การค้นหากลุ่มของเพียร์บนฐานการประยุกต์ใช้งานระบบล่วงรู้บริบท

Multiple Peer to Peer Group Discovery based on

Context Awareness Application

วศิน เที่ยงคุณากฤต

Wasin Thiengkunakrit

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา

วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Computer Engineering

Prince of Songkla University

2553

ลิขสิทธิ์ของมหาวิทยาลัยสงขลานครินทร์

ชื่อวิทยานิพนธ์ การค้นหากลุ่มของเพียร์บนฐานการประยุกต์ใช้งานระบบล่วงรู้บริบท
ผู้เขียน นายวศิน เที่ยงคุณากฤต
สาขาวิชา วิศวกรรมคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

.....
(รองศาสตราจารย์ ดร.สินชัย กมลภิวงศ์)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม

.....
(รองศาสตราจารย์ทศพร กมลภิวงศ์)

คณะกรรมการสอบ

.....ประธานกรรมการ
(ดร.เฉลิมพล ชาญศรีภิญโญ)

.....กรรมการ
(รองศาสตราจารย์ ดร.สินชัย กมลภิวงศ์)

.....กรรมการ
(รองศาสตราจารย์ทศพร กมลภิวงศ์)

.....กรรมการ
(ดร.วิรัช ศรเลิศล้ำวาณิช)

.....กรรมการ
(รองศาสตราจารย์ ดร.มนตรี กาญจนะเดชะ)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

.....
(รองศาสตราจารย์ ดร.เกริกชัย ทองหนู)
คณบดีบัณฑิตวิทยาลัย

ชื่อวิทยานิพนธ์ การค้นหากลุ่มของเพียร์บนฐานการประยุกต์ใช้งานระบบล่วงรู้บริบท
ผู้เขียน นายวศิน เที่ยงคุณากฤต
สาขาวิชา วิศวกรรมคอมพิวเตอร์
ปีการศึกษา 2552

บทคัดย่อ

ระบบล่วงรู้บริบทที่ถูกออกมาในปัจจุบัน อาศัยการออกแบบโดยใช้สถาปัตยกรรมแบบเครื่องแม่ข่าย เครื่องลูกข่าย (client-server) เป็นหลัก ทำให้ระบบล่วงรู้บริบทที่ถูกออกแบบมานั้นประสพปัญหาต่าง ๆ เช่น ปัญหาคอขวด, และปัญหาระบบล้มเหลวเนื่องจากเครื่องแม่ข่ายเพียงจุดเดียว เป็นต้น ซึ่งสถาปัตยกรรมดังกล่าวอาจมีผลกระทบอย่างมาก เมื่อนำมาใช้งานร่วมกันกับระบบล่วงรู้บริบท เนื่องจากโดยทั่วไประบบล่วงรู้บริบทจำเป็นจะต้องมีการแก้ไขข้อมูลอยู่บ่อย ๆ ดังนั้นเครื่องแม่ข่ายจึงต้องรับภาระในส่วนนี้สูงมาก ดังนั้นวิทยานิพนธ์ฉบับนี้จึงได้มีความคิดในการนำระบบเครือข่ายแบบเพียร์ทูเพียร์ มาประยุกต์ใช้ร่วมกันกับระบบล่วงรู้บริบทเพื่อลดปัญหาข้างต้นเหล่านั้น อย่างไรก็ตามการนำระบบเพียร์ทูเพียร์มาใช้ร่วมกันกับระบบล่วงรู้บริบทกลับทำให้เกิดปัญหาอย่างอื่นตามมา เช่น ความเร็วในการค้นหาจะต่ำมากเมื่อจำนวนอุปกรณ์ในระบบมีจำนวนเพิ่มมมากขึ้น, และปัญหาการกระจายออนโทโลยีภายในระบบ เป็นต้น ดังนั้นวิทยานิพนธ์ฉบับนี้ ได้นำเสนอกระบวนการใหม่ในการรวมระบบเครือข่ายเพียร์ทูเพียร์เข้ากับระบบล่วงรู้บริบทโดยใช้ชื่อ MF-P2P (หรือ Multiple Finger table Peer to Peer) ซึ่งระบบดังกล่าวใช้หลักการของระบบเพียร์ทูเพียร์รูปแบบคอร์ด ดังนั้นระบบจึงรองรับการทำงานของโหนดที่ไม่มีการเข้าออกระบบบ่อยนัก โดยระบบ MF-P2P เป็นการแบ่งกลุ่มของโหนดที่อยู่ในระบบออกเป็นกลุ่มของคอร์ดย่อย ๆ ตามบริบทของโหนดที่เข้าร่วมอยู่ และระบบดังกล่าวสามารถแก้ปัญหาเรื่องเวลาในการค้นหาได้เป็นอย่างดี เนื่องจากระบบดังกล่าวถูกออกแบบเพื่อให้ทำงานอยู่บนระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างชนิดคอร์ด ซึ่งเป็นกระบวนการที่ถูกพิสูจน์มาแล้วว่าสามารถทำการค้นหาข้อมูลปลายทางในระบบเพียร์ทูเพียร์ได้อย่างรวดเร็ว

คำสำคัญ: ระบบล่วงรู้บริบท ระบบครือข่ายเพียร์ทูเพียร์ คอร์ด

Thesis Title	Multiple Peer to Peer Group Discovery based on Context Awareness Application
Author	Mr. Wasin Thiengkunakrit
Major Program	Computer Engineering
Academic Year	2009

Abstract

Most of context aware systems are based on client-server model which has some limitations, e.g. a bottle neck problem, single point of failure. The most disadvantage of client-server model is when all nodes need to update their context information frequently and quickly; it will cause massive information flooding in a system. By applying Peer to Peer model to context aware system, this problem can be reduced. However, it will create other problems, e.g. context searching will take more time when a number of hops is large, ontology distribution issues, problems of storing ontology and reasoning process. In this thesis, the model called MF-P2P (Multiple Finger table Peer to Peer) which can reduce a context searching time in P2P networks is proposed. The design is based on Chord mechanism which is suitable for fixed or less mobile nodes model. The proposed model divides nodes into logical Chord groups according to their contexts. By distributing processes to peers, the thesis has shown that the system can perform a fast searching on high complexity of ontology and context.

Keywords: Context awareness system, Peer-to-Peer, Chord

กิตติกรรมประกาศ

ขอขอบพระคุณรองศาสตราจารย์ ดร.สินชัย กมลภิวงศ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก รองศาสตราจารย์ทศพร กมลภิวงศ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่ได้ให้คำปรึกษา ชี้แนะแนวทาง และให้ความรู้ในด้านต่าง ๆ รวมถึงการให้การสนับสนุนในเรื่องอุปกรณ์ในการทำวิจัย ตลอดจนช่วยตรวจและแก้ไขวิทยานิพนธ์ให้เป็นไปอย่างสมบูรณ์

ขอขอบพระคุณ ดร.วิรัช ศรเลิศล้ำวานิช ประธานกรรมการสอบวิทยานิพนธ์ และดร.เฉลิมพล ชาญศรีภิญโญกรรมการสอบวิทยานิพนธ์ ที่ได้ให้คำแนะนำในการปรับปรุงวิทยานิพนธ์ให้สมบูรณ์ยิ่งขึ้น

ขอขอบพระคุณ อาจารย์สุธน แซ่ว่อง ที่ได้ให้คำปรึกษา ชี้แนะแนวทาง และให้ความความรู้ รวมถึงข้อเสนอแนะต่าง ๆ ในการทำวิจัย ตลอดจนช่วยตรวจและแก้ไขวิทยานิพนธ์ให้เป็นไปอย่างสมบูรณ์

ขอขอบพระคุณคณาจารย์ และบุคลากรในภาควิชาวิศวกรรมคอมพิวเตอร์ทุกท่านที่ให้คำปรึกษาและความช่วยเหลือในระหว่างการทำวิทยานิพนธ์

ขอขอบพระคุณคณะวิศวกรรมศาสตร์ที่ได้สนับสนุนทุนการศึกษาในระดับปริญญาโทตลอดระยะเวลา 2 ปีการศึกษา

ขอขอบคุณเพื่อนนักศึกษาปริญญาโทสาขาวิชาวิศวกรรมคอมพิวเตอร์ทุกท่านที่ได้ให้คำแนะนำและเป็นกำลังใจมาโดยตลอด

วศิน เที่ยงคุณากฤต

สารบัญ

กิตติกรรมประกาศ 5
สารบัญ 6
สารบัญรูปภาพ 10
สารบัญตาราง 13
สัญลักษณ์คำย่อและตัวย่อ 14
บทที่ 1 บทนำ 1
1.1 ความสำคัญและที่มาของวิทยานิพนธ์ 1
1.2 วัตถุประสงค์ 2
1.3 ขอบเขตของการวิจัย 3
1.4 ขั้นตอนและวิธีการดำเนินการวิจัย 3
1.5 ประโยชน์ที่คาดว่าจะได้รับ 4
1.6 ภาพรวมของระบบ 5
บทที่ 2 ทฤษฎีและหลักการ 6
2.1 ระบบล่วงรู้บริบท (Context awareness system) 6
2.1.1 รายละเอียดส่วนประกอบของการทำระบบล่วงรู้บริบท 7
2.1.2 เทคโนโลยีการพัฒนาระบบล่วงรู้บริบทในปัจจุบัน 12
2.2 ระบบเครือข่ายเพียร์ทูเพียร์ (Peer-to-Peer network) 12
2.2.1 ประเภทของระบบเครือข่ายเพียร์ทูเพียร์โดยทั่วไป 13
บทที่ 3 การทบทวนวรรณกรรม 25
3.1 การทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด 25
3.2 การเก็บข้อมูลออนโทโลยีบนระบบเครือข่ายเพียร์ทูเพียร์ 29
3.3 การประยุกต์ใช้ระบบล่วงรู้บริบทกับระบบเพียร์ทูเพียร์ 31
3.3.1 การประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง 32

3.3.2 การประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง 34
บทที่ 4 การออกแบบระบบล่วงรู้บริบทด้วยเครือข่ายเพียร์ทูเพียร์แบบคอร์ด 39
4.1 แนวคิดที่ใช้ในการออกแบบ 39
4.2 เทคโนโลยีที่ถูกนำมาใช้ 41
4.3 สถาปัตยกรรมของระบบ MF-P2P 42
4.3.2 การสร้างการเชื่อมต่อในระดับชั้น Context-aware Chord 48
4.3.3 การค้นหาข้อมูลในระบบ MF-P2P 49
4.4 สรุป 60
4.4.1 สรุปการเข้าร่วมกลุ่ม MF-P2P 61
4.4.2 สรุปการค้นหาของ MF-P2P 62
บทที่ 5 การทดลองและการวิเคราะห์ผล 66
5.1 การเปรียบเทียบความเร็วในการค้นหา 66
5.1.1 การวิเคราะห์เชิงคณิตศาสตร์ของระบบ MF-P2P 66
5.1.2 การวิเคราะห์เชิงคณิตศาสตร์ของระบบ ML-Chord 71
5.1.3 ผลการวิเคราะห์เมื่อกำหนดจำนวนเพียร์ในระบบคงที่ 72
5.1.4 ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทให้คงที่ 75
5.1.5 สรุปผล 77
5.2 การเปรียบเทียบความหนาแน่นของข้อมูล 77
5.2.1 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P 77
5.2.2 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord 79
5.2.3 ผลการวิเคราะห์เมื่อกำหนดจำนวนเพียร์ในระบบคงที่ 80
5.2.4 ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทให้คงที่ 83
5.2.5 สรุปผล 85
5.3 การเปรียบเทียบความเร็วในการค้นหา เมื่อมีการค้นหาแบบลูกโซ่ 86
5.3.1 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P 86
5.3.2 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord 87

5.3.3 ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทและจำนวนโหนดคงที่ 88
5.3.4 สรุปผลการทดลอง 89
5.4 การเปรียบเทียบความหนาแน่นในการค้นหา เมื่อมีการค้นหาแบบลูกโซ่ 89
5.4.1 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P 89
5.4.2 การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord 90
5.4.3 ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทและจำนวนโหนดคงที่ 91
5.4.4 สรุปผล 92
5.5 การเปรียบเทียบหน่วยความจำที่ต้องใช้ในระบบ 92
5.5.1 การวิเคราะห์ทางคณิตศาสตร์สำหรับหน่วยความจำที่ต้องใช้ในระบบ 92
5.5.2 เปรียบเทียบหน่วยความจำที่ใช้ในระบบ 95
5.6 สรุปผลการทดลอง 99
บทที่ 6 สรุปผลการวิจัย และข้อเสนอแนะ 102
6.1 สรุปผลการวิจัย 102
6.1.1 การออกแบบระบบล่วงรู้บริบทผ่านทางระบบเครือข่ายเพียร์ทูเพียร์ (MF-P2P) 102
6.1.2 การวิเคราะห์และเปรียบเทียบประสิทธิภาพการทำงานของระบบ MF-P2P 104
6.2 ข้อเสนอแนะ 105
6.2.1 ความสอดคล้องกัน และความน่าเชื่อถือของข้อมูลออนโทโลยี 105
6.2.2 เพิ่มความสามารถในการค้นหาข้อมูลบริบท 105
6.2.3 ทดสอบการใช้งานในสถานการณ์จริง 106
เอกสารอ้างอิง 107
ภาคผนวก ก การออกแบบสมการทางคณิตศาสตร์สำหรับความเร็วในการค้นหาของระบบ ML-Chord 106
ภาคผนวก ข การออกแบบสมการทางคณิตศาสตร์สำหรับความเร็วในการค้นหาระบบ ML-P2P 112
ภาคผนวก ค การออกแบบสมการทางคณิตศาสตร์สำหรับความหนาแน่นของระบบ ML-Chord และ MF-P2P 116

ภาคผนวก ง การออกแบบสมการทางคณิตศาสตร์สำหรับหน่วยความจำของระบบ ML-Chord และ MF-P2P.....119
ประวัติผู้เขียน.....130

