3.4 การประยุกต์ใช้โครงสร้างกราฟ

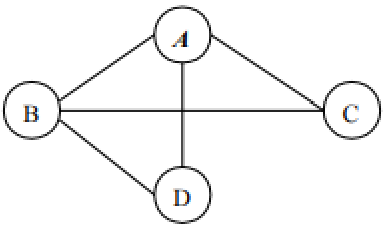
3.4.1 อัลกอริทึมการหาเส้นทางเดินที่ไปได้ทั้งหมด

ในการเดินทางไปที่ไหนสักแห่งหนึ่งนั้น เราไม่จำเป็นต้องเดินทางโดยใช้เส้นทางเดิมทุกครั้งนั่นเป็นเพราะเส้นทางที่ใช้เป็นประจำอาจจะขาด หรือ ชำรุดเสียหายขึ้นมาจึงทำให้ต้องใช้เส้นทางใหม่ แต่เราจะทราบได้อย่างไรว่าจะมีเส้นทางใดจะไปยังจุดหมายปลายทางเดียวกันได้อีก

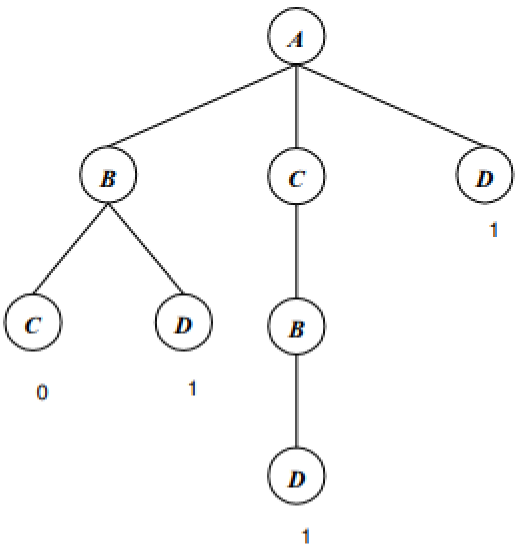
การหาเส้นทางที่เดินไปได้ทั้งหมดระหว่างจุด 2 จุดหรือระหว่างเมือง 2 เมืองจะทำให้เราทราบว่านอกจากเส้นทางที่ใช้อยู่เป็นประจำแล้วยังมีเส้นทางใดอีกบางซึ่งสามารถที่จะกระทำได้โดย

1. กำหนดจุดตั้งต้นให้เป็นจุดหลัก
2. พิจารณาว่ามีจุดใดเชื่อมต่อกับจุดหลักบ้างโดยที่ไม่ทำให้เกิดเป็นวงจร
3. ถ้าจุดที่เชื่อมต่อกับจุดหลักเป็นจุดปลายทางให้คิดเป็น 1 เส้นทาง
4. ถ้าจุดที่เชื่อมต่อกับจุดหลักไม่ใช่จุดปลายทางให้จุดที่เชื่อมต่ออยู่นี้ทำหน้าที่เป็น จุดหลักแทน แล้วทำซ้ำขั้นตอนที่ 2 จนพบจุดปลายทาง

ตัวอย่างที่ 1 จากกราฟรูปที่ 3.70 มีเส้นทางเดินจาก A ไปถึง D ที่เส้นทาง



รูปที่ 3.70 โครงสร้างกราฟ



รูปที่ 3.71 การหาเส้นทางที่ไปได้

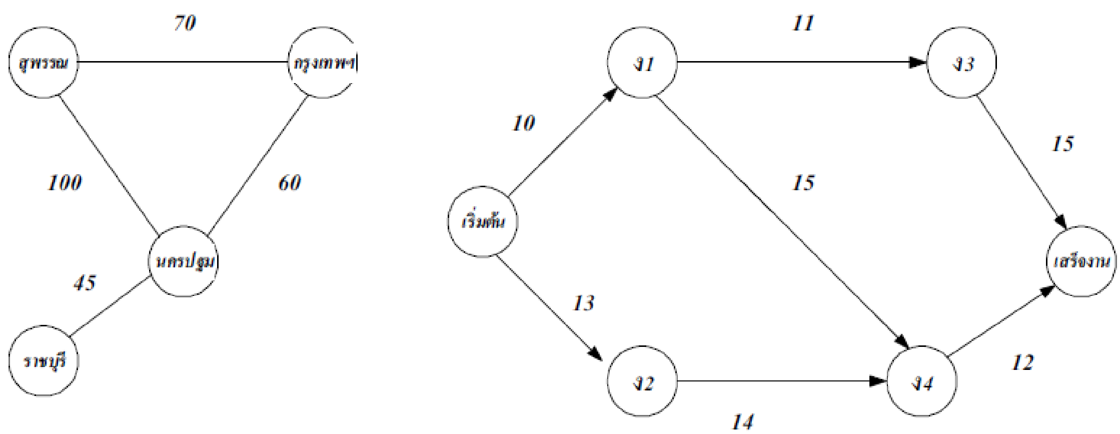
รูปที่ 3.71 แสดงการหาเส้นทางที่ไปได้ทั้งหมดจาก A ไปถึง D โดยเริ่มที่ การ กำหนดให้ A เป็นจุดหลัก จะเห็นได้ว่ามีจุดเชื่อมกับ A อยู่ 3 จุดด้วยกันคือ จุด B,C,D จึงทำให้แยกออกเป็น 3 ทางด้วยกัน จากนั้นตรวจสอบว่าจุดที่มาเชื่อมต่อนั้นมีจุดที่เป็นจุด ปลายทางหรือไม่ ปรากฏว่ามีจุด D เชื่อมต่ออยู่ก็ทำการนับเป็น 1 เส้นทางคือ A-D จากนั้น กำหนดให้ B ทำหน้าที่เป็นจุดหลักก็จะพบอีก

