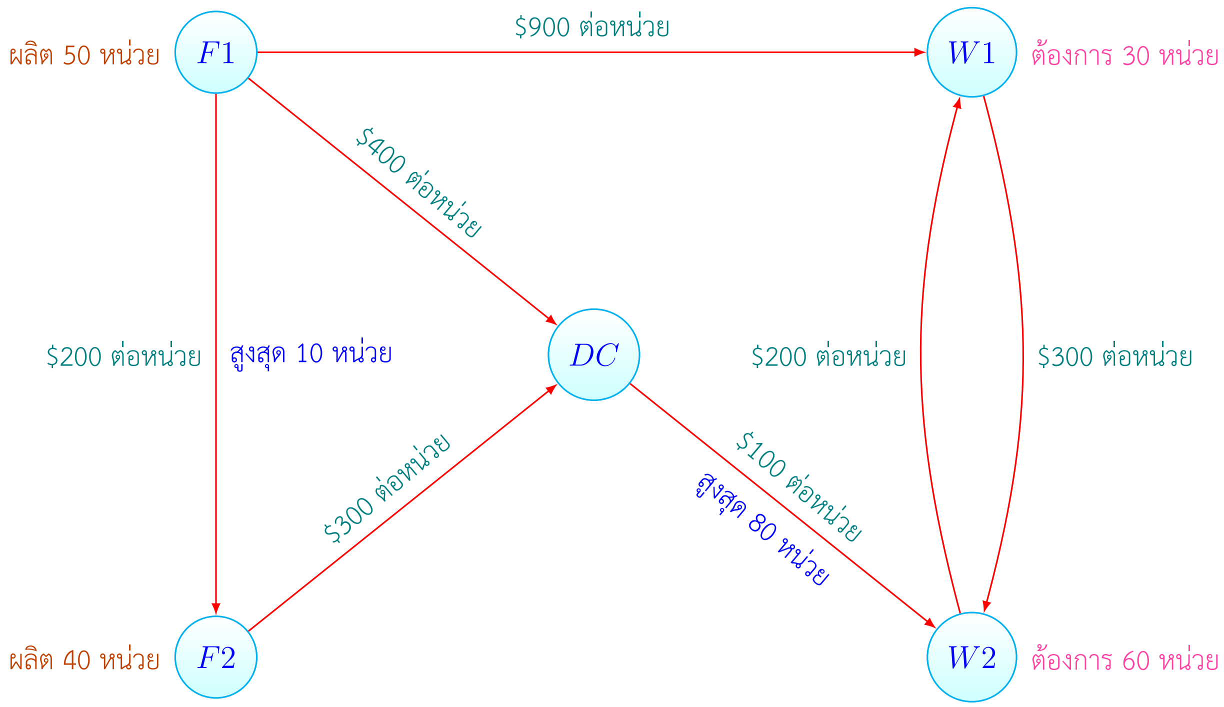
ตัวอย่าง 2.2.2. ปัญหา transport ตามตัวอย่างที่ 2.0.2 เป็นปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด Minimum cost flow problem

ตัวอย่าง 2.2.3. พิจารณาปัญหาการขนส่งและการกระจายสินค้าดังต่อไปนี้ [7]

บริษัท กระจายสินค้า จำกัด (บกจ) จะผลิตผลิตภัณฑ์ใหม่ที่โรงงานสองแห่งและส่งผลิตภัณฑ์ไปยังคลังเก็บรักษา (warehouse) สองแห่ง โครงข่ายสำหรับการขนส่งเป็นไปตามภาพที่ 2.25 โดย F1 และ F2 คือโรงงานทั้งสองแห่ง W1 และ W2 คือคลังเก็บรักษาทั้งสองแห่ง ส่วน DC คือศูนย์กระจายสินค้า ปริมาณผลิตภัณฑ์ที่จะขนส่งจาก โรงงาน F1 และ F2 สองแห่งแสดงทางซ้ายมือและ ปริมาณผลิตภัณฑ์ที่จะจัดเก็บที่คลังเก็บรักษา W1 และ W2 แสดงทางขวามือ ลูกศรแสดงเส้นทางส่งผ่านที่เป็นไปได้ ดังนั้นจึงส่งผ่านจาก F1 ไปยัง W1 ได้โดยตรง ส่วนการส่งผ่านจาก F1 ไปยัง W2 สามารถทำได้โดยสามเส้นทางที่เป็นไปได้คือ $F1 \to DC \to W2, F1 \to F2 \to DC \to W2,$ และ $F1 \to W1 \to W2$ ค่าขนส่งต่อหน่วยแสดงทางด้านข้างของลูกศร ปริมาณทางด้านข้างของ $F1 \to F2$ และ $DC \to W2,$ คือปริมาณสูงสุดที่สามารถส่งผ่านลิ้งค์นี้ได้ ส่วนเส้นทางอื่นไม่มีข้อจำกัดทางด้านปริมาณการขนส่งขั้นสูงสุด

สิ่งที่ต้องตัดสินใจคือปริมาณผลิตภัณฑ์ที่จะส่งผ่านในแต่ละเส้นทาง เพื่อให้เป็นไปตามวัตถุประสงค์ของต้นทุนรวมที่น้อยที่สุด

รูปที่ 2.25: โครงข่ายของปัญหาการขนส่งและการกระจายสินค้า ตัวอย่างที่ 2.2.3



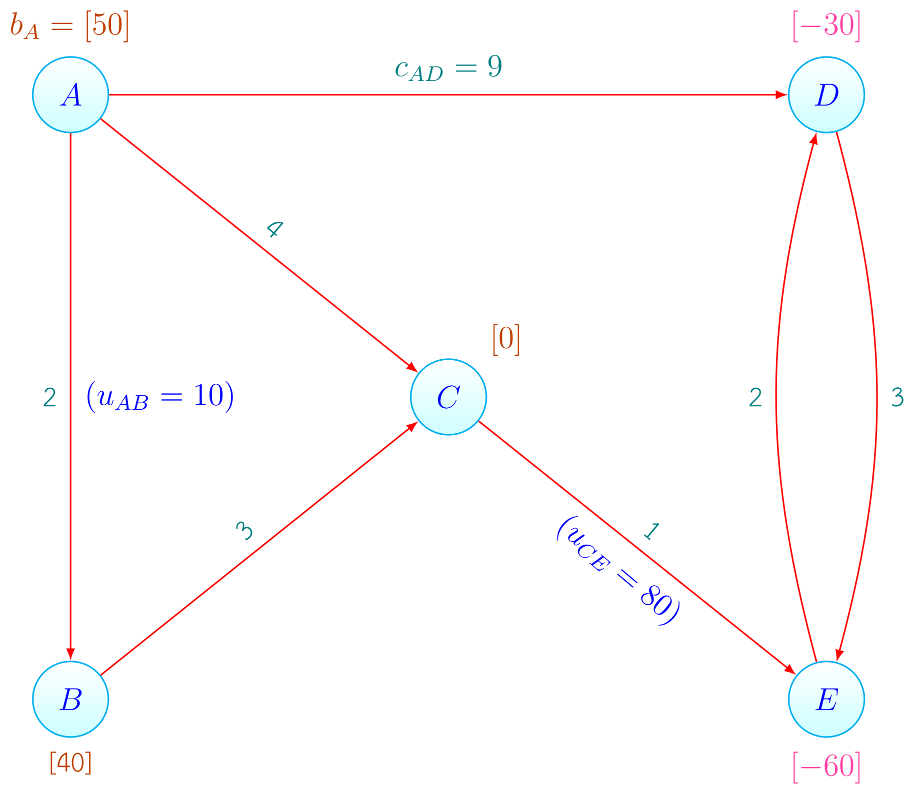
วิธีทำ

ปัญหานี้เป็นปัญหาการขนส่งที่เป็นปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด (MCFP) โดยสามารถพิจรณาได้ดังต่อไปนี้ เครือข่ายการกระจายสำหรับ บกจ. ที่แสดงตามรูปที่ 2.25 สามารถแปลงเป็น เครือข่ายการกระจายสำหรับปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด Minimum cost flow problem ได้ตามรูปที่ 2.26 ดังต่อไปนี้

ค่าพารามิเตอร์ตามรูปที่ 2.25 ให้ค่า $b_i, c_{ij},$ และ u_{ij} ดังในรูปที่ 2.26 ค่า b_i ในรูปที่ 2.26 แสดงในวงเล็บใหญ่ในแต่ละโหนด โหนดที่เป็น supply มีค่า $b_i > 0$ คือโหนด A และ B แทนโรงงานทั้งสองแห่ง ส่วนโหนดที่เป็น demand มีค่า $b_i < 0$ คือโหนด D และ E แทนคลังสินค้า warehouse ทั้งสองแห่ง โหนดที่เป็นโหลดส่งผ่าน transshipment

มีค่า $b_i = 0$ คือโหนด C ในที่นี้แทนศูนย์กระจายสินค้า ค่า c_{ij} แสดงอยู่ด้านข้างของแต่ละลิ้งค์ ในตัวอย่างนี้มีสองทางเชื่อมที่มีขีดจำกัดในการส่งไม่เกินค่าที่ให้ คือ $u_{AB} = 10$ และ $u_{CE} = 80$ ส่วนทางเชื่อมอื่นๆ $u_{ij} = \infty$

รูปที่ 2.26: โครงข่ายของปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด ของตัวอย่างที่ 2.2.3



แบบจำลองเชิงเส้นที่ได้จะเป็นไปตาม (2.11) ดังนี้

$$
\begin{array}{llllllllr}
max & z = & 2x_{AB} + 4x_{AC} & +9x_{AD} + 3x_{BC} & +x_{CE} + 3x_{DE} & +2x_{ED} & & & \text{(2.11a)} \\
s.t. & & x_{AB} + x_{AC} & +x_{AD} & & & = 50 & & \text{(2.11b)} \\
 & & -x_{AB} & + x_{BC} & & & = 40 & & \text{(2.11c)} \\
 & & - x_{AC} & - x_{BC} & +x_{CE} & & = 0 & & \text{(2.11d)} \\
 & & & -x_{AD} & + x_{DE} & -x_{ED} & = -30 & & \text{(2.11e)} \\
 & & & & -x_{CE} - x_{DE} & +x_{ED} & = -60 & & \text{(2.11f)}
\end{array}
$$

and

$$x_{AB} \leq 10,\ x_{CE} \leq 80,\ \text{all } x_{ij} \geq 0. \tag{2.12}$$

รูปที่ 2.27: แบบจำลองทางคณิตศาสตร์ของปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด (MCFP) ในรูปที่ 2.26

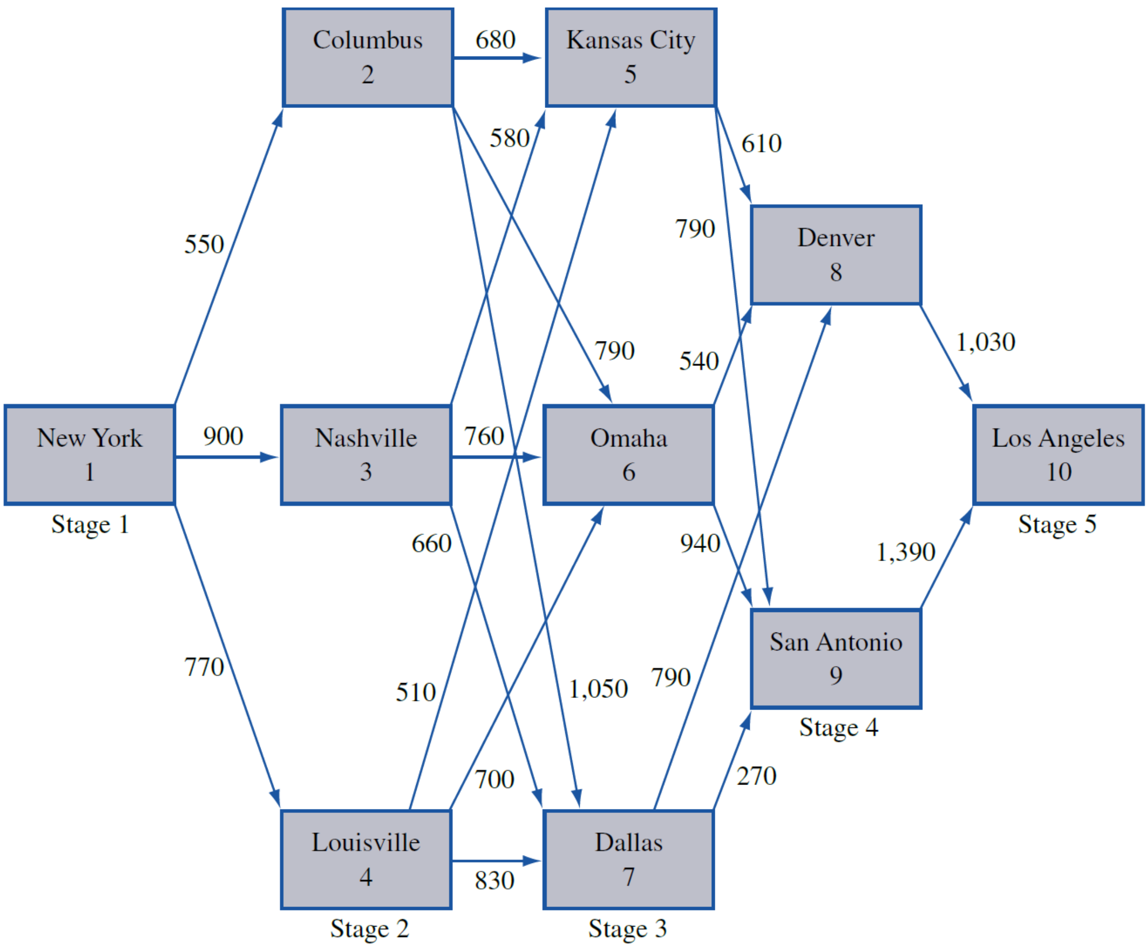
■

แบบฝึกหัด 2.2.11. แก้ปัญหาต้นทุนการส่งผ่านที่น้อยที่สุด (MCFP) ข้างต้นตามตัวอย่างที่ 2.2.3 รูปที่ 2.25 ด้วยโปรแกรม GAMS

2. ปัญหาเส้นทางเชื่อมที่สั้นที่สุด shortest path problem (SPP) เป็นปัญหาเครือข่ายที่มีความซับซ้อนน้อยที่สุด คือการหาเส้นทางจาก source node s ไป sink node t ที่มีต้นทุนรวมน้อยที่สุด โดยมีต้นทุนการเดินทาง c_{ij} จาก node i ไปยัง node j

