ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์

วิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์)

ปริญญา

วิศวกรรมคอมพิวเตอร์	วิศวกรรมคอมพิวเตอร์
สาขา	**ภาควิชา**

เรื่อง อัลกอริทึมสำหรับหาการจับคู่ที่ดีที่สุด

Algorithm for Finding Optimal Semi-matching

นามผู้วิจัย นายบัณฑิต เลขานุกิจ

ได้พิจารณาเห็นชอบโดย

ประธานกรรมการ .....

(อาจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D.)

กรรมการ .....

(อาจารย์ชัยพร ใจแก้ว, Ph.D.)

กรรมการ .....

(ผู้ช่วยศาสตราจารย์พันธุ์ปิติ เปี่ยมสง่า, D.Sc.)

หัวหน้าภาควิชา .....

(อาจารย์พีรวัฒน์ วัฒนพงศ์, Ph.D.)

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์รับรองแล้ว

.....

(รองศาสตราจารย์วินัย อาจคงหาญ, M.A.)

คณบดีบัณฑิตวิทยาลัย

วันที่ 5 **เดือน** เมษายน **พ.ศ.** 2549

วิทยานิพนธ์

เรื่อง

อัลกอริทึมสำหรับหาการจับกึ่งคู่ที่ดีที่สุด

Algorithm for Finding Optimal Semi-matching

โดย

นายบัณฑิต เลขานุกิจ

เสนอ

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์
เพื่อความสมบูรณ์แห่งปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์)
พ.ศ. 2549

ISBN 974-16-1486-1

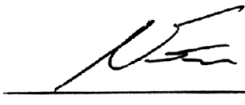
บัณฑิต เลขานุกิจ 2549: อัลกอริทึมสำหรับหาการจับกึ่งคู่ที่ดีที่สุด ปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์) สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ ประธานกรรมการที่ปรึกษา: อาจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D. 59 หน้า
ISBN 974-16-1486-1

การจับกึ่งคู่บนกราฟสองส่วน $G=(U \cup V,E)$ คือเซตของเส้นเชื่อม $M \subseteq E$ ที่แต่ละจุดยอดใน U ประชิดกับเส้นเชื่อมหนึ่งเส้นใน M. Harvey, Ladner, Lovász, and Tamir (2003) ได้พิจารณาการจับกึ่งคู่ในรูปของการมอบหมายงานใน U ให้กับเครื่องใน V โดยให้นิยามค่าใช้จ่ายของการจับกึ่งคู่ M เป็นเวลารอรวมของแต่ละงาน และแสดงว่าการการจับกึ่งคู่ที่มีเวลารอรวมต่ำที่สุดจะให้การกระจายภาระงานที่ดีที่สุดในการวัดผลหลายแบบซึ่งรวมทั้งเวลารอรวมสูงสุดที่เครื่องจักรและค่าความแปรปรวนของภาระงาน พร้อมทั้งเสนออัลกอริทึมที่ทำงานในเวลา $O(|U||E|)$ ซึ่งพัฒนามาจากวิธีการแบบฮังกาเรียนที่ใช้ในการแก้ปัญหาการจับคู่สองส่วน ในงานวิจัยชิ้นนี้ เราได้เสนออัลกอริทึมแบบแบ่งแยกแล้วจัดการที่ทำงานในเวลา $O(|E||U|^{1/2}\log|U|)$ อัลกอริทึมดังกล่าวสามารถทำงานได้ในกรณีทั่วไปเมื่อฟังก์ชันค่าใช้จ่ายในแต่ละเครื่องจักรสามารถแตกต่างกันโดยใช้เวลา $O(|E||U|^{1/2}\log|E|)$ นอกจากนี้ยังได้พัฒนาไปสู่ในกรณีที่กราฟมีน้ำหนักและนำเสนออัลกอริทึมที่ทำงานในเวลา $O(|U||E|\ \log|U|)$ เมื่อเปรียบเทียบกับอัลกอริทึมที่ดีที่สุดที่รู้จักซึ่งเป็นของ Edmonds and Karp (1970) และTomizawa (1971) จะมีเพียงตัวประกอบแบบลอการิทึมเพิ่มเข้ามาเท่านั้น

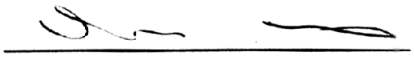
ลายมือชื่อนิสิต ลายมือชื่อประธานกรรมการ 29 / มี.ค. / 49

Bundit Laekhanukit 2006: Algorithm for Finding Optimal Semi-matching.
Master of Engineering (Computer Engineering), Major Field: Computer Engineering,
Department of Computer Engineering. Thesis Advisor: Mr. Jittat Fakcheroenphol, Ph.D.
59 pages.
ISBN 974-16-1486-1

A semi-matching on a bipartite graph $G=(U \cup V,E)$ is a set of edges $M \subseteq E$ such that each vertex in U is incident to exactly one edge in M. Harvey, Ladner, Lovász, and Tamir (2003) consider the matching as an assignment for tasks in U to machines in V. This motivates the definition of the cost of a semi-matching M to be the sum of delay for each task. They show that optimizing this sum of delay is equivalent to optimizing the load in various metrics, including the makespan, the flow time, and the variance of the loads. They give an $O(|U||E|)$ algorithm based on the Hungarian algorithm for bipartite matching. In this research, we give a divide-and-conquer algorithm which runs in time $O(|E||U|^{1/2}\log|U|)$. Our algorithm also works in a more general settings where the cost functions for each machine can be different in time $O(|E||U|^{1/2}\log|E|)$. Furthermore, for the weighted case where edge weights are allowed, we give an algorithm that runs in time $O(|U||E|\log|U|)$, comparable to the best known time bound as one of Edmonds-Karp (1970) and Tomizawa (1971)'s algorithm for the bipartite matching problem with additional logarithmic factor.



Student's signature



Thesis Advisor's signature

29 / MAR / 06

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลงได้ด้วยความช่วยเหลือจากบุคคลหลายท่าน ข้าพเจ้าขอขอบพระคุณ อาจารย์จิตร์ทัศน์ ฝักเจริญผล ประธานกรรมการที่ปรึกษา ผู้ให้แนวทางการวิจัยและข้อเสนอแนะที่เป็นประโยชน์ อาจารย์ชัยพร ใจแก้ว กรรมการที่ปรึกษาวิชาเอก ผู้ช่วยศาสตราจารย์พันธุ์ปิติ เปี่ยมสง่า กรรมการที่ปรึกษาวิชารอง และผู้ช่วยศาสตราจารย์อุศนา ตัณฑุลเวศม์ อาจารย์ผู้แทนบัณฑิตวิทยาลัย ที่กรุณาให้คำปรึกษา และข้อเสนอแนะที่มีคุณค่าเพื่อให้วิทยานิพนธ์นี้สมบูรณ์ยิ่งขึ้น

ข้าพเจ้าขอขอบคุณนายดนุพล ณ หนองคาย ผู้ร่วมทำงานวิจัย เพื่อนๆ นิสิตปริญญาโท และสมาชิกกลุ่มวิจัยเชิงทฤษฎีทุกท่านที่ให้คำปรึกษาเกี่ยวกับการทำงานวิจัย และเจ้าหน้าที่ธุรการภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ทุกท่านสำหรับความช่วยเหลือด้านการประสานงาน และดำเนินการเอกสารต่างๆ

บัณฑิต เลขานุกิจ

มีนาคม 2549

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญภาพ

อัลกอริทึมสำหรับหาการจับกึ่งคู่ที่ดีที่สุด

Algorithm for Finding Optimal Semi-matching

คำนำ

ปัญหาการจับคู่สูงสุดบนกราฟสองส่วน (maximum bipartite matching problems) เป็นปัญหาที่สำคัญปัญหาหนึ่ง ในการศึกษาด้านการหาค่าดีที่สุดเชิงการจัด (combinatorial optimization) ให้กราฟสองส่วน $G=(U\cup V,E)$ *การจับคู่* $M\subseteq E$ บน G คือ เซตของเส้นเชื่อม ที่ไม่มีเส้นเชื่อมคู่ใดใน M ใช้จุดปลายร่วมกัน เรียกการจับคู่ M ที่ทุกจุดยอด $u\in U$ มีเส้นเชื่อมที่ติดกับ u ใน M ว่าเป็น *การจับคู่แบบสมบูรณ์* ถ้าเรามีฟังก์ชันค่าใช้จ่าย c บนเส้นเชื่อมใน E ค่าใช้จ่ายของการจับคู่ M คือ $\sum_{e\in M}c(e)$ *ปัญหาการจับคู่ที่มีน้ำหนักต่ำที่สุด* (minimum-weight matching) คือ การหา*การจับคู่สูงสุด* (maximum matching) ที่มีค่าใช้จ่ายน้อยที่สุด เนื่องจากปัญหาอยู่บนกราฟสองส่วนจะเรียกว่า*ปัญหาการจับคู่สองส่วน* (bipartite matching problems) สำหรับปัญหา*การจับคู่สองส่วนแบบมีน้ำหนัก* (weighted bipartite matching problems) จะเป็นปัญหาเดียวกับ *ปัญหาการมอบหมายงาน* (assignment problems)

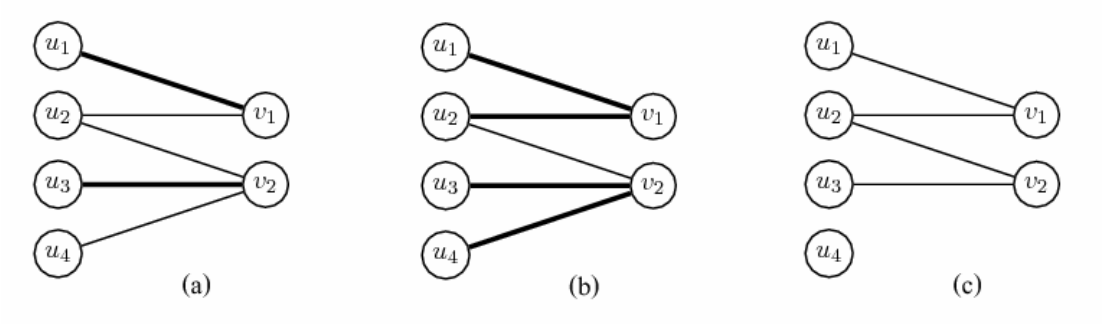
งานวิจัยชิ้นนี้พิจารณา *ปัญหาการจับกึ่งคู่ที่ดีที่สุด* (optimum semi-matching) ซึ่งนิยามโดย Harvey et al. (2003) หรือเรียกสั้น ๆ ว่า *ปัญหาการจับกึ่งคู่* ปัญหาดังกล่าวเป็นปัญหาที่ได้จากการปรับเปลี่ยนเงื่อนไขของการจับคู่ในกราฟสองส่วนมาเป็นการจับกึ่งคู่ (semi-matching) ซึ่งนิยาม *การจับกึ่งคู่* คือ เซตของเส้นเชื่อม $M\subseteq E$ ที่ทุกจุดยอด $u\in U$ ติดกับเส้นเชื่อมใน M อย่างน้อยหนึ่งเส้น จากนิยามดังกล่าวกราฟจะไม่มีการจับกึ่งคู่ถ้ามีจุดยอดใน U เป็น *จุดยอดเอกเทศ* (isolated point) คือ จุดยอดที่ไม่ประชิดกับเส้นเชื่อมใดเลย ตัวอย่างของการจับคู่และการจับกึ่งคู่ในกราฟสองส่วนแสดงในภาพที่ 1 (a) และ (b) ตามลำดับ ส่วนในภาพ (c) แสดงตัวอย่างของกราฟที่ไม่มีการจับกึ่งคู่ เพราะมีจุดยอด $u_4\in U$ เป็นจุดยอดเอกเทศ สำหรับจุดยอด $v\in V$ ให้ $deg_M(v)$ แทนจำนวนเส้นเชื่อมใน M ที่ติดกับ v นิยามค่าใช้จ่ายของ M ที่จุดยอด $v\in V$ เป็น

$$cost_M(v)=\sum_{i=1}^{deg_M(v)} i=(deg_M(v)+1)(deg_M(v))/2$$

ค่าใช้จ่ายของการจับกึ่งคู่ M คือ

$$cost(M)=\sum_{v\in V} cost_M(v)$$

ปัญหาการจับกึ่งคู่ที่ดีที่สุด คือ การหาการจับกึ่งคู่ที่มีค่าใช้จ่ายต่ำที่สุด



ภาพที่ 1 ตัวอย่างการจับกึ่งคู่ ภาพ (a) แสดงตัวอย่างของการจับคู่บนกราฟสองส่วน แสดงการจับคู่ด้วยเส้นทึบ, ภาพ (b) แสดงตัวอย่างของการจับกึ่งคู่บนกราฟสองส่วน แสดงการจับกึ่งคู่ด้วยเส้นทึบ และ (c) แสดงตัวอย่างของกราฟที่ไม่มีการจับกึ่งคู่

ปัญหาการจับกึ่งคู่ เป็นรูปแบบหนึ่งของการปรับเงื่อนไขของปัญหาการจับคู่สองส่วนเพื่อใช้ในงานเฉพาะเจาะจง รูปแบบของค่าใช้จ่ายที่ Harvey et al. (2003) นิยามขึ้นมานั้นมีที่มาจากการประยุกต์ใช้ด้านการกระจายภาระงาน (load-balancing) พิจารณาเซต U ว่าเป็นเซตของ “งาน” ในขณะที่ V เป็นเซตของเครื่องจักร จากนั้นให้พิจารณาการจับกึ่งคู่เป็นการมอบหมายงานกับเครื่องจักร โดยเครื่องจักรหนึ่งเครื่องอาจได้รับงานมากกว่าหนึ่งชิ้นก็ได้ ค่าใช้จ่ายของการจับกึ่งคู่จะเป็นผลรวมของเวลารอจนกว่าทุกงานทำเสร็จ สมมติว่าเครื่องจักรสามารถทำงานได้หนึ่งชิ้นต่อเวลาหนึ่งหน่วย ดังนั้น งานชิ้นที่ i จะต้องใช้เวลารอก่อนจะเข้าประมวลผล i-1 และจะประมวลผลเสร็จโดยใช้เวลาทั้งหมดi หน่วย ดังนั้นเวลารอของงานทั้งหมดที่เครื่องจักร v คือ

$$cost_M(v)=\sum_{i=1}^{deg_M(v)} i=(deg_M(v)+1)(deg_M(v))/2$$

ตามนิยามข้างต้น ปัญหานี้สามารถพิจารณาได้ว่าเป็นรูปแบบหนึ่งของ*ปัญหาการมอบหมายงาน* (assignment problems) เมื่อมีความไม่สมมาตรเกิดขึ้นตัวอย่างเช่น ปัญหาการมอบหมายงาน 10 งานให้กับเครื่องจักร 5 เครื่องและงานทุกชิ้นต้องได้รับการทำงาน คือต้องมอบหมายงานทุกชิ้นให้กับเครื่องใดเครื่องหนึ่งโดยที่เครื่องจักรแต่ละเครื่องอาจจะรับงานมากกว่าหนึ่งงานได้ งานวิจัยชิ้นนี้มีจุดมุ่งหมายเพื่อพัฒนาอัลกอริทึมที่เร็วขึ้นสำหรับแก้ปัญหาการจับกึ่งคู่ที่ดีที่สุดในกรณีที่กราฟไม่มีน้ำหนัก (unweighted graph) และพัฒนาอัลกอริทึมสำหรับกรณีที่กราฟมีน้ำหนัก (weighted graph)

นิยามของปัญหาอย่างเป็นรูปนัย

ปัญหาการจับกึ่งคู่ที่ดีที่สุดแบบไม่มีน้ำหนัก

ให้กราฟสองส่วน $G=(U\cup V,E)$ ที่ไม่มีจุดยอดเอกเทศ และ ฟังก์ชันน้ำหนัก $w:E\rightarrow R$ เขียน $deg_M(v)$ แทนจำนวนของเส้นเชื่อมใน M ที่ประชิดกับจุดยอด v นิยามฟังก์ชันค่าใช้จ่ายสำหรับแต่ละจุดยอด $v\in V$ ที่สัมพันธ์กับค่าการจับกึ่งคู่ M ดังนี้

$$cost_M(v)=\sum_{i=1}^{deg_M(v)} i=(deg_M(v)+1)(deg_M(v))/2$$

และจะได้ค่าใช้ของการจับกึ่งคู่เป็น

$$cost(M)=\sum_{v\in V} cost_M(v)$$

ปัญหาการจับคู่ที่ดีที่สุดแบบไม่มีน้ำหนัก ต้องการหาการจับกึ่งคู่ $M\subseteq E$ ที่มีค่าใช้จ่ายต่ำที่สุด

ปัญหาการจับกึ่งคู่ที่ดีที่สุดแบบมีน้ำหนัก

ในกรณีที่เส้นเชื่อมมีน้ำหนักเราจะนิยามค่าใช้จ่ายของการจับกึ่งคู่โดยอ้างอิงกับปัญหาการกระจายภาระงานในกรณีนี้เวลาที่งานแต่ละงานต้องรอในการกำหนดงานให้กับเครื่องจักรนั้นนอกจากจะขึ้นกับจำนวนงานที่ถูกกำหนดให้เครื่องจักรแล้วยังขึ้นกับลำดับการประมวลผลงานเหล่านั้นด้วย อย่างไรก็ตามสำหรับเซตของเวลาที่ใช้ในการประมวลผลงานใด ๆลำดับการประมวลผลที่ให้เวลารวมน้อยที่สุดคือลำดับที่ประมวลผลงานที่เสร็จเร็วสุดก่อน(พิสูจน์ในทฤษฎีบทย่อยที่ 8) ดังนั้นเราสามารถนิยามปัญหาการจับคู่ที่ดีที่สุดแบบมีน้ำหนักได้ดังนี้ ให้กราฟสองส่วน $G=(U\cup V,E)$ ที่ไม่มีจุดยอดเอกเทศ และ ฟังก์ชันน้ำหนัก $w:E\rightarrow R$ เขียน $deg_M(v)$ แทนจำนวนของเส้นเชื่อมใน M ที่ประชิดกับจุดยอด $v\in V$ นิยามฟังก์ชันค่าใช้จ่ายสำหรับแต่ละจุดยอด $v\in V$ ที่สัมพันธ์กับการจับกึ่งคู่M โดยพิจารณาจากจุดยอด $u_1,u_2,...,u_{degM(v)}$ ที่ประชิดกับ v เรียงตามลำดับของน้ำหนักของ $w(u_i,v)$ จากมากไปน้อย จะได้ฟังก์ชันค่าใช้จ่ายที่จุดยอด v ดังนี้

$$cost_M(v)=\sum_{i=1}^{deg_M(v)} i\cdot w(u_i,v)$$

และจะได้ค่าใช้จ่ายของการจับกึ่งคู่เป็น

$$cost(M)=\sum_{v \in V} cost_M(v)$$

ปัญหาการจับคู่ที่ดีที่สุดแบบมีน้ำหนัก ต้องการหาการจับกึ่งคู่ $M \subseteq E$ ที่มีค่าใช้จ่ายต่ำที่สุด

ผลการวิจัย

การวิจัยมีผลลัพธ์ดังนี้

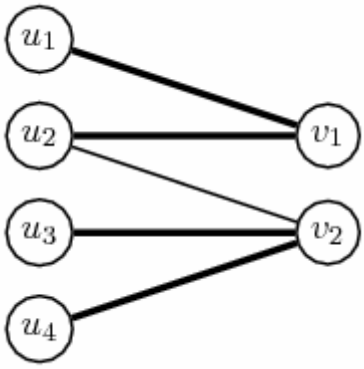
1 อัลกอริทึมที่ทำงานในเวลา $O(|E|\sqrt{(|U|)}\log|U|)$ สำหรับกรณีที่เส้นเชื่อมไม่มีนำหนัก ซึ่งปรับปรุงจากอัลกอริทึมของ Harvey et. al. ซึ่งทำงานในเวลา $O(|U||E|)$

2 นิยามของปัญหาและคุณสมบัติพื้นฐานของปัญหาการจับกึ่งคู่ในกรณีที่เส้นเชื่อมมีน้ำหนัก

3 อัลกอริทึมที่ทำงานในเวลา $O(|U|(|E|+|U|\log|U|))$ ในกรณีที่เส้นเชื่อมมีน้ำหนักซึ่งเป็นอัลกอริทึมแบบเวลาพหุนามอัลกอริทึมแรกของปัญหานี้

ตัวอย่าง

สำหรับตัวอย่างของปัญหาแสดงในภาพที่ 2 เป็นตัวอย่างของการมอบหมายงานที่จำนวนเครื่องจักรมากกว่าจำนวน ในภาพเส้นทึบแสดงการจับกึ่งคู่



ภาพที่ 2 ตัวอย่างปัญหาการจับกึ่งคู่ เส้นทึบแสดงการจับกึ่งคู่

การประยุกต์ใช้งานอื่น ๆ

ปัญหาการจับกึ่งคู่ที่ดีที่สุด ตามนิยามดังกล่าวมีจุดมุ่งหมายเพื่อกระจายภาระงาน ซึ่ง Harvey et. al. ได้แสดงว่าการจับกึ่งคู่ที่ดีที่สุดตามนิยามดังกล่าวจะเป็นการกระจายภาระงานที่ดีที่สุดในการวัดผลหลายรูปแบบ คือ ฟังก์ชันวัดผลที่ขึ้นกับดีกรีของจุดยอด และ การวัดผลโดยใช้ค่านอร์ม (norm) ทุกรูปแบบรวมทั้งค่าความแปรปรวน (variance) และ ค่าภาระงานสูงสุด (maximum load) นอกจากนี้อัลกอริทึมสำหรับแก้ปัญหายังสามารถนำไปใช้งานได้ดีในการแก้ปัญหาอื่น เช่น ในระบบไมโครซอฟต์แอคทิฟไดเรคทอรี่ (Microsoft Active Directory system) (อ่านรายละเอียดเพิ่มเติมในงานวิจัยของ Harvey et. al. (2003))

วัตถุประสงค์

1. เพื่อศึกษาปัญหาการจับกึ่งคู่และคุณสมบัติที่สำคัญ
2. เพื่อพัฒนาอัลกอริทึมในการแก้ปัญหาการจับกึ่งคู่

ขอบเขตของงานวิจัย

ศึกษาปัญหาการจับกึ่งคู่ที่ดีที่สุด ในกรณีที่กราฟมีน้ำหนักและกราฟไม่มีน้ำหนัก และหาอัลกอริทึมในการแก้ปัญหา โดยทำการวิจัยเฉพาะด้านทฤษฎี

การตรวจเอกสาร

ปัญหาการจับกึ่งคู่ถูกกล่าวถึงครั้งแรกในหนังสือของ Lawler (2001) ซึ่งกล่าวถึงปัญหาการจับกึ่งคู่แบบมีน้ำหนัก โดยมีจุดประสงค์เพื่อหาการจับกึ่งคู่ที่มีน้ำหนักรวมของเส้นเชื่อมมากที่สุด ปัญหาดังกล่าวสามารถแก้ได้โดยใช้อัลกอริทึมแบบละโมบ (greedy algorithm) ปัญหาการจับกึ่งคู่นี้ถูกนำมากล่าวถึงอีกครั้งโดย Harvey et al. (2003) ซึ่งมองปัญหานี้เป็นการกำหนดงานให้กับเครื่องจักร และสามารถกำหนดงานหลายงานให้กับเครื่องจักรเครื่องเดียวได้ โดยคิดค่าใช้จ่ายเป็นเวลาในการรอการทำงานของแต่ละงาน พร้อมทั้งเสนออัลกอริทึมสองวิธีสำหรับแก้ปัญหาในกราฟ $G=(U\cup V,E)$ โดยทำงานในเวลา $O(|U||E|)$ และ $O(|U|^{3/2}|E|)$ นอกจากนี้ Harvey et al. (2003) ยังได้แสดงวิธีการลดทอนปัญหา (problem reduction) เป็นปัญหาการไหลสูงสุดในเครือข่ายที่มีค่าใช้จ่ายต่ำที่สุด (minimum-cost maximum-flow problems) และปัญหาการจับคู่สองส่วนเชิงน้ำหนัก (weighted bipartite matching problems) (อยู่ในส่วนของการนำเสนอผลงาน แต่ไม่ปรากฏในวารสารวิชาการ)