2 จุดที่มาเชื่อมต่ออยู่คือจุด C และ D ก็จะพบ อีก 1 เส้นทางคือ A-B-D ส่วน A-B-C ไม่สามารถที่จะไป ยังจุดปลายทางได้เพราะจุดที่ ต่อกับ C คือ A ซึ่งจะทำให้ทางเดินเกิดเป็นวงจรขึ้นคือ A-B-C-A

จากนั้นมาพิจารณาจุด C ที่มาจาก A จะพบว่ามีทางไปยังจุด B แล้วเชื่อมต่อไปยัง จุด D ทำให้เราได้อีก 1 เส้นทางคือ A-C-B-D

3.4.2 เวทเต็ดกราฟ (Weighted Graph)

เวทเต็ดกราฟ หรือเรียกอีกชื่อว่าข่ายงาน (Network) คือกราฟที่มีตัวเลขกำกับเส้นโยง ซึ่งตัวเลขนั้นอาจแสดงถึงระยะทาง เวลา ค่าใช้จ่าย หรือ ความจุของน้ำ เป็นต้น รูปที่ 3.72 แสดงตัวอย่างของเวทเต็ดกราฟ รูปที่ 3.72 a) เป็นกราฟแสดงระยะทาง (ก.ม.) ระหว่าง จังหวัด 3 จังหวัด รูปที่ 3.72 b) แสดงระยะเวลา(วัน) ที่ใช้ทำงานจากเริ่มต้นจนถึงเสร็จงาน ซึ่งแต่ละจุดแทนเหตุการณ์และเส้นโยงแทนจำนวนวันที่ใช้เช่นจากจุดเริ่มต้นใช้เวลา 10 วัน ทำ ง1 ใช้เวลา 13 วัน ทำ ง2 และเมื่อ ง1 เสร็จแล้วเริ่มทำงาน ง4 จนเสร็จใช้เวลา 15 วัน เมื่อเสร็จ ง2 ทำ ง4 จนเสร็จใช้เวลา 14 วัน เป็นต้น



a) b)

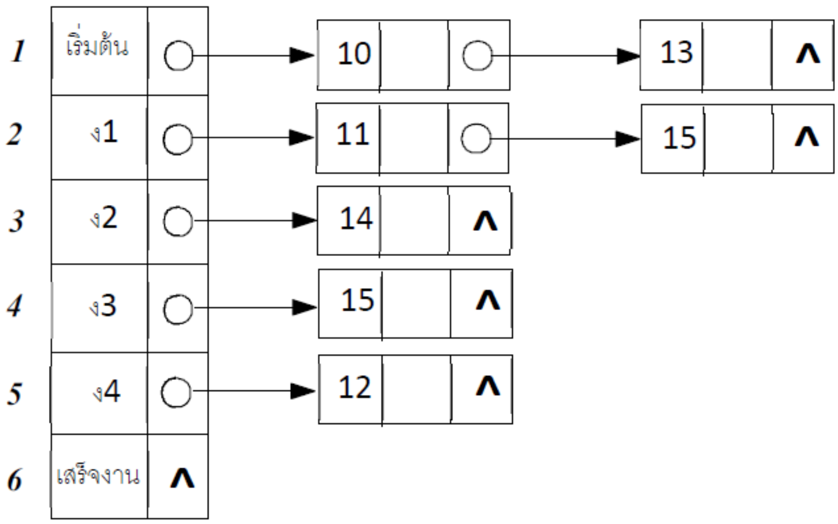
รูปที่ 3.72 เวทเต็ดกราฟ

การแทนที่เวทเต็ดกราฟ

เวทเต็ดกราฟ ก็เช่นเดียวกับกราฟสามารถใช้เมตริกซ์หรือลิงค์ลิสต์แทนที่ได้ ถ้าใช้เมตริกซ์ ค่าในเซลในเมตริกซ์เป็นเวท (Weight) ของกราฟ รูป 3.73 a) เป็นเมตริกซ์แทน กราฟรูป 3.72 b) ส่วน รูป 3.73 b) เป็นการแทนกราฟเดียวกันด้วยลิงค์ลิสต์

$$
\begin{array}{c}
 \\
\text{เริ่มต้น} \\
 \\
\text{ง1} \\
 \\
\text{ง2} \\
\text{ง3}
\end{array}
\begin{array}{c}
\begin{array}{cccccc}
\text{เริ่มต้น} & \text{ง1} & \text{ง2} & \text{ง3} & \text{ง4} & \text{เสร็จงาน}
\end{array} \\
\begin{bmatrix}
0 & 10 & 13 & 0 & 0 & 0 \\
0 & 0 & 0 & 11 & 15 & 0 \\
0 & 0 & 0 & 0 & 14 & 0 \\
0 & 0 & 0 & 0 & 0 & 15 \\
0 & 0 & 0 & 0 & 0 & 12 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

(a)



(b)

