ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์

วิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์)

ปริญญา

วิศวกรรมคอมพิวเตอร์	วิศวกรรมคอมพิวเตอร์
สาขา	**ภาควิชา**

เรื่อง อัลกอริทึมแบบเวลาเชิงเส้นสำหรับปัญหาการคัดลอกหลายยีน

A Linear-Time Algorithm for the Multiple Gene Duplication Problem

นามผู้วิจัย นายวัชรพัฐ เมตตานันท

ได้พิจารณาเห็นชอบโดย

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก .....

(ผู้ช่วยศาสตราจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D.)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม .....

(ผู้ช่วยศาสตราจารย์พีรวัฒน์ วัฒนพงศ์, Ph.D.)

หัวหน้าภาควิชา .....

(ผู้ช่วยศาสตราจารย์เขมะฑัต วิภาตะวนิช, Ph.D.)

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์รับรองแล้ว

.....

(รองศาสตราจารย์กัญจนา ธีระกุล, D.Agr.)

คณบดีบัณฑิตวิทยาลัย

วันที่ ..... **เดือน** ..... **พ.ศ.** .....

วิทยานิพนธ์

เรื่อง

อัลกอริทึมแบบเวลาเชิงเส้นสำหรับปัญหาการคัดลอกหลายยีน

A Linear-Time Algorithm for the Multiple Gene Duplication Problem

โดย

นายวัชรพัฐ เมตตานันท

เสนอ

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์
เพื่อความสมบูรณ์แห่งปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์)
พ.ศ. 2552

วัชรพัฐ เมตตานันท 2552: อัลกอริทึมแบบเวลาเชิงเส้นสำหรับปัญหาการคัดลอกหลายยีน ปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมคอมพิวเตอร์) สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก: ผู้ช่วยศาสตราจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D. 36 หน้า

การสร้างต้นไม้วิวัฒนาการ เป็นหนึ่งในเป้าหมายที่สำคัญที่สุดในการวิจัยทางชีววิทยาด้านที่เกี่ยวข้องกับการวิวัฒนาการ ในวิทยานิพนธ์ฉบับนี้ เราสนใจประเด็นหนึ่งเกี่ยวกับการพิจารณาความสัมพันธ์ในเชิงวิวัฒนาการโดยดูจากความสัมพันธ์ของยีน ในรูปแบบที่เรียกว่าต้นไม้ยีน ซึ่งยีนแต่ละชนิดจะให้ความสัมพันธ์ระหว่างสิ่งมีชีวิตแตกต่างกัน ความขัดแย้งตรงนี้เกิดได้จากกระบวนการที่เรียกว่า การคัดลอกยีน ซึ่งเป็นกระบวนการที่ผิดปรกติ Guigo, Muchnik, และ Smith ได้นิยามปัญกาการคัดลอกหลายยีนขึ้น ซึ่งเป็นการหาจำนวนการคัดลอกหลายยีนที่น้อยที่สุดที่อธิบายความขัดแย้งของต้นไม้สปีชีส์และต้นไม้ยีนทั้งหมดได้ เร็วๆนี้ Bansal และ Eulenstein ได้เสนออัลกอริทึมแรกที่สามารถแก้ปัญหาการคัดลอกหลายยีนนี้ได้ ซึ่งอัลกอริทึมนั้นใช้เวลาทำงานเป็นกำลังสองของขนาดข้อมูลเข้า ในวิทยานิพนธ์ฉบับนี้เราได้เสนออัลกอริทึมใหม่สำหรับปัญหาการคัดลอกหลายยีน ซึ่งเป็นอัลกอริทึมแรกที่ใช้เวลาทำงานเชิงเส้นบนขนาดข้อมูลเข้า และยังได้เสนออัลกอริทึมที่ใช้เวลาทำงานใกล้เคียงเชิงเส้น แต่นำไปใช้งานจริงได้ง่ายกว่า ซึ่งเมื่อทำการทดลอง ได้ผลว่าอัลกอริทึมที่ใช้เวลาใกล้เคียงเชิงเส้นนี้ยังทำงานได้เร็วกว่าอัลกอริทึมเดิมอย่างน้อย 2 เท่า

_____ _____ ___ / ___ / ____

ลายมือชื่อนิสิต ลายมือชื่ออาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

Vacharapat Mettanant 2009: A Linear – Time Algorithm for the Multiple Gene Duplication Problem. Master of Engineering (Computer Engineering), Major Field: Computer Engineering, Department of Computer Engineering. Thesis Advisor: Assistant Professor Jittat Fackcheroenphol, Ph.D. 36 pages.

Reconstructing the Tree of Life is one of the most important research goals in evolutionary biology. In this thesis, we consider one of the issues regarding inconsistency among the trees reconstructed using sequences from different genes, called gene trees. This inconsistency is often a result of gene duplication. Given a species tree and a set of gene trees, the Multiple Gene Duplication Problem as formulated by Guigo, Muchnik, and Smith asks for an assignment of gene duplication events on the species tree node to minimize the number of duplication episodes. Recently, Bansal and Eulenstein give the first efficient algorithm for solving that problem. Their algorithm runs in quadratic time. In this thesis we give the first linear-time algorithm for the problem. Our algorithm can be seen as a faster implementation of Bansal and Eulenstein's method. We also present a more practical algorithm that runs in near-linear time. Experiments on synthetic data show a very good improvement on the running time (at least twice faster).

_____ _____ ___ / ___ / ____

Student's signature Thesis Advisor's signature

กิตติกรรมประกาศ

วิทยานิพนธ์ อัลกอริทึมแบบเวลาเชิงเส้นสำหรับปัญหาการคัดลอกหลายยืน สำเร็จลุล่วงไปได้ด้วยดี ข้าพเจ้าขอขอบพระคุณ ผู้ช่วยศาสตราจารย์จิตร์ทัศน์ ฝักเจริญผล อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก ผู้ให้แนวทางการวิจัยและให้คำแนะนำที่เป็นประโยชน์มากมาย ทั้งในด้านการทำงานวิจัย และในด้านการดำเนินชีวิต ผู้ช่วยศาสตราจารย์พีรวัฒน์ วัฒนพงศ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่ให้ความรู้พื้นฐานเกี่ยวกับการวิจัยด้านชีววิทยาเชิงคำนวณ และช่วยให้แนวคิดในมุมมองที่นอกเหนือจากการแก้ปัญหาด้วยคอมพิวเตอร์ ซึ่งทำให้ทราบถึงความสำคัญของแต่ละปัญหา และการใช้งานอย่างแท้จริง

นอกจากนี้ ข้าพเจ้าขอขอบคุณเพื่อนๆนิสิตปริญญาโท พี่ๆสมาชิกกลุ่มวิจัยเชิงทฤษฎีทุกท่าน ที่ได้คอยช่วยเหลือและแลกเปลี่ยนความคิด ความรู้ใหม่ๆ จนทำให้ข้าพเจ้าสามารถทำงานวิจัยนี้ได้สำเร็จ ขอบคุณเจ้าหน้าที่โครงการบัณฑิตศึกษา วิศวกรรมคอมพิวเตอร์ ที่คอยอำนวยความสะดวกและช่วยประสานงานในการดำเนินการต่างๆ ขอบคุณนภาวดี โรจนธรรม สำหรับการช่วยเหลือด้านเอกสาร ขอบคุณเพื่อนสมัยโรงเรียน และเพื่อนปริญญาตรีที่คอยให้กำลังใจอย่างเสมอ และสุดท้ายนี้ ขอขอบคุณบิดามารดา ที่คอยสนับสนุนและให้กำลังใจตลอดมา

วัชรพัฐ เมตตานันท

มีนาคม 2552

สารบัญ

หน้า

สารบัญตาราง

สารบัญภาพ

อัลกอริทึมแบบเวลาเชิงเส้นสำหรับปัญหาการคัดลอกหลายยีน

A Linear-Time Algorithm for the Multiple Gene Duplication Problem

คำนำ

ในการศึกษาวิจัยด้าน *ชีวสารสนเทศ* (bioinformatics) หรือ *ชีววิทยาเชิงคำนวณ* (computational biology) ปัญหาที่สำคัญที่สุดปัญหาหนึ่งก็คือ การหาลำดับวิวัฒนาการของสิ่งมีชีวิตแต่ละชนิด โดยความสัมพันธ์ระหว่างสิ่งมีชีวิตชนิดต่างๆสามารถแสดงให้อยู่ในรูปแบบของต้นไม้ที่เรียกว่า *ต้นไม้วิวัฒนาการ* (phylogenetic tree)ในสมัยแรกๆ การสร้างต้นไม้วิวัฒนาการนี้ จะพิจารณาจากความคล้ายคลึงของลักษณะทางกายภาพของสิ่งมีชีวิต ซึ่งไม่สามารถวัดความน่าเชื่อถือได้ และความซับซ้อนของสิ่งมีชีวิตหลายๆชนิดก็ทำให้วิธีการดังกล่าวทำได้ยาก ปัจจุบันนี้การศึกษาด้านชีววิทยามีความก้าวหน้าขึ้นมาก ความรู้เกี่ยวกับหน่วยย่อยๆเช่นรหัสพันธุกรรมถูกนำมาใช้สร้างประโยชน์อย่างมากมาย จึงมีแนวคิดที่จะสร้างต้นไม้วิวัฒนาการโดยพิจารณาจากการเปลี่ยนแปลงในระดับรหัสพันธุกรรม ซึ่งมีความน่าเชื่อถือมากกว่าแบบเดิม รายละเอียดเพิ่มเติมสามารถดูได้จาก Moret *et al*. (2005)

จากแนวคิดดังกล่าว เราสามารถพิจารณาความคล้ายคลึงระหว่างสิ่งมีชีวิตได้โดยดูจาก *หน่วยพันธุกรรม* หรือ *ยีน* (gene) ที่ทำหน้าที่เดียวกันของสิ่งมีชีวิตแต่ละชนิด ตามข้อสมมติที่ว่าหากสิ่งมีชีวิตมียีนคล้ายกันมาก ก็น่าจะแยกจากกันเป็นคนละชนิดเมื่อไม่นานมานี้ เช่นหากพิจารณายีนที่ทำหน้าที่บอกลักษณะของผิวหนังของ ม้า, ยีราฟ, และแมวน้ำ แล้วได้ผลว่า ยีนดังกล่าวของม้ากับยีราฟเป็นคู่ที่มีความคล้ายคลึงกันมากที่สุด แสดงว่า จากตอนแรกที่เป็นสิ่งมีชีวิตชนิดเดียวกันหมด เมื่อเวลาผ่านไป แมวน้ำน่าจะแยกไปเป็นสิ่งมีชีวิตต่างชนิดก่อน โดยม้าและยีราฟยังเป็นชนิดเดียวกัน และเมื่อเวลาผ่านไปอีก จึงมาแยกจากกันทีหลัง ทั้งนี้การวัดความคล้ายคลึงระหว่างยีนทำได้โดยใช้วิธีการที่เกี่ยวกับสายอักขระต่างๆ ในปัจจุบันได้มีงานวิจัยมากมายเกี่ยวกับการอนุมานต้นไม้วิวัฒนาการ โดยใช้ข้อมูลจากยีนดังที่กล่าวไว้ (Daskalakis *et al*., 2006; Mihaescu *et al*., 2005; Swofford, 2003; Stamatakis *et al*., 2006; Goloboff *et al*., 2000)

การใช้ลักษณะยีนช่วยในการสร้างต้นไม้วิวัฒนาการนี้ มีปัญหาหลักคือ เมื่อเรานำยีนที่มีหน้าที่แตกต่างกันมาสร้างต้นไม้แสดงการเปลี่ยนแปลงของยีนซึ่งจะนำไปอนุมานเป็นต้นไม้

วิวัฒนาการต่อไปนั้น ปรากฏว่ายีนในแต่ละหน้าที่กลับให้ต้นไม้ดังกล่าวไม่เหมือนกัน (Field *et al*., 1988; Lake, 1987) ปรากฏการณ์นี้สามารถอธิบายได้ด้วยกระบวนการที่เรียกว่า *การคัดลอกยีน* (gene duplication) ซึ่งจะกล่าวโดยละเอียดอีกครั้งในหัวข้อถัดไป

โดยทั่วไปแล้ว การคัดลอกยีนนี้ถือว่าเป็นกระบวนการที่ผิดปรกติ จึงได้มีการคิดค้นอัลกอริทึมที่จะสร้างต้นไม้วิวัฒนาการซึ่งเกิดการคัดลอกยีนที่ใช้อธิบายต้นไม้ยีนทั้งหมดน้อยครั้งที่สุด (Guigo *et al*., 1996) ปัญหานี้ถูกพบว่าอยู่ในกลุ่มปัญหาเอ็นพีบริบูรณ์ (NP-complete) โดย Ma *et al*. (2001)

นอกจากนี้ยังมีแนวคิดที่ว่า การคัดลอกยีนอาจไม่ได้เป็นอิสระจากกัน โดยมองว่าการคัดลอกของยีนที่ทำหน้าที่ต่างกัน อาจเกิดจากการคัดลอกขนาดใหญ่ในสายพันธุกรรมครั้งเดียวกันก็ได้ ในเดือนกรกฎาคม 2008 นี้ Bansal และ Eulenstein ได้นิยาม *ปัญหาการคัดลอกหลายยีน* (the multiple gene duplication problem) และเสนออัลกอริทึมที่แก้ปัญหานี้อย่างมีประสิทธิภาพ อย่างไรก็ตาม อัลกอริทึมของเขายังใช้เวลาการทำงานเป็นกำลังสอง (quadratic time) จึงยังมีความต้องการอัลกอริทึมที่ให้ผลเร็วขึ้นเนื่องจากปัญหานี้ถือเป็นปัญหาย่อยส่วนหนึ่งในการสร้างต้นไม้วิวัฒนาการซึ่งมีความสำคัญมาก (Ma *et al*., 2001; Page, 1994)

ในงานวิจัยนี้ เราทำการเสนออัลกอริทึมใหม่สำหรับปัญหาการคัดลอกหลายยีนซึ่งใช้เวลาทำงานเป็นเชิงเส้น (linear time) อัลกอริทึมใหม่นี้สามารถมองเป็นการพัฒนาอัลกอริทึมเดิมของ Bansal และ Eulenstein ให้ทำงานได้เร็วขึ้น อย่างไรก็ตาม อัลกอริทึมแบบเวลาเชิงเส้นนี้นำไปใช้ในทางปฏิบัติได้ยาก เราจึงได้เสนอทางเลือกที่นำไปปฏิบัติจริงได้ง่ายขึ้นโดยใช้เวลาทำงานใกล้เคียงกับเวลาเชิงเส้น เมื่อทำการทดลองโดยใช้วิธีดังกล่าว ผลการทดลองแสดงให้เห็นว่าอัลกอริทึมแบบใช้เวลาใกล้เคียงเชิงเส้นยังทำงานได้เร็วกว่าอัลกอริทึมของ Bansal และ Eulenstein สองเท่าแม้มีข้อมูลเข้าขนาดเล็ก

