ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์

วิศวกรรมศาสตรดุษฎีบัณฑิต (วิศวกรรมคอมพิวเตอร์)

ปริญญา

วิศวกรรมคอมพิวเตอร์ | วิศวกรรมคอมพิวเตอร์

สาขา | **ภาควิชา**

เรื่อง การหาเส้นทางสำรองและการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์โดยใช้การโปรแกรมเชิงเส้น

Multicast Recovery and Oblivious Routing using Linear Programming

นามผู้วิจัย นายสุวรา สุระประเสริฐ

ได้พิจารณาเห็นชอบโดย

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D.)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม

(รองศาสตราจารย์ศิริพร อ่องรุ่งเรือง, MS.)

หัวหน้าภาควิชา

(ผู้ช่วยศาสตราจารย์ภุชงค์ อุทโยภาศ, Ph.D.)

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์รับรองแล้ว

(รองศาสตราจารย์กัญจนา ธีระกุล, D.Agr.)

คณบดีบัณฑิตวิทยาลัย

วันที่ **เดือน** **พ.ศ.**

วิทยานิพนธ์

เรื่อง

การหาเส้นทางสำรองและการหาเส้นทางแบบออบลิเวียส ในเครือข่ายมัลติคาสต์โดยใช้การโปรแกรมเชิงเส้น

Multicast Recovery and Oblivious Routing using Linear Programming

โดย

นายสุวรา สุระประเสริฐ

เสนอ

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์
เพื่อความสมบูรณ์แห่งปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต (วิศวกรรมคอมพิวเตอร์)
พ.ศ. 2555

สุวรา สุระประเสริฐ 2555: การหาเส้นทางสำรองและการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์โดยใช้การโปรแกรมเชิงเส้น ปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต (วิศวกรรมคอมพิวเตอร์) สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก: ผู้ช่วยศาสตราจารย์จิตร์ทัศน์ ฝักเจริญผล, Ph.D. 70 หน้า

งานวิจัยนี้ศึกษาปัญหาที่เกี่ยวข้องกับการหาเส้นทางสำหรับการส่งข้อมูลแบบมัลติคาสต์ โดยจะประกอบด้วยงานวิจัยสองส่วน ในส่วนแรกจะเป็นการสร้างเส้นทางเพื่อส่งข้อมูลสำรองในเครือข่ายมัลติคาสต์ เพื่อให้เครือข่ายมัลติคาสต์สามารถส่งข้อมูลได้อย่างต่อเนื่องเมื่อเกิดปัญหาขึ้นกับลิงค์ในเครือข่าย โดยใช้การโปรแกรมเชิงเส้นเพื่อคำนวณเส้นทางสำรองที่สามารถส่งข้อมูลได้ปริมาณมากที่สุด ในงานวิจัยจะมีการนำ recovery scheme แบบต่างๆมาเปรียบเทียบประสิทธิภาพ และเสนอ recovery scheme รูปแบบใหม่ที่มีชื่อว่า Path Restricted Recovery scheme การเปรียบเทียบประสิทธิภาพกระทำกับเครือข่ายทั้งแบบสุ่มที่ได้จากโมเดลแบบ Barabasi-Albert และเครือข่ายจริงคือ Carrier Backbone Network และ US Long Distance Network ผลการทดลองพบว่า scheme ที่นำเสนอให้ค่าประสิทธิภาพของปริมาณข้อมูลที่ส่งได้ใกล้เคียงกับ scheme ที่ดีที่สุดแต่ใช้พื้นที่ในการเก็บข้อมูลที่น้อยกว่า

ในส่วนที่สองเป็นการหาเส้นทางการส่งข้อมูลแบบออบลิเวียสในเครือข่ายมัลติคาสต์ซึ่งประกอบไปด้วยส่วนของการพิสูจน์ทางทฤษฎีที่แสดงให้เห็นความเท่ากันของการหาเส้นทางแบบออบลิเวียสบนเครือข่ายยูนิคาสต์และมัลติคาสต์ ส่วนถัดมาจะเป็นการใช้การโปรแกรมเชิงเส้นเพื่อหาเส้นทางแบบออบลิเวียสที่ดีที่สุดสำหรับการส่งข้อมูลของกลุ่มมัลติคาสต์ ซึ่งใช้การรวบรวมข้อมูลเพิ่มเติมเพื่อทำให้ลักษณะของการส่งข้อมูลอ้างอิงกับการใช้งานจริงมากยิ่งขึ้น ซึ่งต่างจากการหาเส้นทางแบบออบลิเวียสแบบแท้ๆซึ่งจะต้องคิดทุกกรณีที่เป็นไปได้จากลักษณะของเครือข่ายอย่างเดียวทำให้วิธีที่นำข้อมูลบางส่วนมาช่วยในการหาเส้นทางให้ค่า congestion ที่ต่ำกว่า

_____ _____ ___ / ___ / ___

ลายมือชื่อนิสิต ลายมือชื่ออาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

Suwara Suraprasert 2012: Multicast Recovery and Oblivious Routing using Linear Programming. Doctor of Engineering (Computer Engineering), Major Field: Computer Engineering, Department of Computer Engineering Thesis Advisor: Assistant Professor Jittat Fakjaroenphol, Ph.D. 70 pages.

This thesis considers routing problems in multicast networks. There are two main parts. The first part considers the problem of finding the backup trees in multicast networks to handle the situation when the link failure occurs in the network. We study many recovery schemes that have been proposed before and compare their performance. In this work, we use throughput as the metric for evaluating their performance. We also propose the new recovery scheme called Path Restricted Recovery Scheme. To compare the performance of each recovery scheme, we perform experiments on random networks generated under the Barabasi-Albert model and the two real networks: Carrier Backbone Network and US Long Distance Network. The propose recovery scheme performs better than all other schemes except the Unrestricted Recovery Scheme (UR) that recomputed the whole multicast tree from scratch. However, the new propose scheme use less memory to keep the backup trees than the UR scheme.

The second part of this study is to find the oblivious routing in multicast networks. We prove a theorem that shows the equivalence between oblivious routing in unicast networks and in multicast networks. We also consider extending the oblivious routing framework to a more practical setting by limiting the demand space considered by the algorithm. We proposed to use real demand samples and apply the technique that solves the general oblivious routing problem to find the optimal oblivious routing in this setting. We prove a simple performance guarantee for the algorithm and from the experiments we conclude that this approach may be useful when a more fine-grain information is available.

_____ _____ ___ / ___ / ___

Student's signature Thesis Advisor's signature

กิตติกรรมประกาศ

ผู้วิจัยขอกราบขอบพระคุณ ผศ.ดร.จิตรทัศน์ ฝักเจริญผล ประธานกรรมการที่ปรึกษาวิทยานิพนธ์ รศ.ศิริพร อ่องรุ่งเรือง กรรมการที่ปรึกษาสาขาวิชาเอก ที่ให้คำปรึกษาในการเรียน การค้นคว้าวิจัย ตลอดจนการตรวจแก้ไขวิทยานิพนธ์จนกระทั่งเสร็จสมบูรณ์

ขอกราบขอพระคุณอาจารย์ภาควิชาวิศวกรรมคอมพิวเตอร์ทุกท่าน ที่ได้อบรมสั่งสอนและมอบความรู้อันเป็นประโยชน์อย่างยิ่งในการนำไปใช้ประโยชน์ต่อไป

สุวรา สุระประเสริฐ

กุมภาพันธ์ 2555

สารบัญ

หน้า

สารบัญ	(1)
สารบัญตาราง	(2)
สารบัญภาพ	(3)
คำนำ	1
วัตถุประสงค์	7
การตรวจเอกสาร	8
อุปกรณ์และวิธีการ	16
อุปกรณ์	16
วิธีการ	16
ผลและวิจารณ์	57
สรุปและข้อเสนอแนะ	65
เอกสารและสิ่งอ้างอิง	67
ประวัติการศึกษาและการทำงาน	70

สารบัญตาราง

ตารางที่		หน้า
1	ค่า complexity จากการคำนวณในแต่ละวิธี	12
2	การแบ่งปันการใช้งานแบนวิธ	24
3	แสดงจำนวนแบคอัพทรีเฉลี่ย	60
4	แสดงจำนวนเฉลี่ยของลิงค์ที่ใช้ในการสร้างเส้นทางสำรองต่อ 1 กลุ่มมัลติคาสต์	61
5	แสดงเวลาเฉลี่ยที่ใช้ในการประมวลผล (วินาที)	62
6	แสดงค่า congestion จากการส่งข้อมูลแบบออบลิเวียส	63
7	เปรียบเทียบค่าเฉลี่ย congestion สูงสุดระหว่างวิธีการหาเส้นทางการส่งข้อมูลแบบปรับเปลี่ยนได้กับวิธีการหาเส้นทางแบบออบลิเวียส	64

สารบัญภาพ

ภาพที่		หน้า
1	recovery scheme ในยูนิคาสต์	11
2	แสดง recovery scheme แบบต่างๆ	18
3	แสดง suspended part ของ global recovery scheme	19
4	แสดง suspended part ของ local recovery scheme	20
5	การสร้าง maximal disjoint path	22
6	เครือข่ายที่ A และ C มีค่า ingress หนึ่งหน่วย ขณะที่ B และ D มีค่า egress หนึ่งหน่วย	47
7	แสดงดีมานด์ที่เป็นไปได้จากค่า ingress-egress ในภาพที่ 6	48
8	ดีมานด์แบบต่างๆ ที่แต่ละโหนดมีค่า ingress-egress ที่เท่ากัน	48
9	แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย Carrier-Backbone	58
10	แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย US-Backbone	58
11	แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย Random-Dense	59
12	แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย	59

การหาเส้นทางสำรองและการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์ โดยใช้การโปรแกรมเชิงเส้น

Multicast Recovery and Oblivious Routing using Linear Programming

คำนำ

ในยุคปัจจุบันมีโปรแกรมประยุกต์บนเครือข่ายอินเทอร์เน็ตเกิดขึ้นอย่างมากมาย ทั้งหมดนี้เพื่อทำให้คนจากทั่วทุกมุมโลกสามารถติดต่อสื่อสารกับคนในอีกฟากหนึ่งของโลกได้อย่างรวดเร็ว การสื่อสารผ่านเครือข่ายอินเทอร์เน็ตได้พัฒนาจากยุคเริ่มแรกที่มีการใช้เมล์อิเลคโทรนิคส์ ซึ่งจัดเป็นการสื่อสารแบบออฟไลน์ ไปเป็นการสื่อสารที่มีรูปแบบเป็นเวลาจริงมากขึ้น เช่น การใช้โปรแกรมแชทผ่านตัวอักษร ความต้องการคุณภาพในการสื่อสารที่มีลักษณะเหมือนได้พูดคุยอยู่ตรงหน้าเพิ่มขึ้นเรื่อย ๆ จนในปัจจุบันได้พัฒนามาเป็นระบบการสื่อสารผ่านวิดีโอ ซึ่งรองรับทั้งการพูดคุยแบบหนึ่งต่อหนึ่ง หรือเป็นการสนทนาแบบเป็นกลุ่มหรือวิดีโอคอนเฟอเรนซ์ นอกจากนี้การสื่อสารแบบวิดีโอยังครอบคลุมไปถึงการถ่ายทอดรายการทางอินเทอร์เน็ตอีกด้วย

สังเกตว่ารูปแบบในการสื่อสารบนอินเทอร์เน็ตภายในโปรแกรมประยุกต์ที่ได้กล่าวมานั้นสามารถจำแนกได้เป็นสองแบบใหญ่ ๆ ตามจำนวนผู้รับข้อมูลชุดเดียวกันที่ส่งออกมาจากเครื่องต้นทาง เราจะเรียกการส่งข้อมูลว่าเป็นการส่งแบบยูนิคาสต์ (unicast) เมื่อเป็นการส่งข้อมูลจากเครื่องต้นทางหนึ่งเครื่องไปยังเครื่องปลายทางเพียงเครื่องเดียว และเรียกว่าเป็นการส่งแบบมัลติคาสต์ (multicast) เมื่อมีการส่งข้อมูลจากเครื่องต้นทางไปยังกลุ่มของเครื่องรับที่มากกว่าหนึ่งเครื่อง และสุดท้าย เราจะเรียกการส่งว่าเป็นบรอดคาสเมื่อเป็นการส่งจากเครื่องต้นทางไปยังทุกๆเครื่องที่อยู่ในเครือข่าย

ในงานวิจัยนี้เราจะเน้นศึกษาการส่งข้อมูลแบบมัลติคาสต์เป็นหลัก โดยเราจะพิจารณาประเด็นปัญหาสองกลุ่ม คือ (1) การออกแบบจัดสรรแบนด์วิธในการส่งข้อมูลแบบมัลติคาสต์เพื่อรองรับปัญหาที่เกิดขึ้นกับลิงค์ส่งข้อมูล และ (2) ปัญหาเกี่ยวข้องกับการหาเส้นทางสำหรับการส่งข้อมูลแบบมัลติคาสต์

เราจะกล่าวถึงงานวิจัยในส่วนแรกซึ่งเกี่ยวกับการสร้างเส้นทางสำรองสำหรับการส่งข้อมูลในกรณีที่เกิดปัญหาบนเครือข่ายมัลติคาสต์เพื่อที่จะทำให้สามารถส่งข้อมูลที่เป็นภาพและเสียงได้อย่างต่อเนื่องและมีคุณภาพที่ดี

การจัดการกับการส่งข้อมูลเมื่อเกิดปัญหาขึ้นกับระบบเครือข่ายเป็นสิ่งที่สำคัญ สังเกตว่าการประยุกต์ใช้การส่งข้อมูลแบบมัลติคาสต์มักเป็นการส่งข้อมูลที่มีลักษณะเป็นเวลาจริง เช่น การถ่ายทอดสดหรือการประชุมออนไลน์เป็นต้น สำหรับงานในลักษณะที่กล่าวมานี้ สิ่งสำคัญที่จะรับประกันความพอใจของผู้ใช้ก็คือความต่อเนื่องของภาพและเสียงที่ไปปรากฏอยู่ทางด้านของฝั่งผู้รับ

ปัจจัยที่เกี่ยวข้องกับคุณภาพของการรับชมหรือรับฟังข้อมูลภาพและเสียงผ่านระบบอินเทอร์เน็ตมีอยู่หลายประการ ในกลุ่มแรกเป็นปัจจัยภายในระบบเครือข่ายเอง เช่น การจัดสรรแบนด์วิธที่ใช้ในการส่งข้อมูลที่เพียงพอกับความต้องการของข้อมูลภาพและเสียงต้องการส่งไปยังผู้รับ และดีเลย์ที่เกิดขึ้นภายในระบบเครือข่าย รวมไปถึงจิตเตอร์ที่หมายถึงเวลาที่ข้อมูลแต่ละชุดถูกส่งออกไปว่าห่างกันเพียงใด

นอกจากปัจจัยภายในระบบเครือข่ายแล้ว ยังมีปัจจัยภายนอกซึ่งเป็นการยากมากที่ผู้ดูแลระบบเครือข่ายจะสามารถป้องกันและคาดการณ์ได้ ยกตัวอย่างเช่น อาจจะเกิดกรณีที่ลิงค์ในระบบเกิดความผิดพลาดขึ้นมาทำให้ไม่สามารถส่งข้อมูลไปบนลิงค์นั้นได้ตามปกติ ซึ่งกรณีนี้อาจจะต้องมีวิธีการเพื่อให้เราสามารถส่งข้อมูลต่อไปได้แม้ว่าจะเกิดความผิดพลาดใดๆเกิดขึ้นบนลิงค์ของการส่งข้อมูล

งานวิจัยในส่วนแรกจะเป็นการศึกษาเพื่อรับมือกับปัญหาที่เกิดขึ้นกับลิงค์ใดๆ ในเครือข่ายจนทำให้ไม่สามารถใช้งานลิงค์นั้นได้ อาศัยหลักการของการสร้างแบคอัพทรี (backup tree) หรือเส้นทางสำรอง ระบบจะต้องจัดสรรแบนด์วิธเพื่อใช้สำหรับใช้ในเส้นทางสำรองเพื่อส่งข้อมูลในกรณีที่ลิงค์ปกติที่ใช้ส่งข้อมูลเกิดปัญหาขึ้น โดยแต่ละลิงค์ที่ถูกใช้ในไพรมารีทรี (primary tree) หรือเส้นทางหลัก จะต้องสร้างแบคอัพทรีหนึ่งชุดเอาไว้เพื่อส่งข้อมูลสำรอง สำหรับในเครือข่ายยูนิคาสต์นั้น เส้นทางสำรองที่สร้างขึ้นจะเรียกว่าแบคอัพพาธ (backup path)

ในเครือข่ายแบบยูนิคาสต์ได้มีการนำเสนอ recovery scheme อยู่หลายชนิดซึ่งอธิบายวิธีการในการกำหนดเส้นทางสำรอง และจัดสรรแบนด์วิธเพื่อให้มีการใช้ทรัพยากรของเครือข่ายอย่างมีประสิทธิภาพ งานวิจัยของ Cohen and Nakibly (2010) ได้แบ่งวิธีการเหล่านี้เป็นหมวดหมู่ โดยมีสมมติฐานเบื้องต้นว่าไพรมารีพาธได้ถูกกำหนดให้มาแล้ว และได้แสดงการใช้การโปรแกรมเชิงเส้น (linear programming) เพื่อที่จะหาวิธีการจัดสรรแบนด์วิธที่ทำให้เครือข่ายสามารถส่งข้อมูลได้มากที่สุดโดยใช้ scheme ที่ต่างกัน

การส่งข้อมูลแบบมัลติคาสต์เป็นการส่งข้อมูลชนิดเดียวกันไปยังหลายๆ จุดปลายทาง เมื่อเกิดปัญหากับลิงค์ภายในไพรมารีทรี การส่งข้อมูลไปตามเส้นทางสำรองจะเกิดขึ้นเพื่อที่จุดปลายทางทั้งหมดที่ได้รับผลกระทบจะได้รับข้อมูลครบถ้วนเช่นเดิม ส่วนจุดปลายทางอื่นๆ ที่ไม่ได้รับผลกระทบโดยตรงจากลิงค์ที่มีปัญหาก็ยังสามารถส่งข้อมูลในไพรมารีทรีได้ตามปกติ เราได้ให้ความสนใจกับ recovery scheme แบบต่างๆ ในเครือข่ายยูนิคาสต์ ที่จะนำมาประยุกต์ใช้สำหรับเครือข่ายแบบมัลติคาสต์ อาศัยคุณสมบัติของการส่งข้อมูลแบบมัลติคาสต์ที่สามารถแชร์แบนด์วิธกันได้ ทั้งในกลุ่มมัลติคาสต์เดียวกันและแบคอัพทรีสำหรับที่ไม่ได้ใช้งานพร้อมกันอีกด้วย ในงานวิจัยทั่วไปเกี่ยวกันการสร้างเส้นทางสำรองจะมีสมมติฐานเบื้องต้นอีกอย่างหนึ่งว่าจะไม่มีกรณีที่มีลิงค์มีปัญหาเกินกว่าหนึ่งลิงค์

Recovery scheme ที่นำเสนอในงานวิจัยนี้เป็นในลักษณะที่ข้อมูลสำรองจะสามารถแบ่งส่งเป็นเส้นทางย่อยๆได้ (fractional) โดยในการส่งข้อมูลจะใช้ coding theory เพื่อให้การส่งข้อมูลเป็นไปอย่างถูกต้อง เนื่องจาก Cohen and Nakibly (2010) ได้แสดงให้เห็นแล้วว่าถ้าไม่สามารถแบ่งเส้นทางการส่งข้อมูลเป็นส่วนย่อยได้ จะกลายเป็นปัญหาแบบ NP-hard ดังนั้นงานวิจัยนี้พิจารณาเฉพาะการส่งข้อมูลที่สามารถแยกส่งเป็นส่วนย่อยได้เท่านั้น

วิธีการที่เราเลือกใช้สำหรับการหาเส้นทางส่งข้อมูลสำรองจะเป็นการออกแบบล่วงหน้าแบบออฟไลน์เพื่อสร้างแบคอัพทรี และเมื่อเป็นการทำงานแบบออฟไลน์ ข้อมูลต่างๆของระบบเครือข่ายทั้งหมดจะต้องถูกรวบรวมไว้ให้ได้ก่อนที่จะมีการดำเนินการใดๆ เช่นเดียวกับงานวิจัยของ Cohen and Nakibly (2010)

งานวิจัยในส่วนแรกนี้ เป็นงานวิจัยแรกที่แสดงวิธีการคำนวณหาการจัดสรรแบนด์วิธสำรองสำหรับ recovery scheme ต่าง ๆ ที่ดีที่สุด โดยใช้การโปรแกรมเชิงเส้น ผลจากการทดลอง

บนเครือข่ายจริงสองเครือข่ายและเครือข่ายที่สุ่มขึ้นภายใต้โมเดลของ Barabasi-Albert จากงานวิจัยของ Barabasi et al. (2000) พบว่าประสิทธิภาพของ recovery scheme ทั้งหมดที่นำมาใช้ในการทดลอง มีแนวโน้มใกล้เคียงกับสิ่งที่ Cohen and Nakibly (2010) เคยเสนอไว้ในเครือข่ายยูนิคาสต์ นั่นคือการสร้างเส้นทางสำรองที่เป็นอิสระไม่มีข้อบังคับใดๆจะให้ประสิทธิภาพดีที่สุด แต่เนื่องจากวิธีการดังกล่าวต้องใช้หน่วยความจำมากเพื่อเก็บข้อมูลของเส้นทางสำรอง เราจึงได้นำเสนอ recovery scheme อีกรูปแบบหนึ่งซึ่งได้ประสิทธิภาพใกล้เคียงกันแต่ใช้พื้นที่หน่วยความจำน้อยกว่า 60% รายละเอียดของการทดลองจะอยู่ในส่วนของสรุปผลและวิจารณ์

ส่วนที่สองของงานวิจัยนี้ ศึกษาการสร้างเส้นทางสำหรับการส่งข้อมูลแบบมัลติคาสต์ ในทางทฤษฎีได้เราสามารถจำแนกรูปแบบของการหาเส้นทางได้สองแบบ คือ การหาเส้นทางแบบปรับเปลี่ยนได้ (adaptive routing) และ การหาเส้นทางแบบออบลิเวียส (oblivious routing) การหาเส้นทางในแบบแรกจะเป็นการหาเส้นทางโดยอาศัยข้อมูลของการส่งข้อมูลในเครือข่ายในปัจจุบันเพื่อเป็นตัวกำหนดเส้นทางสำหรับคำขอการส่งข้อมูลที่เข้ามาใหม่ ซึ่งจะตรงกันข้ามกับการส่งข้อมูลแบบออบลิเวียสซึ่งไม่สนใจจำนวนโหลดที่มีอยู่ในเครือข่ายในปัจจุบัน เพราะเส้นทางที่จะส่งข้อมูลถูกกำหนดไว้ก่อนหน้าที่จะเริ่มส่งข้อมูลแล้ว

ประสิทธิภาพของการหาเส้นทางนิยมวัดจาก congestion ซึ่งนิยามว่ามีค่าเท่ากับอัตราส่วนที่มากที่สุดของแบนด์วิธรวมที่ต้องการบนลิงค์หนึ่ง ๆ ต่อแบนด์วิธที่ลิงค์นั้นรองรับได้

แน่นอนว่าการหาเส้นทางแบบปรับเปลี่ยนได้มักให้ผลลัพธ์ของเส้นทางที่มีค่า congestion น้อยที่สุดเท่าที่จะเป็นไปได้ในขณะนั้น แต่การจะปรับเปลี่ยนเส้นทางของข้อมูลที่กำลังส่งอยู่นั้นนอกจากจะกระทำได้ยากแล้ว ยังต้องการการทำงานประสานกันระหว่างอุปกรณ์จำนวนมากในระบบเครือข่าย ในทางกลับกัน ถ้าเราใช้การหาเส้นทางแบบออบลิเวียสเราสามารถวางแผนทุกอย่างไว้ล่วงหน้า และไม่จำเป็นต้องปรับเปลี่ยนเส้นทางในการส่งข้อมูลใด ๆ เลย เมื่อได้เริ่มใช้งานแล้ว เมื่อดูโดยผิวเผินแล้ว การหาเส้นทางแบบออบลิเวียส แม้จะสะดวกในแง่ของการนำไปประยุกต์ใช้งาน แต่ประสิทธิภาพของเส้นทางที่หาได้ ย่อมไม่มีทางจะมีคุณภาพเทียบเท่ากับการส่งข้อมูลแบบปรับเปลี่ยนได้

อย่างไรก็ตาม งานที่สำคัญมากของ Räcke (2002) ได้แสดงให้เห็นแล้วว่าสำหรับการส่งข้อมูลแบบยูนิคาสต์ การหาเส้นทางแบบออบลิเวียสสามารถทำได้โดยมีคุณภาพใกล้เคียงกับการหา

เส้นทางแบบปรับเปลี่ยนได้ ทำให้การใช้การหาเส้นทางแบบออบลิเวียสกลับมาน่าสนใจอีกครั้งหนึ่ง ต่อมาไม่นาน Azar et al. (2003) ก็ได้มีงานวิจัยที่แสดงว่าสามารถวางแผนการหาเส้นทางแบบออบลิเวียส ที่ได้ผลลัพธ์ที่ดีที่สุดที่สามารถทำได้ใน polynomial time โดยใช้เทคนิค Ellipsoid เข้ามาช่วย แต่การใช้เทคนิค Ellipsoid จะค่อนข้างจะเป็นในทางทฤษฎี ดังนั้น Applegate and Cohen (2001) ได้แสดงการใช้เทคนิค primal-dual เพื่อแปลง separation oracle ให้อยู่ในรูปที่สามารถใช้การโปรแกรมเชิงเส้นแก้ปัญหาได้โดยตรง

งานวิจัยในส่วนที่สองของเราพยายามที่จะประยุกต์ใช้งานแนวคิดของการหาเส้นทางแบบออบลิเวียสกับการส่งข้อมูลแบบมัลติคาสต์

เราเริ่มโดยการพิสูจน์ว่า แท้จริงแล้ว ปัญหาการหาเส้นทางแบบออบลิเวียสสำหรับการส่งข้อมูลแบบมัลติคาสต์นั้น สมมูลกับการหาเส้นทางแบบออบลิเวียส์สำหรับการส่งข้อมูลแบบยูนิคาสต์ กล่าวคือ เราได้พิสูจน์ว่า ผลลัพธ์ที่ดีที่สุดของการหาเส้นทางในเครือข่ายแบบยูนิคาสต์ ก็เป็นเส้นทางที่ดีที่สุดในกรณีของการส่งข้อมูลแบบมัลติคาสต์ด้วย

แม้ว่าการหาเส้นทางแบบออบลิเวียสจะถูกศึกษาอย่างมากในทางทฤษฎี แต่ในทางปฏิบัติจริงนั้น การหาเส้นทางแบบออบลิเวียสแทบไม่ได้ถูกใช้เลย สาเหตุหนึ่งของช่องว่างระหว่างทฤษฎีและการปฏิบัตินี้เกี่ยวข้องกับขอบเขตของการรับประกันคุณภาพของการหาเส้นทางแบบออบลิเวียส

ผลลัพธ์ของการหาเส้นทางแบบออบลิเวียสจากงานของ Azar et al. (2003) รับประกันว่าสำหรับ**ทุก ๆ รูปแบบ**ของความต้องการในการส่งข้อมูล เส้นทางที่วางแผนไว้ จะมีอัตราส่วนระหว่างค่า congestion โดยใช้เส้นทางที่หาไว้ก่อนนี้เทียบกับ congestion ที่ได้จากการหาเส้นทางแบบปรับเปลี่ยนได้ มีค่าน้อยที่สุดเท่าที่จะทำได้

อย่างไรก็ตาม เนื่องจากขอบเขตการรับประกันดังกล่าวครอบคลุมถึงรูปแบบความต้องการใด ๆ ก็ได้ ผลลัพธ์ที่ได้อาจจะไม่ดีพอเมื่อเราพิจารณาเฉพาะรูปแบบความต้องการที่เกิดขึ้นจริงในระบบ

ในงานวิจัยโดย Kodialam and Sengupta (2004) พยายามที่จะจำกัดขอบเขตของรูปแบบความต้องการ โดยใส่เงื่อนไขระบุขนาดแบนด์วิธที่มากที่สุดที่จะส่งออกและรับเข้าโดยอุปกรณ์ใด ๆ ในระบบเครือข่าย เงื่อนไขนี้สอดคล้องกับรูปแบบการส่งข้อมูลแบบสองรอบ (two-phase routing) ที่พัฒนาโดยผู้เขียน สำหรับการวางแผนเส้นทางบนเครือข่ายที่ความต้องการแปรเปลี่ยนได้มาก

สังเกตว่าข้อมูลที่ใช้ในการระบุเงื่อนไขโดย Kodialam and Sengupta (2004) จะสนใจเฉพาะข้อมูลที่เกี่ยวข้องกับอุปกรณ์แต่ละตัว โดยไม่เกี่ยวข้องกับอุปกรณ์อื่น ๆ งานวิจัยชิ้นนี้ได้นำเสนอการกำหนดขอบเขตของความต้องการที่จะใช้พิจารณาในการหาเส้นทางแบบออบลิเวียส โดยใช้การสุ่มตัวอย่างของความต้องการใช้งานที่เกิดขึ้นจริงในระบบมาจำนวนหนึ่ง ข้อมูลที่ใช้ในการระบุขอบเขตนี้แสดงความสัมพันธ์ระหว่างอุปกรณ์ต่าง ๆ ในระบบเครือข่ายที่ชัดเจนมากขึ้น เช่น ในการเก็บข้อมูลครั้งหนึ่ง ๆ เราจะเห็นว่าอุปกรณ์ใดส่งข้อมูลให้อุปกรณ์ใดด้วย