ตัวอย่าง 2.2.4. พิจารณาปัญหาเส้นทางเชื่อมที่สั้นที่สุด (SPP) ดังต่อไปนี้ [14] Joe Cougar อาศัยอยู่ ณ มหานครนิวยอร์ค แต่ต้องการเดินทางไปยังแอลเลเพื่อตามความฝันในการเป็นดารา Joe มีต้นทุนที่จำกัดจึงตัดสินใจพักระหว่างทางในแต่ละคืนที่บ้านของเพื่อนๆ Joe มีเพื่อนในเมือง Columbus, Nashville, Louseville, Kansas City, Omaha, Dallas, San Anthonio, และ Denver ภายใน 1 คืนจากมหานครนิวยอร์ค Joe Cougar สามารถไปถึงยัง Columbus, Nashville, หรือ Louseville ภายใน 2 คืน Joe Cougar สามารถไปถึงยัง Kansas City, Omaha, หรือ Dallas และภายใน 3 คืน Joe Cougar สามารถไปถึงยัง San Anthonio หรือ Denver ในคืนที่ 4 เขาสามารถไปถึงยัง Los Angeles เพื่อประหยัดค่าน้ำมัน Joe Cougar จะวางแผนการนอนที่ใดในแต่ละคืน โดยการเลือกเส้นทางที่สั้นที่สุด แผนที่รวมเป็นไปตามภาพที่ 2.28

รูปที่ 2.28: กราฟแสดงปัญหาเส้นทางเชื่อมที่สั้นที่สุด shortest path problem



แบบฝึกหัด 2.2.12. แก้ปัญหาเส้นทางเชื่อมที่สั้นที่สุด shortest path problem ข้างต้นด้วยโปรแกรม GAMS

3. ปัญหาปริมาณส่งผ่านที่มากที่สุด maximum flow problem (MFP) ปัญหานี้เสมือนปัญหาเติมเต็มของปัญหาเส้นทางเชื่อมที่สั้นที่สุด shortest path problem เพราะปัญหาเส้นทางเชื่อมที่สั้นที่สุด shortest path problem ไม่มี

การจำกัดปริมาณการขนส่งจาก i ไป j แต่ปัญหาปริมาณส่งผ่านที่มากที่สุดนี้จำกัดให้ปริมาณการขนส่งรวมจาก i ไป j เป็น u_{ij}

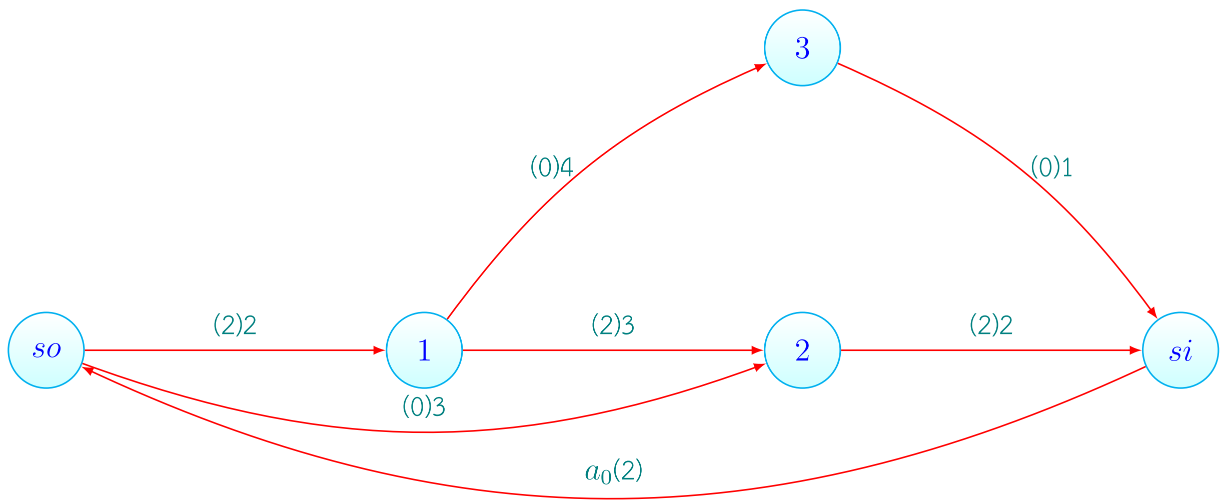
ตัวอย่าง 2.2.5. พิจารณาปัญหาปริมาณส่งผ่านที่มากที่สุด (MFP) ดังต่อไปนี้ [14]

Sunco Oil ต้องการส่งน้ำมันให้ได้ปริมาณสูงที่สุด (ต่อชั่วโมง) ผ่านท่อส่งจากแหล่ง so ไปยังปลายทาง si ตามรูปที่ 2.29 โดยระหว่างทางจะต้องผ่านสถานีที่ 1, 2, และ 3 แต่ละทางเชื่อมมีพารามิเตอร์ต่างๆที่แตกต่างกัน ปริมาณการส่งผ่านน้ำมันสูงสุดในหน่วยล้านบาเรลต่อชั่วโมงแสดงตามตารางที่ 2.6 ตัวเลขแต่ละตัวแสดงคือปริมาณความสามารถในการส่งสูงสุดของแต่ละทางเชื่อม (**arc capacity**) สร้างแบบจำลองเชิงเส้นเพื่อหาปริมาณการส่งผ่านที่สูงที่สุดจาก node so ไปยัง node si โดยให้ a_0 เป็นเส้นทางเชื่อมจำลอง

ตารางที่ 2.6: arc capacities สำหรับปัญหาปริมาณส่งผ่านที่มากที่สุด (MFP)

Arc	Capacity
$(so, 1)$	2
$(so, 2)$	3
$(1, 2)$	3
$(1, 3)$	4
$(3, si)$	1
$(2, si)$	2

รูปที่ 2.29: กราฟแสดงตัวอย่าง a feasible flow ของปัญหาปริมาณส่งผ่านที่มากที่สุด (MFP)



แบบฝึกหัด 2.2.13. แก้ปัญหาปริมาณส่งผ่านที่มากที่สุด (MFP) ข้างต้นด้วยโปรแกรม GAMS

ทฤษฎี 2.2.1. Max Flow - Min Cut Theorem [6] ปริมาณส่งผ่านที่มากที่สุด MFP เท่ากับประมาณรวม capacity ของเส้นตัด so จาก si ที่น้อยที่สุด

ปัญหาเครือข่ายอื่นๆที่สำคัญมีดังต่อไปนี้

- ปัญหาการมอบหมายงาน assignment problem

- ปัญหาการขนส่ง transportation problem
- ปัญหาการไหลแบบทั่วไป generalized flow problem
- ปัญหาการส่งผ่าน multicommodity flow problem
- ปัญหาการ minimum spanning tree problem
- ปัญหาการจับคู่ matching problem

2.2.5 assignment problem

แบบจำลองทางคณิตศาสตร์สำหรับปัญหา assignment problem เป็นดังต่อไปนี้

$$max \quad z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{2.13a}$$

$$s.t. \quad \sum_{j=1}^{n} x_{ij} = 1 \tag{2.13b}$$

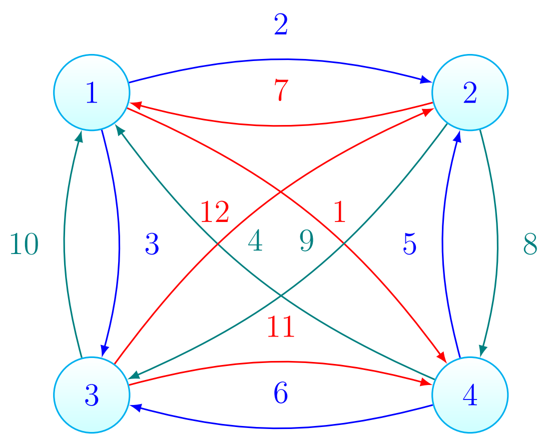
$$\sum_{i=1}^{m} t_{ij}x_{ij} \leq T \tag{2.13c}$$

$$x \in \{0,1\}^{m\times n}$$

2.2.6 ปัญหาการเดินทางของเซลส์แมน travelling salesman problem

ปัญหาการเดินทางของเซลส์แมน เป็นปัญหาที่ในปี 1937 Merrill M. Flood ได้จากเพื่อนร่วมงานวิจัยแล้วนำมา คิดค้นหาวิธีหาคำตอบ [9] ตัวอย่างรูปแบบของปัญหา

รูปที่ 2.30: ตัวอย่างกราฟแสดงปัญหาการเดินทางของเซลส์แมน 1



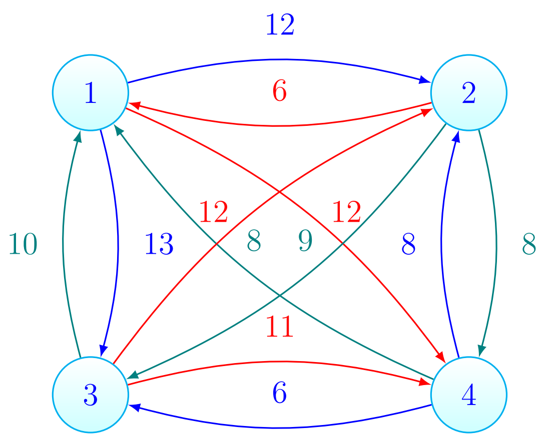
ซึ่งตัวอย่างดังกล่าวมีแบบจำลองทางคณิตศาสตร์ดังต่อไปนี้

$$\begin{aligned} max \quad z = \sum_i \sum_j c_{ij} x_{ij} && (2.14a) \\ s.t. \quad \sum_j x_{ij} = \quad 1 && (2.14b) \\ \sum_i x_{ij} = \quad 1 && (2.14c) \\ x_{ij} \in \{0, 1\} \end{aligned}$$

สามารถเขียนโค๊ด GAMS ได้ดังนี้

แบบฝึกหัด 2.2.14. แก้ปัญหาต่อไปนี้ด้วยโปรแกรม GAMS

รูปที่ 2.31: ตัวอย่างกราฟแสดงปัญหาการเดินทางของเซลส์แมน 2



แบบฝึกหัด 2.2.15. หาเส้นทางที่ดีที่สุดและวิจารณ์ผลที่ได้และวิธีการ ของปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองดังต่อไปนี้

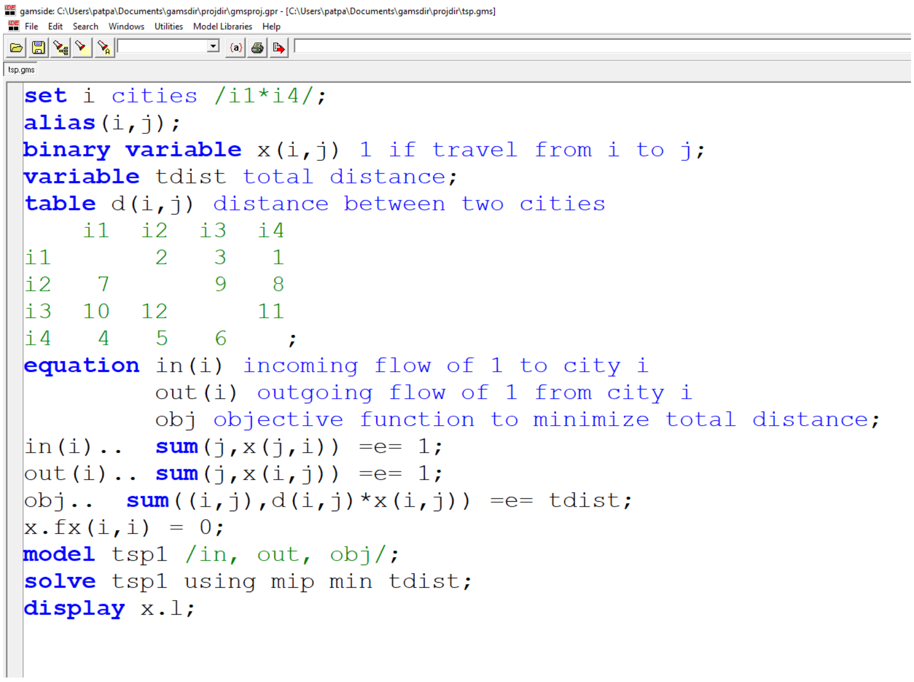
รูปที่ 2.32: ระยะทางระหว่างเมืองของปัญหาการเดินทางของเซลส์แมน 01

```
table d(i,j) distance between two cities
     i1  i2  i3  i4
i1        2   3   1
i2    7       9   8
i3   10  12      11
i4    4   5   6    ;
```

วิธีทำ

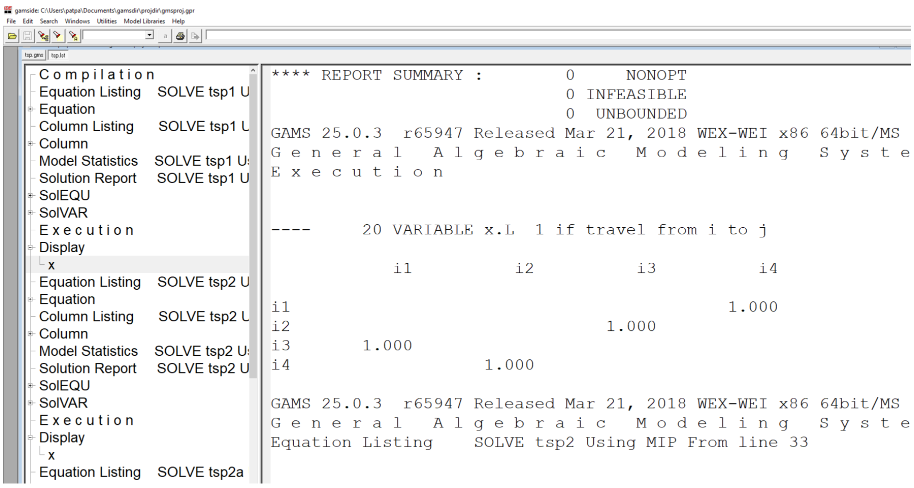
■

รูปที่ 2.33: โค้ดโปรแกรม GAMS สำหรับปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองตามรูป 2.32



```
set i cities /i1*i4/;
alias(i,j);
binary variable x(i,j) 1 if travel from i to j;
variable tdist total distance;
table d(i,j) distance between two cities
     i1   i2   i3   i4
i1         2    3    1
i2    7         9    8
i3   10   12        11
i4    4    5    6     ;
equation in(i) incoming flow of 1 to city i
         out(i) outgoing flow of 1 from city i
         obj objective function to minimize total distance;
in(i)..   sum(j,x(j,i)) =e= 1;
out(i).. sum(j,x(i,j)) =e= 1;
obj..   sum((i,j),d(i,j)*x(i,j)) =e= tdist;
x.fx(i,i) = 0;
model tsp1 /in, out, obj/;
solve tsp1 using mip min tdist;
display x.l;
```

รูปที่ 2.34: solution ของโค้ดโปรแกรม GAMS สำหรับปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองตามรูป 2.32

