การจัดการจราจรข้อมูลในโครงข่าย IP ด้วยวิธีจุดภายใน

IP Network Traffic Engineering by Interior Point Method

ณัฐชามญฑ์ ศรีจำเริญรัตนา

โปรแกรมวิชาคอมพิวเตอร์ธุรกิจ คณะวิทยาการจัดการ

มหาวิทยาลัยราชภัฏนครปฐม ตำบลนครปฐม อำเภอเมือง จังหวัดนครปฐม 73000

ธนโชติ ธิติวิเชียรเลิศ, ณัฐดนัย สุขแสง และกายรัฐ เจริญราษฎร์*

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ กำแพงแสน

มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน อำเภอกำแพงแสน จังหวัดนครปฐม 73140

Natchamol Srichumroenrattana

Department of Business Computer, Faculty of Management Science,

NakhonPathom Rajabhat University, Muang, Nakhon Pathom, 73000

Thanachot Thitivichienlert, Natdanai Suksaeng and Kairat Jaroenrat*

Department of Computer Engineering, Faculty of Engineering at Kamphaeng Saen,

Kasetsart University, Kamphaeng Saen Campus, Kamphaeng Saen, Nakhon Pathom, 73140

บทคัดย่อ

ปัจจุบัน ระบบเครือข่ายมีบทบาทสำคัญมากขึ้นเนื่องจากมีการใช้งานคอมพิวเตอร์อย่างแพร่หลาย จึงเกิดความต้องการที่จะเชื่อมต่อคอมพิวเตอร์เหล่านั้นเข้าถึงกันเป็นเครือข่ายคอมพิวเตอร์เพื่อเพิ่มความสามารถของระบบต่าง ๆ ให้มีประสิทธิภาพสูงขึ้น โดยการส่งข้อมูลข้ามเครือข่ายภายในโครงข่ายขนาดใหญ่ เช่น อินเทอร์เน็ต จะต้องมีโพรโทคอลชี้เส้นทาง ที่ซึ่งในปัจจุบันนิยมใช้กันอย่างแพร่หลายมากที่สุด คือ OSPF เนื่องจากโครงข่ายโอเอสพีเอฟใช้อัลกอริทึมในการค้นหาเส้นทางด้วยตัวเอง โดยคำนวณจากเส้นทางที่มีค่าน้ำหนักรวมน้อยที่สุด และด้วยการใช้ค่าน้ำหนักมาตรฐานของช่องสัญญาณอาจทำให้การจัดเส้นทางให้แก่การไหลของข้อมูลไม่ตรงกับความต้องการและเกิดความคับคั่งเกินไปในบางเส้นทาง ผู้วิจัยจึงนำวิธีจุดภายในมาทดลองประยุกต์ใช้ในการหาชุดค่าน้ำหนักที่เหมาะสมของสายสัญญาณ โดยวิธีจุดภายในนั้นเป็นวิธีที่สามารถแก้ไขปัญหาที่ซับซ้อนสูงและให้ประสิทธิภาพที่ดีทั้งยังใช้เวลาประมวลผลไม่นานมาก จากการทดลองในส่วนแรกพบว่าประสิทธิภาพการไหลของการกำหนดค่าน้ำหนักด้วยวิธีจุดภายในเมื่อเทียบกับวิธีการใช้ค่าน้ำหนักมาตรฐานที่โครงข่ายเราเตอร์ขนาด 10 โหนด มีค่าประสิทธิรูปที่ใกล้เคียงกัน แต่เมื่อเพิ่มความขนาดของโครงข่ายเป็น 14 โหนด พบว่าวิธีจุดภายในจะให้ค่าประสิทธิรูปที่ดีกว่าเมื่อโครงข่ายมีความซับซ้อนสูงโดยเฉพาะที่โหนดดีกรีเท่ากับ 2.0 จะมีประสิทธิภาพดีกว่าวิธีการใช้ค่าน้ำหนักมาตรฐานถึง 132.96 % ในการทดลองส่วนที่สองผู้วิจัยพบว่าการกำหนดค่าน้ำหนักด้วยวิธีจุดภายในนี้ให้ค่าประสิทธิภาพการไหลของ

***ผู้รับผิดชอบบทความ :** kairat.j@ku.ac.th DOI 10.14456/tstj.2015.56

ข้อมูลด้อยกว่าวิธีซิมเพล็กซ์เล็กน้อย แต่เวลาในการประมวลผลของวิธีจุดภายในจะเร็วกว่าวิธีซิมเพล็กซ์มาก โดยเฉพาะในกรณีของโครงข่ายที่ซับซ้อนสูง ซึ่งจากการทดสอบกับโครงข่ายขนาด 10 โหนด ที่โหนดดีกรีที่ 2.0 พบว่าจะได้ประสิทธิภาพการไหลที่เป็น 82.35 % ของวิธีซิมเพล็กซ์ แต่มีประสิทธิภาพด้านเวลาดีกว่าถึง 63 เท่า

คำสำคัญ : วิธีจุดภายใน; การจัดการจราจรข้อมูล; โครงข่ายไอพี

Abstract

Currently, the computer networks are very important because there is the widespread use of computers and we need to connect them together to enhance the efficiency of the computer systems. Sending data across the network, such as Internet, we need routing protocols. The most widely used routing protocol is OSPF (open shortest path first) because OSPF uses a shortest path algorithm to find the best paths by choosing the path with the least weight. By using the default weight, OSPF may make the path of the data flow not match the user needs and make some congested link. Then we apply an interior point method to determine the appropriate links' weights from the knowledge that Interior point method is the high performance method to resolve complex problems with not very long calculation time. In first part of our experiment, we use the default weight and compare with the interior point method and found that the efficiency of 10-node networks similar. But in 14-nodes networks with high node degree (2.0) experiment, the interior point method provides 132.96 % better performance than the default weight method. In the second part of our experiment, we compare the simplex method to our interior point method with the network of node degree 2.0 and the number of node is 10. We found that, the data flow performance of the networks with interior point method's weights is nearly to the Simplex method but the time performance if interior point method is 63-fold better.

Keywords: interior point; traffic engineering; IP network

1. บทนำ

ปัจจุบันระบบเครือข่ายคอมพิวเตอร์โดยเฉพาะโครงข่ายอินเตอร์เน็ตมีบทบาทสำคัญมากขึ้น เพราะมีการใช้งานคอมพิวเตอร์กันอย่างแพร่หลายและเกิดความต้องการที่จะเชื่อมต่อกันเป็นระบบเครือข่ายเพื่อเพิ่มความสามารถของระบบให้มีประสิทธิภาพสูงขึ้นและลดต้นทุนของระบบโดยรวมลง โดยอุปกรณ์สำคัญที่ทำหน้าที่ชี้เส้นทางเพื่อให้ส่งข้อมูลข้ามเครือข่ายในโครงข่ายไอพีหรือโครงข่ายอินเตอร์เน็ตได้ก็คือเราเตอร์ (router) ซึ่งมีโพรโทคอลในการชี้เส้นทาง (routing protocols) ที่ในปัจจุบันนิยมใช้อย่างแพร่หลายมากที่สุด คือ open shortest path first (OSPF) [1] เนื่องจากมีจุดเด่นในหลายด้าน เช่น การใช้ Dijkstra’s algorithm ในการค้นหาเส้นทางที่สั้นที่สุดด้วยตัวเอง การรับรู้ถึงการเปลี่ยนแปลงในรูปแบบการเชื่อมต่อหรือการเปลี่ยนแปลงของเส้นทางในระบบโครงข่ายได้อย่างรวดเร็ว แต่การใช้ OSPF อาจจะทำให้การจัดเส้นทางให้กับการไหลของข้อมูลไม่ตรงกับความต้องการและ

เกิดความคับคั่งเกินไป และการกำหนด routing metric ที่ตอบสนองความต้องการนั้นทำได้ยาก จึงทำให้โครงข่ายนั้นมีประสิทธิภาพด้อยกว่าที่ควรเป็น

เพื่อให้การจัดเส้นทางของข้อมูลในโครงข่ายสามารถทำงานได้อย่างมีประสิทธิภาพผู้จัดทำจึงใช้วิธีจุดภายใน[2] ในการคำนวณชุดค่าน้ำหนัก (weight) ของสายสัญญาณในโครงข่าย และนำโครงข่ายกับค่าน้ำหนักนี้ไปทดสอบและวิเคราะห์การไหลของข้อมูลในโครงข่ายเพื่อเปรียบเทียบประสิทธิภาพ [3,4] กับโครงข่ายที่หาชุดค่าน้ำหนักสายสัญญาณด้วยวิธีใช้ค่าน้ำหนักมาตรฐานและวิธีซิมเพล็กซ์