นอกจากนี้จากการศึกษาทำให้ทราบว่า เทคนิคและวิธีการที่ใช้ในการแก้ปัญหาการไหลสูงสุดในเครือข่าย (maximum flow problem) และ ปัญหาการหาวิถีที่สั้นที่สุด (shortest problem) สามารถนำมาใช้ในการแก้ปัญหาการจับกึ่งคู่ได้เช่นกัน ในบทนี้จะอธิบายทฤษฎีที่เกี่ยวข้องกับการจับกึ่งคู่ กล่าวคือในส่วนแรกจะอธิบายทฤษฎีการจับคู่ (matching theory) ส่วนที่สองกล่าวถึงทฤษฎีการไหลในเครือข่าย (network flow theory) สำหรับเนื้อหาในส่วนที่ผ่านมาผู้อ่านที่สนใจสามารถค้นคว้าเพิ่มเติมได้จากหนังสือมาตรฐาน เช่น หนังสือ Data structures and network algorithms ของ Tarjan (1983), หนังสือ Combinatorial Optimization: Networks and Matriods (2001), หนังสือ Network flows: theory, algorithms, and applications ของ Ahuja, Magnanti and Orlin (1993), หนังสือ Combinatorial optimization : polyhedra and efficiency ของ Schrijver (2003) และ หนังสือ Matching Theory ของ Lovász and Plummer (1986) ในส่วนสุดท้ายของบทนี้เราจะกล่าวถึงงานวิจัยในอดีตของปัญหาการจับกึ่งคู่

ทฤษฎีการจับคู่

ทฤษฎีการจับคู่กล่าวถึงคุณสมบัติการกราฟที่มีการจับคู่สมบูรณ์ คุณสมบัติของการจับคู่สูงสุด ปัญหาในทฤษฎีการจับคู่แบ่งได้เป็นสองกลุ่ม คือ ปัญหาการจับคู่ในกราฟสองส่วน หรือเรียกอีกอย่างหนึ่งว่าการจับคู่สองส่วน (bipartite matching problems) และ ปัญหาการจับคู่บนกราฟที่ไม่ใช่กราฟสองส่วน หรือ เรียกอีกอย่างหนึ่งว่า (non-bipartite matching problems) ซึ่งในที่นี้เราจะสนใจเฉพาะปัญหาการจับคู่สองส่วนเท่านั้น ปัญหาการจับคู่สองส่วน (bipartite matching problems) เป็นปัญหาที่มีความสำคัญในด้านการหาค่าเหมาะสมที่สุด(optimization problems) และเป็นปัญหาที่มีผู้ศึกษากันมากที่สุดปัญหาหนึ่ง งานวิจัยในด้านนี้ประกอบด้วยรูปแบบของปัญหาและเทคนิคในการแก้ปัญหาที่แตกต่างกันเป็นจำนวนมาก และมีการนำไปประยุกต์ใช้ในรูปแบบของการมอบหมายงาน (assignment problems) และปัญหาการขนส่ง (transportation) ปัญหาการจับกึ่งคู่ (semi-matching problems) ก็เป็นอีกรูปแบบหนึ่งของปัญหาในกลุ่มนี้เช่นกัน ปัญหาการจับคู่สองส่วนสามารถแบ่งออกได้เป็นสองประเภท คือ ปัญหาการจับคู่สองส่วนเชิงขนาด (cardinality bipartite matching problem) และปัญหาการจับคู่สองส่วนเชิงน้ำหนัก (weighted bipartite matching problems) ในหัวข้อต่อไปเราจะกล่าวถึงทฤษฎีพื้นฐานของการจับคู่และนิยามของปัญหาการจับคู่

ทฤษฎีพื้นฐาน

ในกรณีที่เส้นเชื่อมไม่มีน้ำหนัก ทฤษฎีบทของ Hall ได้จำแนกกรณีที่กราฟมีการจับคู่แบบสมบูรณ์

ทฤษฎีบทที่ 1 (ทฤษฎีบทของ Hall (Lovász, 1986)) *ให้ $G=(U\cup V,E)$ เป็นกราฟสองส่วน และ ให้ $\Gamma(X)$ แทนทุกจุดยอดใน $U\cup V$ ที่ติดกับจุดยอดอย่างน้อยหนึ่งจุดใน X แล้ว G จะมีการจับคู่จาก U ไปยัง V ก็ต่อเมื่อ $|\Gamma(X)|\geq|X|$ สำหรับทุก $X\subseteq U$*

ทฤษฎีบทของ Egerváry แสดงเงื่อนไขของความดีที่สุดของการจับคู่ M ในกรณีที่กราฟมีน้ำหนัก ดังด้านล่าง

ทฤษฎีบทที่ 2 (ทฤษฎีบทของ Egerváry (Schrijver, 2003)) *ให้ $G=(U\cup V,E)$ เป็นกราฟสองส่วน และให้ $w:E\rightarrow R^+$ เป็นฟังก์ชันน้ำหนัก แล้วน้ำหนักสูงสุดของการจับคู่ใน G จะเท่ากับค่าต่ำสุดของ $\alpha(U\cup V)$, เมื่อ $\alpha:U\cup V\rightarrow R^+$ มี*

$$\alpha(u)+\alpha(v)\geq w(u,v)$$

สำหรับแต่ละเส้นเชื่อม $e=(u,v)$ ถ้า w เป็นตัวเลขจำนวนเต็มเราสามารถหา α ที่เป็นจำนวนเต็มได้

ทฤษฎีบทดังกล่าวโดยแก่นแท้แล้วเป็นทฤษฎีบทภาวะคู่กันของการโปรแกรมเชิงเส้น (linear programming duality) นั่นเอง ทฤษฎีที่กล่าวมาแล้วให้วิธีการแก้จำแนกกราฟที่มีการจับคู่แบบสมบูรณ์ แต่ไม่ได้กล่าวถึงวิธีการแก้ปัญหา ทฤษฎีบทต่อไปเป็นทฤษฎีบทของ Berge (1957) ซึ่งจำแนกลักษณะของการจับคู่ที่เป็นการจับคู่สูงสุด โดยใช้แนวคิดของ*วิถีสลับ* (alternating path) และ*วิถีแต่งเติม* (augmenting path) ซึ่งนิยามได้ดังนี้ ให้ G เป็นกราฟใดๆ (ไม่จำเป็นต้องเป็นกราฟสองส่วน) และ M เป็นการจับคู่ใดๆ บน G เรียก $P=v_1,v_2,...,v_m$ เป็น*วิถีสลับที่สัมพันธ์กับ M* (M-alternating path) ถ้า $(v_i,v_{i+1})\in M$ ก็ต่อเมื่อ $(v_{i+1},v_{i+2})\notin M$ เรียกจุดยอด v เป็น*จุดยอดที่ไม่ถูกจับคู่* ถ้าไม่มีเส้นเชื่อมใน M ประชิดกับ v และ เรียกวิถีสลับที่เชื่อมจุดยอดที่ไม่ถูกจับคู่สองจุดว่า *วิถีแต่งเติมที่สัมพันธ์กับ M* (M-augmenting path) สังเกตว่าถ้าเราลบเส้นเชื่อมที่อยู่ในวิถีแต่งเติม P ออกจาก M แล้วเพิ่มเส้นเชื่อมที่อยู่ใน P แต่ไม่อยู่ใน M เข้าไปใน M จะได้การจับคู่ที่มีขนาดใหญ่ขึ้น เรียกกระบวนการนี้ว่าการแต่งเติมการจับคู่ M ด้วย P

ทฤษฎีบทที่ 3 *ให้ M เป็นการจับคู่ในกราฟ G แล้ว M จะเป็นการจับคู่สูงสุดก็ต่อเมื่อไม่มีวิถีแต่งเติมใน G ที่สัมพันธ์กับ M*

<u>ปัญหาการจับคู่สองส่วนเชิงขนาด</u>

นิยาม

ให้กราฟสองส่วน $G=(U\cup V,E)$ *ปัญหาการจับคู่สองส่วนเชิงขนาด* คือ ปัญหาการหาการจับคู่ M บนกราฟสองส่วนที่มีขนาดสูงสุด เรียกอีกอย่างว่า*ปัญหาการจับคู่สูงสุดในกราฟสองส่วน* (maximum bipartite matching)

เทคนิคในการแก้ปัญหา

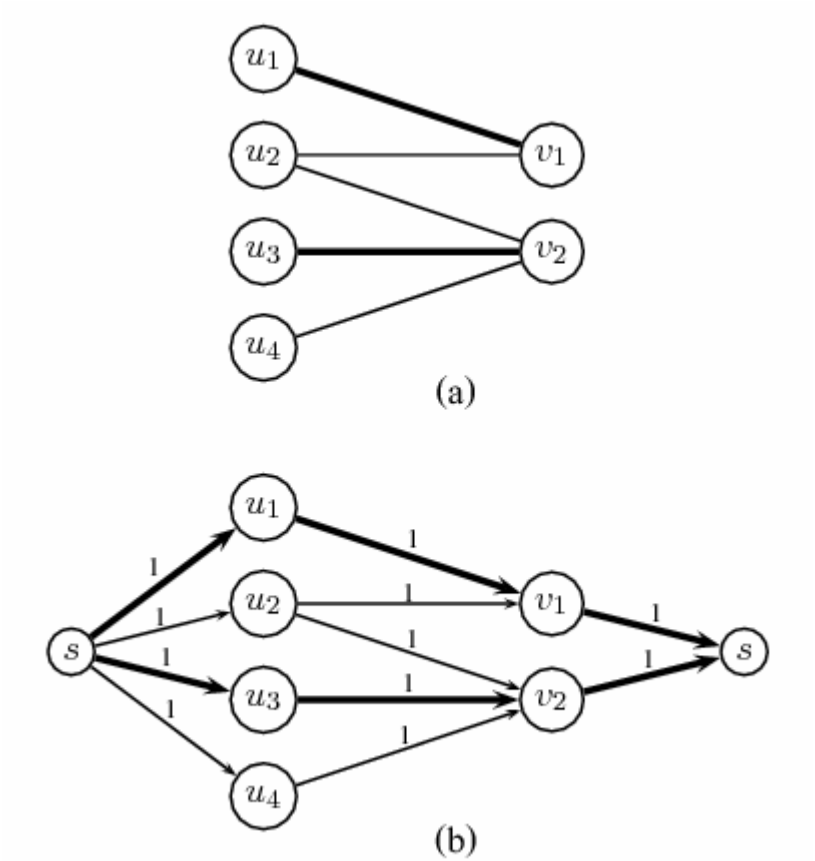
จากทฤษฎีบทที่ได้กล่าวมาแล้วนำไปสู่เทคนิคพื้นฐานในการแก้ปัญหาการจับคู่สูงสุดในกราฟสองส่วน $G=(U \cup V,E)$ หลายแบบดังนี้

วิธีแบบฮังกาเรียน (Hungarian method)

วิธีนี้จะเพิ่มขนาดของการจับคู่ M โดยหาวิธีแต่งเติม P แล้วแต่งเติมการจับคู่ M ด้วย P ซึ่งทำโดยลบเส้นเชื่อมใน $M \cap P$ และเพิ่มเส้นเชื่อมใน P-M เข้าไปใน M นั่นคือทำการสลับเส้นเชื่อมที่อยู่ใน M และ P ออกมาแล้วนำเส้นเชื่อมที่อยู่ใน M แต่ไม่อยู่ใน P เข้าไปแทน ซึ่งจะทำให้ M มีขนาดเพิ่มขึ้น เพราะจุดเริ่มต้นและจุดปลายของ P ติดกับจุดยอดที่ไม่ถูกจับคู่ จากทฤษฎีบทที่ 3 ของ Berge เมื่อไม่พบวิธีแต่งเติมที่สัมพันธ์กับ M ใน G จะได้ M เป็นการจับคู่สูงสุดใน G

ลดทอนเป็นปัญหาการไหลในเครือข่าย (reduction to network flow problems)

วิธีนี้จะมองปัญหาการจับคู่สองส่วนเป็นกรณีเฉพาะของปัญหาการไหลสูงสุด ซึ่งลดทอนปัญหาโดยระบุทิศทางของเส้นเชื่อมจากจุดยอดใน U ไปจุดยอดใน V แล้วเพิ่มแหล่งต้นทาง s และ แหล่งปลายทาง t จากนั้นสร้างเส้นเชื่อมระบุทิศทางจาก s ต่อไปยังทุกจุดยอดใน U และสร้างเส้นเชื่อมระบุทิศทางจากทุกจุดยอดใน V ไปยัง t กำหนดให้ทุกเส้นเชื่อมมีความจุเป็น 1 ตัวอย่างการลดทอนปัญหาแสดงในภาพที่ 3 ภาพ (a) แสดงปัญหาการจับคู่สูงสุดโดยเส้นทึบแสดงการจับคู่สูงสุด และภาพ (b) แสดงปัญหาการไหลสูงสุดโดยเส้นทึบแสดงการไหลสูงสุด รายละเอียดของปัญหาการไหลสูงสุดจะกล่าวถึงในหัวข้อปัญหาการไหลสูงสุด



ภาพที่ 3 ตัวอย่างการลดทอนปัญหาการจับคู่สูดงสุดไปเป็นปัญหาการไหลสูงสุด ภาพ (a) แสดงปัญหาการจับคู่สูงสุด และ ภาพ (b) แสดงปัญหาการไหลสูงสุดเส้นทึมแสดงการไหล

ตารางที่ 1 ตารางสรุปเวลาการทำงานของอัลกอริทึมที่สำคัญสำหรับแก้ปัญหาการจับคู่สูงสุดในกราฟสองส่วน

ปี	เจ้าของผลงาน	เวลาการทำงาน	วิธีการ
1931	Kɔnig	$O(nm)$	Hungarian method
1973	Hopcroft and Karp	$O(\sqrt{(n)}m)$	Blocking flow

n แทนจำนวนของจุดยอด และ m แทนจำนวนของเส้นเชื่อม

ปัญหาการจับคู่สองส่วนเชิงน้ำหนัก

นิยาม

ให้กราฟสองส่วน $G=(U\cup V,E)$ และ ฟังก์ชันน้ำหนัก $w:E\rightarrow R$ *ปัญหาการจับคู่สองส่วนเชิงน้ำหนัก* (weighted bipartite matching problems) คือ ปัญหาการหาการจับคู่ที่มีน้ำหนักรวมสูงสุดที่สุด เรียกอีกอย่างว่า*ปัญหาการจับคู่น้ำหนักสูงสุดในกราฟสองส่วน* (maximum-weight bipartite matching problems) และสมมูลกับ*ปัญหาการจับคู่สูงสุดที่มีน้ำหนักต่ำสุดในกราฟสองส่วน* (minimum-weight maximum bipartite matching problems) ซึ่งเป็นปัญหาเดียวกับปัญหาการมอบหมายงาน (assignment problems)

เทคนิคในการแก้ปัญหา

เทคนิคในการแก้ปัญหาการจับคู่สูงสุดสามารถนำมาใช้ในปัญหาการจับคู่เชิงน้ำหนักได้ โดยใช้ทฤษฎีบทภาวะคู่กันของการโปรแกรมเชิงเส้น ซึ่งจากทฤษฎีบทของ Egerváry ทำให้ทราบว่าสำหรับการจับคู่ที่มีน้ำหนักสูงสุด M ใดๆ จะมีฟังก์ชัน $\alpha:V\rightarrow R$ ที่สอดคล้องกับคุณสมบัติสองข้อด้านล่าง

$$\alpha(u)+\alpha(v)\geq w(e),\forall(u,v)\in E$$

$$\alpha(u)+\alpha(v)=w(e),\forall(u,v)\in M$$

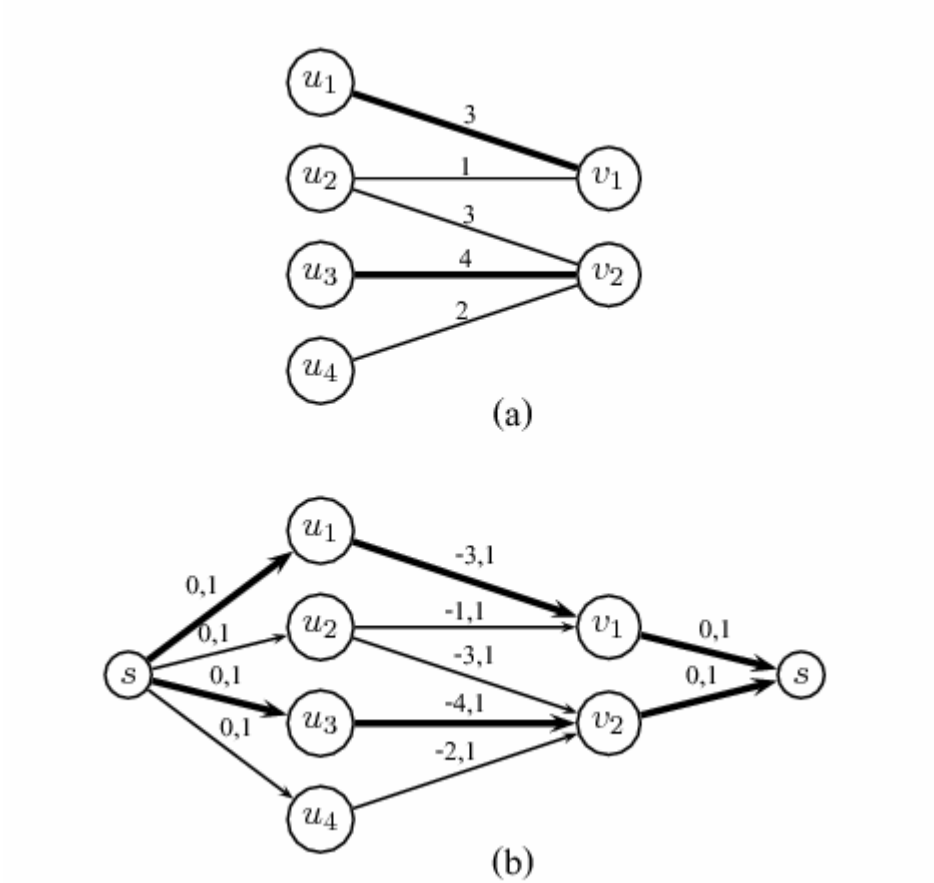
เราจะเรียก α ว่าเป็น*ตัวแปรคู่กัน (dual variables)* ทฤษฎีบทข้างต้นสามารถอธิบายอีกแบบได้ว่า เส้นเชื่อมที่สามารถอยู่ในการจับคู่ที่มีน้ำหนักสูงสุดได้นั้นน้ำหนักของเส้นเชื่อมต้องเท่ากันพอดีกับผลรวมของตัวแปรคู่กันที่เส้นเชื่อมนั้นประชิดอยู่ เรียกเส้นเชื่อมเหล่านี้ว่า*เส้นเชื่อมที่ยอมรับได้* (admissible edge) และเรียกกราฟที่ประกอบด้วยเส้นเชื่อมที่ยอมรับได้ว่า *กราฟที่ยอมรับได้* (admissible graph) ซึ่งนำไปสู่เทคนิคพื้นฐานในการแก้ปัญหาการจับคู่เชิงน้ำหนักที่สำคัญดังนี้

วิธีแบบฮังกาเรียน (Hungarian method)

ใช้การปรับค่าตัวแปรคู่กัน และเพิ่มขนาดของการจับคู่ M โดยหาวิธีแต่งเติม P ในกราฟที่ยอมรับได้ แล้วลบเส้นเชื่อมใน M ที่อยู่ใน P และเพิ่มเส้นเชื่อมใน P ที่ไม่อยู่ใน M เข้าไปใน M ซึ่งจะได้การจับคู่ที่มีขนาดเพิ่มขึ้น

ลดทอนเป็นปัญหาการไหลในเครือข่าย (reduction to network flow problems)

วิธีการนี้จะมองปัญหาการจับคู่สองส่วนเชิงน้ำหนักเป็นกรณีเฉพาะของปัญหาการไหลถูกสุด ซึ่งจะกล่าวถึงในหัวข้อการไหลสูงสุดที่มีค่าใช้จ่ายต่ำที่สุด การลดทอนทำโดยระบุทิศทางของเส้นเชื่อมจากจุดยอดใน U ไปจุดยอดใน V เพิ่มแหล่งต้นทาง s และ แหล่งปลายทาง t สร้างเส้นเชื่อมระบุทิศทางจาก s ต่อไปยังทุกจุดยอดใน U และสร้างเส้นเชื่อมระบุทิศทางจากทุกจุดยอดใน V ไปยัง t กำหนดค่าใช้จ่ายของเส้นเชื่อมจาก s ไป U และเส้นเชื่อมจาก V ไป t เท่ากับ 0 เส้นเชื่อมอื่นให้มีค่าใช้จ่ายเป็น $-w$ สำหรับรายละเอียดของทฤษฎีการไหลถูกสุดจะกล่าวถึงในหัวข้อทฤษฎีบทพื้นฐานในเรื่องปัญหาการไหลสูงสุดที่มีค่าใช้จ่ายต่ำที่สุด ตัวอย่างการลดทอนปัญหาแสดงในภาพที่ 4



ภาพที่ 4 ตัวอย่างการลดทอนปัญหาการจับคู่น้ำหนักสูงสุดเป็นปัญหาการไหลถูกสุด ภาพ (a) แสดงปัญหาการจับคู่น้ำหนักสูงสุด และ ภาพ (b) แสดงปัญหาการการไหลถูกที่สุด โดยเส้นทึบแสดงการไหล

ตารางที่ 2 ตารางสรุปเวลาการทำงานของอัลกอริทึมที่สำคัญสำหรับแก้ปัญหาการจับคู่น้ำหนักสูงสุดในกราฟสองส่วน

ปี	เจ้าของผลงาน	เวลาการทำงาน	วิธีการ
1970	Edmonds and Karp	$O(n \cdot SP_+(n,m,W)$	reduction to MCMF
1989	Gabow and Tarjan	$O(n^{1/2} m \log nW)$	Hungarian + Scaling

n แทนจำนวนจุดยอด, m แทนจำนวนเส้นเชื่อม, W แทนน้ำหนักสูงสุดของเส้นเชื่อม และ $SP_+(n,m,W)$ แทนเวลาที่ใช้ในการหาเส้นทางสั้นที่สุดเมื่อความยาวเส้นเชื่อมไม่ติดลบ

ทฤษฎีการไหลในเครือข่าย

ให้กราฟระบุทิศทาง $G=(V,E)$ กำหนดฟังก์ชัน*ความจุ*ของแต่ละเส้นเชื่อมเป็น $u:E\rightarrow R^+$ โดยสมมติว่าเส้นเชื่อมที่ไม่อยู่ในกราฟมีความจำเป็น 0 ให้*แหล่งต้นทาง* (source) $s\in V$ และ*แหล่งปลายทาง* (sink) $t\in V$ *การไหลจาก s ไป t* คือฟังก์ชัน $V\times V\rightarrow R$ ที่มีคุณสมบัติคือ

1 *สมมาตรเสมือน* (skew symmetry) คือ $f(v,a)=-f(a,v)$

2 *เงื่อนไขความจุ* (capactiy constraint) คือ $f(v,a)\leq u(v,a)$

3 *การอนุรักษ์การไหล* (flow conservation constraint) คือ $\sum_{a\in V}f(a,v)=0$ สำหรับทุกจุดยอด $v\in V-s,t$