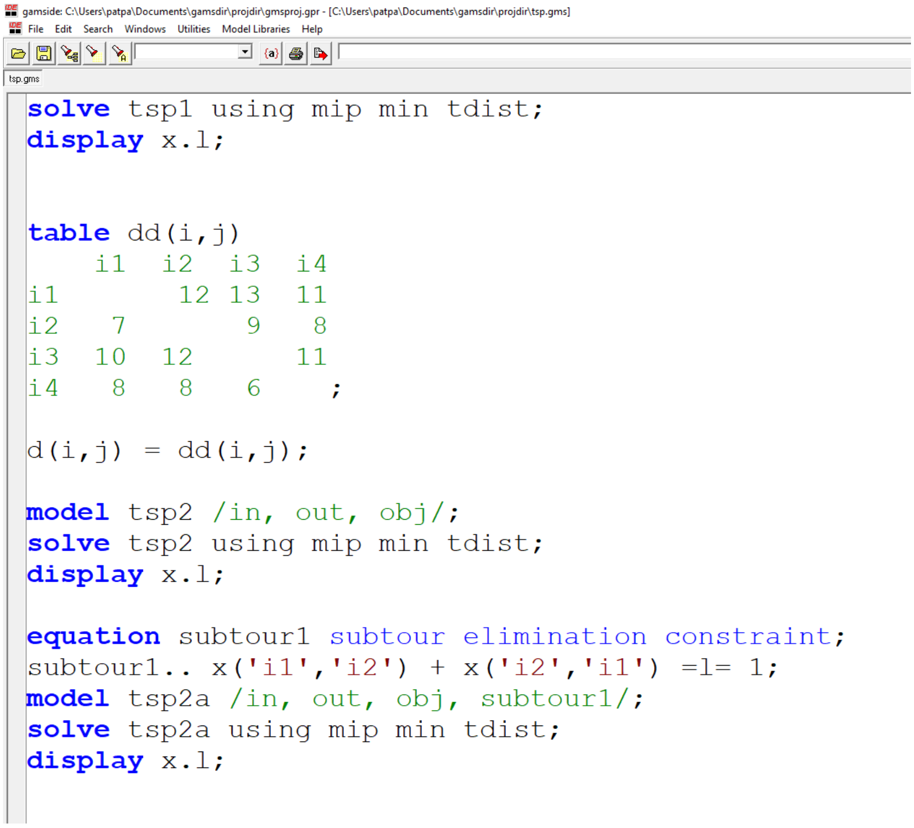


รูปที่ 2.35: ระยะทางระหว่างเมืองของปัญหาการเดินทางของเซลส์แมน 02

```
table dd(i,j) distance between two cities
     i1   i2   i3   i4
i1        12   13   11
i2    7         9    8
i3   10   12        11
i4    8    8    6     ;
```

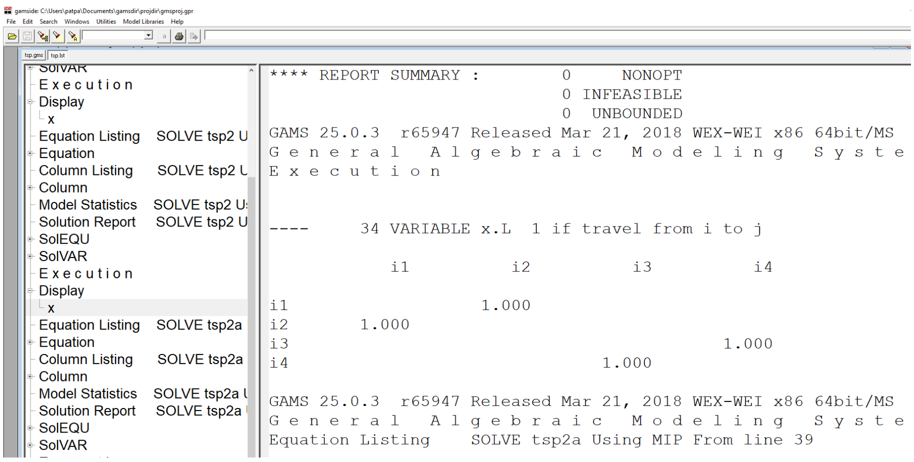
วิธีทำ

รูปที่ 2.36: โค้ดโปรแกรม GAMS สำหรับปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองตามรูป 2.35



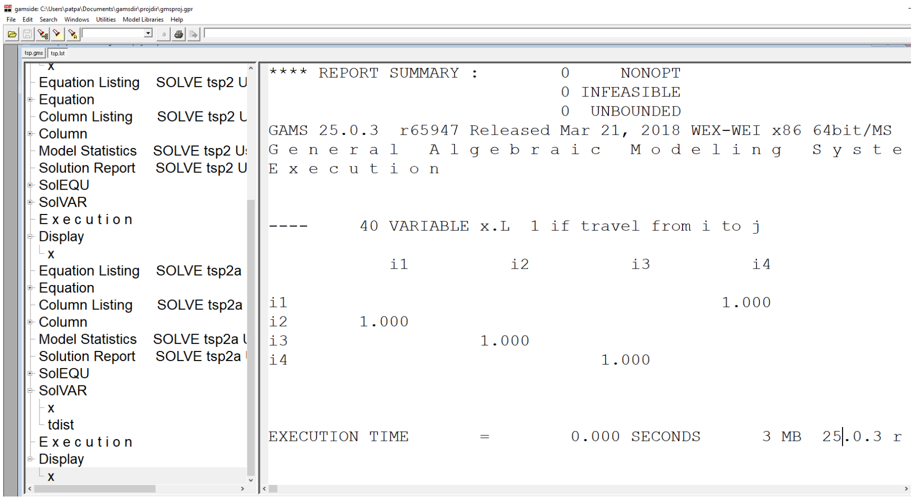

```
solve tsp1 using mip min tdist;
display x.l;


table dd(i,j)
    i1   i2   i3   i4
i1        12  13   11
i2   7          9   8
i3  10   12        11
i4   8    8    6     ;

d(i,j) = dd(i,j);

model tsp2 /in, out, obj/;
solve tsp2 using mip min tdist;
display x.l;

equation subtour1 subtour elimination constraint;
subtour1.. x('i1','i2') + x('i2','i1') =l= 1;
model tsp2a /in, out, obj, subtour1/;
solve tsp2a using mip min tdist;
display x.l;
```

รูปที่ 2.37: solution 1 ของโค้ดโปรแกรม GAMS สำหรับปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองตามรูป 2.35



■

รูปที่ 2.38: solution 2 ของโค้ดโปรแกรม GAMS สำหรับปัญหาการเดินทางของเซลส์แมนที่มีระยะทางระหว่างเมืองตาม รูป 2.35



รูปที่ 2.39: ระยะทางระหว่างเมืองของปัญหาการเดินทางของเซลส์แมน 03

```
table ddd(i,j) distance between two cities
     i1   i2   i3   i4
i1        11   13   11
i2   10         9   11
i3   10   12        11
i4    8    8    6      ;
```

รูปที่ 2.40: ระยะทางระหว่างเมืองของปัญหาการเดินทางของเซลส์แมน 04

```
table dddd(i,j) distance between two cities
     i1   i2   i3   i4
i1        12   13   12
i2    6         9    8
i3   10   12        11
i4    8    8    6      ;
```

แบบฝึกหัด 2.2.16. แก้ปัญหา TSP เช่นเดียวกันกับแบบฝึกหัด 2.2.15 โดยให้ปัญหามีขนาดที่ใหญ่ขึ้นและใช้การสุ่มเลือกระยะทางระหว่างเมือง

บทที่ 3

วิธีการหาคำตอบที่ดีที่สุด Optimization solution method

``Beauty is the first test: there is no permanent place in the world for ugly mathematics.''
Feng Jin, Shiji Song, and Cheng Wu

การหาคำตอบที่ดีที่สุดประกอบด้วยสองส่วน ส่วนแรกคือการหาคำตอบที่ดีที่สุด ส่วนที่สองคือการยืนยันว่าคำตอบนั้นเป็นคำตอบที่ดีที่สุด

3.1 ปัญหากำหนดการเชิงเส้น

3.1.1 วิธีการหาคำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้น optimization solution method for lp

หนึ่งในวิธีหลักของการหาคำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้น คือวิธี revised simplex ซึ่งใช้เมตริกซ์ในการคำนวนในแต่ละขั้นตอนของวิธีการซิมเพล็ก ผู้อ่านสามารถหาตัวอย่างได้ใน ภาคผนวกของ [1]

3.1.2 วิธีพิสูจน์คำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้น

เราสามารถพิสูจน์ว่าคำตอบที่มีเป็นคำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้นผสมจำนวนเต็มได้หลายวิธี เช่น

1. ใช้วิธีการ simplex ตาม 3.1.1ในภาคผนวกของ [1] หา optimality condition เช่นการแสดงว่า reduced cost เป็นไม่เป็นค่าลบ

2. ใช้ strong duality theorem

3. ใช้ complementarity slackness condition

3.2 ปัญหากำหนดการเชิงเส้นผสมจำนวนเต็ม

3.2.1 การหาคำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้นผสมจำนวนเต็ม

วิธีหลักของการหาคำตอบที่ดีที่สุดสำหรับปัญหากำหนดการเชิงเส้นผสมจำนวนเต็มคือวิธี branch-and-bound ซึ่งอาจจะลดเวลาของกระบวนการได้โดยวิธีการ cutting plane ซึ่งทั้งสองวิธีนี้อาศัยหลักการผ่อนคลายของปัญหา

3.2.2 การผ่อนคลายและการพิสูจน์ค่าที่ดีที่สุด

กำหนดปัญหากำหนดการเชิงเส้นผสมจำนวนเต็มดังนี้

$$max \quad z = c'x \tag{3.1a}$$

$$s.t. \quad Ax \le b \tag{3.1b}$$

$$x \ge \quad 0, \text{ and } x_i \text{ are integer for some i} \tag{3.1c}$$

ในกระบวนการหาคำตอบที่ดีที่สุด optimal solution ของปัญหา (3.1) ถ้าเราจะได้คำตอบที่เป็นไปได้ feasible solution $\bar{x}_1, \bar{x}_2, \bar{x}_3, \ldots, \bar{x}_s$ โดยที่ $\bar{z}_1 \le \bar{z}_2 \le \bar{z}_3 \le \cdots \le \bar{z}_s \le z^*$ เมื่อจำนวนดังกล่าวเป็นคู่กันตามลำดับ เราจะได้ว่าแต่ละตัวของ $\bar{x}_i$ ให้**ขอบเขตล่าง** lobwer bound ของปัญหาเป็น $\bar{z}_i$

ขอบเขตบน ของปัญหาคือค่า $\bar{w}_j$ ใดๆที่ $z^* \le \bar{w}_j$

เราจะได้ว่า $\bar{z}_i \le z^* \le \bar{w}_j$ และเมื่อเราสามารถหาค่าที่ i และ j ซึ่งมี $\bar{z}_i = \bar{w}_j$ เราจะได้คำตอบที่ดีที่สุดของปัญหา (3.1)

Primal Bounds ขอบเขตจากปัญหาหลัก

ทุกค่าที่เป็นไปได้ของปัญหา (3.1) ให้ขอบเขตล่าง

Dual Bounds ขอบเขตจากปัญหาคู่ควบ

การหาขอบเขตบนมีความซับซ้อนกว่าการหาขอบเขตล่าง วิธีการที่สำคัญในการหาขอบเขตบนคือการผ่อนคลายปัญหา relaxation แล้วหาคำตอบ ที่ดีที่สุดของปัญหาที่ผ่อนคลายนั้น

นิยาม 3.2.1. a relaxation

ปัญหา (RP) $z_R =$ max $\{f(x) : x \in T \subseteq R^n\}$ เป็นปัญหาผ่อนคลายของ (IP) $z =$ max $\{c(x) : x \in X \subseteq R^n\}$ ถ้า

1. $X \subseteq T$ และ
2. $f(x) \ge c(x)$ ในทุก $x \in X$

ทฤษฎี 3.2.1. ถ้าปัญหา RP เป็นปัญหาผ่อนคลายของปัญหา IP เราจะได้ว่า $z^R \ge z$

พิสูจน์. LTR □

ทฤษฎี 3.2.2. ปัญหา RMIP เป็นปัญหาผ่อนคลายของปัญหา MIP

พิสูจน์. LTR □

แบบฝึกหัด 3.2.1. จงหาขอบเขตบนและขอบเขตล่างของปัญหาต่อไปนี้

$$\begin{array}{lll}
\text{max} \quad z = & 4x_1 - x_2 & \\
s.t. & 7x_1 - 2x_2 & \le \quad 14 \\
& x_2 & \le \quad 3 \\
& 2x_1 - 2x_2 & \le \quad 40 \\
& x_1, x_2 \in Z_+^2 &
\end{array}$$

ทฤษฎี 3.2.3. ถ้าแบบจำลองทางคณิตศาสตร์ของปัญหาหนึ่งสามารถเขียนได้เป็น P_1 และ P_2 โดยที่ $P_1 \subset P_2$ นั่นคือ แบบจำลอง P_1 ดีกว่าแบบจำลอง P_2 เราจะได้ว่า $z_1^{LP} \leq z_2^{LP}$

พิสูจน์. LTR □

ทฤษฎี 3.2.4.