สารบัญรูปภาพ

รูปที่ 1-1 ภาพรวมของระบบที่นำเสนอในวิทยานิพนธ์ฉบับนี้ 5
รูปที่ 2-1 ตัวอย่างของแผนภาพออนโทโลยีแบบ Schematic ontology [4] 8
รูปที่ 2-2 ตัวอย่างของภาษา SPARQL 11
รูปที่ 2-3 การเชื่อมต่อแบบ Hybrid P2P 13
รูปที่ 2-4 การเชื่อมต่อแบบ Super P2P 14
รูปที่ 2-5 การเชื่อมต่อแบบ Pure P2P 15
รูปที่ 3-1 การเชื่อมต่อของคอร์ด 26
รูปที่ 3-2 ตัวอย่างการค้นหาข้อมูลในคอร์ดโดยใช้ DHT 27
รูปที่ 3-3 วิธีการกระจายเก็บข้อมูลในกลุ่มเพียร์ทูเพียร์ชนิดคอร์ด 30
รูปที่ 3-4 ระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง 32
รูปที่ 3-5 แผนภาพการทำงานของระบบล่วงรู้บริบทบนเพียร์ทูเพียร์แบบไร้โครงสร้าง 33
รูปที่ 3-6 การทำงานของระบบ ML-Chord 34
รูปที่ 3-7 การทำงานของระบบ Hierarchical P2P 35
รูปที่ 3-8 การทำงานของระบบ GloServ 36
รูปที่ 4-1 ภาพรวมของระบบ MF-P2P 42
รูปที่ 4-2 Finger table ตามปกติของคอร์ดสำหรับโหนดหมายเลข 0 45
รูปที่ 4-3 Finger table ที่ใช้งานใน MF-P2P สำหรับหรับโหนดหมายเลข 0 46
รูปที่ 4-4 ตัวอย่างของ Contextual table 48
รูปที่ 4-5 ตัวอย่างของภาษา SPARQL 50
รูปที่ 4-6 ส่วนประกอบของคำสั่งสืบค้น (Query) 51
รูปที่ 4-7 ตัวอย่างของการเลือก Finger table 52
รูปที่ 4-8 การค้นหาโดยที่ทุกโหนดมีความสำคัญเท่าเทียมกัน 1 54
รูปที่ 4-9 การค้นหาโดยที่ทุกโหนดมีความสำคัญเท่าเทียมกัน 2 55
รูปที่ 4-10 การค้นหาโดยมีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท 56
รูปที่ 4-11 การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง 57
รูปที่ 4-12 การค้นหาโดยกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท 59
รูปที่ 4-13 แผนภาพสรุปการเข้าร่วมกลุ่ม MF-P2P 61

รูปที่ 4-14 แผนภาพแสดงการค้นหาเมื่อโหนดที่ต้องการค้นหาอยู่ในบริบทเดียวกันกับโหนดปลายทาง..... 63
รูปที่ 4-15 แผนภาพแสดงการค้นหาเมื่อโหนดที่ต้องการค้นหาอยู่ต่างบริบทกันกับโหนดปลายทาง..... 64
รูปที่ 5-1 ระยะทางที่ใช้ในการค้นหาของ MF-P2P..... 67
รูปที่ 5-2 ระยะทางที่ใช้ในการค้นหาข้ามบริบทเมื่อทุกโหนดทำงานเท่าเทียมกัน 68
รูปที่ 5-3 ระยะทางที่ใช้ในการค้นหาข้ามบริบทเมื่อมีการกำหนดซูเปอร์โหนด..... 69
รูปที่ 5-4 ระยะทางที่ใช้ในการค้นหาของ ML-Chord..... 72
รูปที่ 5-5 กราฟเปรียบเทียบความเร็วในการค้นหาเมื่อจำนวนโหนดในระบบมีค่าคงที่ โดยแกน X แสดงจำนวนบริบทที่เพิ่มขึ้น และแกน Y แสดงให้เห็นจำนวนโหนดที่ใช้ในการสืบค้น..... 74
รูปที่ 5-6 กราฟเปรียบเทียบความเร็วในการค้นหาเมื่อจำนวนบริบทในระบบมีค่าคงที่ โดยแกน X แสดงจำนวนโหนดที่เพิ่มขึ้น และแกน Y เป็นจำนวนโหนดที่ใช้ในการสืบค้น..... 76
รูปที่ 5-7 ภาพแสดงการกระจายตัวของโหนดใน MF-P2P ซึ่งแสดงบริบทแตกต่างกันโดยใช้เครื่องหมายต่าง ๆ 78
รูปที่ 5-8 แผนภาพแสดงให้เห็นถึงสภาพแวดล้อมที่นำมาวิเคราะห์สำหรับ ML-Chord 80
รูปที่ 5-9 กราฟแสดงความหนาแน่นของระบบเมื่อโหนดมีจำนวน 100,000 โหนด โดยแกน X แสดงถึงจำนวนบริบทที่เพิ่มขึ้นและแกน Y แสดงจำนวนข้อมูลที่ไหลผ่านซูเปอร์โหนด..... 82
รูปที่ 5-10 กราฟแสดงความหนาแน่นของระบบเมื่อมีจำนวนบริบทคงที่คือ 5,000 บริบท โดยแกน X แสดงจำนวนโหนดที่เพิ่มขึ้นและแกน Y แสดงจำนวนข้อมูลที่ไหลผ่านซูเปอร์โหนด..... 84
รูปที่ 5-11 กราฟแสดงความเร็วในการค้นหาเมื่อมีการค้นหาแบบลูกโซ่ โดยจำนวนโหนดคงที่คือ 100,000 โหนด และจำนวนบริบทคงที่ 5,000 บริบท แต่เพิ่มจำนวนครั้งของการหาแบบลูกโซ่ขึ้นตามข้อมูลในแกน X ส่วนความเร็วในการหาจะแสดงในแกน Y 88
รูปที่ 5-12 กราฟแสดงความหนาแน่นในระบบเมื่อมีการค้นหาแบบลูกโซ่ โดยจำนวนโหนดคงที่คือ 100,000 โหนด และจำนวนบริบทคงที่ 5,000 บริบท แต่เพิ่มจำนวนครั้งของการหาแบบลูกโซ่ขึ้นตามข้อมูลในแกน X ส่วนความหนาแน่นที่เกิดขึ้นจะแสดงในแกน Y..... 91
รูปที่ 5-13 จำนวนข้อมูลที่ต้องเก็บในแต่ละโหนดที่มีการค้นหาแบบ ML-Chord..... 93
รูปที่ 5-14 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนบริบทโดยแกน X หมายถึงจำนวนบริบทที่เพิ่มขึ้นในขณะที่แกน Y แสดงจำนวนคู่อันดับที่ใช้ในการเก็บข้อมูลตาราง DHT 96
รูปที่ 5-15 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนทริปเปิลของออนโทโลยีโดยแกน X แสดงจำนวนทริปเปิลที่เพิ่มขึ้นและแกน Y แสดงจำนวนคู่อันดับที่ใช้ในตาราง DHT 97

รูปที่ 5-16 แสดงการใช้หน่วยความจำเมื่อเพิ่มข้อมูลในตาราง Contextual tableโดยแกน X หมายถึงจำนวนคู่อันดับที่เพิ่มขึ้นในตาราง contextual table ส่วนแกน Y แสดงให้เห็นถึงจำนวนคู่อันดับที่ใช้ในตาราง DHT 97
รูปที่ 5-17 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนซูเปอร์โหนดสำหรับ MF-P2P 4 โดยแกน X แสดงให้เห็นถึงจำนวนซูเปอร์โหนดที่ถูกกำหนดในแต่ละบริบทและแกน Y เป็นจำนวนข้อมูลที่ต้องเก็บในตาราง DHT 98
รูปที่ ก-1 การค้นหาของระบบ ML-Chord..... 105
รูปที่ ก-2 กราฟแสดงความเร็วการค้นหาเมื่อกำหนดให้จำนวนโหนดคงที่ 1024 โหนดแล้วเพิ่มจำนวนบริบทขึ้นเรื่อย ๆ โดยอ้างอิงจาก [45]..... 106
รูปที่ ก-3 กราฟแสดงความเร็วการค้นหาเมื่อกำหนดให้จำนวนบริบทคงที่ 16 บริบท แล้วเพิ่มจำนวนโหนดขึ้นเรื่อย ๆ โดยอ้างอิงจาก [45] 107
รูปที่ ก-4 ความเร็วการค้นหาของ ML-Chord เมื่อกำหนดจำนวนโหนดคงที่ 1024 โหนด โดยแกน X เป็นจำนวนบริบทที่เพิ่มขึ้น และแกน Y เป็นความเร็วในการค้นหา..... 108
รูปที่ ก-5 ความเร็วในการค้นหาของ ML-Chord และ Chord เมื่อกำหนดจำนวนบริบทคงที่ 16 บริบท โดยแกน X เป็นจำนวนโหนดที่เพิ่มขึ้น และแกน Y เป็นความเร็วในการค้นหา..... 109
รูปที่ ข-1 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 1..... 111
รูปที่ ข-2 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 2 และ MF-P2P 4..... 113
รูปที่ ข-3 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 3..... 114
รูปที่ ค-1 แสดงความหนาแน่นที่ซูเปอร์โหนดของระบบ ML-Chord..... 115
รูปที่ ค-2 การกระจายตัวของโหนดใน Global Chord ของระบบ MF-P2P 1 และ MF-P2P 3..... 116
รูปที่ ค-3 ความหนาแน่นที่เกิดขึ้นของ MF-P2P 2..... 116
รูปที่ ค-4 ความหนาแน่นที่เกิดขึ้นของ MF-P2P 4..... 117
รูปที่ ง-1 หน่วยความจำที่ใช้สำหรับ ML-Chord..... 118
รูปที่ ง-2 หน่วยความจำที่ใช้ในระบบ MF-P2P 1..... 119
รูปที่ ง-3 หน่วยความจำที่ใช้ในระบบ MF-P2P 2..... 120
รูปที่ ง-4 หน่วยความจำที่ใช้ในระบบ MF-P2P 3..... 121
รูปที่ ง-5 หน่วยความจำที่ใช้ในระบบ MF-P2P 4..... 122

สารบัญตาราง

ตารางที่ 2-1 ตารางเปรียบเทียบภาษา RDF และ OWL 9
ตารางที่ 2-2 ตารางเปรียบเทียบระบบ Hybrid P2P, Super P2P และ Pure P2P..... 15
ตารางที่ 2-3 ตารางเปรียบเทียบระบบ Pure P2P แบบมีโครงสร้าง 20
ตารางที่ 2-4 ตารางเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนเครือข่ายเพียร์ทูเพียร์ชนิดไร้โครงสร้างและชนิดมีโครงสร้าง 21
ตารางที่ 3-1 ตารางเปรียบเทียบการเก็บข้อมูลแบบคอร์ดธรรมดาและแบบ [40]..... 31
ตารางที่ 3-2 ตารางเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนเครือข่ายเพียร์ทูเพียร์ชนิดมีโครงสร้างทั้งสามชนิด 37
ตารางที่ 5-1 ตารางเปรียบเทียบประสิทธิภาพของระบบต่าง ๆ 100

สัญลักษณ์คำย่อและตัวย่อ

CAN	Content Addressable Network
DHT	Distributed Hash-table
DL	Description Language
ML-Chord	Multi-Layer Chord
MF-P2P	Multiple Fingertable Peer to Peer
OWL	Ontology Web Language
P2P	Peer-to-Peer
RDF	Resource Description Framework
SAP	Sevice Annoucement Protocol
SC	Semantic Cluster
SPARQL	SPARQL Protocol and RDF Query Language

บทนำ

ความสำคัญและที่มาของวิทยานิพนธ์

ระบบล่วงรู้บริบทหรือ Context awareness system เป็นระบบที่ใช้สำหรับการเพิ่มความสามารถในการค้นหาข้อมูลที่ต้องการให้กับอุปกรณ์คอมพิวเตอร์ต่าง ๆ ซึ่งระบบล่วงรู้บริบทยังเป็นเทคโนโลยีที่ใหม่และขาดมาตรฐานที่ชัดเจนในการพัฒนา ทำให้ระบบล่วงรู้บริบทยังไม่เป็นที่แพร่หลายนักในการนำไปใช้งานจริง อย่างไรก็ตามระบบล่วงรู้บริบทกำลังถูกนำไปประยุกต์ใช้ในเวปไซต์ต่าง ๆ ในชื่อของ semantics web ซึ่งระบบล่วงรู้บริบทส่งผลให้การจัดการข้อมูลของหน้าเวปไซต์ต่าง ๆ เป็นไปอย่างมีระบบมากยิ่งขึ้น ส่งผลให้เวปไซต์เหล่านั้นสามารถเข้าถึงข้อมูลได้ในระดับขององค์ความรู้ (knowledge) แทนที่จะเป็นการเข้าถึงข้อมูลในระดับของข้อมูลธรรมดา (plain data) ดังนั้นจะเห็นว่าระบบล่วงรู้บริบทสามารถทำให้อุปกรณ์คอมพิวเตอร์มีความสามารถโต้ตอบกับมนุษย์ได้อย่างชาญฉลาดยิ่งขึ้น อย่างไรก็ตามการพัฒนาระบบล่วงรู้บริบทยังต้องอาศัยการเชื่อมต่อโดยผ่านเครื่องแม่ข่ายกลาง (client-server model) ซึ่งสามารถก่อให้เกิดปัญหาในด้านของความล้มเหลวจากจุดเดียว (single point of failure) ปัญหาคอขวด (Bottle neck) ซึ่งปัญหาเหล่านั้น จะส่งผลกระทบต่อความน่าเชื่อถือของระบบ (Reliability) และการใช้ระบบเครือข่ายแบบเซิร์ฟเวอร์กลางนี้จะยากต่อการขยายขนาดของระบบ (Scalability) เนื่องจากการขยายขนาดของระบบเครือข่ายเพื่อให้รองรับผู้ใช้จำนวนมากนั้นจำเป็นต้องอาศัย การขยายขนาดจำนวนและประสิทธิภาพของเครื่องแม่ข่ายซึ่งจำเป็นต้องอาศัยทรัพยากรจำนวนมาก