รูปที่ 3.73 การแทนเวทเต็ดกราฟ

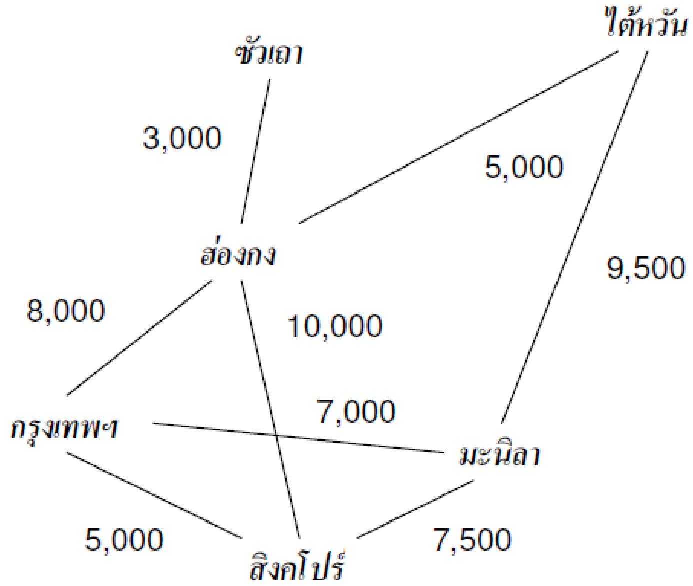
3.4.3 อัลกอริทึมการหาเส้นทางที่สั้นที่สุด (Shortest Path)

3.4.3.1 การหาทางเดินที่สั้นที่สุด แบบทางเดียว

ในหัวข้อนี้ เราจะศึกษาปัญหาของการหาเส้นทางเดินในกราฟแบบไม่มีทิศทาง โดย สมมติว่า มีกราฟแบบไม่มีทิศทาง G = (V,E) อันหนึ่ง ซึ่งแต่ละเส้นโยงไม่มีค่าเป็นลบ ใน จุดใดจุดหนึ่ง จะถูกกำหนดให้เป็นจุดเริ่มต้น (SOURCE) ปัญหาของเราก็คือ พิจารณา Cost (อาจจะเป็น ค่าใช้จ่าย ระยะทาง หรือเวลา ฯลฯ) ของระยะทางที่น้อยที่สุด จากจุดเริ่มต้น ไปยังจุดอื่น ๆ ทุก ๆ จุดใน V โดยที่ความยาวของเส้นทาง (Length of path) ก็คือผลรวม ของ Cost บนเส้นทางนั้น

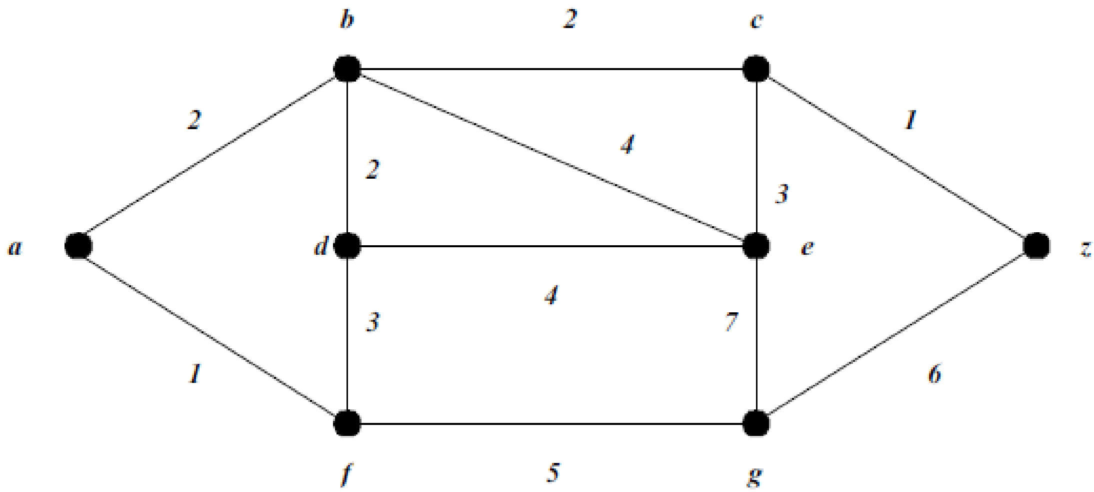
การหาทางเดินสั้นที่สุด (Shortest Path) ระหว่างคู่จุดใด ๆ ในข่ายงาน เป็น ปัญหาซึ่งเกิดขึ้นบ่อยมาก โดยในที่นี้ความยาวของทางเดิน ถูกกำหนดให้เป็นผลรวมของ

น้ำหนักของเส้นโยงซึ่งประกอบกันเป็นทางเดินนั้น เช่นเส้นทางการบินของสายการบินหนึ่ง ถูกจำลองได้โดยใช้ขายงานซึ่งมีจุดแทนเมือง เส้นโยงระหว่างเมืองแทนเส้นทางบิน และมี น้ำหนักของเส้นโยงซึ่งอ ซึ่งอาจแทนระยะทาง เวลาในการเดินทางหรือค่าโดยสารของเส้นทางบิน การหาทางเดินซึ่งมีผลรวมของน้ำหนักน้อยที่สุด (เรียกว่าทางเดินที่สั้นที่สุด) ระยะเมืองสองเมือง จะบอกถึงคุณสมบัติบางประการของทางเดินที่หาได้ ซึ่งขึ้นอยู่กับความหมายของ น้ำหนักของเส้นโยง อาจเป็นระยะทางที่สั้นที่สุด เวลาการเดินทางเร็วที่สุด หรือราคาค่า โดยสารรวมถูกที่สุด เป็นต้น (ดูรูปที่ 3.74)