วัตถุประสงค์

1. ศึกษาและวิเคราะห์คุณสมบัติของปัญหาการคัดลอกหลายยีน และอัลกอริทึมของ Bansal และ Eulenstein ที่ใช้แก้ปัญหานี้

2. ออกแบบอัลกอริทึมสำหรับปัญหาการคัดลอกหลายยีนใหม่ ให้ใช้เวลาทำงานเร็วกว่าอัลกอริทึมเดิมของ Bansal และ Eulenstein

3. พิสูจน์ความถูกต้อง และเวลาทำงานของอัลกอริทึมใหม่ ในเชิงทฤษฎี

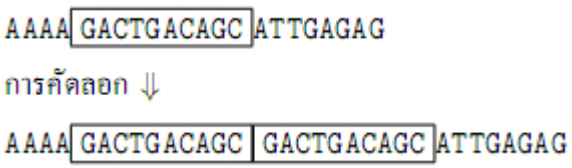
4. ทำการทดลองเปรียบเทียบผลลัพธ์ และเวลาที่ใช้ในการทำงานของอัลกอริทึมใหม่กับอัลกอริทึมเดิม

การตรวจเอกสาร

หนึ่งในปัญหาพื้นฐานทางด้านชีววิทยา คือการทำความเข้าใจการสืบทอดทางพันธุกรรม ในปี 1865 Gregor Johann Mendel ค้นพบแบบจำลองทางคณิตศาสตร์ที่ใช้ในการสืบทอด โดยมีหลักการคร่าวๆคือ จะมีหน่วยพื้นฐานที่บ่งบอกลักษณะทางกายภาพของสิ่งมีชีวิต และเมื่อมีการสืบทอด ลักษณะทางกายภาพนั้นๆของลูกหลานเกิดจากหน่วยพื้นฐานที่ได้รับมาจากรุ่นก่อน หน่วยพื้นฐานดังกล่าวนี้ถูกเรียกว่า หน่วยพันธุกรรม หรือ ยีน อย่างไรก็ตาม ลักษณะทางธรรมชาติของยีนก็เป็นที่สงสัยต่อมาเป็นเวลานาน จนกระทั่งในปี 1944 จึงมีการค้นพบว่า ยีนถูกสร้างขึ้นจากสายดี*เอ็นเอ* (DNA) โดยยีนแต่ละตัวจะเป็นลำดับย่อยของสายดีเอ็นเอที่บริเวณต่างๆ หลังจากนั้น ในปี 1953 James Dewey Watson และ Francis Harry Compton Crick จึงได้เสนอลักษณะโครงสร้างของดีเอ็นเอแบบเกลียวคู่ ที่ใช้กันมาจนถึงปัจจุบัน

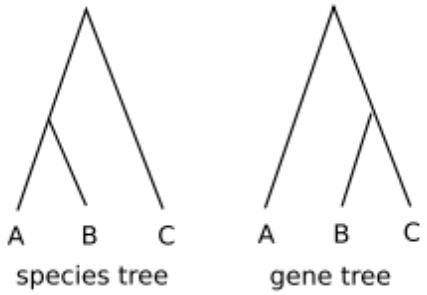
เนื่องจากยีนเป็นลำดับย่อยของสายดีเอ็นเอ ความแตกต่างระหว่างยีนแต่ละตัวจึงสามารถวัดได้ค่อนข้างชัดเจน และมีแนวคิดในการนำความสัมพันธ์ของยีนของสิ่งมีชีวิตต่างๆมาใช้เพื่ออนุมานความสัมพันธ์ของสิ่งมีชีวิตเหล่านั้น ซึ่งก็คือการสร้างต้นไม้วิวัฒนาการนั่นเอง ซึ่งในงานวัจัยนี้ เราจะเรียกต้นไม้วิวัฒนาการของสิ่งมีชีวิตว่า *ต้นไม้สปีชีส์* (species tree) โดยความสัมพันธ์ของยีนสามารถแสดงในรูปแบบต้นไม้ได้เช่นกัน โดยเรียกว่า *ต้นไม้ยีน* (gene tree) ปัญหาที่เกิดขึ้นต่อมาก็คือ ต้นไม้ยีนที่สร้างจากยีนในหน้าที่ต่างๆกัน อาจจะให้ความสัมพันธ์ระหว่างสิ่งมีชีวิตแต่ละชนิดไม่เหมือนกัน นั่นแสดงว่า จะต้องมีต้นไม้ยีนบางต้นมีลักษณะแตกต่างจากต้นไม้สปีชีส์

ความขัดแย้งนี้สามารถอธิบายได้ ด้วยกระบวนการที่เรียกว่า การคัดลอกยีน ซึ่งเกิดจากการที่บริเวณหนึ่งในสายดีเอ็นเอที่ครอบคลุมยีนตัวนั้นเกิดการคัดลอกตัวเอง ดังภาพที่ 1 การคัดลอกยีนสามารถเกิดขึ้นจากความผิดพลาดในกระบวนการทางพันธุกรรม ซึ่งหลังจากเกิดการคัดลองยีนแล้ว ยีนที่เป็นผลจากการคัดลอกทั้งคู่อาจเกิดการเปลี่ยนแปลงต่อไปแตกต่างกัน



ภาพที่ 1 ตัวอย่างแสดงการคัดลอกยีน

เพื่อความสะดวกในการอธิบายผลจากการคัดลอกยีนที่ทำให้ต้นไม้ยีนมีลักษณะแตกต่างไปจากต้นไม้สปีชีส์ ให้พิจารณาตัวอย่างในภาพที่ 2 เพื่อความชัดเจน ในต้นไม้ยีนเราจะกำกับยีนแต่ละตัวด้วยสปีชีส์ที่ยีนนั้นๆปรากฏอยู่ จะเห็นว่าหากพิจารณายีนที่สนใจ สปีชีส์ B จะดูเหมือนว่ามีลักษณะใกล้เคียงกับสปีชีส์ C มากกว่าสปีชีส์ A

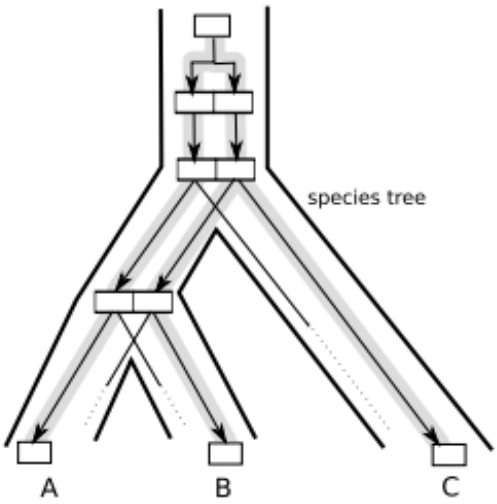


ภาพที่ 2 ต้นไม้ยีนที่ต่างจากต้นไม้สปีชีส์

ภาพที่ 3 อธิบายขั้นตอนที่ทำให้เกิดความขัดแย้งนี้ พิจารณายีน g ซึ่งแสดงด้วยรูปสี่เหลี่ยมในภาพ ตอนแรกที่รากของต้นไม้สปีชีส์เกิดการคัดลอกยีนขึ้น กลายเป็นยีนสองตัวที่เหมือนกันในลำดับถัดมา ซึ่งยีนทั้งคู่ก็จะเกิดการเปลี่ยนแปลงแตกต่างกันต่อๆไป ต่อมา สปีชีส์ C แยกตัวออกไปจาก A และ B ก่อน ระหว่างเส้นทางที่วิวัฒนาการมาเป็นสปีชีส์ C นั้น ได้ทำยีนตัวหน้าหายไป ต่อมาสปีชีส์ A และ B แยกจากกัน โดยสปีชีส์ A ทำยีนตัวหลังหายไป ในขณะที่สปีชีส์ B ทำยีนตัวหน้าหายไป จากเหตุการณ์ดังกล่าว สปีชีส์ B และ C มียีนที่มาจากผลการคัดลอกตัวเดียวกัน จึงมีลักษณะใกล้เคียงกันมากกว่ายีนในสปีชีส์ A ต้นไม้ยีนที่เป็นผลจากเหตุการณ์นี้แสดงด้วยแถบแรเงาในภาพ

นิยามเบื้องต้น

ในงานวิจัยนี้ โครงสร้างแบบจำลองหลักๆที่เราสนใจก็คือ *ต้นไม้ทวิภาคแบบเต็ม* (full binary tree) ซึ่งเป็น *ต้นไม้ทวิภาค* (binary tree) ที่ *โหนดภายใน* (internal node) มีจำนวนโหนดลูกเท่ากับสองเสมอ สมมติให้ T เป็นต้นไม้ทวิภาคแบบเต็มต้นหนึ่ง เราจะใช้สัญลักษณ์ $V(T)$, $E(T)$, $L(T)$, และ $r(T)$ แทนเซตของโหนด, เซตของเส้นเชื่อม, เซตของ *ใบ* (leaf), และ *โหนดราก* (root node) ของ T ตามลำดับ สำหรับโหนด u ใดๆใน T เราจะให้ T_u แทนต้นไม้ย่อยของ T ที่มี u เป็นโหนดราก และสำหรับต้นไม้ T ใดๆ เราจะให้ $h(T)$ แทนความสูงของ T โดยวัดจากจำนวนโหนดบนเส้นทางที่ยาวที่สุดจากโหนดรางไปยังโหนดใบ



ภาพที่ 3 การคัดลอกยีนบนต้นไม้สปีชีส์

งานวิจัยนี้สนใจต้นไม้อยู่สองชนิด คือ ต้นไม้สปีชีส์ และ ต้นไม้ยีน โดยต้นไม้สปีชีส์คือต้นไม้ทวิภาคแบบเต็ม ที่แต่ละโหนดใบกำกับไว้ด้วยสปีชีส์ หรือชนิดของสิ่งมีชีวิตที่แตกต่างกัน แสดงความสัมพันธ์ในเชิงวิวัฒนาการระหว่างสปีชีส์เหล่านั้น ต้นไม้ยีนก็เป็นต้นไม้ทวิภาพแบบเต็ม ซึ่งแสดงความสัมพันธ์ในเชิงการเปลี่ยนแปลงของยีนในกลุ่มเดียวกันของสปีชีส์ที่สนใจ ดังนั้น ที่โหนดใบของต้นไม้ยีนจะกำกับไว้ด้วยยีนที่นำมาพิจารณา แต่เพื่อความสะดวกในการวิเคราะห์ต่อไป เรานิยมกำกับที่โหนดใบของต้นไม้ยีนด้วยสปีชีส์ที่ยีนนั้นๆปรากฏอยู่ จากตรงนี้ สังเกตว่าในต้นไม้ยีนอาจมีโหนดใบบางคู่ที่กำกับด้วยสปีชีส์เดียวกัน เนื่องจากยีนทั้งคู่นั้นถูกพบในสิ่งมีชีวิตชนิดเดียวกัน ในขณะที่ในต้นไม้สปีชีส์ แต่ละโหนดใบจะกำกับไว้ด้วยสปีชีส์ที่แตกต่างกันทั้งหมด เนื่องจากเราใช้ต้นไม้ทวิภาพแสดงความสัมพันธ์ในเชิงวิวัฒนาการ ดังนั้น โหนดภายในจะแสดงถึงสปีชีส์ที่เป็นบรรพบุรุษสำหรับต้นไม้สปีชีส์ และแทนยีนต้นแบบสำหรับต้นไม้ยีน

สมมติว่ามีต้นไม้ยีน G และต้นไม้สปีชีส์ S เราสามารถทำการเปรียบเทียบต้นไม้ทั้งคู่ว่ายีนแต่ละตัวใน G ควรอยู่ในสปีชีส์ใดใน S โดยการหาฟังก์ชั่นจาก $V(G)$ ไปยัง $V(S)$ โดยที่แต่ละโหนด $u \in V(G)$ จะถูก *ส่ง* (map) ไปยังสปีชีส์ที่สอดคล้องใน S โดยที่ หาก u เป็นโหนดใบ จะถูกส่งไปยังโหนดใบใน S ที่กำกับไว้ด้วยสปีชีส์เดียวกัน และหาก u เป็นโหนดภายใน สังเกตว่า u เป็นต้นแบบของยีนที่ปรากฏในสปีชีส์ที่เป็นโหนดใบทั้งหมดใน G_u ดังนั้น สปีชีส์ v ที่เป็นเจ้าของยีน u ก็ควรเป็นบรรพบุรุษของทุกๆสปีชีส์ใน $L(G_u)$ ด้วย แนวคิดนี้ทำให้เราสามารถนิยาม *การ*

ส่ง (mapping) ที่เรียกว่า *บรรพบุรุษร่วมที่เล็กที่สุด* (least common ancestor [LCA]) ได้ดังนิยามต่อไปนี้

ให้ G และ S เป็นต้นไม้ยีนและต้นไม้สปีชีส์ตามลำดับ ฟังก์ชัน $LCA: V(G) \rightarrow V(S)$ เป็นฟังก์ชันตามนิยามต่อไปนี้

นิยามที่ 1 *สำหรับโหนด* $u \in V(G)$ $LCA(u)$ *คือโหนด* $v \in V(S)$ *ที่มีระยะห่างจาก* $r(S)$ *มากที่สุดที่* $L(G_u) \subseteq L(S_v)$

จากนิยาม สำหรับโหนด u ในต้นไม้ยีน $LCA(u)$ อาจจะเท่ากับ $LCA(P(u))$ หรือเป็นลูกหลานของ $LCA(P(u))$ เมื่อ $P(u)$ เป็น *โหนดแม่* (parent node) ของ u ถ้า $LCA(u) = LCA(P(u))$ แสดงว่ายีนต้นแบบที่ $P(u)$ และยีนที่เป็นผลจากการเปลี่ยนแปลงต้นแบบนั้นที่โหนด u ปรากฏอยู่ในสปีชีส์เดียวกัน แสดงว่ายีนที่โหนด u เป็นผลจากการคัดลอกยีนของยีนที่โหนด $P(u)$ จากตรงนี้ เราสามารถให้นิยามของยีนที่เกิดการคัดลอกได้ดังนี้

นิยามที่ 2 *สำหรับโหนด* $u \in V(G)$ *เราจะเรียก* u *ว่าเป็น* โหนดคัดลอก *(duplication node) ถ้า* u *มีโหนดลูก* u' *บางโหนดที่* $LCA(u') = LCA(u)$

เพื่อความสะดวกต่อไป เราจะใช้ $Dup(G, S)$ แทนเซตของโหนดคัดลอกของต้นไม้ยีน G เมื่อเทียบกับต้นไม้สปีชีส์ S