จากความสำคัญของระบบล่วงรู้บริบทข้างต้นส่งผลให้เกิดวิทยานิพนธ์ฉบับนี้ขึ้น ซึ่งวิทยานิพนธ์นี้เป็นการกล่าวถึงการนำเทคโนโลยีเครือข่ายเพียร์ทูเพียร์ (Peer-to-peer network หรือ P2P) เข้ามาใช้เพื่อกำจัดการให้บริการที่เครื่องแม่ข่ายออกไป และกระจายงานที่มีอยู่ให้กับเครื่องลูกข่ายในการช่วยกันประมวลผล ทำให้ระบบเพียร์ทูเพียร์สามารถนำมาแก้ไขข้อด้อยของการมีเครื่องแม่ข่ายกลาง ปรับปรุงให้ระบบสามารถขยายขนาดได้โดยง่าย ซึ่งความสามารถเหล่านี้เกิดขึ้นเนื่องจากระบบ P2P ในที่นี้เป็นระบบที่อาศัยการเชื่อมต่อของอุปกรณ์ต่าง ๆ เข้าด้วยกันโดยตรง ทำให้มีการเพิ่มช่องทางการเชื่อมต่อระหว่างเพียร์จึงสามารถลดปัญหา single point of failure และ bottle neck ลงได้ พร้อมกันนั้นในการขยายขนาดของระบบยังสามารถทำได้โดยง่าย นั่นคืออุปกรณ์อื่น ๆ สามารถเข้ามาร่วมการเชื่อมต่อได้โดยการเชื่อมต่อเข้ามายังเพียร์ใด ๆ ในระบบ

ดังที่ได้กล่าวจุดเด่นของระบบเพียร์ทูเพียร์มาแล้วข้างต้น ซึ่งส่งผลให้ระบบสามารถเพิ่มความน่าเชื่อถือและขยายขนาดได้โดยสะดวก เมื่อนำมาประยุกต์ใช้กับระบบล่วงรู้บริบทเพื่อเพิ่มความน่าเชื่อถือของระบบ และรองรับการขยายจำนวนของอุปกรณ์จำนวนมากได้ ซึ่งวิทยานิพนธ์ฉบับนี้นำเสนอการเชื่อมต่อระหว่างอุปกรณ์ล่วงรู้บริบทต่าง ๆ เข้าด้วยกันผ่านทางเครือข่ายเพียร์ทูเพียร์ เพื่อทำให้อุปกรณ์เหล่านั้นสามารถแลกเปลี่ยนบริบท (Context) กันอย่างชาญฉลาดและสามารถตอบสนองต่อผู้ใช้ได้อย่างเป็นธรรมชาติมากขึ้น โดยวิทยานิพนธ์ฉบับนี้ใช้ชื่อว่า “การค้นหากลุ่มของเพียร์บนฐานการประยุกต์ใช้งานระบบล่วงรู้บริบท” (Multiple Peer to Peer Group Discovery based on Context Awareness Application) ซึ่งระบบที่ได้นำเสนอขึ้นมาในวิทยานิพนธ์ฉบับนี้ เป็นระบบที่ใช้สำหรับค้นหาโหนดปลายทางโดยใช้ข้อมูลบริบทเป็นตัวช่วยในการค้นหา และระบบที่ได้ออกแบบจะทำการค้นหาอยู่บนระบบเครือข่ายเพียร์ทูเพียร์ชนิดคอร์ด (Chord)

วัตถุประสงค์

1. ศึกษาการทำงานร่วมกันระหว่างระบบล่วงรู้บริบทและเครือข่ายเพียร์ทูเพียร์ ว่ามีผู้อื่นได้ศึกษามาก่อนหรือไม่ และสามารถเพิ่มเติมหรือแก้ไขการทำงานเดิมให้ดีขึ้นได้อย่างไร
2. ออกแบบและปรับปรุงระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด ให้รองรับการทำงานของระบบค้นหาแบบล่วงรู้บริบท ทำให้เพิ่มความน่าเชื่อถือ และง่ายต่อการขยายขนาดของระบบ
3. ระบบที่ได้ออกแบบมาจำเป็นจะต้องคำนึงถึงความเร็วในการค้นหา และความยืดหยุ่นของระบบ เพื่อให้ที่อุปกรณ์อาจมีความสามารถในการประมวลผลต่ำซึ่งไม่สนใจเวลาที่ใช้ในการค้นหา หรืออุปกรณ์ที่มีความสามารถการประมวลผลสูงสามารถค้นหาได้อย่างรวดเร็ว
4. ทดสอบและเปรียบเทียบการทำงานของระบบที่ได้ออกแบบไว้ กับระบบอื่นที่ได้ศึกษาไว้ เพื่อเป็นการพิสูจน์ว่าระบบที่ได้ออกแบบไว้มีประสิทธิภาพในการรองรับระบบล่วงรู้บริบทในแง่ของความเร็วในการค้นหา การลดความหนาแน่นของข้อมูล (traffic load) และการนำไปประยุกต์ใช้งานจริงกับเทคโนโลยีระบบล่วงรู้บริบทในปัจจุบัน

ขอบเขตของการวิจัย

1. ระบบที่ได้นำเสนอในวิทยานิพนธ์ฉบับนี้ จะมุ่งเน้นในเรื่องการออกแบบระบบเพื่อให้สามารถค้นหาโหนดปลายทางบนระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างชนิดคอร์ด โดยใช้ข้อมูลบริบทเป็นตัวช่วยในการค้นหา ซึ่งการนำข้อมูลบริบทมาใช้จะเป็นการประยุกต์ระบบการค้นหาแบบล่วงรู้บริบท ซึ่งใช้งานบนระบบเครือข่ายแบบแม่ข่ายลูกข่ายมาทำงานอยู่บนระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างชนิดคอร์ด
2. การออกแบบระบบเพียร์ทูเพียร์จะใช้การเชื่อมต่อในรูปแบบของคอร์ดเท่านั้น เนื่องจากเป็นรูปแบบการเชื่อมต่อที่ถูกใช้งานอย่างแพร่หลาย พร้อมทั้งสามารถค้นหาได้อย่างรวดเร็ว
3. การออกแบบจะเป็นการนำเสนอการเชื่อมต่อของระบบเครือข่าย และการปรับปรุงการเชื่อมต่อเพียร์ทูเพียร์แบบคอร์ด เพื่อให้สามารถทำงานร่วมกับระบบล่วงรู้บริบทได้
4. วิทยานิพนธ์ฉบับนี้จะไม่คำนึงถึงรูปแบบที่ดีสำหรับการออกแบบออนโทโลยี (Ontology) ซึ่งออนโทโลยีที่นำมาใช้ในการทดลอง จะเป็นออนโทโลยีตัวอย่างเพื่อใช้สำหรับการทดลองเท่านั้น
5. การเชื่อมต่อเพียร์ทูเพียร์ที่เกิดขึ้น จะไม่คำนึงถึงความปลอดภัยต่าง ๆ ที่จะเกิดขึ้นภายในเครือข่าย และถือว่าข้อมูลที่ทุกเพียร์มีอยู่นั้นเป็นข้อมูลที่ถูกต้องทั้งหมด
6. ระบบดังกล่าวถูกออกแบบขึ้นมาเพื่อใช้สำหรับอุปกรณ์ที่อยู่กับที่เท่านั้น และไม่ได้คำนึงถึงกรณีที่อุปกรณ์เคลื่อนที่เข้าออกในระบบบ่อย ๆ ดังนั้นในการวัดผลจะไม่คำนึงถึงทรัพยากรณ์ที่ถูกใช้งานเพิ่มเติม (Overhead) อันเนื่องมาจากการเข้าออกของอุปกรณ์ในระบบ

ขั้นตอนและวิธีการดำเนินการวิจัย

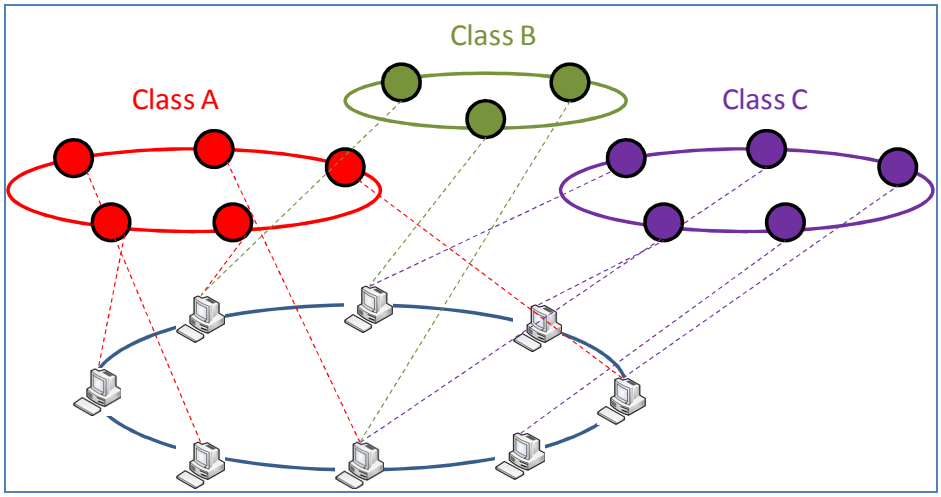
1. ศึกษาระบบล่วงรู้บริบทที่ใช้งานในปัจจุบัน และสรุปผลเทคโนโลยี พร้อมทั้งมาตรฐานที่ใช้สำหรับการสร้างระบบล่วงรู้บริบท
2. ศึกษาการทำงานของระบบเพียร์ทูเพียร์ แบ่งกลุ่มรูปแบบการเชื่อมต่อของระบบเครือข่ายเพียร์ทูเพียร์ออกเป็นกลุ่มย่อย ๆ และศึกษาถึงข้อดีข้อเสียของการใช้งานระบบเพียร์ทูเพียร์ในแต่ละรูปแบบ แล้วจึงสรุปผลการเลือกเครือข่ายเพียร์ทูเพียร์ที่เหมาะสมสำหรับการนำมาประยุกต์ใช้งานกับระบบล่วงรู้บริบท
3. ศึกษาถึงระบบอื่น ๆ ที่ได้เคยถูกออกแบบไว้สำหรับการใช้งานระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์

4. ออกแบบระบบล่วงรู้บริบทให้ทำงานบนเครือข่ายเพียร์ทูเพียร์ โดยคำนึงถึงการนำไปประยุกต์ใช้กับระบบล่วงรู้บริบทที่ถูกใช้งานในปัจจุบัน
5. ใช้วิธีทางคณิตศาสตร์คำนวณประสิทธิภาพของระบบที่ออกแบบขึ้นมาใหม่ ในแง่ของความเร็วในการค้นหา ความหนาแน่นของข้อมูล และการใช้งานบนระบบล่วงรู้บริบทปัจจุบัน ในสถานการณ์ต่าง ๆ
6. นำผลที่ได้จากการวิเคราะห์ทางคณิตศาสตร์มาเปรียบเทียบเพื่อหาข้อดีข้อเสียกับระบบล่วงรู้บริบทอื่น ๆ
7. รวบรวมผลการทดลองและจัดทำวิทยานิพนธ์

ประโยชน์ที่คาดว่าจะได้รับ

1. เรียนรู้กระบวนการสร้างระบบล่วงรู้บริบทว่ามีส่วนประกอบใดบ้างและแต่ละส่วนทำงานเกี่ยวข้องกันอย่างไร และยังได้เรียนรู้กระบวนการนำระบบล่วงรู้บริบทไปใช้งานจากโปรแกรมตัวอย่าง
2. เรียนรู้การทำงานของระบบเครือข่ายเพียร์ทูเพียร์ แยกแยะชนิดของเครือข่ายเพียร์ทูเพียร์ พร้อมทั้งเรียนรู้สถาปัตยกรรมของแต่ละแบบ และเรียนรู้ถึงข้อดีข้อด้อยของการเชื่อมต่อชนิดเพียร์ทูเพียร์
3. เรียนรู้ถึงวิธีการนำระบบล่วงรู้บริบทมาทำงานบนเครือข่ายเพียร์ทูเพียร์ โดยได้เรียนรู้ถึงกระบวนการทำงานของระบบที่เคยถูกนำเสนอมาแล้วก่อนหน้า และวิเคราะห์ข้อดีข้อเสียของระบบดังกล่าวพร้อมทั้งหาทางแก้ไขให้ดียิ่งขึ้น
4. เรียนรู้การออกแบบและสร้างระบบการค้นหาแบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์เพื่อให้ระบบที่ได้ออกแบบ เหมาะสมกับเทคโนโลยีการทำระบบล่วงรู้บริบทในปัจจุบัน และรองรับการเพิ่มขนาดของผู้ใช้เมื่อมีจำนวนมากขึ้น
5. เรียนรู้วิธีการทดสอบประสิทธิภาพของโปรแกรม โดยการใช้แบบจำลองทางคณิตศาสตร์เข้ามาช่วยวิเคราะห์ เพื่อเปรียบเทียบประสิทธิภาพการทำงานของระบบที่ได้ออกแบบมากับระบบล่วงรู้บริบทที่เคยถูกนำเสนอมาแล้ว
6. นำระบบไปประยุกต์ใช้งานเพื่อสร้างกลุ่มของอุปกรณ์คอมพิวเตอร์ที่มีความฉลาดเพื่ออำนวยความสะดวกกับผู้ใช้งาน