รูปที่ 3.74 ข่ายงานแสดงราคาค่าโดยสารของสายการบินหนึ่ง

ตัวอย่างที่ 2 ถ้าเราให้เมืองแทนจุด และถนนที่เชื่อมระหว่างเมืองแทนด้วยด้านเมื่อเรากำหนดความยาว ของ ถนนให้ เราจะได้กราฟที่มีน้ำหนัก ดังรูปที่ 3.75 เลขที่กำกับบนด้าน คือความยาวของ ถนนที่เชื่อมระหว่างเมือง



รูปที่ 3.75 กราฟแสดงเส้นทางระหว่างเมือง

บ่อยครั้งที่เราใช้น้ำหนักแทนค่าใช้จ่าย เช่น ถ้าจุดแทนเมือง และด้านแทนถนนที่ กำลังก่อสร้าง น้ำหนักของด้านด้านหนึ่งอาจแทนค่าใช้จ่ายในการสร้างถนนนั้น น้ำหนักของ กราฟรูปหนึ่ง คือ ผลบวกของน้ำหนักของด้านทุกด้านในกราฟ น้ำหนักของทางเดินปกติจะ กล่าวว่าเป็น ความยาวของทางเดิน ในกราฟที่มีน้ำหนัก เรามักต้องการหา **ทางเดินที่สั้นที่สุด** สักเส้นทางหนึ่ง (หมายถึงทางเดินเส้นทางหนึ่งที่มีความยาวสั้นที่สุด) ระหว่างจุด 2 จุด เรา กล่าวถึงแบบการคำนวณของ ดิสตรา (Dijkstra) ซึ่งมีประสิทธิภาพในการแก้ปัญหานี้ ในตอนนี้เราให้ G แทนกราฟที่มีน้ำหนักและติดต่อกัน สมมติว่าน้ำหนักเป็นจำนวนบวกและ ต้องการหาทางเดินที่สั้นที่สุด เส้นทางจากจุด a ไปยังจุด z

แบบการคำนวณของดิสตราเป็นการกำหนดค่าประจำให้จุด ให้(v) แทนค่าประจำ จุด v ณ ขั้นใด ๆ จุดบางจุดได้ค่าประจำจุดชั่วคราวและจุดที่เหลือจะได้ค่าประจำจุดถาวร ให้ T แทนเซตของจุดที่ได้รับค่าประจำจุดชั่วคราว ในการแสดงแบบการคำนวณนี้ เราจะวงกลม จุดที่มีค่าประจำจุดถาวร เราจะแสดงต่อไปว่า L(v)เป็นค่าประจำจุดถาวรของจุด v จะได้ว่า L(v) เป็นความยาวของทางเดินที่สั้นที่สุดเส้นหนึ่งจากจุด a ไป ยังจุด v เมื่อเริ่มต้นทุกจุดมีค่า ประจำจุดชั่วคราว แต่ละการทำซ้ำกันของแบบการคำนวณ จะเปลี่ยนสภาพของค่าประจำจุด ค่าหนึ่งจากชั่วคราวเป็นถาวร ดังนั้นเราอาจจบแบบการคำนวณได้เมื่อ z ได้รับค่าประจำจุดถาวร ณ จุดนี้ L(z) จะทำให้ความยาวของทางเดินที่สั้นที่สุดเส้นหนึ่งจากจุด a ไปยังจุด z

อัลกอริทึมในการหาทางเดินสั้นที่สุดบนข่ายงานที่จะนำเสนอต่อไปนี้ ได้ถูกคิดขึ้น โดยนักคณิตศาสตร์ชาวดัทช์ชื่อดิสตรา (Dijkstra) เมื่อปีค.ศ. 1959

แบบการคำนวณหาทางเดินที่สั้นที่สุดของดิสตรา

แบบการคำนวณนี้หาความยาวของทางเดินที่สั้นที่สุดจากจุด a ไปยังจุด z ในกราฟที่ มีน้ำหนักและติดต่อกัน น้ำหนักของด้าน (i, j) ซึ่งคือ w(i, j) > 0 และค่าประจำจุด x คือ L(x) เมื่อจบการคำนวณ L(z) เป็นความยาวของทางเดินที่สั้นที่สุดจากจุด a ไปยังจุด z

1. [เริ่มต้น] ให้ L(a) := 0 สำหรับจุดทุกจุด x <> 2 ให้ L(x) := ∞ ให้ T เป็นเซต ของจุดทั้งหมดในกราฟ

2. [ทำเสร็จหรือยัง] ถ้า z ไม่อยู่ใน T หยุดการคำนวณได้ (L(z) เป็นความยาวของ ทางเดินทีสั้นที่สุดเส้นหนึ่งจากจุด 3 ไปยังจุด z)