1. ถ้าปัญหาผ่อนคลายของ RP infeasible so as IP
2. if x^* is an optimal solution of RP, feasible to IP and $f(x^*) = c(x^*)$ then x^* is an optimal solution of IP

พิสูจน์. LTR □

แบบฝึกหัด 3.2.2. พิจารณาคำตอบ $x^* = (1, 1, 0, 0)$ ของปัญหาต่อไปนี้

$$\begin{aligned} \text{max} \quad z = \quad & 7x_1 + 4x_2 + 5x_3 + 2x_4 \\ s.t. \quad & 3x_1 + 3x_2 + 4x_3 + 2x_4 \leq 6 \\ & x \in B^4 \end{aligned}$$

3.2.3 วิธีการ branch-and-bound

วิธีการ branch-and-bound คือการแก้ปัญหากำหนดการเชิงเส้นผสมจำนวนเต็มโดยการแบ่งเป็นปัญหาย่อยและการผ่อนข้อจำกัดการเป็นจำนวนเต็มของตัวแปร ซึ่งปัญหาย่อยที่มีการผ่อนข้อจำกัดการเป็นจำนวนเต็มของตัวแปรนั้นเราจะแก้ได้โดยวิธี simplex โดยปัญหาใหญ่จะถูกแบ่งเป็นปัญหาย่อยสองปัญหาด้วยการเพิ่มข้อจำกัด เช่นการเพิ่มข้อจำกัด $x_1 \leq 5$ และ $x_1 \geq 6$

ตัวอย่าง 3.2.1. พิจารณาปัญหา MIP ต่อไปนี้

$$\begin{aligned} \text{max} \quad z = \quad & 5x_1 + 2x_2 && (3.4a) \\ s.t. \quad & 3x_1 + x_2 \leq 12 && (3.4b) \\ & x_1 + x_2 \leq 5 && (3.4c) \\ & x_1, x_2 \geq 0; x_1, x_2 \text{ integer} \end{aligned}$$

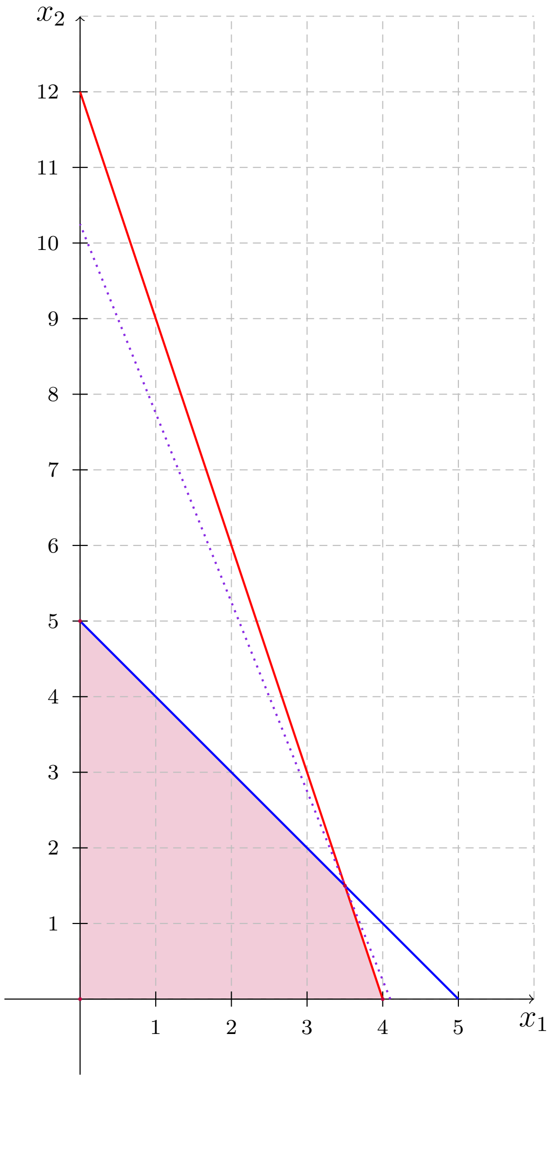
เราได้คำตอบที่ดีที่สุดของปัญหากำหนดการเชิงเส้นผ่อนคลายเป็น $\bar{z} = 20.5$ และ $\bar{x} = (3.5, 1.5)$ ซึ่งสามารถดูได้จากรูป 3.1 ดังต่อไปนี้

ตัวอย่าง 3.2.2. พิจารณาปัญหา MIP ต่อไปนี้

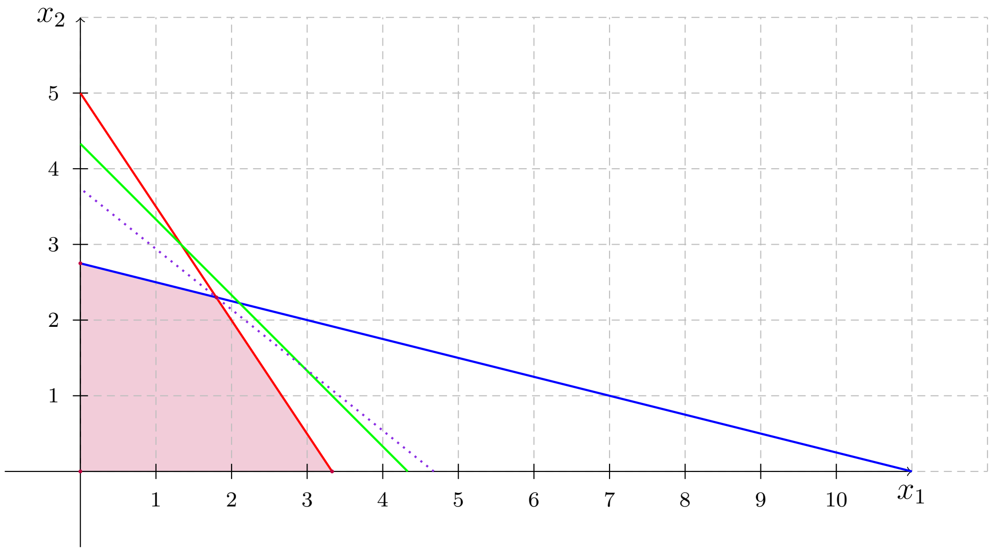
$$\begin{aligned} \text{max} \quad z = \quad & 4x_1 + 5x_2 && (3.5a) \\ s.t. \quad & 3x_1 + 2x_2 \leq 10 && (3.5b) \\ & x_1 + 4x_2 \leq 11 && (3.5c) \\ & 3x_1 + 3x_2 \leq 13 && (3.5d) \\ & x_1, x_2 \geq 0; x_1, x_2 \text{ integer} \end{aligned}$$

เราได้คำตอบที่ดีที่สุดของปัญหากำหนดการเชิงเส้นผ่อนคลายเป็น $\bar{z} = 18.7$ และ $\bar{x} = (1.8, 2.3)$ ซึ่งสามารถดูได้จากรูป 3.2 ดังต่อไปนี้

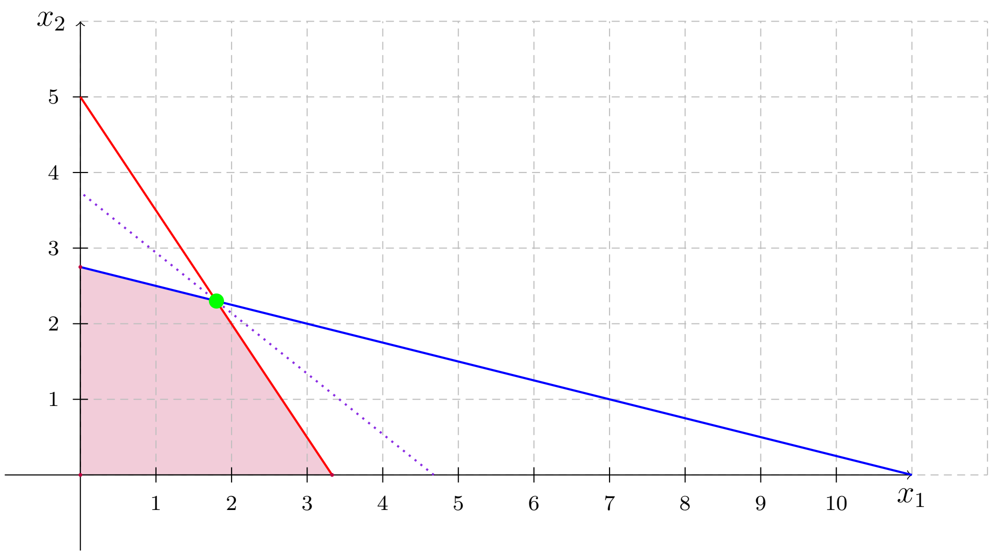
รูปที่ 3.1: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาผ่อนคลายของ (3.4)



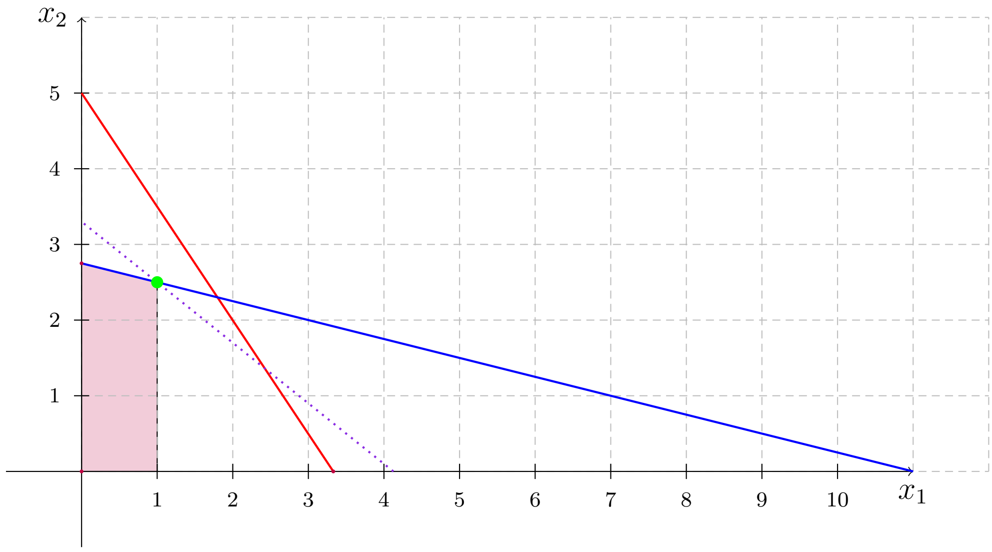
รูปที่ 3.2: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาผ่อนคลายของ (3.5)



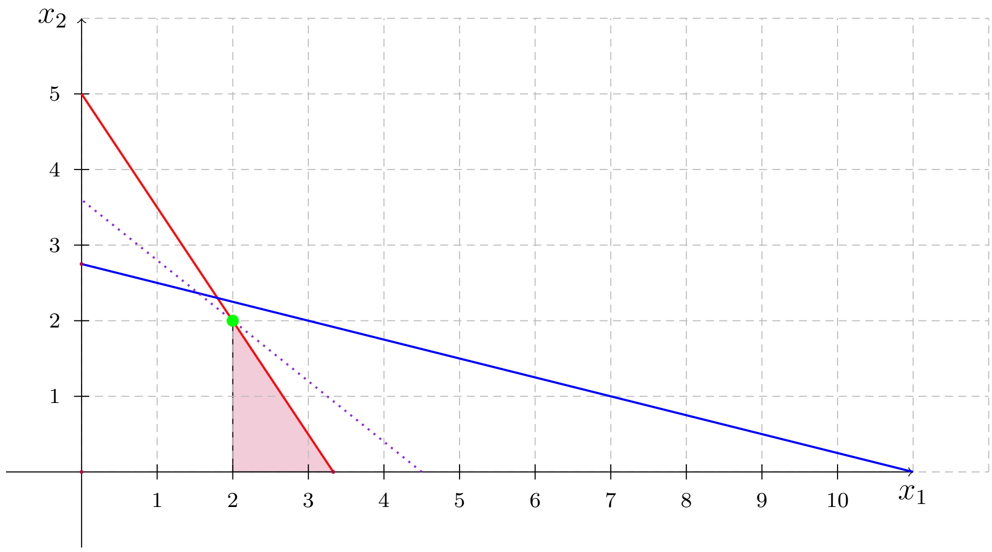
รูปที่ 3.3: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาผ่อนคลายของ (3.5) $\bar{x} = (1.8, 2.3)$, $\bar{z} = 18.7$



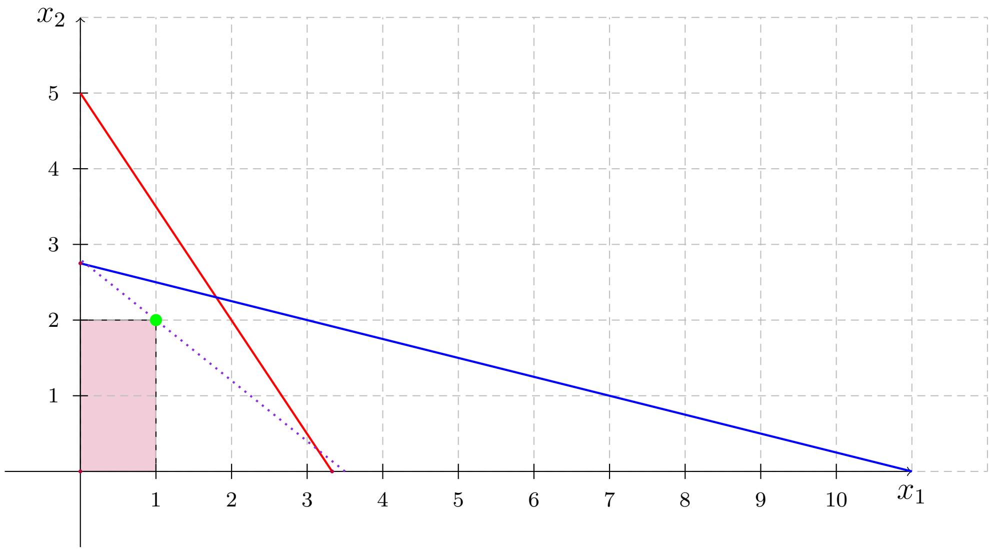
รูปที่ 3.4: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาย่อยของ (3.5) $x_1 \leq 1$,$\bar{x} = (1, 2.5)$, $\bar{z} = 16.5$



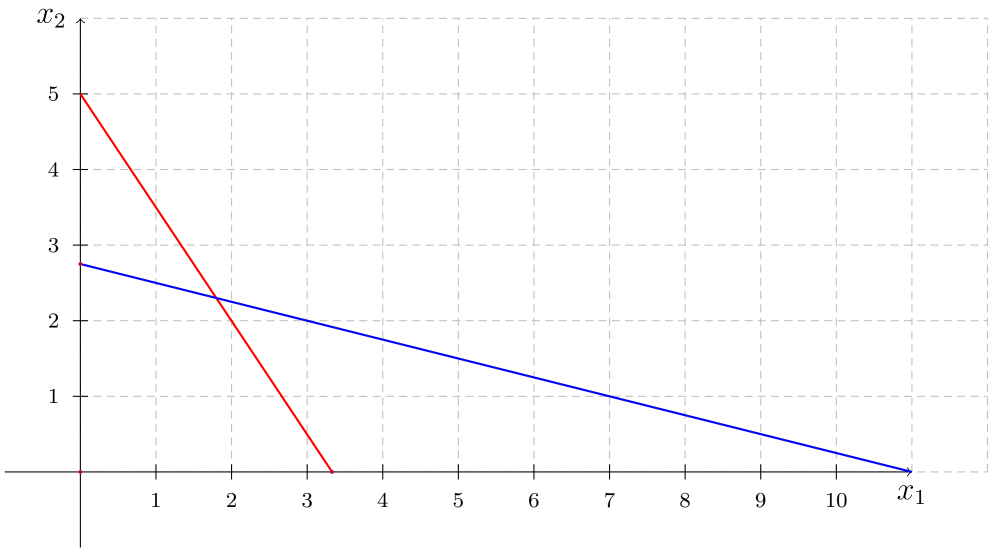
รูปที่ 3.5: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาย่อยของ (3.5) $x_1 \geq 2$,$\bar{x} = (2, 2)$, $\bar{z} = 18$



รูปที่ 3.6: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาย่อยของ (3.5) $x_1 \leq 1$ $x_2 \leq 2$ และ ,$\bar{x} = (1, 2)$, $\bar{z} = 14$