ภาพรวมของระบบ



รูปที่ 0-1 ภาพรวมของระบบที่นำเสนอในวิทยานิพนธ์ฉบับนี้

จากรูปที่ 0-1 เป็นการแสดงให้เห็นถึงภาพรวมของระบบ ซึ่งกลุ่มของอุปกรณ์ที่สร้างขึ้นมา จะรวมกลุ่มกันอยู่ที่บริเวณชั้นด้านล่างที่แสดงให้เห็นด้วยรูปคอมพิวเตอร์เล็ก ๆ เชื่อมต่อกัน โดยในระดับชั้นดังกล่าวนั้นจะถูกเรียกว่าชั้นคอร์ดพื้นฐาน (Based Chord) ซึ่งมีหน้าที่ให้อุปกรณ์ทั้งหมดเข้ามาสร้างเส้นทางเชื่อมต่อกันจริง ๆ ตามมาตรฐานเพียร์ทูเพียร์แบบคอร์ด และบริเวณชั้นด้านบนที่แสดงให้เห็นด้วยรูปวงกลมเล็ก ๆ เรียกว่าชั้นคอร์ดบริบท (Context Chord) ซึ่งเป็นชั้นที่มีการเชื่อมต่อในรูปแบบเพียร์ทูเพียร์ของคอร์ดในรูปแบบของนามธรรม (Logical) เนื่องจากชั้นคอร์ดบริบทเกิดจากการที่อุปกรณ์ในชั้นคอร์ดพื้นฐานมีการเก็บตาราง finger table เพิ่มเติมขึ้น ทำให้การเชื่อมต่อในรูปแบบที่นำเสนอไปแล้ว สามารถสร้างคอร์ดบริบทได้เป็นจำนวนมากโดยที่แต่ละอุปกรณ์ยังคงรู้จักกัน และส่งข้อมูลถึงกันได้ในชั้นของคอร์ดพื้นฐาน ส่งผลให้เกิดความยืดหยุ่นในการสร้างระบบการล่วงรู้บริบท เพื่อทำให้ระบบสามารถเข้าถึงออนโทโลยีได้ตลอดเวลาที่ต้องการและเป็นการเพิ่มความรวดเร็วในการค้นหาโดยการส่งข้อมูลระหว่างคอร์ดบริบทกันได้โดยตรงโดยผ่านทางคอร์ดพื้นฐานอีกด้วย

ทฤษฎีและหลักการ

ในบทนี้จะเป็นการกล่าวถึงทฤษฎีและหลักการพื้นฐานที่เกี่ยวข้องกับการทำวิทยานิพนธ์ โดยเนื้อหาที่กล่าวถึงจะประกอบไปด้วยหลักการทำงานของระบบล่วงรู้บริบท (Context awareness system) โดยจะเป็นการกล่าวถึงรายละเอียดของส่วนประกอบต่าง ๆ ที่ใช้ในระบบล่วงรู้บริบท แนวการสร้างระบบล่วงรู้บริบท เทคโนโลยีที่ใช้ในระบบล่วงรู้บริบทปัจจุบัน จากนั้นจะเป็นการกล่าวถึงระบบเพียร์ทูเพียร์ (Peer-to-Peer) ซึ่งจะกล่าวถึงประเภท รวมถึงเทคนิคการทำงานของระบบเพียร์ทูเพียร์รูปแบบต่าง ๆ

ระบบล่วงรู้บริบท (Context awareness system)

ระบบล่วงรู้บริบทหมายถึง ระบบที่อุปกรณ์ในระบบมีความสามารถในการรับรู้ถึงบริบทของตนเอง แล้วสามารถประพฤติตัวเองให้สอดคล้องกับบริบทเหล่านั้น เพื่อเพิ่มความสามารถในการตอบสนองต่อผู้ใช้อุปกรณ์ให้เป็นธรรมชาติยิ่งขึ้น เช่น ระบบล่วงรู้บริบทในร้านขายจักรยาน ซึ่งระบบดังกล่าวมีความสามารถที่จะแนะนำจักรยานที่เหมาะสมกับผู้ซื้อซึ่งเข้ามาในร้านแต่ละคนได้ โดยระบบล่วงรู้บริบทของร้านจะมีความรู้เกี่ยวกับรถจักรยานแต่ละชนิด และความรู้เกี่ยวกับจักรยานโดยทั่วไปว่าประกอบไปด้วย คันบังคับ, ที่นั่ง, ล้อสองล้อ, จานและโซ่คล้องล้อ และระบบล่วงรู้บริบทจะสามารถแยกแยะชนิดของรถจักรยานตามสภาพการใช้งานต่าง ๆ ได้ เช่น รถจักรยานเสือภูเขา จะมียางที่ใช้ในล้อทั้งสองล้อต่างจากจักรยานชนิดอื่น เพื่อให้เหมาะสมกับการวิ่งในทางวิบาก, รถจักรยานสำหรับครอบครัว จะมีที่นั่งซึ่งเป็นเบาะแบบพิเศษทำให้นั่งได้อย่างสบาย ซึ่งเมื่อลูกค้าเข้ามาบอกความต้องการของตนเอง เช่น ความต้องการที่จะนำรถไปใช้งาน, หรือความสูงของคนขับ เป็นต้น ระบบก็จะพยายามแนะนำรถจักรยานที่เหมาะสมกับลูกค้า ตามข้อมูลของรถจักรยานที่มีอยู่ในร้าน เป็นต้น

ดังที่ได้ยกตัวอย่างมาแล้วแสดงให้เห็นว่าระบบล่วงรู้บริบท เป็นการทำงานของอุปกรณ์ในระบบโดยคำนึงถึงบริบทต่าง ๆ ที่อยู่รอบระบบนั้น เช่น ความต้องการของผู้ซื้อรถจักรยาน, ลักษณะโดยทั่วไปของรถจักรยาน, และลักษณะเฉพาะของรถจักรยานต่าง ๆ ดังนั้นการทำงานของระบบล่วงรู้บริบท จะเป็นการรับรู้คุณสมบัติของสิ่งต่าง ๆ ที่อยู่ในระบบ (Thing) และ ความสัมพันธ์กันของสิ่งเหล่านั้นที่อยู่ในระบบ (Relationship) ดังนั้นในการพัฒนาระบบล่วงรู้บริบทจึงจำเป็นต้องมีกระบวนการเพื่อจัดการกับข้อมูลบริบทเหล่านั้น ทำให้การพัฒนาระบบล่วงรู้บริบทไม่สามารถใช้รูปแบบการพัฒนาแบบโปรแกรมประยุกต์ทั่วไปได้ เนื่องจากข้อมูลบริบทสามารถเปลี่ยนแปลงได้ตลอดเวลา เช่น ความต้องการของผู้ใช้งาน รวมถึงสิ่งของในระบบและ

ความสัมพันธ์ของสิ่งของเหล่านั้นก็มีโอกาสเปลี่ยนแปลงได้ตลอดเช่นกัน เช่น โครงสร้างของรถจักรยานแบบใหม่ ๆ

รายละเอียดส่วนประกอบของการทำระบบล่วงรู้บริบท

ระบบล่วงรู้บริบทประกอบไปด้วยส่วนสำคัญสองส่วนทำงานร่วมกันอยู่ นั่นคือ ออนโทโลยี (Ontology) [1] [2] [3] และตัวให้เหตุผล (Reasoner) ซึ่งออนโทโลยีนั้นใช้สำหรับการอธิบายความสัมพันธ์ของสิ่งต่าง ๆ ส่วนตัวให้เหตุผลทำหน้าที่ในการประมวลผลและตัดสินใจถึงผลลัพธ์ที่ควรจะได้จากการทำงานแต่ละครั้ง

อย่างไรก็ตามวิทยานิพนธ์ฉบับนี้ไม่ได้มีวัตถุประสงค์เพื่อการออกแบบออนโทโลยี และตัวให้เหตุผลที่ดี แต่จะเน้นประเด็นเกี่ยวกับการสร้างระบบ เพื่อให้สามารถรองรับออนโทโลยีและตัวให้เหตุผลต่าง ๆ ได้ และยังมุ่งเน้นถึงการนำระบบล่วงรู้บริบทมาประยุกต์ใช้ในระบบเครือข่ายเพียร์ทูเพียร์มากกว่าเน้นการออกแบบออนโทโลยีที่ดีอีกด้วย

2.0.0.1 ภาพรวมของระบบล่วงรู้บริบท

ระบบล่วงรู้บริบทประกอบไปด้วยส่วนประกอบสำคัญสองส่วนดังที่กล่าวมาแล้วข้างต้นนั่นคือ ออนโทโลยีและตัวให้เหตุผลซึ่งการทำงานของระบบล่วงรู้บริบทโดยทั่วไปประกอบไปด้วยวิธีหลัก ๆ ดังต่อไปนี้

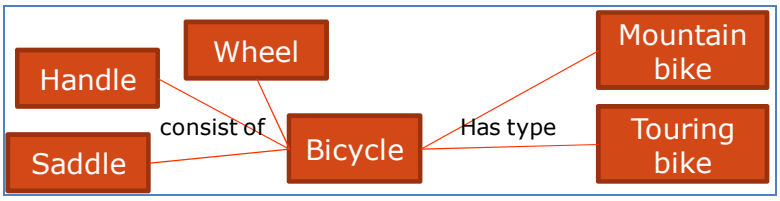
1. การสร้างหรือการดึงเอาออนโทโลยีขึ้นมาเพื่อใช้เตรียมพร้อมต่อการประมวลผลต่อไป ซึ่งกระบวนการนี้ เป็นการให้ผู้ใช้นำออนโทโลยีที่ได้ออกแบบไว้แล้วอ่านขึ้นมาเก็บไว้ภายในหน่วยความจำของเครื่อง
2. นำออนโทโลยีมาผูกความสัมพันธ์กันด้วยตัวให้เหตุผล ทำให้สามารถทำการค้นหาได้อย่างชาญฉลาดยิ่งขึ้น
3. รอรับการค้นหาจากผู้ใช้ โดยเมื่อได้รับข้อมูลที่ต้องการค้นหามาแล้วจึงนำข้อมูลที่ต้องการค้นหา ไปค้นหาภายในออนโทโลยีที่ได้รับการผูกความสัมพันธ์
4. หลังจากนั้นระบบจะตอบกลับมาโดยข้อมูลที่ตอบกลับมาจะเป็นข้อมูลที่ตอบโดยทางตรงและโดยทางอ้อม (ข้อมูลที่ผ่านการให้เหตุผลแล้ว)

สำหรับวิทยานิพนธ์ฉบับนี้ จะไม่เน้นกระบวนการให้เหตุผลหรือการสร้างออนโทโลยี โดยจะสมมติว่าข้อมูลเหล่านั้นมีอยู่พร้อมแล้ว ดังนั้นวิทยานิพนธ์ฉบับนี้จะไม่เน้นถึง

กระบวนการสร้างออนโทโลยีที่ดี อย่างไรก็ตามข้อมูลออนโทโลยีจะถูกกล่าวพอสังเขปเพื่อเป็นพื้นฐานสำหรับการทำความเข้าใจวิทยานิพนธ์ฉบับนี้ ซึ่งจะถูกกล่าวในหัวข้อถัดไป

2.0.0.2 ออนโทโลยี (Ontology)

ออนโทโลยี หมายถึงแผนภูมิที่ใช้สำหรับอธิบายคุณลักษณะของสิ่งของต่าง ๆ และความสัมพันธ์ของสิ่งของเหล่านั้น ซึ่งอาจเขียนอธิบายให้ง่ายขึ้นด้วยการบรรยายออกมาเป็นรูปภาพแสดงแผนภูมิความสัมพันธ์ดังตัวอย่างของรูปที่ 0-1



รูปที่ 0-1 ตัวอย่างของแผนภาพออนโทโลยีแบบ Schematic ontology [4]

สำหรับรูปที่ 0-1 เป็นการแสดงให้เห็นถึงแผนภาพความสัมพันธ์ซึ่งอธิบายสิ่งของต่าง ๆ และความสัมพันธ์ของสิ่งเหล่านั้น ซึ่งจากรูปแสดงให้เห็นตัวอย่างของระบบล่วงรู้บริบทอย่างง่าย ของส่วนประกอบของรถจักรยาน ซึ่งในแผนภาพมีการบอกได้ว่า รถจักรยานโดยทั่วไปประกอบไปด้วยส่วนประกอบสามส่วนนั่นคือ ล้อ (Wheel), คันบังคับ (Handle), และที่นั่ง (Saddle) ซึ่งรถจักรยานสามารถแบ่งประเภทได้เป็นสองประเภทด้วยกันคือ จักรยานเสือภูเขา (Moutain bike), และจักรยานครอบครัว (Touring bike) และในออนโทโลยีดังกล่าวยังสามารถอนุมาน (inference) ได้อีกว่ารถจักรยานไม่ว่าจะเป็นแบบจักรยานเสือภูเขา หรือจักรยานครอบครัว ก็จะต้องประกอบไปด้วย ล้อ, คันบังคับ, และที่นั่งเสมอ

ออนโทโลยีสามารถแบ่งออกได้เป็นสองส่วนใหญ่ ๆ นั่นคือ

1. Schematic ontology ทำหน้าที่สำหรับการแสดงความสัมพันธ์ของระบบที่กำลังใช้งานอยู่ ซึ่งแผนภาพนี้จะเป็นส่วนมโนภาพที่ไม่มีการนำมาใช้งานโดยตรง แต่จะใช้สำหรับการตัดสินใจเมื่อมีการให้เหตุผลเกิดขึ้น ซึ่งตัวอย่างของ schematic ontology นี้ได้แสดงให้เห็นในรูปที่ 0-1 ดังที่ได้อธิบายไว้ใต้ภาพแล้ว นั่นคือเมื่อมอง schematic ontology จะเห็นว่าหากมีการค้นหาไปยังโหนด woman แล้วจึงสามารถอนุมานได้ว่าการค้นหานั้น อาจต้องการเชื่อมโยงไปยัง female, adult, person และ animal ได้เช่นกัน

2. Instance ontology คือส่วนของออนโทโลยีที่มีไว้สำหรับการนำออกมาใช้งานจริง ซึ่งเปรียบเสมือนออปเจ็กของระบบการเขียนโปรแกรมแบบเชิงวัตถุ (Object Oriented Programming) [5] โดยการสร้างออนโทโลยีชนิดนี้ เป็นการสร้างโดยอิงจาก schematic ontology ส่วนตัวอย่างของ instance ontology ที่สอดคล้องกับตัวอย่างนี้ เช่น สมมติให้อุปกรณ์ชนิดหนึ่งแทนด้วยชื่อ Jane โดยกำหนดให้ Jane เป็น instance ontology ของ woman นั่นหมายความว่า Jane เป็นส่วนที่เป็นนามธรรมสำหรับออนโทโลยีในรูปที่ 0-1 และมีคุณสมบัติอื่น ๆ ที่เหมือนกับ woman อื่น ๆ ทุกประการ ได้แก่ female, adult, person, และ animal ซึ่ง instance ontology นี้อาจถูกเรียกอีกชื่อหนึ่งว่า individual