เมื่อนำอัลกอริทึมสำหรับวางแผนการหาเส้นทางแบบออบลิเวียสที่ดีที่สุดมาประยุกต์ใช้งานในการวางแผนการหาเส้นทางที่ดีที่สุดสำหรับความต้องการที่สุ่มมาทำให้เราสามารถวางแผนเส้นทางที่ดีที่สุดสำหรับความต้องการหลาย ๆ แบบพร้อม ๆ กัน

เราได้พิสูจน์ทฤษฎีบทที่รับประกันคุณภาพของเส้นทางที่วางแผนได้ และได้ทดลองเพื่อยืนยันถึงประสิทธิภาพของเส้นทางที่คำนวณได้ นอกจากนี้เรายังมีตัวอย่างรวมถึงผลการทดลองที่ชี้ให้เห็นว่า ขอบเขตของความต้องการที่ใช้โดย Kodialam and Sengupta (2004) นั้นกว้างไป และการใช้ความต้องการที่สุ่มมาเป็นขอบเขตทำให้เราได้แผนการหาเส้นทางเฉพาะเจาะจงกับสถานการณ์ที่อาจจะเกิดขึ้นจริงมากกว่า และทำให้ได้ค่า congestion ที่ดีขึ้น

วัตถุประสงค์

1. เพื่อศึกษาและออกแบบวิธีการจองแบนด์วิธสำหรับการสร้างแบคอัพทรีบนเครือข่ายมัลติคาสต์เพื่อใช้ส่งข้อมูลในกรณีเกิดปัญหาขึ้นกับลิงค์ในเครือข่าย
2. เพื่อศึกษาลักษณะความสัมพันธ์ของการส่งข้อมูลแบบออบลิเวียสบนเครือข่ายยูนิคาสต์และมัลติคาสต์
3. เพื่อพัฒนาอัลกอริทึมสำหรับหาเส้นทางแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ โดยใช้ข้อมูลภายในเครือข่ายที่ละเอียดมากขึ้น

การตรวจเอกสาร

เราจะทำการตรวจเอกสารในส่วนของการสร้างเส้นทางสำรองในเครือข่ายยูนิคาสต์ จากนั้นจะเป็นการตรวจเอกสารในเรื่องที่เกี่ยวกับการหาเส้นทางสำรองในเครือข่ายมัลติคาสต์ จากนั้นจะเป็นการตรวจเอกสารเกี่ยวกับงานวิจัยในส่วนที่สองซึ่งเป็นการตรวจเอกสารเกี่ยวกับการสร้างเส้นทางการส่งข้อมูลในเครือข่ายยูนิคาสต์แบบออบลิเวียส แต่เนื่องจากเราใช้กราฟเครือข่ายที่ได้จากการสุ่ม ดังนั้นเราจะเริ่มตรวจเอกสารในส่วนของโมเดลที่ใช้ในการสร้างกราฟก่อนเป็นอันดับแรก

โมเดลของเครือข่าย

ในงานวิจัยนี้จะในส่วนหลักจะเป็นการจำลองการทำงานของการส่งข้อมูลในเครือข่าย ซึ่งใช้เครือข่ายอยู่หลายแบบ หนึ่งในนั้นก็คือเครือข่ายแบบสุ่ม เราจะอาศัยโมเดลของ Barabasi-Albert ในการสร้างกราฟเครือข่ายแบบสุ่มนั้นขึ้นมา ซึ่งโมเดลนี้จะมีอัลกอริทึมในการสร้างกราฟของเครือข่ายซึ่งมีการกระจายของดีกรีเป็นไปตามกฎ power-law

การสร้างเส้นทางสำรองในเครือข่ายยูนิคาสต์

ได้มีการพัฒนาเทคนิควิธีการในการสร้างแบคอัพทรีในเครือข่ายมัลติคาสต์ ดังเช่นในงานวิจัยของ Liu et al. (2001), Bhatia et al. (2005), Cohen and Nakibly (2010) เพื่อที่จะรับประกันว่าจะมีแบนด์วิธเพียงพอสำหรับการส่งข้อมูลสำรองเมื่อเกิดปัญหาขึ้นกับลิงค์ในระบบเครือข่าย ซึ่งระบบจะทำการสร้างเส้นทางการส่งข้อมูลเป็นสองชุดก็คือเซอร์วิสพาธ หรือไพรมารีพาธที่ใช้สำหรับส่งข้อมูลในกรณีปกติ และเส้นทางส่งข้อมูลสำรองเอาไว้ส่งข้อมูลในกรณีที่มีปัญหาซึ่งก็คือแบคอัพพาธ ซึ่งโครงสร้างหรือรูปแบบจะเปลี่ยนไปซึ่งขึ้นอยู่กับ recovery scheme นั่นเอง

ในปัจจุบันได้มีการนำเสนอ recovery scheme หลายรูปแบบซึ่งแต่ละ recovery scheme ก็จะมีลักษณะเงื่อนไขในการสร้างแบคอัพพาธที่ต่างกัน เพื่อรองรับการเกิดปัญหาขึ้นกับทุกๆลิงค์ที่อยู่ใน ไพรมารีพาธ ที่มีอยู่ในระบบซึ่งหมายความว่าอาจจะมีแบคอัพพาธ มากกว่าหนึ่งชุดสำหรับไพรมารีพาธหนึ่งเส้นทางก็เป็นได้ การสร้างแบคอัพพาธ โดยทั่วไปจะมีเป้าหมายหลักอยู่สองอย่างคือเพื่อให้มีการใช้แบนด์วิธน้อยที่สุดซึ่งจะมีสมมติฐานหลักว่าขนาดของแบนด์วิธบนลิงค์มีค่าไม่จำกัด

หรือไม่ก็มีขนาดใหญ่มากๆ ส่วนอีกแบบหนึ่งจะเป็นการสร้างแบคอัพพาธบนลิงค์ที่มีขนาดจำกัด โดยมีเป้าหมายให้สามารถส่งข้อมูลภายในระบบได้มากที่สุด

ในงานวิจัยที่มีเป้าหมายเพื่อที่จะทำให้การใช้แบนด์วิธน้อยที่สุดหรือ Spare Capacity Allocation (SCA) วิธีการนี้ซึ่งใช้ขนาดของแบนด์วิธรวมที่ใช้สำหรับแบคอัพพาธเป็นตัววัดประสิทธิภาพ ซึ่ง Shyur et al. (1999) ได้เสนอการแก้ปัญหาโดยการใช้ โปรแกรมเชิงเส้นแบบจำนวนเต็ม (Interger Linear Programming) หลังจากนั้น Liu et al. (2001) ได้เสนอวิธีการแบบ heuristics เพื่อหาแบคอัพพาธบนเครือข่ายแบบ mesh โดยใช้การหาค่าแบบประมาณ (approximate) ซึ่งมีชื่อว่า successive survivable routing หรือ SSR ในขั้นตอนแรกจะมีการสร้าง spare provision matrix (SPM) เพื่อรองรับการส่งข้อมูลสำรองของคู่ต้นทาง-ปลายทางของการรับส่งข้อมูลแต่ละคู่ จากนั้นจะนำเมทริกซ์ที่ได้มาคำนวณหาเส้นทางสำรองของการส่งข้อมูลโดยรวมอีกครั้งหนึ่ง ในขณะที่ Kennington and Lewis (2001) ได้เสนอการสร้างแบคอัพพาธ โดยการใช้อัลกอริทึมที่ชื่อว่า branch-and-bound ร่วมกับการทำงานของโปรแกรมเชิงเส้น

เนื่องจากสมมติฐานที่ว่าแบนด์วิธบนลิงค์ต่างๆมีค่าไม่จำกัดไม่ตั้งอยู่บนพื้นฐานของความเป็นจริง ดังนั้น Cohen and Nakibly (2010) ได้ให้เหตุผลว่าการใช้ SCA เป็นตัววัดประสิทธิภาพจึงไม่เหมาะสม และได้เสนอให้ใช้ปริมาณของข้อมูลที่สามารถส่งได้ในเครือข่ายเป็นตัววัดประสิทธิภาพแทน ซึ่งจะสอดคล้องกับความเป็นจริงและสามารถนำไปใช้งานกับระบบเครือข่ายที่มีอยู่แล้วได้ ในการวัดประสิทธิภาพจากปริมาณข้อมูลที่สามารถส่งในเครือข่ายได้ จะวัดจำนวนข้อมูลทั้งหมดที่สามารถส่งได้หลังจากบางส่วนของช่องสัญญาณถูกจองไว้สำหรับการสร้างไพรมารีพาธและแบคอัพพาธ

ภายใต้การใช้จำนวนข้อมูลเป็นตัวชี้วัดประสิทธิภาพของ recovery scheme นี้ Bhatia et al. (2008) ได้เสนอ Local recovery scheme (LR) ซึ่งใช้การโปรแกรมเชิงเส้นเพื่อสร้างแบคอัพพาธ ซึ่งอัลกอริทึมที่นำเสนอสามารถหาคำตอบได้ในโพลิโนเมียล หรือ Fully Polynomial Time Approximation Scheme (FPTAS) สำหรับการสร้างแบคอัพพาธในกรณีที่ลิงค์หรืออุปกรณ์เกิดปัญหา โดยสร้างโปแกรมเชิงเส้นรองรับปัญหาทั้งสองกรณีรวมไปถึงแสดงโค๊ดเสมือน (pseudo-code) เพื่อแสดงการทำงานของโปรแกรมเชิงเส้นดังกล่าว พร้อมทั้งแสดงการพิสูจน์ทฤษฎีบทเพื่อรองรับผลที่ได้ หลังจากนั้น Cohen and Nakibly (2010) ได้จัดระบบของ recovery scheme ในเครือข่ายยูนิคาสต์ และแสดงให้เห็นว่าการจัดสรรแบนด์วิธสำหรับแต่ละ recovery scheme ที่ใช้วัด

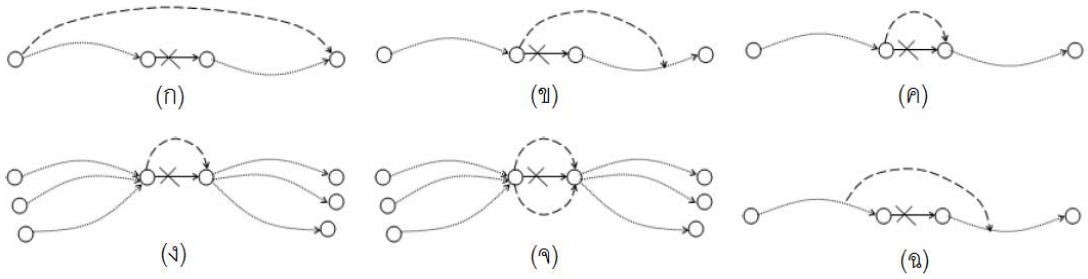
ประสิทธิภาพจากจำนวนของข้อมูลที่สามารถส่งได้ในเครือข่ายสามารถทำได้โดยการโปรแกรมเชิงเส้น

นอกจากนี้ยังมีงานวิจัยที่เกี่ยวข้องอีกหลายงานที่พิจารณาในแง่มุมอื่นๆของปัญหา โดย Alicherry and Bhatia (2007) ได้เสนอวิธีที่ชื่อว่า k-facility local recovery scheme ซึ่งเป็นวิธีแบบ heuristics ซึ่งสร้างแบคอัพพาธโดยที่มีจุดมุ่งหมายเพื่อให้ใช้แบนด์วิธน้อยที่สุด ในขณะที่ Lai et al. (2008) ได้เสนอวิธีการ bypass tunnel สำหรับแต่คู่ของ source-destination โดยงานวิจัยนี้ให้ความสนใจที่เงื่อนไขของคุณภาพของบริการหลักเมื่อจะต้องเปลี่ยนเส้นทางส่งข้อมูลไปใช้แบคอัพพาธ

งานวิจัยของ Cohen and Nakibly (2010) ได้มีการจำแนกรูปแบบการกู้คืนในเครือข่ายเอ็มพีแอลเอส โดยมีวัตถุประสงค์เพื่อให้มีประสิทธิภาพการส่งข้อมูลในเครือข่ายได้มากที่สุดโดยมองว่าการใช้ปริมาณข้อมูลที่ส่งได้ในเครือข่ายจะเป็นรูปแบบที่นำไปใช้ได้จริงมากกว่า และได้จำแนกการสร้างแบคอัพพาธในเครือข่ายแบบยูนิคาสต์หกชนิดตามลักษณะการทำงานดังนี้

- Global recovery scheme(GR) (ภาพที่ 1(ก)): ในแต่ละไพรมารีพาธจะสร้างแบคอัพพาธซึ่งมีจุดต้นทางและปลายทางเหมือนกัน แต่แบคอัพพาธจะไม่มีการใช้ลิงค์ที่ใช้ไปแล้วในไพรมารีพาธ
- Local recovery scheme(LR) (ภาพที่ 1(ข)): จะประกอบไปด้วยจำนวนแบคอัพพาธที่มากกว่า GR โดยจะมีจำนวนของแบคอัพพาธเท่ากับจำนวนลิงค์ที่ประกอบอยู่ในไพรมารีพาธ สำหรับการส่งข้อมูลสำรองจะเริ่มต้นส่งจากโหนดที่อยู่ก่อนลิงค์ที่เกิดปัญหา ไปยังจุดปลายทาง โดยแบคอัพพาธอาจจะใช้ลิงค์ในไพรมารีพาธ (ที่ไม่ได้เกิดปัญหา) ร่วมด้วยก็ได้
- Restricted local recovery scheme(RLR) (ภาพที่ 1(ค)): จะเป็นการสร้างแบคอัพพาธเพื่อส่งข้อมูลสำรองจากโหนดทางปลายด้านหนึ่งของลิงค์ที่มีปัญหา ไปยังโหนดอีกปลายด้านหนึ่งโดยไม่ผ่านลิงค์ที่มีปัญหานั้น
- Facility local recovery scheme(FLR) (ภาพที่ 1(ง)): จะคล้ายกับ RLR แต่แบคอัพพาธจะถูกสร้างเพื่อรองรับการส่งข้อมูลบนด้านนั้นทั้งหมด ต่างจาก RLR ที่จะสร้างแบคอัพพาธหนึ่งชุดสำหรับแต่ละไพรมารีพาธที่มีการส่งข้อมูลผ่านลิงค์ที่มีปัญหานั้น
- k-facility local recovery scheme(k-FLR) (ภาพที่ 1(จ)): จะเป็นการพัฒนาจาก FLR โดยจะมีการสร้างแบคอัพพาธมากกว่าหนึ่งชุดสำหรับแต่ละลิงค์ที่มีปัญหา

- Unrestricted recovery scheme(UR) (ภาพที่ 1(ฉ)): จะเป็นวิธีที่ยืดหยุ่นมากที่สุด เนื่องจากการสร้างแบคอัพพาธสามารถเกิดจากโหนดใดก็ได้ในไพรมารีพาธที่อยู่ก่อนลิงค์ที่มีปัญหาไปยังโหนดอีกฝั่งหนึ่งของลิงค์นั้นโดยที่ยังสามารถใช้ลิงค์ที่เป็นส่วนประกอบของ ไพรมารีพาธได้อีกด้วย หรือกล่าวอีกอย่างหนึ่งว่าแบคอัพพาธจะสร้างอย่างไรก็ได้ให้สามารถส่งข้อมูลสำรองไปยังปลายทางเท่านั้นก็พอ



ภาพที่ 1 recovery scheme ในยูนิคาสต์ (ก) GR (ข) LR (ค) RLR (ง) FLR (จ) k-FLR (ฉ) UR

ภาพที่ 1 แสดงรูปแบบการทำงานของ recovery scheme ที่นำเสนอในงานวิจัยของ Cohen and Nakibly (2010) ไพรมารีพาธจะถูกแทนด้วยเส้นทึบ และแบคอัพพาธแสดงโดยเส้นประ

ในงานวิจัยของ Cohen and Nakibly (2010) ได้แบ่งลักษณะของการส่งข้อมูลในเครือข่ายที่สนใจออกเป็นสองชนิดก็คือในเครือข่ายที่สามารถส่งข้อมูลเป็นส่วนย่อยได้ (S) และเครือข่ายที่ไม่สามารถส่งข้อมูลเป็นส่วนย่อย (U) ซึ่งจะมีผลต่อการความแตกต่างของวิธีการหาเส้นทางสำรองด้วย

ในงานวิจัยยังได้แสดงให้เห็นว่าถ้าเป็นปัญหาที่ไม่สามารถแยกส่วนของ flow ที่จะส่งได้ทำให้หลายๆ recovery scheme เป็นปัญหาแบบ NP-hard ที่จะสามารถกะประมาณคำตอบไม่ได้ดีกว่า $|E|^{1/2-\varepsilon}$ ซึ่ง E เป็นเซ็ตของเส้นเชื่อมต่อภายในระบบเครือข่าย และเมื่อปัญหาในเครือข่ายมัลติคาสต์เป็นซุปเปอร์เซ็ต(superset) ของปัญหาในเครือข่ายยูนิคาสต์ จึงทำให้การหาคำตอบภายใต้เครือข่ายแบบมัลติคาสต์จึงซับซ้อนตามไปด้วย

นอกจากนั้นยังแยกกรณีที่ต้องจัดสรรแบนด์วิธทั้งสำหรับเส้นทางหลักและเส้นทางสำรองกับกรณีที่กำหนดเส้นทางหลักมาให้(P) ซึ่งค่าดังตารางที่ 1 โดยแต่ละค่าที่แสดงในตารางจะแสดงจะ

แสดงให้ทราบว่าปัญหานั้นเป็น NP-Complete หรือ P หรือไม่ทราบค่า โดยปัญหาที่เป็น NP-Complete ก็จะแสดงขอบด้านล่างด้วยค่าประมาณจากวิธีการแบบ heuristics

โดยได้ทำการทดลองบนกราฟเครือข่ายแบบสุ่มที่ได้จากโปรแกรม BRITE Simulator ที่มีค่าพารามิเตอร์สามค่าคือ เครือข่ายที่ประกอบด้วย 20 โหนดมีค่าเฉลี่ยดีกรี 3, เครือข่ายที่ประกอบด้วย 20 โหนดมีค่าเฉลี่ยดีกรี 5 และ เครือข่ายที่ประกอบด้วย 40 โหนดมีค่าดีกรีเฉลี่ย 3 ซึ่งถือว่าเป็นกราฟที่มีความหนาแน่นของด้านสูงโดยเฉพาะกราฟในรูปแบบที่สอง และจากการทดลองสรุปได้ว่า UR, GR และ LR ให้ backup path ที่มีประสิทธิภาพที่ต่างกันแบบละประมาณ 5% และ RLR ให้ backup path ที่มีประสิทธิภาพต่ำที่สุด ซึ่งถ้ากราฟเครือข่ายยิ่งมีความหนาแน่นของด้านสูงขึ้นค่าประสิทธิภาพของแต่ละ recovery scheme ก็ยิ่งใกล้เคียงกันมากขึ้นอีกด้วย

ตารางที่ 1 ค่า complexity จากการคำนวณในแต่ละวิธี

รูปแบบการกู้คืน	S-PRFP	U-PRFP	S-RFP	U-RFP
GR	P	NP-C	?	NP-C
LR	P	NP-C	?	NP-C
RLR	P	NP-C	P	NP-C
FLR	P	NP-C	P	NP-C
k-FLR	P	NP-C	P	NP-C
UR	P	NP-C	?	NP-C

การสร้างเส้นทางสำรองในเครือข่ายมัลติคาสต์

ในระบบเครือข่ายที่มีการส่งข้อมูลจากเครื่องต้นทางหรือเครื่องเซิร์ฟเวอร์ไปยังกลุ่มของเครื่องปลายทางหรือเรียกว่าการส่งข้อมูลแบบมัลติคาสต์ จะเริ่มต้นโดยการสร้างเส้นทางเบื้องต้นที่ใช้ในการส่งข้อมูลเรียกว่าเซอร์วิสทรี(service tree) หรือไพรมารีทรี ซึ่งถ้าเกิดปัญหาขึ้นกับลิงค์ในเครือข่ายก็จะมีผลกระทบกับการส่งข้อมูลในไพรมารีทรี เช่นเดียวกันกับในยูนิคาสต์ ดังนั้นจึงจำเป็นจะต้องจัดสรรแบนด์วิธไว้สำหรับกรณีส่งข้อมูลสำรองในเครือข่ายเช่นกัน

การศึกษาการจัดสรรแบนด์วิธสำหรับส่งข้อมูลสำรองในเครือข่ายมัลติคาสต์ส่วนใหญ่จะทำตามแนวทางที่ปฏิบัติกันในเครือข่ายยูนิคาสต์ ซึ่งเราได้ศึกษางานวิจัยที่ผ่านมาพบว่าจะใช้ประสิทธิภาพของการจัดสรรแบนด์วิธที่ถูกจัดสรรไว้สำหรับการสร้างเส้นทางสำรองเป็นตัวชี้วัดหลัก

ในงานวิจัยของ Fei et al. (2001) เสนอให้สร้างแบคอัพทรีต่างหากอีกหนึ่งชุดที่ไม่มีการใช้ลิงค์ร่วมกับไพรมารีทรี เพื่อที่ว่าเมื่อเกิดปัญหาขึ้นกับไพรมารีทรีก็จะสามารถส่งข้อมูลสำรองไปในแบคอัพทรีได้ทันที ซึ่งให้ชื่อวิธีนี้ว่า dual-tree ซึ่งเริ่มต้นโดยไพรมารีทรีจากอัลกอริทึมแบบ OSPF และสร้างแบคอัพทรี โดยการเลือกใช้เส้นเชื่อมต่อที่ยังไม่ได้ถูกใช้ไปในไพรมารีทรี ดังนั้นจึงแน่ใจได้ว่าแบคอัพทรีและไพรมารีทรีจะไม่มีการใช้ลิงค์ร่วมกัน

Kodialam and Lakshman (2002) ได้เสนออัลกอริทึมแบบออนไลน์ ซึ่งใช้ heuristics เพื่อคำนวณหาแบนด์วิธที่จะต้องใช้สำหรับการส่งข้อมูลสำรอง โดยใช้ RLR scheme ซึ่งจะใช้อัลกอริทึมที่มีชื่อว่า Nearest Neighbor First (NNF) เพื่อสร้างไพรมารีทรี จากนั้นจึงสร้างแบคอัพทรีโดยเน้นให้มีการใช้งานแบนด์วิธร่วมกัน หลังจากนั้น Aggarwal et al. (2004) ได้เสนอ heuristics ง่ายๆ เพื่อสร้างเซ็ตของเส้นทางจากจุดต้นทางไปยังจุดปลายทางทุกๆจุด สำหรับรองรับการส่งข้อมูลสำรอง

Lau et al. (2005) ได้ริเริ่มที่จะใช้ optimization ในการจัดสรรแบนด์วิธสำหรับใช้ในเส้นทางสำรองภายในเครือข่ายมัลติคาสต์ โดยใช้อัลกอริทึมแบบออนไลน์ประกอบกับการโปรแกรมเชิงเส้นแบบจำนวนเต็มเพื่อจะสร้างเส้นทางสำรองภายใต้ UR scheme ซึ่งทุกๆเส้นทางการส่งข้อมูลภายในไพรมารีทรีที่เกี่ยวข้องกับส่วนของความผิดพลาดไม่ว่าจะเป็นลิงค์หรือว่าอุปกรณ์ การส่งข้อมูลในไพรมารีทรีนั้นจะถูกระงับแล้วเริ่มส่งข้อมูลใหม่ในแบคอัพทรีที่ได้สร้างเอาไว้ และได้นำผลที่ได้ไปเปรียบเทียบกับงานวิจัยของ Kodiaiam and Lakshman (2002) รวมไปถึงงานวิจัยของ Singhal et al. (2003) ซึ่งเสนออัลกอริทึมในการหา optimal path-pair เพื่อทำการสร้างแบคอัพทรีสำหรับเครือข่ายมัลติคาสต์บนเครือข่ายแบบเมช เนื่องจากการแก้สมการโปรแกรมเชิงเส้นแบบจำนวนเต็มจะใช้เวลามาก Lau et al. (2005) ได้ออกแบบอัลกอริทึมแบบออนไลน์เพื่อหาค่าประมาณสำหรับปัญหานี้ และนำมาเปรียบเทียบกับผลที่ได้จากส่วนก่อนหน้านี้อีกด้วย ในท้ายที่สุด Li et al. (2006) ได้นำเสนอ RLR scheme เพื่อรองรับกรณีที่เกิดปัญหากับอุปกรณ์ โดย

เสนอวิธีการสร้างแบคอัพทรี โดยปรับปรุงจากวิธี backup path local repair โดยใช้อัลกอริทึมแบบ heuristics เพื่อให้การมีการแชร์แบนด์วิธกันระหว่างภายในแบคอัพทรี

การสร้างเส้นทางการส่งข้อมูลแบบออบลิเวียส

สำหรับการหาเส้นทางแบบออบลิเวียสในเครือข่ายแบบยูนิคาสต์ Azar et al. (2003) ได้แสดงให้เห็นว่าการหาเส้นทางที่ดีที่สุดในรูปแบบออบลิเวียสในเครือข่ายยูนิคาสต์สามารถทำได้ในเวลาโพลิโนเมียลโดยใช้เทคนิคของอีลิปซอยด์ จากนั้น Applegate and Cohen (2003) ได้จัดรูปแบบของโปรแกรมเชิงเส้นใหม่ โดยใช้เทคนิค primal-dual แปลง separation oracle ในงานวิจัยของ Azar et al. (2003) ทำให้อยู่ในรูปของ dual ทำให้โปรแกรมเชิงเส้นที่ได้สามารถหาคำตอบได้ง่ายยิ่งขึ้น ซึ่งวิธีนี้สามารถใช้ได้ทั้งเครือข่ายแบบมีทิศทาง(directed network) และเครือข่ายแบบไม่ได้ระบุทิศทาง(undirected network)

Räcke (2008) ได้แสดงให้เห็นว่า bound ของ competitive ratio ของเส้นทางการส่งข้อมูลแบบออบลิเวียสบนเครือข่ายที่ไม่มีการกำหนดทิศทางสามารถทำได้ใน O(log n) เมื่อ n เป็นจำนวนของอุปกรณ์ในระบบเครือข่ายหรือก็คือจำนวนโหนดในเครือข่าย ซึ่งปรับปรุงมาจากงานวิจัยก่อนหน้าซึ่งมีขอบเขตบนของ competitive ratio เป็น $O(\log^3 n)$

Kodialam and Sengupta (2004) ได้เสนอวิธีการหาเส้นทางการส่งข้อมูลในเครือข่ายยูนิคาสต์แบบออบลิเวียส โดยการเก็บข้อมูล ingress และ egress เพิ่มเติมเพื่อใช้ในการตัดสินใจในการสร้างเส้นทาง เนื่องจากการหาเส้นทางแบบออบลิเวียสโดยปกติจะต้องพิจารณากรณีของการส่งข้อมูลทั้งหมดที่เป็นไปได้ในเครือข่ายซึ่งทำให้ต้องมีการคำนวณในปริมาณที่มากซึ่งบางครั้งมากเกินความจำเป็น ซึ่ง Kodialam and Sengupta (2004) เสนอให้สร้างเส้นทางส่งข้อมูลแบบออบลิเวียสจากข้อมูล ingress และ egress ที่สามารถรวบรวมได้จากเครือข่าย เพื่อนำไปสร้างเส้นทางการส่งข้อมูลแบบออบลิเวียสที่สามารถรับประกันค่า congestion สูงสุดที่จะเกิดขึ้นได้ ซึ่งเส้นทางที่สร้างขึ้นจะใช้เทคนิคที่เรียกว่าการส่งข้อมูลแบบสองรอบ โดยจะมีการส่งข้อมูลเป็นสองเฟส โดยเมื่อโหนดต้องการส่งข้อมูลจะเริ่มด้วยการส่งข้อมูลไปที่ intermediate node ก่อน แล้ว intermediate node จะส่งข้อมูลนั้นไปยังปลายทางอีกครั้ง

โดย intermediate node ที่ใช้ในการส่งข้อมูลอาจจะมีตั้งแต่หนึ่งโหนดไปจนถึงเท่ากับจำนวนของโหนดในเครือข่าย ดังนั้นข้อมูลที่จะถูกส่งไปยังปลายทาง R_i จะถูกแยกเป็นส่วนย่อยๆ เพื่อที่จะส่งไปยังแต่ละ intermediate node ก่อน โดยกำหนดค่าอัตราส่วนการแบ่งข้อมูลสำหรับแต่ละ intermediate node $\alpha_1, \alpha_2, \ldots, \alpha_n$ โดยที่ $\sum_{i=1}^{n} \alpha_i = 1$ ซึ่งหมายถึงสัดส่วนที่จะต้องส่งข้อมูลให้แต่ละ intermediate node ในเฟสที่สอง intermidate node ก็จะส่งข้อมูลเหล่านั้นไปอย่างปลายทาง ซึ่งให้ค่า congestion ที่ดีสำหรับการส่งข้อมูลแบบออบลิเวียส โดยใช้การโปรแกรมเชิงเส้นในการหาผลลัพธ์

สำหรับโปรแกรมประยุกต์ทั่วไป เช่น การส่งข้อมูลมัลติมีเดียและการแจกจ่ายแฟ้มข้อมูลนั้น เนื่องจากเป็นการส่งข้อมูลชุดเดียวกันไปยังปลายทางหลายๆที่ ดังนั้นเราอาจจะมองว่าเป็นการส่งข้อมูลแบบมัลติคาสต์ซึ่งจะสามารถใช้ประโยชน์จากแบนด์วิธที่สามารถใช้งานร่วมกันได้ทำให้ประสิทธิภาพในการใช้งานเครือข่ายดียิ่งขึ้น ซึ่ง Awerbuch and Azar (1995) ได้เคยนำเสนอรูปแบบการหาเส้นทางซึ่งเป็นอัลกอริทึมแบบออนไลน์ซึ่งมีขอบจำกัดเป็น $O(\log^2 n)$ สำหรับการหาเส้นทางในเครือข่ายมัลติคาสต์แบบไม่มีทิศทางที่มี n โหนด