2. ทฤษฎีและหลักการที่เกี่ยวข้อง

การวิจัยเรื่องการใช้วิธีการโปรแกรมเชิงเส้นในการตั้งค่าน้ำหนักสายสัญญาณ (weight) ให้แก่โครงข่าย OSPF ผู้วิจัยได้ศึกษาหลักการของทฤษฎีและเทคโนโลยีต่าง ๆ ที่เกี่ยวข้องกับการพัฒนาระบบที่สามารถนำมาประยุกต์ใช้กับงานได้โดยแบ่งออกเป็นหัวข้อต่าง ๆ ดังต่อไปนี้

2.1 OSPF (open shortest path first)

OSPF เป็นโพรโทคอลชี้เส้นทางตัวหนึ่งนิยมใช้กันอย่างแพร่หลายมากที่สุดในระบบโครงข่าย เนื่องจากมีจุดเด่นในหลายด้าน เช่น การที่เป็นโพรโทคอลค้นหาเส้นทางแบบสถานะแต่ละอินเตอร์เฟส (link state) และการที่มีอัลกอริทึมในการค้นหาเส้นทางได้ด้วยตัวเองได้นี้ โอเอสพีเอฟจะใช้อัลกอริทึมของ Dijkstra เพื่อคำนวณเส้นทางที่สั้นที่สุด [1] โดยเบื้องต้นจะกำหนดค่าน้ำหนักมาตรฐาน [5] ของแต่ละสายสัญญาณตามสมการ $weight = \frac{10^8}{Bandwidth}$ (1)

2.2 การโปรแกรมเชิงเส้น

โปรแกรมเชิงเส้น (linear programming) [3] เป็นเทคนิคที่รู้จักกันแพร่หลายและเป็นส่วนหนึ่งของการวิจัยดำเนินงาน (operations research) ในหลาย ๆ ด้าน นักบริหาร วิศวกรหรือนักวิทยาศาสตร์ในหลายหน่วยงานได้ประยุกต์ใช้วิธีการทางโปรแกรมเชิงเส้นในการแก้ปัญหาทางการจัดสรรปัจจัยหรือทรัพยากร (allocating resource) โดยที่ปัจจัยหรือทรัพยากรมีความหมายรวมถึงวัตถุดิบ กำลังคน เวลา สถานที่ เงินตรา หรือความรู้ความสามารถต่าง ๆ ปัญหาการจัดสรรปัจจัยและทรัพยากรเกิดขึ้นเมื่อเราต้องการจัดสรรทรัพยากรที่มีอยู่จำกัดทั้งขนาด ปริมาณและขอบเขตของการใช้งาน เพื่อให้เกิดประโยชน์สูงสุด โดยรูปแบบแทนระบบทางคณิตศาสตร์ของโปรแกรมเชิงเส้นประกอบไปด้วย 2 ส่วน ดังนี้

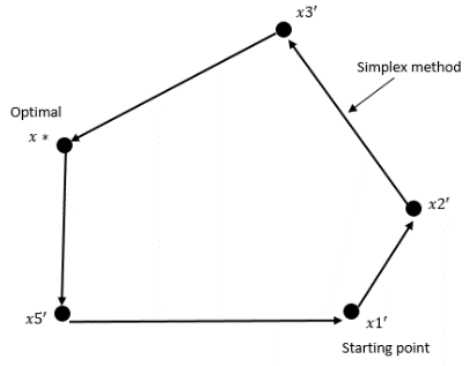
2.2.1 ฟังก์ชันวัตถุประสงค์ (objective function) คือ ฟังก์ชันที่ต้องการหาค่าต่ำสุดหรือค่าสูงสุด โดยจะต้องทำการกำหนดฟังก์ชันวัตถุประสงค์ให้อยู่ในรูปของสมการทางคณิตศาสตร์

2.2.2 เงื่อนไข (constraints) เป็นเงื่อนไขหรือข้อจำกัดของฟังก์ชันวัตถุประสงค์ ที่จะต้องทำตามหรือหลีกเลี่ยงไม่ได้ ปกติจะอยู่ในรูปแบบสมการหรืออสมการก็ได้

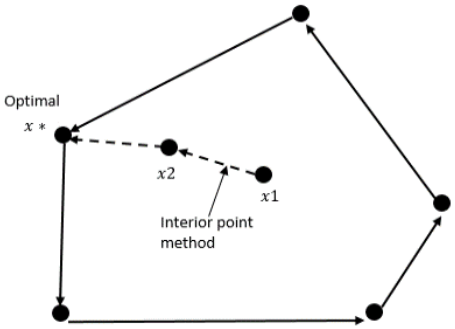
2.3 วิธีจุดภายใน

ในสมัยก่อนเครื่องคอมพิวเตอร์มีความเร็วไม่มากนัก ทำให้การแก้ไขปัญหาขนาดใหญ่ เช่น ปัญหาในระบบไฟฟ้ากำลัง ต้องใช้เวลาในการคำนวณแก้ปัญหาเป็นเวลานาน ดังนั้นนักวิจัยหลายท่านจึงได้ทำการค้นคว้าและวิจัยเพื่อหาอัลกอริทึมที่สามารถแก้ปัญหาขนาดใหญ่ได้รวดเร็วขึ้น จนกระทั่งนักวิจัยชื่อ Karmarkar [2] นำเสนอวิธีการแก้ปัญหาโปรแกรมเชิงเส้นขนาดใหญ่ที่เรียกว่าวิธีจุดภายใน ซึ่งมีความเร็วในการลู่เข้าคำตอบได้ดีกว่าวิธีโปรแกรมเชิงเส้นแบบเดิมที่เรียกว่าวิธีซิมเพล็กซ์ ทั้งนี้เป็นเพราะว่าลักษณะการเคลื่อนที่ของตัวแปรตามขอบเขตและจุดมุมของพื้นที่คำตอบที่เป็นไปได้ จนกว่าจะได้คำตอบที่ดีที่สุด แต่สำหรับวิธีจุดภายในตัวแปรจะเคลื่อนที่อยู่ภายในพื้นที่

คำตอบที่เป็นไปได้จนกว่าจะได้คำตอบที่ดีที่สุด ส่งผลทำให้วิธีจุดภายในใช้จำนวนรอบในการทำซ้ำเพื่อเข้าสู่คำตอบที่ดีที่สุดน้อยกว่าวิธีซิมเพล็กซ์



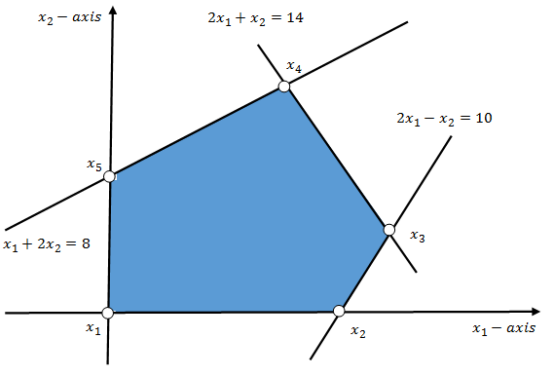
รูปที่ 1 ลักษณะการลู่เข้าสู่คำตอบของวิธีซิมเพล็กซ์ (ที่มา : http://home.ubalt.edu/ntsbarsh /opre640a/nonlinear.htm)



รูปที่ 2 ลักษณะการลู่เข้าสู่คำตอบของวิธีจุดภายใน (ที่มา: http://home.ubalt.edu/ntsbarsh /opre640a/nonlinear.htm)

จากรูปที่ 1 และ 2 เป็นการแสดงลักษณะการค้นหาคำตอบของวิธีซิมเพล็กซ์และวิธีจุดภายในโดยเมื่อนำมาเปรียบเทียบกันแล้ว วิธีจุดภายในจะเริ่มต้นค้นหาคำตอบจากภายในพื้นที่คำตอบที่เป็นไปได้ แตกต่างจากวิธีซิมเพล็กซ์ ที่เริ่มค้นหาคำตอบจากขอบพื้นที่คำตอบที่เป็นไปได้ ดังนั้นทำให้วิธีจุดภายในจะค้นหาคำตอบของปัญหาเชิงเส้นขนาดใหญ่ได้เร็วกว่าวิธีซิมเพล็กซ์