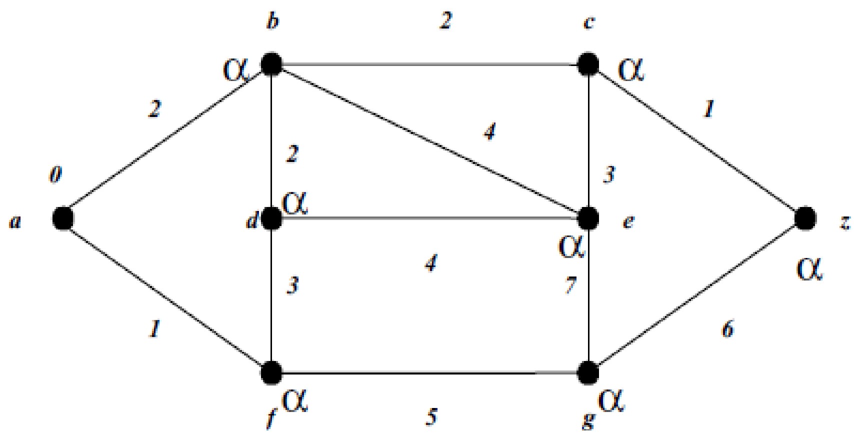
3. [นำจุดต่อไปเข้ามา] เลือก V อยู่ใน T ที่มีค่า (V) ต่ำสุดให้ T := T - {v}

4. [ปรับปรุงค่าประจำจุดใหม่] สำหรับแต่ละจุด x อยู่ใน T ที่อยู่ติดกับ v ให้ L(x) := min { L(x), L(v) + w(v, x) } ย้อนกลับไปขั้นที่ 2

ตัวอย่าง 3 จงหาความยาวของทางเดินที่สั้นที่สุดของกราฟรูปที่ จากจุด a ไปยังจุด z โดยใช้แบบการคำนวณของดิสตรา

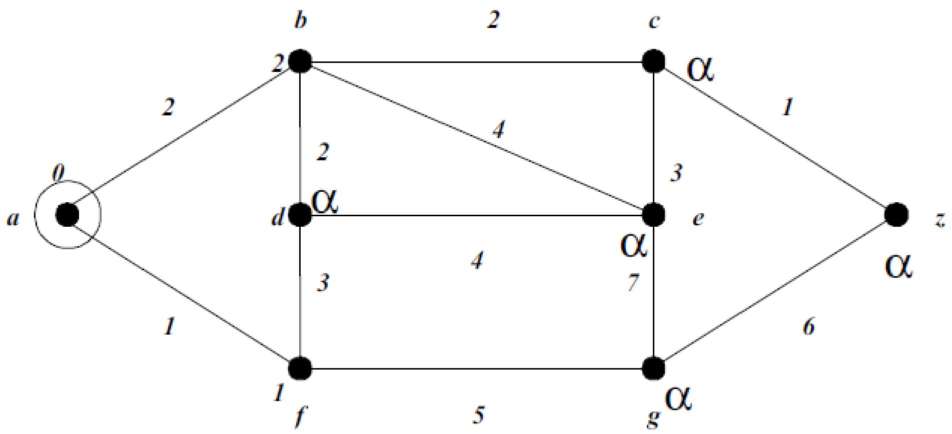
วิธีทำ

เนื่องจากจุดใน T ยังไม่มีวงกลม และมีค่าประจำจุดชั่วคราว (จุดที่มีวงกลมจะมีค่า ประจำจุดถาวร) รูปที่ 3.76 แสดงผลลัพธ์ที่ได้จากการทำขั้นที่ 1



รูปที่ 3.76 ผลลัพธ์ที่ได้จากขั้นตอนที่

ในขั้นที่ 2 นี้ z ยังไม่มีวงกลม จึงต่อไปข้อที่ 3 เลือกจุด a ซึ่งเป็นจุดที่ไม่มีวงกลมเมีค่าประจำจุดต่ำสุด เราวงกลมจุด a ดังรูปที่ 3.77



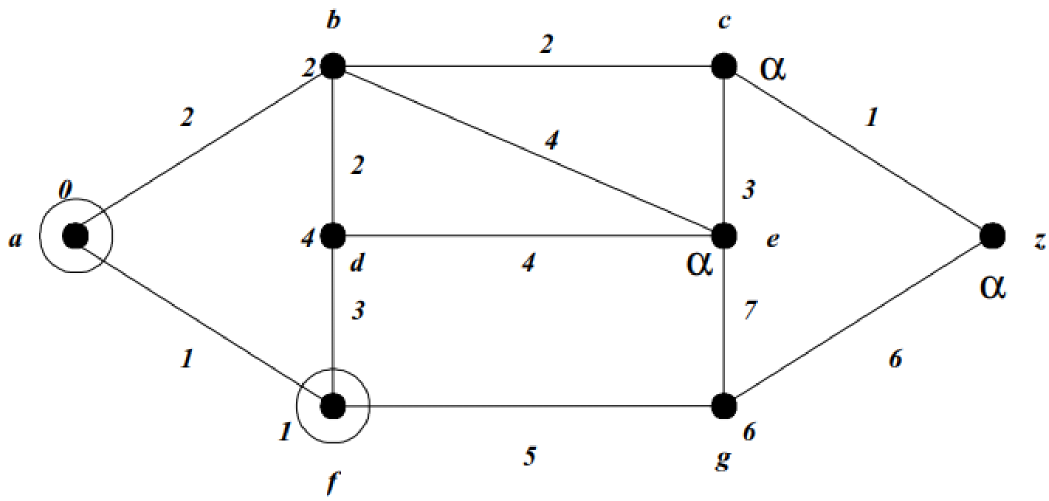
รูปที่ 3.77 กำหนดจุดเริ่มต้นให้มีน้ำหนักต่ำสุด