ค่าของการไหล (value of flow) คือ ผลรวมของการไหลที่วิ่งออกจาก s ซึ่งเท่ากับผลรวมของการไหลที่วิ่งเข้าสู่ t เขียนแทนด้วย $|f|=val(f)=\sum_{v\in V}f(s,v)=\sum_{v\in V}f(v,t)$ *ค่าใช้จ่ายของการไหล* (cost of flow) เขียนแทนด้วย $cost(f)=\sum_{e\in E}f(e)\cdot c(e)$ การไหลที่ไม่สามารถเพิ่มปริมาณการไหลได้โดยไม่เปลี่ยนเส้นทางการไหลเรียกว่า *การไหลปิดล้อม* (blocking flow) นิยาม *ความจุตกค้าง*ของเส้นเชื่อม เป็น $r_f(e)=u(e)-f(e)$ คือ ค่าความจุที่เหลืออยู่ของเส้นเชื่อมหลังการไหลผ่านวิ่งผ่าน และจะได้นิยามของ*กราฟตกค้าง* เป็น $R_f(G)=(V,E_f)$ เมื่อ $E_f=\{e|e\in E,r_f(e)>0\}$ คือ กราฟที่ประกอบด้วยเส้นเชื่อมที่มีความจุตกค้างมากกว่า 0 ในมุมมองของการไหลเราจะนิยาม*วิถีแต่งเติมที่สัมพันธ์กับ f* เป็นวิถีจาก s ไป t ใน $R_f(G)$ และ *ส่วนตัดระหว่าง s และ t* คือ เซตของเส้นเชื่อม $X\in E$ ที่เมื่อลบ X ออกจาก G แล้วไม่มีวิถีจากไป s ไป t โดยคิดค่าใช้จ่ายของส่วนตัดระหว่าง X เป็นผลรวมของความจุของเส้นเชื่อมใน X คือ $c(X)=\sum_{e\in X}u(e)$

ทฤษฎีพื้นฐาน

ปัญหาการการไหลสูงสุด (maximum flow problems) เป็นปัญหาที่รู้จักกันดีปัญหาหนึ่ง ซึ่งปัญหานี้สามารถนำมาใช้ในการแก้ปัญหาการจับคู่สองส่วนเชิงขนาดได้ อัลกอริทึมที่ใช้ในการแก้ปัญหาการไหลสูงสุดนี้มีพื้นฐานมาจากการแต่งเติมการไหล (flow augmentation) ตามทฤษฎีบทของ Ford and Fulkerson (1954, 1957) (Schrijver, 2003)

ทฤษฎีบทที่ 4 *การไหล f จะเป็นการไหลสูงสุดก็ต่อเมื่อไม่มีวิถีแต่งเติมที่สัมพันธ์กับ f*

สำหรับทฤษฎีบทพื้นฐานที่สำคัญอีกบทหนึ่ง ซึ่งแสดงให้เห็นว่าถ้าความจุในเส้นเชื่อมเป็นจำนวนเต็มแล้ว การไหลสูงสุดจะมีปริมาณเป็นจำนวนเต็มด้วยคือ ทฤษฎีบทของ Ford and Fulkerson (1954, 1956) (Schrijver, 2003) ซึ่งรู้จักกันดีในชื่อทฤษฎีบทการไหลสูงสุด-ส่วนตัดต่ำสุด (max-flow min-cut theorem)

ทฤษฎีบทที่ 5 (ทฤษฎีบท การไหล-สูงสุด ส่วนตัด-ต่ำสุด (Ford and Fulkerson, 1954, 1956)) *ให้ $G=(V,E)$ เป็นกราฟแบบมีทิศทาง, ให้ $s,t \in V$, และให้ $u:E \rightarrow R^+$ แล้วปริมาณการไหลสูงสุดจาก s ไป t ภายใต้ u จะเท่ากับความจุส่วนตัดต่ำสุดระหว่าง s และ t*

ทฤษฎีบทที่กล่าวไปแล้วนำไปสู่วิธีการแก้ปัญหาที่มีเวลาการทำงานขึ้นกับปริมาณการไหลสูงสุด โดยใช้วิธีการแต่งเติมการไหล ไปจนกว่าจะไม่มีวิถีแต่งเติมเหลืออยู่

<u>ปัญหาการไหลสูงสุด</u>

นิยาม

ให้กราฟ $G=(V,E)$, แหล่งต้นทาง (source) s, แหล่งปลายทาง (destination) t และ ฟังก์ชันความจุ $u:E \rightarrow R^+$ ปัญหาการไหลสูงสุด คือ ปัญหาการหาการไหลจาก s ไป t ที่มีค่าการไหลสูงที่สุด

เทคนิคในการแก้ปัญหา

เทคนิคที่สำคัญในการแก้ปัญหาการไหลสูงสุดมีดังนี้

การไหลปิดล้อม (Blocking Flow)

เป็นเทคนิคสำหรับหาการไหลสูงสุดโดยหาการไหลปิดล้อมบนกราฟตกค้าง และใช้การไหลปิดล้อมในการแต่งเติมการไหล วิธีการนี้ถูกนำมาใช้ครั้งแรกโดย Dinic (1970)

Push-Relabel

วิธีการนี้ถูกเสนอแนวคิดขึ้นโดย Karzanov (1974) ในขั้นตอนแรกจะเริ่มจากการหา *การไหลภายหน้า* (preflow) ซึ่งเป็นฟังก์ชัน $E \rightarrow R$ ที่มีคุณสมบัติคล้ายการไหล คือ เป็นไปตามเงื่อนไขความจุ แต่ไม่เป็นตามเงื่อนไขอนุรักษ์การไหล วิธีการนี้จะประกอบด้วยฟังก์ชันการทำงานหลักคือ push และ relabel โดยจะปรับการไหลภายหน้า โดยใช้ฟังก์ชัน push ดันการไหลไปยังแหล่งปลายทางให้ได้มากที่สุด ซึ่งจะมี*การไหลส่วนเกิน* (excess flow) เหลืออยู่ในแต่ละจุดยอด เมื่อไม่สามารถดันการไหลไปได้อีกการไหลส่วนเกินนี้จะถูกดันกลับไปที่แหล่งต้นทาง จนกระทั่งไม่มีการไหลส่วนเกินเหลืออยู่ในกราฟ จะได้การไหลภายหน้าในรอบสุดท้ายเป็นไปตามเงื่อนไขอนุรักษ์การไหล และเป็นการไหลสูงสุด เทคนิคเหล่านี้มักจะถูกนำมาใช้ร่วมกัน นอกจากนี้ยังมีการใช้เทคนิคการปรับอัตราส่วน (scaling algorithm) ปรับอัตราส่วนความจุที่นำมาพิจารณาให้มีค่าน้อยลง ซึ่งสามารถช่วยให้แก้ปัญหาได้ง่ายขึ้น เทคนิคนี้ถูกเสนอโดย Edmonds and Karp (1972) และ Gabow (1983) ซึ่งพัฒนาขึ้นเพื่อแก้ปัญหาเกี่ยวกับเครือข่ายหลายปัญหา

ตารางที่ 3 ตารางสรุปเวลาการทำงานของอัลกอริทึมที่สำคัญสำหรับแก้ปัญหาการไหลสูงสุด

ปี	เจ้าของผลงาน	เวลาการทำงาน	วิธีการ
1951	Dantzig	$O(n^2mU)$	Simplex
1956	Ford and Fulkerson	$O(n^2mU)$	augmenting path
1970	Dinic	$O(nm^2)$	Blocking flow
1970	Dinic	$O(n^2m)$	Blocking flow
1972	Edmond and Karp	$O(nm^2)$	augmenting path
1972	Edmond and Karp	$O(m^2\log U)$	augmenting path, scaling
1973	Dinic	$O(nm\log U)$	augmenting path, scaling
1974	Karzanov	$O(n^3)$	augmenting path, scaling
1977	Cherkassky	$O(n^2\sqrt{(m)})$	blocking preflow
1980	Galil and Naamad	$O(nm\log^2 m)$	augmenting path, data structure
1983	Sleator and Tarjan	$O(nm\log n)$	augmenting path, dynamic tree
1986	Goldberg and Tarjan	$O(nm\log n^2/m)$	augmenting path, dynamic tree
1987	Ahuja และ Orlin	$O(nm+n^2\log U)$	push and relabeling, scaling
1987	Ahuja, Orlin and Tarjan	$O(nm\log(n/m)\sqrt{(\log U)})$	push and relabeling, scaling, dynamic tree
1989	Cheriyan and Hagerup	$O(nm+n^2\log^2 n)$	Randomized
1990	Cheriyan, Hagerup and Mehlhorn	$O(n^3/\log n)$	-
1990	Alon	$O(n(m+n^{5/3}\log n))$	derandomized from previous work
1992	King, Rao and Tarjan	$O(nm+n^{2+\epsilon})$	-
1993	Phillips and Westbrook	$O(nm\log_{m/n} n+n^2\log^{2+\epsilon} n)$ (for all $\epsilon>0$)	-
1994	King, Rao and Tarjan	$O(nm\log_{m/(n\log_n)} n)$	-
1998	Goldberg and Rao	$O(n^{2/3}m\log n^2/m\log U)$	binary blocking flow

n แทนจำนวนจุดยอด, m แทนจำนวนเส้นเชื่อม, U แทนขนาดความจุมากที่สุดของเส้นเชื่อม

ปัญหาการไหลสูงสุดที่มีค่าใช้จ่ายต่ำที่สุด

ปัญหาการไหลสูงสุดที่มีค่าใช้จ่ายต่ำที่สุด (minimum-cost maximum-flow problem) หรือเรียกโดยย่อว่า*ปัญหาการไหลถูกสุด* (minimum-cost flow problems) เป็นปัญหาที่รู้จักกันดีปัญหาหนึ่ง ซึ่งสามารถมองเป็นการนัยทั่วไปของปัญหาการไหลสูงสุด และปัญหาการหาวิถีที่สั้นที่สุด วิถี *s-t* ที่สั้นที่สุดสามารถนิรนัยได้จากการไหล *s-t* ถูกสุดที่มีปริมาณ 1 หน่วย และการไหล *s-t* สูงสุดจะเท่ากับการไหล *s-t* ถูกสุด ถ้ากำหนดค่าใช้จ่ายของเส้นเชื่อมออกจาก *s* เป็น -1 และเส้นเชื่อมที่เหลือเป็น 0 (สมมติว่าไม่มีเส้นเชื่อมเข้าหา *s*) ปัญหาที่มีความสัมพันธ์กับปัญหาการไหลถูกสุดคือ *ปัญหาการไหลเวียนที่ถูกสุด* (minimum-cost circulation) และ *ปัญหาการขนถ่ายที่ถูกสุด* (minimum-cost transshipment) ซึ่งปัญหาทั้งสามนั้นสมมูลกัน เมื่อกล่าวถึงอัลกอริทึมสำหรับแก้ปัญหาในกลุ่มนี้ จะหมายถึงปัญหาที่ตรงกับรูปแบบของทฤษฎีบทที่ระบุปัญหา

นิยาม

ปัญหาการไหลถูกที่สุดสามารถนิยามได้ดังนี้ ให้กราฟ $G=(V,E)$, แหล่งต้นทาง (source) s, แหล่งปลายทาง (destination) t, ฟังก์ชันความจุ $u:E\rightarrow R^+$ และ ฟังก์ชันค่าใช้จ่ายของแต่ละเส้นเชื่อม $c:E\rightarrow R$ สำหรับฟังก์ชัน $f:E\rightarrow R$ ใดๆ นิยามค่าใช้จ่ายของ f เป็น $cost(f)=\sum_{e\in E}c(e)f(e)$ ปัญหาการไหลถูกที่สุด คือ ปัญหาการหาการไหลสูงสุดจาก s ไป t ที่มีค่าใช้จ่ายน้อยน้อยที่สุดเทียบกับการไหลสูงสุดอื่น ให้ฟังก์ชันอุปสงค์ $d:E\rightarrow R^+$ *การไหลเวียนที่มีอุปสงค์ d* คือ ฟังก์ชัน $f:E\rightarrow R$ ที่มีเงื่อนไขคือ

1 *สมมาตรเสมือน* (skew symmetry) คือ $f(v,a)=-f(a,v)$
2 *เงื่อนไขความจุ* (capactiy constraint) คือ $f(v,a)\leq u(v,a)$
3 *การอนุรักษ์* (conservation constraint) คือ $\sum_{a\in V}f(a,v)=d(v)$

นิยามกราฟตกค้างของการไหลเวียนเช่นเดียวกับการไหล และนิยามค่าใช้จ่ายเป็น $cost(f)=1/2\sum_{e\in E}c(e)f(e)$ *วงรอบค่าใช้จ่ายลบ* (negative cost cylce) คือ วงรอบในกราฟตกค้างที่มีค่าใช้จ่ายรวมของทุกเส้นเชื่อมเป็นค่าลบ ปัญหาการไหลเวียนถูกสุด คือ ปัญหาการหาการไหลเวียนที่มีค่าใช้จ่ายถูกที่สุด

ทฤษฎีบทพื้นฐาน

เมื่อพิจารณาปัญหาการไหลถูกสุดในรูปของปัญหาการไหลเวียน และพิจารณาในกราฟตกค้าง $R_f(G)$ Ford and Fulkerson (1962) ได้แสดงความสัมพันธ์ระหว่างการไหลเวียนกับวงรอบค่าใช้จ่ายลบในกราฟตกค้างดังนี้

ทฤษฎีบทที่ 6 *ให้กราฟ $G=(V,E)$ และ $d,u,c:E\to R$ ให้ $f:E\to R$ เป็นการไหลเวียน แล้ว f จะเป็นการไหลเวียนถูกสุดก็ต่อเมื่อไม่มีวงรอบค่าใช้จ่ายลบใน $R_f(G)$*

ทฤษฎีบทดังกล่าวทำให้สามารถตรวจสอบว่าการไหลเวียนมีค่าใช้จ่ายต่ำที่สุดหรือไม่ได้ในเวลาพหุนาม และนำไปสู่วิธีการในการแก้ปัญหาการไหลถูกที่สุดโดยเริ่มจากการหาการไหลสูงสุด แล้วปรับค่าใช้จ่ายของการไหลโดยการกำจัดวงรอบค่าใช้จ่ายลบ นอกจากนี้ Ford and Fulkerson (1962) ยังได้ให้วิธีในการจำแนกการไหลเวียนที่ถูกที่สุดโดยพิจารณาภาวะคู่กันทางกำหนดการเชิงเส้น ในรูปของ*ฟังก์ชันราคา* p ซึ่งเป็นฟังก์ชันบนจุดยอด และค่าใช้จ่ายลดทอน $c_p(u,v)=p(u)-p(v)+c(u,v)$ ดังทฤษฎีบทต่อไปนี้

ทฤษฎีบทที่ 7 *ให้กราฟ $G=(V,E)$ และ $d,u,c:E\to R$ ให้ $f:E\to R$ เป็นการไหลเวียน แล้ว f จะเป็นการไหลเวียนถูกสุดถ้ามีฟังก์ชันราคา p ที่ $\forall e\in E$,*

$$c_p(u,v)<0 \Rightarrow f(v,w)=u(v,w)$$

ทฤษฎีบทแทรกนี้แสดงลักษณะของความสัมพันธ์แบบต่ำสุด-สูงสุด (min-max relation) สำหรับปัญหาการไหลถูกที่สุด ซึ่งนำไปสู่วิธีการแก้ปัญหาแบบแก้ปัญหาคู่กัน (dual approach)

เทคนิคในการแก้ปัญหา

ทุกอัลกอริทึมสำหรับแก้ปัญหาที่เป็นที่รู้จักและสามารถทำงานได้ในเวลาพหุนามใช้แนวคิดแบบการปรับอัตราส่วน ซึ่งเริ่มนำมาใช้โดย Edmonds and Karp (1972) ในแบบปรับอัตราส่วนความจุ และ มีการนำเทคนิคปรับอัตราส่วนค่าใช้จ่ายดังในงานของ Röck (1980) และงานของ Goldberg and Tarjan (1987) (1990) นอกจากนี้ยังมีการใช้การปรับอัตราส่วนทั้งความจุและค่าใช้จ่ายดังปรากฏในงานของ Ahuja et al. (1992) เทคนิคที่สำคัญที่นำมาใช้ร่วมกับเทคนิคการปรับอัตราส่วนมีดังต่อ

วิธีการกำจัดวงรอบค่าใช้จ่ายลบ (negative cost cycle canceling)

วิธีการในกลุ่มนี้จะเริ่มจากการหาการไหลสูงสุด f แล้วการกำจัดวงรอบค่าใช้จ่ายลบในกราฟตกค้าง $R_f(G)$ เพื่อลดค่าใช้จ่ายของการไหล ซึ่งผลลัพธ์ในขั้นตอนสุดท้ายเป็นการไหลสูงสุดที่ไม่มีวงรอบค่าใช้จ่ายลบเหลืออยู่จะเป็นการไหลถูกที่สุด ตามทฤษฎีบทที่ 4

วิธีแก้ปัญหาคู่กัน (dual approach)

วิธีการนี้มาจากทฤษฎีบทภาวะคู่กัน (duality theorem) ตามทฤษฎีบทที่ 2 โดยจะทำการหาฟังก์ชันราคาที่ทำให้เงื่อนไขเป็นจริง ซึ่งวิธีหนึ่งที่สามารถทำได้คือการใช้แต่งเติมการไหลผ่านวิถีแต่งเติมที่มีค่าใช้จ่ายต่ำที่สุด ซึ่งเป็นแนวคิดของฟังก์ชันสุดขีด จะเรียกว่า f ว่า *สุดขีด* (extreme) ถ้า $cost(f) \geq cost(f')$ สำหรับทุก $|f|=|f'|$ นั่นคือ f มีค่าใช้จ่ายถูกที่สุดในการไหลที่ยอมรับได้ที่มีค่าการไหลเท่ากัน

ตารางที่ 4 ตารางสรุปเวลาการทำงานของอัลกอริทึมที่สำคัญสำหรับหาการไหลถูกสุด

ปี	เจ้าของผลงาน	เวลาการทำงาน	วิธีการ
1972	Edmonds and Karp	$O(m\log U \cdot SP_+(n,m,K))$	solving dual, scaling
1988	Orlin	$O(m\log n \cdot SP_+(n,m,C))$	solving dual, scaling
1990	Golberg and Tarjan	$O(nm\log n^2/m\log nC)$	scaling

n แทนจำนวนจุดยอด, m แทนจำนวนเส้นเชื่อม, U แทนขนาดความจุมากที่สุดของเส้นเชื่อม, C แทนค่าใช้จ่ายสูงที่สุดของเส้นเชื่อม, และ $SP_+(n,m,C)$ แทนเวลาที่ใช้ในการหาเส้นทางสั้นที่สุดเมื่อความยาวเส้นเชื่อมไม่ติดลบ

การหาวิถีที่สั้นที่สุด

ปัญหาการหาวิถีที่สั้นที่สุด เป็นปัญหาพื้นฐานด้านเครือข่าย ในทางปฏิบัติมักจะพบปัญหานี้ได้ทั่วไป และมักจะเป็นปัญหาย่อยของปัญหาทางด้านเครือข่ายปัญหาอื่น นอกจากนี้ยังเป็นปัญหาที่มีความใกล้ชิดกับปัญหาการจับคู่ และปัญหาการไหลถูกสุดอีกด้วย รูปแบบของปัญหาในกลุ่มนี้ที่สำคัญคือปัญหาวิถีที่สั้นที่สุดจากแหล่งต้นทางเดียว (single-source shortest path) ปัญหาการหาวิถีที่สั้นที่สุดจากหลายแหล่งต้นทาง (multiple-source shortest path) และปัญหาการหาวิถีที่สั้น

ที่สุดของทุกคู่ (all-pair shortest path) ในที่นี้เราจะสนใจเฉพาะปัญหาการหาวิถีที่สั้นที่สุดจากแหล่งต้นทางเดียว หัวข้อถัดไปเราจะให้นิยามอย่างเป็นรูปนัยของปัญหานี้

นิยาม

ให้กราฟระบุทิศทาง $G=(V,E)$ และ ฟังก์ชันความยาว $l:E\rightarrow R$ และแหล่งต้นทาง $s\in V$ จุดมุ่งหมายของปัญหาการหาวิถีที่สั้นที่สุดจากแหล่งต้นทางเดียว คือ ต้องการหาระยะทางที่สั้นที่สุดจาก s ไปยังจุดยอดอื่นๆ ที่เหลือ หรือหา*วงรอบระยะทางลบ* คือ วงรอบที่มีผลรวมของความยาวของเส้นเชื่อมเป็นลบ ถ้า G มีวงรอบระยะทางลบเราจะเรียกว่าปัญหาไม่สามารถหาคำตอบได้ (infeasible problem) เพราะวิถีจาก s ที่ผ่านวงรอบระยะทางลบสามารถวิ่งซ้ำวงรอบเดิมให้มีระยะทางรวมสั้นลงได้เรื่อยๆ *ฟังก์ชันราคา* (price function) คือ ฟังก์ชัน $V\rightarrow R$ ให้ฟังก์ชันราคา p เรานิยามฟังก์ชัน*ค่าใช้จ่ายลดทอน* $l_p:E\rightarrow R$ ดังนี้

$$l_p(u,v)=l(u,v)+p(u)-p(v)$$

ถ้า $p(u)=p(v)=\infty$ เรานิยามให้ $l_p(u,v)=l(u,v)$ เราจะเรียกฟังก์ชันราคา p ที่ทำให้ $l_p\geq 0$ ว่าฟังก์ชันราคาที่เป็นไปได้ (feasible price function)

เทคนิคในการแก้ปัญหา

เทคนิคที่สำคัญในการแก้ปัญหาวิถีที่สั้นที่สุด คือ *วิธีปรับป้ายระยะทาง* (labeling method) ซึ่งมีการทำงานคือ แต่ละจุดยอด v จะระบุ*ป้ายระยะทาง* (label) $d(v)$ และมีสถานะเป็น *labelled*, *unlabelled* หรือ *scanned* โดยเริ่มต้นกำหนดให้จุดยอด s มีสถานะเป็น *labelled* และ $d(s)=0$ สำหรับทุก $v\in V-s$ กำหนดสถานะเป็น *unlabelled* และ $d(v)=\infty$ วิธีนี้จะปรับป้ายระยะทางสำหรับจุดยอด v ที่มีสถานะเป็น *labelled* โดยการดำเนินการ SCAN ซึ่งแสดงรายเอียดในภาพที่ โดยที่สำหรับทุกเส้นเชื่อม $(v,w)\in E$ ที่ $d(v)+l(v,w)<d(w)$ จะปรับค่า $d(w)\leftarrow d(v)+l(v,w)$ และกำหนดสถานะ w เป็น *labelled* เมื่อจบการดำเนินการจะกำหนดสถานะของ v เป็น *scanned* วิธีนี้จะจบการทำงานก็ต่อเมื่อ G ไม่มีวงรอบระยะทางลบ และถ้าจบการทำงานจะได้ $d(v)$ เป็นระยะทางที่สั้นที่สุดจาก s ไป v

Operation SCAN$(v, G = (U \cup V, E), M, d, H)$

1. for all $(v, w) \in E$
2. if $d(v) + l(v, w) < d(w)$
3. $d(w) \leftarrow d(v) + l(v, w)$
4. mark w as *labelled*
5. mark v as *scanned*

ภาพที่ 5 การดำเนินการ SCAN

นอกจากนี้ยังมีการใช้อัลกอริทึมแบบปรับอัตราส่วน (scaling algorithms) มาใช้ในการแก้ปัญหาด้วยดังงานวิจัยของ Gabow (1983), Gabow and Tarjan (1989) และ Goldberg (1993) อัลกอริทึมสำคัญที่เราจะกล่าวถึงคืออัลกอริทึมของ Bellman-Ford-Moore และอัลกอริทึมของ Dijkstra

อัลกอริทึมของ Bellman-Ford-Moore

อัลกอริทึมของ Bellman-Ford-Moore (ในงานวิจัยของ Bellman (1958), Ford (1962), Moore (1959)) จัดลำดับของการ SCAN จุดยอดที่มีสถานะ *labelled* ตามลำดับของการถูกเปลี่ยนสถานะเป็น *labelled* คือจุดยอดที่ถูกเปลี่ยนสถานะก่อนจะได้รับการ SCAN ก่อน อัลกอริทึมดังกล่าวมีเวลาการทำงานเป็น $O(nm)$ เมื่อ n คือจำนวนจุดยอด และ m คือจำนวนเส้นเชื่อม อัลกอริทึมนี้สามารถใช้ได้กรณีที่กราฟมีเส้นเชื่อมระยะทางลบ (แต่ไม่มีวงรอบระยะทางลบ) อัลกอริทึมของ Bellman-Ford-Moore เป็นอัลกอริทึมที่ดีที่สุดเท่าที่เป็นที่รู้จักที่ทำงานกับกราฟที่มีความยาวของเส้นเชื่อมเป็นจำนวนจริง