วิธีจุดภายในเป็นวิธีการหนึ่งที่ใช้เทคนิคการกระทำซ้ำ (interative technique) ในการหาคำตอบโดยสมมุติ โดยประกอบไปด้วยการกำหนดฟังก์ชันวัตถุประสงค์ เช่น minimize f(x)=5x_1+2x_2+4x_3 และสมการเงื่อนไข เช่น x_1+2x_2≤8, 2x_1+x_2≤ 14, 2x_1-x_2≤10, x_1 AND x_2≥0 ส่วนการหาคำตอบจะเริ่มจากขั้นตอนแรกคือการหาขอบเขตของคำตอบที่เป็นไปได้ทั้งหมด (feasible region) จากสมการเงื่อนไขต่าง ๆ โดยหาจุดและขอบของขอบเขตที่เป็นไปได้ ดังตัวอย่างสมการเงื่อนไข จะหาจุดและขอบของคำตอบที่เป็นไปได้ ด้วยสมการ x_1+2x_2=8, 2x_1+x_2=14, 2x_1-x_2=10, x_1 AND x_2=0 จะได้ขอบเขตของคำตอบที่เป็นไปได้ดังรูปที่ 3

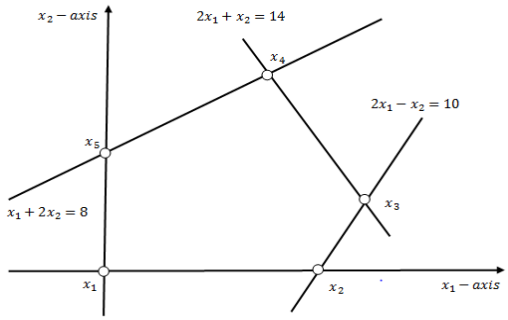


รูปที่ 3 ตัวอย่างพื้นที่ขอบเขตของคำตอบที่เป็นไปได้ (ที่มา: http://home.ubalt.edu/ntsbarsh /opre640a/nonlinear.htm)

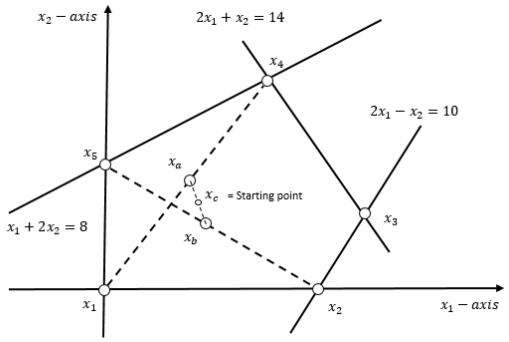
ต่อมาเป็นขั้นตอนการหาจุด critical point โดยถ้าเป็นปัญหาเชิงเส้น จุด critical point ก็คือจุดที่มุมทุกมุมของขอบเขตที่เป็นไปได้ ดังรูปที่ 4

จากนั้นเป็นขั้นตอนการหาจุดเริ่มต้นในการค้นหาคำตอบที่ดีที่สุด โดยการหาคำตอบจากจุดภายใน

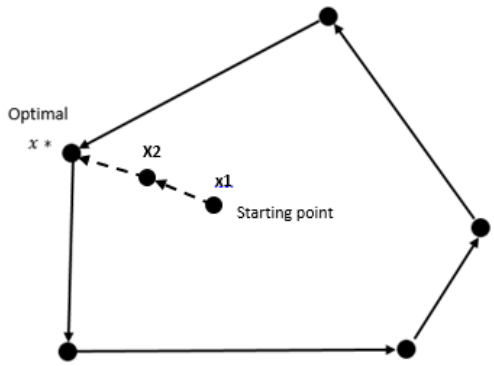
ขอบเขต (interior-point) และใช้จุดนั้นเป็นจุดเริ่มต้น จากตัวอย่างในรูปที่ 5 เป็นการสร้างความสัมพันธ์ (เส้นประ) ระหว่างคู่ critical point โดยตัวอย่างได้เลือกคู่จุด X_1,X_4 และ X_2,X_5 แล้วทำการสุ่มเลือกจุดบนเส้นความสัมพันธ์ทั้งสองเส้น ซึ่งก็คือจุด X_a และ X_b จากนั้นสร้างความสัมพันธ์ระหว่าง X_a,X_b และสุ่มเลือกจุดบนความสัมพันธ์ได้จุด X_c ซึ่งจะใช้เป็นจุดเริ่มต้นในการค้นหาคำตอบที่ดีที่สุด และสุดท้ายจะเป็นขั้นตอนการวนซ้ำคำนวณหาจุดที่ดีที่สุด โดยจะหยุดทำการวนซ้ำก็ต่อเมื่อเข้าสู่เงื่อนไขการสิ้นสุดการประมวลผล (tolerance) ซึ่งถือว่าเป็นจุด optimal ดังรูปที่ 6



รูปที่ 4 ตัวอย่างการแสดงจุด critical point (ที่มา: http://home.ubalt.edu/ntsbarsh/opre640a/nonlinear.htm)



รูปที่ 5 ตัวอย่างการแสดงจุดเริ่มต้นในการค้นหาคำตอบ (ที่มา: http://home.ubalt.edu/ntsbarsh/opre640a/nonlinear.htm)



รูปที่ 6 ลักษณะการลู่เข้าหาคำตอบที่ดีที่สุด (ที่มา: http://home.ubalt.edu/ntsbarsh/opre640a/nonlinear.htm)

2.4 วิธีซิมเพล็กซ์

วิธีซิมเพล็กซ์ [6] เป็นอัลกอรึทึมที่นำมาทดสองและเปรียบเทียบผลการทดลองกับวิธีจุดภายในโดยซิมเพล็กซ์เป็นวิธีที่นิยมวิธีหนึ่งของโปรแกรมเชิงเส้น ขั้นตอนวิธีซิมเพล็กซ์เป็นวิธีการที่มีประสิทธิภาพในทางปฏิบัติ วิธีนี้มีการพัฒนาจากวิธีทางพีชคณิตที่อาศัยทฤษฎีของแมทริกซ์เข้าร่วมจัดรูปแบบปัญหาให้มีระบบยิ่งขึ้น ช่วยให้ลังเกตความเปลี่ยนแปลงตัวแปรได้ง่ายและสามารถเข้าใจแนวทางที่ตัวแปรแต่ละตัวจะเปลี่ยนไปอย่างมีเหตุมีผล วิธีดังกล่าวจะเริ่มต้นการเปลี่ยนตัวแปรต่าง ๆ ให้มีผลต่อสมการกำหนดเป้าหมายโดยมีผลแนวโน้มสู้เป้าหมายในทางที่เร็วที่สุด การจัดรูปสมการเข้าเป็นตารางแล้วดำเนินการตามขั้นตอนที่ถูกต้องจะต้องทำให้ได้ผลลัพธ์ตามเป้าหมาย ผลลัพธ์ที่ดีที่สุด

การแก้ปัญหาโดยวิธีซิมเพล็กซ์จะต้องมีการกำหนดฟังค์ชันวัตถุประสงค์และฟังค์ชันเงื่อนไขในการหาค่าที่ดีที่สุดของฟังค์ชัน วิธีซิมเพล็กซ์เป็นการนำเมทริกซ์มาใช้ในการแก้ปัญหาโปรแกรมเชิงเส้นที่มีจำนวนตัวแปรตัดสินใจเป็นจำนวนมาก ซึ่งกระบวนการในการหาคำตอบด้วยวิธีนี้สามารถนำไปเขียน

โปรแกรมคอมพิวเตอร์ได้ โดยขั้นตอนของวิธีการซิมเพล็กซ์มีดังนี้

2.4.1 ขั้นตอนเริ่มต้น เริ่มจากการหาคำตอบเริ่มต้น ซึ่งคือคำตอบที่อยู่ในพื้นที่ที่เป็นคำตอบของสมการแสดงขอบข่ายคำนวณหาค่าของสมการกำหนดเป้าหมาย

2.4.2 ขั้นตอนทำซ้ำ เลื่อนไปสู่จุดยอดที่อยู่ติดกันบนพื้นที่ที่สอดคล้องกับข้อจำกัดทั้งหมด ถ้าค่าของสมการกำหนดเป้าหมายให้ผลลัพธ์ที่ดีกว่า (ทำซ้ำข้อ 2 จนกว่าเงื่อนไขในข้อ 3 จะเป็นจริง)