นิยามเบื้องต้นที่กล่าวมานี้ ถูกนำไปใช้ในการสร้างแบบจำลองทางคณิตศาสตร์ สำหรับปัญหาที่เกี่ยวกับการสร้างต้นไม้สปีชีส์และการคัดลอกยีนอย่างมากมาย ในที่นี้เราจะกล่าวถึงอยู่สองปัญหา ได้แก่ ปัญหาต้นไม้สปีชีส์ที่ดีที่สุด และปัญหาการคัดลอกหลายยีน

ปัญหาต้นไม้สปีชีส์ที่ดีที่สุด

ประเด็นที่สำคัญที่สุดในงานวิจัยด้านนี้ก็คือ การนำข้อมูลที่มีอยู่เกี่ยวกับยีนของสิ่งมีชีวิต ได้แก่ต้นไม้ยีน มาใช้เพื่ออนุมานความสัมพันธ์ในเชิงวิวัฒนาการของสิ่งมีชีวิตเหล่านั้น ในที่นี้ความสัมพันธ์ดังกล่าวก็คือต้นไม้สปีชีส์นั่นเอง ปัญหาจึงอยู่ในรูปของการหาต้นไม้สปีชีส์ โดยสมมติว่าเรามีต้นไม้ยีนอยู่จำนวนหนึ่ง ซึ่งมีการสร้างแบบจำลองสำหรับปัญหานี้อยู่ค่อนข้างมาก

แบบจำลองหนึ่งที่น่าสนใจใช้สมมติฐานที่ว่าการคัดลอกยีนไม่ควรเกิดขึ้นเป็นจำนวนมาก เนื่องจากเราถือเป็นกระบวนการที่ผิดปรกติ แบบจำลองนี้จึงพยายามสร้างต้นไม้สปีชีส์ที่ทำให้เกิดการคัดลอกยีนรวมแล้วน้อยที่สุด ซึ่งมีนิยามแบบเป็นทางการดังนี้

นิยามที่ 3 *ปัญหาต้นไม้สปีชีส์ที่ดีที่สุดโดยอิงการคัดลอกยีนคือ เมื่อมีต้นไม้ยีน* $G_1, G_2, \ldots, G_k$ *จงหาต้นไม้สปีชีส์* S *ที่ทำให้*

$$\sum_{i=1}^{k} |Dup(G_i, S)|$$

มีค่าน้อยที่สุด

ปัญหาต้นไม้สปีชีส์ที่ดีที่สุดโดยอิงการคัดลอกยีนนี้ ถูกพิสูจน์แล้วว่าอยู่ในกลุ่มปัญหาเอ็นพีบริบูรณ์ และในปัจจุบันยังไม่มีการค้นพบ *อัลกอริทึมแบบประมาณ* (approximation algorithm) สำหรับปัญหานี้

จากตัวอย่างปัญหา สังเกตว่าเรานับจำนวนการคัดลอกยีนโดยนับที่จำนวนโหนดคัดลอกทั้งหมด ซึ่งถือว่าการคัดลอกยีนแต่ละครั้งนั้นเป็นอิสระจากกัน แต่ในความเป็นจริง การคัดลอกยีนหลายๆตำแหน่งอาจเกิดจากคัดลอกตัวเองของสายดีเอ็นเอบริเวณกว้างที่ครอบคลุมยีนเหล่านั้นอยู่พอดี ทำให้มียีนที่เกิดขึ้นใหม่เป็นจำนวนมาก ทั้งๆที่เกิดการคัดลอกขึ้นเพียงครั้งเดียว จากตรงนี้จึงได้มีการพยายามสร้างแบบจำลองใหม่ที่นับการคัดลอกขนาดใหญ่นี้ให้น้อยที่สุด แต่การจะนับว่าโหนดคัดลอกโหนดใดสามารถเกิดจากการคัดลอกขนาดใหญ่ครั้งเดียวกันได้นี้ค่อนข้างซับซ้อน จึงทำให้เกิดปัญหาการคัดลอกหลายยีนขึ้น

ปัญหาการคัดลอกหลายยีน

ในปัญหานี้ เราจะกลับมาพิจารณาถึงความสัมพันธ์ระหว่างต้นไม้ยีนและต้นไม้สปีชีส์ใหม่ เราทราบแล้วว่า หากต้นไม้ยีนมีลักษณะแตกต่างจากต้นไม้สปีชีส์ แสดงว่าจะต้องมียีนที่โหนดบางโหนดในต้นไม้ยีนมีการคัดลอกเกิดขึ้น ซึ่งเราสามารถตรวจสอบได้ว่ายีนตัวใดบ้างที่น่าจะเกิดการคัดลอก โดยใช้นิยามที่ 2 แต่ตอนนี้เราคาดว่าการคัดลอกยีนที่ตำแหน่งต่างๆในต้นไม้ยีน อาจเกิดจาก

การคัดลอกขนาดใหญ่ครั้งเดียวกัน เราจึงมีความพยายามที่จะนำการคัดลอกยีนที่สามารถเกิดด้วยกันได้มารวมกันให้ได้มากที่สุด

แนวคิดในการนับการคัดลอกหลายยีน หรือการคัดลอกสายดีเอ็นเอบริเวณกว้างนั้น มีหลักการคร่าวๆคือ หากเราทราบว่ามียีนกลุ่มนึงเกิดการคัดลอกขึ้นขณะที่อยู่ในสปีชีส์ v เดียวกัน เราจะถือว่าการคัดลอกทั้งหมดนั้นเกิดขึ้นในครั้งเดียวกัน แต่หากมียีนคู่หนึ่งที่เกิดการคัดลอกในขณะที่อยู่ในสปีชีส์ v เหมือนกันโดยยีนตัวหนึ่งเป็นผลจากการเปลี่ยนแปลงของยีนอีกตัวหนึ่ง หรือก็คือโหนดของยีนตัวหนึ่งเป็นลูกของโหนดของยีนอีกตัวหนึ่ง จะได้ว่า การคัดลอกของยีนทั้งคู่นี้ ไม่สามารถเกิดจากการคัดลอกครั้งเดียวกันได้ ดังนั้น เราจึงสามารถนับการคัดลอกหลายยีนได้ โดยพิจารณาจากความสัมพันธ์ของยีนที่เกิดการคัดลอกแต่ละตัว

สำหรับนิยามอย่างเป็นทางการ สมมติให้มีต้นไม้ยีน G และต้นไม้สปีชีส์ S เราจะเริ่มจากการนิยามช่วงที่การคัดลอกยีนใดๆใน G สามารถเกิดขึ้นได้เมื่อเทียบกับ S สำหรับโหนด u และ v ใน S ที่ u เป็น *ผู้สืบเชื้อสาย* (descendant) ของ v เราจะให้ $Mid(u, v)$ แทนเซตของโหนดใน $V(S)$ ที่เป็นบรรพบุรุษของ u และเป็นผู้สืบเชื้อสายของ v

นิยามที่ 4 *สำหรับโหนด* $u \in Dup(G, S)$ *ช่วง* $I(u)$ *เป็นเซตของโหนดใน* S *ดังนี้*

$$I(u) = \begin{cases} \{LCA(u)\} \cup \{v \in V(S) | v \text{ เป็นบรรพบุรุษของ } u\} & \text{เมื่อ } u \text{ เป็นโหนดราก} \\ \{LCA(u)\} \cup \{Mid\left(LCA(u), LCA(P(u))\right)\} & \text{กรณีอื่นๆ} \end{cases}$$

และสำหรับโหนด $u \notin Dup(G, S)$ $I(u) = \{LCA(u)\}$

เราจะเรียกฟังก์ชันการส่ง M จาก $V(G)$ ไปยัง $V(S)$ ว่าเป็นการส่งที่ *สมเหตุสมผล* (valid) ก็ต่อเมื่อสำหรับแต่ละโหนด $u \in V(G)$ $M(u) \in I(u)$

ในปัญหาจริงๆนั้น อาจมีต้นไม้ยีนมากกว่าหนึ่งต้นก็ได้ ซึ่งก็ได้มาจากความสัมพันธ์ระหว่างยีนในหน้าที่ต่างๆกัน ซึ่งการคัดลอกยีนที่เกิดขึ้นบนต้นไม้ยีนคนละต้นกันก็สามารถเกิดขึ้นจากการคัดลอกในบริเวณกว้างครั้งเดียวกันได้เช่นกัน สมมติให้มีต้นไม้ยีนอยู่ k ต้น เราอาจจะเรียกฟังก์ชันการส่ง M ว่าเป็นฟังก์ชันจากเซตของโหนดของต้นไม้ยีนทุกต้นไปยังเซตของโหนดในต้นไม้สปีชีส์ โดยหมายถึงยูเนียนของฟังก์ชันดังกล่าวจากต้นไม้ยีนแต่ละต้นนั่นเอง

สำหรับฟังก์ชันการส่ง M และโหนด v ใดๆใน $V(S)$ เราจะให้ $M^{-1}(v)$ แทนเซต $\{u|M(u) = v\}$ ซึ่งก็คือเซตของโหนดที่แทนยีนทั้งหมดที่ถูกส่งมายังสปีชีส์ที่โหนด v ตามการส่ง M และจะให้ $F_{M^{-1}}(v)$ เป็น ป่า (forest) ใน $\bigcup_{i=1}^{k} G_i$ ที่สร้างจากโหนดใน $M^{-1}(v)$ สำหรับป่า F ใดๆ เราจะให้ความสูง $h(F)$ เป็นความสูงของต้นไม้ที่สูงที่สุดใน F จากที่กล่าวมานี้ เราสามารถนิยามปัญหาการคัดลอกหลายยีนได้ดังนี้

นิยามที่ 5 *ให้ต้นไม้ยีน k ต้น $G_1, G_2, \ldots, G_k$ และต้นไม้สปีชีส์ S ปัญหาการคัดลอกหลายยีนคือการหาฟังก์ชันการส่ง M ที่สมเหตุสมผลโดยที่*

$$\sum_{v \in V(S)} h(F_{M^{-1}}(v))$$

มีค่าน้อยที่สุด

ปัญหาการคัดลอกหลายยีนนี้สามารถหาคำตอบได้อย่างมีประสิทธิภาพ อัลกอริทึมแรกที่มีการเสนอคืออัลกอริทึมของ Bansal และ Eulenstein ซึ่งจะกล่าวในหัวข้อถัดไป

อัลกอริทึมของ Bansal และ Eulenstein

ในหัวข้อนี้ เราจะอธิบายอัลกอริทึมของ Bansal และ Eulenstein พร้อมทั้งแสดงการวิเคราะห์ความถูกต้องและเวลาการทำงานของอัลกอริทึมนี้ ตามที่พวกเขาแสดงไว้ในบทความวิชาการ เราจะเริ่มต้นด้วยการนิยามเพิ่มเติมเล็กน้อย

สำหรับฟังก์ชันการส่ง M และโหนด $v \in V(S)$ เราจะเรียกโหนด $u \in M^{-1}(v)$ ว่าเป็น *โหนดนำ* (leading node) ถ้า u เป็นโหนดรากของต้นไม้ที่สูงที่สุดต้นหนึ่งใน $F_{M^{-1}}(v)$ และถ้าโหนดแม่ของ v เป็นสมาชิกใน $I(u)$ เราจะเรียก u ว่าเป็น *โหนดอิสระ* (free node)

ขั้นตอนการทำงาน

อัลกอริทึมของ Bansal และ Eulenstein แสดงอยู่ในภาพที่ 4 การทำงานเริ่มต้นด้วยการหาการส่ง LCA หลังจากนั้นจะทำการพิจารณาแต่ละโหนด v ในต้นไม้สปีชีส์ S ตามลำดับแบบล่าง

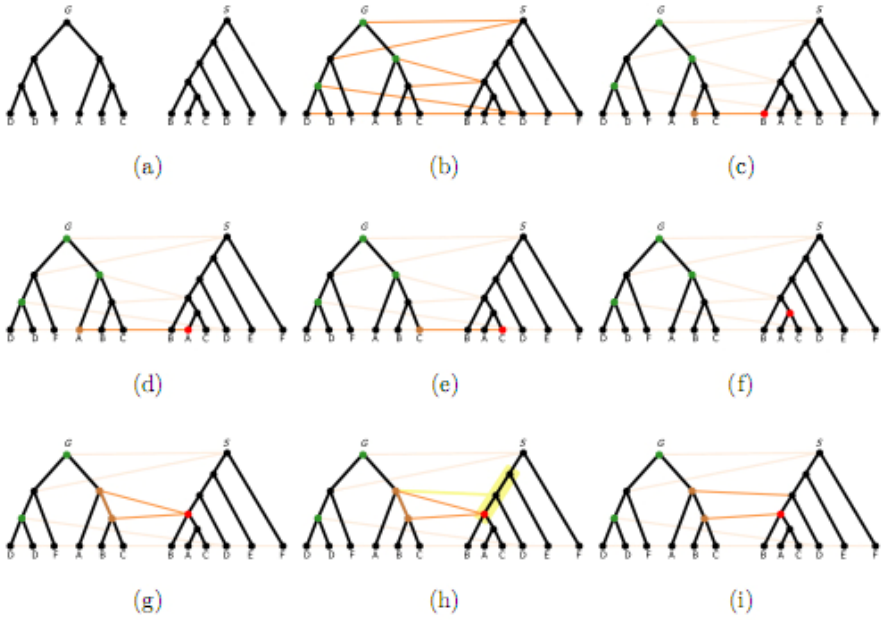
ขึ้นบน (post order) หากโหนดนำทั้งหมดใน $M^{-1}(v)$ เป็นโหนดอิสระ ก็จะขยับการส่งจากโหนดนำเหล่านั้นไปยังโหนดแม่ของ v ทำเช่นนี้ไปเรื่อยๆจนพิจารณาโหนดใน S ครบก็จะได้การส่งที่เป็นคำตอบ ตัวอย่างการทำงานของอัลกอริทึมนี้แสดงอยู่ในภาพที่ 5

The Bansal-Eulenstein algorithm

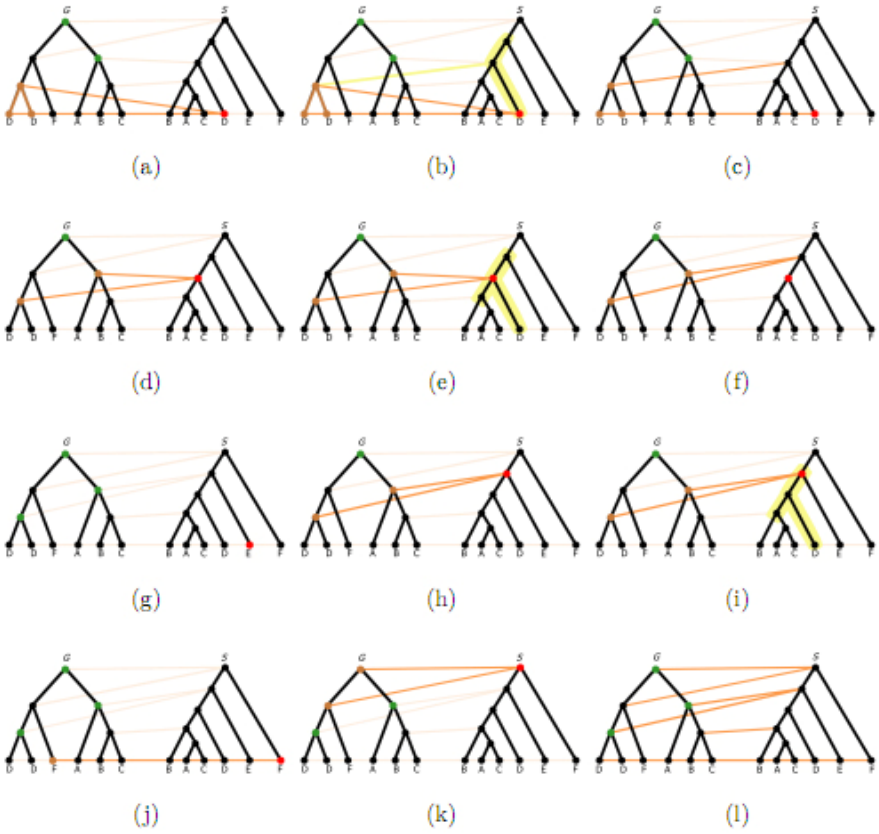
input: gene trees $G_1, ..., G_k$ and a species tree S

1: Initialize M to be the LCA mapping from each G_i to S.
2: Compute $I(u)$ for each node u in the gene trees.
3: **for** each node $v \in S$ in a post-order traversal **do**
4: **if** $|M^{-1}(v)| > 0$ **then**
5: **if** all leading nodes in $M^{-1}(v)$ are free **then**
6: Update M by moving the maps of all these leading nodes to $P(v)$.
7: **end if**
8: **end if**
9: **end for**
10: Return M.