อัลกอริทึมของ Dijkstra

อัลกอริทึมของ Dijkstra (1959) เลือกจุดยอด v ที่มีสถานะ *labelled* ที่ป้ายระยะทาง $d(v)$ มีค่าต่ำสุดเป็นจุดยอดที่จะ SCAN เป็นตัวถัดไป ในกรณีที่กราฟไม่มีเส้นเชื่อมความยามลบอัลกอริทึมของ Dijkstra จะ SCAN แต่ละจุดยอดเพียงครั้งเดียว โดย

อาศัยแนวคิดที่ว่า $d(v)$ ป้ายระยะทางต่ำสุด ดังนั้นระยะทางจาก s ไปยังจุดยอดที่มีสถานะ *labelled* อื่น อ้อมมาที่ v จะต้องมีระยะทางไม่ต่ำกว่า $d(v)$ ดังนั้น $d(v)$ เป็นระยะทางที่สั้นที่สุดจาก s ไปยัง v แต่ถ้ามีเส้นเชื่อมที่มีความยาวลบ เส้นทางที่วิ่งอ้อมสามารถให้ระยะทางที่ต่ำกว่าได้ ทำให้ต้อง SCAN แต่ละจุดยอดมากกว่าหนึ่งครั้ง ซึ่งเราสามารถแสดงได้ว่าจำนวนครั้งของการแสดงอาจจะใช้เวลาเป็นแบบเลขชี้กำลัง (exponential time)

ในการใช้งานอัลกอริทึมของ Dijkstra หากใช้โครงสร้างข้อมูลฟีโบนัคชีฮีป (Fibonacci heap) มาช่วยจะได้เวลาการทำงาน $O(m+n\log n)$ ซึ่งเป็นเวลาที่ดีที่สุดเท่าที่เป็นที่รู้จักของอัลกอริทึมที่ทำงานบนกราฟที่มีความยาวเส้นเชื่อมเป็นจำนวนจริงไม่ลบ

อัลกอริทึมของ Dijkstra สามารถนำมาประยุกต์ใช้กับกราฟที่มีเส้นเชื่อมความยาวลบได้โดยใช้ฟังก์ชันราคาและคิดความยาวเส้นเชื่อมเป็นค่าใช้จ่ายลดทอน ซึ่งฟังก์ชันราคาที่เป็นไปได้จะให้ค่าใช้จ่ายลดทอนที่มีค่าไม่ลบ โดยที่วิถีที่สั้นที่สุดบนค่าใช้จ่ายลดทอนจะเป็นวิถีที่สั้นที่สุดบนความยาวเดิมด้วย

งานวิจัยในอดีตของปัญหาการจับคู่

ปัญหาการจับคู่ถูกกล่าวถึงครั้งแรกในหนังสือของ Lawler (2001) ซึ่งกล่าวถึงปัญหาการจับคู่แบบมีน้ำหนัก โดยมีจุดประสงค์เพื่อหาการจับคู่ที่มีน้ำหนักรวมของเส้นเชื่อมมากที่สุด ปัญหาดังกล่าวสามารถแก้ได้โดยใช้อัลกอริทึมแบบกรีดี้ (greedy algorithm) ปัญหาการจับคู่นี้ถูกนำมากล่าวถึงอีกครั้งโดย Harvey et al. (2003) ซึ่งมองปัญหานี้เป็นการกำหนดงานให้กับเครื่องจักร และสามารถกำหนดงานหลายงานให้กับเครื่องจักรเครื่องเดียวได้ โดยคิดค่าใช้จ่ายเป็นเวลาในการรอการทำงานของแต่ละงาน พร้อมทั้งเสนออัลกอริทึมสองวิธีสำหรับแก้ปัญหาในกราฟ $G=(U\cup V,E)$ โดยทำงานในเวลา $O(|U||E|)$ และ $O(|U|^{3/2}|E|)$ นอกจากนี้ Harvey et al. (2003) ยังได้แสดงวิธีการลดทอนปัญหา (problem reduction) เป็นปัญหาการไหลสูงสุดในเครือข่ายที่มีค่าใช้จ่ายต่ำที่สุด (minimum-cost maximum-flow problems) และปัญหาการจับคู่สองส่วนเชิงน้ำหนัก (weighted bipartite matching problems) (อยู่ในส่วนของการนำเสนอผลงาน แต่ไม่ปรากฏในวารสารวิชาการ)

วิธีการ

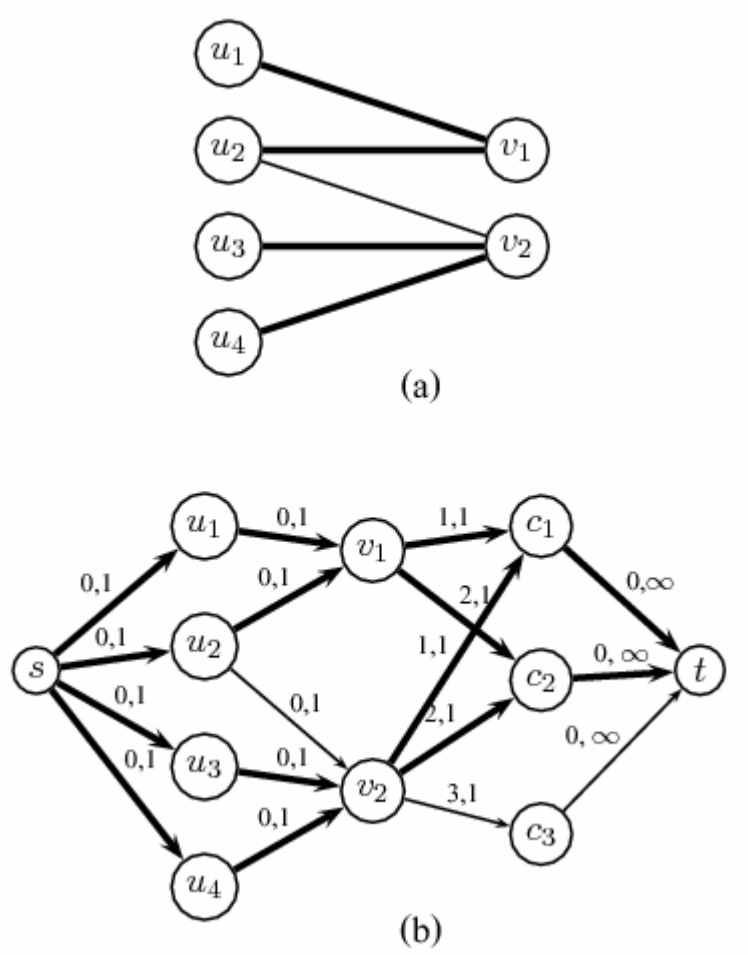
ในการแก้ปัญหาการจับกึ่งคู่ เราใช้วิธีการลดทอนปัญหาให้ปัญหาอยู่ในรูปที่สามารถแก้ได้ง่ายขึ้น และศึกษาโครงสร้างของปัญหา เพื่อหาคุณสมบัติเฉพาะของปัญหาหลังจากทำการลดทอน เราจะแบ่งเนื้อหาของบทนี้ออกเป็น 2 ส่วน คือ ส่วนแรก จะกล่าวถึงการลดทอนปัญหา และส่วนที่สองจะกล่าวถึงคุณสมบัติของปัญหา

การลดทอนปัญหา

กรณีที่กราฟไม่มีน้ำหนัก

วิธีการแปลงปัญหาการจับกึ่งคู่ถูกที่สุดไปเป็นปัญหาการไหลที่ถูกที่สุดนั้น เราจะใช้วิธีการเดียวกับใน (Harvey et al., 2003) กล่าวคือให้ $G=(U\cup V,E)$ เราจะสร้างกราฟระบุทิศทาง N ดังนี้ ให้ Δ แทนจำนวนดีกรีมากที่สุดของจุดยอดใน V เพิ่มเซตของจุดยอดที่ต่อไปจะเรียกว่า *ศูนย์ค่าใช้จ่าย* $C=\{c_1,c_2,...,c_\Delta\}$ และเชื่อมต่อแต่ละจุดยอด $v\in V$ ไปยัง c_i ด้วยเส้นเชื่อมที่มีความจุ 1 และค่าใช้จ่าย i, สำหรับทุก $1\le i\le deg(v)$ หลังจากนั้นเพิ่ม s และ t แทนจุดยอดที่เป็นแหล่งต้นทางและแหล่งปลายทาง สำหรับแต่ละจุดยอดใน U เพิ่มเส้นเชื่อมจาก s ไปยังจุดยอดเหล่านั้นด้วยค่าใช้จ่าย 0 และ มีความจุ 1 หน่วย สำหรับแต่ละศูนย์ค่าใช้จ่าย c_i เพิ่มเส้นเชื่อมไปยัง t ด้วยค่าใช้จ่าย 0 และความจุเป็นอนันต์ สุดท้ายระบุทิศทางเส้นเชื่อมแต่ละเส้น $e\in E$ จาก U ไปยัง V และกำหนดให้มีความจุ 1 และค่าใช้จ่าย 0 จะได้กราฟใหม่ N ที่ประกอบด้วยจุดยอด $O(|U|+|V|)$ จุด และเส้นเชื่อม $O(|E|)$ เส้น ดังตัวอย่างในภาพที่ 6 ซึ่ง Harvey et. al. (2003) ได้แสดงให้เห็นว่าวิธีการลดทอนดังกล่าวถูกต้อง กล่าวคือ การจับคู่ M ใดๆ ใน G สามารถแปลงเป็นการไหล f ใน N โดยให้ f ไหลผ่านเส้นเชื่อมจาก U ไป V ตามที่ระบุใน M และให้การไหลจาก $v\in V$ ไปยัง t เลือก c_i ที่ถูกที่สุดเท่าที่สามารถเลือกได้ นั่นคือค่าใช้จ่ายของการจับกึ่งคู่ที่ดีที่สุดใน G จะเท่ากับค่าใช้จ่ายของการไหลสูงสุดที่ถูกที่สุดจาก s ไปยัง t ใน N

สังเกตว่าในการแปลงนี้ ค่าใช้จ่ายสูงสุดจะเป็น $O(|U|)$ ซึ่งทำให้สามารถใช้แนวคิดของการปรับอัตราส่วนของค่าใช้จ่าย (cost-scaling) ในการแก้ปัญหาได้โดยอัลกอริทึมในงานวิจัยชิ้นนี้ถูกค้นพบจากข้อสังเกตดังกล่าว



<u>ภาพที่ 6</u> ตัวอย่างการลดทอนปัญหาการจับคู่แบบไม่มีน้ำหนักเป็นปัญหาการไหลที่ถูกที่สุด เส้นเชื่อมแต่ละเส้นจะมีค่าใช้จ่ายและค่าความจุระบุอยู่ในตัวเลขทงซ้ายและขวาตามลำดับ

<u>กรณีที่กราฟมีน้ำหนัก</u>

ในการแก้ปัญหาเราลดทอนปัญหาเป็นปัญหาการจับคู่สูงสุดที่มีน้ำหนักต่ำสุดในกราฟสองส่วน (minimum-weight maximum bipartite matching) ให้กราฟสองส่วน $G=(U\cup V,E)$ และฟังก์ชันน้ำหนัก $w:E\rightarrow R$ สังเกตจุดยอด $v\in V$ ใดๆ ในปัญหาการจับคู่ สามารถประชิดกับเส้นเชื่อมในการจับคู่ M ได้ เท่ากับจำนวนดีกรีของ v ในกราฟ G เพื่อให้เงื่อนไขของการจับคู่เชิงน้ำหนักเป็นจริง การแปลงปัญหาต้องสร้างจุดยอดที่สมนัยกันอย่างน้อยเท่ากับจำนวนดีกรีของ v และเส้นเชื่อม $(u,v)\in E$ ใดๆ ต้องสร้างเส้นเชื่อมที่สมนัยกับเท่ากับจำนวนดีกรีของ v เช่นกัน เขียนแทนดีกรีของ v ในกราฟ G ด้วย $deg_G(v)$ ถ้าเป็นที่เข้าใจชัดเจนในบริบทว่าหมายถึงกราฟใดๆ เราจะเขียนเป็น $deg(v)$ แทน การกำหนดน้ำหนักให้กับเส้นเชื่อม พิจารณาค่าใช้จ่ายที่จุดยอด v ซึ่งเกิดจาก (u,v) เมื่อ (u,v) อยู่ในการจับคู่ M และเป็นค่ามากที่สุดลำดับที่ i ของเส้นเชื่อมที่อยู่

ใน M ที่ประชิดกับ v (u,v) จะทำให้เกิดค่าใช้จ่าย $i \cdot w(u,v)$ ในค่าใช้จ่ายที่ v โดยให้จุดยอด v^i ที่สร้างจาก v ใช้รองรับ เส้นเชื่อมหนักสุดลำดับที่ i ใน M ที่ประชิดกับ v เราจะกำหนดน้ำหนักให้กับเส้นเชื่อมเท่ากับ i เท่าของน้ำหนักในกราฟตั้งต้น เราจะเขียนการลดทอนปัญหาอย่างเป็นรูปนัยดังนี้

ให้กราฟสองส่วน $G=(U \cup V,E)$ และฟังก์ชันน้ำหนัก $w:E \rightarrow R$ ของปัญหาการจับกึ่งคู่แบบมีน้ำหนัก สร้างกราฟสองส่วน $G'=(U \cup V',E')$ และฟังก์ชันน้ำหนัก $w':E' \rightarrow R$ เป็นกราฟที่สมนัยกันของปัญหาการจับคู่เชิงน้ำหนัก สำหรับทุกจุดยอด $v \in V$ ที่มีดีกรี (degree) d สร้างจุดยอด $v^1,v^2,...,v^d$ ใน V' เพื่อความสะดวกในการกล่าวถึงเราจะเขียน $S(v)$ แทนเซตของจุดยอดเหล่านี้ สำหรับแต่ละเส้นเชื่อม $(u,v) \in E$ ที่ประชิดกับ v สร้างเส้นเชื่อม d เส้น $(u,v^1),(u,v^2),...,(u,v^d)$ ใน E' พร้อมกำหนดน้ำหนักเป็น $1 \cdot w(u,v), 2 \cdot w(u,v),...,d \cdot w(u,v)$ ตามลำดับ นั่นคือ $w'(u,v^i)=i \cdot w(u,v)$

การลดทอนปัญหาโดยวิธีนี้จะได้กราฟ V' มีขนาดเท่ากับผลรวมของดีกรีของทุกจุดยอดใน V บน G ซึ่งเท่ากับจำนวนเส้นเชื่อมใน E และแต่ละเส้นเชื่อมใน E จะมีเส้นเชื่อมที่สมนัยกันไม่เกิน $|U|$ เส้น นั่นคือจำนวนเส้นเชื่อมใน E' จะมีขนาดไม่เกิน $|U||E|$ จะได้ว่ากราฟ G' มีจำนวนจุดยอด $|U|+|E|$ จุด และมีเส้นเชื่อมไม่เกิน $|U||E|$ เส้น และน้ำหนักสูงสุดของเส้นเชื่อมใน G' จะมีน้ำหนักไม่เกิน $|U||W|$ เมื่อ W คือ น้ำหนักสูงสุดของเส้นเชื่อมใน G ตัวอย่างการลดทอนปัญหาแสดงในภาพที่ 7 และความถูกต้องของการลดทอนปัญหาแสดงในทฤษฎีบทย่อยต่อไปนี้

ทฤษฎีบทย่อยที่ 8 *ให้ M' เป็นคำตอบของปัญหาจับคู่สูงสุดที่มีน้ำหนักต่ำสุดในกราฟสองส่วน $G'=(U \cup V',E')$ จะมีการจับกึ่งคู่ M ที่สมนัยกับ M' บน $G'=(U \cup V,E)$ และ M เป็นการจับกึ่งคู่ที่ดีที่สุด*

พิสูจน์

พิจารณาจากการจับคู่สูงสุดที่มีน้ำหนักต่ำสุด M' ใน G' เราสามารถสร้างการจับกึ่งคู่ M ที่สมนัยกัน และมีค่าใช้จ่ายเท่ากันได้ โดยสำหรับแต่ละเส้นเชื่อม $(u,v^i) \in M'$ สร้างเส้นเชื่อม (u,v) ใน M เขียน $N(X)$ แทนจุดยอดที่ประชิดกับจุดยอดใน X จากการสร้าง G' สำหรับจุดยอด $v \in V$ มีจุดยอดที่สมนัยกัน $v^1,v^2,...,v^{deg(v)} \in V'$ ทำให้เราได้ว่าทุก $X \subseteq U$ จะมี $|U| \leq |N(X)|$ ตามทฤษฎีบทที่ 1 จะได้ว่ามีการจับคู่จาก U ไป V' นั่นคือ $|M|=|U|$ และ จะได้ M' เป็นการจับกึ่งคู่ และเมื่อ

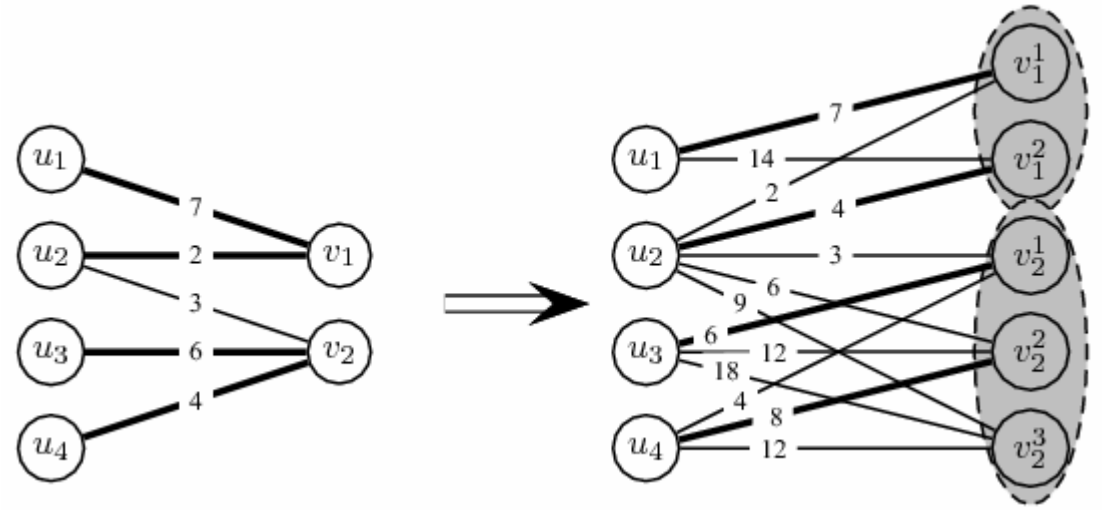
พิจารณาจุดยอด $v^1,v^2,...,v^{deg_{M'}(v)} \in V'$ ที่จับคู่กับจุดยอด $u_1,u_2,...,u_{degM(v)}$ เรียงตามลำดับค่า $w(u_i,v)$ จากมากไปน้อย รูปแบบที่ดีที่สุดในการจับคู่จะต้องไม่มี (u_i,w^j) และ $(u_{i'},w^{j'})$ ใน M' ที่ $i<i'$ แต่ $j'>j'$ เพราะเราสามารถแทนที่เส้นเชื่อมทั้งสองด้วย $(u_{i'},w^j)$ และ $(u_i,w^{j'})$ เพื่อสร้างการจับคู่ที่มีน้ำหนักต่ำกว่าได้ นั่นคือ v^i ที่มีหมายเลขน้อยจะต้องจับคู่กับ u_j ที่มีหมายเลขน้อย เพื่อให้ได้น้ำหนักรวมน้อยที่สุด ซึ่งจะได้น้ำหนักรวมเป็น

$$\begin{aligned} &w'(u_1,v_1)+w'(u_2,v_2)+...+w'(u_{degM(v)},v_{degM(v)}) \\ =&1\cdot w(u_1,v)+2\cdot w(u_2,v)+...+deg_{M'}(v)\cdot w(u_{degM(v)},v) \\ =&cost_{M'}(v) \end{aligned}$$

ซึ่งจะได้ว่าน้ำหนักรวมของ M' จะเท่ากับ $cost(M)$ นั่นคือการจับกึ่งคู่ที่ดีที่สุดใน G จะมีค่าใช้จ่ายไม่มากกว่าน้ำหนักของ M'

ในทางกลับกันเราจะสามารถสร้างการจับคู่ M'' จากการจับกึ่งคู่ M^* ใดๆ ที่น้ำหนักรวมของเส้นเชื่อมใน M'' เท่ากับค่าใช้จ่ายของ M^* เพื่อแสดงว่าการจับคู่สูงสุดที่มีน้ำหนักต่ำสุด จะมีน้ำหนักไม่มากกว่าค่าใช้จ่ายของการจับกึ่งคู่ M^* โดยเราจะสร้าง M'' ดังนี้ สำหรับเส้นเชื่อม $(u_1,v),(u_2,v),...,(u_{degM^*(v)},v)$ ใน M^* เรียงตามลำดับน้ำหนักจากมากไปน้อย สร้างเส้นเชื่อม $(u_1,v^1),(u_2,v^2),...,(u_{degM^*(v)},v^{deg_{M^*}(v)})$ ใน M'' จะได้การจับคู่ M'' ที่มีน้ำหนัก เท่ากับ $cost(M)$ และได้ว่าการจับคู่สูงสุดที่มีน้ำหนักต่ำสุด จะมีน้ำหนักไม่มากกว่าน้ำหนักการจับกึ่งคู่ที่ดีที่สุด นั่นคือจะได้ M เป็นการจับกึ่งคู่ที่ดีที่สุด

■



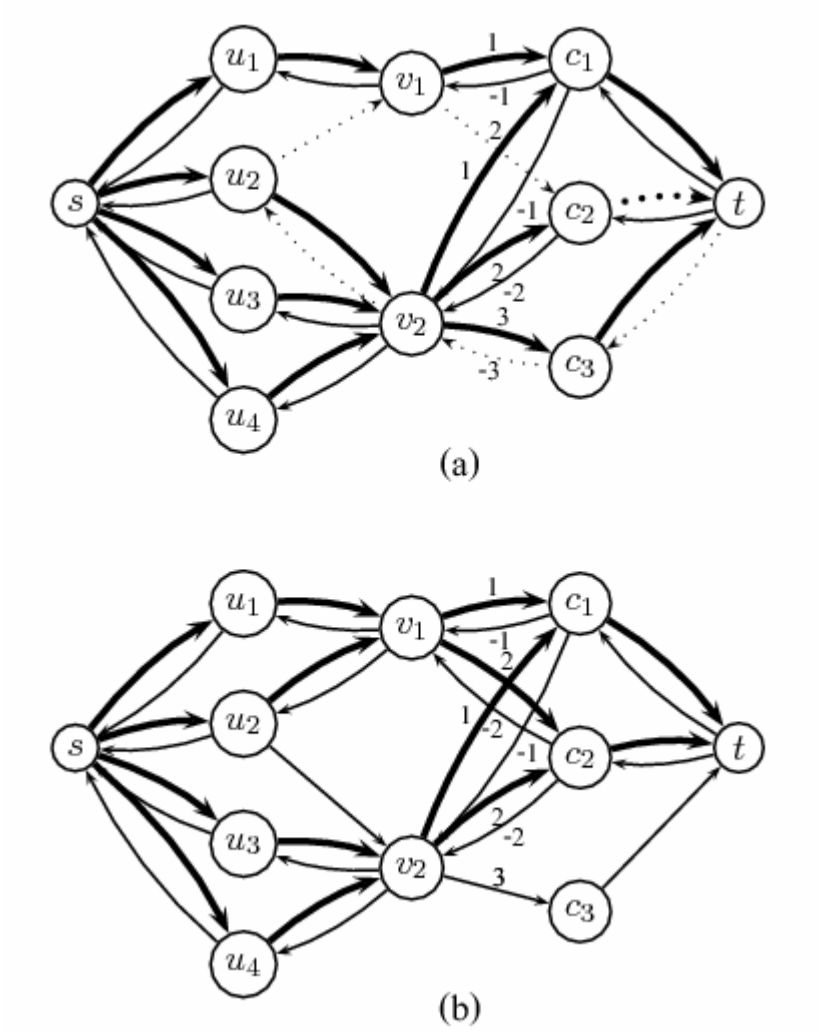
ภาพที่ 7 ตัวอย่างการลดทอนปัญหาการจับกึ่งคู่แบบมีน้ำหนักไปเป็นปัญหาการจับคู่เชิงน้ำหนัก