2.4.3 ขั้นตอนการตรวจสอบค่าที่ดีสุด จุดยอดที่ได้จะเป็นคำตอบถ้าไม่มีจุดยอดใด ๆ ที่ติดกันบนพื้นที่ที่สอดคล้องกับข้อจำกัดทั้งหมดที่ให้ผลลัพธ์ที่ดีกว่า

อย่างไรก็ตาม Klee และ Minty [7] ได้เสนอตัวอย่างที่ที่แสดงให้เห็นว่าในกรณีเลวร้ายที่สุดประสิทธิภาพการทำงานของขั้นตอนวิธีซิมเพล็กซ์จะใช้เวลาเป็นเอกซ์โพเนนเชียล ตั้งแต่นั้นมาการดำเนินการด้วยขั้นตอนวิธีนี้ก็ถูกมองว่าเป็นขั้นตอนตอนวิธีที่มีประสิทธิภาพเชิงเวลาไม่ดี ต่อมาเพื่อให้สามารถแก้ไขปัญหาได้เร็วขึ้นจึงมีการเสนอหลักการผ่อนปรนของเงื่อนไข (constraint) ขึ้นมาหลายรูปแบบ [8-10] ซึ่งต่างก็มีการกำหนดค่า penalty coefficient ซึ่งถ้ากำหนดน้อยไปเงื่อนไขจะไม่ค่อยถูกผ่อนปรน จะทำให้ได้คำตอบที่คุณภาพดีแต่ก็ยังใช้เวลานานมาก แต่หากกำหนดค่า coefficient นี้มากไปก็จะทำให้ใช้เวลาน้อยลง แต่จะได้คำตอบที่มีคุณภาพไม่ดี

2.5 การวิเคราะห์การไหลของข้อมูล

การวิเคราะห์การไหลของข้อมูลเป็นการนำความต้องการเชิงสมรรถนะของการประยุกต์ใช้เครือข่าย มาวิเคราะห์ร่วมกับข้อมูลตำแหน่งของอุปกรณ์ในเครือข่าย อุปกรณ์แม่ข่ายและสถานีปลายทาง การไหลของข้อมูลถือเป็นข้อมูลที่มีความสำคัญมากในการออกแบบระบบเครือข่าย เพราะการไหลของข้อมูลสามารถแสดงถึงภาระที่เครือข่ายจะต้องรองรับ การไหลของข้อมูลถือเป็นข้อมูลสำคัญที่ระบุทิศทางของข้อมูลพร้อมกับคุณสมบัติเชิงสมรรถนะของการไหล ในการวิเคราะห์ระบบเพื่อการออกแบบเครือข่ายจะใช้การไหลเป็นตัวแทนความต้องการในการใช้งานเครือข่ายของโปรแกรมหรือการส่งผ่านข้อมูลแต่ละชนิด ซึ่งการไหลของข้อมูลในโครงข่าย OSPF มีข้อจำกัดหลาย ๆ อย่าง ซึ่งการจะทำให้ได้คำตอบที่ดีที่สุดนั้นจะต้องให้ได้คำตอบที่สอดคล้องกับฟังก์ชันวัตถุประสงค์และสมการเงื่อนไขของทุก ๆ ข้อจำกัด

2.6 การวัดประสิทธิภาพภายในเครือข่าย

การวัดประสิทธิภาพภายในเครือข่ายสามารถวิเคราะห์ด้วยวิธีคิวอิ้ง [11] ดังสมการที่ 2

$$delay = Tp\frac{U}{1-U} \quad (2)$$

เมื่อ U คือ utilization โดย $U = \frac{l}{c}$;

$$T_p = \frac{Avg.Packet\ size}{Bandwidth}$$

l คือ load ของสายสัญญาณ

c คือ capacity หรือค่าปริมาณการไหลของข้อมูล

T_pคือ ค่าเวลาเฉลี่ยที่ใช้ในการส่งแพ็กเก็ต (transmission delay) (หมายเหตุ : ค่า T_p จะไม่นำมาคำนวณเนื่องจากหากเป็นเครือข่ายเดียวกันที่มี traffic requirement เหมือนกัน ก็จะมี T_p เท่ากันด้วย จึงไม่จำเป็นที่จะเอาค่า T_p เพราะเป็นการคำนวณค่า delay cost ที่ใช้สำหรับเปรียบเทียบ

ดังนั้นจะได้สมการใหม่ [11] เป็น

$$Delay\ Cost(\phi) = \frac{U}{1-U} \quad (3)$$

เมื่อแทนค่า U ในสมการที่ (3) จะได้เป็นสมการใหม่ที่ใช้ในการหาค่าค่า delay cost (ϕ) [11] ดังสมการ (4)

$$\phi_a(\text{delay cost}) = \frac{l_a}{C_a - l_a} \quad (4)$$

แต่สมการนี้ไม่สามารถนำมาใช้ได้เนื่องจากเมื่อค่า *l=c* หรือผลลัพธ์ค่า delay cost จะเป็นอนันต์

จึงต้องมีสมการใหม่มาแก้ไขปัญหานี้ โดยจะมีการแบ่งสมการออกเป็นส่วน ๆ ในลักษณะของ piece-wise linear แล้วนำค่า delay cost สูงสุดของแต่ละสายสัญญาณมาใช้คำนวณ และเมื่อนำผลรวมค่า delay cost สูงสุดของแต่ละสายสัญญาณมาคำนวณ ผลลัพธ์คือค่า delay cost [5] ของทั้งเครือข่าย (Φ) ดังแสดงในสมการที่ (5)

$$\Phi = \sum_{a\epsilon A}(\phi_a(l_a, C_a)) \quad (5)$$

โดย $\phi_a \geq l_a$, $\phi_a \geq 3l_a - \frac{2}{3}C_a$, $\phi_a \geq 10l_a - \frac{16}{3}C_a$, $\phi_a \geq 70l_a - \frac{178}{3}C_a$, $\phi_a \geq 500l_a - \frac{1468}{3}C_a$ และ $\phi_a \geq 5000l_a - \frac{19468}{3}C_a$; $l_a = \sum_{s,t\epsilon N} f_a^{s,t}$

เมื่อ Φ คือ ค่า delay cost ทั้งหมดในเครือข่าย

ϕ_a คือ ค่า delay cost ที่ใช้ในสายสัญญาณที่พิจารณา

l_a คือ ค่า load ของสายสัญญาณที่พิจารณา

C_a คือ ค่า bandwidth ของสายสัญญาณที่พิจารณา

a คือ สายสัญญาณ *(link)* ที่พิจารณา

f คือ ปริมาณข้อมูลที่ไหลจากต้นทางไปยังปลายทาง (flow) และผ่านสายสัญญาณที่กำลังพิจารณา

s คือ โหนดต้นทาง

t คือ โหนดปลายทาง

N คือ เซ็ตของโหนดทั้งหมด

2.7 optimum solution

ปัญหาการหาเส้นทางที่ดีที่สุด [12,13] ในระบบเครือข่าย มีเป้าหมายเพื่อหาค่าประสิทธิภาพในเชิงระยะเวลาหน่วงที่ดีที่สุด โดยรวมของทั้งเครือข่าย ซึ่งสามารถเขียนเป็นสมการได้ ดังนี้

<u>ฟังก์ชันวัตถุประสงค์</u>

$$\text{Min } \Phi(delay\ cost) = \sum_{a\epsilon A}(\phi_a(l_a, C_a)) \quad (6)$$

<u>เงื่อนไขที่ 1</u>

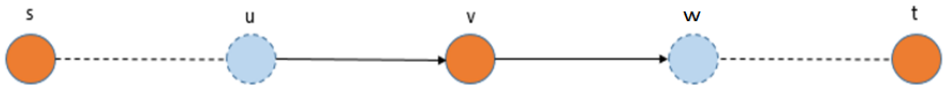
$$\sum_{u,v\in N} f_{u,v}^{s,t} - \sum_{u,v\in N} f_{v,w}^{s,t} = \begin{cases} d_{st} & if\ v = t \\ -d_{st} & if\ v = s \\ 0 & otherwise \end{cases} \quad (7)$$

$v, w, s, t \epsilon N,$

เมื่อ u คือ โหนดบนสายสัญญาณที่กำลังพิจารณา ด้านที่ใกล้โหนดต้นทาง

v คือ โหนดบนสายสัญญาณที่กำลังพิจารณา ที่อยู่ระหว่างโหนด u และ w

w คือ โหนดบนสายสัญญาณที่กำลังพิจารณา ด้านที่ใกล้โหนดปลายทาง



รูปที่ 7 สมการเงื่อนไข flow conservation

ทั้งนี้ สมการที่ (7) เป็นสมการ flow conservation [5] ซึ่งเป็นสมการเงื่อนไข (constraint) ที่บังคับ flow (f) ในแต่ละสายสัญญาณ โดยสามารถแบ่งออกเป็น 3 กรณี