ภาพที่ 4 อัลกอริทึมของ Bansal และ Eulenstein



ภาพที่ 5 ตัวอย่างการทำงานของอัลกอริทึมของ Bansal และ Eulenstein



ภาพที่ 5 (ต่อ)

ความถูกต้องของอัลกอริทึม

สำหรับในการพิสูจน์ความถูกต้องของอัลกอริทึม เราจะให้ M_{BE} เป็นการส่งที่ได้จากอัลกอริทึม และให้ h_{opt} เป็นค่าของ $\sum_{v \in V(S)} h(F_{M^{-1}}(v))$ ที่น้อยที่สุด เราจะแสดงว่า M_{BE} เป็นการส่งที่สมเหตุสมผล และ

$$\sum_{v \in F(S)} h\left(F_{M_{BE}^{-1}}(v)\right) = h_{opt}$$

ทฤษฎีบทย่อยที่ 1 M_{BE} *เป็นการส่งที่สมเหตุสมผล*

พิสูจน์ อัลกอริทึมของ Bansal และ Eulenstein เริ่มต้นจากการให้ $M = LCA$ ซึ่งเป็นการส่งที่สมเหตุสมผลจากนิยาม และในแต่ละรอบการทำงานของวนรอบ for การส่ง M ก็อาจถูกปรับเปลี่ยน

ได้ แต่อัลกอริทึมจะทำการปรับการส่งจากโหนดในต้นไม้ยืนเมื่อโหนดนั้นเป็นโหนดอิสระเท่านั้น ซึ่งผลที่ได้ก็ยังคงเป็นการส่งที่สมเหตุสมผลอยู่ ดังนั้นการส่งทั้งหมดที่ได้เป็นผลจากอัลกอริทึม ซึ่งรวมทั้ง M_{BE} ด้วย จะเป็นการส่งที่สมเหตุสมผลทั้งหมด ■

ทฤษฎีบทย่อยที่ 2 *ให้* M *เป็นการส่งที่สมเหตุสมผลใดๆ และ* v *เป็นโหนดใน* S

1. *ถ้า* $LCA^{-1}(v) \neq \emptyset$ *จะได้ว่า*
$$h\left(F_{LCA^{-1}}(v)\right) - 1 \leq h(F_{M^{-1}}(v)) \leq h(F_{LCA^{-1}}(v))$$
2. *ถ้า* $LCA^{-1}(v) = \emptyset$ *จะได้ว่า* $0 \leq h(F_{M^{-1}}(v)) \leq 1$

พิสูจน์ ในส่วนแรก เมื่อ $LCA^{-1}(v) \neq \emptyset$ สมมติให้ A เป็นเซตของโหนดที่อยู่ใน $LCA^{-1}(v)$ แต่ไม่อยู่ใน $M^{-1}(v)$ และให้ B เป็นเซตของโหนดที่อยู่ใน $M^{-1}(v)$ แต่ไม่อยู่ใน $LCA^{-1}(v)$ จากนิยาม สังเกตว่าทุกๆโหนดใน A จะต้องเป็นโหนดรากของต้นไม้ใน $F_{LCA^{-1}}(v)$ โดยอาจจะเป็นโหนดนำหรือไม่ก็ได้ หากเราเปลี่ยนการส่งของโหนดนำทั้งหมดไป จะทำให้ความสูง $h(F_{LCA^{-1}}(v))$ มีค่าลดลง 1 พอดี จากตรงนี้จึงได้ว่า

$$h\left(F_{LCA^{-1}}(v)\right) - 1 \leq h(F_{M^{-1}}(v))$$

ในส่วนที่สอง เมื่อ $LCA^{-1}(v) = \emptyset$ จะได้ว่า $B = M^{-1}(v)$ ด้วยเหตุผลแบบเดียวกับด้านบน เราสามารถบอกได้ว่าทุกๆโหนดใน B จะเป็นต้นไม้หนึ่งโหนดทั้งหมด จึงได้ว่า

$$h(F_{M^{-1}}(v)) \leq 1$$

ตรงตามทฤษฎีบทย่อย ■

สำหรับพิสูจน์ความถูกต้องของอัลกอริทึมของ Bansal และ Eulenstein เราจำเป็นต้องพิสูจน์ทฤษฎีบทย่อยสามข้อต่อไปนี้ก่อน

ทฤษฎีบทย่อยที่ 3 *สำหรับฟังก์ชันการส่ง* M *ที่สมเหตุสมผล และ* $v \in V(S)$ *ถ้า* $h\left(F_{M_{BE}^{-1}}(v)\right) > h(F_{M^{-1}}(v))$ *จะได้ว่า* $h\left(F_{M_{BE}^{-1}}(v)\right) = 1$

พิสูจน์ เราจะแบ่งเป็นสองกรณี ได้แก่ เมื่อ $LCA^{-1}(v) = \emptyset$ และเมื่อ $LCA^{-1}(v) \neq \emptyset$ ในกรณีแรก $LCA^{-1}(v) = \emptyset$ จากทฤษฎีบทย่อยที่ 2 ส่วนที่ 2 จะได้ว่า $h\left(F_{M^{-1}}(v)\right) = 0$ และ $h\left(F_{M_{BE}^{-1}}(v)\right) = 1$ ในกรณีที่ $LCA^{-1}(v) \neq \emptyset$ ถ้า $h(F_{LCA^{-1}}(v)) < 2$ ทฤษฎีบทย่อยที่ 2 ส่วนแรกจะให้ผลตามต้องการทันที ดังนั้น เราจะสมมติให้ $h(F_{LCA^{-1}}(v)) \geq 2$ ซึ่งจากทฤษฎีบทย่อยที่ 2 ส่วนแรก เราจะรู้ได้ว่า

$$h\left(F_{M^{-1}}(v)\right) = h\left(F_{LCA^{-1}}(v)\right) - 1$$

และ

$$h\left(F_{M_{BE}^{-1}}(v)\right) = h(F_{LCA^{-1}}(v))$$

ให้ A เป็นเซตของโหนดที่อยู่ใน $LCA^{-1}(v)$ แต่ไม่อยู่ใน $M^{-1}(v)$ และให้ B เป็นเซตของโหนดที่อยู่ใน $M_{BE}^{-1}(v)$ แต่ไม่อยู่ใน $LCA^{-1}(v)$ สังเกตว่าโหนดใน A ทั้งหมดจะต้องเป็นโหนดรากของต้นไม้ใน $F_{LCA^{-1}}(v)$ และ เนื่องจากโหนดเหล่านี้ไม่ได้อยู่ใน $M^{-1}(v)$ แสดงว่าจะต้องเป็นโหนดอิสระทั้งสิ้น ในขณะที่โหนดทั้งหมดใน B จะเป็นต้นไม้หนึ่งโหนดใน $F_{M_{BE}^{-1}}(v)$ (ดูจากบทพิสูจน์ของทฤษฎีบทย่อยที่ 2) ดังนั้น ทุกๆโหนดนำของ $M_{BE}^{-1}(v)$ ต้องอยู่ใน $LCA^{-1}(v)$ และต้องไม่อยู่ใน $M^{-1}(v)$ เนื่องจาก $h\left(F_{M^{-1}}(v)\right) = h\left(F_{LCA^{-1}}(v)\right) - 1$ ดังนั้นโหนดนำทั้งหมดของ $M_{BE}^{-1}(v)$ จะต้องอยู่ในเซต A ซึ่งส่งผลให้โหนดนำทั้งหมดนี้เป็นโหนดอิสระทั้งสิ้น หากเราพิจารณาขั้นตอนการทำงานของอัลกอริทึมของ Bansal *และ* Eulenstein เมื่ออัลกอริทึมทำการพิจารณาที่โหนด v ฟังก์ชันการส่งจากโหนดนำแต่ละโหนดของ v จะต้องถูกเปลี่ยน แสดงว่า M_{BE} ไม่ใช่คำตอบจากอัลกอริทึม เกิดเป็นข้อขัดแย้งขึ้น ดังนั้น ข้อสมมติที่ให้ $h(F_{LCA^{-1}}(v)) \geq 2$ จึงเป็นไปไม่ได้ ■

ทฤษฎีบทย่อยที่ 4 *ให้ u เป็นโหนดที่ $M_{BE}(u)$ เป็นบรรพบุรุษของ $LCA(u)$ ถ้า M เป็นการส่งที่สมเหตุสมผลโดยที่ $M(u) = LCA(u)$ จะได้ว่า*

$$h\left(F_{M_{BE}^{-1}}(LCA(u))\right) < h(F_{M^{-1}}(LCA(u)))$$

พิสูจน์ เงื่อนไขนี้แสดงว่า $M_{BE}(u)$ เป็นบรรพบุรุษของ $M(u)$ เนื่องจากอัลกอริทึมจะปรับการส่งจาก LCA เฉพาะโหนดที่เป็นโหนดนำเท่านั้น แสดงว่า u ต้องเป็นโหนดนำใน $LCA^{-1}(u)$ ซึ่งจะได้ตามมาว่า

$$h(F_{M^{-1}}(LCA(u))) \geq h(F_{LCA^{-1}}(LCA(u)))$$

นอกจากนี้ จากการทำงานของตัวอัลกอริทึม ถ้า $M_{BE}(u) \neq LCA(u)$ จะได้ว่า

$$h\left(F_{M_{BE}^{-1}}(LCA(u))\right) = h\left(F_{LCA^{-1}}(LCA(u))\right) - 1$$

ซึ่งเป็นผลให้ได้ตัวทฤษฎีบทย่อยตามต้องการ ■

ทฤษฎีบทย่อยที่ 5 *ให้* u *เป็นโหนดที่* $M_{BE}(u)$ *เป็นบรรพบุรุษของ* $LCA(u)$ *ถ้า* $\Gamma = \{x | x$ *เป็นบรรพบุรุษของ* $LCA(u)$ *และเป็นผู้สืบเชื้อสายของ* $M_{BE}(u)\}$ *จะได้ว่า* $h\left(F_{M_{BE}^{-1}}(x)\right) = 0$ *สำหรับทุกๆ* $x \in \Gamma$

พิสูจน์ พิจารณาโหนด x ใดๆ จะต้องมีการส่ง M ที่สมเหตุสมผลที่ $M(u) = x$ เกิดขึ้นระหว่างการทำงานของอัลกอริทึม อย่างไรก็ตาม เมื่ออัลกอริทึมทำงานต่อ การส่งจากโหนด u ก็เปลี่ยนไป แสดงว่า u จะต้องเป็นโหนดนำใน $M^{-1}(x)$ ซึ่งจะเห็นว่า u จะเป็นโหนดนำใน $M^{-1}(x)$ ได้เมื่อ $h(F_{M^{-1}}(x)) = 1$ เท่านั้น และการที่การส่งจาก u ถูกเปลี่ยนไปแสดงว่าโหนดนำทั้งหมดจะต้องเป็นโหนดอิสระ ดังนั้นเมื่อเปลี่ยนการส่งไปแล้ว จนจบการทำงาน จึงไม่มีโหนดใดที่ถูกส่งมายัง x อีก จึงทำให้ $h\left(F_{M_{BE}^{-1}}(x)\right) = 0$ ■

ทฤษฎีบทที่ 1 *อัลกอริทึมของ Bansal และ Eulenstein แก้ปัญหาการคัดลอกหลายยีนได้ถูกต้อง*

พิสูจน์ ในทฤษฎีบทย่อยที่ 1 เราได้พิสูจน์แล้วว่า M_{BE} เป็นการส่งที่สมเหตุสมผล ดังนั้น ส่วนที่เหลือที่ต้องพิสูจน์คือ ต้องแสดงว่า $\sum_{v \in V(S)} h\left(F_{M_{BE}^{-1}}(v)\right) = h_{opt}$ สมมติให้มีการส่ง M ที่สมเหตุสมผลที่ $\sum_{v \in V(S)} h\left(F_{M_{BE}^{-1}}(v)\right) > \sum_{v \in V(S)} h(F_{M^{-1}}(v))$ ซึ่งแสดงว่าจะต้องมีโหนด $v \in V(S)$ อย่างน้อยหนึ่งโหนดที่ $h\left(F_{M_{DE}^{-1}}(v)\right) > h(F_{M^{-1}}(v))$ เราจะสมมติเพิ่มให้ M มีจำนวนโหนดดังกล่าวน้อยที่สุด เราจะพิจารณาโหนด v ดังกล่าวที่ไม่มีโหนด w อื่นใน $V(S_v)$ ที่ $h\left(F_{M_{DE}^{-1}}(w)\right) > h(F_{M^{-1}}(w))$ อีกแล้ว