ในขั้นที่ 4 ปรับปรุงแต่ละจุดที่ยังไม่มีวงกลม ในที่นี้ b และ f ที่อยู่ติดกับ a โดยเราจะให้ค่าประจำจุดใหม่ดังนี้

L(b) := min { ∞, 0+2 } = 2

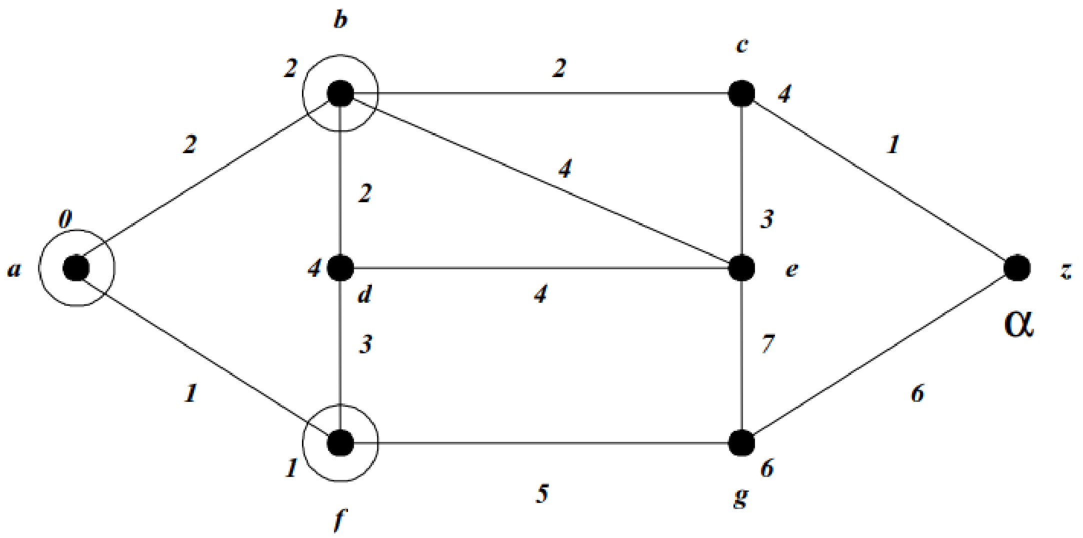
L(f) := min { ∞, 0+1 } = 1

ดูรูปที่ 3.77 แล้วย้อนกลับไปที่ขั้นที่ 2 เพราะ z ยังไม่มีวงกลมและมีค่าประจำจุดที่ ยังไม่มีวงกลมและ<u>มีค่าประจำจุดต่ำสุด แล้ววงกลมจุด *f*</u> ดังรูปที่ 3.78

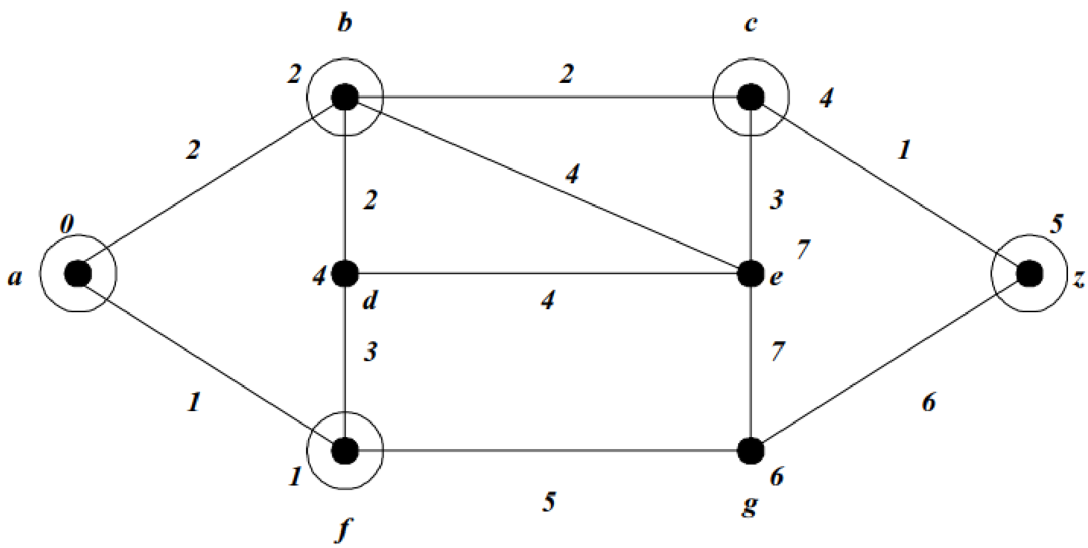


รูปที่ 3.78 วงกลมที่จุด f ซึ่งเป็นจุดที่มีค่าประจำต่ำสุด

ต่อไปขั้นที่ 4 ปรับปรุงค่าประจำจุดที่ยังไม่มีวงกลม d และ g ที่อยู่ติดกับ f ได้ดังรูป ที่ 3.78 เสร็จแล้วลองตรวจสอบกับค่าที่แสดงไว้ในรูปที่ 3.79 และ 3.80