คุณสมบัติของปัญหา

การจับกึ่งคู่แบบไม่มีน้ำหนัก

จากการศึกษาคุณสมบัติของปัญหาการจับกึ่งคู่แบบไม่มีน้ำหนัก เราจะได้เงื่อนไขที่ระบุการเป็นค่าที่ดีที่สุดของการจับกึ่งคู่ ซึ่งสามารถอธิบายในรูปแบบของการไหลที่ถูกที่สุด โดยใช้วิธีการลดทอนปัญหาการจับกึ่งคู่ไปเป็นปัญหาการไหลที่ถูกที่สุดตามหัวข้อการลดทอนปัญหา เงื่อนไขจะอยู่ในรูปของวิถีบางประเภทในกราฟที่ได้จากการลดทอนปัญหา สำหรับกราฟ N ใดๆ และการไหล f ใน N ให้ $R_f(N)$ แทน *กราฟตกค้าง* ของ N ที่สัมพันธ์กับ f ซึ่งเราจะเขียน R_f แทน $R_f(N)$ เมื่อรู้ชัดเจนว่าหมายถึง N เราจะเรียกวิถีระหว่างศูนย์ค่าใช้จ่ายสองศูนย์ใน R_f ว่า *วิถีที่ยอมรับได้* (admissible path) และจะเรียกวิถีที่ยอมรับได้จาก c_i ไป c_j ว่าเป็นแบบ *ผันกลับ* (reverse) ถ้า $i>j$

สังเกตว่าถ้ากราฟมีวิถีที่ยอมรับได้แบบผันกลับ แสดงว่าในกราฟตกค้างมีวงรอบค่าใช้จ่ายลบ นั่นคือเราสามารถเปลี่ยนวิถีของการไหลให้มีค่าใช้จ่ายลดลงได้ โดยการเปลี่ยนการไหลไปตามวิถีที่ยอมรับได้แบบผันกลับ ดังตัวอย่างในภาพที่ 8 เงื่อนไขของการมีวิธีที่ยอมรับได้แบบผันกลับจะสมมูลกับเงื่อนไขที่ระบุว่า $R_f(N)$ ไม่มีวงรอบค่าใช้จ่ายลบ



ภาพที่ 8 ตัวอย่างวิถีที่ยอมรับได้แบบผันกลับ ภาพ (a) แสดงกราฟตกค้าง โดยเส้นทึมแสดงวิถีการไหล และเส้นประแสดงวิถีที่ยอมรับได้แบบผันกลับ มีค่าใช้จ่ายระบุอยู่บนเส้นเชื่อมที่มีค่าใช้จ่ายไม่เป็น 0 สังเกตว่าวิถีที่ยอมรับได้แบบผันกลับจะเป็นวงรอบค่าใช้จ่ายลบ ภาพ (b) แสดงกราฟตกค้างในภาพ (a) หลังจากการเปลี่ยนวิถีที่ยอมรับได้แบบผันกลับ ซึ่งได้การไหลที่มีค่าใช้จ่ายต่ำลง

ทฤษฎีบทย่อยที่ 9 *การไหล f จะเป็นการไหลถูกที่สุดใน N ก็ต่อเมื่อไม่มีวิถีที่ยอมรับได้แบบผันกลับใน $R_f(N)$*

การจับคู่แบบมีน้ำหนัก

ในหัวข้อนี้จะอธิบายถึงคุณสมบัติของการจับคู่แบบมีน้ำหนักบนกราฟ $G=(U\cup V,E)$ โดยพิจารณาเป็นการปัญหาการจับคู่สูงสุดที่มีน้ำหนักต่ำสุดที่สมมูลกันบนกราฟ $G'=(U\cup V',E')$ ตามวิธีการลดทอนในหัวข้อการลดทอนปัญหาและศึกษาการนำอัลกอริทึมของ Edmonds and Karp (1970) และ Tomizawa (1971) มาประยุกต์ใช้ในปัญหานี้ อัลกอริทึมดังกล่าวแก้ปัญหาโดยการเพิ่มขนาดของการจับคู่ครั้งละหนึ่ง ในการทำงานแต่ละรอบจะเพิ่มขนาดของการจับคู่โดยแต่งเติม (augment) การจับคู่ด้วยวิถีแต่งเติมที่มีค่าใช้จ่ายต่ำที่สุด/สั้นที่สุด (cheapest/shortest augmenting path) การจับคู่ที่ได้จากการทำงานในแต่ละรอบนี้จะเป็นการจับคู่ที่มีน้ำหนักน้อยที่สุดในการจับคู่ที่มีขนาดเท่ากัน เรียกการจับคู่ที่มีลักษณะเช่นนี้ว่า*สุดขีด* (extreme) อัลกอริทึมดังกล่าวแสดงในภาพที่ 9 โดยอัลกอริทึมรับกราฟสองส่วน $G=(U\cup V,E)$ และฟังก์ชันน้ำหนัก $w:E\rightarrow R$ และให้ผลลัพธ์เป็นการจับคู่สูงสุดที่มีน้ำหนักต่ำสุด M เพื่ออธิบายอัลกอริทึมดังกล่าวเราจะให้นิยามดังนี้

กราฟตกค้างที่สัมพันธ์กับการจับคู่ M เขียนแทนด้วย $R_M(G)=(U\cup V,A)$ เป็นกราฟระบุทิศทางโดยแต่ละเส้นเชื่อม $(u,v)\in(E-M)$ จะมีเส้นเชื่อมระบุทิศทาง $(u,v)\in A$ มีน้ำหนักเป็น $w(u,v)$ และสำหรับทุกเส้นเชื่อม $(u,v)\in M$ จะมีเส้นเชื่อมระบุทิศทาง $(v,u)\in A$ มีน้ำหนักเป็น $-w(u,v)$ แทนเซตของจุดยอดใน U ที่ไม่ประชิดกับเส้นเชื่อมใน M ด้วย U_M และแทนเซตของจุดยอดใน V ที่ไม่ประชิดกับเส้นเชื่อมใน M ด้วย V_M นิยาม*วิถีแต่งเติมที่สัมพันธ์กับการจับคู่ M* เป็นวิถีจาก U_M ไป V_M ใน $R_M(G)$ *วิถีแต่งเติมที่สั้นที่สุด* หรือ *วิถีแต่งเติมถูกที่สุด* คือ วิถีแต่งเติมที่มีน้ำหนักรวมของเส้นเชื่อมใน $R_M(G)$ ต่ำที่สุด *วงรอบค่าใช้จ่ายลบ* คือวงรอบใน $R_M(G)$ ที่มีผลรวมของน้ำหนักในเส้นเชื่อมเป็นค่าลบ

ถ้า M เป็นการจับคู่สุดขีด จะไม่มีวงรอบค่าใช้จ่ายลบใน $R_M(G)$ ซึ่งสามารถหาวิถีแต่งเติมที่สั้นที่สุดได้ด้วยอัลกอริทึมของ Bellman-Ford ที่ทำงานในเวลา $O(|U\cup V||E|)$ ได้ อย่างไรก็ตามเราสามารถใช้อัลกอริทึมของ Dijkstra ที่ทำงานในเวลา $O(|E|+|U\cup V|\log|U\cup V|)$ ได้ โดยใช้*ฟังก์ชันราคา* (price function) $p:V\rightarrow R$ และคิดความยาวเส้นเชื่อมเป็น*ค่าใช้จ่ายลดทอน* (reduced cost) $w'_p(u,v)=p(u)-p(v)+w(u,v)$ เรียกฟังก์ชันที่ให้ค่าใช้จ่ายลดทอนไม่เป็นค่าลบว่า *ฟังก์ชันราคาที่เป็นไปได้* (feasible price function) ในกรณีนี้*ป้ายระยะทาง* (distance label) ที่ได้จากการหาวิถีแต่งเติมที่สั้นที่สุดในแต่ละรอบซึ่งเป็นระยะทางที่สั้นที่สุดจาก U_M ใน $R_M(G)$ เป็นฟังก์ชันราคาที่เป็นไปได้บน $R_M(G)$ อัลกอริทึมของ Edmonds and Karp ใช้ป้ายระยะทางที่ได้ใน

แต่ละรอบเป็นฟังก์ชันราคาในรอบถัดไป รายละเอียดของอัลกอริทึมดังกล่าวแสดงในอัลกอริทึม Min-Matching ในภาพที่ 9

Algorithm Min-Matching$(G = (U \cup V, E), w)$

1. $M \leftarrow \emptyset$
2. $\forall v \in U \cup V, p(v) \leftarrow 0$
3. while there is an augmenting path
4. Find SAP P, and distance label d using reduced cost w_p
5. Augment M along P
6. $p \leftarrow d$
7. end while
8. return M

ภาพที่ 9 อัลกอริทึมสำหรับหาการจับคู่เชิงน้ำหนัก เขียน SAP เป็นตัวย่อแทนคำว่า Shortest Augmenting Path

ในภาพที่ 9 แสดงอัลกอริทึมของ Edmonds and Karp เริ่มต้นให้การ M เท่ากับเซตว่าง จากนั้นจะเพิ่มขนาดของ M โดยการหาวิถีที่สั้นที่สุดจาก U_M ไปยัง V_M ใน $R_M(G)$ ซึ่งเรียกว่าวิถีแต่งเติมที่สั้นที่สุด (ในภาพเขียน SAP แทน Shortest Augmenting Path) เขียนแทนวิถีดังกล่าวด้วย P ซึ่งจะใช้ป้ายระยะทาง d ที่ได้จากการหา P มาใช้เป็นฟังก์ชันราคาในรอบถัดไป เมื่อหา P แล้วแต่งเติม M ด้วย P

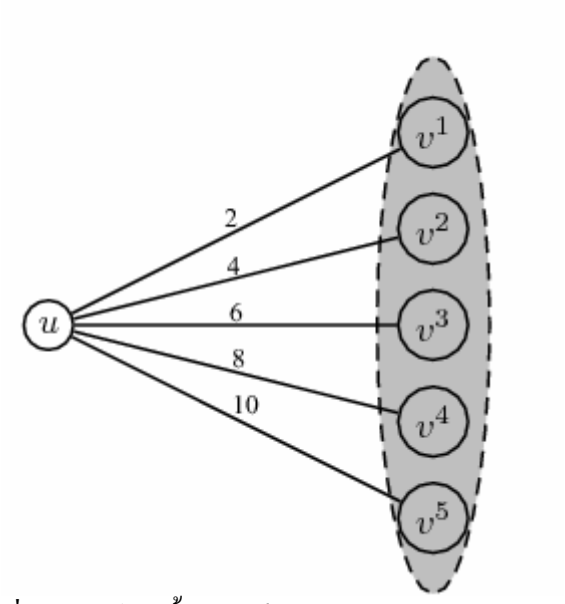
ในงานวิจัยนี้ได้ศึกษาคุณสมบัติของค่าต่างๆ ที่เกี่ยวข้องในการแก้ปัญหาการจับคู่เชิงน้ำหนักบนกราฟ $G'=(U\cup V',E')$ และฟังก์ชันน้ำหนัก $w':U\cup V'\rightarrow R$ ซึ่งมาจากการลดทอนปัญหาจากการจับกึ่งคู่จากกราฟตั้งต้น G ตามวิธีการในหัวข้อการลดทอนปัญหา สังเกตว่าสำหรับเส้นเชื่อม $(u,v)\in E$ จะมีเส้นเชื่อมที่สมนัยกันใน E' ซึ่งประชิดกับ $S(v)$ เมื่อพิจารณาค่าของน้ำหนักเทียบกับหมายเลขจุดยอดใน $S(v)$ จะมีค่า $w'(u,v^i)=i\cdot w(u,v)$ นั่นค่าของน้ำหนักจะเป็นฟังก์ชันเส้นตรงเทียบกับ i ดังแสดงในภาพที่ 10 แต่เมื่อนำมาพิจารณาในค่าใช้จ่ายลดทอนค่าดังกล่าวไม่

เป็นเส้นตรง ดังนั้นเราจึงศึกษาคุณสมบัติอื่นเพิ่มเติม เพื่อพัฒนาให้การแก้ปัญหาในกรณีเฉพาะนี้การทำงานเร็วขึ้น คุณสมบัติสำคัญที่ได้จากการศึกษามีดังต่อไปนี้

1 คุณสมบัติของการจับคู่สุดขีด

2 คุณสมบัติของค่าใช้จ่ายลดทอน

3 คุณสมบัติในการเปรียบเทียบระยะทาง



ภาพที่ 10 ฟังก์ชันน้ำหนักเป็นเส้นตรงบนหมายเลขจุดยอด

คุณสมบัติของการจับคู่สุดขีด

ในแต่ละรอบของการทำงานจะได้การจับคู่สุดขีด M พิจารณาจุดยอด $v \in V$ ใดๆ และการจับคู่ที่ประชิดกับจุดยอดใน $S(v)$ หมายเลขของจุดยอดที่ถูกจับคู่ใน $S(v)$ จะเรียงลำดับต่อเนื่องกันเป็น $v^1, v^2, \ldots, v^{deg_M(v)}$ ไม่เช่นนั้นเราจะสามารถสร้างการจับคู่ที่มีน้ำหนักรวมต่ำกว่าได้ คุณสมบัติดังกล่าวแสดงในทฤษฎีบทย่อยดังนี้

ทฤษฎีบทย่อยที่ 10 *ให้การจับคู่สุดขีด M บน G' และ $S(v)=\{v^1, v^2, \ldots, v^{deg(v)}\} \subseteq V'$ เป็นเซตของจุดยอดที่สมนัยกับ v ใน V ถ้า $v^i \in S(v)$ เป็นจุดยอดที่ไม่ถูกจับคู่ แล้ว $v^{i+1} \in S(v)$ จะเป็นจุดยอดที่ไม่ถูกจับคู่ด้วย*

พิสูจน์

พิสูจน์โดยใช้ความขัดแย้ง สมมติว่า $v^i \in S(v)$ เป็นจุดยอดที่ไม่ถูกจับคู่ แต่ $v^{i+1} \in S(v)$ เป็นจุดยอดที่จับคู่แล้ว ให้ $u \in U'$ เป็นจุดยอดที่จับคู่กับ v^{i+1} นั่นคือมี $(u,v^{i+1}) \in M$ สร้างการจับคู่ $M'=M-(u,v^{i+1})+(u,v^i)$ จะได้ M' เป็นการจับคู่ที่มีน้ำหนักรวมต่ำกว่า M ซึ่งขัดแย้งกับความเป็นค่าต่ำสุดของ M

■

ทฤษฎีบทตามที่ 11 *ถ้าหาการจับคู่สูงสุดที่มีน้ำหนักรวมต่ำที่สุดโดยใช้วิธีขยายขนาดการจับคู่สุดขีด M ครั้งละหนึ่ง แล้ว ใน $S(v)$ ใดๆ จุดยอด $v^i \in S(v)$ ใดๆ จะถูกจับคู่ในรอบการทำงานก่อน $v^{i+1} \in S(v)$ ถูกจับคู่เสมอ*

พิสูจน์

พิสูจน์โดยใช้ความขัดแย้ง สมมุติว่ามี $v^i, v^{i+1} \in S(v)$ ที่ v^{i+1} ถูกจับคู่ในรอบการทำงานก่อนที่ v^i จะถูกจับคู่ แสดงว่า มีรอบการทำงานที่ v^{i+1} ถูกจับคู่ แต่ v^i ไม่ถูกจับคู่ซึ่งขัดแย้งกับทฤษฎีบทย่อย 10

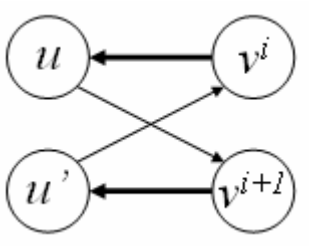
■

คุณสมบัติดังกล่าวสามารถนำมาใช้ลดจำนวนจุดยอดที่จะพิจารณาในแต่ละรอบการทำงานได้ โดยในแต่ละเซต $S(v)$ จะพิจารณาจุดยอดที่ไม่ถูกจับคู่และมีหมายเลขต่ำที่สุดเพียงจุดเดียว เนื่องจากจุดยอดที่ถูกจับคู่จะมีขนาดไม่เกิน $2|M|<2|U|$ ดังนั้นจำนวนเส้นเชื่อมที่จะใช้พิจารณาในแต่ละรอบจะมีจำนวนไม่เกิน $|M|^2+|E|=O(|U|^2)$

คุณสมบัติของค่าใช้จ่ายลดทอน

พิจารณาที่ $S(v)$ ใดๆ และจุดยอด $u \in U$ ที่มีเส้นเชื่อมประชิดกับ $S(v)$ สังเกตได้ว่าน้ำหนักของเส้นเชื่อม $(u,v^1),(u,v^2),...,(u,v^{deg_G(v)})$ เป็นฟังก์ชันเส้นตรงบนลำดับของหมายเลขจุดยอด แต่เมื่อพิจารณาน้ำหนักเป็นค่าของฟังก์ชันของค่าใช้จ่ายลดทอนแล้ว การเปลี่ยนแปลงของน้ำหนักไม่เป็นฟังก์ชันเส้นตรง อย่างไรก็ตามเราได้ศึกษาลักษณะการเปลี่ยนแปลงของค่าใช้จ่ายลดทอน โดยศึกษาภาพแบบการเปลี่ยนแปลงของฟังก์ชันราคาของจุดยอดใน $S(v)$ ซึ่งเราจะเรียกระยะห่างของฟังก์ชันราคาระหว่างจุดยอดสองจุดที่มีหมายเลขเรียงกันใน $S(v)$ ว่า*ระยะห่างของฟังก์ชันราคา* (gap of price function) ดัง

แสดงไว้ในทฤษฎีบทย่อยต่อไปนี้



ภาพที่ 11 ภาพประกอบทฤษฎีบทย่อยที่ 12

ทฤษฎีบทย่อยที่ 12 *ถ้า M เป็นการจับคู่สุดขีดบน G' และ p เป็นฟังก์ชันราคาที่เป็นไปได้ สำหรับทุกจุดยอดที่จับคู่แล้ว ถ้าจุดยอด* v^i *จับคู่กับ u และ* v^{i+1} *จับคู่กับ u' แล้ว จะได้* $w(u,v) \geq w(u',v)$ *และ* $w(u',v) \leq p(v^{i+1}) - p(v^i) \leq w(u,v)$

พิสูจน์

เงื่อนไข $w(u,v) \geq w(u',v)$ ได้จากความเป็นการจับคู่สุดขีดของ M เพราะถ้าไม่เป็นตามเงื่อนไขนี้เราสามารถลดน้ำหนักของการจับคู่ได้ โดยให้ v^i จับคู่กับ u' และ v^{i+1} จับคู่กับ u แทน ซึ่งจะได้การจับคู่ที่มีน้ำหนักต่ำลงแต่มีขนาดเท่าเดิม

ดูภาพประกอบในภาพที่ 11 พิจารณาที่เส้นเชื่อม (v^i,u) และ (u,v^{i+1}) เนื่องจาก p เป็นฟังก์ชันราคาที่เป็นไปได้ เพราะฉะนั้น $w'_p(v^i,u)$ และ $w'_p(u,v^{i+1})$ จะไม่เป็นค่าลบ และจะได้ระยะทางจาก v^i ไป v^{i+1} ผ่าน (v^i, u) และ (u, v^{i+1}) มีระยะทางไม่เป็นค่าลบ

$$w'_p(v^i, u) + w'_p(u, v^{i+1}) \geq 0$$

จากนิยามของค่าใช้จ่ายลดทอนจะได้

$$\begin{aligned} w'_p(v^i, u) + w'_p(u, v^{i+1}) &= p(v^i) - w'(u, v^i) - p(u) + p(u) + w'(u, v^{i+1}) - p(v^{i+1}) \\ &= p(v^i) - p(v^{i+1}) + w'(u, v^{i+1}) - w'(u, v^i) \\ &= p(v^i) - p(v^{i+1}) + w(u, v) \end{aligned}$$

นั่นคือ

$$\begin{aligned} p(v^i) - p(v^{i+1}) + w(u, v) &\geq 0 \\ p(v^{i+1}) - p(v^i) &\leq w(u, v) \end{aligned}$$

สังเกตว่าถ้าระยะห่างของ $p(v^i)$ และ $p(v^{i+1})$ มากกว่า $w(u, v)$ จะมีเส้นเชื่อมที่มีค่าใช้จ่ายลดทอนเป็นลบเกิดขึ้น

ใช้เหตุผลเช่นเดียวกันที่เส้นเชื่อม (v^{i+1}, u') และ (u', v^i) จะได้ ระยะทางจาก v^{i+1} ไป v^i ผ่าน (v^{i+1}, u') และ (u', v^i) มีระยะทางเป็น $p(v^{i+1}) - p(v^i) - w(u', v)$ และมีค่าไม่เป็นลบนั่นคือจะได้

$$p(v^{i+1}) - p(v^i) - w(u', v) \geq 0$$
$$w(u', v) \leq p(v^{i+1}) - p(v^i)$$

ซึ่งจะได้ทฤษฎีบทดังกล่าว

■

ทฤษฎีบทตามที่ 13 *สำหรับทุก* $v \in V$, *และ* $1 < i < deg_M(v)$ จะมีค่า $p(v^i)-p(v^{i-1}) \geq p(v^{i+1})-p(v^i)$

พิสูจน์

พิจารณาจุดยอด $u_1,u_2,u_3 \in U'$ ที่ $w(u_1,v) \geq w(u_2,v) \geq w(u_3,v)$ และจับคู่กับ v^{i-1}, v^i และ v^{i+1} ตามลำดับ จากทฤษฎีบทย่อยที่ 12 จะได้

$$w(u_1,v) \geq p(v^i)-p(v^{i-1}) \geq w(u_2,v)$$
$$w(u_2,v) \geq p(v^{i+1})-p(v^i) \geq w(u_3,v)$$

ซึ่งจะได้ $p(v^i) - p(v^{i-1}) \geq p(v^{i+1}) - p(v^i)$ ตามทฤษฎีบทตาม

■

จากทฤษฎีบทย่อยที่ 12 พิจารณาที่ $S(v)$ จะเห็นว่าระยะห่างของฟังก์ชันราคาของจุดยอดใน $S(v)$ ตามลำดับหมายเลขจุดยอดจะมีค่าไม่เพิ่มและจากนิยามของค่าใช้จ่ายลดทอน พิจารณาที่จุดยอด $u \in U$ ใดๆ ที่ประชิดกับ $S(v)$ ค่าของค่าใช้จ่ายลดทอน $w'_p(u,v^i)=p(u)-p(v^i)+i \cdot w(u,v)$ เทียบกับหมายเลขจุดยอดใน $S(v)$ เมื่อระยะห่างของค่าใช้จ่ายที่ $S(v)$ ไม่มากกว่า $w(u,v)$ จะมีค่าไม่เพิ่ม และเมื่อระยะห่างมากกว่า $w(u,v)$ ค่าของฟังก์ชันจะไม่ลดลงอีก ดังแสดงในทฤษฎีบทย่อยต่อไปนี้