จากทฤษฎีบทย่อยที่ 3 เราจะได้ว่า $h\left(F_{M_{BE}^{-1}}(v)\right)=1$ และ $h\left(F_{M^{-1}}(v)\right)-0$ หรือกล่าวได้ว่า $F_{M^{-1}}(v)=\emptyset$ เราจะแสดงต่อไปว่า จะต้องมีโหนด $w\in V(S_v)\backslash\{v\}$ ที่ทำให้ $h\left(F_{M_{BE}^{-1}}(w)\right)<h(F_{M^{-1}}(w))$ ซึ่งจะส่งผลให้

$$\sum_{x\in V(S_v)} h(F_{M_{BE}^{-1}}(x)) \leq \sum_{x\in V(S_v)} h(F_{M^{-1}}(x))$$

ให้ $A=M_{BE}^{-1}(v)$ เป็นที่แน่นอนว่า $A\neq\emptyset$ เราจะแบ่งการพิจารณาออกเป็นสองกรณี ได้แก่ เมื่อทุกโหนด u ใน A $M(u)$ เป็นบรรพบุรุษของ v และเมื่อมีโหนด u บางโหนดใน A ที่ $M(u)$ เป็นผู้สืบเชื้อสายของ v สังเกตว่ากรณีแรกจะทำให้โหนดทั้งหมดของ A เป็นทั้งโหนดนำและโหนดอิสระ อัลกอริทึมจะต้องปรับการส่งจากโหนดเหล่านี้ต่อ ดังนั้นจึงมีกรณีหลังเท่านั้นที่เป็นไปได้

เราจะแบ่งการพิจารณาเป็นสองกรณีอีกครั้ง คือเมื่อ $M(u)=LCA(u)$ และเมื่อ $M(u)$ เป็นบรรพบุรุษของ $LCA(u)$ ในกรณีแรก เมื่อดูจากทฤษฎีบทย่อยที่ 4 เราได้ว่า $h\left(F_{M^{-1}}(M(u))\right)>h(F_{M_{BE}^{-1}}(M(u)))$ สำหรับกรณีหลัง ทฤษฎีบทย่อยที่ 5 ก็ได้บอกเราว่า $h\left(F_{M_{BE}^{-1}}(M(u))\right)=0$ แต่ $h(F_{M^{-1}}(M(u)))\neq 0$ ดังนั้น ไม่ว่ากรณีใดก็ตาม เราก็ได้ว่าจะต้องมีโหนด $w\in V(S_v)\backslash\{v\}$ ที่ทำให้ $h\left(F_{M_{BE}^{-1}}(w)\right)<h(F_{M^{-1}}(w))$

คราวนี้ ให้ $P=\bigcup_{x\in V(S_v)} M_{BE}^{-1}(x)$ และ $Q=\bigcup_{x\in V(S_v)} M^{-1}(x)$ สมมติให้มีโหนด u ที่อยู่ใน Q แต่ไม่อยู่ใน P หรือก็คือ $M(u)\in V(S_v)$ และ $M_{BE}(u)\in V(S)\backslash V(S_v)$ จะเห็นว่า ถ้า $M(u)=v$ จะขัดแย้งกับที่ได้ว่า $M^{-1}(v)=\emptyset$ และถ้า $M(u)\in V(S_v)\backslash\{v\}$ ทฤษฎีบทย่อยที่ 5 ก็บอกเราว่า $M_{BE}^{-1}(v)=\emptyset$ ซึ่งเกิดข้อขัดแย้งเช่นกัน ดังนั้น เราจึงได้ว่า $Q\subseteq P$

ถึงตรงนี้ เราได้ว่าหากพิจารณาในต้นไม้ย่อย S_v การส่ง M จะให้ค่าใช้จ่าย ซึ่งก็คือค่าผลรวมของความสูงของป่าที่เกิดจากการสะท้อนของ M ไม่น้อยไปกว่าของ M_{BE} แม้ว่า $\bigcup_{x\in V(S_v)} M^{-1}(x)\subset\bigcup_{x\in V(S_v)} M_{BE}^{-1}(x)$ พิจารณาการส่งตัวใหม่ M' ซึ่งมีนิยามดังนี้

$$M'(u)=\begin{cases} M_{BE}(u) & \text{ถ้า } M_{BE}(u)\in V(S_v), \\ M(u) & \text{กรณีอื่นๆ} \end{cases}$$

จะเห็นได้ว่า $\sum_{v \in V(S)} h(F_{M'^{-1}}(v)) \leq \sum_{v \in V(S)} h(F_{M^{-1}}(v))$ และ M' มีโหนด v ที่ $h\left(F_{M_{BE}^{-1}}(v)\right) > h(F_{M'^{-1}}(v))$ น้อยกว่า M ซึ่งขัดแย้งกับที่ให้ M มีจำนวนโหนดดังกล่าวน้อยที่สุด เราจึงได้ว่า ไม่มีการส่งที่สมเหตุสมผลอื่นใดที่เสียค่าใช้จ่ายน้อยกว่า M_{BE} ดังนั้น M_{BE} จึงเป็นคำตอบที่ดีที่สุดตามต้องการ ■

เวลาการทำงาน

คราวนี้เราจะมาวิเคราะห์เวลาการทำงานของอัลกอริทึมของ Bansal และ Eulenstein เราจะให้ n แทนจำนวนสปีชีส์ที่มีทั้งหมด และให้ m_i แทนจำนวนยีนในต้นไม้ยีน G_i โดย $m = \sum_{i=1}^{k} m_i$

ทฤษฎีบทที่ 2 *อัลกอริทึมของ Bansal และ Eulenstein ใช้เวลาการทำงานเป็น* $O(mn)$

พิสูจน์ ในขั้นตอนเริ่มต้น (บรรทัดที่ 1-2) อัลกอริทึมทำการคำนวณการส่ง LCA และช่วง $I(u)$ สำหรับโหนดยีน u แต่ละโหนด เพื่อใช้ในการตรวจสอบว่า u เป็นโหนดอิสระหรือไม่ในเวลา $O(1)$

การทำงานของอัลกอริทึมนี้ มีส่วนที่ใช้เวลามากที่สุดอยู่ในบรรทัดที่ 4-8 สำหรับแต่ละโหนด v ตามบรรทัดที่ 3 อัลกอริทึมจะตรวจสอบโหนดใน $M^{-1}(v)$ ทั้งหมดซึ่งมีขอบเขตบนเป็น $\sum_{i=1}^{k} m_i = m$ ดังนั้น อัลกอริทึมจะใช้เวลา $O(m)$ สำหรับแต่ละโหนด v และต้องคำนวณเช่นนี้ทั้งหมด n รอบ จึงได้เวลาการทำงานเป็น $O(mn)$ ■

อุปกรณ์และวิธีการ

อุปกรณ์

1. อุปกรณ์สำนักงานสำหรับวิเคราะห์และพิสูจน์ทฤษฎีทางคณิตศาสตร์
   1.1. สมุดบันทึก
   1.2. ดินสอ หรือปากกา
   1.3. ยางลบ

2. อุปกรณ์สำหรับการเขียนโปรแกรมทดลองและบันทึกผล
   2.1. เครื่องคอมพิวเตอร์ Intel® Pentium® D CPU 3.40 GHz, 0.99 GB of RAM
   2.2. โปรแกรม Microsoft Visual Studio 2005
   2.3. โปรแกรม Microsoft Office Excel 2007

วิธีการ

1. ศึกษาและวิเคราะห์อัลกอริทึมของ Bansal และ Eulenstein

ศึกษาและวิเคราะห์อัลกอริทึมเดิมของ Bansal และ Eulenstein อย่างละเอียด วิเคราะห์หาคุณสมบัติของตัวอัลกอริทึม หาข้อดีและข้อด้อยของตัวอัลกอริทึม

2. ออกแบบอัลกอริทึมใหม่สำหรับปัญหาการคัดลอกหลายยีน

นำข้อดีและข้อด้อยของอัลกอริทึมเดิมของ Bansal และ Eulenstein มาเป็นฐานในการออกแบบอัลกอริทึมใหม่ โดยปรับปรุงข้อด้อยของอัลกอริทึมเดิมให้ได้มากที่สุด หลังจากได้อัลกอริทึมใหม่แล้ว ต้องสามารถพิสูจน์ความถูกต้องและเวลาในการทำงานในเชิงทฤษฎีได้ด้วย

3. ทำการทดลองเปรียบเทียบเวลาระหว่างอัลกอริทึมใหม่และอัลกอริทึมเดิม

นำอัลกอริทึมใหม่ที่ออกแบบได้ มาเขียนโปรแกรมทดลอง เปรียบเทียบกับโปรแกรมจากอัลกอริทึมเดิมของ Bansal และ Eulenstein นำผลที่ได้มาสร้างกราฟเวลาการทำงานเพื่อดูแนวโน้ม

ผลและวิจารณ์

ผล

1. ศึกษาและวิเคราะห์อัลกอริทึมของ Bansal และ Eulenstein

จากการศึกษาอัลกอริทึมของ Bansal และ Eulenstein พบว่า ในขั้นตอนส่วนแรกที่ทำการหาการส่ง LCA นั้น สามารถทำงานได้อย่างมีประสิทธิภาพสูงอยู่แล้ว แต่ในขั้นตอนการปรับการส่งยังสามารถปรับปรุงให้ทำงานรวดเร็วยิ่งขึ้นได้ โดยเราได้พิสูจน์คุณสมบัติของอัลกอริทึม 2 ข้อต่อไปนี้ เพื่อเป็นแนวทางในการปรับปรุงอัลกอริทึมต่อไป

ทฤษฎีบทย่อยที่ 6 *สำหรับโหนด* $v \in V(S)$ *และการส่ง* M *ที่สมเหตุสมผลที่ทุกๆ โหนดนำใน* $M^{-1}(v)$ *เป็นโหนดอิสระทั้งหมด ให้* u *เป็นโหนดยีนใน* $M^{-1}(v)$ *และให้* M' *เป็นการส่งหลังจากปรับการส่งจากโหนดนำใน* $M^{-1}(v)$ *จาก* v *ไปยัง* $P(v)$ *จะได้ว่า* u *เป็นต้นไม้หนึ่งโหนดใน* $M'^{-1}(P(v))$

พิสูจน์ ให้ c เป็นโหนดลูกของ u ในต้นไม้ยีน c จะเป็นไปได้สองกรณี คือ $c \in M^{-1}(v)$ หรือ $c \notin M^{-1}(v)$ ถ้า $c \in M^{-1}(v)$ แสดงว่า c ก็ต้องเป็นโหนดลูกของ u ใน $F_{M^{-1}}(v)$ และไม่มีทางเป็นโหนดนำได้ การส่งจาก c ไม่มีทางถูกปรับขึ้นเป็น $P(v)$ ถ้า $c \notin M^{-1}(v)$ จากนิยามของการส่งที่สมเหตุสมผล เราจะได้ว่า $M(c)$ จะต้องเป็นผู้สืบเชื้อสายของ v เช่นเดียวกันกับทุกๆ โหนดใน $I(c)$ ซึ่งจะเห็นว่า $P(v) \notin I(c)$ ดังนั้น การส่งจาก c จึงไม่สามารถถูกปรับไปเป็น $P(v)$ ได้

จากทั้งสองกรณี จะเห็นว่าไม่มีโหนดลูกของ u ปรากฏอยู่ใน $F_{M'^{-1}}(P(v))$ จึงได้ว่า u เป็นต้นไม้หนึ่งโหนดใน $F_{M^{-1}}(P(v))$ ■

ทฤษฎีบทย่อยที่ 7 *ให้* $LN(v)$ *เป็นเซตของโหนดนำทั้งหมดใน* $M^{-1}(v)$ *จำนวนครั้งที่แต่ละโหนดใน* $LN(v)$ *จะถูกปรับการส่ง ระหว่างการทำงานของอัลกอริทึมของ Bansal และ Eulenstein จะมีค่าไม่เกินขนาดที่น้อยที่สุดของ* $I(u)$ *ลบด้วย* 1 *สำหรับ* $u \in LN(v)$ *กล่าวอีกทางหนึ่งก็คือแต่ละโหนดใน* $LN(v)$ *จะถูกปรับการส่งไม่เกิน* $min_{u \in LN(v)}(|I(u)| - 1)$ *ครั้ง*

พิสูจน์ จากอัลกอริทึมของ Bansal และ Eulenstein การส่งของโหนดใน $LN(v)$ จะถูกเปลี่ยนเมื่อโหนดทั้งหมดใน $LN(v)$ เป็นโหนดอิสระเท่านั้น ซึ่งหลังจากถูกปรับการส่งแล้ว โหนดเหล่านี้จะกลายเป็นต้นไม้หนึ่งโหนดของการส่งถัดไป ซึ่งอาจจะยังเป็นโหนดนำทั้งหมด หรือไม่เช่นนั้นก็ไม่มีโหนดใดเป็นโหนดนำอีกต่อไป แสดงว่า การปรับการส่งของโหนดใน $LN(v)$ จะหยุดพร้อมกันทุกโหนด ดังนั้น เราหาขอบเขตของการปรับการส่งจากโหนดเหล่านี้ได้ด้วยค่าที่น้อยที่สุดของ $|I(u)|-1$ (จำนวนตำแหน่งที่เป็นไปได้ของการคัดลอกของ u ที่ยังเหลืออยู่) เมื่อ $u \in LN(v)$ ■

2. อัลกอริทึมใหม่สำหรับปัญหาการคัดลอกหลายยีน

จากทฤษฎีบทย่อยที่ 6 จะเห็นว่าหลังจากที่โหนดนำ u ใดๆถูกปรับการส่งจาก v เป็น $P(v)$ โหนดนั้นจะยังคงเป็นโหนดนำก็ต่อเมื่อโหนดนำเดิมของ $P(v)$ เป็นรากของต้นไม้หนึ่งโหนดอยู่แล้ว หรือไม่ก็ไม่มีโหนดใดๆถูกส่งมาที่ $P(v)$ เลย ถ้าโหนด u ที่ถูกปรับการส่งมาแล้วยังเป็นโหนดนำอยู่ แทนที่เราจะต้องตรวจสอบ $I(u)$ เพื่อดูว่า u ยังเป็นโหนดอิสระอยู่หรือไม่ เราสามารถใช้ทฤษฎีบทย่อยที่ 7 เข้าช่วยในการตัดสินใจแทนได้ แนวคิดนี้นำไปสู่อัลกอริทึมใหม่ที่แสดงอยู่ในภาพที่ 6