อุปกรณ์และวิธีการ

อุปกรณ์

1. เครื่องเซิร์ฟเวอร์ Linux server , 6 GB of RAM
2. เครื่องคอมพิวเตอร์ Intel® Pentium® D CPU 3.40 GHz, 2 GB of RAM
3. โปรแกรม glpsol (http://www.gnu.org/software/glpk/)
4. โปรแกรม gnuplot (http://www.gnuplot.info/)

วิธีการ

ในงานวิจัยนี้มีส่วนของงานวิจัยที่แบ่งเป็นสองส่วนหลัก คือการสร้างแบคอัพทรีในเครือข่ายแบบมัลติคาสต์ และการหาเส้นทางแบบออบลิเวียส ในเครือข่ายมัลติคาสต์ ซึ่งจะแสดงรายละเอียดวิธีการดังต่อไปนี้

1. การสร้างเส้นทางสำรองบนเครือข่ายมัลติคาสต์

เนื่องจากถ้าเรากำหนดให้การส่งข้อมูลจะต้องส่งไปในเส้นทางเดียวกันทั้งหมดไม่สามารถแยกเป็นส่วนย่อยได้ ปัญหาในการจัดสรรแบนด์วิธสำหรับเส้นทางสำรองจะเป็นปัญหาแบบ NP-hard ดังที่ Cohen and Nakibly (2010) ได้แสดงไว้แล้ว ดังนั้นงานวิจัยนี้พิจารณาเฉพาะการส่งข้อมูลที่สามารถแยกส่วนได้เท่านั้น

1.1 ศึกษาและออกแบบโมเดลของการสร้างเส้นทางสำรองในเครือข่ายมัลติคาสต์

เราจะสร้างเส้นทางสำรองสำหรับเครือข่ายมัลติคาสต์ได้โดยเริ่มจากสมมติฐานที่ว่าไพรมารีทรี จะต้องมีการกำหนดมาให้ก่อนแล้ว จากนั้นเราจะใช้การโปรแกรมเชิงเส้นแก้สมการเพื่อหาคำตอบที่ดีที่สุด และจาก recover scheme ที่มีการใช้งานในทั้งเครือข่ายยูนิคาสต์ และมัลติคาสต์ เราได้เลือก recovery scheme ที่เหมาะสมจะใช้ในเครือข่ายมัลติคาสต์มากที่สุดมา 4 scheme ด้วยกัน เราจะสรุปแยกเป็นชนิดต่างๆ โดยสังเกตจากลักษณะวิธีการสร้างแบคอัพทรีได้ดังนี้

- Global recover scheme(GR)
- Local recovery scheme(LR)
- Restricted local recovery scheme(RLR)
- Unrestricted recovery scheme(UR)

ซึ่งแต่ละ recovery scheme จะสร้างแบคอัพทรีย่อยๆ เพื่อรองรับกับ failed egde แต่ละ failed edge ที่สามารถเกิดขึ้นได้ ทำให้แต่ละไพรมารีทรีจะมีแบคอัพทรีจำนวนมากจนอาจจะเท่ากับจำนวนของ edge ที่มีอยู่ในไพรมารีทรีก็เป็นได้

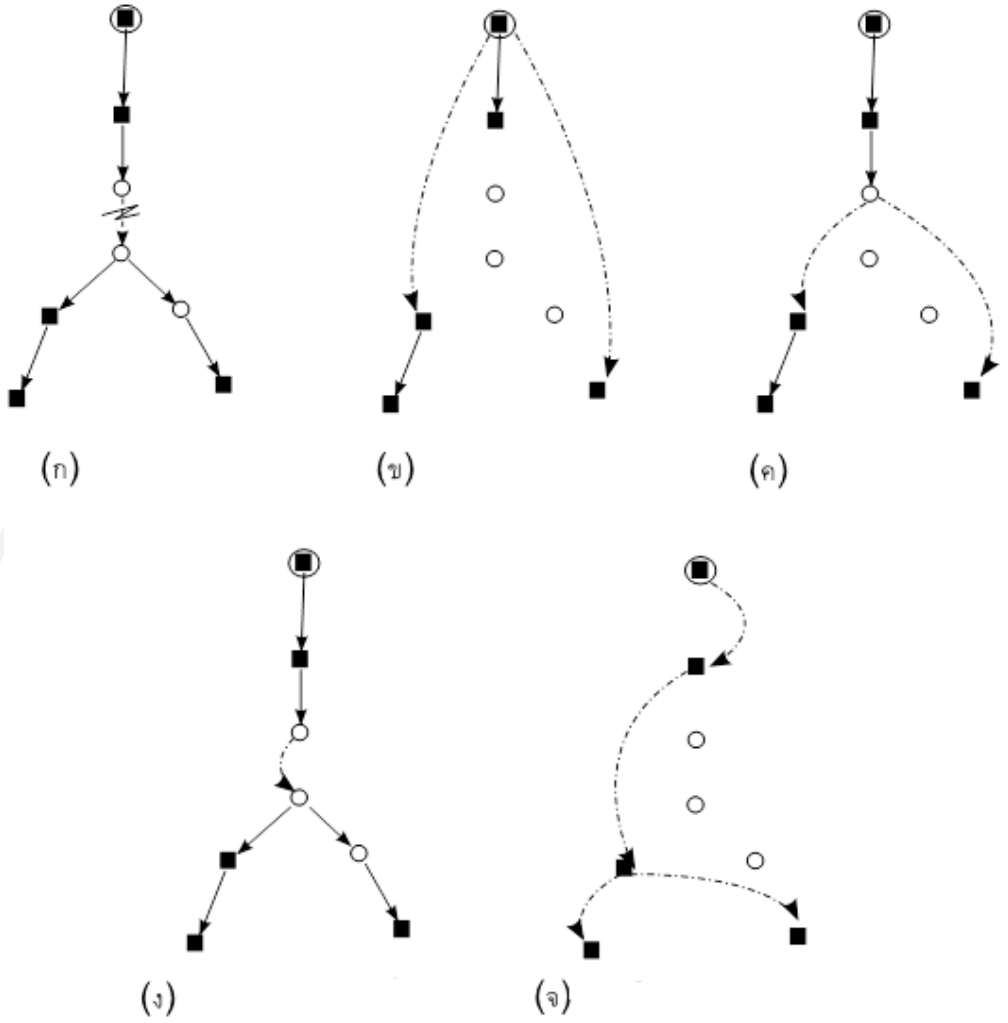
1.1.1 Recovery scheme

1.1.1.1 Global recovery scheme - เป็นการสร้างแบคอัพทรีที่ทุกๆ ชุดจะมี root ของไพรมารีทรีเป็น root node และเทอร์มินอล(terminal)ของแบคอัพทรีจะเป็นเทอร์มินอลของไพรมารีทรีที่ได้รับผลกระทบจาก failed edge โดยตรง ดังแสดงในภาพที่ 2(ข)

1.1.1.2 Local recovery scheme - ความแตกต่างของ LR กับ GR จะอยู่ตรงที่ root node ของแบคอัพทรี จะเป็น node ที่อยู่ติดกับ failed edge ด้านที่เป็นอยู่ใกล้ root node ในไพรมารีทรีหรือเรียกอีกอย่างหนึ่งว่า immediate upstream node ดังแสดงในภาพที่ 2(ค)

1.1.1.3 Restricted recovery scheme - RLR ใน multicast จะเป็น recovery scheme ที่คล้ายกับ RLR ในยูนิคาสต์มากที่สุด เนื่องจากเป็นการสร้างแบคอัพทรีเพื่อส่งข้อมูลระหว่าง node ปลายทั้งสองด้านของ failed edge โดย root node ของแบคอัพทรีจะเป็น immediate upstream node และเทอร์มินอลเพียงเทอร์มินอลเดียวก็คือจุดปลายอีกด้านของ failed edge หรือเรียกว่า immediate downstream node ดังภาพที่ 2(ง)

1.1.1.4 Unrestricted recovery scheme - การสร้างแบคอัพทรีใน UR จะสร้างแบคอัพทรีที่มี root node เดียวกันกับไพรมารีทรีเช่นเดียวกับ GR แต่เทอร์มินอลของแบคอัพทรีจะเป็นทุกๆ เทอร์มินอลในไพรมารีทรีทั้งหมด ซึ่งลักษณะการทำงานกับ UR ก็คือสำหรับแต่ละ failed edge $\tilde{e}$ ใดๆ UR จะสร้างแบคอัพทรีไปยังเทอร์มินอลทั้งหมดใหม่ โดยมีข้อจำกัดเพียงอย่างเดียวคือจะไม่ใช้ edge $\tilde{e}$ ในแบคอัพทรีที่สร้างขึ้น ดังแสดงในภาพที่ 2(จ)



ภาพที่ 2 แสดง recovery scheme แบบต่างๆ: (ก)แสดงตำแหน่ง failed edge ที่เกิดขึ้น (ข) GR (ค) LR (ง) RLR (จ) UR

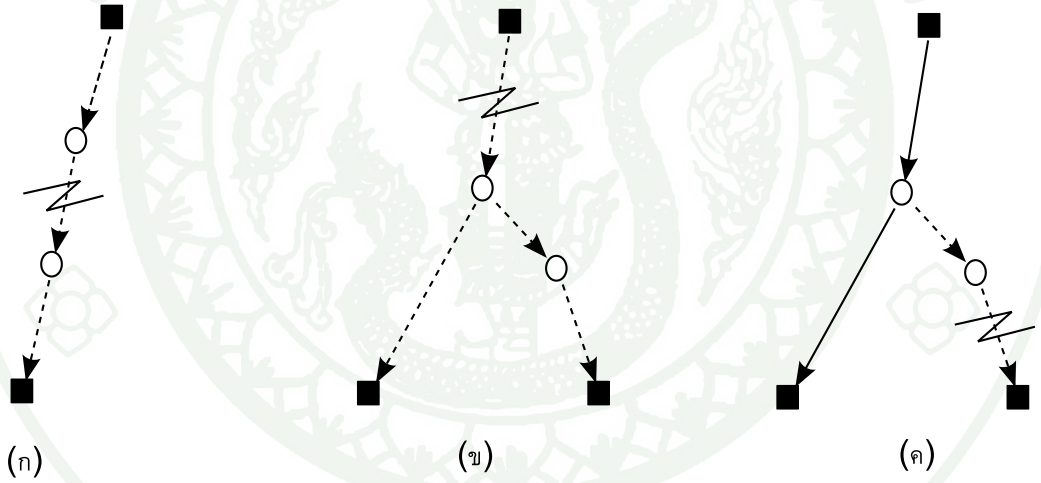
จากนิยามการทำงานของ recovery scheme แต่ละชนิดเราจำเป็นต้องนิยาม suspended part เพื่อให้สามารถอธิบายการทำงานของแต่ละ recovery scheme ได้ชัดมากขึ้น

1.1.2 Suspended part

สำหรับการส่งข้อมูลในไพรมารีทรี ในกรณีที่ยังไม่มี failed edge เกิดขึ้นในระบบ แต่ละ edge ในไพรมารีทรีจะถูกใช้เพื่อส่งข้อมูลทั้งหมด แต่เมื่อเกิดกรณีที่มี failed edge เกิดขึ้นในไพรมารีทรีใดๆแล้ว ระบบจะหยุดส่งข้อมูลในบางเส้นทางของไพรมารีทรี เส้นทางที่เกิด

การระงับการส่งข้อมูลเราจะเรียกว่า suspended part ซึ่งจะแตกต่างกันไปตามชนิดของ recovery scheme ที่เลือกใช้เพื่อสร้างแบคอัพทรีซึ่งต่อไปเราจะอธิบายลักษณะของ suspended part ของแต่ละ recovery scheme กัน

1.1.2.1 Suspended part บน Global recovery scheme - เนื่องจาก global recovery จะสร้างแบคอัพทรีสำหรับแต่ละ failed edge โดยการเริ่มส่งข้อมูลสำรองจาก root ไปยังเทอร์มินอลที่ได้รับผลกระทบจาก failed edge ซึ่งก็คือเทอร์มินอลที่อยู่ถัดลงไปจาก failed edge ซึ่งอาจจะมีมากกว่าหนึ่งเทอร์มินอลก็เป็นได้ ดังนั้นจะเห็นได้ว่าข้อมูลในไพรมารีทรีที่ส่งจากเทอร์มินอลที่อยู่ก่อน failed edge จะส่งไปไม่ถึงเทอร์มินอลที่อยู่ถัดไป จึงไม่จำเป็นต้องส่งข้อมูลในส่วนนั้นอีกต่อไป เราเรียกกลุ่มของ edge ที่จะหยุดการส่งข้อมูลนั้นว่าเป็น suspended part ของ global recovery ดังภาพที่ 3



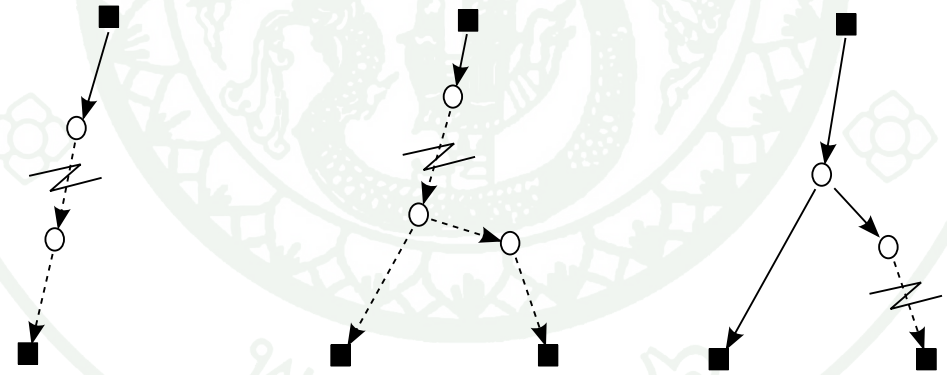
ภาพที่ 3 แสดง suspended part ของ global recovery scheme: (ก) กรณีมีเทอร์มินอลที่อยู่ถัดจาก failed edge เพียงเทอร์มินอลเดียว (ข) กรณีที่มีหลายเทอร์มินอลที่อยู่ถัดจาก failed edge หลายเทอร์มินอล (ค) กรณีที่มีเทอร์มินอลที่อยู่ถัดจาก failed edge เพียงเทอร์มินอลเดียว แต่มี edge ที่ไปยังเทอร์มินอลก่อน failed edge มีการใช้ร่วมกัน

1.1.2.2 Suspended part บน Local recovery scheme - สำหรับ local recovery scheme มีความแตกต่างจาก global recovery scheme ในส่วนหลักๆ ก็คือ node ที่เป็น node ที่เป็น root ของแบคอัพทรีจะเริ่มจาก node ที่อยู่ติดกับ failed edge ที่เป็น node ที่รับข้อมูลจากไพรมารีทรี และส่งข้อมูลไปใน failed edge หรือเรียกว่า immediate upstream node ดังนั้นเมื่อ

immediate upstream node ไม่สามารถส่งข้อมูลไปบน failed edge ได้ บรรดา edge ต่างๆในไพรมารีทรี ตั้งแต่ failed edge ไปจนถึงเทอร์มินอลถัดลงไปก็ไม่สามารถใช้ส่งข้อมูลหลักได้ จึงกลายเป็น suspended part โดยปริยาย ซึ่งแสดงได้ดังภาพที่ 4

1.1.2.3 suspended part บน Restricted local recovery scheme - ลักษณะของ RLR เป็นการสร้างแบคอัพทรีเพื่อส่งข้อมูลระหว่างจุดปลายทั้งสองของ failed edge จึงไม่มีการหยุดส่งข้อมูลบน edge ใดๆ นอกจาก failed edge เองดังนั้น suspended part ของ RLR สำหรับ failed edge $\tilde{e}$ จึงมีเพียง edge $\tilde{e}$ เพียง edge เดียว

1.1.2.4 suspended part บน Unrestricted recovery scheme - เนื่องจาก UR จะสร้างแบคอัพทรีไปยังทุกๆ เทอร์มินอลใหม่ทั้งหมดโดยไม่สนใจการส่งข้อมูลเดิมบนไพรมารีทรีเลย ดังนั้นการส่งข้อมูลที่อยู่บนไพรมารีทรีที่มีการใช้ failed edge $\tilde{e}$ จะถูกยกเลิกทั้งหมดและไปสร้างแบคอัพทรีเพื่อส่งข้อมูลไปทุกๆ เทอร์มินอลแทน ดังนั้น suspended part ของ UR ก็คือด้านทั้งหมดบนไพรมารีทรีนั่นเอง



ภาพที่ 4 แสดง suspended part ของ local recovery scheme: (ก) กรณีมีเทอร์มินอลที่อยู่ถัดจาก failed edge เพียงเทอร์มินอลเดียว (ข) กรณีที่มีหลายเทอร์มินอลที่อยู่ถัดจาก failed edge หลายเทอร์มินอล (ค) กรณีที่มีเทอร์มินอลที่อยู่ถัดจาก failed edge เพียงเทอร์มินอลเดียว แต่มี edge ที่ไปยัง immediate upstream node มีการใช้ร่วมกัน

1.1.3 Path restricted recovery scheme - ต่อไปเป็นการทำงานของ recovery scheme ที่มีการนำเสนอในงานวิจัยนี้ โดยจะเป็นการเพิ่มประสิทธิภาพให้กับการใช้งานของหน่วยความจำที่ใช้ใน UR

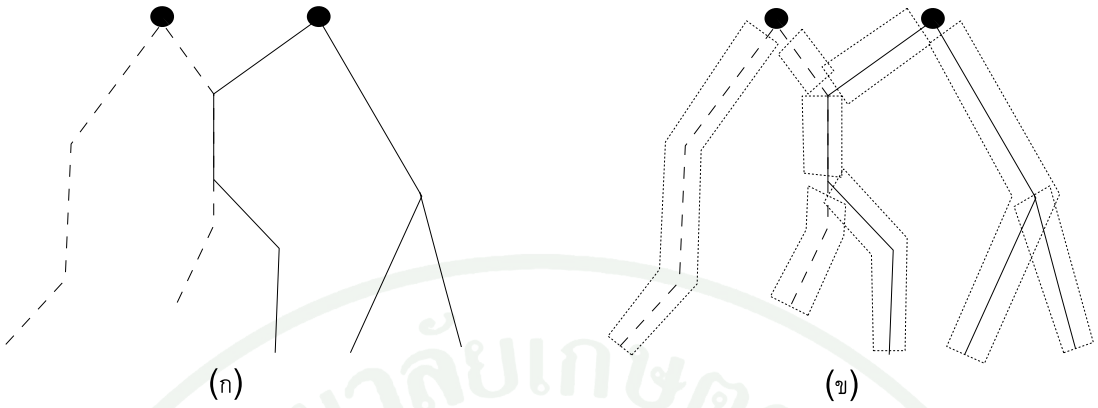
1.1.3.1 ลักษณะการทำงาน

จากที่ได้ศึกษา recovery scheme แบบต่างๆ พบว่า UR เป็น scheme ที่ให้ผลลัพธ์ในแง่ของการส่งข้อมูลรวมได้มากที่สุดแต่ยังมีข้อจำกัดอยู่ที่ต้องใช้พื้นที่หน่วยความจำมากเพื่อที่จะเก็บข้อมูลเกี่ยวกับแบคอัพทรี ซึ่งจะต้องมีหนึ่งชุดสำหรับแต่ละ failed edge ในไพรมารีทรี ดังนั้นงานวิจัยนี้จึงได้คิดและเสนอ recovery scheme แบบใหม่ซึ่งจะใช้พื้นที่ในการเก็บข้อมูลที่น้อยลง โดยจะสร้างแบคอัพทรีหนึ่งชุดสำหรับแต่ละพาธในไพรมารีทรีเท่านั้น โดยพาธประกอบไปด้วยด้านตั้งแต่หนึ่งด้านขึ้นไป ทำให้จำนวนของแบคอัพทรีลดลงอย่างเห็นได้ชัด

โดยวิธีนี้จะเริ่มโดยการสร้าง maximal disjoint path $\mathcal{P}$ ของจากไพรมารีทรีทั้งหมด จากนั้นจะสร้างแบคอัพทรีขึ้นมาสำหรับรองรับปัญหาที่จะเกิดขึ้นกับทุกพาธ $P \epsilon \mathcal{P}$ ซึ่งหมายความว่า สำหรับทุกๆ ด้าน $e \in P$ จะใช้แบคอัพทรีร่วมกัน โดยการสร้างแบคอัพทรีจะเริ่มต้นที่การส่งข้อมูลสำรองจาก root node ไปยังทุกๆ เทอร์มินอลโดยไม่ใช้ด้าน $e \in P$ และเนื่องจากเป็นการสร้างเส้นทางส่งข้อมูลขึ้นมาใหม่ทั้งหมดเช่นเดียวกันกับใน UR ดังนั้น suspended part จึงเป็นด้านในไพรมารีทรีทั้งหมดที่มีพาธนั้นเป็นส่วนประกอบ

1.1.3.2 Maximal disjoint path

ในขณะที่ UR จะทำการสร้างแบคอัพทรี 1 ชุดสำหรับป้องกันปัญหาที่จะเกิดขึ้นกับ failed edge แต่ละด้าน แต่ PR จะจัดกลุ่มของด้านเหล่านั้นให้อยู่ในรูป maximal disjoint path ซึ่งก็คือเซ็ตของ failed path ที่จะนำไปอ้างอิงในการสร้างแบคอัพทรีต่อไป ดังนั้นแล้วเมื่อมี failed path น้อยลงจำนวนของแบคอัพทรีก็จะน้อยลงตามไปด้วย โดยรูปแบบแสดงตัวอย่างของการกำหนด maximal disjoint path แสดงดังภาพที่ 5



ภาพที่ 5 การสร้าง maximal disjoint path (ก) primary path (ข) maximal disjoint path

การสร้าง maximal disjoint path จะเริ่มที่แต่ละด้าน $e \in V$ โดยเราจะให้ T_e แสดงเซ็ตของไพรมารีทรีที่มีการส่งข้อมูลผ่านมายังด้าน e ดังนั้นเราจะกล่าวได้ว่าด้าน e และ e' จะอยู่ในเซ็ตของมัลติคาสต์เดียวกันก็ต่อเมื่อ $T_e = \mathrm{T_{e'}}$ ซึ่ง maximal disjoint path จะเกิดจากเซ็ตของ subtree ที่แต่ละด้านใน subtree จะต้องอยู่ในเซ็ตของมัลติคาสต์เดียวกัน ซึ่งเซ็ตของ subtree จะต้องประกอบไปด้วยด้าน ในไพรมารีทรีทั้งหมด สำหรับอัลกอริทึมในการสร้าง maximal disjoint path เริ่มต้นโดยเลือก leaf node v และเลือก path ยาวที่สุดไปยัง root แต่ด้านที่เลือกจะต้องไม่เคยถูกเลือกไปก่อนหน้านี้แล้ว จากนั้นนำ path ที่ได้ไปใส่ไว้ในลิสต์ของ candidate path ซึ่งกระบวนการนี้จะถูกทำซ้ำๆไปจนกระทั่งได้เลือก leaf node ไปทั้งหมด เมื่อเสร็จสิ้นกระบวนการนี้เราก็จะได้เซ็ต candidate path ที่จะนำไปสร้าง maximal disjoint path ต่อไป

ในหัวข้อถัดไปเราจะอธิบายถึงการกำหนดรายละเอียดของปัญหาเพื่อให้เห็นภาพได้ดียิ่งขึ้น โดยอาศัยสัญลักษณ์ทางคณิตศาตร์ จากนั้นจะเป็นการแสดงการใช้โปรแกรมเชิงเส้นเพื่อหาแบคอัพทรีในแต่ละ recovery scheme

1.2 คำอธิบายปัญหา

ในส่วนต่อไปนี้เราจะกำหนดรูปแบบของปัญหาให้อยู่ในรูปมาตรฐาน ซึ่งในปัญหาของการส่งข้อมูลใหม่ในกรณีที่ไม่สามารถส่งข้อมูลได้ในเส้นทางปกติ เรากำหนดสัญลักษณ์ของเครือข่ายแบบมีทิศทางเป็น $G = (V, E)$ สำหรับทุกๆด้าน $e \in E$ แบนด์วิธ C_e และเซ็ตของกลุ่มการส่งข้อมูล k กลุ่ม โดยกลุ่มมัลติคาสต์กลุ่มที่ i แต่ละกลุ่มซึ่ง $1 \leq i \leq k$ จะสามารถแสดงโดย

$< s_i, t_i, D_i, w_i >$ โดยสัญลักษณ์แต่ละตัวมีความหมายดังนี้ s_i คือ source node และ $t_i = \{t_{i1}, t_{i2}, t_{i3}, \dots, t_{ij}\}$ เป็นเซ็ตของเทอร์มินอลและ D_i เป็นปริมาณข้อมูลที่เทอร์มินอลแต่ละโหนดในกลุ่มมัลติคาสต์ที่ i ต้องการได้รับ และ w_i เป็น profit ต่อหนึ่งหน่วยของแบนด์วิธที่กลุ่มมัลติคาสต์นั้นสามารถส่งได้ ซึ่งแต่ละกลุ่มมัลติคาสต์จะมีการกำหนดไพรมารีทรี T_i สำหรับกลุ่มมัลติคาสต์ที่ i ซึ่งสามารถส่งข้อมูลจาก source ไปยังทุกๆเทอร์มินอลได้ในสภาวะการทำงานที่เป็นปกติ

เราต้องการที่จะจัดสรรแบนด์วิธให้กับแต่ละกลุ่มมัลติคาสต์เพื่อในกรณีที่เกิดปัญหากับด้านใด แล้วเครือข่ายยังจะสามารถมีแบนด์วิธที่กว้างพอที่จะส่งข้อมูลไปในเส้นทางสำรอง โดยทั่วไปแล้วเราจะต้องหาว่าสำหรับแต่ละกลุ่มมัลติคาสต์ i จะสามารถส่งข้อมูลเป็นอัตราส่วน x_i เท่าไรเมื่อเทียบกับปริมาณจริงที่เทอร์มินอลต้องการ ปริมาณจริงๆที่สามารถส่งไปได้มีค่าเท่ากับ $x_i \cdot D_i$ จะถูกส่งไปตามไพรมารีทรีเพื่อจะให้ได้ profit สูงสุด

$$\sum_{i=1}^{k} w_i \cdot x_i$$

โดยในขณะที่มีแบนด์วิธที่ยังไม่ได้ใช้งานเหลือเพียงพอที่จะสามารถใช้ส่งข้อมูลสำรองในกรณีที่เกิดปัญหากับด้านในระบบ

1.3 โปรแกรมเชิงเส้น

ในส่วนนี้เราจะอธิบายถึงรายละเอียดของโปรแกรมเชิงเส้นที่จะช่วยเราในการคำนวณปริมาณข้อมูลที่สามารถส่งไปในเครือข่ายทั้งแบบที่ไม่มีการจัดสรรแบนด์วิธสำรองเอาไว้สำหรับการสร้างแบคอัพทรี และแบบที่มีการจัดสรรแบนด์วิธสำหรับแบคอัพทรี ซึ่งในกรณีทั้งหมดที่เราสนใจจะมีสมมติฐานว่าไพรมารีทรีทั้งหมดมีการกำหนดมาให้อยู่แล้ว โดยในกรณีที่เป็นโปรแกรมเชิงเส้นสำหรับการสร้างแบคอัพทรีจะให้แสดงรายละเอียดตามชนิดของ recovery scheme แต่ละชนิดอย่างละเอียด โดยเราจะเปรียบเทียบประสิทธิภาพที่ได้จากแต่ละ recovery scheme กับประสิทธิภาพที่ได้จากการจัดสรรแบนด์วิธของเส้นทางที่ดีที่สุดบนไพรมารีทรีเดียวกัน ที่ไม่มีการจองแบนด์วิธไว้สำหรับการสร้างแบคอัพทรี จุดนี้ให้สังเกตไว้ว่าการพิจารณาเฉพาะการเกิดปัญหาขึ้นเพียงจุดเดียวหรือมีเพียงด้านเดียวในเครือข่ายที่มีปัญหา กล่าวคือ recovery scheme จะไม่รองรับ

กรณีที่เกิดปัญหาขึ้นหลายๆด้านในเวลาเดียวกัน ซึ่งจะยกเว้นสำหรับการ PR จะสามารถรองรับปัญหาที่เกิดพร้อมกันหลายด้านได้ตราบใดที่ด้านที่มีปัญหายังอยู่ใน failed path เดียวกัน

ตารางที่ 2 แสดงให้เห็นความสัมพันธ์ของความสามารถในการใช้แบนด์วิธที่จองเอาไว้เพื่อการสร้างเส้นทางสำรองในกรณีต่างๆร่วมกัน ซึ่งเราจะเริ่มต้นพิจารณาที่กรณีของการใช้แบนด์วิธร่วมกันสำหรับกรณีการเกิดปัญหาบนด้านที่ต่างกันว่าเหตุใดจึงสามารถแบ่งปันการใช้แบนด์วิธเพื่อการส่งข้อมูลร่วมกันได้ เนื่องจากเราทราบแล้วจากสมมติฐานที่ว่าปัญหาของด้านจะต้องเกิดขึ้นเพียงด้านเดียวในช่วงเวลาหนึ่งๆ จะต้องไม่มีปัญหาขึ้นพร้อมๆกัน ดังนั้นเราจะเห็นได้ว่าจะไม่มีการใช้แบคอัพทรีที่เกิดจากปัญหาของด้านสองด้านพร้อมกัน ทำให้แบนด์วิธในการส่งข้อมูลสำรองที่เกิดขึ้นจากปัญหาของด้านที่ต่างกันสามารถใช้งานร่วมกันกันได้ และเมื่อเรามาลองพิจารณาในกรณีที่เกิดปัญหาบนด้านเดียวตามสมมติฐานที่ได้ตั้งไว้ ลองพิจารณาแบนด์วิธสำหรับจุดปลายทางที่ต่างกันซึ่งถึงแม้เส้นทางการส่งข้อมูลอาจจะมองได้ว่าแตกต่างกัน แต่เมื่อพิจารณาว่าแบนด์วิธที่จองเอาไว้จะส่งข้อมูลที่เหมือนกันดังนั้นเราจึงสรุปได้คร่าวๆว่าแบนด์วิธนั้นจะสามารถใช้ร่วมกันได้แต่อาจจะต้องใช้เทคนิคการเข้ารหัสข้อมูล(coding theory) เพื่อทำให้สามารถส่งข้อมูลภายในกลุ่มมัลติคาสต์ได้อย่างถูกต้อง แต่ในกรณีของแบนด์วิธที่ใช้สำหรับกลุ่มมัลติคาสต์คนละกลุ่มจะไม่สามารถใช้ร่วมกันได้อย่างแน่นอน เพราะเนื่องจากเกิดจากปัญหาที่เกิดกับด้านเดียวกันทำให้ต้องมีการใช้แบนด์วิธที่จองไว้พร้อมกันและข้อมูลที่จะส่งก็เป็นที่แน่ชัดว่าเป็นคนละข้อมูลแน่นอนทำให้ต้องมีการจองแบนด์วิธสำรองไว้เป็นของตัวเองต่างหาก