ในการทำงานจริงออนโทโลยีมิได้เขียนอยู่ในรูปของแผนภูมิดังตัวอย่างในรูปที่ 0-1 แต่จะถูกเขียนอยู่ในรูปของภาษาต่าง ๆ ตามที่ต้องการ อย่างไรก็ตามองค์กร W3C [6] ได้สร้างมาตรฐานสำหรับการเขียนออนโทโลยีออกมาเป็นภาษาสองตัวนั่นคือ RDF (Resource Description Framework) [7] [8] [9] และ OWL (Web Ontology Language) [10] [11](ซึ่งปัจจุบันนี้เป็นเวอร์ชั่น 2 โดยใช้ชื่อ OWL2 [12]) ซึ่งภาษาทั้งสองนี้สามารถนำมาใช้อธิบายออนโทโลยีได้เหมือนกันโดยที่มีข้อแตกต่างกันดังตารางที่ 0-1

ตารางที่ 0-1 ตารางเปรียบเทียบภาษา RDF และ OWL

	ภาษา RDF	**ภาษา OWL**
รูปแบบการเขียน (Syntax)	เขียนในรูปของ Extensible Markup Language (XML)	เขียนในรูปของ Extensible Markup Language (XML)
การสร้าง triple [13]	สามารถสร้างเป็น triple ได้	สามารถสร้างเป็น triple ได้ในบางกรณี
ความยืดหยุ่นในการเขียน	ขาดความยืดหยุ่นในการเขียน	สามารถเขียนได้อย่างเป็นธรรมชาติมากกว่า
ความสามารถของภาษา	สามารถอธิบายสิ่งของในออนโทโลยีได้แบบโดยตรงเท่านั้น	สามารถอธิบายโดยอ้อมได้และมีคำสั่งให้เลือกใช้อย่างกว้างขวางกว่า

จากตารางที่ 0-1แสดงให้เห็นถึงความสามารถของภาษาทั้งสอง ซึ่งจากตารางจะเห็นได้อย่างชัดเจนว่าภาษา RDF มีความสามารถน้อยกว่าภาษา OWL เนื่องจากภาษา OWL เป็น

ภาษาที่ถูกพัฒนาขึ้นมาใหม่โดยมี RDF เป็นภาษาพื้นฐาน ดังนั้นการเขียนภาษา RDF แม้ว่าจะทำให้ได้ความสามารถน้อยกว่า แต่ก็สามารถทำให้ได้รับความสะดวกในการเขียนนั่นคือ ความไม่ซับซ้อนในภาษา และสามารถนำไปใช้กับภาษา OWL ได้ทันที

สิ่งสำคัญของการเปรียบเทียบในตารางที่ 0-1นั่นคือ การที่ภาษาทั้งสองสามารถเขียนอยู่ในรูปของ triple ได้ ถึงแม้ว่าภาษา OWL จะไม่สามารถแปลงเป็น triple ได้ทั้งหมดโดยตรงก็ตาม ซึ่งภาษา OWL สามารถแบ่งได้เป็นสามแบบนั่นคือ OWL-Lite, OWL-DL, และ OWL-Full ซึ่งภาษา OWL-Lite สามารถเขียนอยู่ในรูปแบบของ triple ได้ ในขณะที่ OWL-DL และ OWL-Full ไม่สามารถรับประกันได้ว่าระบบจะแสดงผลออกมาในรูปแบบของ triple ได้ เนื่องจากรูปแบบการเขียนของภาษา OWL-Full ไม่เป็นที่ตายตัว โดยในวิทยานิพนธ์ฉบับนี้จะออกแบบโดยมีพื้นฐานอยู่ที่ว่าออนโทโลยีไม่ว่าจะเป็น schematic ontology หรือ instance ontology ก็สามารถนำไปเขียนเป็น triple ได้

ดังนั้น Triple [13] จึงหมายถึงส่วนประกอบย่อยของออนโทโลยีซึ่ง เป็นช่องทางอีกทางหนึ่งที่ใช้ในการแสดงออนโทโลยีออกมานอกเหนือจากการใช้ภาษา RDF และ OWL ซึ่ง triple ประกอบไปด้วยส่วนประกอบสำคัญสามส่วนนั่นคือ subject, predicate, และ object โดยทั้งสามส่วนประกอบกันจะทำให้สามารถอธิบายสิ่งต่าง ๆ ที่รวมกันเป็นในออนโทโลยีได้

สำหรับเครื่องมือที่ใช้สำหรับการสร้างและอ่านออนโทโลยีในวิทยานิพนธ์ฉบับนี้จะใช้โปรแกรม Protégé [14] เนื่องจากเป็นเครื่องมีที่มีความยืดหยุ่นสูงและสามารถนำมาประยุกต์ใช้เพื่อสร้างออนโทโลยีที่ซับซ้อนได้ ยิ่งไปกว่านั้นการทำงานของ Protégé ได้รวมตัวให้เหตุผลเข้าไปด้วย ทำให้เพิ่มความสะดวกในการตรวจสอบความถูกต้องของออนโทโลยีที่ได้ออกแบบมา

2.0.0.3 ตัวให้เหตุผล (Reasoner)

ในการสร้างตัวให้เหตุผลหรือ Reasoner เป็นกระบวนการนำออนโทโลยีมาเพิ่มความสัมพันธ์ให้มากขึ้นทำให้การค้นหาแต่ละครั้งสามารถได้รับข้อมูลที่เป็นข้อมูลทางอ้อมได้ แต่กระบวนการให้เหตุผลจะไม่กล่าวถึงในวิทยานิพนธ์ฉบับนี้ เนื่องจากการทำงานของระบบที่ได้ออกแบบนั้นไม่ได้เน้นที่กระบวนการสร้างออนโทโลยีขึ้นมาใหม่โดยใช้ตัวให้เหตุผล แต่จะเน้นที่การสร้างระบบเพื่อใช้เก็บออนโทโลยีและค้นหาปลายทางให้ถูกต้องมากกว่า

สำหรับตัวให้เหตุผลในปัจจุบันได้ถูกพัฒนาขึ้นมาจากหลายผู้ให้บริการ เช่น Jena [15], Pellet [16], FaCT [17], FaCT++ [18] เป็นต้น ซึ่งแต่ละเครื่องมือให้เหตุผลสามารถทำงานได้ในสถานะที่เหมาะสมต่างกัน ซึ่งการใช้งานตัวให้เหตุผลเหล่านั้นจะไม่ส่งผลต่อวิทยานิพนธ์ฉบับนี้ จึงไม่มีการเปรียบเทียบข้อมูลเหล่านั้นให้เห็น อย่างไรก็ตามการเปรียบเทียบได้มีการกระทำไว้แล้วโดย [19] ซึ่งในวิทยานิพนธ์ฉบับนี้ Pellet ได้ถูกนำมาใช้สำหรับการทดลอง

เนื่องจากเป็นตัวให้เหตุผลที่เขียนด้วยภาษาจาวาที่ง่ายต่อการพัฒนา และยังมีความสามารถเหนือกว่าตัวให้เหตุผลจาวาตัวอื่น เช่น Jena เป็นต้น

2.0.0.4 การค้นหาข้อมูล (Query)

สำหรับการค้นหาข้อมูลในระบบล่วงรู้บริบทปัจจุบันได้มีการนำเสนอระบบการสืบค้นขึ้นมาหลายแบบ แต่ที่ถูกนิยมนำมาใช้นั่นคือ SWRL (Semantic Web Rule Language) [20] และ SPARQL (SPARQL Protocol and RDF Query Language) [21] [22] [23] ซึ่งเครื่องมือ SWRL เป็นเครื่องมือที่ถูกออกแบบมาเพื่อให้ใช้งานกับภาษา OWL เพื่อเพิ่มความสามารถในการค้นหาของระบบล่วงรู้บริบทที่ซับซ้อน

ในวิทยานิพนธ์ฉบับนี้จะเน้นไปที่การนำ SPARQL มาใช้งานในการสร้างสัญญาณเพื่อใช้ส่งไปในระบบเพื่อสืบค้นข้อมูล และตอบกลับมาตามออนโทโลยีที่ได้ถูกออกแบบไว้ เนื่องจากการใช้ภาษา SPARQL มีจุดมุ่งหมายเพื่อการสืบค้นข้อมูลโดยตรงโดยจะสมมติว่าออนโทโลยีที่มีอยู่ มีองค์ประกอบครบถ้วนสมบูรณ์แล้ว ซึ่งหากออนโทโลยีมีความสมบูรณ์พอในตัวเองแล้วการค้นหาด้วยภาษา SPARQL และภาษา SWRL จะให้ผลลัพธ์ที่ใกล้เคียงกัน

ภาษา SPARQL เป็นภาษาการค้นหาที่ประกอบไปด้วยภาษา SQL (Structured Query Language) [24] ร่วมกับการใช้ออนโทโลยีเพื่อสืบค้นข้อมูลที่ต้องการ ซึ่งตัวอย่างของภาษาเป็นดังแสดงในรูปที่ 0-2

```
PREFIX abc: <http://example.com/exampleOntology#>
SELECT ?capital ?country
WHERE {
  ?x abc:cityname ?capital ;
     abc:isCapitalOf ?y .
  ?y abc:countryname ?country ;
     abc:isInContinent abc:Africa .
}
```

รูปที่ 0-2 ตัวอย่างของภาษา SPARQL

จากรูปที่ 0-2 แสดงให้เห็นถึงตัวอย่างของภาษา SPARQL ซึ่งประกอบไปด้วยส่วนที่เขียนเหมือนกับภาษา SQL ธรรมดานั่นคือการเขียน SELECT และ WHERE โดยสิ่งที่อยู่ใน WHERE จะเป็นการใช้ triple เพื่ออธิบายถึงสิ่งที่ต้องการจะค้นหา ซึ่งจากการค้นหาด้านบนผลลัพธ์ที่ได้ออกมาจะเป็นชื่อของประเทศ และเมืองหลวงทั้งหมดที่อยู่ทวีปแอฟริกา

เทคโนโลยีการพัฒนาระบบล่วงรู้บริบทในปัจจุบัน

ในปัจจุบันการพัฒนาระบบล่วงรู้บริบทโดยส่วนมาก จะถูกกระทำโดยใช้เครื่องแม่ข่ายเป็นหลัก โดยเทคโนโลยีที่กำลังถูกนำไปใช้อย่างแพร่หลายคือเทคโนโลยีของการทำ Semantic web [25] [26] หรือเวปเชิงความหมาย นั่นคือการทำให้ระบบเวปไซต์มีความสามารถในการตีความหมายของข้อมูลที่ถูกเก็บเอาไว้ ซึ่งข้อมูลเหล่านั้นจะถูกเชื่อมโยงกันเองภายในเครือข่ายของเวปไซต์ ส่งผลให้เวปไซต์เหล่านั้นมีความชาญฉลาดยิ่งขึ้นและสามารถโต้ตอบกับผู้ใช้ได้อย่างเป็นธรรมชาติมากขึ้นด้วยเช่นกัน

สำหรับตัวอย่างของระบบล่วงรู้บริบทที่ถูกนำมาใช้เป็นเวปไซต์ที่นำมาเป็นตัวอย่างในวิทยานิพนธ์ฉบับนี้คือ AceWiki [27] ซึ่งเป็นการนำระบบล่วงรู้บริบทมาใช้งานร่วมกับระบบสารานุกรม (Wikipedia) โดยจะมีการสร้างออนโทโลยีสำหรับสิ่งของที่สนใจไว้สามชนิดนั่นคือ สถานที่, โปรตีน, และเทคนิคการเขียนเวปเชิงความหมาย หลังจากนั้นระบบจะนำออนโทโลยีที่ได้เก็บไว้มาแสดงผลให้อยู่ในรูปที่มนุษย์สามารถเข้าใจได้โดยง่าย ยิ่งไปกว่านั้นระบบของเวปไซต์ดังกล่าวยังสามารถให้ผู้ใช้แต่ละคนเข้าไปใส่ข้อมูลเพิ่มเติมได้เอง โดยระบบจะมีการตรวจสอบว่าข้อมูลที่เพิ่มเข้าไปมีการขัดแย้งกับข้อมูลก่อนหรือไม่ หลังจากนั้นระบบจะเชื่อมโยงข้อมูลที่เพิ่งใส่เข้าไปใหม่กับข้อมูลของออนโทโลยีที่มีอยู่ แล้วจึงแสดงผลของการวิเคราะห์โดยตัวให้เหตุผลออกมาในภาษาที่เข้าใจได้โดยง่าย และยังสามารถแสดงเส้นทางการวิเคราะห์ของระบบให้เหตุผลเพื่อให้ผู้พัฒนารายอื่นสามารถตรวจสอบวิธีการทำงานของเวปไซต์ดังกล่าวได้อีกด้วย

ระบบเครือข่ายเพียร์ทูเพียร์ (Peer-to-Peer network)

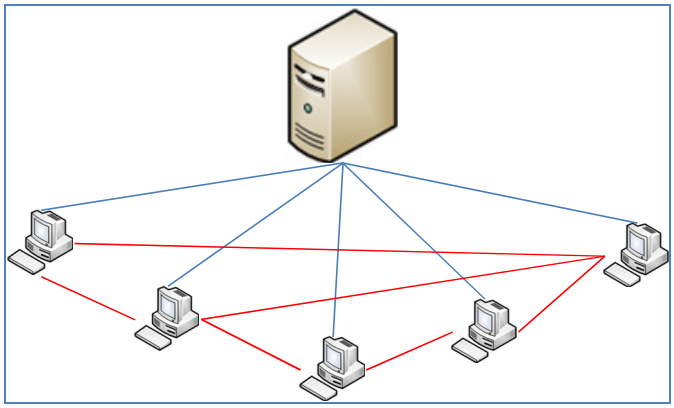
ระบบเครือข่ายเพียร์ทูเพียร์เป็นการเชื่อมต่อโดยที่อุปกรณ์ที่เข้าร่วมการเชื่อมต่อจะถูกเรียกว่าโหนดหรือเพียร์ ซึ่งโหนดเหล่านี้เชื่อมต่อถึงกันโดยตรงโดยที่แต่ละโหนดกระทำตัวเสมือนเป็นทั้งเครื่องแม่ข่ายและเครื่องลูกข่ายในเวลาเดียวกัน