2.7.1 จากรูปที่ 7 กรณีที่โหนด v เป็นโหนดปลายทาง จะมีข้อมูลไหลออกเท่ากับ 0 ดังนั้น ข้อมูลที่ไหลเข้า v ลบกับข้อมูลที่ไหลออก v จะมีค่าเท่ากับปริมาณข้อมูลที่ต้องการส่งจากต้นทาง s ไปยังปลายทาง t (d_{st})

2.7.2 จากรูปที่ 7 กรณีที่โหนด v เป็นโหนดต้นทาง จะมีข้อมูลไหลเข้าโหนดเท่ากับ 0 ดังนั้น ข้อมูลที่ไหลเข้า v ลบกับข้อมูลที่ไหลออก v จะมีค่าเท่ากับผลลบของปริมาณข้อมูลที่ต้องการส่งจากต้นทาง s ไปยังปลายทาง t (d_{st})

2.7.3 จากรูปที่ 7 กรณีที่โหนด v ไม่เป็นทั้งต้นปลายและปลายทาง ข้องมูลที่ไหลเข้า v จะเท่ากับข้อมูลที่ไหลออกจาก v ผลลบจึงมีค่าเท่ากับ 0

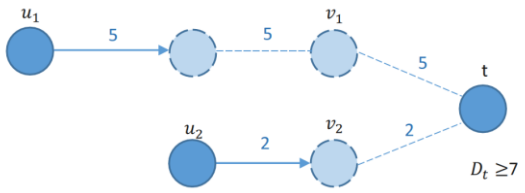
เงื่อนไขที่ 2

$$f_{u,v}^{s,t} \leq x_{u,v}^{s,t} * D_t \quad (8)$$

เมื่อ x คือ เส้นทางการไหลของข้อมูล ซึ่งมีค่าเป็นได้แค่ 0 หรือ 1 เท่านั้น

D_t คือ การไหลรวม ณ จุดปลายทาง

สมการที่ (8) เป็นสมการเงื่อนไขที่บังคับให้ข้อมูลที่ไหลจากโหนด u ใด ๆ ไปยังโหนด v ใด ๆ จะต้องมีค่าไม่เกินการไหลรวม ณ จุดปลายทาง t ดังแสดงในรูปที่ 8



รูปที่ 8 สมการเงื่อนไขของสมการที่ (8)

เงื่อนไขที่ 3

$$\sum_{u,v \in N} x_{u,v}^{s,t} \leq 1 \quad (9)$$

สมการที่ (9) เป็นสมการเงื่อนไขบังคับให้เส้นทางการไหลข้อมูลทุกแพ็กเก็ตจากโหนดใด ๆ ไปยังปลายทางมีได้เพียงเส้นทางเดียว

เงื่อนไขที่ 4

$$w_{uv} = w_{vu} \quad (10)$$

เมื่อ w คือ ค่า weight ของเส้นทาง

สมการที่ (10) เป็นสมการเงื่อนไขบังคับให้ค่า weight ของเส้นทางจาก u ไป v ต้องเท่ากับค่า weight ของเส้นทางจาก v ไป u

เงื่อนไขที่ 5, 6 และ 7

$$w_{uv} + P_{vt} - P_{ut} \geq 0 \quad (11)$$

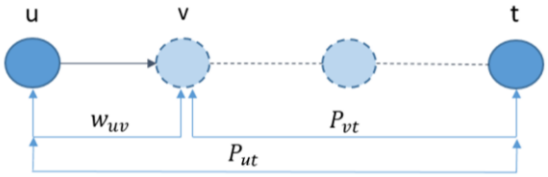
$$x_{u,v}^{s,t} + w_{uv} + P_{vt} - P_{ut} \geq 1 \quad (12)$$

$$x_{u,v}^{s,t} + (w_{uv} + P_{vt} - P_{ut})/M \leq 1 \quad (13)$$

เมื่อ P คือ ค่า weight รวมของเส้นทาง

M คือ ค่า Weight สูงสุดของเส้นทาง

จากรูปที่ 9 พบว่าจากอสมการที่ (11) ค่า weight เส้นทางจาก u ไป v บวกกับ ค่า weight รวมของเส้นทาง v ไป t ต้องไม่น้อยกว่าผลรวมค่า weight ของเส้นทางจาก u ไป t โดยค่า weight ของเส้นทางในแต่ละเส้นต้องคำนวณจากเส้นทางในการส่งข้อมูล และเพื่อไม่ให้เลือกเส้นทางที่ไม่มีค่า weight (ทางที่ไม่มีค่า weight คือ ทางที่ x มีค่าเป็น 0) ดังแสดงในสมการที่ (12) และ (13)



รูปที่ 9 เงื่อนไขที่ 5, 6 และ 7

2.8 เงื่อนไขการสิ้นสุดการประมวลผล (tolerance)

เป็นค่าที่กำหนดความแตกต่างที่น้อยที่สุดที่ยอมรับได้ของผลลัพธ์ที่ดีที่สุดในรอบปัจจุบัน (X_i) กับรอบคำนวณก่อนหน้า (X_{i-1}) จะสิ้นสุดเมื่อ $|(X_i - X_{i-1})| < Tolerance * (1 + |X_i|)$ ดังตัวอย่างในตาราง

รอบที่	X_i	$f(x)$	$\lvert X_i - X_{i-1} \rvert$	Tolerance*(1+$\lvert X_i \rvert$)
1	0.5	20	-	$1.5*10^{-7}$
2	0.4	18	0.1	$1.4*10^{-7}$
3	0.35	17	0.05	$1.35*10^{-7}$
4	0.351	16	0.001	$1.351*10^{-7}$
5	0.3509999	15.99	$1*10^{-7}$	$1.3509999*10^{-7}$

จากตัวอย่าง เมื่อค่า tolerance = 10^{-7} พบว่าในรอบที่ 5 $|(X_i - X_{i+1})|$ มีค่าน้อยกว่า $Tolerance * (1 + |X_i|)$ จึงสิ้นสุดการทำงานที่รอบที่ 5 และได้ค่าคำตอบ (X) เท่ากับ 0.3509999 ที่ให้ค่าสมการวัตถุประสงค์ $[f(x)]$ เท่ากับ 15.99

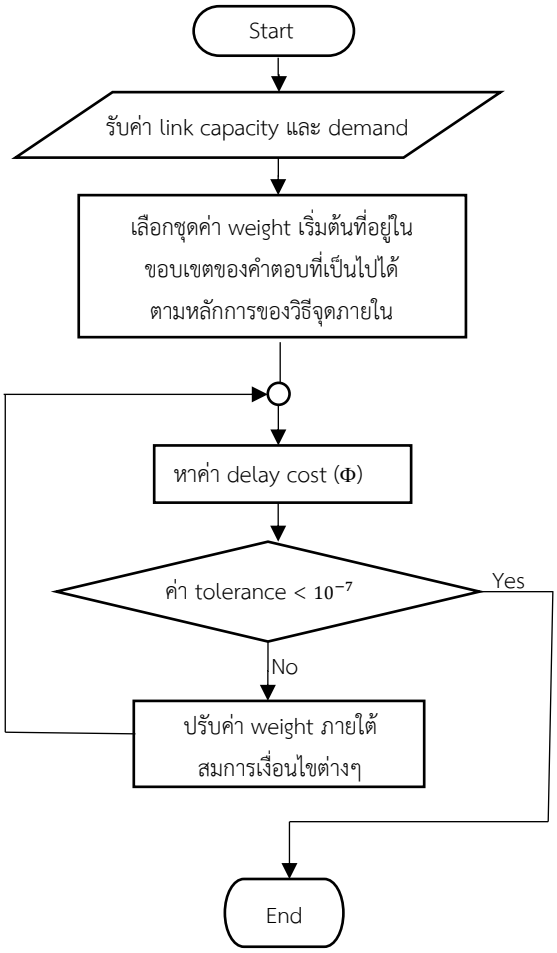
3. วิธีการดำเนินงานวิจัย

การทดลองจะทำการเปรียบเทียบระหว่างโครงข่ายที่กำหนดค่าน้ำหนักด้วยวิธีจุดภายใน กับวิธีการใช้ค่าน้ำหนักพื้นฐาน และวิธีซิมเพล็กซ์ โดยในการทดลองจะมีการกำหนดจำนวนโหนดและโหนดดีกรี ซึ่งโหนดดีกรีคือสัดส่วนจำนวนลิงค์ต่อจำนวนโหนด เพื่อใช้ในการสุ่มค่า bandwidth และค่า demand (ความต้องการไหลในโครงข่าย) โดยการเปรียบเทียบจะเปรียบเทียบค่า delay cost และประสิทธิภาพด้านเวลาในการคำนวณหาค่า weight