ในขั้นตอนเริ่มต้น เราเพิ่มส่วนของการคำนวณหาโหนดนำและจำนวนครั้งที่โหนดยีนแต่ละโหนดจะถูกปรับการส่ง และในส่วนของการวนรอบเพื่อปรับการส่งนั้น เราตรวจสอบว่าที่โหนดสปีชีส์ที่สนใจอยู่มีโหนดนำหรือไม่ ถ้ามี เราจะทำการตรวจสอบว่ายังสามารถปรับการส่งของทุกๆโหนดนำได้หรือไม่ โดยดูจากจำนวนครั้งที่ยังปรับการส่งได้ (ค่าของตัวแปร $Step$) ถ้ายังสามารถปรับการส่งของโหนดนำทั้งหมดต่อได้ เราจะทำการตรวจสอบทันทีว่า โหนดเหล่านี้จะยังคงเป็นโหนดนำของการส่งใหม่อยู่หรือไม่ ถ้ายังเป็นอยู่ เราก็ทำการปรับเซตของโหนดนำของโหนดสปีชีส์ใหม่และปรับค่าของตัวแปร $Step$ ของโหนดสปีชีส์ใหม่นั้น ทำเช่นนี้ไปเรื่อยๆจนตรวจสอบโหนดสปีชีส์ครบทั้งหมดในต้นไม้ก็จะได้การส่งที่เป็นคำตอบ ตัวอย่างการทำงานแสดงอยู่ในภาพที่ 7 เราจะแสดงต่อว่าการส่งที่ได้เป็นผลลัพธ์จากอัลกอริทึมนี้ เป็นคำตอบที่ถูกต้องของปัญหาการคัดลอกหลายยีน และในการวนรอบแต่ละรอบของอัลกอริทึม เราสามารถตรวจสอบเงื่อนไขที่กล่าวมาได้ในเวลาคงที่ (constant time)

A new algorithm

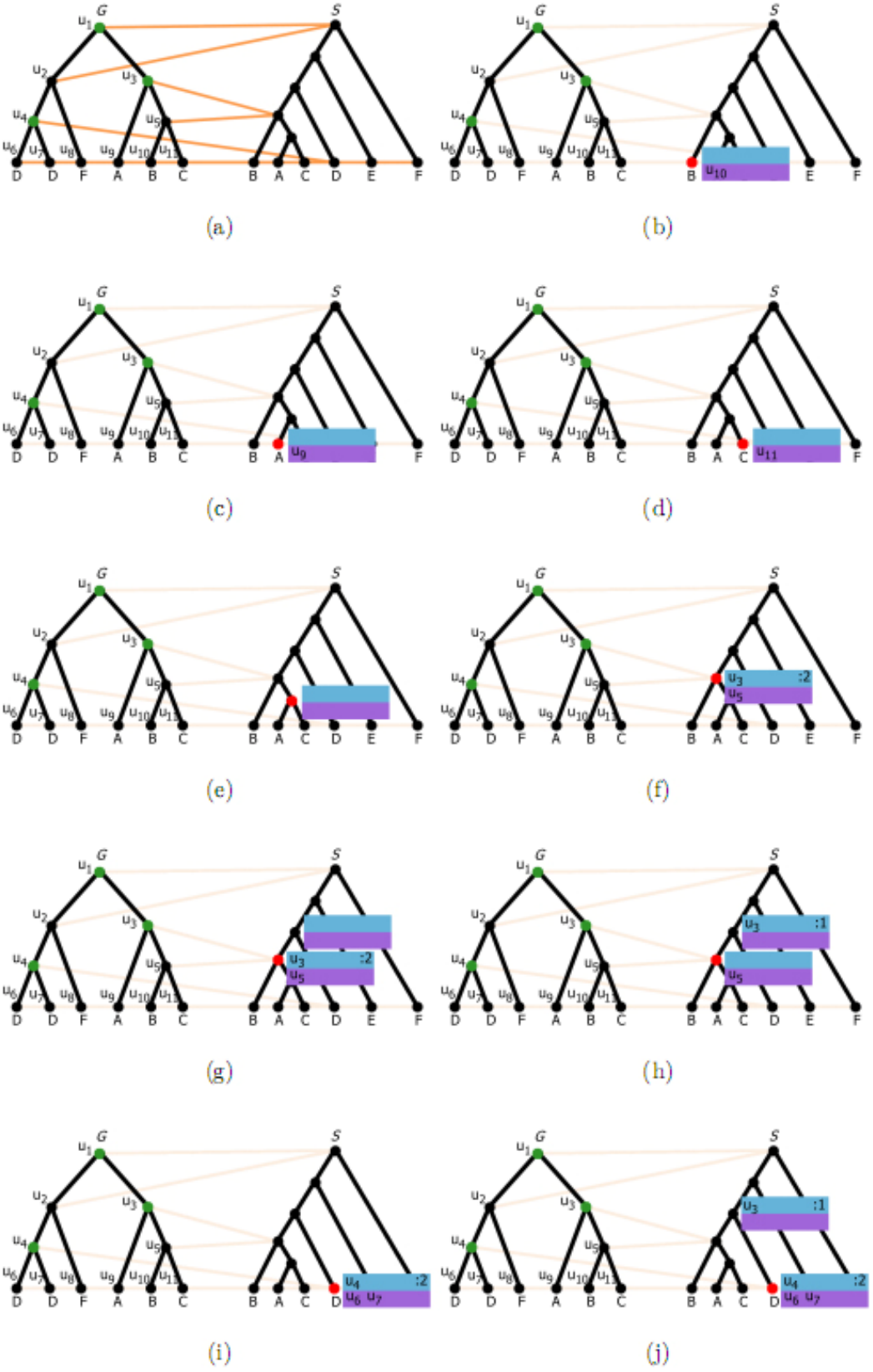
input: gene trees $G_1, \ldots, G_k$ and a species tree S

1: Initialize M to be the LCA mapping from each G_i to S.
2: Compute $|I(u)|$ for each node u in the gene trees.
3: Compute $LN(v)$ for each $v \in S$, and Let $H(v) = h(F_{M^{-1}}(v))$.
4: Compute the number of steps which all nodes in $LN(v)$ can move in, denoted by $Step(v) = \min_{u \in LN(v)}(|I(u)| - 1)$ for each $v \in S$.
5: **for** each node $v \in S$ in a post-order traversal **do**
6: **if** $LN(v) \neq \emptyset$ and $Step(v) > 0$ **then**
7: Update M by moving the maps of all nodes in $LN(v)$ from v to $P(v)$.
8: $Step(v) \leftarrow Step(v) - 1$
9: **if** $H(P(v)) = 1$ **then**
10: $Step(P(v)) \leftarrow \min(Step(P(v)), Step(v))$
11: $LN(P(v)) \leftarrow LN(P(v)) \cup LN(v)$
12: **else if** $H(P(v)) = 0$ **then**
13: $Step(P(v)) \leftarrow Step(v)$
14: $LN(P(v)) \leftarrow LN(v)$
15: $H(P(v)) \leftarrow 1$
16: **end if**
17: **end if**
18: **end for**
19: Return M.

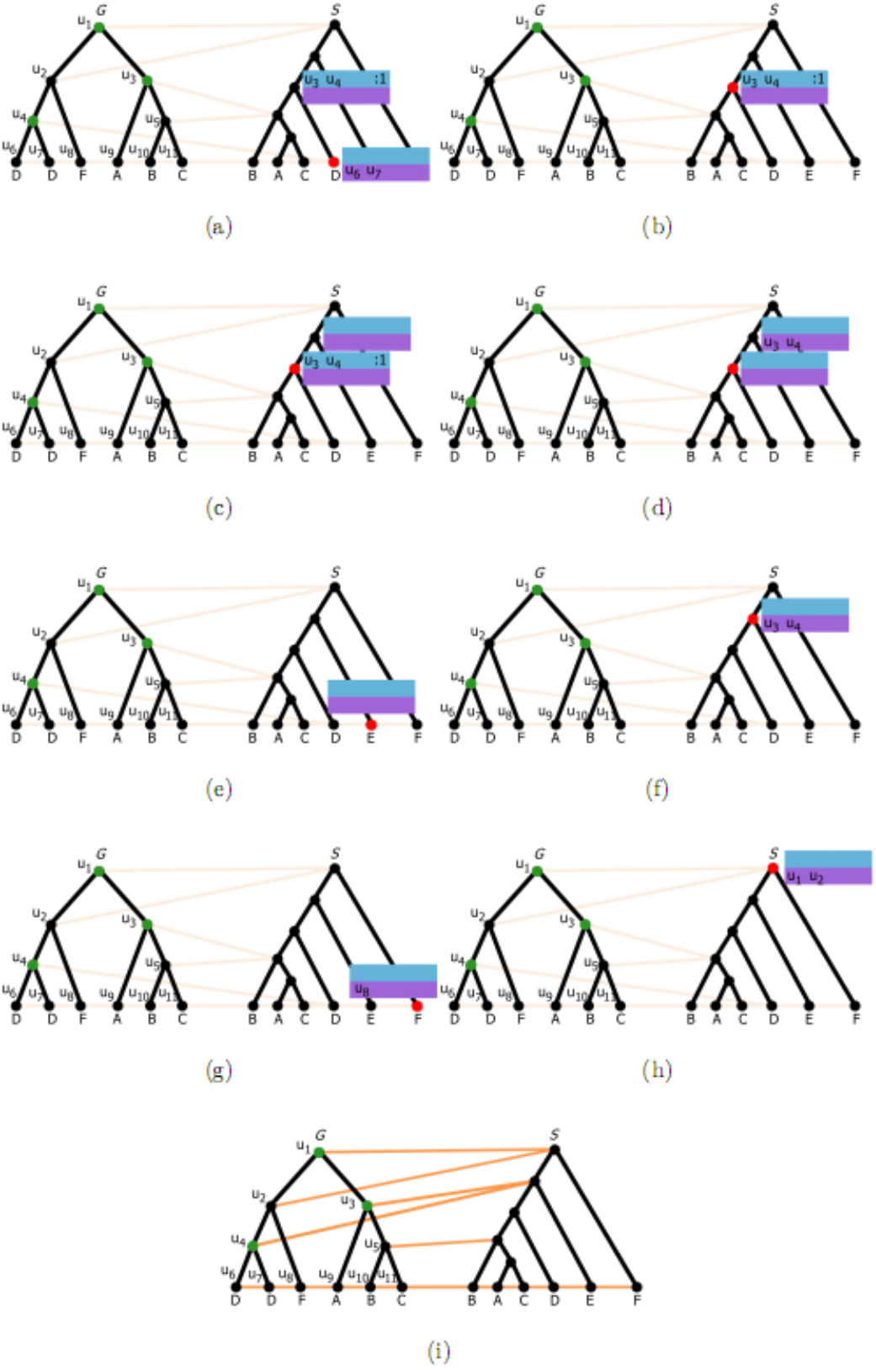
ภาพที่ 6 อัลกอริทึมใหม่

ในด้านความถูกต้อง เนื่องจากอัลกอริทึมใหม่นี้ทำการตรวจสอบและปรับการส่งจาก ***LCA*** ที่เป็นการส่งเริ่มต้นของอัลกอริทึม โดยมีการปรับเป็นลักษณะเดียวกับอัลกอริทึมของ Bansal และ Eulenstein เราสามารถเห็นได้ว่าอัลกอริทึมใหม่นี้ ให้ผลลัพธ์เดียวกันกับอัลกอริทึมเดิมของ Bansal และ Eulenstein เสมอ ซึ่งอัลกอริทึมเดิม ได้ถูกพิสูจน์แล้วว่า ให้ผลลัพธ์ที่ถูกต้องของปัญหาการคัดลอกหลายยีน ดังนั้น อัลกอริทึมใหม่นี้จึงสามารถแก้ปัญหาการคัดลอกหลายยีนได้เช่นกัน

ทฤษฎีบทที่ 3 *อัลกอริทึมใหม่ให้ผลลัพธ์เป็นคำตอบของปัญหาการคัดลอกหลายยีน*



ภาพที่ 7 ตัวอย่างการทำงานของอัลกอริทึมใหม่



ภาพที่ 7 (ต่อ)

คราวนี้เราจะมาวิเคราะห์ถึงเวลาการทำงานของอัลกอริทึมใหม่ ซึ่งพัฒนามาจากอัลกอริทึมเดิมของ Bansal และ Eulenstein ให้ n แทนจำนวนสปีชีส์ทั้งหมด และ m แทนจำนวนยีนรวมจากต้นไม้ยีนทุกต้น

ทฤษฎีบทที่ 4 *อัลกอริทึมใหม่ใช้เวลาการทำงานเป็น* $O(m+n)$

พิสูจน์ พิจารณาในบรรทัดที่ 1 อัลกอริทึม ฟังก์ชันการส่ง LCA สามารถคำนวณได้ในเวลา $O(m+n)$ โดยใช้ $O(n)$ ในขั้นตอนการเตรียมข้อมูล และ $O(m)$ ในการคำนวณการส่งจากโหนดยีนทั้งหมด ในบรรทัดที่ 2 เราไม่จำเป็นต้องคำนวณหาเซต $I(u)$ สำหรับแต่ละโหนดยีน u เนื่องจากในอัลกอริทึม เราต้องการใช้เพียงขนาดของมันเท่านั้น ซึ่งการคำนวณขนาดของ $I(u)$ สามารถทำได้ในเวลา $O(m)$ โดยดูจากความห่างของโหนดสปีชีส์จากการส่ง LCA ของ u และ $P(u)$ บรรทัดที่ 3 และ 4 ก็สามารถคำนวณได้ใน $O(m+n)$ เช่นกัน สำหรับในส่วนของวนรอบในบรรทัดที่ 5 – 18 การปรับ M ในบรรทัดที่ 7 สามารถทำได้ใน $O(1)$ หากใช้โครงสร้างข้อมูลประเภทรายการ (list) (ดูรายละเอียดในหัวข้อ การนำไปใช้ในทางปฏิบัติ) ดังนั้น วนรอบจะใช้เวลาทำงานรอบละ $O(1)$ และเนื่องจากทำเป็นจำนวน n รอบ ส่วนของบรรทัดที่ 5 – 18 จึงใช้เวลาเป็น $O(n)$ ดังนั้น เวลาการทำงานทั้งหมดของอัลกอริทึมจึงเป็น $O(m+n)$ ■

3. การทดลองและผลการทดลอง

ในงานวิจัยนี้ นอกจากการวิเคราะห์อัลกอริทึมทางทฤษฎี เราได้ทำการทดลองพัฒนาอัลกอริทึมใหม่สำหรับปัญหาการคัดลอกหลายยีน และนำมาเปรียบเทียบกับอัลกอริทึมเดิมของ Bansal และ Eulenstein ทั้งผลลัพธ์ที่ได้ และเวลาที่ใช้ในการทำงาน โดยการคำนวณฟังก์ชันการส่ง LCA เราใช้อัลกอริทึมที่ใช้เวลาในขั้นตอนจัดเตรียมข้อมูลเป็น $O(n \log n)$ ดังที่จะแสดงในหัวข้อวิจารณ์ ชุดข้อมูลนำเข้าทั้งหมดถูกสร้างขึ้นโดยการสุ่ม เมื่อกำหนดจำนวนสปีชีส์ และจำนวนต้นไม้ยีนไว้