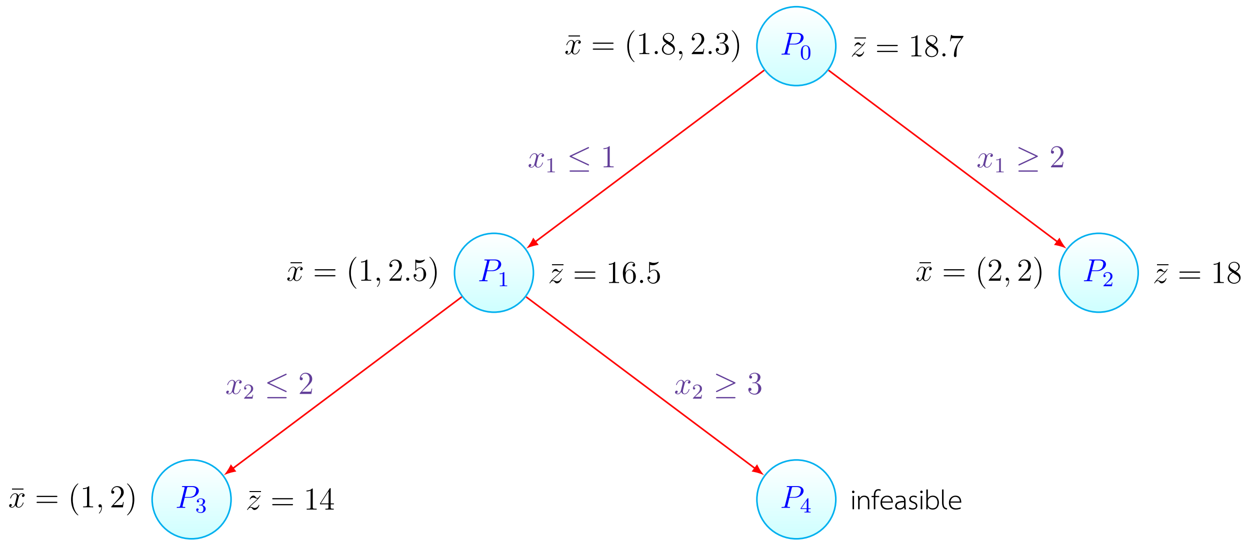


รูปที่ 3.7: กราฟแสดงคำตอบที่ดีที่สุดของปัญหาย่อยของ (3.5) $x_1 \leq 1$ และ $x_2 \geq 3$ infeasible



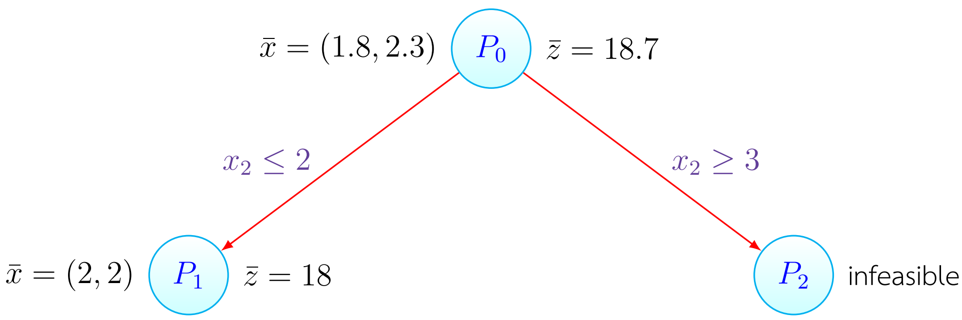
การแก้ปัญหากำหนดการเชิงเส้นด้วยวิธี branch-and-bound มีความช้าเร็วในการหาคำตอบที่ดีที่สุดขึ้นอยู่กับหลักเกณฑ์ในการเลือก ตัวแปรที่จะแตกกิ่ง(สร้างปัญหาย่อย) และการเลือกลำดับกิ่งที่จะแก้(เลือกลำดับปัญหาย่อยที่จะแก้) ในที่นี้เราใช้หลักการแตกกิ่งโดย**เลือกตัวแปรแรกสุดที่เป็นทศนิยม** และ **เลือกลำดับกิ่งที่จะแก้ในแนวระนาบจากซ้ายไปขวา** (ถ้ากิ่งซ้ายคือ $x_1 \leq 3$ กิ่งขวาจะเป็น $x_1 \geq 4$)

รูปที่ 3.8: กราฟแสดงขั้นตอนการแก้ปัญหา (3.5) โดยวิธี branch-and-bound



ถ้าเราใช้หลักเกณฑ์ที่การทำ branch-and-bound ที่ต่างกัน จะส่งผลต่อเวลาในการแก้ปัญหา เช่นถ้าเราเลือก ใช้หลักการแตกกิ่งโดย**เลือกตัวแปรท้ายสุดที่เป็นทศนิยม** และ **เลือกลำดับกิ่งที่จะแก้ในแนวระนาบจากซ้ายไปขวา** เราจะได้กราฟการแตกกิ่งแสดงการแก้ปัญหาดังนี้

รูปที่ 3.9: กราฟแสดงขั้นตอนการแก้ปัญหา (3.5) โดยวิธี branch-and-bound



แบบฝึกหัด 3.2.3. ให้ผู้อ่านลองตรวจสอบการคำนวนตามรูป 3.9 ด้วยตัวเอง

3.2.4 Cutting Planes

ถึงแม้ว่าเราสามารถแก้ปัญหากำหนดการเชิงเส้นผสมจำนวนเต็มได้ด้วยวิธี branch-and-bound แต่เวลาที่ใช้ในการแก้ปัญหาอาจจะเป็นเวลาที่นานมาก เกินเวลาที่เหมาะสมสำหรับความเป็นจริง เช่นถ้าเราจะวางแผนการผลิตหรือการจัดการส่งของประจำวันในตอนเช้าของทุกวัน เราอาจจะต้องการใช้เวลาในการหาคำตอบที่ดีที่สุดไม่เกินหนึ่งถึงสองชั่วโมง ซึ่งถ้าเราใช้แต่วิธี branch-and-bound เวลา หนึ่งถึงสองชั่วโมงนั้นอาจจะไม่เพียงพอ วิธี cutting planes เป็นอีกวิธีหนึ่งที่ช่วยให้เวลาในการหาคำตอบลดลง ถ้าเราทำได้อย่างเหมาะสม

นิยาม 3.2.2. Convex Set

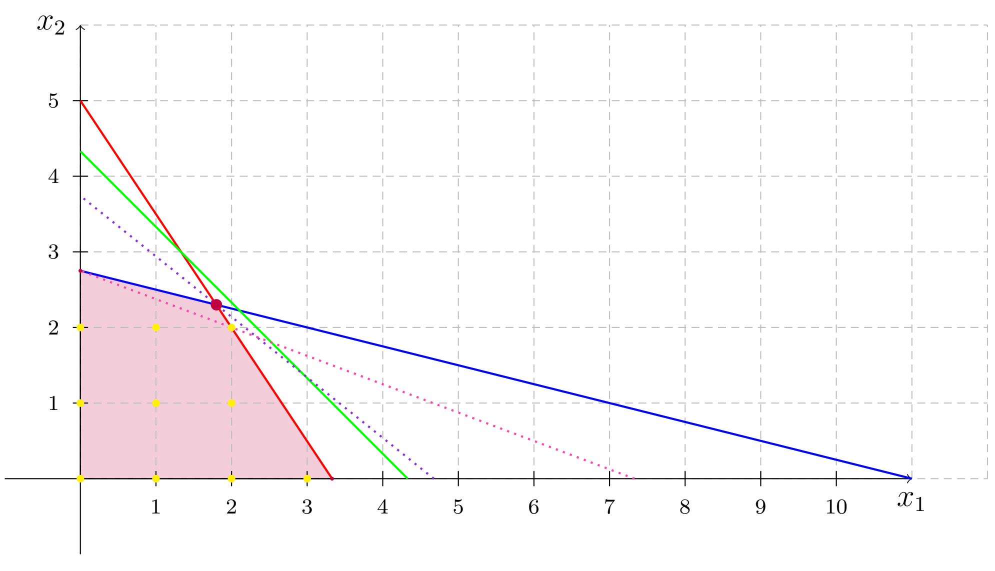
นิยาม 3.2.3. Convex Combination S ของ feasible points $x \in X$คือ
$S = \{y|y = \sum_{j=1}^{n} \lambda_j x_j$, where $\sum_{j=1}^{n} \lambda_j = 1, \lambda_j \geq 0$, and $x_j \in X\}$

นิยาม 3.2.4. Convex Hull ของ คือเซต Convex Combination ของ feasible points $x \in X$

บทแทรก 3.2.1. Convex Hull ของปัญหากำหนดการเชิงเส้นเป็น Convex Set

วิธี cutting planes คือการเพิ่มสมการ(หรืออสมการ)ข้อจำกัดลงในปัญหา เพื่อให้คำตอบของ relaxed mixed integer program ที่ไม่เป็นจำนวนเต็มโดนตัดออก ดังตัวอย่างในภาพที่ 3.10

รูปที่ 3.10: กราฟแสดงคำตอบที่ดีที่สุด $\bar{x} = (1.8, 2.3)$ ของปัญหาผ่อนคลายของ 3.5 ที่โดนตัดออกโดย cutting plane

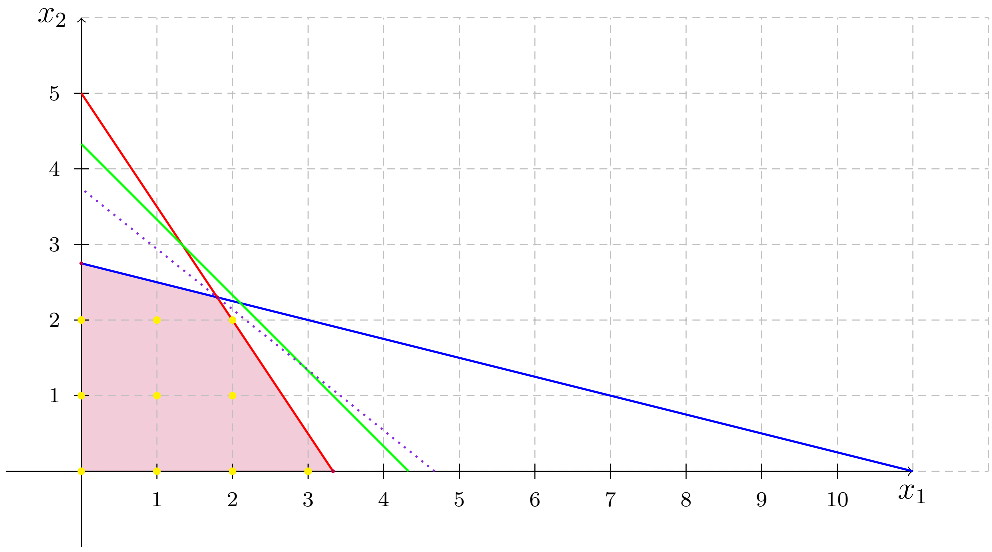


ซึ่งถ้าเราพิจารณาคำตอบที่เป็นจำนวนเต็มทั้งหมด เราจะได้ตามกราฟ ดังนี้

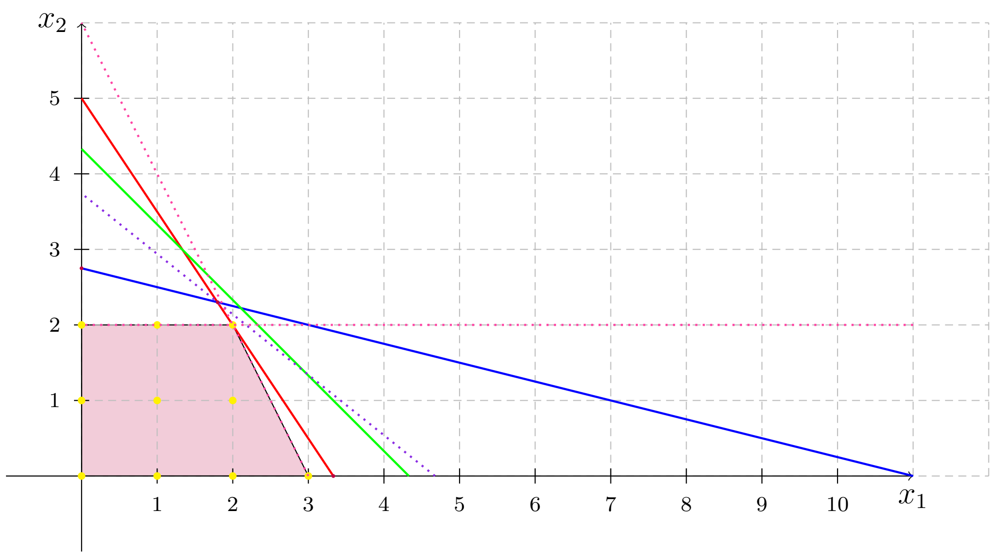
ความต้องการที่เรามีคือต้องการหาสมการ(หรืออสมการ)ข้อจำกัดที่ตัดส่วนที่เกินออก เราจะเรียกส่วนที่เหลือว่า **Convex Hull** หรือ **convex combination** ของ feasible points ดังภาพ

ตัวอย่าง 3.2.3. จงหาคำตอบที่ดีที่สุดของ RMIP และสมการข้อจำกัดที่แบ่งคำตอบนั้นออกจากคำตอบที่ดีที่สุดของ MIP

รูปที่ 3.11: กราฟแสดงคำตอบของปัญหา 3.5 และเส้น iso-profit



รูปที่ 3.12: กราฟ convex hull ของ feasible points ของปัญหา 3.5 และเส้น iso-profit



บทที่ 4

วิธีฮิวริสติกส์ Heuristic methods

``According to the No Free Lunch Theorem, all algorithms equal to the randomly blind search if no problem information is known."
Feng Jin, Shiji Song, and Cheng Wu

A **heuristic** method is a procedure that is likely to discover a very good feasible solution, but not necessarily an optimal solution, for the specific problem being considered. No guarantee can be given about the quality of the solution obtained, but a well-designed heuristic method usually can provide a solution that is at least nearly optimal (or conclude that no such solutions exist). The procedure also should be sufficiently efficient to deal with very large problems. The procedure often is a full-fledged iterative algorithm, where each iteration involves conducting a search for a new solution that might be better than the best solution found previously. When the algorithm is terminated after a reasonable time, the solution it provides is the best one that was found during any iteration.

Heuristic methods often are based on relatively simple common-sense ideas for how to search for a good solution. These ideas need to be carefully tailored to fit the specific problem of interest. Thus, heuristic methods tend to be ad hoc in nature. That is, each method usually is designed to fit a specific problem type rather than a variety of applications.

For many years, this meant that an OR team would need to start from scratch to develop a heuristic method to fit the problem at hand, whenever an algorithm for finding an optimal solution was not available. This all has changed in relatively recent years with the development of powerful metaheuristics. A **metaheuristic** is a general solution method that provides both a general structure and strategy guidelines for developing a specific heuristic method to fit a particular kind of problem. Metaheuristics have become one of the most important techniques in the toolkit of OR practitioners.

The nature of metaheuristics: A metaheuristic is a general kind of solution method that orchestrates the interaction between local improvement procedures and higher level strategies to create a process that is capable of escaping from local optima and performing a robust search of a feasible region.