การเปรียบเทียบกับวิธีการใช้ค่าน้ำหนักพื้นฐานจะจำลองโครงข่ายขนาด 10, 14 โหนด โดยมีโหนดดีกรีเท่ากับ 1.3, 1.5, 1.7, 2.0 อย่างละ 4 โครงข่าย รวมทั้งหมด 24 โครงข่าย ส่วนการเปรียบเทียบกับวิธีซิมเพล็กซ์จะจำลองโครงข่ายขนาด 10 โหนด เท่านั้น เนื่องจากวิธีซิมเพล็กซ์ใช้เวลาประมวลผลนานมาก และหากเครือข่ายมีขนาดเกิน 10 โหนด จะไม่สามารถประมวลผลเสร็จได้ในเวลา 1 สัปดาห์ โดยเครือข่ายที่ใช้ในการทดลองมีโหนดดีกรีเท่ากับ 1.3, 1.5, 1.7, 2.0 อย่างละ 4 โครงข่าย รวมทั้งหมด 50 โครงข่าย

จากรูปที่ 10 แสดงกระบวนการหาค่าน้ำหนักของสายสัญญาณที่ดีที่สุดให้กับโครงข่าย โดยมีขั้นตอนดังนี้

3.1 รับค่า bandwidth คือ ความกว้างของช่องสัญญาณที่เชื่อมแต่ละคู่โหนดและรับ demand คือ ปริมาณความต้องการการไหลของข้อมูล โดยผู้วิจัยได้ใช้โปรแกรม Gen [3] ในการสร้างขึ้นมา



รูปที่ 10 กระบวนการหาค่า weight ที่เหมาะสมที่สุดด้วยวิธีจุดภายใน

3.2 เลือกชุดค่า weight เริ่มต้นที่อยู่ในขอบเขตของคำตอบที่เป็นไปได้ ตามหลักการของวิธีจุดภายใน

3.3 หาค่า delay cost รวมทั้งหมดในโครงข่ายโดยอ้างอิงจากสมการที่ (4)

3.4 (a) โปรแกรมจะหยุดการทำซ้ำก็ต่อเมื่อค่า tolerance ซึ่งเป็นค่าที่กำหนดค่าความแตกต่างที่น้อยที่สุดที่ยอมรับได้ของผลลัพธ์ที่ดีที่สุดในรอบการคำนวณปัจจุบัน (x_i) กับรอบการคำนวณถัดไป (x_{i+1}) ทั้งนี้การวนซ้ำจะสิ้นสุดเมื่อ $|x_i - x_{i+1}| < Tol * (1 + |x_i|)$

หรือ (b) โปรแกรมจะหยุดการทำซ้ำก็ต่อเมื่อค่า Tolerance ซึ่งเป็นค่าที่กำหนดค่าความแตกต่างที่น้อยที่สุดที่ยอมรับได้ของฟังก์ชันผลลัพธ์ดีที่สุดในรอบการคำนวณปัจจุบัน $f(x_i)$ กับรอบการคำนวณถัดไป $f(x_{i+1})$ ทั้งนี้การวนซ้ำจะสิ้นสุดเมื่อ $|f(x_i) - f(x_{i+1})| < Tol * (1 + |f(x_i)|)$

3.5 ชุดของค่า weight ที่เปลี่ยนแปลงในแต่ละรอบ จะมาจากการเคลื่อนที่จากจุดเริ่มต้น ภายในขอบเขตของคำตอบที่เป็นไปได้ โดยพิจารณาจากการปรับชุดค่า weight รูปแบบไหนที่ทำให้ค่า delay cost ภายในโครงข่ายดีขึ้นจะปรับเปลี่ยนค่า weight ในรูปแบบนั้นต่อไป จนกว่าจะปรับชุดค่า weight ที่ทำให้ค่า delay cost ไม่ดีขึ้นหรือแย่ลง จะต้องคำนวณหารูปแบบการปรับ weight ใหม่ ที่ทำให้ค่า delay cost ในโครงข่ายดีขึ้น

การทดสอบหาชุดค่า weight และค่า delay cost ที่เหมาะสม เพื่อให้มีความสามารถในการค้นหาคำตอบได้อย่างมีประสิทธิภาพ ทำได้โดยคำนวณประสิทธิภาพเชิงสมรรถนะจากสมการ (14)

$$\text{Cost efficient (\%)} = \frac{C*100}{DC} \quad (14)$$

เมื่อ DC คือ ค่า delay cost ในโครงข่ายที่คำนวณด้วยวิธีจุดภายใน

C คือ ค่า delay cost ในโครงข่ายที่คำนวณด้วยวิธี branch exchange วิธีซิมเพล็กซ์ และวิธีการใช้ค่าน้ำหนักพื้นฐาน

ส่วนการเปรียบเทียบประสิทธิภาพด้านเวลาในการคำนวณหาค่า weight สามารถคำนวณได้จากสมการ (15)

$$\text{Time efficient (\%)} = \frac{T*100}{TC} \quad (15)$$

เมื่อ TC คือ ค่าเวลาที่ใช้ในการคำนวณหาค่า weight ด้วยวิธีจุดภายใน

T คือค่าเวลาที่ใช้ในการคำนวณหาค่า weight ด้วย วิธี branch exchange และวิธีซิมเพล็กซ์

ในการทดลองปรับเปลี่ยนชุดค่า weight ที่ทำให้โครงข่ายมีประสิทธิภาพด้วยวิธีจุดภายในเริ่มต้นด้วยการเลือกชุดค่า weight เริ่มต้นที่อยู่ในขอบเขตของคำตอบที่เป็นไปได้ ดังเช่นตัวอย่างโครงข่ายขนาด 6 โหนด 8 ลิงค์ เริ่มต้นด้วยชุดค่า weight (3,345; 1,375; 2,934; 789; 2,931; 2,464; 4,144; 1,748) ค่า delay cost ในโครงข่ายเท่ากับ 8,128 ในรอบการทำงานที่ 1 หารูปแบบการปรับค่า weight ที่ทำให้ค่า delay cost ในโครงข่ายดีขึ้น จากการพิจารณาหารูปแบบการปรับค่า weight เราได้เลือกปรับลิงค์ที่มีค่า weight 1,375 เป็น 1,675 ทำให้ค่า delay cost ในโครงข่ายดีขึ้นจาก 8,128 เป็น 5,238 โดยมีชุดค่า weight (3,346; 1,675; 2,934; 789; 2,931; 2,464; 4,144; 1,748) จากนั้นในรอบการทำงานที่ 2 เมื่อปรับค่า weight รูปแบบเดิมคือปรับที่ลิงค์ที่มีค่า weight 1,675 จะทำให้ค่า delay cost ในโครงข่ายไม่ดีขึ้น จึงต้องหาการปรับ weight รูปแบบใหม่ โดยจากการพิจารณาหารูปแบบการปรับค่า weight เราเลือกปรับลิงค์ที่มีค่า weight 2,934 เป็น 3,355 ทำให้ค่า delay cost ในโครงข่ายดีขึ้นจาก 5,328 เป็น 3,788 ด้วยชุดค่า weight (3,345; 1,675; 3,355; 789; 2,931; 2,464; 4,144; 1,748) ส่วนในรอบการทำงานที่ 3 จากการพิจารณาหารูปแบบการปรับค่า weight เราเลือกปรับลิงค์ที่มีค่า weight 1,748 เป็น 2,248 แต่ค่า delay cost ในโครงข่ายเท่าเดิม และไม่ว่าจะปรับค่า weight ลิงค์ไหนอีกก็ไม่สามารถทำให้ค่า delay cost ในโครงข่ายดีขึ้น จึงสิ้นสุดการประมวลผลตามเงื่อนไข tolerance และสรุปได้ว่าชุดค่า weight (3,346; 1,675; 3,355; 789; 2,931; 2,464; 4, 144; 2,248) เป็นชุดค่า weight ที่ทำให้โครงข่ายมีประสิทธิภาพดีที่สุด