สำหรับในปัจจุบันนี้ระบบเพียร์ทูเพียร์ ได้ถูกนำมาใช้กันอย่างแพร่หลายทั้งในเรื่องของการแบ่งปันข้อมูลในระบบเครือข่ายอินเทอร์เนต (Bittorrent) [28] [29] หรือการถ่ายทอดสัญญาณสื่อประสมภาพและเสียง (IPTV) [30] ซึ่งระบบเหล่านั้นเป็นระบบที่ถูกใช้กันอย่างมาก เนื่องจากระบบเหล่านั้นสามารถเชื่อมต่อได้อย่างรวดเร็ว, มีความน่าเชื่อที่ของระบบสูงเนื่องจากไม่มีเครื่องแม่ข่ายโดยตรงทำให้ระบบไม่ต้องสนใจเกี่ยวกับปัญหาคอขวดและการล้มเหลวจากจุดเดียว, ระบบเพียร์ทูเพียร์สามารถสร้างขึ้นได้อย่างรวดเร็วและขยายตัวได้อย่างรวดเร็ว, และรูปแบบการทำงานของระบบเพียร์ทูเพียร์ส่งผลให้ระบบสามารถได้รับข้อมูลจากโหนดหรือเพียร์รอบข้างได้ตลอดเวลา จึงเป็นการเพิ่มความเร็วในการรับข้อมูลมาเก็บไว้อีกด้วย

ประเภทของระบบเครือข่ายเพียร์ทูเพียร์โดยทั่วไป

ระบบเครือข่ายเพียร์ทูเพียร์โดยทั่วไปนั้นจะถูกแบ่งออกได้เป็นสามประเภทนั่นคือ Hybrid P2P, Super P2P, และ Pure P2P ดังที่จะกล่าวต่อไปนี้

2.1.0.1 Hybrid P2P



รูปที่ 0-3 การเชื่อมต่อแบบ Hybrid P2P

สำหรับการเชื่อมต่อเพียร์ทูเพียร์ประเภทแรกที่นำเสนอในรูปที่ 0-3 เป็นการเชื่อมต่อแบบเพียร์ทูเพียร์แบบ Hybrid P2P หมายถึงการเชื่อมต่อกระทำโดยมีการช่วยเหลือโดยเครื่องแม่ข่ายเพื่อช่วยอำนวยความสะดวกในการค้นหาเพียร์อื่น ๆ ที่มีทรัพยากรที่ต้องการ ซึ่งจากรูปที่ 0-3 แสดงให้เห็นการเชื่อมต่อดังกล่าว ซึ่งแสดงให้เห็นว่าโหนดจะต้องเชื่อมต่อกับเครื่องแม่ข่าย ในขณะเดียวกันโหนดเหล่านั้นสามารถเชื่อมต่อกันเองเพื่อแลกเปลี่ยนข้อมูลอีกด้วย

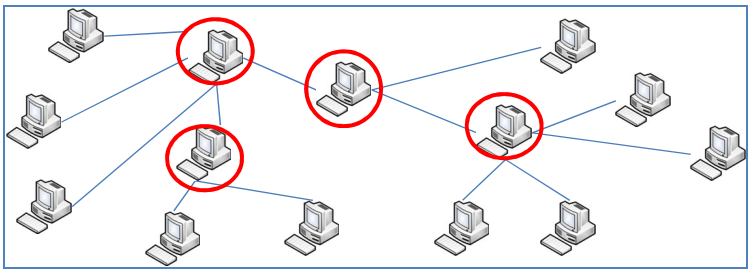
สาเหตุหลักของการมีเครื่องแม่ข่ายกลางเนื่องจากในการค้นหาว่าโหนดไหนมีข้อมูลที่ต้องการเป็นเรื่องยากในระบบเครือข่ายเพียร์ทูเพียร์ เนื่องจากแต่ละโหนดเชื่อมต่อกันอย่างไม่เป็นระเบียบ ดังนั้นเครื่องแม่ข่ายจะมีหน้าที่สำหรับการจัดระบบภายในเครือข่าย โดยเครื่องแม่ข่ายจะทำการเก็บข้อมูลเกี่ยวกับโหนด และข้อมูลที่แต่ละโหนดทำการเก็บไว้อีกด้วย ซึ่งหลังจากระบบทำการค้นหาโหนดปลายทางที่ต้องการเชื่อมต่อจากเครื่องแม่ข่ายได้แล้ว โหนดต้นทางจะทำการเชื่อมต่อไปยังโหนดปลายทางโดยตรงเพื่อร้องขอทรัพยากรที่ต้องการจากโหนดปลายทาง

สำหรับโปรแกรมประยุกต์ตัวอย่างที่ใช้งานระบบในรูปแบบนี้ได้แก่โปรแกรม Bittorrent โดยการทำงานของระบบนี้คือ เมื่อโหนดต้องการค้นหาไฟล์จากระบบเครือข่าย โหนดนั้นจำเป็นจะต้องค้นหาไฟล์ tracker มาเก็บไว้เสียก่อน หลังจากนั้นโหนดจะใช้ข้อมูลในไฟล์เพื่อ

เชื่อมต่อกับ tracker แม่ข่ายเพื่อขอข้อมูลว่าไฟล์ที่ต้องการจะค้นหาอยู่ที่โหนดใดบ้างในระบบเครือข่าย ซึ่งเครื่อง tracker แม่ข่ายจะตอบกลับมาเป็นที่อยู่ของโหนดต่าง ๆ ที่มีไฟล์อยู่ หลังจากนั้นโหนดต้นทางจะเชื่อมต่อไปยังโหนดปลายทางเหล่านั้นโดยตรงเพื่อทำการดาว์นโหลดไฟล์จากโหนดเหล่านั้น

2.1.0.2 Super P2P

การเชื่อมต่อแบบ Super P2P เป็นการเชื่อมต่อโดยที่จะมีการกำหนดความสำคัญหรือความสามารถของโหนดแต่ละตัวในระบบที่ต่างกัน โดยจะมีโหนดบางตัวที่รับหน้าที่มากกว่าโหนดทั่วไปเรียกว่าซูเปอร์เพียร์ ซึ่งหน้าที่หลักของซูเปอร์โหนดคือการกระจายข้อมูลที่ได้รับมาให้โหนดลูกข่ายอื่น ๆ ได้รับทราบ หรือทำตัวเป็นตัวกลางสำหรับการส่งผ่านข้อมูลของกลุ่มเพียร์ดังแสดงในรูปที่ 0-4



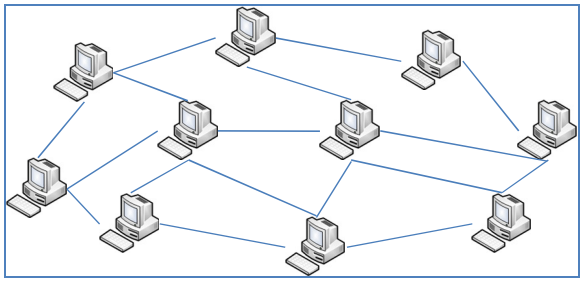
รูปที่ 0-4 การเชื่อมต่อแบบ Super P2P

ซึ่งการเชื่อมต่อดังรูปที่ 0-4 แสดงให้เห็นถึงการเชื่อมต่อของโหนดต่าง ๆ เข้าด้วยกันโดยผ่านทางซูเปอร์เพียร์ซึ่งหมายถึงโหนดที่ถูกล้อมด้วยกรอบวงกลม ข้อดีของระบบซูเปอร์เพียร์ คือความสามารถในการจัดกลุ่มโหนดออกเป็นกลุ่มที่ต้องการ เช่น โหนดที่อยู่ในพื้นที่เดียวกัน หรือโหนดที่มีความต้องการข้อมูลหรือทรัพยากรเดียวกัน แต่สำหรับข้อเสียของวิธีดังกล่าวคือเมื่อโหนดที่เป็นซุปเปอร์เพียร์ทำงานไม่ได้ จะทำให้เกิดปัญหาที่เกิดกับระบบการเชื่อมต่อแบบมีเครื่องแม่ข่าย อย่างไรก็ตามปัญหาจะเกิดขึ้นเพียงชั่วครู่เท่านั้น เพราะระบบจะมีการคัดเลือกซุปเปอร์โหนดตัวใหม่ เพื่อให้โหนดอื่น ๆ ในระบบเข้าไปเชื่อมต่อ

สำหรับประเด็นที่ยังเป็นไม่ชัดเจนสำหรับระบบรูปแบบนี้คือวิธีการเลือกซุปเปอร์โหนดในระบบ ซึ่งซุปเปอร์โหนดจำเป็นจะต้องเป็นโหนดที่มีความสามารถสูงกว่าโหนดอื่น ๆ ทั่วไป เช่น มีความสามารถในการประมวลผลสูงกว่า, มีหน่วยความจำที่มากกว่า, หรือการมีแบนวิธมากกว่าเป็นต้น ซึ่งกระบวนการค้นหาโหนดยังไม่มีมาตรฐานที่แน่นอน

อย่างไรก็ตามระบบ Super P2P ก็ยังถูกนำไปใช้อย่างแพร่หลายทั้งในระบบซุปเปอร์เพียร์ หรือการนำไปใช้ร่วมกับระบบ Pure P2P เพื่อเพิ่มประสิทธิภาพในการทำงาน ส่วนตัวอย่างของระบบที่ใช้การเชื่อมต่อในรูปแบบนี้เช่น Kazaa, Gnutella แบบประยุกต์ เป็นต้น

2.1.0.3 Pure P2P



รูปที่ 0-5 การเชื่อมต่อแบบ Pure P2P

สำหรับการเชื่อมต่อแบบ Pure P2P ดังแสดงในรูปที่ 0-5 เป็นการเชื่อมต่อโดยที่อุปกรณ์ทั้งหมดเชื่อมต่อกันโดยให้ความสำคัญกับทุกเพียร์เท่าเทียมกัน ดังนั้นการทำงานของระบบจะมองว่าโหนดทุกตัวมีความสามารถเท่าเทียมกันทั้งหมดและโหนดเหล่านั้นจะกระจายการทำงานออกไปให้โหนดลูกข่ายอื่น ๆ ทำงานอย่างเท่าเทียมกัน ซึ่งระบบการเชื่อมต่อแบบ Pure P2P นี้ได้ถูกแบ่งออกสองแบบใหญ่ ๆ ได้แก่ แบบไร้โครงสร้าง และแบบมีโครงสร้าง ซึ่งจะอธิบายในหัวข้อ 2.1.0.5 และหัวข้อ 2.1.0.6

2.1.0.4 เปรียบเทียบการทำงานของระบบเพียร์ทูเพียร์แบบต่าง ๆ

ดังจะเห็นได้ในหัวข้อที่ 2.1.0.1, 2.1.0.2, และ2.1.0.3 ว่าระบบเครือข่ายแบบเพียร์ทูเพียร์สามารถแบ่งออกได้เป็นสามรูปแบบใหญ่ ๆ นั่นคือ Hybrid P2P, Super P2P และ Pure P2P โดยทั้งสามรูปแบบสามารถสรุปข้อดีข้อเสียได้ตามตารางที่ 2-2

ตารางที่ 0-2 ตารางเปรียบเทียบระบบ Hybrid P2P, Super P2P และ Pure P2P

	Hybrid P2P	**Super P2P**	**Pure P2P**
ความรวดเร็วในการค้นหาข้อมูล	สามารถค้นหาได้เร็วที่สุดเนื่องจากข้อมูลทั้งหมดถูกเก็บอยู่บนเครื่องแม่ข่าย	ค้นหาได้ช้ากว่า Hybrid P2P เนื่องจากมีการกระจายเก็บข้อมูลไปยังซูเปอร์โหนด	การค้นหารูปแบบนี้จะมีความรวดเร็วน้อยที่สุดเมื่อเทียบกับการค้นหาก่อนหน้าทั้งสองแบบ

ตารางที่ 2-2 ตารางเปรียบเทียบระบบ Hybrid P2P, Super P2P และ Pure P2P (ต่อ)

	Hybrid P2P	**Super P2P**	**Pure P2P**
ความน่าเชื่อถือของระบบ	มีความน่าเชื่อถือที่ต่ำ เนื่องจากข้อมูลทั้งหมดถูกเก็บอยู่ที่เครื่องแม่ข่าย ทำให้เมื่อข้อมูลที่เครื่องแม่ข่ายไม่สามารถเข้าถึงได้จะส่งผลกระทบถึงระบบทั้งหมด	ระบบมีความน่าเชื่อถือมากกว่าแบบ Hybrid P2P เนื่องจากมีการกำหนดซูเปอร์โหนดขึ้นมาเพื่อเก็บข้อมูลบางส่วนของระบบ ซึ่งหากซูเปอร์โหนดไม่สามารถเข้าถึงได้ จะมีกระบวนการกำหนดซูเปอร์โหนดขึ้นมาใหม่อัตโนมัติ	มีความน่าเชื่อถือของระบบมากที่สุด เนื่องจากข้อมูลจะถูกกระจายเก็บอยู่ในแต่ละโหนดในกลุ่มเพียร์ ทำให้มีขนาดของหน่วยเก็บข้อมูลที่สูงมาก จึงสามารถเก็บข้อมูลที่ซ้อนกันได้ เพื่อป้องกันการสูญหาย และโหนดในระบบสามารถเข้าถึงกันได้ตลอดเวลา
การขยายตัวของระบบ	ระบบสามารถขยายตัวได้ยาก เนื่องจากโหนดทุกโหนดจำเป็นจะต้องเชื่อมต่อไปยังเครื่องแม่ข่ายรับข้อมูลในการเชื่อมต่อโหนดอื่น ๆ ดังนั้นการขยายขนาดของระบบจะต้องอาศัยเครื่องแม่ข่ายที่มีความสามารถในการประมวลผลที่สูง	สามารถขยายระบบได้ง่าย เนื่องจากโหนดเข้ามาเชื่อมต่อกับซูเปอร์โหนดเรื่อย ๆ หากมีโหนดเข้ามาเชื่อมต่อซูเปอร์โหนดมากจนเกินไป ซูเปอร์โหนดจะเปลี่ยนตัว	สามารถขยายระบบได้ง่าย เพราะโหนดสามารถเข้าร่วมในส่วนใดของระบบก็ได้
ความหนาแน่นของข้อมูลที่เกิดระบบ (Congestion)	ปริมาณของข้อมูลที่ไหลผ่านเครื่องแม่ข่ายมีจำนวนมาก	ระบบจะกระจายความหนาแน่นของข้อมูลไปยังซูเปอร์โหนดต่าง ๆ ทำให้ความหนาแน่นของข้อมูลที่เกิดขึ้นในแต่ละซูเปอร์โหนดมี	ระบบมีการกระจายตัวอย่างเต็มที่ทำให้ข้อมูลที่วิ่งผ่านในระบบไม่มีการกระจุกตัวอยู่ที่โหนดใดโหนดหนึ่งมากจนเกินไป