ตารางที่ 2 การแบ่งปันการใช้งานแบนด์วิธ

	กลุ่มมัลติคาสต์เดียวกัน	คนละกลุ่มมัลติคาสต์
Failed edge เดียวกัน	สามารถใช้ด้วยกันได้	ใช้ด้วยกันไม่ได้
Failed edge ต่างกัน	สามารถใช้ด้วยกันได้	สามารถใช้ด้วยกันได้

ในส่วนถัดไปเราจะได้อธิบายถึงลักษณะและวิธีการทำงานของโปรแกรมเชิงเส้นที่นำเสนอในงานวิจัยนี้ทั้งในส่วนของโปรแกรมเชิงเส้นที่แสดงการหาปริมาณข้อมูลพื้นฐานซึ่งสมมติว่าไม่มีการเกิดปัญหาใดๆเกิดขึ้นภายในระบบเครือข่าย ซึ่งเราจะเรียกรูปแบบนี้ว่ารูปแบบในอุดมคติเพราะแทบจะไม่มีโอกาสเกิดขึ้นเลยสำหรับเครือข่ายใดๆที่จะไม่มีเหตุการณ์ที่เกิดปัญหากับส่วนใด

ส่วนหนึ่งในระบบเครือข่าย จากนั้นจะอธิบายรายละเอียดของโปรแกรมเชิงเส้นทั้งห้าโปรแกรมที่ทำหน้าที่ในการจัดสรรแบนด์วิธให้กับการส่งข้อมูลสำรอง

1.3.1 การส่งข้อมูลโดยไม่มีการจองแบนด์วิธเพื่อการสร้างเส้นทางสำรอง

ในส่วนนี้เราจะแสดงรายละเอียดของโปรแกรมเชิงเส้นที่ทำการจัดสรรการส่งข้อมูลเพื่อทำการส่งไปในกลุ่มมัลติคาสต์แต่ละกลุ่ม (โดยระบุเป็นอัตราส่วนของข้อมูลที่ส่งได้เมื่อเทียบกับ D_i สำหรับการส่งข้อมูลไปยังกลุ่มมัลติคาสต์ที่ i) เพื่อที่จะให้ได้ profit สูงสุด โดยโปรแกรมเชิงเส้นนี้และถัดๆไป ไม่ได้มีหน้าที่ในการหาไพรมารีทรี เนื่องจากจะถือว่าไพรมารีทรีเป็นสิ่งที่กำหนดให้มาแล้วก่อนหน้านี้ และสำหรับโปรแกรมเชิงเส้นที่จะนำเสนอเป็นโปรแกรมแรกนี้จะไม่มีการจองแบนด์วิธสำหรับการสร้างเส้นทางสำรองแต่อย่างไร เพราะมีจุดประสงค์ที่จะให้เป็นผลอ้างอิงเอาไว้สำหรับโปรแกรมเชิงเส้นสำหรับ recovery scheme ที่จะนำเสนอเป็นลำดับถัดไป และเพื่อประกอบการทำงานของโปรแกรมเชิงเส้นเราจะประกาศตัวแปรเพิ่มเติมดังนี้

- $m_i(e), for\ 1 \le i \le k$ สำหรับด้าน $e \in E$, ซึ่งค่านี้จะเป็นค่าเศษส่วนของดีมานด์ D_i ที่ส่งผ่านด้าน e

โปรแกรมเชิงเส้นเพื่อหาสัดส่วนของดีมานด์ที่สามารถส่งไปในเครือข่ายมัลติคาสต์ที่ไม่มีการจองแบนด์วิธไว้สำหรับการสร้างเส้นทางสำรองมีดังนี้

$$
\begin{aligned}
&max && \sum_i w_i \cdot x_i && \\
&Subject\ to && && \\
&(P\text{-}CAP) && \sum_i m_i(e) \cdot D_i \le C_e && \forall e \in E \\
&(P\text{-}FLOW) && m_i(e) = \begin{cases} x_i & e \in T_i \\ 0 & otherwise, \end{cases} && \\
& && 0 \le x_i \le 1 && \forall i \in \{1,2,\dots,k\} \\
& && 0 \le m_i(e) \le 1 && \forall e \in E, \forall i \in \{1,2,\dots,k\}
\end{aligned}
$$

เมื่อมีการกำหนดเส้นทางการส่งข้อมูลแบบต้นไม้เบื้องต้นมาให้แล้ว โปรแกรมเชิงเส้นนี้จึงมีหน้าที่เพียงแต่คำนวณหาปริมาณของสัดส่วนข้อมูลที่จะไม่ทำให้เกิดการใช้แบนด์วิธเกินความกว้างของแบนด์วิธที่แต่ละด้าน e ในเครือข่ายสามารถรับได้เท่านั้น สำหรับ

โปรแกรมเชิงเส้นนี้ประกอบไปด้วยส่วนของจุดเป้าหมาย (objective) คือต้องการให้ค่าของผลรวมของสัดส่วนของข้อมูลของแต่ละกลุ่มมัลติคาสต์คูณด้วย profit ที่กำหนดสำหรับแต่ละกลุ่มมัลติคาสต์มีค่ามากที่สุด แต่มีข้อสังเกตตรงที่ว่าค่าของ w_i สำหรับแต่ละกลุ่มมัลติคาสต์เป็นค่าที่กำหนดมาให้ก่อนหน้าแล้ว ดังนั้นตัวแปรที่จะทำให้เกิดความแตกต่างของค่าเป้าหมายก็คือค่า x_i ซึ่งจะได้จากการแก้สมการของโปรแกรมเชิงเส้นนี้ โดยการที่จะทำให้ค่าเป้าหมายมีค่าสูงสุดได้ก็คือต้องให้มีการส่งข้อมูลในกลุ่มที่ได้ profit สูงสุดแต่ทั้งนี้ทั้งนั้นจะมีปัจจัยอีกหลายอย่างเนื่องจากค่า x_iเป็นค่าอัตราส่วนข้อมูลที่ส่งได้เมื่อเทียบกับ D_{i} ซึ่งอาจจะทำให้ไม่ได้ profit รวมสูงสุดจากการส่งข้อมูลในเครือข่ายที่มีแบนด์วิธจำกัดก็เป็นได้ ประกอบกับการทำงานของโปรแกรมเชิงเส้นยังถูกกำหนดด้วยข้อจำกัด (constraint) ซึ่งในโปรแกรมเชิงเส้นข้างต้นประกอบไปด้วยข้อจำกัดที่เป็นเงื่อนไขของโปรแกรมเชิงเส้นที่สำคัญสองเงื่อนไขก็คือ P-CAP และ P-FLOW และข้อจำอื่นที่กำหนดขอบเขตของค่าของตัวแปร

โดยข้อจำกัด P-CAP เป็นเงื่อนไขเพื่อที่จะรับประกันว่าข้อมูลของแต่ละกลุ่มมัลติคาสต์ที่ส่งลงไปในแต่ละด้าน e จะไม่เกินแบนด์วิธที่จะรับได้ C_{e} โดยคำนวณค่าจากสัดส่วนของข้อมูลของแต่ละกลุ่มมัลติคาสต์ที่ส่งบนด้าน e ซึ่งก็คือ $m_i(e)$ และเพื่อที่จะหาปริมาณข้อมูลจริงๆของกลุ่มมัลติคาสต์นั้นโดยการคูณเข้ากับดีมานด์ D_{i} ของกลุ่มมัลติคาสต์กลุ่มนั้นๆ ซึ่งในเงื่อนไขของข้อจำกัดก็จะระบุให้ค่ารวมของข้อมูลจากทุกกลุ่มมัลติคาสต์ต้องน้อยกว่า C_{e} ในขณะที่ข้อจำกัด P-FLOW จะเป็นตัวควบคุมการกำหนดให้มีการจองใช้แบนด์วิธของกลุ่มมัลติคาสต์กลุ่มเดียวกันให้มีสัดส่วนเท่ากันบนทุกๆด้าน e โดยที่เงื่อนไขยังระบุอีกว่าบนด้าน e ที่ไม่ได้อยู่ในไพรมารีทรีของกลุ่มมัลติคาสต์นั้นจะต้องไม่มีการจองแบนด์วิธเอาไว้ ในส่วนสองบรรทัดสุดท้ายจะเป็นส่วนที่กำหนดขอบเขตของค่าตัวแปรที่เกิดขึ้นในโปรแกรมเชิงเส้น เช่นอัตราส่วนของการส่งข้อมูลไม่ควรจะมีค่ามาก 1 หรือน้อยกว่า 0 ไม่ว่าในกรณีใดๆก็ตาม

1.3.2 Global Recovery scheme

แนวความคิดหลักของ recovery scheme นี้คือการส่งข้อมูลสำรองโดยตรงจากจุดต้นทางไปยังเทอร์มินอลที่ได้รับผลกระทบโดยตรงจากปัญหาของด้านที่สัมพันธ์กัน โดยเมื่อเกิดปัญหาขึ้นบนด้านด้านหนึ่งในไพรมารีทรี บรรดาจุดเทอร์มินอลที่เป็นจุดที่ปกติได้รับรับข้อมูลผ่านด้านมีปัญหาโดยตรงจะถูกรองรับด้วยการส่งข้อมูลผ่านช่องข้อมูลสำรองซึ่งส่งจากจุดต้นทางของกลุ่มมัลติคาสต์กลุ่มนั้นซึ่งจะมีการใช้แบนด์วิธที่จองไว้โดยเฉพาะเพื่อรองรับปัญหาที่เกิดขึ้นกับ

ด้านนั้นโดยเฉพาะ และการระงับการส่งข้อมูลในสภาวะปกติใน suspended part ที่ได้ถูกกำหนดไว้สำหรับการ failed edge นั้น แต่แบคอัพทรีจะยังสามารถใช้ด้านที่ประกอบอยู่ใน suspended part มาประกอบเป็นเส้นทางสำรองได้ ยกเว้นด้านที่เกิดปัญหา ซึ่งตัวอย่างลักษณะการทำงานของ Global recovery scheme ได้แสดงไว้ในภาพที่ 2(ข) ซึ่งโปรแกรมเชิงเส้นสำหรับ Global recovery scheme สามารถแสดงได้ดังนี้

$$
\begin{aligned}
&\textit{max} && \sum_i w_i \cdot x_i && \\
&\textit{Subject to} && && \\
&\textit{(P-CAP)} && \sum_i m_i(e) \cdot D_i \le C_e && \forall e \in E \\
&\textit{(P-FLOW)} && m_i(e) = \begin{cases} x_i & e \in T_i \\ 0 & otherwise, \end{cases} && \\
&\textit{(B-FlowBase)} && B_i^{\tilde{e}}(e) = \begin{cases} 0 & e \in SP_i^{GR}(\tilde{e}) \\ m_i(e) & otherwise, \end{cases} && \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\} \\
&\textit{(FailEdge)} && \acute{b}_i^{\tilde{e}}(\tilde{e}) = 0 && \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\} \\
&\textit{(Cap)} && \sum_i \left[\acute{b}_i^{\tilde{e}}(e) + B_i^{\tilde{e}}(e)\right] \cdot D_i \le C_e && \forall e \in E, \forall \tilde{e} \in E \\
&\textit{(Conservation)} && \sum_{e=(u,v)\in E} b_{ij}^{\tilde{e}}(e) = \sum_{e=(v,w)\in E} b_{ij}^{\tilde{e}}(e) && \forall v \in V, \forall \tilde{e} \in E, v \notin \{s_i, t_{ij}\}, \\
& && && \forall i \in \{1,2,\dots,k\}, \\
& && && \forall j \in \{j | t_{ij} \in t_i\} \\
&\textit{(DemandSat)} && \sum_{e=(u,t_{ij})\in E} \left[b_{ij}^{\tilde{e}}(e) + B_i^{\tilde{e}}(e)\right] = x_i && \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}, \\
& && && \forall j \in \{j | t_{ij} \in t_i\} \\
&\textit{(uBound)} && \acute{b}_i^{\tilde{e}}(e) \ge b_{ij}^{\tilde{e}}(e) && \forall e \in E, \forall \tilde{e} \in E, \\
& && && \forall i \in \{1,2,\dots,k\}, \\
& && && \forall j \in \{j | t_{ij} \in t_i\} \\
& && 0 \le x_i \le 1 && \forall i \in \{1,2,\dots,k\} \\
& && 0 \le m_i(e) \le 1 && \forall e \in E, \forall i \in \{1,2,\dots,k\} \\
& && 0 \le b_i^{\tilde{e}}(e) \le 1 && \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\} \\
& && 0 \le B_i^{\tilde{e}}(e) \le 1 && \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}
\end{aligned}
$$

โปรแกรมเชิงเส้นสำหรับ Global recovery scheme จะเป็นเพิ่มส่วนของข้อจำกัดเพิ่มลงไปจากโปรแกรมเชิงเส้นในส่วนก่อนหน้านี้ที่ทำหน้าที่หาสัดส่วนของดีมานด์ของแต่ละกลุ่มมัลติคาสต์ โดยไม่ต้องการจองแบนด์วิธสำหรับการทำการสร้างแบคอัพทรี โดยการทำงานของ Global recovery scheme จะทำการระงับการส่งข้อมูลใน suspended part ซึ่งเป็นส่วน

หนึ่งของไพรมารีทรี และจะหาเส้นทางที่จะส่งข้อมูลให้กับเทอร์มินอลที่ได้รับผลกระทบจากการระงับการส่งข้อมูลใน suspended part โดยการส่งข้อมูลสำรองจะถูกส่งออกมาจากจุดต้นทาง ซึ่งถ้าพิจารณาจากโปรแกรมเชิงเส้นจะพบว่าในส่วนของวัตถุประสงค์และข้อจำกัด P-CAP และ P-FLOW เป็นสิ่งเดียวกันกับที่ได้แสดงไปก่อนหน้านี้แล้วซึ่งตัวแปร $m_{\mathrm{i}}(\mathrm{e})$ จะเป็นส่วนที่สำคัญที่จะพูดถึงในส่วนข้อจำกัดที่เพิ่มเติมสำหรับ recovery scheme เมื่อเราแทนสัญลักษณ์ของด้านที่มีปัญหาเป็น $\tilde{e}$ และกำหนดสัญลักษณ์ของ suspended part บนไพรมารีทรี T_i ที่เกิดจาก $\tilde{e}$ ว่าคือ $SP_{\mathrm{i}}^{\mathrm{GR}}(\tilde{\mathrm{e}})$ แล้ว เราจะสามารถอธิบายความหมายของข้อจำกัดภายในโปรแกรมเชิงเส้นสำหรับ Global recovery scheme ได้ดังนี้

เริ่มจากข้อจำกัด B-FlowBase จะจำลองการส่งข้อมูลจากไพรมารีทรีออกมาแต่ต่างกันตรงที่จะไม่มีการส่งข้อมูลใน suspended part โดยการกำหนดค่าให้กับตัวแปร $B_{\mathrm{i}}^{\tilde{\mathrm{e}}}(\mathrm{e})$ โดยตัวแปรนี้จะเก็บค่าที่จะระบุถึงแบนด์วิธบนด้าน e ที่ใช้ส่งข้อมูลตามเส้นทางหลักของกลุ่มมัลติคาสต์ที่ i เมื่อเกิดปัญหาขึ้นที่ด้าน $\tilde{e}$ ซึ่งจากสมการจะเห็นได้ว่าค่าของตัวแปรนี้จะเท่ากับค่าที่ได้จากไพรมารีทรีแต่จะถูกกำหนดค่าเป็นศูนย์เมื่อด้าน e อยู่ใน suspended part ที่เกิดปัญหาของด้าน $\tilde{e}$ บนมัลติคาสต์กลุ่มที่ i จากนั้นข้อจำกัด DemandSat จะเป็นส่วนที่กำหนดให้เทอร์มินอลทุกๆ เทอร์มินอลจะต้องได้รับการจัดสรรแบนด์วิธเท่ากับจำนวนที่ได้ในการรับข้อมูลในสภาวะปกติ x_{i} โดยแบนด์วิธที่นำมาคิดในส่วนนี้จะคิดจากสองส่วนด้วยกันคือส่วนที่ได้รับตามไพรมารีทรีที่ไม่ได้ถูกระงับไป $B_{\mathrm{i}}^{\tilde{\mathrm{e}}}(\mathrm{e})$ ซึ่งถ้าเทอร์มินอลที่ได้รับการจัดสรรแบนด์วิธจาก $B_{\mathrm{i}}^{\tilde{\mathrm{e}}}$ ครบแล้วก็ไม่จำเป็นต้องรับข้อมูลจากเส้นทางสำรองอีก แต่ถ้าในกรณีที่เส้นทางปกติถูกระงับไป เทอร์มินอลนั้นก็จะต้องได้รับข้อมูลจากช่องทางส่งข้อมูลสำรองที่เรากำหนดขึ้นโดยตัวแปร $b_{\mathrm{ij}}^{\tilde{\mathrm{e}}}$ โดย $b_{\mathrm{ij}}^{\tilde{\mathrm{e}}}$ จะเป็นตัวแปรที่แสดงแบนด์วิธที่จะถูกจองเพื่อใช้ส่งข้อมูลสำรองจากจุดต้นทางของกลุ่มมัลติคาสต์กลุ่มที่ i ไปยังเทอร์มินอลที่ j ภายในกลุ่มมัลติคาสต์เดียวกัน ซึ่งถ้าเทอร์มินอลได้รับข้อมูลปกติตามที่ระบุในตัวแปร $B_{\mathrm{i}}^{\tilde{\mathrm{e}}}$ แล้วค่าในตัวแปร $b_{\mathrm{ij}}^{\tilde{\mathrm{e}}}(e) = 0$

ข้อจำกัด Conservation หรือข้อจำกัดการเท่ากันของการจัดสรรแบนด์วิธจะเป็นส่วนที่กำหนดให้ในทุกๆการส่งข้อมูลผ่านโหนดใดโหนดหนึ่งในเครือข่ายจะต้องจองแบนด์วิธที่ใช่ส่งข้อมูลเข้าโหนดๆนั้นเท่ากันกับการจองแบนด์วิธที่จะส่งข้อมูลออกจากโหนดๆนั้น เพื่อให้การส่งข้อมูลเป็นไปอย่างถูกต้อง ซึ่งแบนด์วิธที่ผ่านโหนดที่เราพิจารณาก็คือแบนด์วิธของเส้นทางส่งข้อมูลสำรอง $b_{\mathrm{ij}}^{\tilde{\mathrm{e}}}$ ซึ่งในส่วนของโปรแกรมเชิงเส้นจะพิจารณาเป็นการส่งข้อมูลแบบจุดต่อจุดจากต้นทางไปยังเทอร์มินอลที่ j ซึ่งข้อกำหนดนี้จะใช้กับทุกๆโหนดในเครือข่ายยกเว้นที่จุดต้นทางและ

กลุ่มของเทอร์มินอลของมัลติคาสต์กลุ่มที่ i นี้ เพราะในส่วนส่วนของจุดตั้งต้นจะยกเว้นที่ไม่จำเป็นจะต้องมีข้อจำกัดเรื่องการเท่ากันของแบนด์วิธสำหรับข้อมูลเข้าออกนี้ เนื่องจากจุดต้นทางไม่จำเป็นจะต้องรับข้อมูลจากจุดอื่น แต่สำหรับเทอร์มินอล t_{ij} จะเป็นจุดปลายทางซึ่งรับข้อมูลเพียงอย่างเดียวจึงไม่จำเป็นต้องมีข้อจำกัดของการเท่ากันของแบนด์วิธเข้าออก และข้อจำกัด FailEdge จะกำหนดไม่ให้มีการจองใช้แบนด์วิธบนด้านที่มีปัญหา $\tilde{e}$ โดยกำหนดให้การจองแบนด์วิธของมัลติคาสต์กลุ่มที่ i คือ $\acute{b}_i^{\tilde{e}}(\tilde{e})$ มีค่าเป็นศูนย์

โดยแบนด์วิธที่จองเอาไว้เพื่อการส่งข้อมูลสำรองบนด้าน e ของมัลติคาสต์กลุ่มที่ i เมื่อด้าน $\tilde{e}$ เกิดปัญหาหรือ $\acute{b}_i^{\tilde{e}}(e)$ จะต้องไม่น้อยกว่าค่าที่มากที่สุดที่จะใช้ส่งข้อมูลสำรองจากจุดต้นทางของกลุ่มมัลติคาสต์ i ไปยังเทอร์มินอล j ที่ได้รับผลกระทบจากด้านที่มีปัญหาบนด้าน e นั้นหรือ $b_{ij}^{\tilde{e}}(e)$ ซึ่งเหตุผลที่ไม่ใช้ค่าที่เป็นผลรวมเพราะว่าแบนด์วิธที่ใช้ส่งข้อมูลสำรองของกลุ่มมัลติคาสต์กลุ่มเดียวกันสามารถใช้ด้วยกันได้นั่นเอง ซึ่งข้อกำหนดนี้ถูกระบุโดยข้อจำกัด uBound และในข้อจำกัด Cap จะเป็นส่วนที่กำหนดว่าผลรวมของความกว้างที่ใช้ในการส่งข้อมูลบนด้าน e ในสภาวะที่มีปัญหาเกิดขึ้นที่ด้าน $\tilde{e}$ จะต้องไม่มากกว่าแบนด์วิธของด้าน e นั้น

1.3.3 Local recovery scheme

เมื่อมีปัญหาเกิดขึ้นกับด้าน $\tilde{e}$ และโหนดที่ส่งข้อมูลผ่าน $\tilde{e}$ พบว่าไม่สามารถส่งข้อมูลผ่าน $\tilde{e}$ ได้อีกต่อไป โหนดนั้นจะเปลี่ยนเส้นทางการส่งข้อมูลไปส่งทางอื่นแทน เพื่อให้การส่งข้อมูลของเครือข่ายมัลติคาสต์สามารถทำงานต่อไปได้ สมมติว่า $\tilde{e} = (a, b)$ แล้วจุด a จะหยุดการส่งข้อมูลไปในด้านที่ล้มเหลว $\tilde{e}$ และหันมาส่งข้อมูลผ่านเส้นทางสำรองที่เตรียมเอาไว้แทน เพื่อให้เทอร์มินอลที่ได้รับผลกระทบจากปัญหาที่เกิดบนด้าน $\tilde{e}$ สามารถที่จะได้รับข้อมูลจากกลุ่มมัลติคาสต์เหมือนเดิม ซึ่งจากนี้เราจะเรียกว่าเทอร์มินอลที่อยู่ถัดลงไปที่ได้รับผลกระทบจากด้านที่มีปัญหา $\tilde{e}$ โดยตรง จะได้รับการปกป้องจากโหนดที่เป็นผู้ส่งข้อมูลให้กับด้านที่มีปัญหาในไพรมารีทรี ดังนั้น Local recovery scheme จะมีการทำงานที่ต่างจาก GR ตรงที่จุดที่เป็นต้นทางการส่งข้อมูลสำรองจะไม่ใช่ root node ซึ่งจุดที่น่าสังเกตก็คือโหนดที่เป็นจุดเริ่มต้นของการส่งข้อมูลสำรองอาจจะเป็นเทอร์มินอลหรือโหนดภายในของไพรมารีทรีก็ได้ ดังแสดงภาพการทำงานของ Local recovery schemeในภาพที่ 2(ค) ซึ่งโปรแกรมเชิงเส้นสามารถแสดงได้ดังนี้

$$\textit{max} \quad \sum_i w_i \cdot x_i$$

Subject to

$$\textit{(P-CAP)} \quad \sum_i m_i(e) \cdot D_i \le C_e \qquad \forall e \in E$$

$$\textit{(P-FLOW)} \quad m_i(e) = \begin{cases} x_i & e \in T_i \\ 0 & otherwise, \end{cases}$$

$$\textit{(B-FlowBase)} \quad B_i^{\tilde{e}}(e) = \begin{cases} 0 & e \in SP_i^{LR}(\tilde{e}) \\ m_i(e) & otherwise, \end{cases} \qquad \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$\textit{(FailEdge)} \quad \acute{b}_i^{\tilde{e}}(\tilde{e}) = 0 \qquad \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$\textit{(Cap)} \quad \sum_i \left[\acute{b}_i^{\tilde{e}}(e) + B_i^{\tilde{e}}(e)\right] \cdot D_i \le C_e \qquad \forall e \in E, \forall \tilde{e} \in E$$

$$\textit{(Conservation)} \quad \sum_{e=(u,v)\in E} b_{ij}^{\tilde{e}}(e) = \sum_{e=(v,w)\in E} b_{ij}^{\tilde{e}}(e) \qquad \begin{array}{l} \forall v \in V, \forall i \in \{1,2,\dots,k\}, \\ \forall j \in \{j | t_{ij} \in t_i\}, \\ \forall \tilde{e} = (a,b), v \notin \{a, t_{ij}\} \end{array}$$

$$\textit{(DemandSat)} \quad \sum_{e=(u,t_{ij})\in E} \left[b_{ij}^{\tilde{e}}(e) + B_i^{\tilde{e}}(e)\right] = x_i \qquad \begin{array}{l} \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}, \\ \forall j \in \{j | t_{ij} \in t_i\} \end{array}$$

$$\textit{(uBound)} \quad \acute{b}_i^{\tilde{e}}(e) \ge b_{ij}^{\tilde{e}}(e) \qquad \begin{array}{l} \forall e \in E, \forall \tilde{e} \in E, \\ \forall i \in \{1,2,\dots,k\}, \\ \forall j \in \{j | t_{ij} \in t_i\} \end{array}$$

$$0 \le x_i \le 1 \qquad \forall i \in \{1,2,\dots,k\}$$

$$0 \le m_i(e) \le 1 \qquad \forall e \in E, \forall i \in \{1,2,\dots,k\}$$

$$0 \le b_i^{\tilde{e}}(e) \le 1 \qquad \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$0 \le B_i^{\tilde{e}}(e) \le 1 \qquad \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

โปรแกรมเชิงเส้นสำหรับ Local recovery scheme จะมีส่วนของโปรแกรมส่วนใหญ่ที่เหมือนกันโปรแกรมเชิงเส้นสำหรับ GR ที่ได้อธิบายไปก่อนหน้านี้ โดยมีความแตกต่างอยู่ที่สองข้อจำกัดก็คือข้อจำกัด B-FlowBase และข้อจำกัด Conservation ซึ่งในส่วนของข้อจำกัด B-FlowBase จะจำลองการส่งข้อมูลจากไพรมารีทรีโดยยกเว้นเส้นทางการส่งข้อมูลที่อยู่ใน suspended part จะถูกกำหนดค่าให้เป็นศูนย์คือจะยังไม่มีการส่งข้อมูลใน suspended part จากผลการหาเส้นทางสำรองจะมีการหยิบยกเอาบางส่วนของ suspended part ไปใช้งาน ซึ่งถ้าไม่สังเกตให้ดีจะรู้สึกเหมือนกับว่าข้อจำกัด B-FlowBase โปรแกรมเชิงเส้นของ LR เหมือนกันกับข้อจำกัดในโปรแกรมเชิงเส้นของ GR แต่ส่วนที่ทำให้มีความแตกต่างระหว่างทั้งสอง scheme ก็คือรูปแบบของ suspended part ที่ต่างกัน และเมื่อพิจารณาที่ข้อจำกัด Conservation ซึ่งจะเป็นข้อจำกัดที่กำหนดการ

เท่ากันของข้อมูลเข้าและออกของแต่ละโหนดในเครือข่าย ดังที่อธิบายไว้ในโปรแกรมเชิงเส้นของ GR ที่จะยกเว้นการเท่ากันไว้ที่จุดต้นทางและปลายทางของการส่งข้อมูล เมื่อพิจารณาประกอบกับเงื่อนไขของข้อจำกัดจะเห็นว่า ข้อจำกัดนี้จะยกเว้นไว้สำหรับเทอร์มินอลและโหนดที่เป็นต้นทางของด้านที่มีปัญหา ซึ่งหมายความว่าข้อมูลสำรองจะเริ่มต้นส่งออกจากโหนดที่เป็นโหนดต้นทางที่ต่ออยู่กับด้านที่ทมีปัญหานั่นเอง