4. ผลการดำเนินงาน

จากการทดลองเพื่อเปรียบเทียบระหว่างโครงข่ายที่กำหนดค่าน้ำหนักด้วยวิธีจุดภายใน วิธีใช้ค่าน้ำหนักพื้นฐาน และวิธีซิมเพล็กซ์ได้ผลค่าประสิทธิภาพเชิงสมรรถนะและประสิทธิภาพด้านเวลา ดังแสดงในหัวข้อ 4.1

ตารางที่ 1 ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในเทียบกับวิธีการใช้ค่าน้ำหนักพื้นฐาน ของโครงข่ายขนาด 10 โหนด

Node degree	Cost efficient (%)
1.3	118.75
1.5	100.15
1.7	106.00
2.0	109.61

ตารางที่ 2 ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในเทียบกับวิธีการใช้ค่าน้ำหนักพื้นฐาน ของโครงข่ายขนาด 14 โหนด

Node degree	Cost efficient (%)
1.3	80.77
1.5	106.28
1.7	117.60
2.0	132.96

4.1 ผลการทดลอง

ตารางที่ 1 และ 2 ซึ่งเป็นผลเปรียบเทียบค่าประสิทธิภาพเชิงสมรรถนะของโครงข่ายที่กำหนดค่าน้ำหนักด้วยวิธีจุดภายในเทียบกับแบบวิธีการใช้ค่าน้ำหนักพื้นฐานของโครงข่ายขนาด 10 และ 14 โหนด ที่มีโหนดดีกรีเท่ากับ 1.3, 1.5, 1.7, 2.0 พบว่าที่โครงข่ายขนาดเล็กหรือโครงข่ายที่มีโหนดดีกรีต่ำ ซึ่งก็คือมีความซับซ้อนต่ำ วิธีกำหนดค่าน้ำหนักด้วยวิธีจุดภายในจะให้โครงข่ายที่มีสมรรถนะใกล้เคียงกับวิธีการใช้ค่าน้ำหนักพื้นฐาน แต่สำหรับโครงข่ายที่ขนาดใหญ่ขึ้นหรือซับซ้อนขึ้น วิธีกำหนดค่าน้ำหนักด้วยวิธีจุดภายในจะให้โครงข่ายที่มีประสิทธิภาพเชิงสมรรถนะที่ดีกว่าวิธีการใช้ค่าน้ำหนักพื้นฐาน

ตารางที่ 3 ค่าประสิทธิภาพเชิงสมรรถนะและเชิงเวลาของวิธีจุดภายในเทียบกับวิธีซิมเพล็กซ์ ของโครงข่ายขนาด 10 โหนด

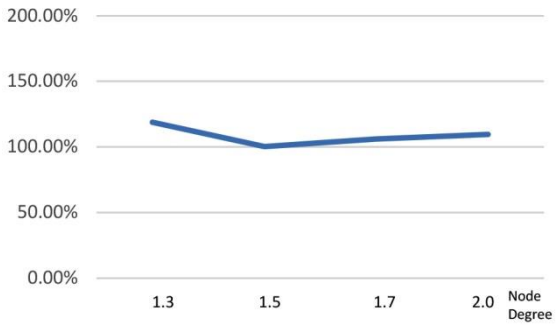
Node degree	Cost efficient (%)	Time efficient (%)
1.3	66.96	1,584.45
1.5	80.50	649.50
1.7	85.23	41,846.31
2.0	82.35	63,102.11

ส่วนตารางที่ 3 เป็นผลเปรียบเทียบค่าประสิทธิภาพเชิงสมรรถนะ และประสิทธิภาพเชิงเวลาในการคำนวณหาค่าน้ำหนัก ระหว่างวิธีจุดภายในเทียบกับวิธีซิมเพล็กซ์ ของโครงข่ายขนาด 10 โหนด ที่โหนดดีกรี 1.3, 1.5, 1.7, 2.0 ซึ่งพบว่าที่โครงข่ายที่มีความซับซ้อนต่ำ วิธีจุดภายในจะให้ประสิทธิภาพเชิงสมรรถนะต่ำกว่าวิธีซิมเพล็กซ์ แต่เมื่อโครงข่ายมีความซับซ้อนมากขึ้น วิธีจุดภายในจะให้ประสิทธิภาพเชิงสมรรถนะที่ดีใกล้เคียงกับวิธีซิมเพล็กซ์มากขึ้น ส่วนประสิทธิภาพเชิงเวลานั้นวิธีจุดภายในจะใช้เวลาในการประมวลผลน้อยกว่าวิธีซิมเพล็กซ์ หากยิ่งโครงข่ายที่มีความซับซ้อนมากขึ้น ประสิทธิภาพในเชิงเวลาของวิธีจุดภายในก็จะยิ่งดีกว่าวิธีซิมเพล็กซ์มากขึ้นมาก

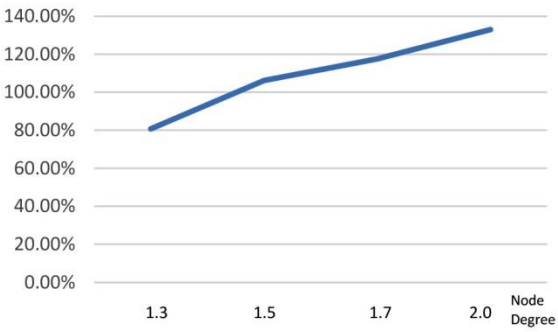
4.2 วิเคราะห์ผลการทดลอง

รูปที่ 11 และ 12 เป็นภาพแสดงผลค่าประสิทธิภาพเชิงสมรรถนะของวิธีโปรแกรมเชิงเส้น

แบบวิธีจุดภายในเทียบกับแบบวิธีการใช้ค่าน้ำหนักพื้นฐานของโครงข่ายขนาด 10 และ 14 โหนด พบว่าที่โครงข่ายขนาด 10 โหนด ประสิทธิภาพเชิงสมรรถนะของทั้งสองวิธีมีค่าใกล้เคียงกัน แต่เมื่อเพิ่มขนาดโครงข่ายเป็น 14 โหนด พบว่าเมื่อโครงข่ายซับซ้อนมากขึ้น ประสิทธิภาพเชิงสมรรถนะมีแนวโน้มดีกว่าอย่างเห็นได้ชัด ส่วนรูปที่ 13 และ 14 เมื่อเปรียบเทียบ

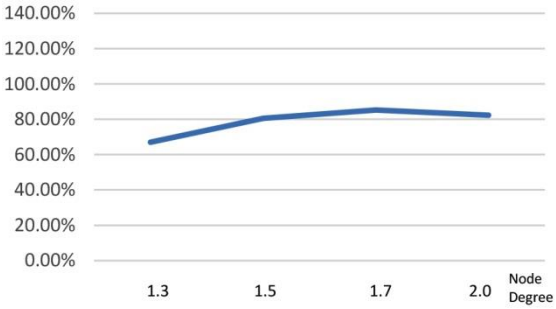


รูปที่ 11 ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในเทียบกับวิธีการใช้ค่าน้ำหนักพื้นฐานของโครงข่ายขนาด 10 โหนด

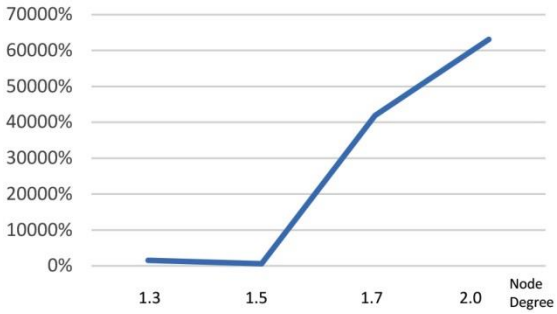


รูปที่ 12 ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในเทียบกับวิธีการใช้ค่าน้ำหนักพื้นฐานของโครงข่ายขนาด 14 โหนด

ประสิทธิภาพเชิงสมรรถนะระหว่างโครงข่ายขนาด 10 โหนด ที่กำหนดค่าน้ำหนักด้วยวิธีจุดภายในกับวิธีซิมเพล็กซ์ พบว่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในดีเกือบเท่ากับวิธีซิมเพล็กซ์ โดยเฉพาะเมื่อมีความซับซ้อนของโครงข่ายสูง แต่เมื่อเปรียบเทียบด้านเวลาที่ใช้ในการหาค่าน้ำหนัก พบว่าวิธีจุดภายในจะมีแนวโน้มที่ดีกว่ามากเมื่อมีความซับซ้อนในโครงข่ายเพิ่มขึ้น