ทฤษฎีบทย่อยที่ 14 *สำหรับเส้นเชื่อม* $(u,v) \in E$ *ใดๆ จะมีค่าจำนวนเต็ม* t *ที่* $1 \le t < deg_M(v)$, $w'_p(u,v^i) \ge w'_p(u,v^{i+1})$ *สำหรับทุก* $1 \le i \le t$ *และ* $w'_p(u,v^i) \le w'_p(u,v^{i+1})$ *สำหรับทุก* $t \le i \le deg_M(v)$ *หรือ* $w'_p(u,v^i) \le w'_p(u,v^{i+1})$, *สำหรับทุก* $1 \le i < deg_M(v)$

พิสูจน์

สมมติว่ามีจำนวนเต็ม t ที่ $1 \le t < deg_M(v)$ และเป็นค่าน้อยที่สุดที่ $w'_p(u,v^t) < w'_p(u,v^{t+1})$ เนื่องจาก $w'_p(u,v^t) < w'_p(u,v^{t+1})$ จะได้

$$w'_p(u,v^{t+1}) - w'_p(u,v^t) = p(v^{t+1}) - p(v^t) + w(u,v) > 0$$
$$p(v^{t+1}) - p(v^t) < w(u,v)$$

จากทฤษฎีบทตามที่ 13 สำหรับทุก i ที่ $t \le i < deg_M(v)$ จะได้

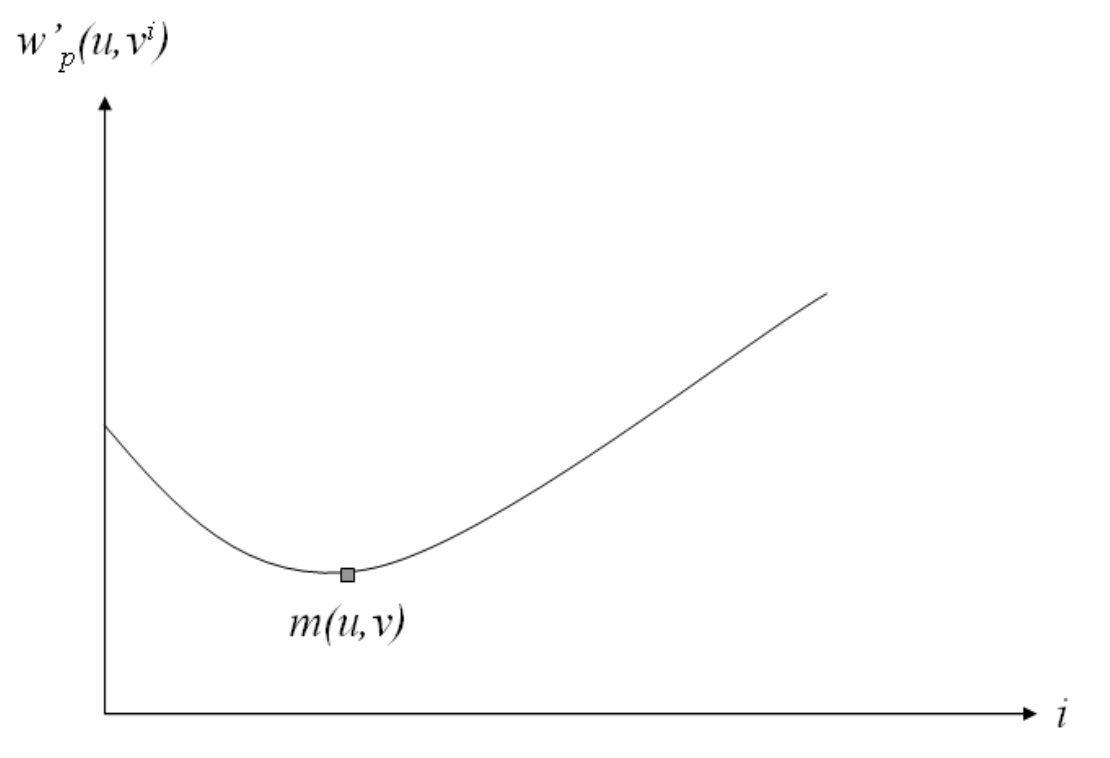
$$p(v^{t+1}) - p(v^t) \ge p(v^{i+1}) - p(v^i)$$
$$w(u,v) \ge p(v^{i+1}) - p(v^i)$$

เพียงพอที่จะสรุปได้ว่า $w'_p(u,v^i) < w'_p(u,v^{i+1})$

ในกรณีที่ไม่มี t ที่ $1 \le t < deg_M(v)$ และ $w'_p(u,v^t) < w'_p(u,v^{t+1})$ จะได้ $w'_p(u,v^i) \le w'_p(u,v^{i+1})$, สำหรับทุก $1 \le i < deg_M(v)$ ตามทฤษฎีบทย่อย

■

จากคุณสมบัติดังกล่าวทำให้เราทราบลักษณะการเปลี่ยนแปลงของค่า $w'_p(u,v^i)$ เทียบกับ i ว่ามีช่วงแรกเป็นฟังก์ชันไม่เพิ่ม และช่วงหลังเป็นฟังก์ชันไม่ลด ซึ่งลักษณะดังกล่าวจะนำไปสู่วิธีในการแก้ปัญหา กราฟแสดงการเปลี่ยนแปลงของค่า $w'_p(u,v^i)$ แสดงในภาพที่ 12



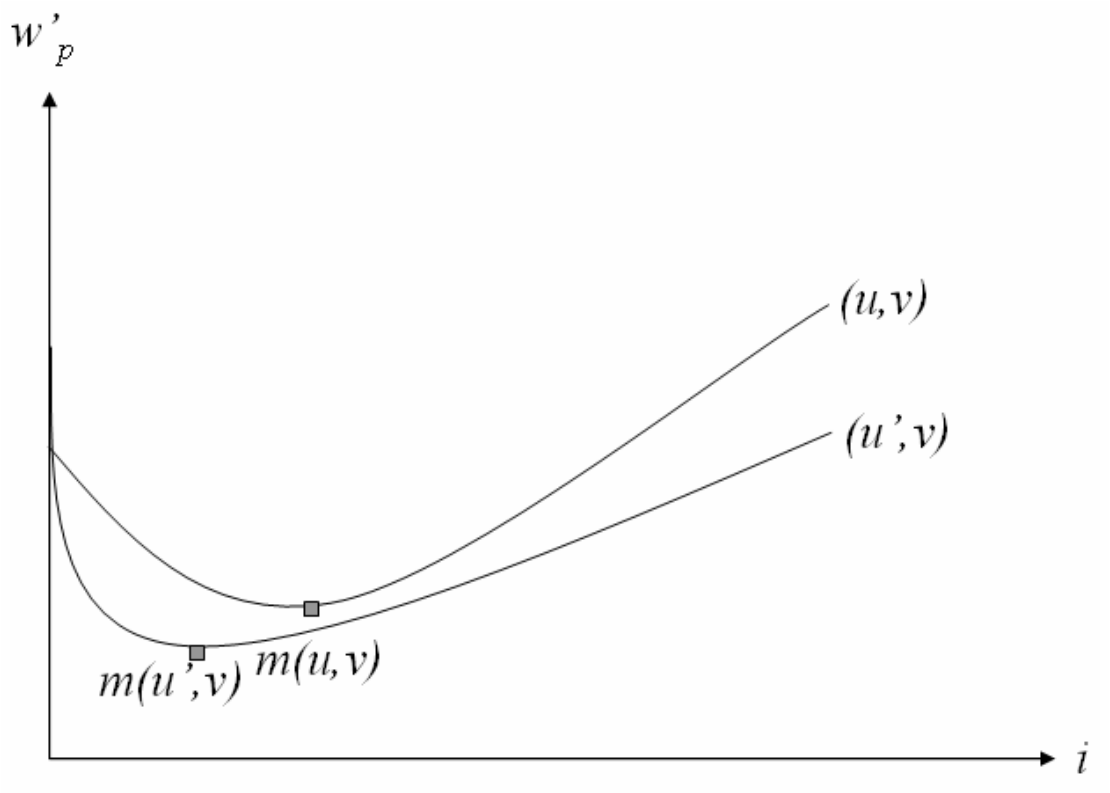
ภาพที่ 12 การเปลี่ยนแปลงของ $w'_p(u,v^i)$ เทียบกับ i จุดสี่เหลี่ยมแสดงตำแหน่งของ $m(u,v)$

เรานิยามฟังก์ชัน $m(u,v)=t$ เมื่อ t เป็นจำนวนเต็มที่ $1 \le t < deg_M(v)$ และค่าน้อยที่สุดที่ทำให้ $w'_p(u,v^t) < w'_p(u,v^{t+1})$ ถ้าไม่มีค่า t ดังกล่าว ให้ $m(u,v)=deg_M(v)$ จากนิยามดังกล่าวของ $m(u,v)$ จะเป็นจุดที่มีค่าใช้จ่ายลดทอนต่ำที่สุดของแต่ละเส้นเชื่อม $(u,v) \in E$ และ เราจะได้ว่า $t=m(u,v)$ เป็นค่าแรกที่ $p(v^{t+1})-p(v^t) > w(u,v)$ ซึ่งค่า $m(u,v)$ มีความสัมพันธ์กับน้ำหนักของเส้นเชื่อม $w(u,v)$ เมื่อเรานำน้ำหนัก ดังทฤษฎีบทย่อยต่อไป

ทฤษฎีบทย่อยที่ 15 *สำหรับเส้นเชื่อมใดๆ $(u,v),(u',v) \in E$, ที่ $w(u,v) > w(u',v)$ จะมีค่า $m(u,v) \le m(u',v)$*

พิสูจน์

ถ้า $m(u',v) = deg_M(v)$ จะได้ $m(u,v) \le m(u',v)$ ตามทฤษฎีบทย่อย เราจะพิสูจน์ในกรณีที่เหลือคือ $m(u',v) < deg_M(v)$ ซึ่งจะพิสูจน์โดยใช้ความขัดแย้ง สมมติให้ $m(u,v) > m(u',v)$ และ ให้ $t = m(u',v)$ จากนิยามของ $m(u',v)$ จะได้ $p(v^{t+1}) - p(v^t) > w(u',v) > w(u,v)$ ซึ่งขัดแย้งกับนิยามของ $m(u,v)$ ■



ภาพที่ 13 แสดงการเปรียบเทียบ (u,v) และ $(u',v) \in E$ ที่ $w(u,v)>w(u',v)$

สำหรับเส้นเชื่อม $(u,v) \in E$ เราสามารถใช้เส้นเชื่อมที่สมนัยกันใน E' ที่มีค่าใช้จ่ายลดทอนต่ำที่สุด เป็นตัวแทนของเส้นเชื่อม โดยค่า m จะช่วยในการบอกจุดต่ำสุดที่เราจะใช้ในการเริ่มพิจารณาสำหรับคุณสมบัติตามทฤษฎีบทย่อย 15 มีความสำคัญในการใช้หาค่า m ของทุกเส้นเชื่อมก่อนการทำงานในแต่ละรอบได้ในเวลา $O(|E|)$

คุณสมบัติในการเปรียบเทียบระยะทาง

พิจารณาเซต $S(v)$ ใดๆ ในการหาวิถีที่สั้นที่สุดจาก U_M ไปยัง $S(v)$ ใน $R_M(G')$ จำเป็นต้องเปรียบเทียบระยะทางที่มาจากทุกจุดยอดใน U ที่ประชิดกับ $S(v)$ พิจารณาการเปรียบเทียบระยะทางจากสองจุดยอด $u,u' \in U$ มายังจุดยอดใดๆ $v_i \in S(v)$

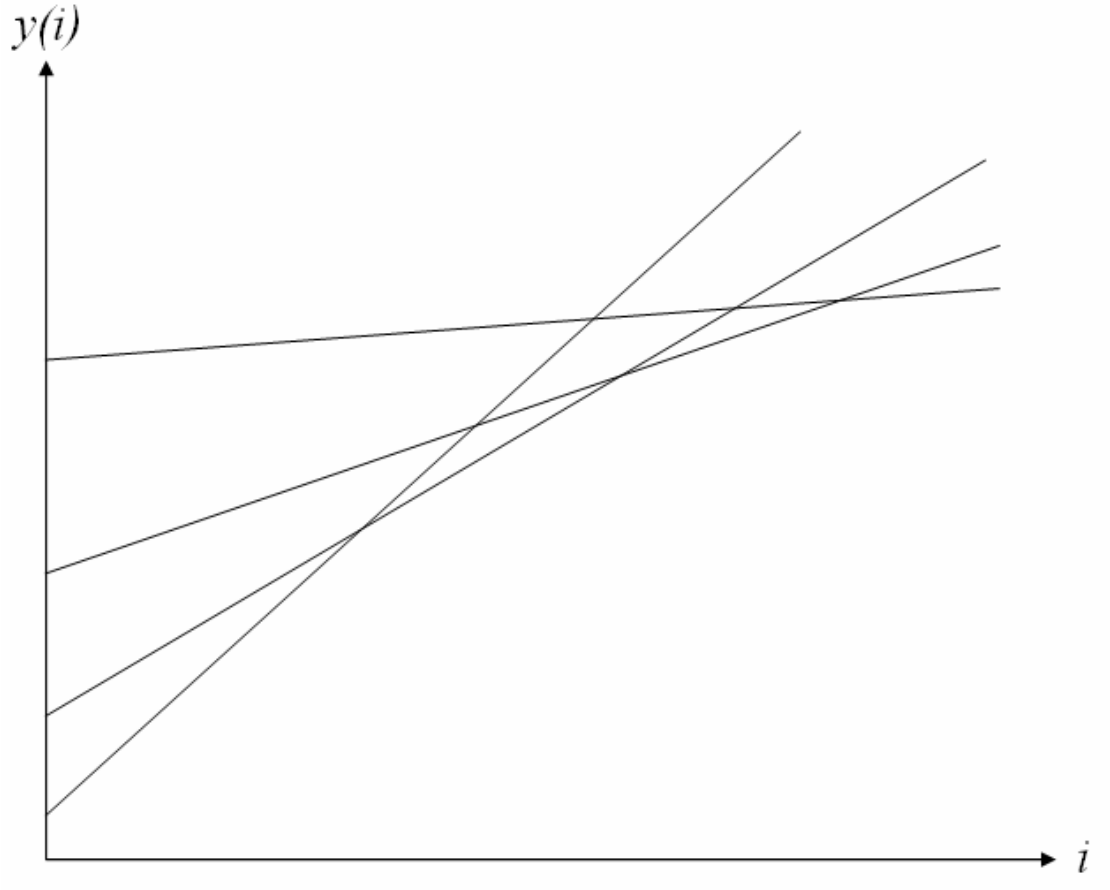
$$d(u)+w'_p(u,v^i) = d(u)+p(u)-p(v^i)+i \cdot w(u,v) \quad (1)$$

$$d(u')+w'_p(u,v^i) = d(u')+p(u')-p(v^i)+i \cdot w(u',v) \quad (2)$$

สมการ (1) เป็นระยะทางจากจุดยอด u และ สมการ (2) เป็นระยะทางจากจุดยอด u' เห็นได้ว่าสมการทั้งสองมีค่าที่ซ้ำกันคือ $p(v^i)$ ดังนั้นในการเปรียบเทียบระยะทางจากจุดยอดสองจุดนี้ไปยังจุดยอดใดๆ ใน $S(v)$ เราสามารถลบค่า $p(v^i)$ ออกไปได้ เขียนแทนค่าดังกล่าวด้วย $y_u(i)$ และ $y_{u'}(i)$ ดังนี้

$$y_u(i) = d(u)+p(u)+i\cdot w(u,v)$$
$$y_{u'}(i) = d(u')+p(u')+i\cdot w(u',v)$$

สังเกตว่า $y_u(i)$ และ $y_{u'}(i)$ เป็นฟังก์ชันเส้นตรง ดังนั้นทั้งสองเส้นนี้จะตัดกันได้เพียงจุดเดียว และในการเปรียบเทียบทุกจุดยอดที่ประชิดกับ $S(v)$ เราสามารถมองเป็นการเปรียบเทียบเส้นตรง y แทนได้ โดยจุดยอด v^i ใดๆ จะมีค่าระยะทางจากสั้นที่สุดจากเส้นที่เป็นขอบล่างของการตัดกันของเส้นตรงเท่านั้น นอกจากนี้เส้นขอบล่างของการตัดกันของเส้นตรง y จะมีรูปร่างเป็นคอนเว็กซ์ (convex) คือ ถ้าเขียน $L_v(i)$ แทนฟังก์ชันของเส้นขอบล่างดังกล่าว จะไม่มีช่วง (a,b) ใดๆ ที่ $L(\ a+b/2\ ) < (L(a)+L(b))/2$ (ตามปกติจะเรียกฟังก์ชันชนิดนี้ว่าคอนเคฟ (concave) แต่เพื่อความสะดวกในการอธิบายเพราะจะอ้างอิงอัลกอริทึมในการจัดการคอนเว็กซ์ฮัล (convex hull)) ซึ่งรูปร่างดังกล่าวง่ายต่อการจัดการ คือ ถ้าเราแทนเส้นเชื่อม (u,v) ด้วยเส้นตรง y_u เราสามารถเพิ่มเส้นตรงเข้าไปในเส้นขอบล่าง L_v ได้ในเวลา $O(\log|U|)$ รูปตัวอย่างของการเปรียบเทียบระยะทางแสดงในรูปที่ 14



ภาพที่ 14 แสดงการเปรียบเทียบระยะทางโดยเขียนเป็นกราฟเทียบค่า $y(i)$ กับ i

ผลการวิจัย

จากการศึกษาปัญหาการหาการจับกึ่งคู่ที่ดีที่สุดในกราฟสองส่วนเราได้แบ่งปัญหาเป็น 2 กรณี คือในกรณีที่กราฟเป็นกราฟไม่มีน้ำหนัก (unweighted graph) และ กรณีที่กราฟมีน้ำหนัก (weighted graph) เรามองปัญหาทั้งสองกรณีเป็นปัญหาการมอบหมายงานให้กับเครื่องจักร และนิยามจุดประสงค์ของปัญหาในลักษณะของการปรับดุลย์ภาระงาน (load-balance) ซึ่งในกรณีที่กราฟไม่มีน้ำหนักเราใช้จุดประสงค์ของปัญหาตามงานวิจัยของ Harvey et al. (2003) ส่วนในกรณีที่กราฟมีน้ำหนักเราได้นิยามจุดประสงค์ของปัญหาในลักษณะเดียวกัน คือมองว่าเมื่อมีเครื่องจักรรับงานมากกว่าหนึ่งงาน ต้องมีการรอให้งานแรกเสร็จก่อนจึงจะทำงานของตัวเองได้ และคิดฟังก์ชันจุดประสงค์เป็นผลรวมของเวลารอของแต่ละงาน เราได้ศึกษาและได้วิธีการในการแก้ปัญหาตามที่อธิบายในส่วนต่างๆ ดังนี้ ส่วนแรกอธิบายผลการศึกษาในกรณีที่กราฟไม่มีน้ำหนัก และส่วนที่สองอธิบายผลการศึกษาในกรณีที่กราฟมีน้ำหนัก

ผลการศึกษาปัญหาในกรณีที่กราฟไม่มีน้ำหนัก

ในกรณีที่กราฟไม่มีน้ำหนัก หรือมีน้ำหนักเป็นปริมาณเท่ากันหมด เราใช้นิยามของปัญหาตามผลงานวิจัยของ Harvey et al. (2003) และใช้วิธีการแก้ปัญหาโดยการแปลงปัญหาเป็นปัญหาการไหลถูกที่สุดในเครือข่าย (min-cost flow problems) และแก้โดยอัลกอริทึมแบบแบ่งแยกและจัดการ (divide and conquer algorithm) เราพิจารณาปัญหานี้เป็นกรณีเฉพาะของปัญหาการไหลที่ถูกที่สุด (รายละเอียดอธิบายในหัวข้อปัญหาการไหลสูงสุดที่ค่าใช้จ่ายต่ำที่สุด) โดยมองการจับกึ่งคู่เป็นการไหล f ได้ว่าเงื่อนไขการดีที่สุดของ Harvey et al. (2003) สมมูลกับเงื่อนไขการไม่มีวงรอบค่าใช้จ่ายลบ (negative-cost cycle) ในกราฟตกค้าง (residual graph) ของ f ซึ่งอัลกอริทึมที่สองของ Harvey et al. (2003) ใช้หลักการที่คล้ายกันนี้ในการกำจัดวงรอบเหล่านี้ใช้การจับกึ่งคู่ อัลกอริทึมของเราใช้ประโยชน์จากโครงสร้างและค่าใช้จ่ายของกราฟ เพื่อกำจัดวงรอบค่าใช้จ่ายลบในแต่รอบของการทำงาน โดยดัดแปลงอัลกอริทึมสำหรับแก้ปัญหาการไหลในกราฟหน่วย (unit network) ซึ่งทำงานในเวลา $O(|E|\sqrt{(|U|)})$ พัฒนามาเป็นอัลกอริทึมที่ทำงานในเวลา $O(|E|\sqrt{(|U|)}\log|U|)$ ซึ่งเมื่อเปรียบเทียบกับการจับคู่สองส่วนที่มากที่สุดจะมีเพียงตัวประกอบเชิงลอการิทึมเพิ่มขึ้นมาเท่านั้น (ดู Feder and Motwani (1995); Karzanov (1973); Hopcroft and Karp (1973); Goldberg and Kennedy (1995)) นอกจากนี้อัลกอริทึมของเรายังสามารถแก้ปัญหาในกรณีทั่วไปได้ ซึ่งเราจะกล่าวถึงเรื่องนี้ในหัวข้อกรณีทั่วไป

อัลกอริทึมแบบแบ่งแยกและจัดการ

อัลกอริทึมในการแก้ปัญหามาจากการลดทอนปัญหาในหัวข้อการลดทอนปัญหา การทำงานมีพื้นฐานมาจากการกำจัด*วิถีที่ยอมรับได้แบบผันกลับ* (reverse admissible path) คือ วิถีจากศูนย์ค่าใช้จ่าย c_i ไป c_j ที่ $i>j$ ในกราฟตกค้าง สำหรับกราฟ N ใดๆ และการไหล f ใน N ให้ $R_f(N)$ แทน *กราฟตกค้าง* ของ N ที่สัมพันธ์กับ f ซึ่งเราจะเขียน R_f แทน $R_f(N)$ เมื่อรู้ชัดเจนว่าหมายถึง N เราจะเรียกวิถีระหว่างศูนย์ค่าใช้จ่ายสองศูนย์ใน R_f ว่า *วิถีที่ยอมรับได้* (admissible path) และจะเรียกวิถีที่ยอมรับได้จาก c_i ไป c_j ว่าเป็นแบบ *ผันกลับ* (reverse) ถ้า $i>j$ ให้การไหลสูงสุด f และวิถีที่ยอมรับได้แบบผันกลับ P เราสามารถหาการไหล f ที่มีค่าใช้จ่ายต่ำกว่าได้โดยการเพิ่มการไหล f ผ่านทางวิถี P หลังจากขั้นตอนนี้ เราจะพบว่าวิถี P จะหายไปจากกราฟตกค้าง ขั้นตอนนี้ต่อไปเราจะเรียกว่า *การกำจัดวิถี* (path canceling) เราสามารถอธิบายอัลกอริทึมได้ดังนี้ อัลกอริทึมของเราจะรับกราฟสองส่วน $G=(U\cup V,E')$ และให้ผลลัพธ์เป็น การจับคู่ที่ดีที่สุด โดยเราจะเริ่มจากการแปลง G ไปเป็นกราฟ N ตามวิธีที่ได้กล่าวไปส่วนที่แล้ว เนื่องจากเราทราบแหล่งต้นทาง s และแหล่งปลายทาง t ดังนั้นกราฟ N สามารถมองเป็น*กราฟสามส่วน* (tripartite graph) ที่มีจุดยอด $U\cup V\cup C$ และเขียนแทนด้วย $N=(U\cup V\cup C,E)$ อัลกอริทึมนี้เริ่มทำงานโดยการหา การไหลสูงสุด f ใด ๆ จาก s ไป t ใน N ซึ่งสามารถทำได้ในเวลาเชิงเส้น เพราะการไหลสูงสุดนี้สมมูลกับการจับคู่ใดๆ ใน G ในการหาการไหลที่ถูกที่สุดใน N อัลกอริทึมจะใช้กระบวนการย่อยชื่อ CANCELALL เพื่อกำจัดวิถีที่ยอมรับได้แบบผันกลับทั้งหมดใน f ซึ่งทฤษฎีบทย่อยที่ 17 จะรับประกันว่าการไหลสุดท้ายที่เราได้จะเป็นค่าที่ดีที่สุด รายละเอียดของอัลกอริทึมนี้จะอยู่ในภาพที่ 15

Algorithm CancelAll($N = (U \cup V \cup C, E)$)

1. If $|C| = 1$ then halt.
2. Divide C into C_1 and C_2.
3. Cancel(N, C_2, C_1).
4. Divide N into N_1 and N_2.
5. CancelAll(N_1).
6. CancelAll(N_2).