รูปที่ 3.79 วงกลมที่จุด b



รูปที่ 3.80 วงกลมที่จุด C และ Z ตามลำดับC

จบแบบการคำนวณ เมื่อ z ถูกวงกลม (เพราะที่ z มีค่า 5 น้อยกว่าที่ e และ g) นั่น คือ ทางเดินที่สั้นที่สุดเส้นทางหนึ่ง คือ (a, b, c, z) มีความยาวเท่ากับ 5

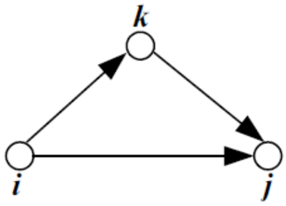
3.4.3.2 การหาทางเดินที่สั้นที่สุดแบบหลายทาง

ในหัวข้อที่ผ่านมา เราได้รู้วิธีการแก้ปัญหาทางเดินที่สั้นที่สุดแบบทางเดียวไปแล้ว ในตอนนี้ เราจะมาศึกษาวิธีการคล้ายๆ กันนี้ แต่ซับซ้อนกว่าเล็กน้อย สมมติว่าเรามีกราฟแบบ มีทิศทางที่กำหนดค่าของเวลาการเดินทางจากเมืองหนึ่งไปยังอีกเมืองหนึ่งมาให้ และเราต้องการสร้างตารางที่ให้ค่าเวลาที่สั้นที่สุดจากเมืองใดเมืองหนึ่ง ไปยังอีกเมืองใดเมืองหนึ่ง เพื่อที่จะกำหนดปัญหาได้อย่างชัดเจน สมมติว่าเรายังมีกราฟแบบมีทิศทาง G = (V, E) ซึ่งใน แต่ละ Edge v ⟶ w จะมีค่าที่ไม่เป็นลบของ cost C[v, w] ปัญหานี้ก็คือการหาแต่ละ คู่ลำดับของ Vertex(v, w) ที่มีความยาวน้อยที่สุดทุกๆ เส้นทางจาก v ไปยัง w เราจะ สามารถแก้ปัญหานี้โดยใช้ Dijkstra s Algorithm กับทุกๆ Vertex ที่เป็นจุดเริ่มต้นก็ได้ แต่ วิธีการที่ใช้แก้ปัญหานี้โดยตรงที่นิยมใช้ก็คือวิธีการของนาย R. W. Floyd เพื่อความสะดวก เราจะกำหนดหมายเลขเวอร์ทิกซ์ใน V คือ 1, 2,..., n อัลกอริทึมของนาย Floyd หรือ Floyd' s algorithm นี้จะใช้เมตริกซ์ A ที่มีขนาด n x n ในการคำนวณหาระยะทางที่สั้นที่สุดเราจะเริ่มจากให้ A[i,j] = C[i,j] สำหรับทุกๆ i≠j ถ้าไม่มี Edgeจาก i ไปยัง j เราจะให้ C[i,j] = α และทุกๆ สมาชิกในแนวทแยงของเมตริกซ์จะเป็น 0

ดังนั้นเราสามารถทำการ**<u>วนรอบ</u>**เมตริกซ์ A เป็นจำนวน n ครั้ง และหลังจากทำไป เป็นครั้งที่ k^{th} A[i,j] จะมีค่าเป็นระยะทาง**<u>น้อยที่สุด</u>**จาก Vertex i ไปยัง Vertex j จะผ่าน Vertex หมายเลขใดๆ ที่**<u>น้อยกว่าหรือเท่ากับ</u>** k ในการวนรอบที่ k^{th} ใดๆ เราจะใช้สูตรต่อไปนี้คำนวณหาเมตริกซ์ A

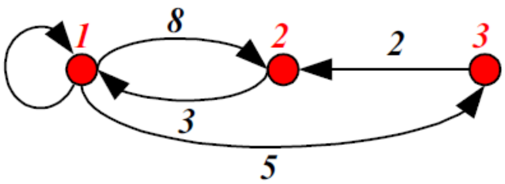
$$A_k[i,j] = \min \left[\begin{array}{l} A_{k-1}[i,j] \\ \\ A_{k-1}[i,k] + A_{k-1}[k,j] \end{array} \right.$$

ตัวห้อย (subscript) k จะบ่งถึงค่าของเมตริกซ์ A หลังจากผ่านการวนรอบการ กระทำครั้งที่ k^{th} ในรูปที่ 3.80 จะใช้อธิบายให้เห็นถึงสูตรนี้ จากรูป ในการคำนวณหาค่า A_k [i,j] เราจะเปรียบเทียบค่า A_k [i,j] ซึ่งเป็นค่าของการเดินทางจาก 1 ไปยัง 3 โดยไม่ผ่าน เวอร์ทิกซ์หมายเลข k หรือมากกว่า กับค่าของ A_{k-1} [i,k] + A_{k-1} [k,j] ซึ่งเป็นค่าของการ เดินทางจาก i ไปยัง k และหลังจากนั้นจาก k ไปยัง j โดยไม่ผ่านเวอร์ทิกซ์หมายเลขมากกว่า k ถ้าการเดินทางผ่าน Vertex k นี้ให้ผลการเดินทางน้อยกว่าที่ได้จาก A_{k-1} [i,j] ดังนั้นเราจะ เลือกทางเดินนี้สำหรับ A_{k-1} [i,j]