รูปที่ 13 ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในเทียบกับวิธีซิมเพล็กของโครงข่ายขนาด 10 โหนด



รูปที่ 14 ค่าประสิทธิภาพเชิงเวลาของวิธีจุดภายในเทียบกับวิธีซิมเพล็กของโครงข่ายขนาด 10 โหนด

5. สรุป

งานวิจัยนี้เป็นการนำหลักการโปรแกรมเชิงเส้นในการตั้งค่าน้ำหนักสายสัญญาณให้แก่โครงข่ายโอเอสพีเอฟด้วยการใช้วิธีจุดภายใน แล้วนำมาวิเคราะห์ประสิทธิภาพเชิงสมรรถนะและประสิทธิภาพเชิงเวลาในการหาค่าน้ำหนัก ซึ่งผู้วิจัยได้พัฒนาระบบที่รับค่าความกว้างของช่องสัญญาณและความต้องการในการ

ไหลของโครงข่ายนั้นแล้วส่งไปคำนวณด้วยโปรแกรม GLPK [14] เพื่อหาค่าน้ำหนักสายสัญญาณที่เหมาะสมที่สุดในโครงข่าย จากนั้นนำข้อมูลของโครงข่ายที่ได้มาคำนวณค่าประสิทธิภาพ โดยดูจากภาระของแต่ละลิงค์ต่อความกว้างของช่องสัญญาณในลิงค์นั้น ๆ เพื่อนำไปเปรียบเทียบกับวิธีการตั้งค่าน้ำหนักสายสัญญาณของโครงข่ายที่คำนวณโดยวิธีการใช้ค่าน้ำหนักพื้นฐานและวิธีโปรแกรมเชิงเส้นแบบซิมเพล็กซ์

จากผลการทดสอบเมื่อเปรียบเทียบผลของวิธีจุดภายในกับวิธีการใช้ค่าน้ำหนักพื้นฐานพบว่าที่โครงข่ายขนาด 10 โหนด ค่าประสิทธิภาพเชิงสมรรถนะ (delay cost) ของทั้งสองวิธีมีค่าใกล้เคียงกัน แต่มีค่าที่ดีกว่าเมื่อเพิ่มขนาดโครงข่ายเป็น 14 โหนด แสดงให้เห็นว่าเมื่อโครงข่ายซับซ้อนมากขึ้น ค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในมีแนวโน้มที่ดีกว่าอย่างเห็นได้ชัด และเมื่อเปรียบเทียบระหว่างวิธีจุดภายในกับวิธีซิมเพล็กซ์ ผู้วิจัยพบว่าค่าประสิทธิภาพเชิงสมรรถนะของวิธีจุดภายในดีเกือบเท่ากับวิธีซิมเพล็กซ์ แต่ใช้เวลาที่ใช้ในการคำนวณที่เร็วกว่ามาก

ดังนั้นจึงสรุปได้ว่าการหาชุดค่าน้ำหนักของสายสัญญาณที่ดีเพื่อใช้ในโครงข่ายโอเอสพีเอฟ วิธีการใช้ค่าน้ำหนักพื้นฐานเป็นวิธีที่คำนวณด้วยวิธีง่าย ๆ เหมาะกับโครงข่ายที่ผู้ดูแลไม่มีเวลารอในการประมวลผลหาค่าน้ำหนักที่เหมาะสมสำหรับโครงข่ายนั้น ๆ ซึ่งค่าน้ำหนักนี้ก็สามารถใช้ในโครงข่ายได้ดีในระดับหนึ่ง แต่อาจไม่ใช่ค่าที่มีประสิทธิภาพมากนัก ส่วนการกำหนดค่าน้ำหนักด้วยวิธีซิมเพล็กซ์เป็นวิธีที่มีประสิทธิภาพสูงที่เหมาะกับโครงข่ายขนาดเล็ก เพราะถ้าโครงข่ายมีขนาดใหญ่จะใช้เวลานานมากในการคำนวณ และส่วนวิธีจุดภายในเป็นวิธีที่เหมาะกับโครงข่ายที่ขนาดใหญ่ขึ้น เพราะใช้เวลาในการประมาณผลที่ค่อนข้างเร็วกว่าวิธีซิมเพล็กซ์ และได้ชุดค่าน้ำหนักที่มีประสิทธิภาพ แต่อย่างไรก็ตาม ถ้าโครงข่ายมีขนาดเล็ก วิธีจุดภายในจะใช้ระยะเวลาในการประมวลผลหาค่าน้ำหนักที่มีประสิทธิภาพนานกว่าวิธีซิมเพล็กซ์

ทั้งนี้ข้อจำกัดของงานวิจัยคือการกำหนดค่า demand แบบตายตัว ทำให้ค่า weight ที่คำนวณออกมาได้เหมาะสมกับเฉพาะเมื่อมี demand คงที่เท่านั้น ซึ่งในความเป็นจริงค่า demand ในโครงข่ายมักไม่คงที่ มีการเปลี่ยนแปลงอยู่ตลอดเวลา แม้กระนั้นผู้วิจัยได้ศึกษางานของ Fortz และ Thorup [15] ซึ่งแสดงให้ทราบว่าค่า weight ที่คำนวณได้นี้ก็ยังสามารถใช้งานได้ดีพอสมควรในช่วงการเปลี่ยนแปลงของค่า demand ที่ไม่เกิน 50-110 %

6. กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนอุดหนุนจาก คณะวิศวกรรมศาสตร์ กำแพงแสน มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน

7. รายการอ้างอิง

[1] Cahn, R., 1998, Wide Area Network Design, Morgan Kaufmann Publisher, San Francisco.

[2] Karmarkar, N.K., 1991, Interior point methods in optimization, Proc. Int. Confer. Ind. Appl. Math. (SIAM) 2: 160-181

[3] กฤษณะ ลานทอง และกายรัฐ เจริญราษฎร์, 2555, การหาค่าน้ำหนักของเส้นทางที่คุ้มค่าที่สุดในโครงข่ายโอเอสพีเอฟด้วยวิธีโปรแกรมเชิงเส้น, การประชุมวิชาการแห่งชาติ ครั้งที่ 9 มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน, นครปฐม.

[4] ทิชากร สำรองทรัพย์ และกายรัฐ เจริญราษฎร์, 2557, การจัดการจราจรข้อมูลบนเครือข่ายโอ

เอสพีเอฟด้วยวิธี Branch Exchange, ว.วิชาการพระจอมเกล้าพระนครเหนือ 24: 503-511.

[5] Fortz, B. and Thorup, M., 2000, Internet traffic engineering by optimizing OSPF weights, Proc. IEEE INFOCOM 2: 519-528.

[6] Nelder, J.A. and Mead, R., 1965, A simplex method for function minimization, Comp. J. 7: 308-313.

[7] Klee, V. and Minty, G.V., 1972, How Good is the Simplex Algorithm, pp. 159-17, In Shisha, O. (Ed.), Inequalities-III, Academic Press, New York.

[8] Michalewicz, Z. and Attia, N., 1994, Evolutionary optimization of constrained problems, Proc. Annu. Conf. Evol. Program. 3: 98-108.

[9] Joines, J. and Houck, C., 1994, On the use of nonstationary penalty functions to solve nonlinear constrained optimization problems with GAs, Proc. IEEE Conf. Evol. Program. 1: 579-584.

[10] Kazarlis, S. and Petridis, V., 1998, Varying fitness functions in genetic algorithms: Studying the rate of increase of the dynamic penalty terms, Proc. Parallel Probl. Solving Nat. (PPSN V) 5: 211-220.

[11] พิสิษฐ์ ชาญเกียรติก้อง, 2550 การออกแบบโครงข่ายคอมพิวเตอร์, สำนักพิมพ์มหาวิทยาลัยรังสิต, ปทุมธานี, 438 น.

[12] Onwubiko, C., 1999, Introduction to Engineering Design Optimization, Prentice Hall, New York, 314 p.

[13] Fourer, R., 2005, Optimization Method, Department of Industrial Engineering and Management Sciences, Northwestern University, Illinois, 180 p.

[14] Makhorin, A.. 2000, GLPK (GNU Linear Programming Kit), แหล่งที่มา : http://www.gnu.org/software/glpk.

[15] Fortz, B. and Thorup, M., 2003, Robust optimization of OSPF/IS-IS weights, Proc. Int. Network Optimiz. Confer. 1: 225-230.