1.3.4 Restricted local recovery scheme

เป็น recovery scheme ในเครือข่ายมัลติคาสต์ที่มีความคล้ายคลึงกับ recovery scheme ในเครือข่ายยูนิคาสต์มาก เนื่องจากเป็นการสร้างเส้นทางสำรองเพื่อส่งข้อมูลจากปลายหนึ่งของด้านที่มีปัญหาไปยังปลายอีกด้านหนึ่ง ด้านที่จะยกเลิกการส่งข้อมูลเมื่อเกิดปัญหาขึ้นกับด้าน $\tilde{e}$ ก็มีเพียงแค่ด้าน $\tilde{e}$ ด้านเดียว ทำให้ง่ายที่จะระบุ suspended part เมื่อเทียบกับ GR และ LR ที่ได้กล่าวมาแล้ว ดังนั้นเมื่อเกิดปัญหากับด้าน $\tilde{e} = (a, b)$ ข้อมูลสำรองจะถูกส่งออกจากโหนด a ไปตามเส้นทางสำรองที่เตรียมเอาไว้โดยหลีกเลี่ยงไม่ใช้งานด้านที่ล้มเหลว $\tilde{e}$ ไปยังโหนด b ซึ่งจะเห็นได้ว่าหลักการทำงานเป็นเช่นเดียวกับในเครือข่ายยูนิคาสต์ ทำให้ใน Restricted local recovery scheme จะมีเส้นทางสำรองสำหรับป้องกันปัญหาของแต่ละด้านที่มีการใช้งานในไพรมารีทรีทั้งหมด ซึ่ง Restricted local recovery scheme แสดงไว้ในภาพที่ 2(ง) ซึ่งโปรแกรมเชิงเส้นมีดังนี้

$$\textit{max} \quad \sum_i w_i \cdot x_i$$

Subject to

$$\textit{(P-CAP)} \quad \sum_i m_i(e) \cdot D_i \le C_e \qquad \forall e \in E$$

$$\textit{(P-FLOW)} \quad m_i(e) = \begin{cases} x_i & e \in T_i \\ 0 & otherwise, \end{cases}$$

$$\textit{(B-FlowBase)} \quad B_i^{\tilde{e}}(e) = \begin{cases} 0 & e = \tilde{e} \\ m_i(e) & otherwise, \end{cases} \qquad \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$\textit{(FailEdge)} \quad \acute{b}_i^{\tilde{e}}(\tilde{e}) = 0 \qquad \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$\textit{(Cap)} \quad \sum_i \left[\acute{b}_i^{\tilde{e}}(e) + B_i^{\tilde{e}}(e)\right] \cdot D_i \le C_e \qquad \forall e \in E, \forall \tilde{e} \in E$$

$$\textit{(Conservation)} \quad \sum_{e=(u,v)\in E} \acute{b}_i^{\tilde{e}}(e) = \sum_{e=(v,w)\in E} \acute{b}_i^{\tilde{e}}(e) \qquad \begin{array}{l} \forall v \in V, \forall i \in \{1,2,\dots,k\}, \\ \forall \tilde{e} = (a,b) \in E, \\ \forall v \in E - \{a,b\} \end{array}$$

$$\textit{(DemandSat)} \quad \sum_{e=(u,t_{ij})\in E} \acute{b}_i^{\tilde{e}}(e) = x_i \qquad \begin{array}{l} \forall \tilde{e} \in E = (a,b), v = b, \\ \forall i \in \{1,2,\dots,k\}, \end{array}$$

$$0 \le x_i \le 1 \qquad \forall i \in \{1,2,\dots,k\}$$

$$0 \le m_i(e) \le 1 \qquad \forall e \in E, \forall i \in \{1,2,\dots,k\}$$

$$0 \le \acute{b}_i^{\tilde{e}}(e) \le 1 \qquad \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

$$0 \le B_i^{\tilde{e}}(e) \le 1 \qquad \forall e \in E, \forall \tilde{e} \in E, \forall i \in \{1,2,\dots,k\}$$

โปรแกรมเชิงเส้นของ RLR ค่อนข้างแตกต่างจากโปรแกรมเชิงเส้นในสามรูปแบบที่ได้นำเสนอไปแล้วข้างต้น เนื่องจากการส่งข้อมูลสำรองไม่ได้เป็นการส่งข้อมูลไปยังเทอร์มินอลของกลุ่มมัลติคาสต์ แต่ส่งไปยังจุดใดๆซึ่งเป็นจุดปลายด้านหนึ่งของด้านที่มีปัญหา กล่าวคือถ้าด้านที่มีปัญหา $\tilde{e} = (a, b)$ แล้ว เส้นทางสำรองจะเป็นเส้นทางการส่งข้อมูลจาก a ไป b เท่านั้น ทำให้จะไม่มีการใช้แบนด์วิธร่วมกันของการส่งข้อมูลสำรองของมัลติคาสต์ภายในกลุ่มเดียวกันแต่อย่างใด ทำให้ข้อจำกัด uBound สำหรับที่เราใช้ในโปรแกรมเชิงเส้นของ scheme ก่อนๆ ถูกตัดออกไป และในส่วนของ B-FlowBase ซึ่งเป็นส่วนจำลองการส่งข้อมูลในไพรมารีทรีซึ่งจะมีการอ้างอิงถึง suspended part ของ scheme นั้นๆ แต่เนื่องจาก suspended part ของ RLR มีเพียงด้านที่มีปัญหาเพียงด้านเดียว จึงสามารถกำหนดไปได้เลยว่าตัวค่าของตัวแปรที่จะถูกเปลี่ยนค่าจากค่าในไพรมารีทรีมีเพียงค่าบนด้านที่มีปัญหาเพียงด้านเดียวเท่านั้น

ข้อจำกัด FailEdge เป็นข้อจำกัดเพื่อป้องกันไม่ให้มีการใช้ด้านที่มีปัญหาเป็นส่วนประกอบของแบคอัพทรีซึ่งเหมือนกันกับในโปรแกรมเชิงเส้นของ GR และ LR ที่ผ่านมา เช่นเดียวกับข้อจำกัด Cap ซึ่งจะกำหนดไม่ให้มีการใช้แบนด์วิธเกินแบนด์วิธบนแต่ละด้าน ส่วนข้อจำกัด Conservation และข้อจำกัด DemandSat จะเปลี่ยนไปจากเดิมเล็กน้อยเนื่องจากไม่มีการใช้แบนด์วิธร่วมกันของเส้นทางสำรองภายในกลุ่มมัลติคาสต์เดียวกัน ข้อจำกัด Conservation จะกลายเป็นข้อจำกัดบนตัวแปร $\acute{b}$ แทนโดยจะจำกัดการจองให้แบนด์วิธขาออกเท่ากับแบนด์วิธขาเข้าแต่จะยกเว้นที่จุดปลายทั้งสองของด้านที่มีปัญหาเพราะที่จุดปลายของด้านที่มีปัญหาจะเป็นจุดต้นทางและปลายทางของการส่งข้อมูลสำรองนั่นเอง ซึ่งการจะกำหนดให้การส่งข้อมูลระหว่างจุดปลายทั้งสองมีการจัดสรรแบนด์วิธเพียงพอต่อการส่งข้อมูลสำรองก็คือจะต้องกำหนดให้จุดปลายทางได้รับปริมาณข้อมูลสำรองเท่ากับข้อมูลที่ได้รับจากไพรมารีทรีดังที่ได้แสดงไว้โดยข้อจำกัด DemandSat และข้อจำกัดในส่วนสุดท้ายมีไว้เพื่อกำหนดขอบเขตให้กับค่าของตัวแปรต่างๆ

1.3.4 Unrestricted recovery scheme

มีความแตกต่างจาก GR ในส่วนของการกำหนด suspended part เท่านั้น โดย suspended part ของ Unrestricted recovery scheme ก็คือไพรมารีทรีที่ทั้งหมด ดังนั้นเมื่อเกิดปัญหาขึ้นกับด้านใดด้านหนึ่งภายในไพรมารีทรี แบคอัพทรีจะถูกคำนวณใหม่ทั้งหมดโดยไม่อิงอยู่กับไพรมารีทรีแต่อย่างไร ซึ่งก็ชัดเจนจากชื่อ Unrestricted ที่แปลว่าไร้ซึ่งข้อจำกัด ลักษะของ UR ได้แสดงไว้ดังภาพที่ 2(จ) โปรแกรมเชิงเส้นที่ใช้สำหรับการสร้างแบคอัพทรี จะเหมือนกันกับโปรแกรมเชิงเส้นที่ใช้สำหรับ GR ทุกอย่างยกเว้นเพียงแต่คำนิยามของ suspended part เท่านั้น

จากผลการวิจัยหลายๆงานที่ผ่านมาได้แสดงให้เห็นว่า UR ให้ประสิทธิภาพของแบคอัพทรีดีที่สุดเมื่อเทียบกับ scheme อื่นๆที่ได้มีการนำเสนอมาแล้ว เพราะแบคอัพทรีจะถูกคำนวณใหม่โดยหาค่าที่ดีที่สุดออกมา แต่เนื่องจากการที่ต้องคำนวณเส้นทางการทำงานใหม่ทั้งหมด ทำให้การคำนวณใช้เวลามากและต้องใช้พื้นที่สำหรับเก็บข้อมูลมากตามไปด้วย ทำให้ UR จึงไม่เป็นที่นิยมในการนำไปใช้งานจริง

1.3.5 Path restricted recovery scheme

Path restricted recovery scheme เป็น scheme ที่มีการพัฒนาโดยได้แรงบันดาลใจมากจาก UR แต่ได้เลือกใช้ส่วนที่เป็นส่วนกำหนดจำนวนแบคอัพทรีจากที่พิจารณาด้านที่มีปัญหา $\tilde{e}$ เป็นการพิจารณาจาก failed path $\tilde{p}$ แทนเนื่องจากมีจำนวนที่น้อยกว่า โดยเมื่อเกิดปัญหาขึ้นกับด้าน $\tilde{e} \in \tilde{p}$ ใดๆ การส่งข้อมูลผ่านไพรมารีทรีจะถูกยกเลิกทั้งหมด ซึ่งก็คือ suspended part ของ PR ก็คือไพรมารีทรีทั้งหมดเหมือนกันกับ UR แต่แบคอัพทรีที่คำนวณได้จะไม่มีการใช้ด้านที่อยู่ใน $\tilde{p}$ ทั้งนี้ก็เพื่อให้ด้านที่ประกอบอยู่ใน failed path เดียวกันสามารถใช้แบคอัพทรีร่วมกันได้นั่นเอง ซึ่งโปรแกรมเชิงเส้นสำหรับ PR สามารถแสดงได้ดังนี้

$$max \quad \sum_i w_i \cdot x_i$$

Subject to

$$\textit{(P-CAP)} \quad \sum_i m_i(e) \cdot D_i \leq C_e \qquad \forall e \in E$$

$$\textit{(P-FLOW)} \quad m_i(e) = \begin{cases} x_i & e \in T_i \\ 0 & otherwise, \end{cases}$$

(B-FlowBase) $B_i^{\tilde{p}}(e) = \begin{cases} 0 & e \in SP_i^{PR}(\tilde{p}) \\ m_i(e) & otherwise, \end{cases}$ $\forall \tilde{p} \in P, \forall i \in \{1,2,\dots,k\}$

(FailEdge) $\acute{b}_i^{\tilde{p}}(\tilde{e}) = 0$ $\forall \tilde{e} \in \tilde{p}, \forall i \in \{1,2,\dots,k\}$

(Cap) $\sum_i \left[\acute{b}_i^{\tilde{p}}(e) + B_i^{\tilde{p}}(e)\right] \cdot D_i \leq C_e$ $\forall e \in E, \forall \tilde{p} \in P$

(Conservation) $\sum_{e=(u,v)\in E} b_{ij}^{\tilde{p}}(e) = \sum_{e=(v,w)\in E} b_{ij}^{\tilde{p}}(e)$ $\forall v \in V, \forall \tilde{p} \in P, v \notin \{s_i, t_{ij}\}, \forall i \in \{1,2,\dots,k\}, \forall j \in \{j | t_{ij} \in t_i\}$

(DemandSat) $\sum_{e=(u,t_{ij})\in E} \left[b_{ij}^{\tilde{p}}(e) + B_i^{\tilde{p}}(e)\right] = x_i$ $\forall \tilde{p} \in P, \forall i \in \{1,2,\dots,k\}, \forall j \in \{j | t_{ij} \in t_i\}$

(uBound) $\acute{b}_i^{\tilde{p}}(e) \geq b_{ij}^{\tilde{p}}(e)$ $\forall e \in E, \forall \tilde{p} \in P, \forall i \in \{1,2,\dots,k\}, \forall j \in \{j | t_{ij} \in t_i\}$

$0 \leq x_i \leq 1$ $\forall i \in \{1,2,\dots,k\}$

$0 \leq m_i(e) \leq 1$ $\forall e \in E, \forall i \in \{1,2,\dots,k\}$

$0 \leq b_i^{\tilde{p}}(e) \leq 1$ $\forall e \in E, \forall \tilde{p} \in P, \forall i \in \{1,2,\dots,k\}$

$0 \leq B_i^{\tilde{p}}(e) \leq 1$ $\forall e \in E, \forall \tilde{p} \in P, \forall i \in \{1,2,\dots,k\}$

โปรแกรมเชิงเส้นของ PR จะมีส่วนคล้ายโปรแกรมเชิงเส้นของ UR แต่แตกต่างกันตรงที่จะให้ความสนใจไปกับ failed path แทนที่จะเป็นด้านที่มีปัญหาเพียงด้านเดียว โดยข้อจำกัด B-FlowBase จะกำหนดส่วนของแบนด์วิธที่จะยังคงส่งข้อมูลต่อไปเมื่อเกิดปัญหาขึ้นกับด้าน $\tilde{e}$ ซึ่งเมื่อ suspended part ของ PR คือไพรมารีทรีทั้งหมดดังนั้น แบนด์วิธจากไพรมารีทรีที่ยังใช้งานอยู่เมื่อเกิดปัญหากับด้าน $\tilde{e}$ ก็คือไพรมารีทรีที่ไม่มีด้าน $\tilde{e}$ ประกอบอยู่ ซึ่งแบคอัพทรีจะถูกสร้างสำหรับทุกๆ $\tilde{p}$ และสำหรับข้อจำกัด FailEdge จะจำกัดไม่ให้มีการใช้แบนด์วิธบนด้าน $\tilde{e}$ ที่เป็นส่วนประกอบของ $\tilde{p}$ เดียวกัน และข้อจำกัด Cap มีหน้าที่ในการกำหนดการใช้งานของแบนด์วิธบนด้านใดๆ ไม่ให้เกินความกว้างของแบนด์วิธบนด้านนั้นๆ ซึ่งเหมือนกันกับโปรแกรมเชิงเส้นที่ได้อธิบายไปแล้วก่อนหน้านี้ เช่นเดียวกันกับข้อจำกัด DemandSat ข้อจำกัด uBound และข้อจำกัด Conservation ซึ่งไม่ต่างไปจากโปรแกรมเชิงเส้นที่ได้กล่าวไปแล้ว

1.4 การทดลองสร้างเส้นทางการส่งข้อมูลสำรอง

ในงานวิจัยนี้ใช้การจำลองการทำงานของโปรแกรมเชิงเส้นเพื่อวัดประสิทธิภาพของแต่ละ recovery scheme โดยวิธีการทดลองจะได้อธิบายดังต่อไปนี้

เราเริ่มจากเริ่มอธิบายว่าเราจะสร้างส่วนประกอบของแต่ละการจำลองการทำงานได้อย่างไร โดยพารามิเตอร์สำหรับแต่ละการทดลองหลักๆก็คือจำนวนของโหนดในกราฟ n และจำนวนของกลุ่มมัลติคาสต์ k และดีกรีเฉลี่ยของโหนด d และโหลดที่เพิ่มให้กับระบบเครือข่าย α ซึ่งเป็นค่าสัดส่วนระหว่างจำนวนของเส้นทางการส่งข้อมูลกับจำนวนโหนดในเครือข่าย

1.4.1 การสร้างกราฟเครือข่ายที่นำมาใช้ในการทดลอง

งานวิจัยในใช้โปรแกรมสร้างกราฟเครือข่าย BRITE Simultor จากงานวิจัยของ Mediana et al. (2001) สำหรับสร้างกราฟเครือข่ายที่ไม่มีทิศทางในโมเดลของ Barabasi-Albert เช่นเดียวกันกับที่ Cohen and Nakibly (2010) ใช้ในงานวิจัยเกี่ยวกับการหาเส้นทางสำรองในระบบเครือข่ายยูนิคาสต์ ซึ่งในการทดลองของเราเราจะใช้กราฟที่มีจำนวน 20 โหนดและค่าดีกรีเฉลี่ย d มีค่าเป็น 2 และ 3 ตามลำดับ

เราต้องจำไว้ว่าเราพิจารณากราฟเครือข่ายที่สร้างจากโปรแกรม BRITE Simulator โดยจะพิจารณาเหมือนกับกว่ากราฟที่ได้เป็นเครือข่ายจริงซึ่งโหนดแต่ละโหนดจะถูกพิจารณาเป็นเราท์เตอร์และด้านจะถูกมองเป็นเส้นเชื่อมต่อระหว่างเราท์เตอร์ในเครือข่าย ในขณะที่โปรแกรมเชิงเส้นจะทำงานโดยพิจารณาเครือข่ายที่มีทิศทางเป็นหลักแต่กราฟเครือข่ายที่เราได้จากโปรแกรม BRITE Simulator กลับเป็นเครือข่ายแบบไม่กำหนดทิศทาง ดังนั้นเราจะต้องแปลงให้เครือข่ายที่ไม่กำหนดทิศทางเป็นเครือข่ายแบบกำหนดทิศทางให้ได้ โดยเราจะแทนด้านๆหนึ่งซึ่งไม่มีทิศทางด้วยด้านสองด้านที่มีขนาดเท่ากันแต่มีทิศทางตรงกันข้าม ในการทดลองเราเลือกโหนดต้นทางจำนวน k โหนดจากกราฟที่กำหนดมาให้ ความน่าจะเป็นที่โหนดแต่ละโหนดจะถูกเลือกให้เป็นโหนดต้นทางจะเป็นสัดส่วนขึ้นอยู่กับดีกรีหรือจำนวนด้านที่ต่ออยู่โหนดโหนดนั้น ซึ่งการกระจายของจำนวนดีกรีในแต่ละโหนดเป็นไปตามกฎ power-law และเพื่อที่จะสร้างเซ็ตของคำร้องขอ (request) เราเลือกเทอร์มินอลและโหนดต้นทางจากต้นทางทั้งหมด k โหนดด้วยวิธีการสุ่มแบบยูนิ

ฟอร์ม โดยมีข้อแม้อยู่อย่างหนึ่งว่าสำหรับทุกๆจุด ต้นทาง จะต้องมีเครื่องปลายทางอย่างน้อย 1 เครื่อง จำนวนของคำร้องขอนี้ขึ้นอยู่กับโหลดที่ให้กับเครือข่าย ซึ่งค่านี้ก็คือ $n \cdot \alpha$

นอกจากนั้นเรายังเลือกกราฟเครือข่ายที่มีการใช้งานจริงเพื่อมาทดสอบประสิทธิภาพของงานวิจัยที่ได้นำเสนออีกด้วย กราฟแรกที่นำมาใช้ก็คือเครือข่าย US Long distance Backbone ซึ่งประกอบด้วยโหนดจำนวน 28 โหนดและมีเส้นเชื่อมต่อ 45 เส้น และกราฟที่สองเป็นกราฟของเครือข่าย Carrier backbone ซึ่งประกอบด้วยโหนด 15 โหนดและเส้นเชื่อมต่อ 28 เส้น

1.4.2 การสร้างไพรมารีทรี

ในการสร้างไพรมารีทรีซึ่งจะนำไปใช้เป็นข้อมูลเบื้องต้นให้กับงานวิจัย เราเลือกที่จะใช้อัลกอริทึมแบบ heuristics ที่มีชื่อว่า Nearest Neighbor First (NNF) ซึ่ง Kodiaiam and Lakshman (2002) ได้ใช้ในการสร้างไพรมารีทรีในงานวิจัยเช่นกัน โดยวิธีการนี้จะเริ่มจากการกำหนดโหนดต้นทาง s และเซ็ตของเทอร์มินอลที่ต้องการส่งข้อมูลไปให้ D การทำงานจะเริ่มที่ s ไปจนกระทั่งได้เส้นทางไปยังเทอร์มินอลได้ครบทั้งหมด โดยอัลกอริทึมมีดังนี้

$Function\ \ NNF(s, D)$
$N = \{s\}$
$P = \emptyset$
$While\ D \neq \emptyset\ do$
    $\forall x \in D, find\ y \in D\ such\ as\ distance(y, N) \leq distance(x, N)$
    $N = N \cup NodeBetween(y, N) \cup y$
    $P = P \cup PathBetween(y, N)$
    $D = D - y$
$T = (N, P)$
$Return\ \ T$

ในอัลกอริทึมนี้มีการเรียกใช้ฟังก์ชั่นหลายฟังก์ชั่นดังนั้นจึงขออธิบายการทำงานของแต่ละฟังก์ชั่นที่มีการเรียกใช้ก่อน ฟังก์ชั่น $distance(v, N)$ ใช้ในการคำนวณหาค่าระยะทางจากโหนด v ไปยังโหนดใดโหนดนึ่งภายในเซ็ต N ส่วนฟังก์ชั่น $NodeBetween(v, N)$ จะหาเซ็ตของโหนดทั้งหมดที่อยู่ในเส้นทางที่สั้นที่สุดจากโหนด v ไปโหนดในเซ็ต N และฟังก์ชั่น $PathBetween(v, N)$ จะหาเซ็ตของด้านทั้งหมดที่อยู่ในเส้นทางที่สั้นที่สุดจากโหนด v ไปยังโหนดในเซ็ต N

การทำงานของอัลกอริทึมนี้การทำงานแบบวนรอบไปจนกว่าเงื่อนไขการกำหนดการวนรอบจะเสร็จสมบูรณ์ เมื่อเริ่มต้นค่าในตัว N หรือเซ็ตของโหนดที่ประกอบอยู่ภายใน tree จะมีเพียงจุดต้นทางเพียงจุดเดียว และเซ็ตของด้านที่อยู่ใน tree เริ่มแรกจะเป็นเซ็ตว่าง ในแต่ละรอบการทำงานของอัลกอริทึมจะมีการเลือกโหนดที่อยู่ใกล้โหนดใดโหนดหนึ่งในเซ็ตของ N มากที่สุด จากนั้นเพิ่มโหนดดังกล่าวและโหนดที่ประกอบอยู่ในเส้นทางไปยังตัวแปร N และเพิ่มด้านที่เป็นส่วนประกอบของเส้นทางนั้นไปยังตัวแปร P และลบโหนดนั้นออกจากเซ็ตปลายทางในตัวแปร D จนกระทั่งไม่เหลือโหนดในตัวแปร D อีกต่อไป

1.4.3 การสร้างเซ็ตของ failed path

สำหรับ PR เราจะต้องระบุเซ็ตของ failed path หรือ maximal disjoint path ก่อนที่จะใช้โปรแกรมเชิงเส้นในการหาแบคอัพทรี ซึ่งเราจะแสดงสร้าง maximal disjoint path จากกลุ่มมัลติคาสต์ที่เราสนใจ โดยอัลกอริทึมจะแสดงการแบ่งไพรมารีทรีทั้งหมดออกเป็นส่วนเส้นทางย่อยๆที่ไม่มีการใช้ด้านร่วมกัน

ซึ่งฟังก์ชั่น $GetPaths$ จะเป็นส่วนของอัลกอริทึมที่ทำหน้าที่ในการแบ่งไพรมารีทรีแต่ละต้นให้อยู่ในรูปของเซ็ตของ disjoint path ซึ่งจะเป็นส่วนแรกเพื่อจะนำผลที่ได้ไปดำเนินการในส่วนต่อไป

$Function\ GetPath(T, D)$
$L = \{\emptyset\}$
$for\ each\ x\ \in D$
    $p = PathToRoot(x, T)$
    $for\ each\ l \in L, p = p - l$
    $L = L \cup \{p\}$
$Return\ L$

อัลกอริทึมในส่วนนี้จะทำการเก็บรักษาเซ็ตของเทอร์มินอล D และใช้ฟังก์ชั่น $PathToRoot(x, T)$ ที่จะหาค่าของเส้นทางที่สั้นที่สุดเพียงเส้นทางเดียวที่ตรงไปยังโหนดต้นทางของ tree แต่ก่อนที่จะมีการเพิ่มเส้นทางที่หาได้ p ไปยังเซ็ตของเส้นทางที่เป็นคำตอบจะต้องมีการลบด้านที่เคยปรากฏอยู่ภายในเซ็ต L ออกจากเส้นทาง p ซะก่อนเพื่อไม่ให้มีการใช้ด้านร่วมกันระหว่างเส้นทางที่ถูกเพิ่มเข้าไปในเซ็ต L ด้วยเหตุนี้เราจึงรับประกันได้ว่าจะไม่มีเส้นทางใดในเซ็ตผลลัพท์ L ที่ใช้ด้านร่วมกันเลย

จากฟังก์ชั่นข้างต้นเราได้ disjoint path จากไพรมารีทรีแต่ละชุด แต่ยังไม่สามารถนำมาใช้งานได้โดยตรงเพราะเส้นทางภายในเซ็ตของ disjoint path ที่ได้จากไพรมารีทรีแต่ละชุดอาจจะมีส่วนที่ใช้ด้านร่วมกันอยู่ ดังนั้นเราจะต้องอัลกอริทึมเพิ่มเติมเพื่อสร้าง maximal disjoint path จากเซ็ตของ disjoint path ที่ได้โดยการแบ่ง disjoint path ที่มีด้านที่ซ้ำกันเป็นส่วนย่อยๆ จนกระทั่งไม่มี disjoint path ใดที่มีด้านร่วมกันอีก

$\mathcal{P} = \{\emptyset\}$
$\mathcal{R} = \{\emptyset\}$
$for\ each\ T_i \in \mathcal{T}, \mathcal{R} = \mathcal{R} \cup GetPath(T_i, D_i)$
$while\ \mathcal{R} \neq \emptyset$
    $let\ x\ be\ some\ path\ in\ \mathcal{R}$
    $\mathcal{R} = \mathcal{R} - \{x\}$
    $if\ there\ exists\ y \in \mathcal{R}\ such\ that\ x \cap y \neq \emptyset\ then$
        $\mathcal{R} = \mathcal{R} - \{y\}$
        $\mathcal{R} = \mathcal{R} \cup \{x - y\} \cup \{y - x\} \cup \{x \cap y\}$ $(*)$
    $else$
        $\mathcal{P} = \mathcal{P} \cup \{x\}$
$return\ \mathcal{P}$

การทำงานของอัลกอริทึมจะเริ่มจากการกำหนดค่าเริ่มต้นให้กับตัวแปรที่เก็บเซ็ตของ disjoint path ทั้งหมดจากไพรมารีทรีของแต่ละกลุ่มมัลติคาสต์ $\mathcal{R}$ เมื่อได้ disjoint path ที่เป็นไปได้ทั้งหมดแล้วจะต้องทำให้ไม่มี path ใดๆที่ใช้ด้านร่วมกัน โดยจะเลือกค่าในเซ็ต $\mathcal{R}$ ออกมาค่าหนึ่งซึ่งสมมติให้เป็นเส้นทางที่ชื่อ x จากนั้นให้ลบเส้นทาง x ออกจากเซ็ต $\mathcal{R}$ แล้วจากนั้นหาเส้นทาง y ภายในเซ็ต $\mathcal{R}$ ที่มีด้านที่ซ้ำกับ x ถ้าพบให้ลบเส้นทาง y ออกยากเซ็ต $\mathcal{R}$ แล้วให้แบ่งเส้นทางออกเป็นเส้นทางย่อย $x - y, y - x, x \cap y$ (ในบรรทัดที่มีเครื่องหมาย $(*)$ ข้างต้น) จากนั้นให้เพิ่มกลับไปยังเซ็ตของ disjoint path $\mathcal{R}$ เพื่อนำไปตรวจสอบต่อไปว่าซ้ำกันกับเส้นทางอื่นๆอีกหรือไม่ แต่ถ้าไม่พบเส้นทาง y ที่ใช้ด้านร่วมกับเส้นทาง x ก็ให้เพิ่มเส้นทาง x ลงไปในเซ็ตของผลลัพธ์เส้นทางที่ล้มเหลว $\mathcal{P}$

อัลกอริทึมจะทำงานซ้ำไปหลายๆครั้งแต่ให้จดจำไว้ว่าในแต่ละครั้งจะแบ่งเส้นทางที่ใช้ด้านร่วมกันเป็น path ที่สั้นลงซึ่งอย่างมากที่สุดก็จะไม่เกิดจำนวนของด้านที่ใช้กลุ่มมัลติคาสต์ทั้งหมด หรือไม่เกินจำนวนของด้านทั้งหมดที่มีอยู่ในกราฟ ดังนั้นอัลกอริทึมนี้ทำงานอยู่ในเวลาแบบโพลิโนเมียล

2. การสร้างเส้นทางส่งข้อมูลบนเครือข่ายมัลติคาสต์แบบออบลิเวียส

ในส่วนนี้จะเป็นการหาเส้นทางการส่งข้อมูลระหว่างเครื่องที่ทำหน้าที่เป็นแม่ข่ายไปยังเทอร์มินอลต่างๆในกลุ่มมัลติคาสต์ โดยการเส้นทางการส่งข้อมูลจะไม่ขึ้นอยู่กับสถานะของการส่งข้อมูลในปัจจุบัน กล่าวคือไม่ว่าโหลดของเครือข่ายจะเป็นอย่างไรจะไม่มีการคำนวณเส้นทางการส่งข้อมูลใหม่ เพื่อสะดวกในการจัดการกับการส่งข้อมูลได้ง่ายขึ้น โดยเป้าหมายของการออกแบบเส้นทางการส่งข้อมูลแบบนี้ก็คือสามารถระบุค่า congestion สูงสุดที่อาจเกิดขึ้นจากเส้นทางส่งข้อมูลนี้ ในงานวิจัยนี้จะแบ่งงานออกเป็นสองส่วนหลักๆ ในส่วนแรกจะมีการแสดงให้เห็นว่าการหาเส้นทางในเครือข่ายมัลติคาสต์แบบออบลิเวียสมีลักษณะที่สำคัญที่มีความสัมพันธ์กับการหาเส้นทางในเครือข่ายยูนิคาสต์จนสามารถใช้เทคนิคการหาเส้นทางด้วยกันได้ ในส่วนถัดมาจะเป็นการจำลองการทำงานของการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์โดยการใช้โปรแกรมเชิงเส้นเพื่อหาเส้นทางจริงๆในเครือข่าย และคำนวณหาค่าประสิทธิภาพเพื่อนำมาเปรียบเทียบประกอบการพิจารณา