รูปที่ 3.81 แสดงให้เห็นการเดินทางจาก Vertex ไปยัง Vertex j โดยมี Vertex k ร่วม อยู่ด้วย

พิจารณากราฟแบบมีทิศทางตามรูปที่ 3.82 ค่า A_0 [i,j] ในตารางนั้นก็หมายถึงการที่ เราเดินทางจาก Vertex i ไปยัง Vertex j โดยไม่ผ่านเวอร์ทิกซ์ใดๆ เลย ดังนั้นค่าจาก Vertex 2 ไปยัง 3 จึงเท่ากับอินพินิต แต่พอต่อมาเมื่อเราวนรอบ 3 รอบ เราจะได้ว่า A_3[i,j] จะมีค่า บางค่าที่ถูกลดลงมาได้ อันเนื่องจากเราสามารถเดินทางผ่านเวอร์ทิกซ์ได้ทั้ง 3 Vertexนั้นทำให้ A_3 [1,2] = 7 หรือ A_3 [3,1] = 5



รูปที่ 3.82 แสดงกราฟแบบมีทิศทางที่กำหนดค่าบน Edge อันหนึ่ง

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

A_0 [i,j] แสดงภาวะเริ่มแรก

	1	2	3
1	0	8	5
2*	3	0	**8**
3	∞	2	0

A_1 [i,j) แสดงการเปลี่ยนของแถว 2

	1	2	3
1	0	8	5
2	3	0	8
3*	**5**	2	0

A_2 [i,j] แสดงการเปลี่ยนของแถว 3

	1	2	3
1*	0	**7**	5
2	3	0	8
3	5	2	0

A_3 [i,j] แสดงการเปลี่ยนของแถว 1

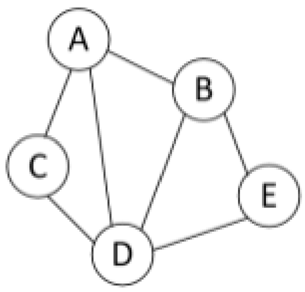
รูปที่ 3.83 แสดงค่าของเมตริกซ์ A ที่ได้รับการวนรอบ 1 รอบใน 3 แถว

สรุป

โครงสร้างข้อมูลกราฟมีลักษณะคล้ายคลึงกับโครงสร้างแบบต้นไม้ คุณสมบัติของ กราฟที่แตกต่างจากต้นไม้ก็คือ ในกรณีที่กราฟไม่มีราก และอาจมีเส้นเชื่อมหลายเส้นระหว่าง 2 จุด กราฟเป็นโครงสร้างข้อมูลที่มีประโยชน์ เพราะสามารถนำมาใช้แก้ปัญหาในการ ทำงานหลายด้าน เช่น สามารถใช้กราฟพิจารณาว่าวงจรหนึ่งสามารถนำไปใช้กับแผงวงจร ไฟฟ้าแบบระนาบ ในกรณีของวิชาเคมีการใช้กราฟทำให้สามารถแยกระหว่างสารเคมี 2 ชนิดที่มีสูตรโมเลกุลเหมือนกันแต่มีโครงสร้างต่างกัน กรณีของเครือข่ายคอมพิวเตอร์สามารถ พิจารณาว่าเครื่องคอมพิวเตอร์ 2 เครื่องต่อเชื่อมกันในกรณีของการสื่อสารใช้โมเดลของ เครือข่ายซึ่งแทนด้วยกราฟที่ประกอบด้วยเส้นเชื่อมน้ำหนัก สามารถใช้หาระยะทางสั้นที่สุด ในการเดินทางระหว่างเมือง 2 เมือง หรือแม้แต่ในการหาสมมติฐานด้านงานวิจัย (Operation Research) และงานทางธุรกิจ ก็สามารถใช้โครงสร้างกราฟในการหาคำตอบตามเงื่อนไขใน แต่ละกรณี เพื่อหาคำตอบที่ดีที่สุดได้

แบบฝึกหัดที่ 3

1. จงนิยามความหมายของคำต่อไปนี้
   - ก. ระดับของโหนด (Level)
   - ข. โหนดที่เป็นใบ (Leaf Node)
   - ค. ดีกรีของโหนด (Degree)
2. การแทนโครงสร้างต้นไม้แบบเรียงโหนดมีลักษณะอย่างไร ให้ยกตัวอย่างพร้อมทั้งเขียนสมการและอสมการอย่างครบถ้วน ?
3. จงสร้างต้นไม้แบบสมบูรณ์ (Complete Binary Tree) แล้วนำชื่อและนามสกุลภาษาอังกฤษของตนเองมาใส่ โดยเมื่อท่องแบบ In-Order จะได้เป็นชื่อและนามสกุลของตนเองออกมาอย่างถูกต้อง ?
4. จงสร้างต้นไม้ไบนารีจำนวน 1 ต้น โดยให้มีจำนวนโหนดไม่ต่ำกว่า 12 โหนด และทำการท่องต้นไม้ให้ครบทั้ง 3 รูปแบบ (Pre order, In order, Post order) แล้วแสดงผลที่ได้จากการท่องในแต่ละแบบ ?
5. จากกราฟดังที่แสดง จงแสดงวิธีการแทนทั้ง 3 แบบ



ก. Adjacency Matrix

ข. Adjacency List

ค. Adjacency Multi list