สำหรับต้นไม้สปีชีส์ หลังจากที่รับจำนวนสปีชีส์ (n) แล้ว เราสร้างต้นไม้สปีชีส์โดยการสร้างเซตที่มีสมาชิก n ตัวก่อน โดยให้สมาชิกเหล่านั้นเป็นโหนดใบของต้นไม้ หลังจากนั้นจะทำการสุ่มสมาชิกสองตัวออกมาจากเซตดังกล่าว และสร้างสมาชิกตัวใหม่ใส่เข้าไปเป็นการแทนโหนดแม่ของโหนดที่สุ่มออกมา เมื่อทำเช่นนี้ไปเรื่อยๆ จนเซตนี้เหลือสมาชิกเพียงตัวเดียว เราก็จะได้ต้นไม้สปีชีส์ที่มี n สปีชีส์ตามต้องการ และสำหรับการสร้างต้นไม้ยีน เราทำการสุ่มจำนวนเต็มบวก

ให้กับแต่ละสปีชีส์ เพื่อแทนจำนวนยีนในต้นไม้ยีนที่พบอยู่ในสปีชีส์นั้นๆ เมื่อเราได้เซตของยีนเป็นที่เรียบร้อยแล้ว เราก็จะสามารถสร้างต้นไม้ยีนได้ด้วยขั้นตอนแบบเดียวกับที่ใช้สร้างต้นไม้สปีชีส์

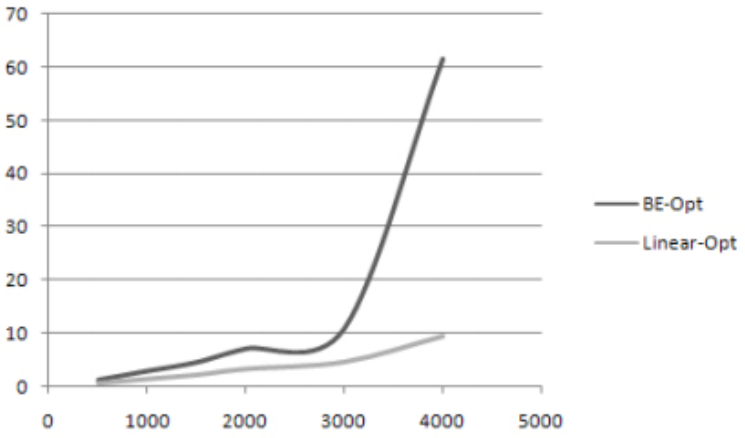
เราได้ทำการทดลองบนชุดข้อมูลขนาดต่างกัน 6 ขนาด โดยแต่ละชุดข้อมูล จะมีจำนวนต้นไม้ยีน 200 ต้น และต้นไม้สปีชีส์ต้นหนึ่ง สำหรับต้นไม้ยีนแต่ละต้น เรากำหนดช่วงให้สปีชีส์หนึ่งๆมียีนที่พบในสปีชีส์นั้นได้ซ้ำกันไม่เกิน 10 ยีน นั่นหมายความว่าจำนวนของยีนทั้งหมด อาจมีมากถึง 2000 เท่าของจำนวนสปีชีส์ก็ได้ เราได้สร้างชุดข้อมูลสำหรับทดลองจำนวน 5 ชุดจากการสุ่ม และคำนวณเวลาเฉลี่ยที่ใช้ของแต่ละอัลกอริทึม ตารางที่ 1 แสดงเวลาทำงานเฉลี่ยที่ได้จากอัลกอริทึมของ Bansal และ Eulenstein แทนเวย BE-Opt และอัลกอริทึมใหม่ แทนด้วย Linear-Opt

ตารางที่ 1 เวลาการทำงานของอัลกอริทึม BE-Opt และ Linear-Opt (วินาที)

จำนวนสปีชีส์	BE-Opt	Linear-Opt
500	1.32	0.71
1000	3.02	1.51
1500	4.63	2.29
2000	7.18	3.40
3000	10.99	4.73
4000	61.43	9.95

การทดลองทั้งหมดในงานวิจัยนี้ เราใช้ระบบปฏิบัติการ Windows XP บนเครื่อง PC Intel® Pentium® DCPU ความเร็ว 3.40 GHz และหน่วยความจำขนาด 1 GB

เมื่อนำผลการทดลองที่วัดได้ มาเขียนเป็นกราฟ จะได้ตามภาพที่ 8 จากภาพจะเห็นว่าแนวโน้มของเวลาทำงานของอัลกอริทึมของ Bansal และ Eulenstein มีลักษณะการเติบโตเป็นพาราโบลา ซึ่งสอดคล้องกับเวลาการทำงานที่ได้จากการวิเคราะห์ว่าเป็น $O(mn)$ เนื่องจากในการทดลองค่าของ m นั้นแปรตาม n ด้วย $O(mn)$ จึงเทียบเท่ากับ $O(n^2)$ นั่นเอง



ภาพที่ 8 กราฟแสดงผลการทดลอง

ในขณะที่ หากดูแนวโน้มของเวลาการทำงานของอัลกอริทึมใหม่นี้ จะเห็นว่ามีลักษณะการเติบโตที่ค่อนข้างช้า เข้าใกล้เส้นตรง สอดคล้องกับผลที่ได้จากการวิเคราะห์ ซึ่งการที่กราฟมีการเติบโตไม่เป็นเส้นตรงเลย อาจเกิดจากความคลาดเคลื่อนระหว่างการทดลอง และการที่เราใช้อัลกอริทึมสำหรับคำนวณฟังก์ชันการส่ง LCA ที่ใช้เวลาการทำงานเป็น $O(n \log n)$ ทำให้เวลาการทำงานของอัลกอริทึมที่ใช้ในการทดลองจริงๆนั้น ไม่ได้เป็นเส้นตรงหรือ $O(n)$ แต่เป็น $O(n \log n)$ ไปด้วย

วิจารณ์

1. ปัญหา LCA ในทางปฏิบัติ

ปัญหาบรรพบุรุษร่วมที่เล็กที่สุดนั้น มีการศึกษากันมาเป็นเวลานาน ในปัจจุบันมีอัลกอริทึมที่ใช้เวลา $O(n)$ ในการเตรียมข้อมูลเริ่มต้นจากต้นไม้ที่มี n โหนด และสามารถตอบโหนดบรรพบุรุษร่วมที่เล็กที่สุดของโหนดคู่ใดๆในต้นไม้ได้ในเวลาคงที่ (Bender and Farach-Colton, 2000; Buchsbaum et al., 1998; Schieber and Vishkin, 1988) อย่างไรก็ตาม อัลกอริทึมดังกล่าวนำไปปฏิบัติจริงได้ยากเนื่องจากมีความซับซ้อนมาก เราจึงเสนอทางเลือกอื่นที่ใช้เวลาการเตรียมข้อมูลเริ่มต้นมากขึ้น แต่นำไปปฏิบัติจริงได้ง่ายกว่ามาก โดยใช้เวลาเป็น $O(n \log n)$ ในการเตรียมข้อมูลเริ่มต้น และสามารถตอบปัญหาจากโหนดคู่ใดๆได้ในเวลาคงที่เช่นกัน วิธีดังกล่าวทำได้โดย แปลงปัญหาของเราให้อยู่ในรูปปัญหาใหม่ที่มีชื่อว่า *ปัญหาการค้นค่าต่ำสุดในช่วง* (the range minimum query problem [RMQ]) ซึ่งมีรายละเอียดดังนี้

อันดับแรก เราจะให้นิยามตัวปัญหา LCA และ RMQ อย่างเป็นทางการก่อน

นิยามที่ 6 *ให้ต้นไม้แบบมีราก (rooted tree)* T *ที่มี* n *โหนด และ* $u, v \in V(T)$ *ปัญหาบรรพบุรุษร่วมที่เล็กที่สุด (LCA) คือ จงหาโหนดใน* T *ที่ไกลจากรากมากที่สุดที่เป็นบรรพบุรุษของทั้ง* u *และ* v

นิยามที่ 7 *ให้อาเรย์* A *ของตัวเลข* n *ตัว และตำแหน่ง* i, j *ที่* $1 \leq i \leq j \leq n$ *ปัญหาการค้นค่าต่ำสุดในช่วง (RMQ) คือ จงหาตำแหน่งของสมาชิกที่มีค่าน้อยที่สุดในอาเรย์ย่อย* $A[i, \ldots, j]$

โดยทั่วไปแล้ว ขั้นตอนในการแก้ปัญหาทั้งคู่นี้ จะแบ่งออกเป็นสองส่วน โดยส่วนแรกคือการเตรียมข้อมูลเริ่มต้นจากต้นไม้ (สำหรับปัญหา LCA) หรืออาเรย์ (สำหรับปัญหา RMQ) และส่วนที่สองคือส่วนของการตอบปัญหาเมื่อมีคำค้นเข้ามา เพื่อความสะดวกต่อไป ถ้าอัลกอริทึมใช้เวลาในส่วนเตรียมข้อมูลเป็น $f(n)$ และใช้เวลาในการตอบคำค้นเป็น $g(n)$ เราจะกล่าวว่าอัลกอริทึมนั้นใช้เวลาการทำงานเป็น $\langle f(n), g(n) \rangle$

วิธีการแก้ปัญหา LCA ที่เราสนใจนี้ ใช้การแปลงปัญหาไปเป็นปัญหา RMQ แล้วแก้ปัญหา RMQ แทน ซึ่งปัญหาทั้งคู่นั้นมีความสัมพันธ์กันอยู่ตามทฤษฎีบทย่อยต่อไปนี้

ทฤษฎีบทย่อยที่ 8 *ถ้ามีอัลกอริทึมสำหรับปัญหา RMQ ที่ใช้เวลา* $\langle f(n), g(n) \rangle$ *จะมีอัลกอริทึมที่แก้ปัญหา LCA ได้ในเวลา* $\langle f(2n-1) + O(n), g(2n-1) + O(1) \rangle$

พิสูจน์ เราจะพิสูจน์โดยแสดงการแปลงปัญหาจาก LCA ไปเป็นปัญหา RMQ ดังต่อไปนี้ โดยสมมติให้ T เป็นต้นไม้ของปัญหา LCA

1. ให้อาเรย์ $E[1, \ldots, 2n-1]$ เก็บโหนดที่พบตามลำดับใน วงจรออยเลอร์ (Euler tour) บน T นั่นคือ $E[i]$ จะเก็บโหนดที่พบในลำดับที่ i

2. ให้ระดับชั้นของโหนดใดๆแทนระยะห่างระหว่างโหนดนั้นกับโหนดราก สร้างอาเรย์ $L[1, \ldots, 2n-1]$ โดย $L[i]$ เก็บค่าระดับชั้นของโหนดใน $E[i]$

3. ให้ตัวแทนของโหนดในวงจรออยเลอร์ เป็นตำแหน่งแรกที่โหนดนั้นปรากฏขึ้นในวงจร นั่นคือ ตัวแทนของโหนด i คือ $\min_{E[j]=i} j$ คำนวณอาเรย์ $R[1,\ldots,n]$ เมื่อ $R[i]$ เป็นตัวแทนของโหนด i

แต่ละขั้นตอนใน 3 ขั้นตอนที่กล่าวมานี้ใช้เวลา $O(n)$ เมื่อรวมกันก็จะใช้เวลาเป็น $O(n)$ สำหรับการคำนวณ LCA ระหว่างโหนด u และ v บน T สังเกตว่า

1. โหนดที่อยู่ระหว่าง u และ v ในวงจรออยเลอร์ จะอยู่ใน $E[R[u],\ldots,R[v]]$ (หรือ $E[R[v],\ldots,R[u]]$)

2. โหนดที่อยู่สูงที่สุดในวงจรย่อย $E[R[u],\ldots,R[v]]$ ก็คือ RMQ ระหว่าง $R[u]$ และ $R[v]$ ในอาเรย์ L เนื่องจาก $L[i]$ เก็บระดับชั้นของโหนดใน $E[i]$ และ RMQ จะให้ตำแหน่งของโหนดที่ระดับชั้นน้อยที่สุด

3. โหนดในตำแหน่งดังกล่าวคือ $E[RMQ_L(R[u],R[v])]$ ซึ่งก็คือผลลัพธ์ของ LCA ระหว่าง u และ v ใน T นั่นเอง

ดังนั้น หากเราสามารถแก้ปัญหา RMQ ได้ในเวลา $\langle f(n),g(n)\rangle$ เราสามารถแปลงปัญหา LCA มาเป็นปัญหา RMQ ได้โดยการสร้างอาเรย์ตามที่กล่าวมา ซึ่งใช้เวลา $O(n)$ และทำขั้นตอนเตรียมข้อมูลบนอาเรย์ L ได้ในเวลา $f(2n-1)$ และในการค้นคำตอบของ LCA ก็ใช้การค้นคำตอบของ RMQ รวมกับการอ้างถึงข้อมูลจากอาเรย์โดยใช้เวลาเป็น $O(1)$ เราจึงสามารถค้นคำตอบได้ในเวลา $g(2n-1)+O(1)$ ตามต้องการ ■

สำหรับการแก้ปัญหา RMQ เราสามารถสร้างอัลกอริทึมที่ใช้เวลาทำงาน $\langle O(n^2),O(1)\rangle$ ได้อย่างง่าย โดยการสร้างอาเรย์จัดเตรียมคำตอบสำหรับคุถามทุกๆกรณี ซึ่งมีได้ทั้งหมด n^2 กรณี โดยในแต่ละกรณี เราก็ทำการไล่หาตำแหน่งในอาเรย์ที่มีค่าน้อยที่สุดตามลำดับ ซึ่งใช้เวลาเป็น $O(n)$ ดังนั้นอัลกอริทึมนี้จึงใช้เวลาในส่วนจัดเตรียมข้อมูลเป็น $O(n^3)$ และสามารถค้นคำตอบได้ในเวลาคงที่

อย่างไรก็ตาม อัลกอริทึมดังกล่าวนี้ไม่ใช่อัลกอริทึมที่ใช้เวลาน้อยที่สุดที่ทราบกันในปัจจุบันนี้ เราสามารถพัฒนาส่วนการจัดเตรียมข้อมูลให้ใช้เวลาเพียง $O(n^2)$ ได้ โดยใช้หลักการของ *กำหนดการพลวัตร* (dynamic programming) จากความสัมพันธ์ที่พบว่า

$$RMQ(i,j) = \begin{cases} RMQ(i,j-1) & \text{ถ้า } A[RMQ(i,j-1)] < A[j], \\ j & \text{กรณีอื่นๆ} \end{cases}$$