2.1 การเท่ากันของเครือข่ายยูนิคาสต์และมัลติคาสต์ในแง่ของการส่งข้อมูลแบบออบลิเวียส

ในปัญหาของการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์ เรามีกราฟของเครือข่าย $G = (V, E)$ ซึ่งสามารถเป็นได้ทั้งกราฟแบบมีทิศทางและไม่มีทิศทาง และสำหรับแต่ละด้าน $e \in E$ เราจะกำหนดสัญลักษณ์ $c(e)$ แทนความแบนด์วิธบนด้านๆนั้น ซึ่งในปัญหานี้ เราต้องการจะหารูปแบบการส่งข้อมูลที่จะมีการกำหนดเส้นทางระหว่างต้นทางและกลุ่มของเทอร์มินอลได้ก่อนที่จะมีการส่งข้อมูลเกิดขึ้น อย่างไรก็ตามจำนวนเซ็ตของเทอร์มินอลมีจำนวนมากมายที่เป็นไปได้สำหรับจุดต้นทางหนึ่งๆ เช่นนี้แล้วทำให้การหาเส้นทางที่เป็นไปได้สำหรับกลุ่มมัลติคาสต์แต่ละกลุ่มจะใช้หน่วยความจำจำนวนมาก อย่างไรก็ดีรูปแบบการส่งข้อมูลของเราจะสมมติให้คู่ของต้นทางและเทอร์มินอลแต่ละคู่เป็นเหมือนกันการส่งข้อมูลยูนิคาสต์ และสามารถประยุกต์ให้เข้ากับรูปแบบการส่งข้อมูลแบบมัลติคาสต์ได้

2.1.1 คำอธิบายปัญหา

สำหรับแต่ละคู่ของโหนด i และ j ใดๆ จะกำหนดสัญลักษณ์การส่งข้อมูลหนึ่งหน่วย f_{ij} เป็นการส่งข้อมูลหนึ่งหน่วยจาก i ไป j และสัญลักษณ์ $f_{ij}(e)$ แทนค่าของข้อมูลที่ส่งผ่านด้าน e ซึ่งข้อมูลที่จะส่งสามารถเป็นส่วนย่อยและส่งไปในเส้นทางที่ต่างกัน ไม่จำเป็นจะต้องส่งไปในเส้นทางเดียวกันทั้งหมด โดยอาศัยทฤษฎีการเข้ารหัสเพื่อสามารถทำให้การส่งข้อมูลส่วนย่อยๆเป็นไปได้ และเพื่อที่จะส่งข้อมูลของแต่ละกลุ่มมัลติคาสต์ เราจะให้ข้อสังเกตุว่าช่องสัญญาณระหว่างเทอร์มินอลหลายๆจุดในกลุ่มมัลติคาสต์เดียวกันสามารถใช้งานร่วมกันได้โดยไม่ต้องจัดสรรแบนด์วิธเพิ่มเติมแต่อย่างใด

เมื่อมีการร้องขอที่จะส่งข้อมูลแบบมัลติคาสต์ทั้งหมด M กลุ่ม เราจะกำหนดตัวแปร k ซึ่ง $1 \leq k \leq M$ จะมีการกำหนดชุดของความต้องการส่งข้อมูลในลักษณะกลุ่มมัลติคาสต์กลุ่มที่ k ประกอบด้วยส่วนประกอบสามส่วน $< s_k, T_k, d_k >$ ซึ่ง s_k คือเครื่องต้นทางที่เป็นแม่ข่ายที่จะส่งข้อมูล ในขณะที่ T_{k} เป็นเซ็ตของเทอร์มินอลของมัลติคาสต์กลุ่มที่ k และ d_{k} เป็นจำนวนข้อมูลที่ต้องการส่งในกลุ่มที่ k ต่อไปเราจะอธิบายว่าจะสามารถส่งข้อมูลตาม $< s_k, T_k, d_k >$ ได้อย่างไร จากคำร้องของเราจะส่งข้อมูลจำนวน d_{k} หน่วยจากจุดต้นทาง s_k ไปยังทุกๆเทอร์มินอล $t \in T_k$ และสำหรับแต่ละเทอร์มินอล $t \in T_k$ เราจะหาเส้นทางการส่งข้อมูลจำนวนหนึ่งหน่วย $f_{\mathrm{s_kt}}$ ซึ่งจะนำไปใช้ส่งข้อมูลขนาด d_{k} และเมื่อบนด้าน e ใดๆมีการส่งข้อมูลที่เป็นชนิดเดียวกันก็สามารถใช้ช่องสัญญาณร่วมกันได้ ดังนั้นแทนที่เราจะต้องจองแบนด์วิธเป็นขนาดเท่ากับผลรวมของแต่ละเส้นทางการส่งข้อมูลย่อย และการใช้ประโยชน์จากทฤษฎีการเข้ารหัสเราจึงจองแบนด์วิธเพียงเท่ากับแบนด์วิธที่กว้างที่สุดที่ต้องใช้เท่านั้น ซึ่งเท่ากับ

$$d \cdot \max_{t \in t_k} f_{s_k t}\,(e)$$

ต่อไปเราจะแนะนำสัญลักษณ์ที่สัมพันธ์กับเมตริกซ์ของความต้องการส่งข้อมูลซึ่งจะใช้ในส่วนต่อๆไป สำหรับแต่ละชุดของกลุ่มมัลติคาสต์ $< s_k, T_k, d_k >$ จะสามารถแทนด้วยเมตริกซ์การส่งข้อมูล $D^k = \{D_{ij}^k\}$ ซึ่งมีขนาด $n \times n$ และสมาชิกในเมตริกซ์ทุกตำแหน่งมีค่ามากกว่าหรือเท่ากับศูนย์ ในขณะที่สมาชิกในแกนทแยง (diagonal) มีค่าเป็นศูนย์ทั้งหมด และสำหรับแต่ละคู่ i, j แล้วค่า D_{ij}^k คือค่าของจำนวนข้อมูลที่ต้องการส่งจากจุดต้นทาง i ไปยัง

เทอร์มินอล j ดังนั้นเมื่อจุดต้นทางของเทอร์มินอลทุกๆจุดคือ s_k แล้วเราสามารถระบุค่า D_{ij}^k เป็นค่า d_{k} สำหรับทุกๆคู่ i และ j ซึ่ง $i = s_k$ และ $j \in T_k$ ในขณะที่ค่าอื่นจะเป็นศูนย์ทั้งหมด

เมื่อต้องการส่งข้อมูลกลุ่มมัลติคาสต์จำนวน M กลุ่ม เราจะกำหนดเซ็ตของเมตริกซ์ความต้องการส่งข้อมูล $\mathcal{D}$ ให้เป็น $\{D^1, D^2, \dots, D^M\}$ ต่อไปนี้เพื่อความสะดวกเราจะเรียก $\mathcal{D}$ ว่า "ดีมานด์" และเพราะว่าการส่งข้อมูลแบบยูนิคาสต์เป็นการส่งข้อมูลแบบมัลติคาสต์ชนิดหนึ่งซึ่งมีจุดปลายทางเพียงจุดเดียว ซึ่งความต้องการส่งข้อมูลยูนิคาสต์สามารถแทนได้ด้วยเมตริกซ์ของความต้องการส่งข้อมูลแบบมัลติคาสต์ได้ และเพื่อความไม่สับสนเราจะแทนสัญลักษณ์ดีมานด์ของยูนิคาสต์ด้วย $\mathcal{D}^u$ และต้องจำไว้ว่าเมตริกซ์ของความต้องการส่งข้อมูลจะมีความแตกต่างไปจากมาตรฐานและงานวิจัยอื่นเล็กน้อยที่ซึ่งหนึ่งเมตริกซ์สามารถแทนความต้องการทั้งหมดของทุกๆกลุ่มมัลติคาสต์ได้ เนื่องจากในงานวิจัยส่วนนี้ต้องการให้มีการใช้งานจุดต้นทางร่วมกันระหว่างแต่ละกลุ่มมัลติคาสต์ได้ แต่ถึงจะมีการกำหนดเมตริกซ์ของความต้องการส่งข้อมูลต่างไปจากปกติแต่การให้ความหมายของความคับคั่งของข้อมูลบนระบบเครือข่ายทั้งสองยังคงเหมือนเดิมทุกประการ

ต่อไปเราจะให้ความหมายของ congestion และประสิทธิภาพของรูปแบบการส่งข้อมูล โดยเราจะทำในรูปแบบคล้ายกันกับงานวิจัยของ Azar et al. (2003) เมื่อกำหนดให้เครือข่าย G และดีมานด์ $\mathcal{D}$ เราจะกำหนดให้ $f_{\mathrm{ij}}(\mathrm{e})$ เป็นข้อมูลที่ส่งจากโหนด i ไปยังโหนด j ในการส่งข้อมูล f

กำหนดให้ฟังก์ชั่น $FLOW(e, f, D^k)$ แสดงค่าของข้อมูลที่ส่งบนด้าน e ซึ่งเกิดจากเมตริกซ์ D^k ภายใต้รูปแบบการส่งข้อมูล f ซึ่งจากคำอธิบายนี้เราจะได้ว่า

$$FLOW(e, f, D^k) = \max_{i,j} D_{ij}^k \cdot f_{ij}(e)$$

ถ้ากำหนดให้ $FLOW(e, f, \mathcal{D})$ แทนค่าของข้อมูลทั้งหมดบนด้าน e จากการจัดสรรเส้นทางทั้งหมดในดีมานด์ $\mathcal{D}$ ภายใต้รูปแบบการหาส่งข้อมูล f คือ

$$FLOW(e, f, \mathcal{D}) = \sum_{1 \le k \le M} FLOW(e, f, D^k)$$

และ congestion ของข้อมูลบนด้าน e หรือ $EDGE - CONG(e, f, \mathcal{D})$ เป็นค่าสัดส่วนของข้อมูลทั้งหมดบนด้าน e หารด้วยแบนด์วิธของด้านนั้นๆ ตัวอย่างเช่น

$$EDGE\_CONG(e, f, \mathcal{D}) = \frac{FLOW(e, f, \mathcal{D})}{c(e)}$$

เราจะกำหนดสัญลักษณ์แทน congestion ของดีมานด์ $\mathcal{D}$ ที่มากที่สุดบนรูปแบบการส่งข้อมูล f ด้วย $CONGESTION(f, \mathcal{D})$

$$CONGESTION(f, \mathcal{D}) = \max_{e \in E} EDGE\_CONG(e, f, \mathcal{D})$$

เมื่อกำหนดเครือข่าย G และด้านที่มีแบนด์วิธ c และดีมานด์ $\mathcal{D}$ เราจะกำหนดให้ $OPT(\mathcal{D})$ เป็นสัญลักษณ์แทน congestion ที่น้อยที่สุดสำหรับดีมานด์ $\mathcal{D}$ บนรูปแบบการส่งข้อมูลใดๆ ดังนั้นอัตราส่วนประสิทธิภาพของรูปแบบการส่งข้อมูล f บนดีมานด์ $\mathcal{D}$ ก็คือ

$$PERF\_RATIO(f) = \frac{CONGESTION(f, \mathcal{D})}{OPT(\mathcal{D})}$$

และสำหรับรูปแบบการส่งข้อมูลที่ใช้ เราจะเรียก $OBLIV\_PERF\_RATIO(f)$ ว่าเป็นค่าอัตราส่วนประสิทธิภาพที่แย่ที่สุดสำหรับรูปแบบการส่งข้อมูล f สำหรับดีมานด์ใดๆ เช่น

$$OBLIV\_PERF\_RATIO(f) = \max_{\mathcal{D}} PERF\_RATIO(f, \mathcal{D})$$

และสำหรับกราฟ G เราจะกำหนดค่าอัตราส่วนการส่งข้อมูลแบบออบลิเวียสที่ดีที่สุดว่าคือ

$$OBLIV\_OPT(G) = \min_{f} OBLIV\_PERF\_RATIO(f)$$

ซึ่งเป้าหมายของเราก็คือหาค่า $OBLIV\_OPT(G)$

ซึ่งเราจะพิจารณาค่าประสิทธิภาพของรูปแบบการส่งข้อมูล f ภายใต้การส่งข้อมูลแบบยูนิคาสต์โดยกำหนดให้

$$OBLIV\_UNI\_PERF\_RATIO(f) = \max_{unicast\ \mathcal{D}} PERF\_RATIO(f, \mathcal{D})$$

จากสัญลักษณ์ข้างต้น เราจะกำหนดกำหนดค่าอัตราส่วนการส่งข้อมูลแบบออบลิเวียสที่ดีที่สุดบนเครือข่ายยูนิคาสต์ว่าคือ

$$OBLIV\_UNI\_OPT(G) = \min_{f} OBLIV\_UNI\_PERF\_RATIO(f)$$

เราต้องจำไว้ว่าในการให้ความหมายของ $OBLIV\_PERF\_RATIO$ และ $OBLIV\_UNI\_PERF\_RATIO$ จะเป็นค่าที่ได้จากการคำนวณค่าของดีมานด์ที่เป็นไปได้ทั้งหมด ซึ่งมีปริมาณมากมายมหาศาล โดยจะทำให้โปรแกรมเชิงเส้นที่จะสร้างขึ้นเพื่อรองรับการทำงานนี้มีจำนวนของข้อจำกัดเกือบจะมีค่าไม่สิ้นสุดเลยทีเดียว อย่างไรก็ตาม สำหรับค่าอัตราส่วนออบลิเวียสของเครือข่ายยูนิคาสต์ซึ่ง Azar et al. (2003) แสดงไว้ว่ามีจำนวนตัวแปรภายในโปรแกรมเชิงเส้นเป็นโพลิโนเมียล และได้ใช้โปรแกรมเชิงเส้นต่างหากอีกโปรแกรมหนึ่งเพื่อแสดงให้เห็นว่าปัญหาของการหาเส้นทางออบลิเวียสบนเครือข่ายยูนิคาสต์สามารถหาได้ในเวลาโพลิโนเมียลโดยใช้วิธีการแบบอีลิปซอยด์

ในขณะที่โปรแกรมเชิงเส้นสำหรับแก้ปัญหาเพื่อหาค่า $OBLIV\_OPT(G)$ ก็น่าจะมีจำนวนตัวแปรเป็นโพลิโนเมียลเช่นเดียวกัน แต่การที่จะใช้การสร้างโปรแกรมเชิงเส้นเพื่อทำงานแบบอีลิปซอยด์เหมือนกับงานของ Azar et al. (2003) ดูเหมือนจะยากเกินไปในกรณีนี้เพราะจะมีปัญหาในเรื่องของการกำหนดค่าเงื่อนไขของค่าสูงสุดของ congestion บนด้าน e ใดๆ

2.1.2 ความเท่ากันของการหาเส้นทางออบลิเวียสบนยูนิคาสต์และมัลติคาส

ในส่วนนี้เราจะแสดงการพิสูจน์ทฤษฎีต่อไปนี้

ทฤษฎีบท 1 เมื่อกำหนดกราฟ G สำหรับรูปแบบการส่งข้อมูล f แล้วจะได้ว่า

$$OBLIV\_PERF\_RATIO(f) = OBLIV\_UNI\_PERF\_RATIO(f)$$

เช่นนั้นแล้วการหาเส้นทางที่ดีที่สุดแบบออบลิเวียสจะเป็นเส้นทางที่ดีที่สุดในการหาเส้นทางแบบออบลิเวียสในเครือข่ายมัลติคาสต์ด้วย

เพื่อที่จะแสดงให้เห็นว่าปัญหาการหาเส้นทางแบบออบลิเวียสในเครือข่ายยูนิคาสต์และมัลติคาสต์มีความเทียบเท่ากัน เราจะเริ่มจากการพิสูจน์ทฤษฎีบทย่อยที่เกี่ยวข้องที่จะสนับสนุนข้อจำกัดที่น่าเชื่อถือเกี่ยวกับดีมานด์ที่แย่ที่สุด เช่น ถ้ามีดีมานด์ที่แย่ที่สุดสำหรับการหาเส้นทางส่งข้อมูลในเครือข่ายมัลติคาสต์ ดีมานด์นั้นจะเป็นดีมานด์ในรูปแบบของยูนิคาสต์ และเพื่อให้มีการเจาะจงลงไป ทฤษฎีบทย่อยจะแสดงการสร้างดีมานด์ของยูนิคาสต์ จากดีมานด์ที่ใช้ในมัลติคาสต์ที่ให้ congestion ของข้อมูลเท่ากัน ยิ่งไปกว่านั้นจะแสดงให้เห็นอีกว่าดีมานด์ของยูนิคาสต์จะเป็นซับเซตของดีมานด์แบบมัลติคาสต์ ซึ่งตามหลักการแล้วเราจะสามารถพูดได้ว่าดีมานด์ $\hat{\mathcal{D}}$ จะ***เป็นส่วนหนึ่งของ*** $\mathcal{D}$ ก็ต่อเมื่อมีการจับคู่แบบหนึ่งต่อหนึ่ง $h: \mathcal{D} \rightarrow \hat{\mathcal{D}}$ ซึ่งสำหรับแต่ละ $D \in \mathcal{D}$ ทุกๆ i และ j จะได้ว่า $h(D)_{ij} \leq D_{ij}$

ทฤษฎีบทย่อยที่ 1 ถ้ากำหนดให้ f เป็นเส้นทางการส่งข้อมูลบนเครือข่าย G แล้วพิจารณาดีมานด์ใดๆ $\mathcal{D}$ และกำหนดให้ $z = CONGESTION(f, \mathcal{D})$ จะมีดีมานด์ของเครือข่ายยูนิคาสต์ $\mathcal{D}^u$ ที่ซึ่งค่า congestion ของเครือข่าย $CONGESTION(f, \mathcal{D}^u) = z$ และยิ่งไปกว่านั้นดีมานด์ $\mathcal{D}^u$ จะเป็นส่วนหนึ่งของ $\mathcal{D}$

พิสูจน์ เมื่อพิจารณาเส้นทางการส่งข้อมูลในเครือข่ายมัลติคาสต์ f ที่ให้มาแล้วนั้น และเมื่อพิจารณาด้าน $\hat{e} \in E$ ซึ่ง

$$EDGE\_CONG(\hat{e}, f, \mathcal{D}) = CONGESTION(f, \mathcal{D})$$

จะพบว่า congestion บนด้าน $\hat{e}$ คือ

$$\frac{\sum_k (\max_{j \in T_k} d_k \cdot f_{s_k j}(\hat{e}))}{c(\hat{e})}$$

เราให้นิยามของดีมานด์แบบยูนิคาสต์ $\mathcal{D}^u$ ว่าสำหรับแต่ละโหนดต้นทาง s_k เราจะเลือกเทอร์มินอลที่แย่ทีสุดสำหรับด้าน $\hat{e}$ เช่น

$$w_k = \arg\max_j \mathcal{D}_{ij}^k \cdot f_{ij}(\hat{e})$$

เช่นนั้นแล้วสำหรับเทอร์มินอล j ใดๆ เราจะกำหนด

$$U_{\mathrm{ij}}^{\mathrm{k}} = \begin{cases} \mathcal{D}_{\mathrm{ij}}^{\mathrm{k}} & \text{if j} = \varpi_{\mathrm{k}} \\ 0 & \text{otherwise} \end{cases}$$

กำหนดให้ $\mathcal{D}^u$ คือ $\{U^1, U^2, \dots, U^k\}$ และเนื่องจากจุดต้นทางแต่ละจุดจะส่งข้อมูลไปที่เทอร์มินอลเพียงจุดเดียว เพราะเป็นดีมานด์แบบยูนิคาสต์ ดังนั้นเราจะพบว่า

$$EDGE\_CONG(\hat{e}, f, \mathcal{D}^u) = EDGE\_CONG(\hat{e}, f, \mathcal{D})$$

เช่นนั้นแล้ว

$$CONGESTION(f, \mathcal{D}^u) = CONGESTION(f, \mathcal{D})$$

และจากคำจำกัดความของ U^{k} จะได้ว่า $\mathcal{D}^u$ เป็นส่วนหนึ่งของ $\mathcal{D}$ ซึ่งทำให้ทฤษฎีบทย่อยที่ 1 เป็นจริงดังที่ได้กล่าวไว้ ■

ขณะนี้เราจะพิสูจน์ทฤษฎีบทหลักของงานวิจัยในส่วนนี้

พิสูจน์ ทฤษฎีบท 1

มีข้อสังเกตว่า

$$OBLIV\_PERF\_RATIO(f) \geq OBLIV\_UNI\_PERF\_RATIO(f)$$

เนื่องจาก $OBLIV\_PERF\_RATIO(f)$ เป็นค่าที่กำหนดมาจากค่าอัตราส่วนประสิทธิภาพที่แย่ที่สุดบนดีมานด์ใดๆในเครือข่ายมัลติคาสต์ แต่จากการให้ความหมายของดีมานด์บนมัลติคาสต์ เราทราบแล้วว่าดีมานด์ในยูนิคาสต์ก็เป็นดีมานด์ของเครือข่ายมัลติคาสต์ด้วยเช่นกัน เนื่องจากปริมาณทางด้านซ้ายเป็นการหาค่าอัตราค่าสูงสุดบนเซ็ตที่มีขนาดใหญ่กว่า ดังนั้นอสมการข้างต้นจึงเป็นจริง ต่อไปเราจะพิสูจน์อสมการในทางตรงกันข้ามเพื่อพิสูจน์ทฤษฎีบทนี้

เมื่อพิจารณาดีมานด์ $\mathcal{D}$ ใดๆในเครือข่ายมัลติคาสต์ ทฤษฎีบทย่อยที่ 1 แสดงให้เราเห็นแล้วว่าเราสามารถหาดีมานด์ $\mathcal{D}^u$ ที่เป็นดีมานด์สำหรับยูนิคาสต์ที่ซึ่ง

$$CONGESTION(f, \mathcal{D}^u) = CONGESTION(f, \mathcal{D})$$

ทฤษฎีบทย่อยที่ 1 ยังแสดงว่า $\mathcal{D}^u$ เป็นส่วนหนึ่งของ $\mathcal{D}$ ดังนั้นผลลัพธ์ของเส้นทางสำหรับมัลติคาสต์ดีมานด์ $\mathcal{D}$ จะเป็นผลลัพธ์สำหรับ $\mathcal{D}^u$ โดยให้ค่าเป้าหมายเหมือนกัน เราสามารถสรุปได้ว่า

$$OPT(\mathcal{D}) \geq OPT(\mathcal{D}^u)$$

ซึ่งสามารถแสดงได้ว่า

$$PERF\_RATIO(f, \mathcal{D}) = \frac{CONGESTION(f, \mathcal{D})}{OPT(\mathcal{D})}$$
$$\leq \frac{CONGESTION(f, \mathcal{D}^u)}{OPT(\mathcal{D}^u)} = PERF\_RATIO(f, \mathcal{D}^u)$$

ดังนั้น

$$OBLIV\_PERF\_RATIO(f) \leq OBLIV\_UNI\_PERF\_RATIO(f)$$

ซึ่งทำให้ทฤษฎีบทเป็นจริง ■

2.2 การหาเส้นทางส่งข้อมูลแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ในการใช้งานจริง

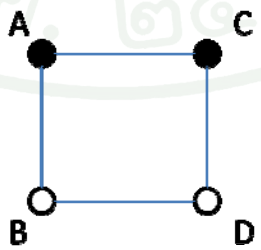
ในส่วนที่ผ่านมาเราได้แสดงว่าในทางทฤษฎีแล้วการหาเส้นทางในเครือข่ายมัลติคาสต์และยูนิคาสต์มีความเหมือนกัน แต่ในส่วนนี้เราจะนำเสนอวิธีการหาเส้นทางการส่งข้อมูลแบบออบลิเวียสบนเครือข่ายมัลติคาสต์โดยอาศัยการโปรแกรมเชิงเส้น โดยอาศัยดีมานด์หลายๆดีมานด์ซึ่งดีมานด์ที่เกิดขึ้นในเครือข่ายจะเป็นส่วนประกอบของดีมานด์ตัวอย่างเหล่านี้ซึ่งจะทำให้เราสามารถหาค่า congestion ในเครือข่ายได้

เราจะแสดงการทดลองบนเครือข่ายมัลติคาสต์จำลอง แต่การกำหนดปัญหาจะแตกต่างจากส่วนที่ผ่านมาเล็กน้อยตรงที่โหนดใดๆจะสามารถเป็นโหนดต้นทางให้กับกลุ่มมัลติคาสต์เพียงกลุ่มเดียวเท่านั้นเพื่อให้ดีมานด์เมตริกซ์หนึ่งเมตริกซ์สามารถแทนกลุ่มของมัลติคาสต์ได้ทั้งหมด

ดังนั้นจำนวนของกลุ่มมัลติคาสต์จะมีจำนวนไม่เกินจำนวนของโหนดที่มีอยู่ในเครือข่าย ในขณะที่โปรแกรมเชิงเส้นในส่วนที่ผ่านมาจะให้ความสนใจกับดีมานด์ที่แย่ที่สุดของเครือข่ายมัลติคาสต์นั้นๆ แต่ในส่วนนี้จะสนใจการใช้งานการหาเส้นทางส่งข้อมูลจริงในเครือข่ายมัลติคาสต์ โดยการสังเคราะห์ดีมานด์จากข้อมูลที่สะสมได้จากการใช้งานเครือข่าย ซึ่งดีมานด์ที่แย่ที่สุดอาจจะไม่เกิดขึ้นในการใช้งานจริงเลยก็เป็นไปได้ ดังนั้นการทดลองของเราจะให้ความสนใจไปที่ดีมานด์ที่เป็นไปได้จริงๆในระบบเครือข่ายจริง ดังนั้นดีมานด์ที่มีลักษณะเหมือนกับที่ใช้งานจริงจะถูกเลือกขึ้นมาทดลองเพื่อที่จะนำไปสร้างชุดของเส้นทางการส่งข้อมูลเพื่อให้ได้เส้นทางที่มี congestion น้อยที่สุด ดังนั้นเมื่อมีการเลือกชุดของเส้นทางการส่งข้อมูลในเครือข่ายแบบมัลติคาสต์ เราจะสามารถรับประกันค่า congestion สูงสุดที่จะเกิดขึ้นภายในเครือข่ายได้

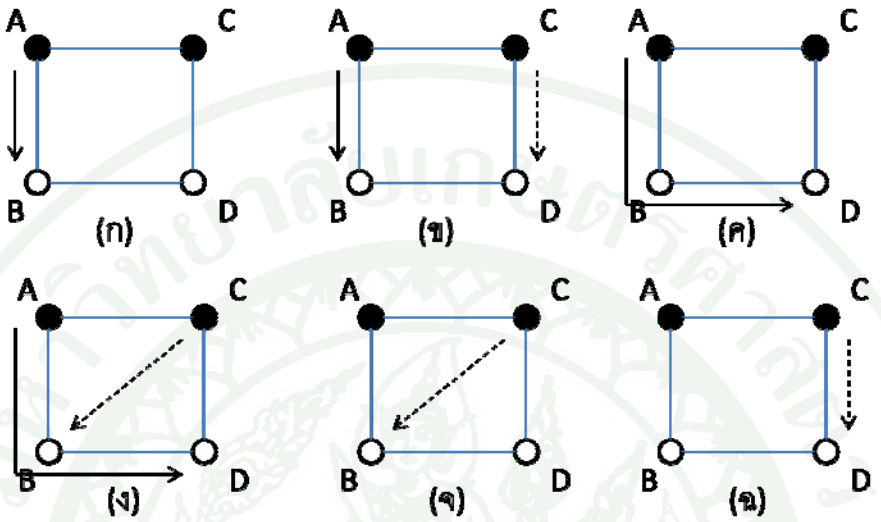
2.2.1 การใช้ข้อมูลเจาะจงมากขึ้นในการสร้างเส้นทางแบบออบลิเวียส

ในการหาเส้นทางแบบออบลิเวียสทางทฤษฎีจะเป็นการหาเส้นทางไว้รองรับทุกๆ ดีมานด์ที่อาจเป็นไปได้ทั้งหมดในเครือข่าย ทำให้เราต้องใช้ดีมานด์ในการพิจารณาเป็นจำนวนมากมายมหาศาล ทั้งๆที่ดีมานด์เหล่านั้นบางส่วนอาจจะไม่ได้เกิดขึ้นด้วยซ้ำ ทำให้ชุดของเส้นทางที่หาได้อาจจะไม่ได้ให้ประสิทธิภาพดีที่สุดสำหรับดีมานด์ที่เกิดขึ้นจริงๆ ในงานวิจัยของ Kodialam and Sengupta (2004) ได้เสนอให้พิจารณาเฉพาะดีมานด์บางส่วนที่อาจจะเกิดขึ้นได้ โดยใช้ค่า ingress และ egress บนแต่ละโหนดมาใช้ในการหาเส้นทางแบบออบลิเวียส แต่ค่า ingress และ egress ก็ยังไม่ได้ให้ข้อมูลของดีมานด์อย่างละเอียดเพียงพอ ตัวอย่างเช่น ในเครือข่ายทั่วไปนั้น การที่โหนด A มีค่า ingress เป็นหนึ่งหน่วย ในขณะที่โหนด B มีค่า egress เป็นหนึ่งหน่วย แต่ในการใช้งานจริงอาจจะไม่มีการส่งข้อมูลจากโหนด A ไปโหนด B เลยก็ได้