วิธีฮิวริสติกส์เป็นวิธีการหาคำตอบปัญหา optimization ที่นิยมใช้กับปัญหาที่มีความซับซ้อน ไม่สามารถหาคำตอบโดยวิธีทางคณิตศาสตร์ได้ทันความต้องการ

วิธีทางคณิตศาสตร์มี วิธีซิมเพลกซ์ วิธี branch and bound และวิธี cutting plane เป็นต้น ส่วนวิธีฮิวริสติกส์ส่วนใหญ่จะเป็นการเลียนแบบสถานการณ์ธรรมชาติ เช่นวิธีก้วนของมด ant colony optimization วิธีการหมุนของอานุภาค particle swarn วิธีการพันธุกรรม Genetic Algorithms และวิธีตะกละมูมมาม greedy algorithm เป็นต้น วิธีฮิวริสติกส์ที่ดีต้องมีความรวดเร็ว แต่วิธีฮิวริสติกส์ นั้นมีข้อด้อยคือไม่มีกระบวนการตราวจสอบว่า คำตอบที่ได้ว่าคำตอบใดที่ได้เป็นคำตอบที่ดีที่สุด ซึ่งเราสามารถแก้ไขได้โดยใช้วิธีผสม (hybrid)คือการรวมวิธีฮิวริสติกส์ กับวิธีการทางคณิตศาสตร์ เข้าด้วยกัน

4.1 วิธีฮิวริสติกส์สำหรับปัญหาการเดินทางของเซลส์แมน

สมบัติหรือลักษณะของวิธีฮิวริสติกส์คือ หาคำตอบได้รวดเร็ว และสามารถหาได้หลายคำตอบ แต่ไม่สามารถประกันว่าจะได้คำตอบที่ดีที่สุด

4.1.1 แบบการสร้างทัวร์ tour contruction

nearest neighbtor $\mathcal{O}(n^2)$

1. เริ่มต้นจากเมืองใดเมืองหนึ่ง (สุ่มเมืองเริ่มต้น)
2. ออกเดินทางไปยังเมืองที่ใกล้ที่สุด
3. ทำต่อจนครบ tour

greedy algorithm $\mathcal{O}(n^2 \log_2 n)$

1. เรียงลำดับระยะทางระหว่างเมืองจากน้อยสุดไปมากสุด
2. เลือกการเดินทางเพิ่มใส่ใน tour เริ่มจากเส้นที่สั้นที่สุด
3. ทำต่อไปเรื่อยๆโดยไล่จากเส้นถัดๆไปที่เพิ่มใส่ tour ได้
4. ทำจนครบ tour

แบบฝึกหัด 4.1.1. ใช้วิธีการฮิวริสติกส์ตาม nearest neightbor และ greedy algorithm กับปัญหา TSP ที่มีขนาด 10 เมือง ที่ได้ระยะทางระหว่างเมืองที่เป็น symmetry จาก โค้ดโปรแกรม GAMS ตามตัวอย่างรูปที่ 4.1 ดังต่อไปนี้

รูปที่ 4.1: การสุ่มระยะทางระหว่างเมืองของปัญหาการเดินทางของเซลส์แมนที่มี 10 เมือง

```
set i '10 cities' /i1*i10/;
alias(i,j);
binary variable x(i,j) '1 if travel from i to j';
variable tdist 'total distance';
parameter d(i,j) 'distance between two cities'
option seed = 10;
d(i,j) = round(uniform(3,26));
d(i,j)$(ord(i) >= 6 or ord(j) >= 6) = round(uniform(28,48));
*display d;
alias(i,k,l);
loop((i,j,k,l)$(ord(i) < ord(j)  and
                not (sameas(i,k) and sameas(j,l))),
      d(i,j)$(d(i,j)=d(k,l)) = d(i,j)+ round(uniform(1,10));
);
*display d;

d(i,j)$(ord(i) > ord(j)) = 0;
display d;
```

ให้คำตอบเป็นเมืองเริ่มต้นถึงเมืองสิ้นสุดและกลับมายังเมืองเริ่มต้น (เป็นเส้นตรง) พร้อมระยะทางระหว่างเมืองบนเส้นลูกศรและระยะรวมที่ node สุดท้ายตามตัวอย่างรูปที่ 4.2

แบบฝึกหัด 4.1.2. เลือกจุดที่น่าสนใจ points of interest 6 จุดในมหาวิทยาลัยเกษตรศาสตร์วิทยาเขตกำแพงแสน สำหรับพาแฟนซ้อนมอเตอไซค์เที่ยวในยามเย็น

วิธีการ heuristics แบบการสร้าง tour อีกวิธีหนึ่งเป็นดังนี้

รูปที่ 4.2: ตัวอย่างคำตอบของปัญหาการเดินทางของเซลส์แมนที่เริ่มจากเมืองที่สอง

$$2 \xrightarrow{5} 1 \xrightarrow{8} 4 \xrightarrow{12} 3 \xrightarrow{14} \cdots \xrightarrow{10} 2 \qquad \sum = 154$$

Nearest Insertion Heuristics $\mathcal{O}(n^2)$

1. เลือกสองเมืองที่มีระยะทางระหว่างกันที่สั้นที่สุดแล้วสร้าง subtour จากสองเมืองนี้
2. เลือกเมืองนอกเหนือจากเมืองที่มีใน subtour ที่มีระยะทางไปยังเมืองที่ใน subtour ที่สั้นที่สุด
3. สร้าง subtour โดยเพิ่มเมืองใหม่นี้เข้าไปโดยให้มีค่าการเพิ่มเมืองที่สั้นที่สุด
4. ทำซ้ำจนครบ tour

Convex Hull $\mathcal{O}(n^2 \log_2 n)$

1. สร้าง subtour จาก Convex Hull ของ node เมืองสองมิติ
2. เลือกเมืองนอกเหนือจากเมืองที่มีใน subtour ที่มีระยะทางไปยังเมืองที่ใน subtour ที่สั้นที่สุด
3. สร้าง subtour โดยเพิ่มเมืองใหม่นี้เข้าไปโดยให้มีค่าการเพิ่มเมืองที่สั้นที่สุด
4. ทำซ้ำจนครบ tour

4.1.2 แบบการปรับปรุงทัวร์ tour improvement

2-opt and 3-opt

1. เริ่มจาก tour ที่สมบูรณ์แล้วเลือกเส้นเชื่อมสองเส้นใดๆ(หรือสาม)มาทดสอบสลับเส้นทาง เพื่อให้ได้ tour ที่สั้นลง
2. ทำไปเรื่อยๆจนไม่ที่คู่ที่สามารถปรับปรุง tour ได้

แบบฝึกหัด 4.1.3. จงให้เหตุผลว่า 2-opt หรือ 3-opt หาคำตอบได้ดีกว่ากัน

Lin-Kernighan

Tabu-Search

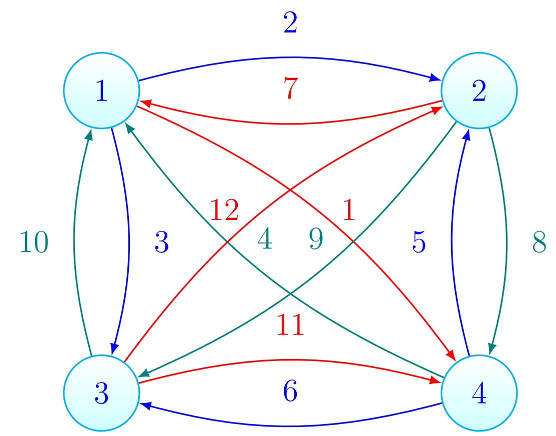
Simulated Annealing

Genetic Algorithms

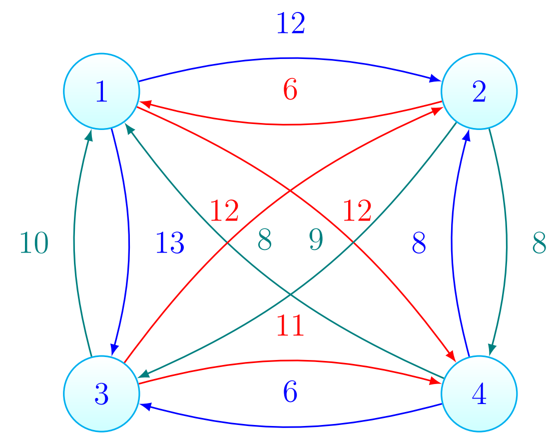
แบบฝึกหัด 4.1.4. แก้ปัญหา TSP ต่อไปนี้ด้วยวิธีฮิวริสติกส์

แบบฝึกหัด 4.1.5. แก้ปัญหา TSP ต่อไปนี้ด้วยวิธีฮิวริสติกส์

รูปที่ 4.3: ตัวอย่างกราฟแสดงปัญหาการเดินทางของเซลส์แมน 1



รูปที่ 4.4: ตัวอย่างกราฟแสดงปัญหาการเดินทางของเซลส์แมน 2



แบบฝึกหัด 4.1.6. แก้ปัญหา TSP ต่อไปนี้ด้วยวิธีฮิวริสติกส์
ห้าปีผ่านไปสมาชิกในกลุ่มได้พบรักแท้และต้องการจัดงานแต่งงานเพื่อเริ่มสร้างครอบครัว หนึ่งในหน้าที่ที่นิสิตได้รับคือทำการแจกบัตรงานแต่งงานกับแขกจำนวน 10 คนโดยมีเส้นทางการเดินทางเป็นไปตามข้อมูลที่ได้ของแต่ละกลุ่ม จงใช้วิธีการแก้ปัญหา TSP แบบต่างๆเพื่อหาเส้นทางที่สั้นที่สุด อธิบายวิธีการหาคำตอบ และวิเคราห์คำตอบที่ได้ของแต่ละวิธี

แบบฝึกหัด 4.1.7. เลือกวิธีการแล้วแก้ปัญหา TSP ในแบบฝึกหัดที่ 4.1.6 โดยคอมพิวเตอร์

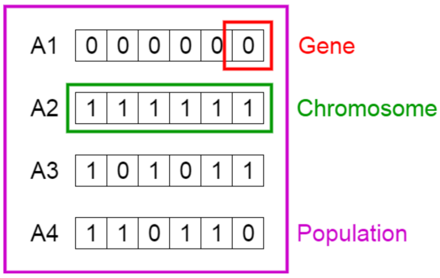
4.2 ขั้นตอนวิธีเชิงพันธุกรรม Genetic algorithm

ขั้นตอนวิธีเชิงพันธุกรรมเป็นวิธีการ metaheuristic นั่นคือเป็นวิธีการที่มีรูปแบบกว้างๆสำหรับการพัฒนาวิธีการ heuristic สำหรับปัญหาเฉพาะด้าน มีขั้นตอนวิธีการเลียนแบบธรรมชาติของการสืบทอดพันธุกรรมของสิ่งมีชีวิต ดังต่อไปนี้

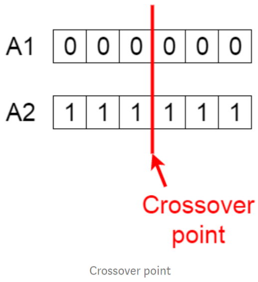
1. เลือกกลุ่มประชากรเริ่มต้น
2. หา fitness function
3. เลือกประชากรกลุ่มเจริญพันธุ์ mating pool

4. ผสม mating with crossover
5. กลายพันธุ์ mutation
6. ทำซ้ำ

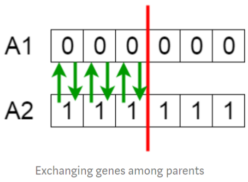
รูปที่ 4.5: ขั้นตอนวิธีเชิงพันธุกรรม 1



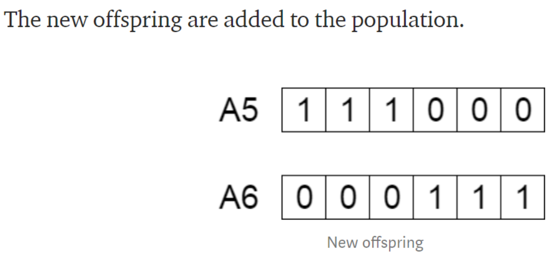
รูปที่ 4.6: ขั้นตอนวิธีเชิงพันธุกรรม 2



รูปที่ 4.7: ขั้นตอนวิธีเชิงพันธุกรรม 3

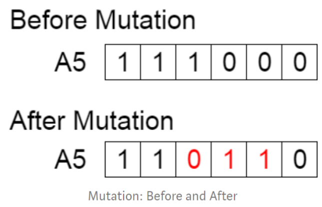


รูปที่ 4.8: ขั้นตอนวิธีเชิงพันธุกรรม 4



ตัวอย่าง 4.2.1.

รูปที่ 4.9: ขั้นตอนวิธีเชิงพันธุกรรม 5



รูปที่ 4.10: ขั้นตอนวิธีเชิงพันธุกรรม 6

Psuedocode

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

ในการสาธิตการประยุกต์แก้ปัญหาโดยวิธีการพันธุกรรม นิสิตสามารถศึกษาได้โดยการทำแบบฝึกหัดปัญหา Max One ดังต่อไปนี้

ให้ชุดเลขสุ่มทั้งหมดที่เหลือในบทที่ 4 นี้เป็นเลขสุ่มระหว่าง 0 ถึง 1 (unless stated otherwise)

แบบฝึกหัด 4.2.1. แก้ปัญหา Max One ด้วย GA โดยทำ 2 iteration ให้ประชากรมีขนาดเป็น 6 และขนาดของสตริงเป็น 10 โดยให้การสุ่มเลือกประชากรเริ่มต้นเป็นไปดังนี้ตามรูปที่ 4.11

รูปที่ 4.11: ข้อมูลสำหรับประชากรเริ่มต้น

	1	2	3	4	5	6	7	8	9	10
1	0.14757	0.05364	0.40244	0.83072	0.35001	0.93695	0.37546	0.98907	0.06863	0.68113
2	0.85955	0.543	0.60104	0.67911	0.60129	0.52039	0.31621	0.01151	0.88552	0.90876
3	0.04918	0.84531	0.68711	0.24433	0.26832	0.25825	0.86917	0.81528	0.82347	0.16289
4	0.77337	0.92592	0.1036	0.37117	0.1743	0.07771	0.16177	0.22861	0.10014	0.19911
5	0.42517	0.41906	0.09207	0.31799	0.60101	0.81496	0.34393	0.65961	0.8262	0.00434
6	0.48795	0.63253	0.78574	0.38785	0.1162	0.01491	0.21093	0.42555	0.5076	0.21411