		จำนวนลดน้อยลง	

ตารางที่ 0-2 แสดงให้เห็นถึงการเปรียบเทียบระบบเครือข่ายเพียร์ทูเพียร์ทั้งสามรูปแบบนั่นคือ การเชื่อมต่อรูปแบบ Hybrid P2P, Super P2P, และ Pure P2P ซึ่งโดยสรุปแล้ว ระบบ Hybrid P2P เหมาะสำหรับระบบที่ต้องการค้นหาข้อมูลอย่างรวดเร็ว ซึ่งเหมาะที่จะนำไปใช้ในระบบเครือข่ายที่มีจำนวนของโหนดไม่มากนักเนื่องจากเครื่องแม่ข่ายซึ่งรับหน้าที่ในการเก็บค่าของข้อมูลต่าง ๆ เอาไว้จะรับภาระงานมากตามจำนวนโหนดในระบบไปด้วย และยิ่งไปกว่านั้นการสร้างระบบ Hybrid P2P จำเป็นจะต้องอาศัยโหนดที่ทำหน้าที่เป็นเครื่องแม่ข่ายอย่างน้อยหนึ่งเครื่องในระบบ ดังนั้นในการจัดตั้งระบบ Hybrid P2P จึงเกิดขึ้นได้ยาก

สำหรับระบบ Super P2P มีการประยุกต์การทำงานของ Hybrid P2P โดยการนำความสามารถของเครื่องแม่ข่ายมากระจายในกลุ่มของเพียร์ เพื่อให้โหนดบางตัวในระบบทำหน้าที่ในการรับภาระงานของเครื่องแม่ข่ายออกไป ซึ่งการทำงานจะช่วยลดความหนาแน่นของข้อมูลที่ต้องส่งไปยังเครื่องแม่ข่ายได้ แต่อย่างไรก็ตามระบบดังกล่าวยังมีการกำหนดโหนดที่ต้องรับภาระงานมากกว่าโหนดอื่น ๆ อยู่ดี และระบบจะมีปัญหาได้เมื่อโหนดเหล่านั้นไม่สามารถเข้าถึงได้ แต่ในการสร้างระบบดังกล่าวสามารถสร้างได้ง่ายกว่าระบบ Hybrid P2P มากเนื่องจากโหนดที่เข้าร่วมในกลุ่มทั้งหมด จะเริ่มจากการมองทุกโหนดให้มีความสามารถเท่าเทียมกัน หลังจากนั้นจึงค่อยแบ่งโหนดที่มีความสามารถสูง (ในแง่ของความสามารถทางด้านระบบเครือข่าย เช่น มีแบนวิธที่มากกว่าโหนดอื่น, หรือในแง่ของความสามารถในการประมวลผล เช่น มีความเร็วของซีพียูที่สูง เป็นต้น) ออกมาเพื่อให้โหนดเหล่านั้นรองรับภาระคำร้องขอจากเพียร์อื่น ๆ ในระบบ

ระบบ Pure P2P ถูกเลือกให้นำมาใช้สำหรับวิทยานิพนธ์ฉบับนี้เนื่องจาก ระบบดังกล่าวสามารถรองรับโหนดจำนวนมากได้ และการสร้างระบบซึ่งประกอบด้วยโหนดจำนวนมากสามารถทำได้โดยง่าย เนื่องจากระบบจะมองโหนดทุกโหนดที่อยู่ในระบบให้มีความสามารถเท่าเทียมกัน ทำให้ระบบ Pure P2P มีความสามารถในการการจายตัวของภาระงานได้ดีมาก และสามารถขยายขนาดของระบบได้ง่ายอีกด้วย โดยที่ไม่จำเป็นต้องมีการคำนึงถึงความสามารถของโหนดแต่ละตัว ซึ่งระบบจะกระจายงานไปให้แก่โหนดต่าง ๆ ในระบบอย่างเท่าเทียมกัน ทำให้ระบบมีความน่าเชื่อถือของข้อมูลมากยิ่งขึ้น เนื่องจากเมื่อเพียร์ใดเพียร์หนึ่งต้องการออกจากระบบเพียร์ดังกล่าวจะส่งสัญญาณโอนข้อมูลที่ตนเองเก็บไว้ไปให้กับเพียร์อื่น ๆ ได้ทันที

เพื่อการลดปัญหาต่าง ๆ ที่เกิดขึ้นจากการรวมกลุ่มของข้อมูลที่จุดเดียว เช่น การใช้งานเครื่องแม่ข่าย ซึ่งก่อให้เกิดปัญหาการสูญหายของข้อมูลที่จุดเดียว (Single point of failure), ปัญหาความคับคั่งของข้อมูลทำให้ระบบไม่สามารถทำงานต่อไปได้ (congestion), ปัญหาการขาดความน่าเชื่อถือของระบบ, ปัญหาการสร้างกลุ่มเพียร์ซึ่งสามารถทำได้ยาก, และปัญหาการขยายตัวของระบบที่สามารถทำได้ยาก วิทยานิพนธ์ฉบับนี้จึงเลือกระบบเครือข่ายเพียร์ทูเพียร์แบบ Pure P2P เพราะระบบดังกล่าวมีการกระจายข้อมูลที่ต้องเก็บไปยังเพียร์ต่าง ๆ ในระบบ อย่างไรก็ตามระบบ Pure P2P ปัญหาอยู่ที่ความเร็วในการค้นหา ซึ่งสามารถบรรเทาลงได้

ด้วยการเลือกใช้การเชื่อมต่อ Pure P2P ในรูปแบบที่เหมาะสมกับงานที่นำไปใช้งาน และการนำการเชื่อมต่อแบบ Super P2P เข้ามาเป็นทางเลือกหนึ่งเพื่อเพิ่มความเร็วให้กับการค้นหา ซึ่งวิทยานิพนธ์ฉบับนี้จะกล่าวถึงกระบวนการแก้ปัญหาดังกล่าวอย่างละเอียดต่อไป

2.1.0.5 Pure P2P แบบไร้โครงสร้าง (Unstructured P2P)

การเชื่อมต่อ Pure P2P แบบไร้โครงสร้าง เป็นการเชื่อมต่อระหว่างโหนดโดยที่แต่ละโหนดไม่รู้ถึงโหนดข้างเคียงดังรูปที่ 0-5 โดยโหนดที่ทำการส่ง จะทำการส่งสัญญาณออกไปแบบบรอดคาสต์เพียงอย่างเดียว ซึ่งระบบดังกล่าวมีข้อดีที่ความสามารถในการขยายจำนวนของโหนดในระบบสามารถทำได้อย่างง่าย และสามารถรองรับการที่โหนดเข้าและออกบ่อย ๆ ได้ดีอีกด้วย แต่อย่างไรก็ตามเนื่องจากการทำงานของระบบแบบไร้โครงสร้างทำงานโดยอาศัยวิธีการบรอดคาสต์ในการส่งข้อมูล ซึ่งกระบวนการบรอดคาสต์ จะเป็นการส่งสัญญาณจากโหนดเริ่มต้นไปยังโหนดข้างเคียง และให้โหนดข้างเคียงกระจายข้อมูลต่อไปเรื่อย ๆ ซึ่งกระบวนการบรอดคาสต์มีการป้องกันปัญหาข้อมูลล้นในระบบเครือข่าย โดยมีการกำหนดจำนวนครั้งสูงสุดที่ข้อมูลจะถูกส่งไปในระบบเครือข่าย (timeout) ดังนั้นหากระบบมีขนาดใหญ่มากกระบวนการ timeout จะทำให้ข้อมูลที่ส่งไปหมดเวลาและถูกทิ้งไปก่อนที่จะไปถึงปลายทางได้ จึงทำให้ข้อมูลที่ใช้ส่งไม่สามารถรับประกันได้ว่าจะส่งถึงผู้รับปลายทางได้เสมอ และไม่สามารถรับประกันเวลาที่ใช้ในการค้นหาได้เช่นกัน

2.1.0.6 Pure P2P แบบมีโครงสร้าง (Structured P2P)

สำหรับการเชื่อมต่อแบบมีโครงสร้าง เป็นการเชื่อมต่อโดยให้โหนดแต่ละตัวมีความสามารถในการเก็บเส้นทางการค้นหาเป็นของตนเองเล็กน้อย ทำให้การค้นหาเส้นทางไปยังปลายทางไม่ได้กระทำโดยการบรอดคาสต์อีกต่อไป โดยแต่ละโหนดจะรับผิดชอบทำหน้าที่ในการช่วยกันค้นหาเส้นทางเพื่อให้ไปถึงปลายทางได้ ซึ่งกระบวนการดังกล่าวมีการประยุกต์ใช้อย่างมากมาย ซึ่งแบบที่ได้รับความนิยมจะเป็นการใช้กระบวนการ Distributed Hash Table (DHT) ซึ่งตัวอย่างของระบบที่ใช้กระบวนการ DHT ได้แก่ Chord [31], CAN [32], Pastry [33], Tapestry [34], Kademlia [35]

ตารางที่ 0-3 ตารางเปรียบเทียบระบบ Pure P2P แบบมีโครงสร้าง

	Chord	**CAN**	**Pastry**	**Tapestry**	**Kademlia**
กระบวนการที่ใช้ในการเก็บข้อมูล	DHT	Virtual coordinate zone	DHT	DHT	DHT
ความเร็วในการค้นหา[i]	O(log N)	$O(dN^{1/d})$[ii]	O(log N)	O(log N)	O(log N)
ความสามารถในการขยายระบบเครือข่ายขนาดใหญ่	สูง	สูง	สูง	ต่ำ	สูง
ความซับซ้อนของระบบ	ต่ำ	ต่ำ	สูง	สูง	สูง
รูปแบบการสร้างของเครือข่าย	วงแหวน	แบ่งกลุ่ม	วงแหวน	วงแหวน	แผนภูมิต้นไม้สมมาตร

i. ความเร็วที่ใช้ในการค้นหาจะถูกแสดงออกไปในรูปแบบของบิ๊กโอ (Big Oh) ซึ่งค่าในตารางตัวแปร N จะหมายถึงจำนวนเพียร์ทั้งหมดที่มีอยู่ในระบบ

ii. สำหรับการค้นหาในรูปแบบ CAN จะมีตัวแปรเพิ่มเติมขึ้นมานั่นคือตัวแปร “d” ซึ่งค่าหมายถึงจำนวนมิติ (dimension) ที่ใช้ในการแบ่งกลุ่มของพื้นที่ เช่น ถ้า d มีค่าเป็น 2 หมายความว่าระบบจะ แบ่งกลุ่มของโหนดออกโดยใช้เส้นแบ่งจากแนวตั้งและแนวนอน แต่หาก d มีค่าเท่ากับ 3 โหนดจะถูกแบ่งโดยเส้นแบ่งจากแนวตั้ง ,แนวนอน, และแนวลึกเป็นต้น

ตารางที่ 0-3 จะเห็นได้ว่าการเก็บข้อมูลเพื่อใช้ในการค้นหาส่วนมากจะอาศัยกระบวนการ Distributed hash table (DHT) ซึ่งระบบที่อาศัยการค้นหารูปแบบดังกล่าวจะมีความเร็วในการค้นหาเป็น O (log N) โดยรูปแบบการค้นหาแบบ CAN จะอาศัยการแบ่งกลุ่มพื้นที่แล้วให้โหนดที่อยู่ในใกล้กันเข้าร่วมกลุ่มพื้นที่เดียวกันโดยเวลาที่ใช้ในการค้นหาจะมากกว่าการค้นหาแบบ DHT

เมื่อเปรียบเทียบรูปแบบการค้นหาที่มีการใช้ DHT ทั้งหมด (Chord, Pastry, Tapestry, และ Kademlia) แล้ว Chord (หรือคอร์ด) จะเป็นรูปแบบการค้นหาที่ดีที่สุด เนื่องจากคอร์ดมีความสามารถในการรองรับการขยายตัวของระบบได้สูง โดยที่รูปแบบการค้นหาไม่ซับซ้อนอีกด้วย

2.1.0.7 การเปรียบเทียบระบบล่วงรู้บริบทเมื่อทำงานบนเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้างและการทำงานบนเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง

ในหัวข้อ 2.1.0.5 และ 2.1.0.6 ได้กล่าวถึงการทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างและไร้โครงสร้างมาแล้ว ซึ่งรายละเอียดความแตกต่างของระบบเหล่านั้นสามารถแสดงถึงข้อเปรียบเทียบดังแสดงในตารางที่ 0-4

ตารางที่ 0-4 ตารางเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนเครือข่ายเพียร์ทูเพียร์ชนิดไร้โครงสร้างและชนิดมีโครงสร้าง