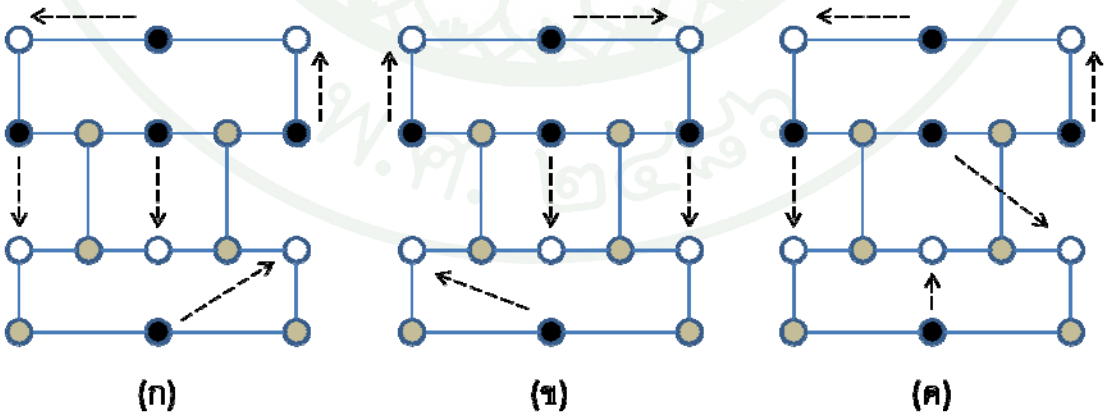
ภาพที่ 6 เครือข่ายที่ A และ C มีค่า ingress หนึ่งหน่วย ขณะที่ B และ D มีค่า egress หนึ่งหน่วย

ภาพที่ 6 แสดงเครือข่ายและค่า ingress-egress ที่แต่ละโหนด จะเห็นได้ว่ามีค่าดีมานด์ที่เป็นไปได้อยู่หกแบบด้วยกันดังภาพที่ 7



ภาพที่ 7 แสดงดีมานด์ที่เป็นไปได้จากค่า ingress-egress ในภาพที่ 6

จากดีมานด์ที่เป็นไปได้ทั้งหมดในภาพที่ 7 ถ้าสมมติว่าโหนด B เป็นโรงเรียนอนุบาล ส่วนโหนด A เป็นเว็บไซต์ข่าวการเมือง จะเห็นได้ว่าในทางปฏิบัติจะดีมานด์ในภาพที่ 7(ก) และ 7(ข) จะไม่มีโอกาสเกิดขึ้นได้เลย



ภาพที่ 8 ดีมานด์แบบต่างๆ ที่แต่ละโหนดมีค่า ingress-egress ที่เท่ากัน

ต่อไปเราจะแสดงให้เห็นว่าการตัดดีมานด์บางส่วนที่ไม่มีโอกาสเกิดขึ้นจริงในเครือข่ายจะทำให้เราสามารถสร้างเส้นทางส่งข้อมูลแบบออบลิเวียสที่ให้ค่า congestion ที่ดีกว่า

จากดีมานด์ในภาพที่ 8 สมมติว่าทุกๆลิงค์ในเครือข่ายมีแบนด์วิธเท่ากับหนึ่งหน่วย และการส่งข้อมูลแต่ละคู่เป็นข้อมูลขนาดหนึ่งหน่วยเช่นกัน พบว่าเมื่อเราใช้วิธีการส่งข้อมูลแบบออบลิเวียสจะทำให้เกิด congestion ในระบบเท่ากับ 1.5 เนื่องจากมีบางลิงค์ต้องใช้พร้อมกัน แต่ถ้าเราสามารถบอกว่าดีมานด์ที่เกิดขึ้นจริงในระบบมีเพียง (ก) กับ (ค) เท่านั้นเราจะสามารถหาเส้นทางแบบออบลิเวียสที่ให้ค่า congestion เพียง 1.0 ได้

ดังนั้นจะเห็นได้ว่าถ้าเราใช้ข้อมูลที่มีความละเอียดมากขึ้นในการสร้างเส้นทางแบบออบลิเวียสเราก็น่าจะได้เส้นทางการส่งข้อมูลที่มีค่า congestion ที่ดีขึ้น

ในส่วนถัดไปจะเป็นการอธิบายการทำงานของโปรแกรมเชิงเส้นสำหรับการหาเส้นทางจริงบนเครือข่ายมัลติคาสต์

2.2.2 โปรแกรมเชิงเส้น

เราจะอธิบายโปรแกรมเชิงเส้นที่จะใช้สำหรับหาชุดของเส้นทางการส่งข้อมูลที่มีค่า congestion ที่ต่ำที่สุดเพื่อที่จะสามารถใช้ส่งดีมานด์ของเครือข่ายแบบมัลติคาสต์ที่ต่างกัน เราจะมีการเปรียบเทียบผลลัพธ์กับค่าที่ดีที่สุดในการส่งข้อมูลของดีมานด์ในแต่ละดีมานด์โดยเฉพาะ โดยเราจะมีโปรแกรมเชิงเส้นอีกส่วนหนึ่งเพื่อมาคำนวณหาค่านี้ ดังนั้นเราจะเริ่มจากการหาค่า congestion จากการส่งข้อมูลของแต่ละดีมานด์และบันทึกไว้เป็นค่าที่ดีที่สุดสำหรับดีมานด์ชุดนั้นบนเครือข่าย หลังจากนั้นเราจะใช้ดีมานด์ทั้งหมดเพื่อประมวลผลหาชุดของเส้นทางการส่งข้อมูลที่ดีที่สุดที่สามารถส่งดีมานด์ทั้งหมดบนเครือข่าย

2.2.2.1 การหาเส้นทางที่ดีที่สุดสำหรับแต่ละดีมานด์ในเครือข่ายมัลติคาสต์

เราจะแสดงโปรแกรมเชิงเส้นเพื่อหา congestion ที่น้อยที่สุดสำหรับการส่งข้อมูลในแต่ละชุดของดีมานด์ในเครือข่ายมัลติคาสต์ ซึ่งเราจะต้องนิยามตัวแปรเพิ่มเติมดังนี้

- $f_{ij}(e)$ คือ สัดส่วนของข้อมูลหนึ่งหน่วยจะถูกส่งจากโหนดต้นทาง i ไปยังเทอร์มินอล j บนด้าน e สำหรับโหนดใดๆ $i \in V$ และ $j \in V$ และด้าน $e \in E$
- d_{ij} คือปริมาณข้อมูลที่ต้องการส่งจากจุดต้นทาง i ไปยังเทอร์มินอล j สำหรับโหนดใดๆ $i \in V$ และ $j \in V$

เราจะเริ่มต้นจากโปรแกรมเชิงเส้นที่ทำหน้าที่ในการหาเส้นทางส่งข้อมูลจำนวนหนึ่งหนึ่งซึ่งมีระหว่างทุกๆคู่ของโหนดภายในเครือข่ายโดยมีลักษณะเป็นเส้นทางย่อยๆ ซึ่งเราจะเรียกโปรแกรมเชิงเส้นส่วนแรกนี้ว่า MC-LP ดังแสดงต่อไปนี้

$$\begin{aligned}
&\textit{min} && z && \\
&\textit{Subject to} && && \\
&\textit{(Capacity)} && \sum_{ij} f_{ij}(e) \cdot d_{ij} \le z \cdot C_e && \forall e \in E \\
&\textit{(Conserv)} && \sum_{e \in out(v)} f_{ij}(e) - \sum_{e \in in(v)} f_{ij}(e) = \begin{cases} 1 & v = i \\ -1 & v = j \\ 0 & otherwise \end{cases} && \\
& && 0 \le f_{ij}(e) \le 1 && \forall e \in E, \forall i, \forall j \in \{1,2,\dots,k\}
\end{aligned}$$

แต่สำหรับโปรแกรมเชิงเส้นเพื่อหาค่า congestion ในเครือข่ายมัลติคาสต์จะต่างออกไปเล็กน้อยเนื่องจากความสามารถในการใช้แบนด์วิธร่วมกับของข้อมูลที่ถูกส่งออกมาจากจุดต้นทางจุดเดียวกัน และเนื่องจากปริมาณข้อมูลที่ต้องการส่ง d_{ij} สามารถลดจำนวนตัวแปรให้เหลือเพียง d_i ได้เนื่องจากปริมาณข้อมูลที่ส่งออกจากจุดต้นทางเดียวกันจะมีปริมาณเท่ากันหมด ดังนั้นจะมีการดัดแปลงโปรแกรมเชิงเส้นเพื่อให้เหมาะสมกับการส่งข้อมูลภายในเครือข่ายมัลติคาสต์ดังนี้

$$
\begin{aligned}
&\textit{min} && z && \\
&\textit{Subject to} && && \\
&\textit{(Capacity)} && \sum_i m_i(e) \cdot d_i \le z \cdot C_e && \forall e \in E \\
&\textit{(Bound)} && f_{ij}(e) \le m_i(e) && \forall e \in E \\
&\textit{(Conserv)} && \sum_{e \in out(v)} f_{ij}(e) - \sum_{e \in in(v)} f_{ij}(e) = \begin{cases} 1 & v = i \\ -1 & v = j \\ 0 & otherwise \end{cases} && \\
& && 0 \le f_{ij}(e) \le 1 && \forall e \in E, \forall i, \forall j \in \{1,2,\dots,k\}
\end{aligned}
$$

ซึ่งโปรแกรมเชิงเส้นมีเป้าหมายที่จะทำให้ค่า congestion ที่ได้มีค่าต่ำที่สุดโดยยังสามารถหาเส้นทางส่งข้อมูลภายในเครือข่ายมัลติคาสต์ได้ ซึ่งมีข้อบังคับ Conserv เพื่อหาเส้นทางการส่งข้อมูลขนาดหนึ่งหน่วยระหว่างทุกคู่ของโหนด และข้อบังคับ Bound เป็นส่วนที่กำหนดตัวแปร $m_{\mathrm{i}}(\mathrm{e})$ ซึ่งเป็นค่าของแบนด์วิธที่จะต้องจองไว้สำหรับแต่ละกลุ่มมัลติคาสต์ เนื่องจากข้อมูลที่ส่งออกจากจุดต้นทางเดียวกันสามารถใช้ร่วมกันได้จึงมีการจองไว้เพียงขนาดแบนด์วิธที่ต้องใช้ส่งข้อมูลที่กว้างที่สุดของข้อมูลที่ส่งมาจากโหนดต้นทาง i เท่านั้น และส่วนข้อบังคับ Capacity จะเป็นส่วนที่กำหนดค่า congestion สูงสุดของเครือข่ายจากเส้นทางส่งข้อมูลที่หาได้

2.2.2.2 การหาเส้นทางที่ดีที่สุดสำหรับดีมานด์ทั่วไปในเครือข่ายมัลติคาสต์

แนวคิดหลักของงานวิจัยนี้คือการประยุกต์ใช้โปรแกรมเชิงเส้นเพื่อทำการคำนวณเส้นทางการส่งข้อมูลที่เหมาะสมที่สุดเมื่อต้องส่งข้อมูล k ดีมานด์ที่ต่างกัน โดยใช้ชุดเส้นทางชุดเดียวกัน ดังนั้นเราจะประยุกต์โปรแกรมเชิงเส้นที่แสดงในส่วนที่แล้วเพื่อที่จะสามารถรองรับการส่งดีมานด์หลายๆ ดีมานด์ได้โดยยังสามารถรับประกันค่า congestion สูงสุดได้ด้วย อย่างไรก็ตามเราพบว่ามีประเด็นที่น่าสนใจอยู่สองประเด็นในการหาเส้นทางแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ ประเด็นแรกก็คือการหาค่าอัตราส่วนของค่า congestion สูงสุดของชุดของเส้นทางที่ได้จากโปรแกรมเชิงเส้นกับค่าที่ได้ที่ดีที่สุดเมื่อหาเส้นทางเพื่อส่งแต่ละดีมานด์โดยเฉพาะ แต่ในการใช้งานจริง เราต้องการที่จะรับประกัน congestion สูงสุดในเครือข่ายเมื่อมีการส่งดีมานด์แบบต่างๆภายในเครือข่าย ดังนั้นผลลัพธ์จากวิธีที่สองจะหาค่า congestion สูงสุดที่มีโอกาสเกิดขึ้น

ได้ในเครือข่าย ซึ่งผลการทดลองของเราจะหาเส้นทางการส่งข้อมูลที่ให้ค่าที่เหมาะสมที่สุดทั้งสองแบบ

2.2.2.3 การหาค่า congestion ที่ต่ำที่สุด

ในส่วนแรกของโปรแกรมเชิงเส้นจะคล้ายกันกับโปรแกรมเชิงเส้นของการหาเส้นทางที่ดีที่สุดของแต่ละดีมานด์แต่ตัวแปรจะมีการเพิ่มมิติเพื่อให้รองรับการคำนวณหลายๆดีมานด์ดังนี้

$$
\begin{aligned}
&\textit{min} && z \\
&\textit{Subject to} && \\
&\textit{(Capacity)} && \sum_i m_i(e) \cdot d_{ij}^t \le z \cdot C_e && \forall e \in E, \forall t \in k \\
&\textit{(Bound)} && f_{ij}(e) \le m_i(e) && \forall e \in E, \forall t \in k \\
&\textit{(Conserv)} && \sum_{e \in out(v)} f_{ij}(e) - \sum_{e \in in(v)} f_{ij}(e) = \begin{cases} 1 & v = i \\ -1 & v = j \\ 0 & otherwise \end{cases} && \\
& && 0 \le f_{ij}(e) \le 1 && \forall e \in E, \forall i, \forall j \in \{1,2,\dots,k\}
\end{aligned}
$$

ความแตกต่างของโปรแกรมเชิงเส้นในการหาค่า congestion ที่ต่ำที่สุดในการส่งข้อมูล k ดีมานด์ที่ต่างกันกับที่ผ่านมา จะเป็นการใช้เซ็ตของดีมานด์ $\mathcal{D} = \{D^1, D^2, \dots, D^k\}$ เข้ามาคำนวณพร้อมๆกัน โดย d_{ij}^t จะเป็นค่าของปริมาณข้อมูลที่ต้องการส่งจากโหนด i ไปยังโหนด j ของดีมานด์ที่ t

2.2.2.4 การหาค่าอัตราส่วน congestion ที่ต่ำที่สุด

การหาค่า congestion ที่ต่ำที่สุดดูเหมือนจะสามารถรับประกันการใช้งานได้ระดับหนึ่ง แต่เนื่องจากจุดประสงค์คือต้องการลด congestion สูงสุดซึ่งแน่นอนว่าคงไม่มีทางต่ำกว่าค่าที่ได้จากการหาค่าที่ดีที่สุดของค่าสูงสุดในเหล่าดีมานด์ที่นำมาใช้ในการทดลอง และจะยังไม่มีการรับประกันว่าดีมานด์ที่ให้ค่า congestion ต่ำอยู่แล้ว เมื่อนำมาใช้กับชุดการส่ง

ข้อมูลที่ได้จะได้ค่าต่ำเหมือนเดิมหรือได้ค่าสูงเกือบเท่าค่า congestion ของดีมานด์ที่แย่ที่สุดที่ได้จากโปรแกรมเชิงเส้นในส่วนที่แล้วหรือไม่

ในส่วนนี้โปรแกรมเชิงเส้นจะมีการเปลี่ยนจุดประสงค์ของโปรแกรมเชิงเส้นเนื่องมาจากจะต้องใช้ค่าของ congestion ที่ดีที่สุดของแต่ละดีมานด์ σ^t ประกอบเข้ามาเพื่อจะคำนวณหาอัตราส่วนที่ดีที่สุด ซึ่งโปรแกรมเชิงเส้นจะมีดังนี้

$$
\begin{aligned}
&\textit{min} && \alpha && \\
&\textit{Subject to} && && \\
&\textit{(Capacity)} && \sum_i m_i(e) \cdot d_{ij}^t \le \alpha \cdot \sigma^t \cdot C_e && \forall e \in E, \forall t \in k \\
&\textit{(Bound)} && f_{ij}(e) \le m_i(e) && \forall e \in E, \forall t \in k \\
&\textit{(Conserv)} && \sum_{e \in out(v)} f_{ij}(e) - \sum_{e \in in(v)} f_{ij}(e) = \begin{cases} 1 & v = i \\ -1 & v = j \\ 0 & otherwise \end{cases} && \\
& && 0 \le f_{ij}(e) \le 1 && \forall e \in E, \forall i, \forall j \in \{1,2,\dots,k\}
\end{aligned}
$$

โปรแกรมเชิงเส้นนี้จะมีเป้าหมายเพื่อทำให้ค่า α มีค่าต่ำที่สุดเพื่อจะรับประกันว่าเส้นทางแบบออบลิเวียสที่ได้จะให้ค่า congestion ต่างจากเส้นทางที่ดีที่สุดของแต่ละดีมานด์น้อยที่สุด

2.2.3 การรับประกันค่าประสิทธิภาพทางทฤษฎี

ในงานวิจัยในส่วนที่แล้วเราได้อธิบายโปรแกรมเชิงเส้นที่ทำหน้าที่หาเส้นทางการส่งข้อมูลที่ดีที่สุดของเซ็ตของดีมานด์ที่กำหนดมาให้ ต่อไปเราจะแสดงทฤษฎีบทย่อยที่แสดงถึงผลลัพธ์ที่ได้จากการใช้ดีมานด์ที่เป็นคอนเว็กซ์คอมบิเนชั่น (convex combination) ของดีมานด์เหล่านั้น

ทฤษฎีบทย่อยที่ 2 ถ้าโปรแกรมเชิงเส้นในส่วนที่ผ่านมาทำการค้นหาค่า congestion ที่น้อยที่สุด (หรือค่าอัตราส่วน congestion ที่ดีที่สุด) ซึ่งกำหนดให้เป็น z สำหรับเซ็ต

ของดีมานด์ $\mathcal{D} = \{D^1, \dots, D^n\}$ ซึ่งสำหรับดีมานด์ D ใดๆ ซึ่งเป็นคอนเว็กซ์คอมบิเนชั่นของดีมานด์เหล่านั้น ค่า congestion สูงสุด(หรอค่าอัตราส่วน congestion สูงสุด) จะมีค่าสูงสุดไม่เกิน z

พิสูจน์:

เราจะกล่าวว่าดีมานด์ D เป็นคอนเว็กซ์คอมบิเนชั่นของเซ็ต $\mathcal{D}$ ก็ต่อเมื่อ

$$D = \alpha_1 D^1 + \alpha_2 D^2 + \alpha_3 D^3 + \cdots + \alpha_n D^n$$

เมื่อ

$$\sum_i \alpha_i = 1$$

พิจารณาด้าน $é$ ซึ่งเกิด congestion z_k สูงที่สุดเมื่อส่งดีมานด์ $D^k \in \mathcal{D}$ ด้วยชุดเส้นทางการส่งข้อมูล f ดังนั้น

$$z_k = congestion_k(é) = \frac{\sum_{i,j} f_{ij}(é) \cdot d_{ij}^k}{c(é)}$$

ปริมาณข้อมูลบนด้าน $é$ สำหรับดีมานด์ D จึงมีค่าเท่ากับ

$$\sum_{i,j} \left(f_{ij}(é) \cdot \sum_n \alpha_n \cdot d_{ij}^n \right) \leq \sum_{i,j} f_{ij}(é) \cdot d_{ij}^k$$

ดังนั้น

$$congestion_{convex}(é) \leq z_k \leq z$$

ซึ่งทำให้ทฤษฎีบทย่อยที่ 2 เป็นจริง ■

2.2.4 การจำลองการทำงานของรูปแบบการหาเส้นทางแบบออบลิเวียส

เราใช้วิธีการจำลองการทำงานเช่นเดียวกันกับงานวิจัยในส่วนแรกที่เกี่ยวกับการหาเส้นทางสำรองของเครือข่ายมัลติคาสต์ เพื่อที่จะวัดผลประสิทธิภาพสำหรับอัลกอริทึมที่ได้

นำเสนอ ซึ่งการทดลองของเราสามารถอธิบายได้โดยเริ่มจาก แต่ละส่วนประกอบของการทดลองสร้างขึ้นได้อย่างไร ซึ่งพารามิเตอร์ของแต่ละชุดการทดลองประกอบไปด้วยจำนวนของโหนด n และจำนวนของดีมานด์ t

2.2.4.1 การสร้างกราฟเครือข่ายที่นำมาใช้ในการทดลอง

สำหรับการจำลองการทำงานของโปรแกรมเชิงเส้น เราจะทดสอบบนกราฟเครือข่ายจริง 2 แบบ รวมไปถึงบนกราฟแบบสุ่มอีกหนึ่งรูปแบบ เราจะใช้เครือข่าย GEANT และ ABILENE และกราฟเครือข่ายแบบสุ่มที่ได้จากโปรแกรม BRITE simulator เช่นเดียวกับการทดลองในส่วนที่ผ่านมา เพื่อทดสอบการทำงานของโปรแกรมเชิงเส้นที่ได้นำเสนอในงานวิจัยส่วนนี้ โดยพารามิเตอร์สำคัญของกราฟเครือข่ายแบบสุ่มจะมีดีกรีเฉลี่ย 2

ซึ่งในการทดลองเราจะพิจารณาแทนเครือข่ายจริงด้วยกราฟสามประเภทที่ได้กล่าวมาโดยให้โหนดแต่ละโหนดแทนอุปกรณ์ภายในเครือข่ายซึ่งก็คือเราท์เตอร์และด้านแต่ละด้านจะให้เป็นเส้นเชื่อมต่อข้อมูลระหว่างเราท์เตอร์แต่ละตัว ซึ่งด้านแต่ละด้านละด้านจะถูกแบ่งออกเป็นด้านที่มีทิศทางสองด้านที่มีขนาดเท่ากันแต่ทิศทางตรงกันข้ามเพื่อแสดงการชื่อต่อที่มีการรับข้อมูลและส่งข้อมูลในแต่ละเส้นทาง และในการทดลองเราจะกำหนดให้ทุกๆเส้นทางข้อมูลมีแบนด์วิธเป็นหนึ่งหน่วย แต่ละดีมานด์ของเครือข่ายจะสร้างโดยคำนวณจากค่า ingress และ egress ของโหนดในระบบเพื่อเป็นตัวอย่างของกลุ่มมัลติคาสต์ที่จะใช้ในระบบ แต่จะมีการกำหนดจำนวนของจุดปลายทางต่อหนึ่งกลุ่มมัลติคาสต์เพื่อไม่ให้จำนวนกลุ่มของมัลติคาสต์มีจำนวนน้อยจนเกินไป

ในหนึ่งชุดการทดลองจะเป็นการใช้ดีมานด์ 10 ชุดที่สร้างจากค่า ingress และ egress ของโหนดต่างๆในเครือข่าย จากนั้นจะนำไปหาค่า congestion ต่ำสุดในการหาเส้นทางสำหรับแต่ละดีมานด์โดยเฉพาะ จากนั้นจะบันทึกค่าทั้งหมดเพื่อนำไปใช้ในการทดลองเพื่อหาอัตราส่วน congestion ที่น้อยที่สุด และบันทึกค่า congestion จากดีมานด์ที่มีค่าสูงที่สุดเพื่อนำมาเปรียบเทียบกับค่า congestion ที่ได้จากการทดลองหาเส้นทางแบบออบลิเวียสที่ congetion ต่ำที่สุด

โดยการหาเส้นทางแบบออบลิเวียสที่มีค่า congestion ต่ำที่สุดจะใช้การโปรแกรมเชิงเส้นใน 2.2.2.3 ที่ใช้ดีมานด์พร้อมกันทั้ง 10 ดีมานด์เพื่อหาเส้นทางการส่งข้อมูล

ที่ให้ค่า congestion ที่ต่ำที่สุดเพื่อนำค่าที่ได้ไปเปรียบเทียบกับค่าที่ได้จาก congestion ที่ได้จากการหาเส้นทางก่อนหน้านี้ จากนั้นจะนำดีมานด์ทั้งหมดไปคำนวณโดยโปรแกรมเชิงเส้นใน 2.2.2.4 เพื่อหาอัตราส่วน congestion ที่ต่ำที่สุด โดยนำค่าที่ได้จากการหาเส้นทางสำหรับแต่ละดีมานด์มาใช้เพื่อหาอัตราส่วนที่ต่ำที่สุด การทดลองที่กล่าวมาจะทำการทดลองเพื่อหาเส้นทางแบบออบลิเวียสทั้งหมด 400 ครั้ง เป็นการหาค่า congestion ที่ต่ำที่สุด 200 ครั้ง และหาค่าอัตราส่วน congestion ที่ต่ำที่สุดอีก 200 ครั้ง และนำค่าที่ได้มาหาค่าเฉลี่ยและเปรียบเทียบผล

ผลและวิจารณ์

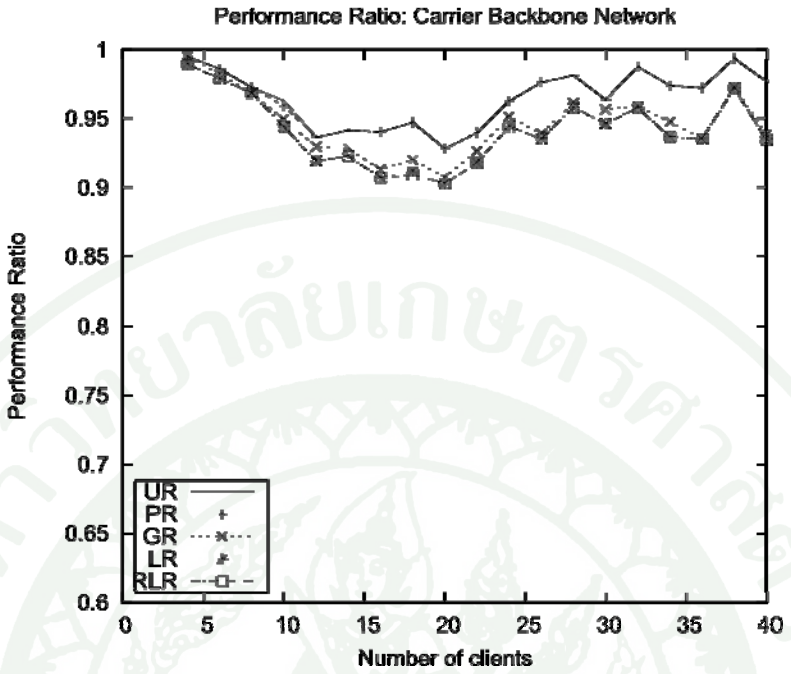
การสร้างเส้นทางสำรองบนเครือข่ายมัลติคาสต์

1. การเปรียบเทียบค่าอัตราส่วนประสิทธิภาพได้จากแต่ละรูปแบบการกู้คืน

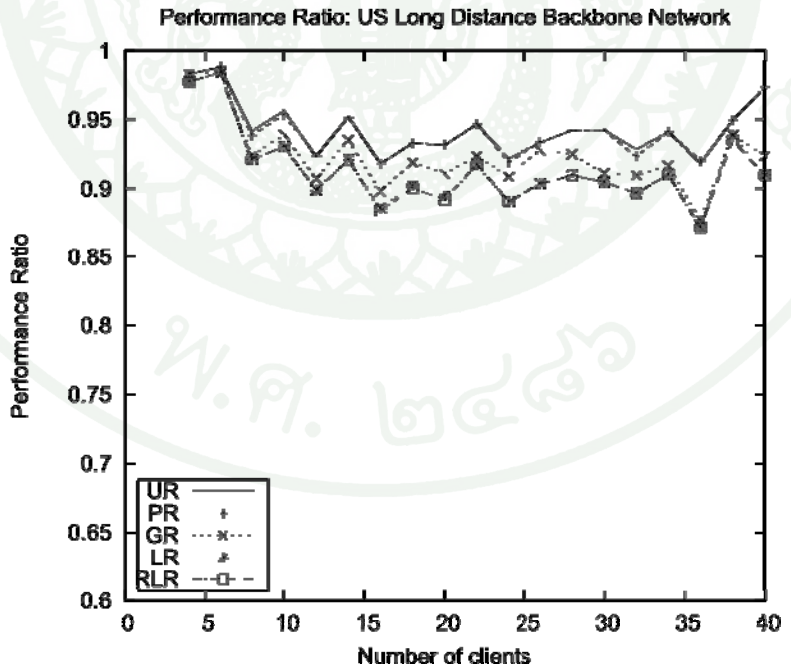
ในส่วนนี้เราจะวัดผลประสิทธิภาพของ recovery scheme ทั้งหมดของเครือข่ายมัลติคาสต์ที่ได้อธิบายไปแล้วในส่วนข้างต้น โดยจะทำการทดสอบการทำงานของแต่ละ scheme บนระบบเครือข่ายสองประเภท ประเภทแรกก็คือเครือข่ายที่ได้จากการสุ่มในโมเดลของ Barabasi-Albert และประเภทที่สองก็คือเครือข่ายที่มีการใช้งานจริง

สำหรับเครือข่ายแบบสุ่มเราจะสร้างโดยมีอ้างอิงกับรูปแบบของ Barabasi-Albert ซึ่งมีเป้าหมายในการสร้างที่ต่างกันอยู่สองประเภท ก็คือ (ก)การสร้างกราฟที่มีความหนาแน่นของด้านสูงซึ่งให้ชื่อในการทดลองว่า Random-dense ซึ่งมีจำนวนโหนดในเครือข่ายจำนวน 20 โหนด มีด้านทั้งหมด 59 ด้าน และ (ข)กราฟเครือข่ายที่มีความหนาแน่นของด้านน้อยให้ชื่อว่า Random-sparse ประกอบด้วย 20 โหนดเท่ากันแต่มีด้านเพียง 37 ด้าน และในส่วนของเครือข่ายที่มีการใช้งานจริงที่นำมาทดสอบรูปแบบการกู้คืนในงานวิจัยนี้เราใช้เครือข่ายที่ใช้ในงานวิจัยของ Lau et al. (2005) โดยใช้ (ค)เครือข่าย US backbone ซึ่งประกอบด้วย 28 โหนดและ 45 ด้านซึ่งในงานวิจัยนี้จะเรียกว่า US-Backbone และ(ง)เครือข่าย carrier backbone ซึ่งประกอบด้วยโหนด 25 โหนดและ 28 ด้านซึ่งจะเรียกว่า Carrier-Backbone และจำนวนเทอร์มินอลที่จะใช้สำหรับเครือข่ายแต่ละเครือข่ายจะแปรผันระหว่าง 4 โหนดไปจนถึง 40 โหนด สำหรับแต่ละพารามิเตอร์สำหรับเครือข่าย ในงานวิจัยนี้จะทำการทดลอง 50 ครั้งสำหรับแต่ละจุดบนกราฟเพื่อหาค่าเฉลี่ยสำหรับแต่ละ recovery scheme ซึ่งมีทั้งหมดห้ารูปแบบ บนกราฟ 4 ชนิด รวมแล้วทำการทดลองประมาณ 20,000 ครั้ง