แบบฝึกหัด 4.2.2. หา fitness function ที่เหมาะสม จากนั้นหาค่าของแต่ละ individual

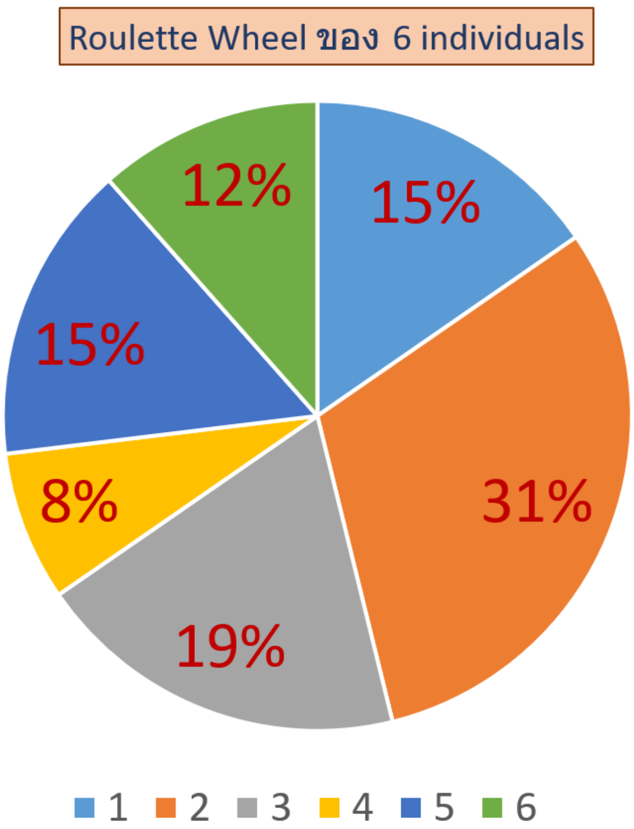
แบบฝึกหัด 4.2.3. หาวิธีใช้ roulette wheel ในการเลือกประชากรใส่ mating pool **ที่เหมาะสม** โดยให้การสุ่มเลือกเป็นไปดังนี้ตามรูปที่ 4.12

รูปที่ 4.12: ข้อมูลสำหรับสุ่มเลือก mating pool

0.88783
0.43109
0.52255
0.11753
0.54521
0.02718

ดู hint ของการใช้ roulette wheelตามรูปที่ 4.13

รูปที่ 4.13: Hint ของการใช้ roulette wheel



แบบฝึกหัด 4.2.4. เลือกการ cross over โดยสุ่มโอกาสเกิดในแต่ละคู่เป็น 20% จากเลขสุ่มตามรูปที่ 4.14 และเลือกตำแหน่งเริ่ม-สิ้นสุดการ cross over ของแต่ละคู่(ถ้าเกิด)จากการสุ่มตามรูปที่ 4.15

รูปที่ 4.14: ข้อมูลสำหรับสุ่มเลือกการ crossover

0.144344
0.967397
0.486600

รูปที่ 4.15: ข้อมูลสำหรับสุ่มเลือกตำแหน่ง crossover (ถ้าเกิด)

0.322	0.640
0.378	0.702
0.326	0.656

แบบฝึกหัด 4.2.5. เลือกการ mutate ของแต่ละยีนส์โดยให้โอกาสเป็น 10% ใช้การสุ่มในการเกิดตามรูปที่ 4.16 จากนั้นหาค่า fitness โดยรวมของประชากรใหม่ และค่าของคำตอบที่ดีที่สุด แล้วเปรียบเทียบกับของเก่า

รูปที่ 4.16: โอกาสการกลายพันธุ์

	1	2	3	4	5	6	7	8	9	10
1	0.387758	0.564693	0.72444	0.09185	0.891471	0.203751	0.761831	0.888312	0.318282	0.286824
2	0.600886	0.73345	0.808836	0.45509	0.611788	0.934798	0.693937	0.511401	0.92469	0.199783
3	0.988315	0.781302	0.557067	0.646717	0.211724	0.449977	0.95986	0.960436	0.935701	0.681191
4	0.904274	0.113455	0.389379	0.138395	0.771472	0.788064	0.823014	0.408451	0.745077	0.022545
5	0.061246	0.003048	0.433812	0.878005	0.666412	0.927552	0.594562	0.942147	0.117169	0.733423
6	0.895554	0.122935	0.242278	0.474771	0.824733	0.708754	0.603359	0.847973	0.829566	0.883137

ข้อมูลเลขสุ่มรูปที่ 4.17 ดังต่อไปนี้เพื่อแบบฝึกหัดที่ 4.2.1 ในส่วนที่เหลือ

รูปที่ 4.17: เลขสุ่มสำหรับแบบฝึกหัดที่ 4.2.1 ใน iteration ที่สอง

		0.18872							
		0.47723		0.78658		0.45537	0.1419		
		0.33732		0.14675		0.805	0.14328		
		0.73043		0.01243		0.67784	0.49479		
		0.76457							
0.16949	0.29871	0.95284	0.28787	0.13838	0.39945	0.11677	0.99601	0.45711	0.3181
0.11823	0.50837	0.29616	0.22622	0.32219	0.13987	0.27456	0.06625	0.15952	0.98252
0.66683	0.12053	0.45769	0.00047	0.04906	0.09479	0.9833	0.42779	0.15854	0.59439
0.34596	0.00332	0.92165	0.725	0.75649	0.45677	0.51764	0.77294	0.1732	0.06646
0.55007	0.21184	0.97021	0.56738	0.47308	0.95594	0.5501	0.41455	0.71531	0.25562
0.89549	0.03265	0.37413	0.6507	0.91426	0.19254	0.93398	0.27378	0.01529	0.19365

แบบฝึกหัด 4.2.6. แก้ปัญหา TSP ในขนาดที่เหมาะสม ด้วยวิธีการ heuristic และ metaheuristic ปรับค่าพารามิเตอร์ให้เหมาะสม วิเคราห์และวิจารณ์ผลที่ได้

1. เลือกขนาดของปัญหาที่เหมาะสมโดยใช้โค๊ดที่ปรับขนาดได้ง่าย (ในที่นี้ให้ใช้เมือง 10 เมือง)
2. generate พารามิเตอร์ (ระยะทางระหว่างเมือง) (ในที่นี้ให้ใช้ระยะทางของแต่ละกลุ่มที่เคยทำกับเมือง 10 เมืองนั้น)
3. ทำฮิวริสติกส์ (ทำ nearest nbhd กับ greedy)
4. ทำเมตาฮิวริสติกส์ (ทำ GA)
5. บันทึกผล เปรียบเทียบ วิเคราะห์ วิจารณ์ (เปรียบเทียบผลโดยเทียบเวลา หรือจำนวน iteration)
6. สรุปผล

วิธีทำ ใช้ option seed เท่ากับเลขประจำกลุ่มตลอดการสุ่ม
สามารถใช้โค๊ดที่มีกับข้อ 1 และข้อ 2 อาจจะปรับแก้เล็กน้อย
ข้อ 3 ให้เลือกหนึ่งหรือสองวิธีจาก section 4.1
ข้อ 3 ให้ใช้ GA ■

ตารางที่ 4.1: การทำ crossover เมื่อสร้าง offsprings สำหรับ GA on TSP

Parents

1	2	3	4	5	6	7	8	9

9	8	7	6	5	4	3	2	1

Offspring 1

					6	7	8	

9	5	4	3	2	6	7	8	1

Offspring 2

					4	3	2	

1	5	6	7	8	4	3	2	9

ตารางที่ 4.2: การทำ mutation เมื่อสร้าง offsprings สำหรับ GA on TSP

1	2	3	4	5	6	7	8	9

1	2	8	4	5	6	7	3	9

บทที่ 5

Special Topic ปัญหาการจัดเส้นทางสำหรับยานพาหนะ VRP

ปัญหาการจัดเส้นทางสำหรับยานพาหนะ หรือ Vehicle routing problem มีลักษณะดังต่อไปนี้

1. จัดรถบรรทุกจำนวน k คันสำหรับส่งสินค้าให้ลูกค้า n แห่ง
2. รถทุกคันต้องออกจากศูนย์กระจายสินค้าและกลับมายังศูนย์กระจายสินค้าเมื่อส่งของเสร็จครบรอบแล้ว
3. ลูกค้าแห่งที่ i มี demand เป็น d_i
4. ส่งสินค้าตามข้อจำกัดโดยให้มีต้นทุนที่ต่ำที่สุด

ความแตกต่างของข้อจำกัดต่างๆ ทำให้มีปัญหาการจัดเส้นทางสำหรับยานพาหนะหลากหลายแบบ ดังนี้

1. รถบรรทุกมีข้อจำกัดในการขนส่ง คือมี capacity จัดเป็นปัญหา Capacitated Vehicle Routing Problem (CVRP)

2. ปัญหา CVRP ที่ minimize ระยะทางรวม จัดเป็นปัญหา Distance-constrained Capacitated Vehicle Routing Problem (DCVRP)

3. ปัญหา CVRP ที่รับของจากลูกค้าอีกกลุ่มเพื่อนำของกลับเรียกว่า Vehicle Routing Problem with Backhaul (VRPB)

4. ปัญหา CVRP ที่ส่งของตามช่วงเวลาที่ตกลงกันก่อนจัดส่งเรียกว่า Vehicle Routing Problem with Time Windows (VRPTW)

5. ปัญหา CVRP ที่ทั้งรับของ และส่งของ เรียกว่า Vehicle Routing Problem with Pickup and Delivery (VRPPD)

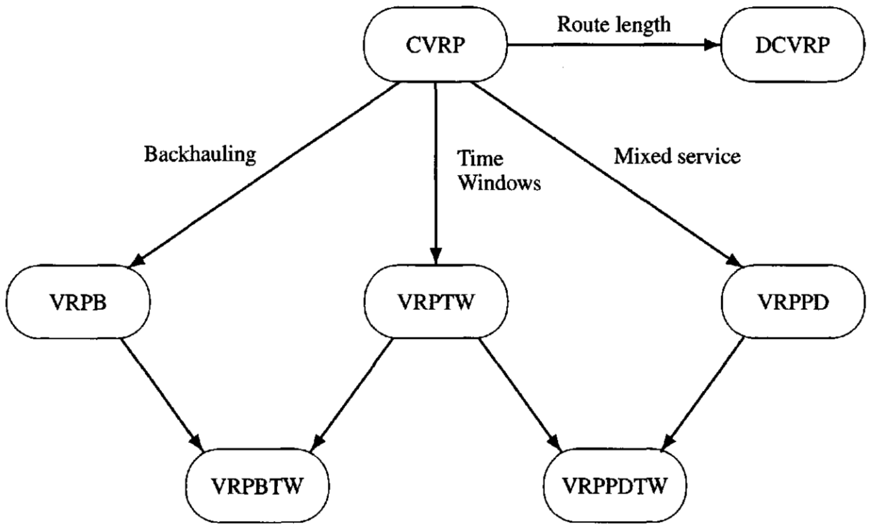
6. ปัญหารวมกันระหว่าง VRPB และ VRPTW เรียกว่า Vehicle Routing Problem with Backhaul and Time Windows (VRPBTW)

7. ปัญหารวมกันระหว่าง VRPTW และ VRPPD เรียกว่า Vehicle Routing Problem with Pickup and Delivery and Time Windows (VRPPDTW)
   ความเชื่อมโยงเป็นไปดังรูปที่ 5.1

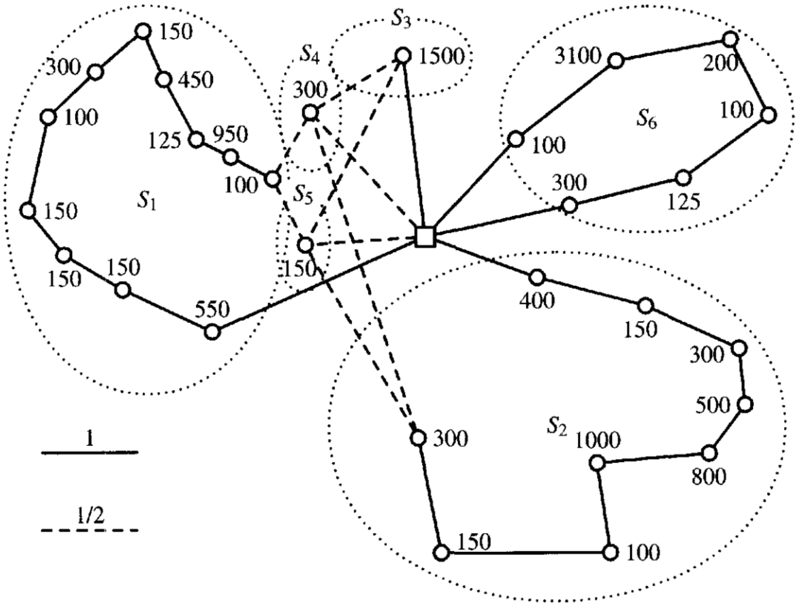
ย่อ

1. รถมี capacity (CVRP)
2. minimize ระยะทางรวม (DCVRP)
3. ส่งเสร็จแล้วขากลับรับสินค้า (VRPB)
4. มีช่วงเวลาจัดส่ง (VRPTW)
5. ทั้งรับ และส่งของ (VRPPD)
6. VRPB + VRPTW ขากลับรับสินค้า + ช่วงเวลาจัดส่ง เรียกว่า (VRPBTW)
7. VRPTW + VRPPD มีช่วงเวลาจัดส่ง + ทั้งรับและส่ง เรียกว่า (VRPPDTW)
   ความเชื่อมโยงเป็นไปดังรูปที่ 5.1

รูปที่ 5.1: ปัญหาการจัดเส้นทางสำหรับยานพาหนะ VRP ในรูปแบบต่างๆและการเชื่อมโยง



รูปที่ 5.2: ตัวอย่างปัญหาการจัดเส้นทางสำหรับยานพาหนะ แบบ cluster



แบบจำลองทางคณิตศาสตร์สำหรับ CVRP เป็นดังรูปที่ 5.3 ดังต่อไปนี้

รูปที่ 5.3: แบบจำลองทางคณิตศาสตร์สำหรับ CVRP [12]

$$\text{Minimize} \quad \sum_{i \in N} \sum_{j \in N} \sum_{v \in V} c_{ij} x_{ij}^{v} \tag{1}$$

$$\text{subject to} \quad \sum_{v \in V} y_i^v = 1 \quad \text{for } i \in N, \tag{2}$$

$$\sum_{i \in N} x_{ij}^v = y_j^v \quad \text{for } j \in N \text{ and } v \in V, \tag{3}$$

$$\sum_{j \in N} x_{ij}^v = y_i^v \quad \text{for } i \in N \text{ and } v \in V, \tag{4}$$

$$\sum_{i \in N} d_i y_i^v \leqslant Q \quad \text{for } v \in V, \tag{5}$$

$$\sum_{i \in N} x_{i1}^v \leqslant 1 \quad \text{for } v \in V, \tag{6}$$

$$\sum_{j \in N} x_{1j}^v \leqslant 1 \quad \text{for } v \in V, \tag{7}$$

สามารถเขียนโค๊ด GAMS ได้ดังนี้

5.1 Heuristic methods for CVRP

5.1.1 Clarke and Wright saving algorithm

1. จัดเส้นทาง 1 เส้นทางไปกลับสำหรับ 1 เป้าหมาย destination ต่อ 1 เส้นทาง
2. คำนวนค่าลดต้นทุนจากการรวมเส้นทาง ทุกคู่ แล้วเรียงลำดับจากมากสุด
3. รวมเส้นทางทีละเส้นตามลำดับจากสูงสุดไปต่ำสุดเท่าที่เป็นไปได้จนหมด