เราสามารถนำความสัมพันธ์ดังกล่าวมาสร้างอัลกอริทึมแบบวนรอบ สำหรับจัดเตรียมคำตอบในตารางขนาด n^2 โดยใช้เวลาคำนวณคำตอบแต่ละช่องเป็น $O(1)$ ได้ดังภาพที่ 9

```
⟨O(n²), O(1)⟩ algorithm for RMQ (preprocessing)
input: Array A[1, ..., n]
 1: Let RMQ(i, j) be the solution of a query i, j.
 2: for i ← 1 to n do
 3:     for j ← i to n do
 4:         if i = j then
 5:             RMQ(i, j) ← i
 6:         else if A[RMQ(i, j − 1) < A[j] then
 7:             RMQ(i, j) ← RMQ(i, j − 1)
 8:         else
 9:             RMQ(i, j) ← j
10:         end if
11:     end for
12: end for
```

ภาพที่ 9 อัลกอริทึมแบบ $\langle O(n^2), O(1) \rangle$ สำหรับปัญหา RMQ (ขั้นตอนจัดเตรียมข้อมูล)

ในปัจจุบัน มีอัลกอริทึมสำหรับแก้ปัญหา RMQ โดยใช้เวลา $\langle O(n), O(1) \rangle$ แต่อัลกอริทึมดังกล่าวมีความซับซ้อนมาก ยากที่จะนำไปใช้ในทางปฏิบัติจริงๆ เราจึงเสนอให้ใช้อัลกอริทึมที่ใช้เวลามากขึ้นเล็กน้อยโดยใช้เวลา $\langle O(n \log n), O(1) \rangle$ แต่มีความซับซ้อนน้อยลงมาก สะดวกต่อการนำไปใช้งานจริง โดยเราจะทำการคำนวณผลลัพธ์ของคำค้นที่มีระยะห่างเป็นกำลังของสองเตรียมไว้ล่วงหน้า ในขั้นตอนจัดเตรียมข้อมูล นั่นคือ สำหรับแต่ละค่า i ตั้งแต่ 1 ถึง n

และแต่ละค่า j ตั้งแต่ 1 ถึง $\log_2 n$ เราจะทำการหาสมาชิกที่มีค่าน้อยที่สุดในอาเรย์ย่อยที่เริ่มที่ตำแหน่งที่ i และมีความยาวเป็น 2^j นั่นคือเราคะคำนวณ $M[i,j] = \text{argmin}_{i \le k \le i+2^j} A[k]$ จากที่กล่าวมา จะเห็นว่าตาราง M จะมีขนาดเป็น $O(n \log n)$ ซึ่งเราสามารถคำนวณค่าใส่ในตารางได้ในเวลา $O(n \log n)$ โดยใช้หลักของกำหนดการพลวัตรเช่นเดียวกับอัลกอริทึมที่ผ่านมา แนวคิดก็คือ เราสามารถหาตำแหน่งของค่าที่น้อยที่สุดในอาเรย์ขนาด 2^j ได้โดยเปรียบเทียบค่าในตำแหน่งที่น้อยที่สุดระหว่างอาเรย์ย่อยขนาด 2^{j-1} สองค่า เราแสดงเป็นความสัมพันธ์แบบเรียกตัวเองได้ว่า

$$M(i,j) = \begin{cases} M(i,j-1) & \text{ถ้า } A[M(i,j-1)] < A\left[M\left(i+2^{j-1}-1, j-1\right)\right], \\ M\left(i+2^{j-1}-1, j-1\right) & \text{กรณีอื่นๆ} \end{cases}$$

จากความสัมพันธ์นี้ เราสามารถนำมาพัฒนาเป็นอัลกอริทึมสำหรับขั้นตอนจัดเตรียมข้อมูลที่ใช้เวลาทำงาน $O(n \log n)$ ได้ดังภาพที่ 10

$\langle O(n \log n), O(1) \rangle$ **algorithm for RMQ (preprocessing)**

input: Array $A[1, \ldots, n]$

```
1: Let M(i, j) be the solution of a query i, i + 2^j − 1.
2: for i ← 1 to n do
3:     M(i, 0) ← i
4: end for
5: for j ← 1 to ⌊log n⌋ do
6:     for i ← 1 to n − 2^j + 1 do
7:         if A[M(i, j − 1)] < A[M(i + 2^(j−1) − 1, j − 1)] then
8:             M(i, j) ← M(i, j − 1)
9:         else
10:            M(i, j) ← M(i + 2^(j−1) − 1, j − 1)
11:        end if
12:    end for
13: end for
```

ภาพที่ 10 อัลกอริทึมแบบ $\langle O(n \log n), O(1) \rangle$ สำหรับปัญหา RMQ (ขั้นตอนจัดเตรียมข้อมูล)

เมื่อเราคำนวณค่าใส่ในตาราง M ได้เรียบร้อยแล้ว ในการค้นคำตอบสำหรับช่วงหนึ่งในอาเรย์ เราสามารถทำได้โดยการหาช่วงย่อยสองช่วงที่อาจซ้อนกัน โดยอาเรย์ย่อยทั้งคู่เมื่อนำมารวมกันจะครอบคลุมช่วงที่ต้องการทั้งหมด หลังจากนั้น ก็นำค่าที่น้อยที่สุดของอาเรย์ย่อยทั้งคู่มาเปรียบเทียบกันแล้วจึงตอบตำแหน่งของตัวที่น้อยกว่า ซึ่งทั้งหมดนี้สามารถทำได้ในเวลาคงที่ หรือใช้เวลาเป็น $O(1)$ นั่นเอง อัลกอริทึมส่วนนี้แสดงอยู่ในภาพที่ 11

$\langle O(n \log n), O(1) \rangle$ **algorithm for RMQ (query)**

input: Array $A[1, \ldots, n]$, index i, j and Table M

1: Let $k = \lfloor \log(j - i) \rfloor$
2: **if** $A[M(i, k)] < A[M(j - 2^k + 1, k)]$ **then**
3: return $M(i, k)$
4: **else**
5: return $M(j - 2^k + 1, k)$
6: **end if**

ภาพที่ 11 อัลกอริทึมแบบ $\langle O(n \log n), O(1) \rangle$ สำหรับปัญหา RMQ (การค้นคำตอบ)

2. โครงสร้างข้อมูล

ในหัวข้อนี้เราจะกล่าวถึงโครงสร้างข้อมูลที่ใช้สำหรับอัลกอริทึมในภาพที่ 6 อย่างละเอียด จากตัวอัลกอริทึมจะเห็นได้ว่า ส่วนที่ค่อนข้างยุ่งยากคือการเก็บฟังก์ชันการส่ง และโหนดนำของแต่ละสปีชีส์ วิธีการแก้ปัญหาตรงจุดนี้ เราใช้โครงสร้างข้อมูลแบบ *รายการโยง* (linked list) โดยสำหรับแต่ละโหนดสปีชีส์ เราจะมีรายการโยงสองรายการ ผูกไว้กับสปีชีส์นั้น รายการแรกเรียกว่า *moving list* และรายการที่สองเรียกว่า *stopped list* โดย moving list จะเก็บ ตัวชี้ (pointer) ที่ชี้ไปยังโหนดยีนที่เป็นโหนดนำของสปีชีส์นั้นๆ และ stopped list จะเก็บตัวชี้ที่ชี้ไปยังโหนดอื่นๆที่ฟังก์ชันการส่ง M ส่งมายังสปีชีส์นั้นๆ

การนำโครงสร้างข้อมูลดังกล่าวไปใช้งาน ในตอนแรก เราทำการคำนวณการส่ง LCA ทำให้ทราบว่าโหนดยีนแต่ละโหนดจะเริ่มต้นจากสปีชีส์ไหนในต้นไม้สปีชีส์ ซึ่งการเก็บการส่งจากโหนดยีนไปยังโหนดสปีชีส์ สามารถทำได้โดยนำตัวชี้ของโหนดยีนไปใส่ไว้ในรายการของโหนดสปีชีส์ตามต้องการ ซึ่งอาจเก็บไว้ใน stopped list ทั้งหมดในตอนแรก หลังจากนั้น เราทำการ

คำนวณหาโหนดนำของแต่ละสปีชีส์ จากตรงนี้เราสามารถแยกโหนดนำออกจากโหนดอื่นได้โดยการย้ายตัวชี้ของโหนดนำมาเก็บไว้ใน moving list ในกรณีที่โหนดนำเหล่านั้นยังเป็นโหนดอิสระ

เมื่อเราต้องการปรับการส่งของโหนดนำทั้งหมดของโหนดสปีชีส์ v เราสามารถทำได้โดยการย้ายหัวรายการของ moving list ของ v ให้ไปต่อท้าย moving list ของ $P(v)$ ถ้าโหนดเหล่านั้นยังเป็นอิสระ หรือไม่เช่นนั้นก็ไปต่อท้าย stopped list ของ $P(v)$ การใช้รายการโยงทำให้ขั้นตอนการปรับฟังก์ชันการส่ง สามารถทำได้อย่างรวดเร็ว โดยไม่ต้องมาไล่กรับการส่งจากแต่ละโหนดยืนซึ่งเป็นการเสียเวลา

เมื่อขั้นตอนการปรับการส่งเสร็จสิ้นลง เราสามารถไล่ค้นในรายการทั้งหมดที่มีของแต่ละสปีชีส์ เพื่อนำไปตอบเป็นการส่งสุดท้ายที่เป็นคำตอบได้

สรุปและข้อเสนอแนะ

สรุป

ในงานวิจัยนี้ เราได้เสนออัลกอริทึมใหม่สำหรับแก้ปัญหาการคัดลอกหลายยืน ซึ่งใช้เวลาการทำงานเร็วขึ้นกว่าอัลกอริทึมที่มีอยู่เดิมซึ่งเสนอโดย Bansal และ Eulenstein โดยเราได้ทำการวิเคราะห์ให้เห็นว่า อัลกอริทึมใหม่ใช้เวลาการทำงานเป็นเชิงเส้นบนขนาดของข้อมูลนำเข้า

นอกจากนี้ งานวิจัยนี้ได้พัฒนาอัลกอริทึมใหม่ขึ้นเพื่อใช้งานจริง และได้ทำการทดลองเปรียบเทียบแวลาการทำงานกับอัลกอริทึมเดิมของ Bansal และ Eulenstein พบว่าผลที่ได้จากการทดลองนั้น สอดคล้องกับการวิเคราะห์ในเชิงทฤษฎี และใช้เวลาการทำงานเรวกว่าอัลกอริทึมเดิมของ Bansal และ Eulenstein

ข้อเสนอแนะ

สำหรับแนวทางการวิจัยต่อในด้านนี้ แม้ว่าผลจากงานวิจัยนี้จะสามารถเพิ่มความเร็วในการวัดความเหมาะสมระหว่างต้นไม้ยืนและต้นไม้สปีชีส์ แต่เมื่อมองย้อนกลับไปถึงปัญหาตั้งต้นที่สำคัญที่สุดในการวิจัยด้านนี้ คือปัญหาต้นไม้สปีชีส์ที่ดีที่สุด จะเห็นว่ามีกระบวนการที่เหมาะสมสำหรับแก้ปัญหาเพียงน้อยนิดเท่านั้น วิธีการที่ดีที่มีการเสนอมาก็สามารถแก้ปัญหาได้ในกรณีพิเศษเพียงเล็กน้อย ปัญหานี้ก็จะยังคงเป็นประเด็นสำคัญในการศึกษาวิจัยด้านนี้ต่อไป

เอกสารและสิ่งอ้างอิง

Bansal, Mukul S. and Oliver Eulenstein. 2008. The multiple gene duplication problem revisited. **Bioinformatics (Oxford, England)**, 24(13): i132-i138

Bender, Michael A. and Martin Farach-Colton. 2000. The lca problem revisited. **LATIN'00: Proceeding of the 4th Latin American Symposium on Theoretical Informatics**, pages 88-94

Buchsbaum, Adam L., Haim Kaplan, Anne Rogers, and Jeffery R. Westbrook. 1998. Linear-time pointer-machine algorithms for least common ancestors, mst verification, and dominators. **STOC'98: Proceedings of the thirtieth annual ACM symposium on Theory of computing**, pages 279-288

Daskalakis, Constantinos, Elchanan Mossel, and S'ebastien Roch. 2006. Optimal phylogenetic reconstruction. **STOC'06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing**, pages 159-168

Field, K.G., G. J. Olsen, D. L. Lane, Giovanni, Ghiselin, Raff, Pace, and Raff. 1988. Molecular phylogeny of the animal kingdom. **Science**, 239:748-753

Goloboff, P., S. Farris, and K. Nixon. 2000. **Tnt (tree analysis using new technology)**, (BETA) Published by the authors, Tucum.

Guigo, Roderic, Ilya Muchnik, and Temple F. Smith. 1996. Reconstruction of ancientmolecular phylogeny. **Molecular Phylogenetics and Evolution**, 6:189-213(25)

Harel, Dov and Robert Endre Tarjan. 1984. Fast algorithms for finding nearest common ancestors. **SIAM J. Comput.**, 13(2): 338-355

Lake, J.A. 1987. Origin of the metazoan**. Proc. Natl. Acad. Sci.**

Ma, Bin, Ming Li, and Louxin Zhang. 2001. From gene trees to species trees. **SIAM Journal on Computing**, 30(3): 729-752

Mihaescu, R., D. Adkins, C. Hill, A. Jaffe, and S. Rao. 2005. A simple polynomial time algorithm for reconstructing all phylogenies from log-sized sequences. **Manuscript**

Moret, B.M.E., J. Tang, and T. Warnow. 2005. Reconstructing phylogenies from gene-content and gene-order data. In O. Gascuel, editor, **Mathematics of Evolution and Phylogeny**, pages 321-352. Oxford University Press

Page, R. 1994. Maps between trees and cladistic analysis of historical associations among genes

Schieber, Baruch and Uzi Vishkin. 1988. On finding lowest common ancestors: simplification and parallelization. **SIAM J. Comput.**, 17(6): 1253-1262

Stamatakis, Alexandros. 2006. Raxml-vi-hpc: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed. **Bioinformatics**

Swofford, D. Paup*: Phylogenetic analysis using parsimony* (and other methods). 4.0b7 beta version.

ประวัติการศึกษา และการทำงาน

ชื่อ –นามสกุล	วัชรพัฐ เมตตานันท
วัน เดือน ปี ที่เกิด	26 สิงหาคม พ.ศ. 2528
สถานที่เกิด	จังหวัด กรุงเทพมหานคร
ประวัติการศึกษา	วศ.บ. (วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเกษตรศาสตร์ (พ.ศ. 2550)
ตำแหน่งหน้าที่การงานปัจจุบัน	-
สถานที่ทำงานปัจจุบัน	-
ผลงานดีเด่นและรางวัลทางวิชาการ	-
ทุนการศึกษาที่ได้รับ	-