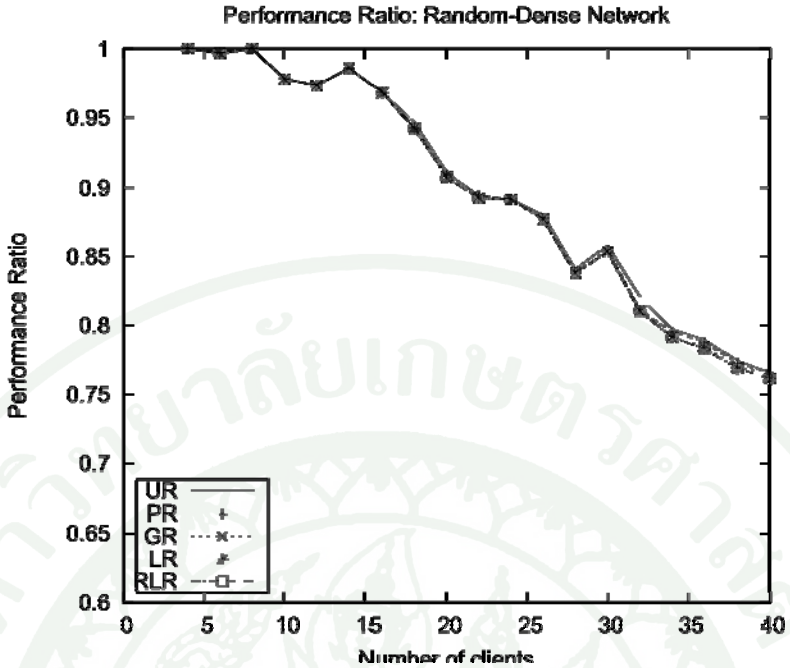
ผลของการทำงานของ recovery scheme จะถูกแสดงผลในรูปแบบของค่าอัตราส่วนประสิทธิภาพ โดยค่าอัตราส่วนประสิทธิภาพเป็นอัตราส่วนระหว่างค่าที่ได้กับค่าที่ดีที่สุดซึ่งในที่นี้ค่าที่ดีที่สุดจะคำนวณได้จากโปรแกรมเชิงเส้นที่คำนวณหาเส้นทางการส่งข้อมูลใหม่ทั้งหมดเมื่อเกิดปัญหากับด้านใดด้านหนึ่ง หรืออธิบายให้ชัดเจนได้อีกแบบหนึ่งว่าในเรามีชุดของการส่งข้อมูลสำรองใหม่ทั้งเครือข่ายสำหรับด้านทุกๆด้านในเครือข่ายถ้าเกิดปัญหาขึ้น ซึ่งผลลัทธ์ได้แสดงในภาพที่ 9 ถึง 12



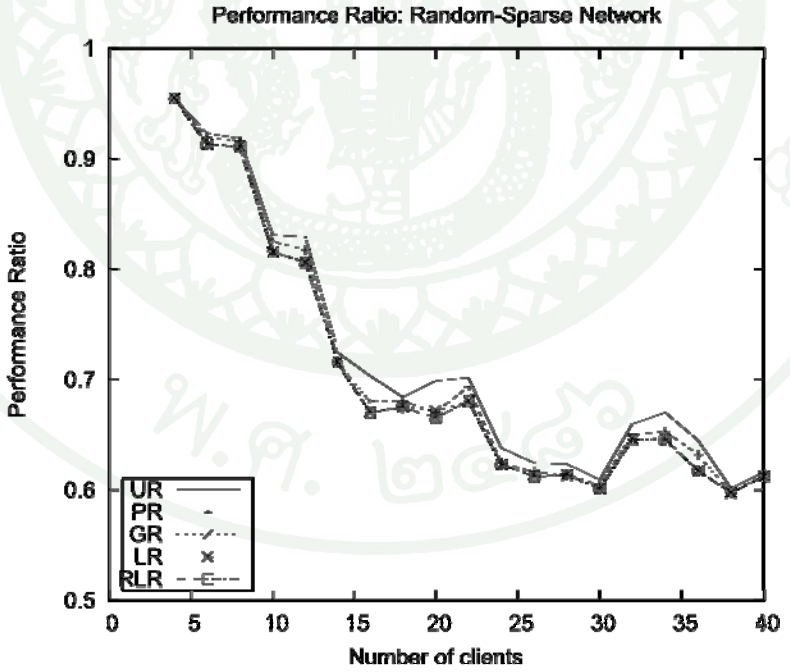
ภาพที่ 9 แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย Carrier-Backbone



ภาพที่ 10 แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย US-Backbone



ภาพที่ 11 แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย Random-Dense



ภาพที่ 12 แสดงอัตราส่วนประสิทธิภาพของ recovery scheme บนเครือข่าย Random-Sparse

ภาพที่ 9 แสดงค่าอัตราส่วนประสิทธิภาพของ recovery scheme ทั้งหมดบนเครือข่าย Carrier-backbone ซึ่งจากกราฟจะเห็นได้ว่าอัตรส่วนประสิทธิภาพของ UR กับ PR มีความใกล้เคียงกันมาก โดย scheme ทั้งสองให้ประสิทธิภาพที่ดีกว่า GR, LR และ RLR โดยประมาณ 2% และผลลัพธ์ที่คล้ายกันก็ยังแสดงให้เห็นในภาพที่ 10 ซึ่งเป็นกราฟของเครือข่าย US-Backbone อีกด้วย และเมื่อเราประยุกต์ recovery scheme กับกราฟเครือข่ายที่ได้จากการสุ่มจากโปรแกรม BRITE Simulator ทั้ง Random-Dense และ Random-Sparse ก็พบว่าค่าอัตราส่วนประสิทธิภาพของแต่ละ scheme ก็เป็นไปในทิศทางเดียวกัน

2. การเปรียบเทียบจำนวนแบคอัพทรี

ในส่วนนี้เราจะทำการเปรียบเทียบค่าขอบเขตของจำนวนแบคอัพทรีของกลุ่มมัลติคาสต์สำหรับ recovery scheme แบบต่างๆ ซึ่งจำนวนของแบคอัพทรีมีความสำพันธ์กับขนาดของหน่วยความจำที่จะต้องใช้ในแต่ละ scheme โดยตรง recovery scheme ทั้งหมดยกเว้นการ PR จะใช้ด้านเป็นตัวกำหนดการสร้างแบคอัพทรี ซึ่งจำนวนด้านที่มีการใช้เป็นส่วนประกอบในไพรมารีทรีนี้เองที่เป็นตัวกำหนดค่าขอบเขตของจำนวนแบคอัพทรี ในทางตรงกันข้ามใน PR จำนวนของแบคอัพทรีจะไม่เกินจำนวน maximal disjoint path ที่ได้จาก primary tree ทั้งหมด ค่าเฉลี่ยของจำนวนแบคอัพทรีที่ได้จาก UR และ PR ได้แสดงให้เห็นในตารางที่ 3

จากการทดลองข้างต้นเราพบว่าจำนวนของแบคอัพทรีใน UR มีจำนวนมากกว่าจำนวนแบคอัพทรีที่ใช้ใน PR อยู่ประมาณ 2.5 เท่า

เนื่องจาก scheme ที่เหลือไม่ได้มีการสร้างแบคอัพทรีใหม่ทั้งต้น ดังนั้นเพื่อเป็นการเปรียบเทียบที่ชัดเจน เราจึงใช้วิธีนับจำนวนลิงค์ที่ประกอบในแบคอัพทรีแทน ดังตารางที่ 4

ตารางที่ 3 แสดงจำนวนแบคอัพทรีเฉลี่ย

	Carrier-Backbone	US-Backbone	Random-Dense	Random-Sparse
PR	11.8	13.94	23.31	24.45
UR	26.52	39.41	25.95	27.85

ตารางที่ 4 แสดงจำนวนเฉลี่ยของลิงค์ที่ใช้ในการสร้างเส้นทางสำรองต่อ 1 กลุ่มมัลติคาสต์

	Carrier-Backbone	US-Backbone	Random-Dense	Random-Sparse
PR	43.85	56.28	54.64	44.93
UR	77.02	91.31	73.66	61.51
GR	12.69	15.11	8.65	9.74
LR	11.74	11.11	8.38	9.03
RLR	17.32	23.63	17.79	16.63

จากตารางที่ 4 เราพบว่าเมื่อนับจากจำนวนลิงค์ที่ใช้สำหรับเส้นทางสำรองสำหรับหนึ่งกลุ่มมัลติคาสต์ PR ก็ยังใช้ลิงค์น้อยกว่า UR ซึ่งมีแนวโน้มตามจำนวนแบคอัพทรีที่ใช้ดังตารางที่ 3 เช่นเดียวกัน แต่เมื่อนำไปเปรียบเทียบกับ scheme อื่นพบว่า GR, LR และ RLR มีการใช้ลิงค์ที่น้อยกว่าเนื่องจากการสร้างเส้นทางสำรองไม่จำเป็นต้องมีการสร้างใหม่ทั้งหมดทำให้มีการใช้ลิงค์น้อยกว่าซึ่งเป็นไปตามที่ได้คาดการณ์เอาไว้

3. การเปรียบเทียบเวลาที่ใช้ในการประมวลผล

ตารางที่ 5 แสดงให้เห็นถึงเวลาในการประมวลผลเฉลี่ยของแต่ละ recovery scheme สำหรับแต่ละดีมานด์ เมื่อใช้งานบนระบบเครือข่ายเดียวกัน ระยะเวลาในการประมวลผลหาได้จากเวลาที่โปรแกรมใช้ในการแก้สมการเชิงเส้นของแต่ละ scheme ซึ่งค่าที่ได้พบว่าโดยเฉลี่ยแล้วโปรแกรมเชิงเส้นของ PR ใช้เวลาเพียง 1 ใน 5 ของเวลาที่โปรแกรมเชิงเส้นสำหรับ UR ใช้เท่านั้น ซึ่งการทดสอบทั้งหมดเป็นการเปรียบเทียบ recovery scheme บนเครือข่ายที่เหมือนกันและมีไพรมารีทรีเหมือนกัน ในที่นี้เราต้องจดจำไว้ว่า PR ให้ผลลัพธ์ที่ดีกว่า scheme อื่นๆ ยกเว้นเว้น RLR แต่อย่างไรก็ตามจากผลการทดลองแสดงให้เห็นแล้วว่า RLR ให้ผลลัพธ์ประสิทธิภาพของการส่งข้อมูลภายในเครือข่ายมัลติคาสต์แย่ที่สุดที่มีการพิจารณาในงานวิจัยนี้

ผลของการทดลองสร้างเส้นทางสำรองบนระบบเครือข่ายมัลติคาสต์ โดยใช้ recovery scheme แบบต่างๆ ให้ผลเป็นไปตามที่ได้คาดหมายเอาไว้ ดังที่ได้แสดงไว้ในผลการทดลองในส่วนที่ผ่านมาที่ว่า UR ให้ประสิทธิภาพดีที่สุดเนื่องจากมีการคำนวณเส้นทางการส่งข้อมูลของเส้นทางหลักที่ใช้ในการส่งข้อมูลใหม่ทั้งหมดสำหรับเส้นทางที่ได้รับผลกระทบจากเหตุการณ์ที่เกิดปัญหา

กับด้านใดด้านหนึ่งในไพรมารีทรี และ PR ที่ได้มีการนำเสนอในงานวิจัยนี้ก็ให้ประสิทธิภาพเกือบเทียบเท่ากับ UR เลยทีเดียว แต่ PR มีการใช้จำนวนของแบคอัพทรีเพียง 61% ของ UR และสามารถประมวลผลเพื่อหาเส้นทางสำรองได้เร็วกว่าถึง 10 เท่า ที่เป็นเช่นนี้เนื่องจากการที่เส้นทางการส่งข้อมูลสำรองของด้านที่มีปัญหาใน failed path เดียวกันสามารถใช้ร่วมกันได้

ตารางที่ 5 แสดงเวลาเฉลี่ยที่ใช้ในการประมวลผล(เป็นวินาที)

	Carrier-Backbone	US-Backbone	Random-Dense	Random-Sparse
UR	23.37	172.98	202.11	59.37
PR	4.11	18.95	18.95	9.47
GR	15.78	100.14	98.53	43.58
LR	14.53	96.63	139.58	42.95
RLR	0.95	3.47	6.32	3.16

นอกจากนั้นยังสามารถสรุปได้ว่าเวลาที่ใช้ในการประมวลผลของ PR ดีที่สุดรองจากเวลาที่ใช้ในการประมวลผลของ RLR แต่อย่างไรก็ตามถ้าวัดประสิทธิภาพของการส่งข้อมูลในเครือข่ายจะพบว่าประสิทธิภาพของการส่งข้อมูลของ PR จะดีกว่า RLR อยู่ประมาณ 3% ในเครือข่ายที่ใช้งานจริง และดีกว่า 0.5% ในเครือข่ายแบบสุ่มที่นำมาทดลอง แต่เนื่องจากในการใช้งานจริงการคำนวณหาเส้นทางแบบออบลิเวียสจะทำเพียงครั้งเดียว ปัจจัยทางด้านเวลาจึงมีความสำคัญน้อยกว่าเรื่องของประสิทธิภาพและพื้นที่หน่วยความจำที่ใช้

ดังนั้นถ้าเราให้ประสิทธิภาพของ recovery scheme เป็นตัวเลือกแรกแล้วให้พื้นที่หน่วยความจำเป็นปัจจัยรองลงมาแล้ว PR จึงเป็นตัวเลือกที่ดีตัวหนึ่งเนื่องจากให้ประสิทธิภาพที่ใกล้เคียงกันแต่ใช้พื้นที่หน่วยความจำน้อยกว่ากันมาก

การสร้างเส้นทางส่งข้อมูลบนเครือข่ายมัลติคาสต์แบบออบลิเวียส

1. การเท่ากันของเครือข่ายยูนิคาสต์และมัลติคาสต์ในแง่ของการส่งข้อมูลแบบออบลิเวียส

ในส่วนนี้เป็นการพิสูจน์ทางทฤษฎีเพื่อแสดงให้เห็นว่าสามารถใช้เทคนิคในการหาเส้นทางการส่งข้อมูลแบบออบลิเวียสในเครือข่ายแบบยูนิคาสต์มาใช้ในเครือข่ายแบบมัลติคาสต์ได้ดังแสดงให้เห็นในการพิสูจน์ทฤษฎีบทที่ 1 ดังนั้นเราสามารถนำเทคนิคในการหาเส้นทางแบบออบลิเวียสใหม่ๆในอนาคตมาประยุกต์เพื่อใช้กับระบบเครือข่ายแบบมัลติคาสต์ได้เช่นกัน

2. การหาเส้นทางส่งข้อมูลแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ในการใช้งานจริง

ในส่วนนี้เราจะทำการเปรียบเทียบค่า congestion ที่ได้จากการสร้างเส้นทางแบบปรับเปลี่ยนสำหรับแต่ละชุดดีมานด์ กับ congestion ที่ได้จากการสร้างเส้นทางแบบออบลิเวียส ทั้งแบบที่มีจุดประสงค์เพื่อหาค่า congestion ที่ต่ำที่สุด และแบบที่มีจุดประสงค์เพื่อให้ค่าอัตราส่วนของ congestion ที่ได้จากการส่งข้อมูลแบบออบลิเวียสกับเส้นทางเฉพาะของดีมานด์นั้นมีค่าต่ำที่สุด

ตารางที่ 6 แสดงค่า congestion จากการส่งข้อมูลแบบออบลิเวียส

กราฟเครือข่าย	การส่งข้อมูลแบบออบลิเวียส	
	ค่าเฉลี่ย congestion	ค่าเฉลี่ยอัตราส่วน congestion
GEANT	3.713	1
ABILENE	2.886	1.012
Random	1.699	1.034

ตารางที่ 6 แสดงให้เห็นถึงค่าเฉลี่ย congestion จากการหาเส้นทางแบบออบลิเวียสที่ใช้สำหรับส่งดีมานด์ทั้งชุด และค่าเฉลี่ยอัตราส่วน congestion ของการหาส่งข้อมูลแบบออบลิเวียสภายในเครือข่ายมัลติคาสต์ เมื่อพิจารณาจากค่าเฉลี่ยอัตราส่วน congestion ที่ได้พบว่ามีค่ามากกว่าค่า congestion ที่ได้จากการหาเส้นทางแบบปรับเปลี่ยนสำหรับแต่ละดีมานด์เพียงเล็กน้อยเท่านั้น ส่วนค่าเฉลี่ย congestion ที่ได้จากการทดลองจะยังไม่สามารถบอกรายละเอียดอะไรได้มากเนื่องจาก

ไม่ได้มีการเปรียบเทียบกับค่าใดๆ ดังนั้นในส่วนถัดไปเราจะแสดงความสัมพันธ์กับค่าเฉลี่ย congestion ที่ได้ กับค่า congestion จากวิธีการหาเส้นทางแบบปรับเปลี่ยน

ค่าเฉลี่ย congestion สูงสุดในวิธีการหาเส้นทางแบบปรับเปลี่ยนได้ หาได้จากหารหาค่า congestion ของแต่ละดีมานด์ในชุดการทดลองด้วยวิธีการหาเส้นทางแบบปรับเปลี่ยนได้ แล้วนำค่า congestion ที่สูงที่สุดของดีมานด์ที่ได้ มาหาค่าเฉลี่ย จากตารางที่ 7 จะเห็นได้ว่าค่าเฉลี่ย congestion สูงสุดของทั้งสองวิธีให้ค่าใกล้เคียงกันมาก นั่นคือเราสามารถรับประกันได้ว่า congestion สูงสุดที่จะเกิดในเครือข่ายเมื่อใช้ส่งดีมานด์ตัวอย่างมีค่าไม่แตกต่างกัน

และจากผลของการทดลองหาเส้นทางการส่งข้อมูลแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ พบว่าค่า congestion บนเครือข่ายจำลองที่ได้จะมีค่าแตกต่างจากค่าที่ดีที่สุดที่เป็นไปได้เพียงเล็กน้อยเท่านั้น ซึ่งถ้าเราสามารถเก็บข้อมูลของการส่งข้อมูลจริงภายในเครือข่ายได้มากขึ้น จนสามารถระบุได้ว่ากรณีของดีมานด์ที่ทำให้ congestion ในระบบมีค่าสูงไม่เกิดขึ้นจริงๆ ในระบบ ก็จะทำให้เราสามารถเลือกเส้นทางแบบออบลิเวียสที่ให้ค่า congestion ที่ต่ำลงอีกด้วย

ตารางที่ 7 เปรียบเทียบค่าเฉลี่ย congestion สูงสุดระหว่าวิธีการหาเส้นทางส่งข้อมูลแบบปรับเปลี่ยนได้กับวิธีการหาเส้นทางแบบออบลิเวียส

กราฟเครือข่าย	ค่าเฉลี่ย congestion สูงสุด	
	วิธีหาเส้นทางแบบปรับเปลี่ยนได้	วิธีหาเส้นทางแบบออบลิเวียส
GEANT	3.713	3.713
ABILENE	2.885	2.886
Random	1.699	1.699

จากการพิสูจน์ทฤษฎีบทที่ 1 สามารถแสดงให้เราเห็นว่าเราสามารถใช้อัลกอริทึมแบบเดียวกันกับที่ใช้ในเครือข่ายยูนิคาสต์เพื่อหาเส้นทางการส่งข้อมูลในเครือข่ายมัลติคาสต์ได้ และจากทฤษฎีบทที่ 2 สามารถแสดงให้เห็นว่าถ้าเราสามารถหาตัวอย่างดีมานด์ที่มีดีมานด์ที่เหลือเป็นคอนเว็กซ์คอมบิเนชั่น มาใช้เป็นดีมานด์ตั้งต้นสำหรับสร้างเส้นทางแบบออบลิเวียส เราก็สามารถรับประกันได้ว่าค่า congestion ที่เกิดขึ้นจะไม่เกินค่าที่เราหาได้จากการโปรแกรมเชิงเส้นนั้น

สรุปและข้อเสนอแนะ

การสร้างเส้นทางสำรองบนเครือข่ายมัลติคาสต์

ในงานวิจัยนี้เราได้นำเสนออัลกอริทึมซึ่งใช้หลักการของโปรแกรมเชิงเส้นเพื่อที่จะเพิ่มจำนวนของข้อมูลที่สามารถส่งได้ในเครือข่ายแบบมัลติคาสต์ให้มากที่สุดภายใต้ recovery scheme ต่างๆ โดยมีสมมติฐานที่ว่าข้อมูลที่ส่งออกไปสามารถแบบส่งเป็นส่วนย่อยๆ และสามารถนำมารวมกันเพื่อใช้งานได้ที่จุดปลายทางต่างๆ ซึ่งเราได้ทำการทดลองและเปรียบเทียบผลการทำงานของ recovery scheme หลายๆแบบ และได้นำเสนอ PR ซึ่งมีการปรับปรุงมาจาก UR ซึ่งเป็นซึ่งให้ประสิทธิภาพของจำนวนข้อมูลที่สามารถส่งได้ในเครือข่ายดีที่สุด สำหรับงานวิจัยนี้ยังมีส่วนที่สามารถปรับปรุงพัฒนาได้อีก อย่างแรกก็คือในงานวิจัยนี้ไพรมารีทรีถูกกำหนดมาให้อยู่ก่อนแล้ว แต่ถ้าเราสามารถสร้างไพรมารีทรีได้เองและเป็นประเภทที่สามารถส่งข้อมูลเป็นส่วนย่อยได้ จะทำให้ประสิทธิภาพของการใช้งานเครือข่ายที่ได้ดียิ่งขึ้นได้โดยสามารถใช้การโปรแกรมเชิงเส้นในงานวิจัยนี้เป็นพื้นฐานได้อีกด้วย ส่วนที่สองก็คือแทนที่เราจะเป็นผู้กำหนดการเลือกใช้ recovery scheme บางทีอาจเป็นไปได้ที่ตัวของโปรแกรมเชิงเส้นเองสามารถค้นหา scheme ที่เหมาะสมกับรูปแบบการใช้งานของเครือข่ายตามที่กำหนดมาในข้อมูลได้ด้วย

การสร้างเส้นทางส่งข้อมูลบนเครือข่ายมัลติคาสต์แบบออบลิเวียส

เราสามารถหาเส้นทางของการส่งข้อมูลบนเครือข่ายมัลติคาสต์โดยพิจารณาดีมานด์หลายๆ ดีมานด์ในการสร้างชุดของเส้นทางการส่งข้อมูลเพียงหนึ่งชุดที่ให้ผลลัพธ์ที่ดีได้ เราพบว่าค่า congestion ที่ได้จากการโปรแกรมเชิงเส้นที่ประมวลผลดีมานด์ทีละหลายๆดีมานด์ จะมีค่าใกล้เคียงกับค่า congestion ที่ได้จากดีมานด์ที่ให้ค่า congestion สูงสุด ซึ่งหมายความว่าค่าที่เรารับประกันได้ยังมีค่าไม่ต่างจากค่าที่ดีที่สุด ซึ่งในส่วนนี้ถ้าเรามีการใช้งานดีมานด์ที่เป็นคอนเว็กซ์คอมบิเนชั่นของดีมานด์ที่นำมาสร้างชุดเส้นทางส่งข้อมูลเราจะสามารถรับประกันค่า congestion ได้นั่นเอง ในทางกลับกันถ้าเรามีข้อมูลของดีมานด์ต่างๆที่เกิดขึ้นในระบบ แล้วนำไปสร้างชุดของดีมานด์ที่มีดีมานด์ที่เก็บได้จริงในระบบเป็นคอนเว็กซ์คอมบิเนชั่นของชุดดีมานด์นั้นๆ และเมื่อเรานำชุดของดีมานด์ที่สร้างขึ้นมาได้ผ่านการคำนวณของโปรแกรมเชิงเส้น เราก็จะได้เส้นทางการส่งข้อมูลแบบออบลิเวียสบนเครือข่ายมัลติคาสต์ที่มีสามารถรับประกันค่า congestion ได้ สำหรับค่าอัตราส่วน congestion

ที่ได้แสดงให้เราเห็นได้ว่าโปรแกรมเชิงเส้นสามารถคำนวณเส้นทางแบบออบลิเวียสซึ่งมีค่า congestion สูงกว่าค่า congestion ที่ดีที่สุดของแต่ละดีมานด์เล็กน้อย

เอกสารและสิ่งอ้างอิง

R. Aggarwal, D. Papadimitriou, and S. Yasukawa. 2004. Extension to rsvp-te for point to multipoint te lsps. Available Source http://tools.ietf.org/html/draft-raggarwa-mpls-rsvp-te-p2mp-00, July 2004.

D. Applegate, and E. Cohen. 2003. Making intra-domain routing robust to changing and uncertain traffic demands; understanding fundamental tradeoffs, pp 313-324. In Proceeding of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications. New York, NY, USA.

B. Awerbuch, and Y. Azar. 1995. Competitive multicast routing. Journal of Wireless Networks 1(1):107-114.

Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke.2003. Optimal oblivious routing in polynomial time. In Proceeding of the 35th annual ACM symposium on Theory of computing. New York, NY, USA.

A.-L. Barabasi, R. Robert, and H. Jeong. 2000. Scale-free characteristics of random networks: the topology of the world-wide web. Physica A: Statistical Mechanics and its Applications 2000;281(1–4):69–77. [Online]. Available: http://dx.doi.org/10. 1016/S0378-4371(00)00018-2.

R. Bhatia, M. S. Kodialam, T.V. Lakshman, and S. Sengupta. 2005. Capacity allocation and routing of locally, pp 2395-2404. In Proceeding of 24th of IEEE INFOCOM.

_____, _____, _____, and_____. 2008. Bandwidth guaranteed routing with fast restoration against link and node failures. IEEE/ACM Transactions on Networking 16(6):1321-1330.

R. Cohen and G.Nakibly. 2010. Maximizing restorable throughput in mpls networks. IEEE/ACM Transactions on Networking 18:568-581.

A. Fei, J. Cui, M. Gerla, and D. Cavendish. 2001. A dual-tree scheme for fault tolerant multicast. IEEE International Conference on Communications 3:690-694.

J. L. Kennington and M. W. Lewis. 2001. The path restoration version of the spare capacity allocation problem with modularity restrictions: Models, algorithms, and an empirical analysis. INFORMS Journal of Computing 13(3):181-190

M. S. Kodialam and T. V. Lakshman. 2002. Dynamic routing of bandwidth guaranteed multicasts with failure backup, pp 258-268. In Proceeding of 10th IEEE International Conference on Network Protocols. 12-15 November 2002. Paris, France.

_____ , _____ and S. Sengupta. 2004. Efficient and Robust Routing of Highly Variable Traffic .In Proceeding of 3rd workshop on Hot Topics in Networks.15-16 November 2004. San Diego, CA, USA.

W. Lau, S. Jha, S. Banerjee. 2005. Efficient bandwidth guaranteed restoration algorithms for multicast connections. In: Boutaba R, Almeroth KC, Puigjaner R, Shen SX, Black JP, editors. NETWORKING. Lecture notes in computer science, vol. 3462, Springer; 2005. p. 1005–17.

G. Li, D. Wang, and R. D. Doverspike. 2006. Efficient distributed mpls p2mp fast reroute, pp 1-11. In Proceeding of 25th IEEE International Conference on Computer Communication. 23-29 April 2006. Barcelona, Catalunya, Spain.

Y. Liu, D. Tipper, and P. Siripongwutikorn. 2005. Approximating optional spare capacity allocation by successive survivable routing. IEEE/ACM Transactions on Networking 13:198-211.

A. Mediana, A. Lakhina, I. Matta, and J. Byers. 2001. Brite: An approach to universal topology generation, pp 346-353. In Proceedings of 9th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. 15-18 Aug 2001.Cincinnati, OH, USA.

H. Räcke. 2002. Minimizing congestion in general networks, pp 43-52. In Proceedings of 43rd Symposium on Foundations of Computer Science. . Washington DC, USA.

_____. 2008. Optimal Hierarchical Decompositions for Congestion Minimization in Networks, pp. 255-264. In Proceeding of the 40th STOC. Co-Winner of Best Paper Award

C.-C. Shyur, T.-C. Lu, and U.-P. Wen. 1999. Applying tabu search to spare capacity planning for network restoration. Computer and Operations Research 26(12):1175-1194.

N. K. Singhal, L. H. Sahasrabuddhe, and B. Mukherjee. 2003. Provisioning of survivable multicast sessions against single link failures in optical wdm mesh networks. Journal of Lightwave Technology 21(11):2587-2594.

ประวัติการศึกษาและการทำงาน

ชื่อ	นายสุวรา สุระประเสริฐ
เกิดวันที่	9 ธันวาคม พ.ศ. 2518
สถานที่เกิด	อำเภอเมือง จังหวัดราชบุรี
ประวัติการศึกษา	วศ.บ. (วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเกษตรศาสตร์,
	วศ.ม. (วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเกษตรศาสตร์
ตำแหน่งปัจจุบัน	นักพัฒนาซอฟแวร์อาวุโส
สถานที่ทำงานปัจจุบัน	บริษัทเคเอสซีคอมเมอร์เชียลอินเทอร์เน็ต