ตัวอย่าง 5.1.1. จากระยะทางตามตารางที่ 5.1 และ demand ตามตารางที่ 5.2 หา savings ของแต่ละ link แล้วลำดับลงในตารางที่ 5.3 จากนั้นใช้ Clarke and Wright saving heuristic หาคำตอบฮิวริสติก โดยที่ให้นถบรรทุกแต่ละคันมี capacity เป็น 23 หน่วย

ตารางที่ 5.1: ระยะทาง (ใต้เส้นทแยง)

	1	2	3	4	5	6	7	8	9	10
1		-	-	-	-	-	-	-	-	-
2	25									
3	43	29								
4	57	34	52							
5	43	43	72	45						
6	61	68	96	71	27					
7	29	49	72	71	36	40				
8	41	66	81	95	65	66	31			
9	48	72	89	99	65	62	31	11		
10	71	91	114	108	65	46	43	46	36	

ตารางที่ 5.2: ปริมาณสินค้าที่ต้องส่งแก่ลูกค้าในแต่ละจุด

ลูกค้า	2	3	4	5	6	7	8	9	10
ปริมาณ	4	6	5	4	7	3	5	4	4

ตารางที่ 5.3: ระยะประหยัด savings list

link	savings	link	savings	link	savings
(6,10)	86	(4,6)	47	(4,7)	15

วิธีทำ ■

ตารางที่ 5.4: ระยะทาง (ใต้เส้นทแยง) และระยะประหยัด (เหนือเส้นทแยง)

	1	2	3	4	5	6	7	8	9	10
1		-	-	-	-	-	-	-	-	-
2	25		39	48	25	18	5	0	1	5
3	43	29		48	14	8	0	3	2	23
4	57	34	52		55	47	15	3	6	20
5	43	43	72	45		77	36	19	26	49
6	61	68	96	71	27		50	36	47	86
7	29	49	72	71	36	40		39	46	57
8	41	66	81	95	65	66	31		78	66
9	48	72	89	99	65	62	31	11		83
10	71	91	114	108	65	46	43	46	36	

บทที่ 6

Special Topic ปัญหาการวางแผนจัดวางตู้คอนเทนเนอร์ภายในเรือสินค้า

Containership storage planning problem [11]

การค้าทางทะเลคิดเป็น 80% ของการค้าระหว่างประเทศทั้งหมด และเป็นการขนส่งที่มีประสิทธิภาพมากที่สุดทางหนึ่ง 90% ของสินค้าแยกชิ้นที่ขนส่งระหว่างประเทศขนส่งด้วยตู้คอนเทนเนอร์ ในปีคศ. 2016 มีการขนส่งรวมทั้งสิ้นประมาณ 140 ล้าน Twenty-foot Equivalent Units (TEUs) เรือขนตู้สินค้าสมัยใหม่สามารถบรรทุกได้ถึง 20,000 TEUs โดยมีลำใหญ่สุดคือ OOCL Hong Kong ที่ขนได้ถึง 21413 TEUs การขนส่งที่ใกล้เต็มขีดจำกัด capacity ทำให้เกิด economy of scale

รูปที่ 6.1: iduna 1

รูปที่ 6.2: iduna 2

รูปที่ 6.3: OOCL Hong Kong

รูปที่ 6.4: msc-gulsun 1

รูปที่ 6.5: msc-gulsun 2

รูปที่ 6.6: msc-gulsun 3

รูปที่ 6.7: madrid-maersk 1

รูปที่ 6.8: madrid-maersk 2

6.1 รูปแบบของปัญหา

พิจารณาเรือเดินทางตามเส้นทางการค้าที่มีท่าเรือ N ท่าเรือ ให้ตู้ทั้งหมดมีขนาดเท่ากันและงดการพิจารณาน้ำหนักและสมดุลของการบรรทุก โดยสามารถพิจารณาสมมติให้เรือมี bay เดียวที่มี R แถว C คอลัมน์ แถวที่ 1 อยู่ล่างสุด คอลัมน์ที่ 1 อยู่ซ้ายสุด

ให้ $\mathcal{N} = \{1, \dots, N\}$

$\mathcal{C} = \{1, \dots, C\}$

$\mathcal{R} = \{1, \dots, R\}$

โดยสัญลักษณ์ดังกล่าวเป็นไปตาม [2] และ [5] การยกตู้คอนเทนเนอร์ขึ้นลงตามเส้นทางการเดินเรือเป็นไปตาม $N \times N$ เมทริกซ์ T โดยสมาชิกในตำแหน่ง i, j แทนจำนวนตู้ที่มาจากท่าเรือที่ i ไปยังท่าเรือที่ j ดังนั้น T จึงเป็นเมตริกซ์ที่สามเหลี่ยมบนเป็นค่าบวก มีสมมติฐานคือเมตริกซ์ T เป็นเมตริกซ์ที่ feasible นั่นคือ

ประพจน์ 6.1.1.

$$\sum_{k=1}^{i} \sum_{j=i+1}^{N} T_{kj} \leq R \cdot C \quad \forall i \in \mathcal{N} : i < N \tag{6.1}$$

การเดินเรือเริ่มจากท่าเรือที่ 1 ไปยังท่าเรือที่ $2, 3, \dots, N$ ตามลำดับ ในแต่ละท่าเรือที่ $i = 1, 2, \dots, N-1$ ตู้คอนเทนเนอร์สถานีปลายทาง i ถูกยกออกจากเรือ และ ยกตู้คอนเทนเนอร์สถานีปลายทาง $j = i+1, \dots, N$ เข้าเรือ **ตู้คอนเทนเนอร์สถานีปลายทาง j เรียกว่า j-container** ณ สถานี N ตู้ที่เหลือทั้งหมดถูกยกออก

แบบจำลองทางคณิตศาสตร์[11]
ให้ตัวแปรตัดสินใจเป็น

$$\forall i,j \in \mathcal{N} \text{ โดย } i \le N-2 \text{ และ } i<j; \forall r \in \mathcal{R}; \forall c \in \mathcal{C},$$
$$x_{ijrc} = \begin{cases} 1, & \text{ถ้าเมื่อออกจากท่าเรือ } i \text{ ตู้ j-container อยู่ในตำแหน่ง } (r,c) \\ 0, & \text{otherwise} \end{cases} \tag{6.2a}$$

$$\forall i,j \in \mathcal{N} \text{ โดย } 1<i<N \text{ และ } i<j; \forall r \in \mathcal{R}; \forall c \in \mathcal{C},$$
$$y_{ijrc} = \begin{cases} 1, & \text{ถ้าตู้ j-container ที่อยู่ในตำแหน่ง } (r,c) \text{ หลังท่าเรือ } i-1 \text{ ถูกขยับโดยไม่เพิ่มมูลค่าที่ท่าเรือ } i \\ 0, & \text{otherwise} \end{cases} \tag{6.3a}$$

จากท่าเรือ $N-1$ ไปยังท่าเรือ N จะมีแต่ N-container ไม่มีการโยกขยับตู้สินค่าโดยไม่เพิ่มมูลค่าที่ท่าเรือ N และตำแหน่งของการวางตู่จากท่าเรือ $N-1$ ไปยังท่าเรือ N ไม่มีความสำคัญต่อปัญหา จึงไม่ต้องพิจารณาตัวแปร $x_{(N-1)Nrc}$ หรือ y_{NNrc}

กำหนดการเชิงเส้นจำนวนเต็มจำลองปัญหาการเรียงตู้คอนเทนเนอร์เป็นดังต่อไปนี้

$$\text{min} \quad \sum_{i=2}^{N-1} \sum_{j=i+1}^{N} \sum_{r=1}^{R} \sum_{c=1}^{C} y_{ijrc} \tag{6.4a}$$

$$\text{st} \quad x_{i-1\,irc} - x_{ijrc} \le y_{ijrc} \tag{6.4b}$$
$$\forall i,j \in \mathcal{N} : 1<i\le N-2, i<j; \forall r \in \mathcal{R}; \forall c \in \mathcal{C},$$

$$x_{i-1\,ir-1\,c} + \sum_{j=i+1}^{N} y_{ijr-1\,c} \le x_{i-1\,irc} + \sum_{j=i+1}^{N} y_{ijrc} \tag{6.4c}$$
$$\forall i \in \mathcal{N} : 1<i<N; \forall r \in \mathcal{R} : r>1; \forall c \in \mathcal{C},$$

$$\sum_{r=1}^{R} \sum_{c=1}^{C} x_{1jrc} = T_{1j} \quad \forall j \in \mathcal{N} : j>1 \tag{6.4d}$$

$$\sum_{r=1}^{R} \sum_{c=1}^{C} (x_{ijrc} - x_{i-1\,jrc}) = T_{ij} \quad \forall i,j \in \mathcal{N} : 1<i\le N-2, i<j, \tag{6.4e}$$

$$\sum_{j=i+1}^{N} x_{ijrc} \le 1 \quad \forall i \in \mathcal{N} : i \le N-2; \forall r \in \mathcal{R}; \forall c \in \mathcal{C}. \tag{6.4f}$$

รูปที่ 6.9: ตัวอย่างการจัดเรียงตู้คอนเทนเนอร์ที่ท่าเรือที่ 2 และ 3

5	3	3	3	4
4	6	3	4	4
3	5	5	3	5
2	6	5	3	5
1	6	6	3	6
	1	2	3	4

Port 2

5	6	4	4	4
4	6	5	4	4
3	6	5	4	5
2	6	5	4	5
1	6	6	4	6
	1	2	3	4

Port 3

ตัวอย่าง 6.1.1. ศึกษารูปแบบการจัดวางตู้คอนเทนเนอร์ตามรูปที่ 6.7 จะช่วยเพิ่มความเข้าใจในแบบจำลองทางคณิตศาสตร์และเข้าใจในรูปแบบต่างๆของการขยับตู้คอนเทนเนอร์โดยไม่เพิ่มมูลค่ามากยิ่งขึ้น รูปที่ 6.7 แสดงการจัดเรียงตู้ ณ ท่าเรือที่ 2 และท่าเรือที่ 3 แต่ละตารางแทนตู้คอนเทนเนอร์ 1 ตู้ โดยตัวเลขแสดงท่าเรือปลายทางของตู้ ตารางสีเทาคือตู้ที่มีการขยับ ณ ท่าเรือที่ 3 เนื่องจากตำแหน่ง (3, 1) มีการเปลี่ยนปลายทางของตู้ระหว่างท่าเรือที่ 2 และ 3 แสดงว่าจะต้องมีการขยับตู้ในตำแหน่งนี้ที่ท่าเรือที่ 3 และสมการข้อจำกัด (6.4b) ทำให้ $y_{3531} = 1$

จากนั้นให้ดู 6-container ที่ตำแหน่ง (4, 1) ณ ท่าเรือที่ 2 ที่จะต้องถูกขยับ ณ ท่าเรือที่ 3 เพื่อจะได้นำ 5-container ออก แต่ ณ ท่าเรือที่ 3 ในตำแหน่ง (4,1) นี้ก็ยังคงเป็นตู้ 6-container (ซึ่งอาจจะเป็นตู้เดิมหรือตู้อื่น) ทำให้สมการข้อจำกัด (6.4b) ไม่จำกัดสถาณการณ์ในกรณีนี้ แต่ถูกบังคับโดยข้อจำกัด (6.4c) เนื่องจากมีการขยับกองที่ 1 ชั้น 3 ณ ท่าเรือที่ 3 ตู้ด้านบนจากนี้ในช่อง (4,1) จะต้องมีจุดหมายปลายทางเป็นท่าเรือที่ 3 หรือมิฉนั้นจะต้องถูกขยับเช่นดังในกรณีนี้ นอกจากนั้นข้อจำกัด (6.4c) บ่งชี้การขยับที่ไม่เพิ่มมูลค่าอันเนื่องมาจากการขยับตู้ที่ต้องลง ณ ท่าเรือจุดหมาย

สุดท้ายให้พิจารณาคอนเทนเนอร์ในตำแหน่งที่ (4, 3) ณ ท่าเรือที่ 2 เนื่องจากมี 3-container ณ ท่าเรือที่ 2 ในตำแหน่ง (3, 3) จึงต้องมีการขยับตู้ทั้งหมดด้านบนตู้ 3-container นี้เพราะตู้เหล่านั้นทั้งหมดมีปลายทางหลังจากท่าเรือที่ 3

บรรณานุกรม

[1] ภัทรพงษ์ ภาคภูมิ. *การวิจัยการดำเนินงานสำหรับวิศวกร*. คณะวิศวกรรมศาสตร์ กำแพงแสน ม.เกษตรศาสตร์, 2019.

[2] Mordecai Avriel, Michal Penn, Naomi Shpirer, and Smadar Witteboon. Stowage planning for container ships to reduce the number of shifts. *Annals of Operations Research*, 76:55--71, 1998.

[3] Kenneth R Baker and Dan Trietsch. *Principles of sequencing and scheduling*. John Wiley & Sons, 2013.

[4] GAMS Development Corporation. General Algebraic Modeling System (GAMS) Release 25.1.3. Washington, DC, USA, 2016.

[5] Ding Ding and Mabel C Chou. Stowage planning for container ships: A heuristic algorithm to reduce the number of shifts. *European Journal of Operational Research*, 246(1):242--249, 2015.

[6] Lester Randolph Ford Jr and Delbert Ray Fulkerson. *Flows in networks*. Princeton university press, 2015.

[7] S Frederick, LIEBERMAN HILLIER, and J GERALD. *Introduction to operations research*. MCGRAW-HILL EDUCATION, 2014.

[8] Simon French. Sequencing and scheduling. *An Introduction to the Mathematics of the Job-shop*, 1982.

[9] Saul I Gass and Arjang A Assad. *An annotated timeline of operations research: An informal history*, volume 75. Springer Science & Business Media, 2005.

[10] Jens Kuhpfahl. *Job shop scheduling with consideration of due dates: Potentials of local search based solution techniques*. Springer, 2015.

[11] Consuelo Parreño-Torres, Ramon Alvarez-Valdes, and Francisco Parreño. Solution strategies for a multiport container ship stowage problem. *Mathematical Problems in Engineering*, 2019, 2019.

[12] Tantikorn Pichpibul and Ruengsak Kawtummachai. An improved clarke and wright savings algorithm for the capacitated vehicle routing problem. *ScienceAsia*, 38(3):307--318, 2012.

[13] Yves Pochet and Laurence A Wolsey. *Production planning by mixed integer programming*. Springer Science & Business Media, 2006.

[14] Wayne L Winston and Jeffrey B Goldberg. *Operations research: applications and algorithms*, volume 3. Thomson Brooks/Cole Belmont, 2004.