ภาพที่ 15 อัลกอริทึมแบบแบ่งแยกและจัดการ

CANCELALL ทำงานแบบแบ่งแยกและจัดการ โดยแบ่งปัญหาออกเป็นสองส่วนแล้วแก้ปัญหาแบบเรียกตัวเอง อัลกอริทึมจะแบ่งเซตของศูนย์ค่าใช้จ่าย C ออกเป็นสองส่วนเท่าๆ กัน คือ เซตย่อย C_1 และ C_2 โดยที่สำหรับทุก $c_i \in C_1$ และ $c_j \in C_2$ จะมีค่า $i<j$ สังเกตว่าเงื่อนไขดังกล่าวทำให้ไม่มีวิถีที่ยอมรับได้แบบผันกลับจาก C_1 ไป C_2 ดังนั้นวิถีที่ยอมรับได้แบบผันกลับระหว่าง C_1 และ C_2 จึงเหลือแค่วิถีที่ยอมรับได้แบบผันกลับจาก C_2 ไป C_1 เท่านั้น อัลกอริทึมจะกำจัดวิถีเหล่านี้โดยใช้วิธีตามหัวข้อการกำจัดวิถีจาก C_2 ไป C_1

หลังจากนั้น สังเกตว่าวิถีที่ยอมรับได้แบบผันกลับที่เหลือจะมีจุดเริ่มและสิ้นสุดอยู่ภายในแต่ละเซตย่อย C_1 หรือ C_2 เท่านั้น อัลกอริทึมจะทำงานแบบเรียกตัวเองเพื่อกำจัดวิถีที่ยอมรับได้แบบผันกลับที่เหลือ โดยแบ่ง N ออกเป็น N_1 และ N_2 ที่บรรจุวิถีที่ยอมรับได้แบบผันกลับทั้งหมดที่มีจุดเริ่มและสิ้นสุดอยู่ภายใน C_1 และ C_2 ตามลำดับ นั่นคือเราจะได้ N_1 ที่มีศูนย์ค่าใช้จ่ายเป็น C_1 และ N_2 มีศูนย์ค่าใช้จ่ายเป็น C_2 หลังจากนั้นจึงแก้ N_1 และ N_2 สำหรับขั้นตอนการแบ่งปัญหาเราจะกล่าวถึงในช่วงท้ายของส่วนนี้ ในหัวข้อการกำจัดวิถีจาก C_2 ไป C_1 เราจะให้รายละเอียดของอัลกอริทึม CANCEL ซึ่งกำจัดทุกวิถีที่ยอมรับได้แบบผันกลับจาก C_2 ไป C_1 ในเวลา $O(|E|\sqrt{(|U|)})$ และ หลังจากขั้นตอนนี้เราต้องการที่จะกำจัดทุกวิถีที่ยอมรับได้แบบผันกลับระหว่างศูนย์ค่าใช้จ่ายที่อยู่ในแต่ละ C_i เนื้อหาที่เหลือของส่วนนี้จะกล่าวถึงวิธีการสร้างปัญหาย่อย N_1 และ N_2 และความถูกต้องของอัลกอริทึม เริ่มแรกเราจะกล่าวถึงวิธีการหา N_1 และ N_2 หลังจากขั้นตอนที่ 3 ใน

อัลกอริทึม CANCELALL ให้ S เป็นเซตของจุดยอดใน N ที่สามารถเข้าถึงได้จาก C_2 เราสามารถสร้าง S ได้ในเวลา $O(|E|)$ โดยใช้อัลกอริทึมมาตรฐานสำหรับค้นหาในกราฟ ให้ N_2 เป็นกราฟย่อยของ N ที่เหนี่ยวนำโดย S และ N_1 เป็นกราฟย่อยที่ถูกเหนี่ยวนำโดยจุดยอดที่เหลือ เราจะพิสูจน์ความถูกต้องในสองทฤษฎีบทย่อยต่อไป

ทฤษฎีบทย่อยที่ 16 *สมมติว่าไม่มีวิถีที่ยอมรับได้แบบผันกลับจากจุดยอดใน C_2 ไปยังจุดยอดใน C_1 ให้ S แทนเซตของจุดยอดที่เข้าถึงได้จาก C_2 จะได้ว่าวิถีที่ยอมรับได้ใด ๆ ระหว่างศูนย์ค่าใช้จ่ายสองศูนย์ใน C_1 จะไม่ผ่านจุดยอดใด ๆ ใน S*

พิสูจน์

เราจะพิสูจน์โดยใช้ความขัดแย้ง สมมติให้มีวิถีที่ยอมรับได้จาก x ไป y เมื่อ $x,y \in C_1$ ที่ผ่านจุดยอด $s \in S$ เนื่องจาก s เข้าถึงได้จากจุดยอดบางจุด $z \in c_2$ ดังนั้นจะต้องมีวิถีที่ยอมรับได้จากจุดยอด z ไป y ซึ่งทำให้เกิดข้อขัดแย้ง

■

ทฤษฎีบทย่อยที่ 17 CANCELALL*(N) จะกำจัดวิถีที่ยอมรับได้แบบผันกลับทุกเส้น ใน N*

พิสูจน์

วิถีที่ยอมรับได้แบบผันกลับทุกเส้นจาก C_2 ไป C_1 จะถูกกำจัดในขั้นตอนที่ 3 แล้วทฤษฎีบทย่อยที่ 16 จะแสดงว่าในขั้นตอนการแบ่งปัญหา ทุกวิถีที่ยอมรับได้แบบผันกลับจากคู่ของศูนย์ค่าใช้จ่ายใน C_1 ทั้งหมดจะยังคงอยู่ใน N_1 นอกจากนี้จุดยอดในวิถีที่ยอมรับได้แบบผันกลับใดๆ จากคู่ของศูนย์ค่าใช้จ่าย ใน C_2 จะต้องเข้าถึงได้จาก C_2 แสดงว่าจุดยอดนั้นจะต้องอยู่ใน S เพราะฉะนั้นหลังจากการเรียกตัวเองจะไม่มีวิถีที่ยอมรับได้แบบผันกลับระหว่างคู่ของศูนย์ค่าใช้จ่ายในปัญหาย่อยเดียวกันใน C_i เหลืออยู่ ซึ่งจะสรุปเป็นทฤษฎีบทย่อยนี้ได้ถ้าเราสามารถแสดงได้ว่ากระบวนการนี้ไม่ทำให้เกิดวิถีที่ยอมรับได้แบบผันกลับจาก C_2 ไป C_1 เพิ่มขึ้น สังเกตว่าทุกเส้นเชื่อมระหว่าง N_1 และ N_2 จะไม่เข้ามาเกี่ยวข้องในการเรียกตัวเอง และเส้นเชื่อมเหล่านี้จะมีทิศทางจาก N_1 ไป N_2 เพราะ S มีความเป็นค่าสูงสุด เพราะฉะนั้นจะได้ว่าไม่มีวิถีที่ยอมรับได้จาก C_2 ไป C_1

■

การวิเคราะห์เวลาการทำงาน

ให้ $T(n,n',m,k)$ แทนเวลาการทำงานของอัลกอริทึมเมื่อ $|U|=n,|V|=n',|E|=m$, และ $|C|=k$ เพื่อความสะดวกเราจะสมมุติว่า k มีค่าเป็นยกกำลังของสอง ในหัวข้อการกำจัดวิถีจาก C_2 ไป C_1 เราจะแสดงว่า CANCEL ทำงานในเวลา $O(|E|\sqrt{(|U|)})$ ด้วยข้อสมมติดังกล่าว เราสามารถเขียนเวลาการทำงานในรูปของฟังก์ชันเวียนเกิดได้ดังนี้

$$T(n,n',m,k)=O(m\sqrt{(n)})+T(n_1,n'_1,m_1,k/2)+T(n_2,n'_2,m_2,k/2)$$

เมื่อ n_i,n'_i และ m_i แทนจำนวนของจุดยอดและเส้นเชื่อมใน N_i ตามลำดับ ซึ่งฟังก์ชันเวียนเกิดนี้สามารถแก้ได้เป็น $T(n,n',m,k)=O(m\log k\sqrt{(n)})$ เนื่องจาก $k=O(|U|)$ ทำให้เราได้ทฤษฎีบทหลักดังนี้

ทฤษฎีบทที่ 18 (ทฤษฎีบทหลัก) *ให้กราฟสองส่วน $G=(U\cup V,E)$ สามารถหาการจับกิ่งคู่ที่ดีที่สุด ในเวลา $O(|E|\sqrt{(|U|)}\log|U|)$*

การกำจัดวิถีจาก C_2 ไป C_1

ในส่วนนี้เราจะอธิบายการทำงานของอัลกอริทึมที่กำจัดทุกวิถีที่ยอมรับได้จาก C_2 ไป C_1 ใน R_f ปัญหานี้สามารถมองเป็นปัญหาการไหลสูงสุดได้ เมื่อแหล่งต้นทาง คือ จุดยอดใน C_2 และแหล่งปลายทาง คือ จุดยอดใน C_1 เราสามารถกำจัดวิถีเหล่านี้ได้โดยการคำนวณหาการไหลสูงสุดจาก C_2 ไป C_1 เพื่อให้ง่ายต่อการเข้าใจ เราจะสมมติว่ามีแหล่งต้นทางพิเศษ (super-source) s และแหล่งปลายทางพิเศษ (super-sink) t เชื่อมต่อไปยังจุดยอดใน C_2 และจุดยอดใน C_1 ตามลำดับ สังเกตว่า N มีความจุเป็นหนึ่งหน่วยทั้งหมด ดังนั้นอัลกอริทึมที่สามารถใช้แก้ปัญหาการไหลสูงสุดที่มีความจุหนึ่งหน่วยจะสามารถใช้ในขั้นตอนนี้ได้ อัลกอริทึมที่ดีที่สุดที่รู้จักในขณะนี้คืออัลกอริทึมของ Even and Tarjan (1975) และ Karzanov (1973) ซึ่งทำงานในเวลา $O(\min\{(|U|+|V|)^{2/3},|E|^{1/2}\}|E|)$ อย่างไรก็ตามในปัญหานี้โครงสร้างซึ่งช่วยให้อัลกอริทึมที่ดัดแปลงมาจากอัลกอริทึมของ Dinic ที่ใช้ในการหาการไหลสูงสุดโดยใช้การไหลกีดขวาง (blocking flow) ทำงานได้ในเวลา $O(|E|\sqrt{(|U|)})$ อัลกอริทึมแบบใช้การไหลกีดขวางของ Dinic มีรายละเอียดโดยย่อดังนี้ ให้เครือข่าย R ที่มีแหล่งต้นทาง s และแหล่งปลายทาง t จะเรียกการไหล g ว่าเป็น *การไหลกีดขวาง* ใน R ถ้าทุกวิถีจากแหล่งต้นทางไปแหล่งปลายทางประกอบด้วย เส้นเชื่อม

อิ่มตัว (saturated edge) ซึ่งก็คือเส้นเชื่อมที่ไม่มีความจุตกค้าง (residual capacity) โดยปกติเราจะเรียกการไหลกีดขวางว่าการไหลแบบละโมบ (greedy flow) เนื่องจากการไหลไม่สามารถเพิ่มได้โดยไม่เปลี่ยนเส้นทางของวิถีการไหลก่อนหน้า ในเครือข่ายที่มีความจุหนึ่งหน่วยเราสามารถใช้การค้นหาแนวลึก (depth-first search) เพื่อการหาการไหลกีดขวางได้ในเวลาเชิงเส้น อัลกอริทึมของ Dinic ทำงานใน *กราฟแบ่งชั้น* (layered graph) ซึ่งเป็นกราฟย่อยที่มีเส้นเชื่อมอย่างน้อยหนึ่งเส้นจาก s ไป t หมายความว่าเราจะต้องเพิ่มการไหลผ่านวิถีที่สั้นที่สุด (shortest path) อัลกอริทึมจะหาการไหลกีดขวางในกราฟแบ่งชั้นของกราฟตกค้างในรอบก่อน ต่อไปจะกล่าวถึงคุณสมบัติสำคัญที่ระบุว่าระยะทางจากแหล่งต้นทางไปแหล่งปลายทางจะเพิ่มขึ้นทุกครั้งที่หาการไหลกีดขวาง

ทฤษฎีบทย่อยที่ 19 *ให้ d_i เป็นความยาวของวิถีจาก s ไป t ที่สั้นที่สุดในกราฟตกค้างในรอบที่ i แล้ว $d_{i+1}>d_i$ สำหรับทุก i ที่เป็นไปได้*

ทฤษฎีบทย่อยนี้แสดงว่าอัลกอริทึมของ Dinic จะจบการทำงานหลังการหาการไหลกีดขวางในรอบที่ n เมื่อ n คือ จำนวนของจุดยอด เพราะหลังจากรอบที่ n ระยะทางระหว่างแหล่งต้นทางและแหล่งปลายทางจะมากกว่า n ซึ่งหมายความว่าไม่มีวิถีที่เพิ่มการไหลจาก s ไป t เหลืออยู่ในกราฟตกค้าง จำนวนรอบนี้จะลดน้อยลงไปตามประเภทของปัญหา Even and Tarjan (1975) และ Karzanov (1973) แสดงว่าในเครือข่ายที่มีความจุหนึ่งหน่วย อัลกอรึทึมของ Dinic จะจบการทำงานหลังรอบที่ $\min(n^{2/3},m^{1/2})$ เมื่อ m คือจำนวนของเส้นเชื่อม ในเครือข่ายแบบหนึ่งหน่วย (unit network) ซึ่งเป็นเครือข่ายที่ทุกจุดยอดมีดีกรีเข้าเท่ากับหนึ่ง หรือมีดีกรีออกเท่ากับหนึ่ง อัลกอริทึมของ Dinic จะจบการทำงานภายใน $O(\sqrt{(n)})$ รอบ (อ่านเพิ่มเติมในหนังสือของ Tarjan (1983))

เนื่องจากเครือข่ายที่เราใช้มีลักษณะคล้ายกับเครือข่ายแบบหนึ่งหน่วย เราสามารถแสดงได้เช่นกันว่าว่าในกรณีของเราอัลกอริทึมของ Dinic จะจบการทำงานใน $O(\sqrt{(|U|)})$ รอบ สำหรับการไหล f ใดๆ *การไหลตกค้าง* (residual flow) f' คือ การไหลในกราฟตกค้าง R_f ของ f ถ้า f' มีขนาดมากที่สุดใน R_f แล้ว $f+f'$ จะเป็นการไหลสูงสุดในกราฟเริ่มต้นด้วย ทฤษฎีบทย่อยถัดไปจะแสดงความสัมพันธ์ระหว่างขนาดของการไหลตกค้างสูงสุดกับระยะทางที่สั้นที่สุดจาก s ไป t ในกรณีของเรา โดยใช้การพิสูจน์ที่ดัดแปลงมาจากทฤษฎีบทที่ 8.8 ในหนังสือของ Tarjan (1983)

ทฤษฎีบทย่อยที่ 20 *ถ้าระยะทางสั้นที่สุดจาก s ไป t ในกราฟตกค้างเท่ากับ d เมื่อ $d>4$ แล้ว การไหลตกค้างสูงสุดจะมีขนาดไม่เกิน $O(|U|/d)$*

พิสูจน์

การไหลตกค้างสูงสุดในเครือข่ายที่มีความจุหนึ่งหน่วยสามารถแบ่งแต่ละส่วนออกเป็นเซต $\mathcal{P}$ ของวิถีที่มีเส้นเชื่อมที่ไม่มีส่วนร่วมกัน (edge-disjoint path) เมื่อจำนวนของวิถีมีขนาดเท่ากับปริมาณของการไหล และทุกวิถีมีความยาวอย่างน้อย d เห็นได้ชัดว่าวิถีแต่ละเส้นจะประกอบด้วยแหล่งต้นทาง, แหล่งปลายทาง และศูนย์ค่าใช้จ่ายสองศูนย์ พิจารณาวิถี $P\in\mathcal{P}$ ใดๆ ที่มีความยาว l จะประกอบด้วยจุดยอด l-3 จุด จาก $U\cup V$ เนื่องจากกราฟเริ่มต้นเป็นกราฟสองส่วน และจะประกอบด้วยจุดยอดอย่างน้อย $\lfloor(l\text{-}3)/2\rfloor\geq\lfloor(d\text{-}3)/2\rfloor\geq(d\text{-}4)/2$ จุดจาก U สังเกตว่าแต่ละวิถีใน $\mathcal{P}$ ประกอบด้วยเซตที่ไม่มีส่วนร่วมกันของจุดยอดใน U เนื่องจากจุดยอดใน U มีดีกรีเข้าเท่ากับหนึ่ง เพราะฉะนั้นเราจะสรุปได้ว่าจะมีวิถีได้ไม่เกิน $2|U|/(d\text{-}4)$ เส้นใน $\mathcal{P}$ และได้ทฤษฎีบทย่อยตามมา เนื่องจากวิถีแต่ละเส้นประกอบด้วยการไหลหนึ่งหน่วย

■

จากทฤษฎีบทย่อยทั้งสองบทนี้ เราจะได้ทฤษฎีบทย่อยหลักดังนี้

ทฤษฎีบทย่อยที่ 21 CANCEL *ทำงานภายในเวลา $O(|E|\sqrt{|U|})$*

พิสูจน์

จากทฤษฎีบทย่อยที่ผ่านมา เราสามารถสรุปได้ว่าในการทำงานรอบที่ $O(\sqrt{|U|})$ จะมีการไหลตกค้างเหลือไม่เกิน $O(\sqrt{|U|})$ หน่วย และในแต่ละรอบอัลกอริทึมจะเพิ่มการไหลอย่างน้อยหนึ่งหน่วย นั่นหมายความว่าอัลกอริทึมจะทำงานไม่เกิน $O(\sqrt{|U|})$ รอบ และเนื่องจากแต่ละรอบทำงานในเวลา $O(|E|)$ ดังนั้นเราจะได้ทฤษฎีบทย่อยดังกล่าว

■

<u>กรณีทั่วไป</u>

เราสามารถมองปัญหานี้ในรูปที่มีความยืดหยุ่นมากขึ้น ในผลงานของ Harvey et. al. พิจารณาเฉพาะในกรณีที่ฟังก์ชันค่าใช้จ่ายเหมือนกันทั้งหมด เราจะลดหย่อนเงื่อนไขนี้ โดยยอมให้จุดยอดแต่ละจุดมีฟังก์ชันค่าใช้จ่ายที่ต่างกัน สำหรับแต่ละ $v\in V$ ให้ $f_v:Z_+\rightarrow R$ เป็นฟังก์ชันไม่ลด

ค่าใช้จ่ายของการจับกึ่งคู่บนจุดยอด v เป็น $f_v(deg_M(v))$ ในกรณีของค่าใช้จ่ายทั่วไป การลดทอนปัญหาจะคล้ายกับที่อธิบายในหัวข้อการลดทอนปัญหา เมื่อขนาดของเซต C ของศูนย์ค่าใช้จ่ายไม่เท่ากับ $|U|$ แต่เป็น $O(|E|)$ ซึ่งเป็นจำนวนของค่าที่แตกต่างกันของ f_v อัลกอริทึมของเราจะทำงานในเวลา $O(|E|\sqrt{(|U|)}\log|C|)=O(|E|\sqrt{(|U|)}\log|E|)$

ผลการศึกษาปัญหาในกรณีที่กราฟมีน้ำหนัก

ในกรณีที่กราฟมีน้ำหนัก เรามองปัญหานี้เป็นปัญหาการมอบหมายงานให้กับเครื่องจักร โดยที่เครื่องจักรหนึ่งเครื่องสามารถรับงานได้มากกว่าหนึ่งงาน และแต่ละงานใช้เวลาไม่เท่ากัน ในลักษณะนี้เมื่อมีการมอบหมายงานให้เครื่องจักรแล้ว เราสามารถเรียงลำดับของงานในแต่ละเครื่องจักร ให้มีเวลารอรวมน้อยที่สุดได้ โดยการเรียงลำดับงานจากงานที่ใช้เวลาน้อยไปยังงานที่ใช้เวลามาก ดังนั้นปัญหาที่เราต้องแก้จึงมีเพียงการจัดรูปแบบการมอบหมายงานให้กับเครื่องจักร

ในหัวข้อนิยามของปัญหาอย่างเป็นรูปนัย เราได้ให้นิยามอย่างเป็นรูปนัยของปัญหาการจับกึ่งคู่แบบมีน้ำหนัก เพื่อให้ได้ผลลัพธ์เป็นรูปแบบการมอบหมายงานที่มีเวลารอรวมน้อยที่สุด ในทฤษฎีบทย่อยต่อไป จะแสดงว่าการให้นิยามของปัญหาในรูปแบบดังกล่าวให้ผลลัพธ์การทำงานที่ถูกต้อง

ทฤษฎีบทย่อยที่ 22 *คำตอบของปัญหาการจับกึ่งคู่ที่ดีที่สุดตามนิยามในหัวข้อการลดทอนปัญหาเป็นรูปแบบการมอบหมายงานให้กับเครื่องจักรที่มีเวลารอรวมน้อยที่สุด*

พิสูจน์

ให้ U แทนเซตของงาน, V แทนเซตของเครื่องจักร และ ให้ M เป็นการมอบหมายงานใดๆ พิจารณาเวลารอรวมของงาน $u_1,u_2,...,u_{degM(v)}$ ที่มอบหมายให้เครื่องจักร $v\in V$ เรียงตามลำดับของ $w(v,u_i)$ จากน้อยไปมาก เขียนแทน $w(v,u_i)$ ด้วย w_i อันดับแรกเราจะอ้างว่าที่เครื่องจักร $v\in V$ เราสามารถจัดลำดับของงานให้ได้เวลารอรวมต่ำที่สุด โดยเรียงลำดับงานตามเวลาที่ใช้ทำงานจากน้อยไปมาก จะได้เวลารอของงานลำดับที่ i เป็น$w_1+w_2+...+w_i$ และได้เวลารอรวมเป็น $w_1+(w_1+w_2)+...+(w_1+w_2+...+w_{degM(v)})$ เนื่องจาก $w_1,w_2,...,w_i$ เป็นค่าน้อยที่สุด i ตัวแรก เพราะฉะนั้นเวลารอของงานลำดับที่ i ซึ่งจัดลำดับงานในรูปแบบอื่นจะมีเวลารอที่นานกว่านี้เสมอ ดังนั้นเราจะได้ว่าการจัดเรียงงานที่เครื่องจัก v ใดๆ ตามลำดับของ w_i จากน้อยไปมากจะให้เวลารอรวมที่ต่ำที่สุด นอกจากนี้เราสามารถเขียนรูปของเวลารอรวมใหม่ได้เป็น

$1 \cdot w_{degM(v)}) + 1 \cdot w_{degM(v)-1}) + ... + deg_M(v) \cdot w_1$ เท่ากับค่า $cost_M(v)$ และได้เวลารอรวมของงานที่ทุกเครื่องจักรเท่ากับ $\sum_{v \in V} cost_M(v) = cost(M)$ นั่นคือนิยามของปัญหาในรูปแบบดังกล่าวให้ผลลัพธ์เป็นการมอบหมายงานที่ให้เวลารอรวมต่ำที่สุด

■

จากการศึกษาคุณสมบัติของปัญหา เราได้วิธีการลดทอนปัญหาการจับกึ่งคู่แบบมีน้ำหนักบนกราฟ $G=(U \cup V, E)$ ไปเป็นปัญหาการจับคู่แบบมีน้ำหนักบนกราฟ $G'=(U', V')$ ดังที่แสดงวิธีการลดทอนในหัวข้อการลดทอนปัญหา

<u>การแก้ปัญหาโดยตรง</u>

จากการลดทอนปัญหา เราจะได้ปัญหาการจับคู่แบบมีน้ำหนักบนกราฟ G' ที่มีจำนวนจุดเป็น $|U|+|E|$ และมีจำนวนเส้นเชื่อมเป็น $O(|U||E|)$ หากเราแก้ปัญหานี้โดยใดยใช้อัลกอริทึมสำหรับการแก้ปัญหาการจับคู่แบบมีน้ำหนักของ Edmonds and Karp (1970) และ Tomizawa (1971) จะใช้เวลาทำงานเป็น $O(|U|^2|E|+|U||E|\log|U|)$ และในกรณีที่กราฟมีน้ำหนักเป็นจำนวนเต็มเราสามารถแก้ได้โดยอัลกอริทึมของ Gabow and Tarjan (1989) ซึ่งใช้เวลาทำงานเป็น $O(|U|^{3/2}|E|\log|U|W)$ เมื่อ W และ W' คือน้ำหนักของเส้นเชื่อมที่หนักที่สุดใน G และ G' ตามลำดับ อย่างไรก็ตามจำนวนจุดยอดใน U มีจำนวนน้อยกว่าจำนวนจุดยอดใน V' มาก เราสามารถใช้เทคนิคในการจับคู่แบบจำนวนจุดยอดไม่สมดุลย์ (vertex unbalanced matching) ของ Kao et al. (2001) ซึ่งเลือกเฉพาะคำนวณเฉพาะเส้นเชื่อมที่มีน้ำหนักน้อยที่สุดจำนวน $|U|^2$ เส้น ทำให้เวลาการทำงานลดลงเป็น $O(|U|^3)$ และ $O(|U|^{7/2}\log|U|W)$ เมื่อน้ำหนักของเส้นเชื่อมเป็นจำนวนเต็ม

<u>เทคนิคการแก้ปัญหาโดยอาศัยคุณสมบัติเฉพาะของปัญหา</u>

การแก้ปัญหาทำโดยแปลงปัญหาการจับกึ่งคู่ที่ดีที่สุดไปเป็นปัญหาการจับคู่แบบมีน้ำหนักบนกราฟสองส่วน โดยใช้อัลกอริทึมที่มีพื้นฐานมาจากอัลกอริทึมของ Edmonds and Karp (1970), และ Tomizawa (1971) ในการแก้ปัญหาการจับคู่แบบมีน้ำหนัก โดยใช้คุณสมบัติเฉพาะดังที่แสดงไว้ในหัวข้อคุณสมบัติของปัญหา เพื่อลดเวลาการทำงานของอัลกอริทึมจากการแก้ปัญหาโดยตรงที่ใช้เวลา $O(|E'|+|U'|\log|U|)=O(|U||E|+|E|\log|U|)$ เป็น $O(|E|\log|U|)$ เราจะแยกอธิบายส่วนสำคัญแต่ละส่วนของอัลกอริทึมดังนี้

1 อัลกอริทึมหลัก

2 โครงสร้างข้อมูลที่จุดยอดพิเศษ

<u>อัลกอริทึมหลัก</u>

อัลกอริทึมหลักที่เราใช้จะหาการจับกึ่งคู่แบบมีน้ำหนักบนกราฟ $G=(U\cup V,E)$ โดยพิจารณาเป็นการปัญหาการจับคู่สูงสุดที่มีน้ำหนักต่ำสุดที่สมมูลกันบนกราฟ $G'=(U\cup V',E')$ ตามวิธีการลดทอนในหัวข้อการลดทอนปัญหา โดยใช้เทคนิคที่ดัดแปลงมาจากอัลกอริทึมของ Edmonds and Karp (1970) และ Tomizawa (1971) มาประยุกต์ใช้ในปัญหาในกรณีพิเศษ ซึ่งได้ศึกษาคุณสมบัติของปัญหาดังกล่าวไว้ในหัวข้อคุณสมบัติของปัญหา อัลกอริทึมดังกล่าวแก้ปัญหาโดยการเพิ่มขนาดของการจับคู่ครั้งละหนึ่ง ในการทำงานแต่ละรอบจะเพิ่มขนาดของการจับคู่โดยแต่งเติม (augment) การจับคู่ด้วยวิถีแต่งเติมที่มีค่าใช้จ่ายต่ำที่สุด (cheapest/shortest augmenting path) การจับคู่ที่ได้จากการทำงานในแต่ละรอบนี้จะเป็นการจับคู่ที่มีน้ำหนักน้อยที่สุดในการจับคู่ที่มีขนาดเท่ากัน เรียกการจับคู่ที่มีลักษณะเช่นนี้ว่า*สุดขีด* (extreme) อัลกอริทึมหลักแสดงในภาพที่ ?

อัลกอริทึมรับกราฟสองส่วน $G=(U\cup V,E)$ และฟังก์ชันน้ำหนัก $w:E\rightarrow R$ และให้ผลลัพธ์เป็นการจับคู่สูงสุดที่มีน้ำหนักต่ำสุด M เพื่ออธิบายอัลกอริทึมดังกล่าวเราจะนิยาม *กราฟตกค้างที่สัมพันธ์กับการจับคู่ M* เขียนแทนด้วย $R_M(G)=(U\cup V,A)$ เป็นกราฟระบุทิศทางโดยแต่ละเส้นเชื่อม $(u,v)\in(E-M)$ จะมีเส้นเชื่อมระบุทิศทาง $(u,v)\in A$ มีน้ำหนักเป็น $w(u,v)$ และสำหรับทุกเส้นเชื่อม $(u,v)\in M$ จะมีเส้นเชื่อมระบุทิศทาง $(v,u)\in A$ มีน้ำหนักเป็น $-w(u,v)$ แทนเซตของจุดยอดใน U ที่ไม่ประชิดกับเส้นเชื่อมใน M ด้วย U_M และแทนเซตของจุดยอดใน V ที่ไม่ประชิดกับเส้นเชื่อมใน M ด้วย V_M นิยาม*วิถีแต่งเติมที่สัมพันธ์กับการจับคู่ M* เป็นวิถีจาก U_M ไป V_M ใน $R_M(G)$ *วิถีแต่งเติมที่สั้นที่สุด* หรือ *วิถีแต่งเติมถูกที่สุด* คือ วิถีแต่งเติมที่มีน้ำหนักรวมของเส้นเชื่อมใน $R_M(G)$ ต่ำที่สุด *วงรอบค่าใช้จ่ายลบ* คือวงรอบใน $R_M(G)$ ที่มีผลรวมของน้ำหนักในเส้นเชื่อมเป็นค่าลบ

อัลกอริทึมจะใช้แนวคิดของฟังก์ชันราคา และค่าใช้จ่ายลดทอน โดยใช้ระยะทางl สั้นที่สุดจาก U_M ในแต่ละรอบเป็นฟังก์ชันราคาในรอบถัดไป ซึ่งจะได้ค่าใช้จ่ายลดทอนสำหรับทุกเส้นเชื่อมใน $R_M(G')$ ไม่เป็นค่าลบ และใช้อัลกอริทึมที่ดัดแปลงมาจาก

อัลกอริทึมของ Dijkstra ในการหาวิถีแต่งเติมที่สั้นที่สุด โดยแทนที่จะจัดการกับกราฟ *G'* โดยตรง เราได้ออกแบบโครงสร้างข้อมูลโดยอาศัยคุณสมบัติที่ได้ศึกษามาเพื่อจัดการกับเส้นเชื่อมใน *E'* อัลกอริทึมจะมองเส้นเชื่อมที่ใน *E'* ที่เป็นเส้นเชื่อมที่สมนัยกันเพียงเส้นเดียวใน *E* ทำให้เวลาการทำงานขึ้นอยู่กับจำนวนเส้นเชื่อมใน *E* แทน

อัลกอริทึมสำหรับหาวิถีแต่งเติมที่สั้นที่สุดดังกล่าวแสดงในภาพที่ 15 โดยประกอบด้วยฟิโบนัคชีฮีป (Fibonacci heap) *H* ซึ่งมีการทำงานคือ enqueue, extract-min และ decrease-key โครงสร้างข้อมูล *D*(*s*) สำหรับเซต *S*(*v*) ซึ่งมีการทำงานคือ init, addline และ extract และอัลกอริทึมย่อย SCAN ดังแสดงในภาพที่ 16

Algorithm SM-Min-Matching$(G = (U \cup V, E), w)$

1. create $G' = (U \cup V', E')$, and $w' : E' \to \mathbb{R}$ from G and w
2. $M \leftarrow \emptyset$
3. $\forall v \in U \cup V', p(v) \leftarrow 0$
4. do
5. $\quad (P, d) \leftarrow \text{SAP}(G, M, w_p)$
6. $\quad p \leftarrow d - p$
7. $\quad M \leftarrow (M - P) \cup (P - M)$
8. while $P \neq \emptyset$
9. return M

ภาพที่ 14 อัลกอริทึมสำหรับหาการจับคู่เชิงน้ำหนัก

Algorithm SAP$(G = (U \cup V, E), M, w_p)$

1. $\forall v \in V$, init$(D(v))$
2. $\forall v \in U \cup V' - U_M, d(v) \leftarrow \infty$
3. $T \leftarrow \emptyset$
4. for all $u \in U_M$
5. $\quad d(u) \leftarrow 0$
6. $\quad$ SCAN(u)
7. do
8. $\quad v \leftarrow$ extract-min(H)
9. $\quad (u, v) \leftarrow$ extract$(D(v))$
10. $\quad$ SCAN(u)
11. while U not contain in T
12. find augmenting path P on shortest path tree T
13. return (P, d)

ภาพที่ 15 อัลกอริทึมย่อยสำหรับหาวิถีแต่งเติมที่สั้นที่สุด

Algorithm SCAN(u)

1. for all $(u, v) \in E - M$
2. addline$((u, v), D(v))$
3. if $v \notin H$
4. enqueue$(d(v), H)$
5. otherwise
6. decrease-key$(d(v), H)$

ภาพที่ 16 อัลกอริทึมย่อย SCAN

ในส่วนต่อไปจะกล่าวถึงโครงสร้างข้อมูลสำหรับเซตของจุดยอด $S(v)$ ซึ่งเป็นส่วนสำคัญในการทำงานของอัลกอริทึม

<u>โครงสร้างข้อมูล</u>

โครงสร้างข้อมูล $D(v)$ สำหรับเซต $S(v)$ จะเก็บเซตของเส้นเชื่อมที่ประชิดกับ $S(v)$ ในรูปของฟังก์ชันเส้นตรง $Y_{u,v}$ และจัดการกับฟังก์ชันของเส้นขอบล่างที่เกิดจากการตัดกันของเส้นตรง L_v ซึ่งมีรูปร่างเป็นคอนเว็กซ์ ส่วนการทำงานของโครงสร้างข้อมูลมีสามส่วน คือ init, addline และ extract ซึ่งมีรายละเอียดดังนี้

ส่วนการทำงานของโครงสร้างมีดังนี้

init ทำการหาค่าของ $m(u,v)$ สำหรับทุกเส้นเชื่อม $(u,v) \in E$ ที่ประชิดกับ $S(v)$ และตั้งค่าเริ่มต้นให้ $p_l(u)$ และ $p_r(u)$ เท่ากับค่า $m(u,v)$ โดยค่า $p_l(u)$ และ $p_r(u)$ เป็นตัวชี้ค่าน้อยที่สุดสำหรับฝั่งที่ฟังก์ชันราคาลดทอนเป็นฟังก์ชันไม่เพิ่ม และฝั่งที่ฟังก์ชันราคาลดทอนเป็นฟังก์ชันไม่ลดตามลำดับ เพื่อใช้เปรียบเทียบหาค่าต่ำสุดของเส้นตรง $Y_{u,v}$

addline เพิ่มเส้นตรง $Y_{u,v}$ เข้าไปในเส้นขอบล่าง L_v ปรับค่า $p_l(u)$ และ $p_r(u)$ ให้เท่ากับจุดต่ำสุดในบริเวณที่ $Y_{u,v}$ เป็นเส้นขอบล่างใน L_v และสำหรับเส้นตรงที่ตัดกับ $Y_{u,v}$ ให้เลื่อน p_l และ p_r ไปบริเวณที่มีค่าต่ำสุดเช่นกัน ซึ่งเส้นตรงที่จะถูกตัดมีได้ไม่เกิน2

เส้น การเพิ่มเส้นตรงสามารถทำงานได้ในเวลา $O(\log|U|)$ โดยใช้เทคนิคพื้นฐานในการจัดการกับรูปร่างคอนเว็กซ์โพลีกอน ซึ่งเมื่อมีการ addline จะต้องมีการปรับค่าต่ำสุดในฮีปซึ่งจะมีการ decrease-key ไม่เกิน 2 ครั้ง เท่ากับจำนวนเส้นที่เส้นใหม่เข้าไปตัด และ เพิ่มค่าเข้าไปในฮีปอีก 1 ครั้ง นอกจากนั้นจะใช้เวลาในการปรับค่าต่ำสุดที่จากเส้นเชื่อม (u,v) อีก 1 ครั้ง ซึ่งรวมทั้งหมดแล้วสามารถทำได้ในเวลา $O(\log|U|)$ กระบวนการนี้จึงใช้เวลาไม่เกิน $O(\log|U|)$

extract แต่ละเส้นเชื่อม (u,v) ใน $D(v)$ จะใช้ค่าต่ำสุดของ $p_l(u)$ และ $p_r(v)$ เป็นตัวแทน ซึ่งจะเก็บค่าเหล่านี้ไว้ในฮีป $Q(v)$ เมื่อมีการเรียกใช้การทำงานนี้ โครงสร้างข้อมูลจะให้เส้นเชื่อม (u,v) ที่มีระยะทางสั้นที่สุดจาก $Q(v)$ ซึ่งเราจะทราบว่าค่านั้นหมายถึงจุดยอด v^i ใดใน $S(v)$ จากตัวชี้ p_l และ p_r จากนั้นจะเลื่อน p_l หรือ p_r ไปยังตำแหน่งต่ำสุดตัวถัดไป ซึ่ง p_l จะลดค่าลงเรื่อยๆ (ตำแหน่งเลื่อนไปทางซ้าย) ส่วน p_r จะเพิ่มค่าขึ้นเรื่อยๆ (ตำแหน่งเลื่อนไปทางขวา) การทำงานนี้จะคืนค่าเป็นเส้นเชื่อม (u,v) เมื่อมีการ extract เกิดขึ้น จะต้องมีการปรับค่าในฮีปหลักเพื่อเปลี่ยนค่าน้อยที่สุดในฮีปหลัก ซึ่งสามารถทำได้ในเวลา $O(\log|U|)$ และ กระบวนการนี้จะเรียกใช้การทำงานในฮีป $Q(v)$ คือ extract-min และ insert-key เพื่อปรับค่าต่ำที่สุดที่ได้จากเส้นเชื่อม (u,v) ซึ่งใช้เวลาไม่เกิน $O(\log|U|)$ เช่นกัน

<u>การจัดการจุดยอดที่ยังไม่ถูกจับคู่</u>

ในส่วนของโครงสร้างข้อมูลนี้เราพิจารณาเพียงจุดยอดที่ถูกจับคู่แล้วเท่านั้น ในส่วนของจุดยอดที่ยังไม่ถูกจับคู่ เราจะเลือกมาพิจารณาจากแต่ละ $S(v)$ เพียงเซตละหนึ่งจุดยอดที่มีหมายเลขต่ำที่สุด ซึ่งเพียงพอต่อความถูกต้องของอัลกอริทึมดังที่แสดงในหัวข้อคุณสมบัติ และจุดยอดที่ยังไม่ถูกจับคู่จะให้มีค่าฟังก์ชันราคาเป็น 0 ทั้งหมด เมื่อได้ระยะทางที่สั้นที่สุดจาก U_M ไปยังทุกจุดที่ถูกจับคู่ เราสามารถหาระยะทางที่สั้นที่สุดไปยังจุดยอดที่ยังไม่ถูกจับคู่โดยการตรวจสอบทุกเส้นเชื่อมได้ในเวลา $O(|E|)$ ซึ่งไม่เกินเวลาการทำงานของการหาระยะทางที่สั้นที่สุดในส่วนแรก

วิเคราะห์เวลาการทำงาน

ในอัลกอริทึมหลักเราจะมีการทำงาน $|U|$ รอบ การทำงานแต่ละรอบจะหาวิถีแต่งเติมที่สั้นที่สุดจะทำงาน อัลกอริทึมจะหาระยะทางที่สั้นที่สุดไปยังจุดยอดที่จับคู่แล้ว ซึ่งเรียกใช้การทำงาน ADDLINE ของโครงสร้างข้อมูล $|E|$ ครั้ง แต่ละครั้งสามารถทำงานได้ในเวลา $O(\log|U|)$ และเรียกใช้การทำงาน EXTRACT ของโครงสร้างข้อมูลไม่เกิน $|U|$ ครั้ง แต่ละครั้งสามารถทำงานได้ในเวลา $O(\log|U|)$ ซึ่งจะได้เวลาการทำงานทั้งหมดในแต่ละรอบเป็น $O(|E|\log|U|)$ ดังนั้นเวลาการทำงานทั้งหมดของอัลกอริทึมคือ $O(|U||E|\log|U|)$

สรุป

จากผลการวิจัยปัญหาการจับกึ่งคู่ ได้อัลกอริทึมสำหรับกรณีที่กราฟมีน้ำหนักที่ทำงานในเวลา $O(\sqrt{(|U|)}|E|\log|U|)$ และได้อัลกอริทึมสำหรับกรณีที่กราฟไม่มีน้ำหนักที่ทำงานในเวลา $O(|U||E|\log|U|)$ ซึ่งทั้งสองกรณีอัลกอริทึมที่ได้มีเวลาทำงานช้ากว่าอัลกอริทึมในการแก้ปัญหาการจับคู่ที่ดีที่สุดที่เป็นที่รู้จักกันเพียงตัวประกอบลอการิทึม เป็นที่น่าสนใจว่ามีอัลกอริทึมในการแก้ปัญหาการจับกึ่งคู่ที่มีเวลาการทำงานเท่ากับอัลกอริทึมสำหรับแก้ปัญหาการจับคู่หรือไม่

เอกสารและสิ่งอ้างอิง

Ahuja, R.K., A.V. Goldberg, J.B. Orlin, and R.E. Tarjan. 1992. **Finding minimum-cost flows by double scaling**. Math. Program., 53(3):243–266.

Bellman, R. 1958. **On a routing problem**. Quarterly of Applied Mathematics, 16(1):87–90.

Berge, C. 1957. **Two theorems in graph theory**. In Proceedings of the Natl. Acad. Sci., volume 43, pages 842–844.

Cherkassky, B.V., A.V. Goldberg, and T. Radzik. 1994. **Shortest paths algorithms: theory and experimental evaluation**. In SODA '94: Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms, pages 516–525. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Dijkstra, E.W. 1959. **A note on two problems in connection with graphs**. Numerische Mathematik, 1:269–271.

Dinic, E.A. 1970**. Algorithm for solution of a problem of maximum flow in networks with power estimation**. Soviet Mathematics Doklady, 11:1277–1280.

Edmonds, J. and R.M. Karp. 1970. **Theoretical improvements in algorithmic efficiency for network flow problems**. In Combinatorical Structures and Their Applications, pages 93–96. Gordon and Breach, New York.

Edmonds, J. and R.M. Karp. 1972. **Theoretical improvements in algorithmic efficiency for network flow problems**. J. ACM, 19(2):248–264.

Even, S. and R.E. Tarjan. 1975. **Network flow and testing graph connectivity**. SIAM J. Comput., 4(4):507–518.

Feder, T. and R. Motwani. 1995. **Clique partitions, graph compression and speeding-up algorithms**. J. Comput. Syst. Sci., 51(2):261–272.

Ford Jr., L.R. and D. R. Fulkerson. 1962. **Flows in Networks**. Princeton University Press, Princeton.

Gabow, H.N. 1983. **Scaling algorithms for network problems**. In FOCS, pages 248–257.

Gabow, H.N. and R.E. Tarjan. 1989. **Faster scaling algorithms for network problems**. SIAM J. Comput., 18(5):1013–1036.

Goldberg, A.V. 1993. **Scaling algorithms for the shortest paths problem**. In SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms, pages 222–231. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Goldberg, A.V. and R. Kennedy. 1995. **An efficient cost scaling algorithm for the assignment problem**. Mathematical Programming, 71:153–177.

Goldberg, A.V. and S. Rao. 1998. **Beyond the flow decomposition barrier**. J. ACM, 45(5):783–797.

Goldberg, A.V. and R.E. Tarjan. 1987. **Solving minimum-cost flow problems by successive approximation**. In STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing, pages 7–18, New York, NY, USA. ACM Press.

Goldberg, A.V. and R.E. Tarjan. 1988. **A new approach to the maximum-flow problem**. Journal of the ACM, 35(4):921–940. Preliminary version in Proc. 18th Annual ACM Symposium on the Theory of Computing, pages 136–146, 1986.

Goldberg, A.V. and R.E. Tarjan. 1990. **Finding minimum-cost circulations by successive approximation**. Math. Oper. Res., 15(3):430–466.

Harvey, N., R. Ladner, L. Lovász, and T. Tamir. 2003. **Semimatchings for bipartite graphs and load balancing**. In Proc. 8th WADS, pages 284–306. to appear in *J. of Algorithms*.

Hopcroft, J. and R. Karp. 1973. **An $n^{5/2}$ algorithm for matchings in bipartite graphs**. SIAM J. on Computing, 2:135–158.

Kao, M.-Y., T.W. Lam, W.-K. Sung, and H.-F. Ting. 2001. **An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings.** J. Algorithms, 40(2):212–233.

Karzanov, A. 1973. **On finding maximum flows in networks with special structure and some applications**. Matematicheskie Voprosy Upravleniya Proizvodstvom, 5:81–94.

Karzanov, A. 1974. **Determining the maximal flow in a network by the method of preflows**. Soviet Mathematics Doklady, 15:434–437.

Lawler, E. 2001. *Combinatorial Optimization: Networks and Matriods*. Dover.

Lovász, L. and M. D. Plummer. 1986. *Matching Theory*. Number 121 in North-Holland Mathematical Studies / Number 29 in Annals of Discrete Mathematics. North-Holland, Amsterdam.

Moore, E.F. 1959. **The shortest path through a maze.** In Proc. of the International Symposium on the Theory of Switching, pages 285–292. Harvard University Press.

Orlin, J. 1988. **A faster strongly polynomial minimum cost flow algorithm**. In STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing, pages 377–387. ACM Press.

Schrijver, A. 2003. **Combinatorial optimization : polyhedra and efficiency**. volume A, paths, flows, matchings, chapter 1-38. Springer. Schrijver.

Sleator, D.D., and R.E. Tarjan. 1983. **A data structure for dynamic trees**. J. Comput. Syst. Sci., 26(3):362–391.

Tarjan, R.E. 1983. **Data structures and network algorithms.** Society for Industrial and Applied Mathematics.

Tomizawa, N. 1971. **On some techniques useful for solution of transportation network problems.** In Networks 1, pages 173–194.

ประวัติการศึกษา และการทำงาน

ชื่อ –นามสกุล	บัณฑิต เลขานุกิจ
วัน เดือน ปี ที่เกิด	12 เมษายน พ.ศ. 2524
สถานที่เกิด	จังหวัด ประจวบคีรีขันธ์
ประวัติการศึกษา	วศ.บ. (วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเกษตรศาสตร์ (พ.ศ. 2547)
ตำแหน่งหน้าที่การงานปัจจุบัน	-
สถานที่ทำงานปัจจุบัน	-
ผลงานดีเด่นและรางวัลทางวิชาการ	-
ทุนการศึกษาที่ได้รับ	-