	ระบบเครือข่ายแบบไร้โครงสร้าง	**ระบบเครือข่ายแบบมีโครงสร้าง**
ความเร็วในการค้นหา	ไม่สามารถคำนวณได้ โดยการค้นหาอาจใช้เวลานานมากหากโหนดมีจำนวนมากเกินไป	Log(N) โดยที่ N หมายถึงจำนวนโหนดทั้งหมดในระบบ
การรับประกันการค้นหา	ไม่สามารถรับประกันได้ว่าจะเจอข้อมูลที่ต้องการค้นหาจริง ๆ เนื่องจากเมื่อมีจำนวนโหนดในระบบมาก อาจส่งผลให้ข้อมูลที่บรอดคาสท์ออกไปค้นหาเกิด timeout ไปเสียก่อน	สามารถรับประกันได้ว่าหากข้อมูลที่ต้องการค้นหามีอยู่นระบบจะต้องค้นหาเจอ เนื่องจากการค้นหาจะมี routing table ที่แน่นอน ซึ่งหากการค้นหาไม่พบข้อมูลปลายทางจะมีการส่งสัญญาณ

		กลับมาบอกโหนดต้นทางเสมอ

ตารางที่ 2-4 ตารางเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนเครือข่ายเพียร์ทูเพียร์ชนิดไร้

	ระบบเครือข่ายแบบไร้โครงสร้าง	**ระบบเครือข่ายแบบมีโครงสร้าง**
การใช้ทรัพยากรของโหนด	โหนดจะเสียพลังงานไปกับการส่งสัญญาณบรอดคาสท์ แต่ไม่จำเป็นต้องเสียหน่วยความจำไปกับการเก็บเส้นทางการค้นหาปลายทาง	โหนดเสียพลังงานที่ต้องส่งต่อข้อความการค้นหาเพื่อให้ไปถึงปลายทาง และต้องเสียพื้นที่หน่วยความจำสำหรับเก็บเส้นทางการค้นหาปลายทาง
การใช้ทรัพยากรของระบบเครือข่าย	มีจำนวนข้อมูลรับส่งกันสูงมาก เนื่องจากการค้นหาใช้รูปแบบบรอดคาสท์เท่านั้น	การค้นหาในระบบไม่ส่งผลกระทบต่อระบบเครือข่ายมากนักเพราะข้อมูลที่ส่งจะตรงไปยังผู้รับตามเส้นทางที่ได้คำนวณไว้
เมื่อโหนดเข้าออกระบบ	ไม่ต้องมีการคำนวณเส้นทางใดใหม่ ทำให้สามารถรองรับระบบที่มีโหนดเข้าออกบ่อย ๆ ได้ดี	จะต้องมีการส่งสัญญาณไปบอกโหนดข้างเคียงเพื่อให้โหนดเหล่านั้นเปลี่ยนแปลงค่าในตารางที่ใช้สำหรับค้นหาปลายทาง ทำให้ไม่สามารถทำงานได้ดีนักเมื่อโหนดเข้าออกระบบบ่อย ๆ

โครงสร้างและชนิดมีโครงสร้าง (ต่อ)

จากตารางที่ 0-4 แสดงให้เห็นถึงการเปรียบเทียบระบบเพียร์ทูเพียร์ทั้งสองแบบ ซึ่งมีข้อดีและข้อเสียที่ต่างกันออกไป โดยระบบที่จะนำมาประยุกต์ใช้ในวิทยานิพนธ์ฉบับนี้ เป็นระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง เนื่องจาก

1. การค้นหาของระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างสามารถทำได้รวดเร็วกว่ามาก
2. ในการค้นหาแต่ละครั้งระบบจะมีการตอบข้อมูลการค้นหากลับมาเสมอว่าการค้นหาแต่ละครั้งเจอโหนดปลายทางหรือไม่

3. ระบบเครือข่าย ไม่ต้องรับภาระหนักเนื่องจากไม่ได้มีการส่งข้อมูลแบบบรอดคาสท์ ทำให้ความหนาแน่นของข้อมูลลดลงไปได้มาก
4. ถึงแม้ว่าต้องเสียทรัพยากรหน่วยความจำเพื่อใช้สำหรับการเก็บเส้นทาง เพื่อใช้ในการค้นหาปลายทางของระบบเพิ่มมากกว่าระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง แต่หน่วยความจำที่ถูกใช้ไปจะเป็นจำนวนไม่มากนัก
5. ข้อเสียของระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างคือไม่สามารถรองรับระบบที่มีการเข้าออกของโหนดบ่อยครั้งเกินไป ซึ่งเมื่อโหนดมีการเข้าออกบ่อย ๆ จะทำให้ระบบไม่สามารถส่งสัญญาณเพื่อเปลี่ยนค่าใน finger table ของโหนดอื่น ๆ ได้ทัน จึงทำให้เกิดปัญหาความผิดพลาดของ finger table ได้ แต่อย่างไรก็ตามวิทยานิพนธ์ฉบับนี้ได้มีการกำหนดระบบที่จะนำมาใช้งานว่าโหนดที่จะเข้าร่วมระบบถูกสมมติให้เป็นโหนดซึ่งไม่มีการเข้าออกจากระบบบ่อยนัก

การทบทวนวรรณกรรม

บทนี้จะเป็นการกล่าวถึงเอกสารที่เกี่ยวข้องที่ใช้ในการทำวิทยานิพนธ์ฉบับนี้ โดยการนำเสนอในบทนี้จะเป็นการนำเสนอเกี่ยวกับกระบวนการทำงานของคอร์ด (Chord) อย่างละเอียด, การประยุกต์ใช้ระบบเพียร์ทูเพียร์ (Peer-to-Peer) สำหรับการเก็บข้อมูลออนโทโลยี (Ontology) เพื่อให้ข้อมูลออนโทโลยีที่เก็บสามารถถูกค้นหาได้อย่างรวดเร็วและมีความน่าเชื่อถือได้ หลังจากนั้นจะเป็นการกล่าวถึงการนำระบบเพียร์ทูเพียร์มาประยุกต์ใช้กับระบบล่วงรู้บริบท (Context awareness system) พร้อมทั้งเปรียบเทียบข้อดีข้อเสียของวิธีต่าง ๆ

การทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด

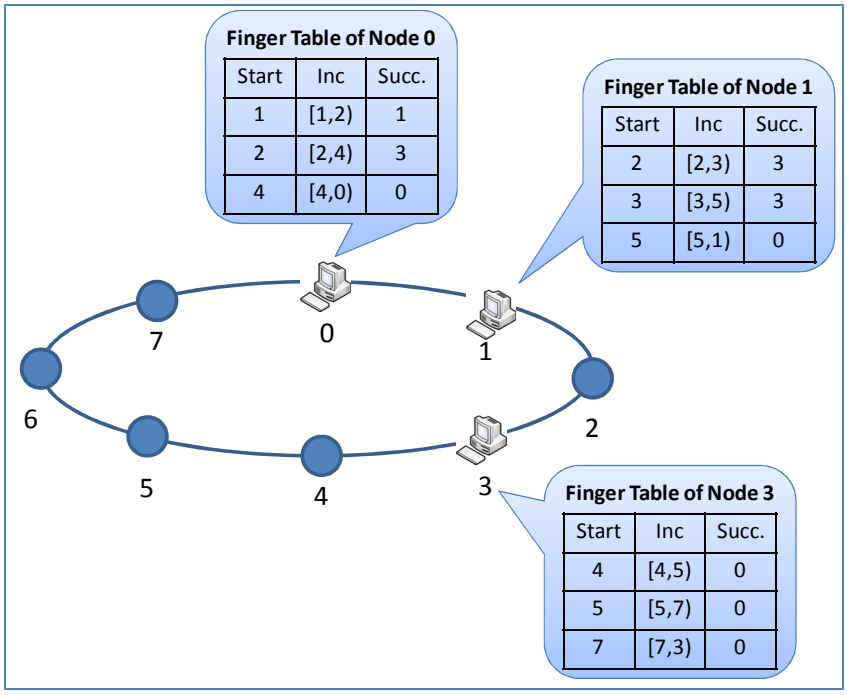
หัวข้อนี้จะเป็นการกล่าวถึงการทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด (Chord) โดยละเอียด เนื่องจากวิทยานิพนธ์ฉบับนี้ได้เลือกใช้คอร์ดเป็นรูปแบบการเชื่อมต่อพื้นฐานสำหรับการทำงานเนื่องจากระบบของคอร์ดเป็นระบบเพียร์ทูเพียร์แบบ Pure P2P แบบมีโครงสร้าง และจากบทที่ 2 แสดงให้เห็นว่าระบบคอร์ดสามารถค้นหาข้อมูลได้อย่างรวดเร็ว โดยยังคงมีความซับซ้อนของระบบน้อย และยิ่งไปกว่านั้นจำนวนทรัพยากรของระบบที่ใช้ในการเก็บข้อมูลมีน้อย ทำให้ระบบของคอร์ดเป็นที่นิยมและใช้งานกันอย่างแพร่หลาย

ก่อนจะทำความเข้าใจระบบการทำงานของคอร์ด จะต้องเข้าใจคำศัพท์ที่จะถูกใช้ในการเขียนอธิบายการทำงานเสียก่อน ซึ่งคำศัพท์เฉพาะซึ่งพบในการทำงานโดยทั่วไปของคอร์ดมีดังต่อไปนี้

1. Finger table เป็นตารางที่ใช้สำหรับเก็บเส้นทางที่ใช้ในการเดินทางของโหนดที่อยู่ในระบบ หรืออาจกล่าวได้ว่า finger table เป็น routing table สำหรับการทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ดนั่นเอง
2. Distributed hash table หรือ DHT หมายถึงตารางที่ใช้สำหรับการเก็บข้อมูลใด ๆ ทั้งหมดที่แต่ละโหนดต้องการกระจายออกมาเก็บไว้ในกลุ่มคอร์ด ซึ่งการทำงานของ DHT จะอยู่ในรูปแบบของคู่อันดับ key, value
3. Successor หมายถึงโหนดซึ่งมีหน้าที่รับผิดชอบโหนดที่ต้องการโดยตำแหน่งของ successor จะเป็นโหนดซึ่งเข้าร่วมกลุ่มคอร์ดตำแหน่งถัดไปจากโหนดที่มีอยู่เดิม เช่น จากรูปที่ 0-1 successor ของโหนดหมายเลข 3 คือโหนดหมายเลข 0 นั่นเอง ซึ่งโดยสรุปการ

ค้นหา successor เป็นการค้นหาโหนดที่อยู่ใกล้กับโหนดหมายเลขปลายทางมาก ที่สุดในทิศทางตามเข็มนาฬิกา

4. Predecessor เป็นคำที่ใช้กล่าวถึงโหนดที่ทำหน้าที่ตรงข้ามกับ successor นั่นคือ predecessor หมายถึงโหนดซึ่งเข้าร่วมคอร์ดอยู่แล้วและวางอยู่ในตำแหน่งก่อนหน้าโหนดที่ต้องการ เช่น รูปที่ 0-1 เมื่อต้องการค้นหา predecessor ของโหนดหมายเลข 3 นั่นจะหมายถึงโหนดหมายเลข 1 ดังนั้นโดยสรุปแล้วการค้นหา predecessor หมายถึงการค้นหาโหนดที่อยู่ใกล้กับโหนดหมายเลขปลายทางมากที่สุดในทิศทางทวนเข็มนาฬิกา

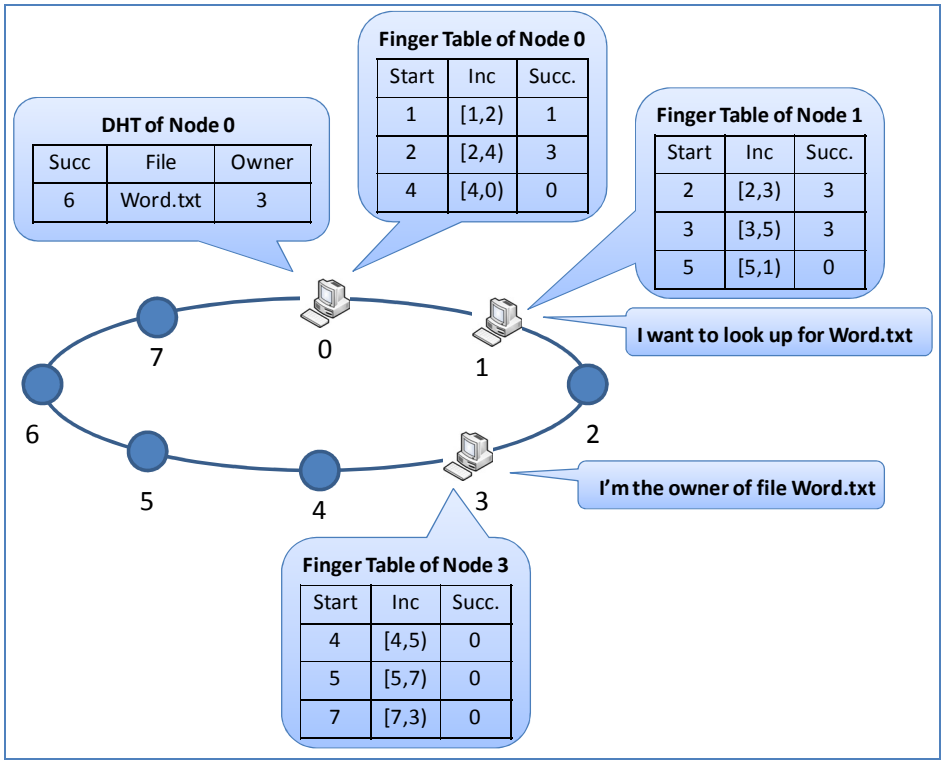


รูปที่ 0-1 การเชื่อมต่อของคอร์ด

จากรูปที่ 0-1แสดงให้เห็นถึงการเชื่อมต่อของคอร์ดซึ่งอยู่ในรูปแบบของวงแหวน (Ring topology) ซึ่งจะมีการกำหนดค่าของหมายเลขประจำโหนด (Node ID) ไว้ที่โหนดแต่ละตัว โดยในระบบข้างต้นเป็นระบบที่มีคาดหวังว่าจะมีจำนวนโหนดเข้าร่วมกลุ่มทั้งหมดสูงสุดไม่เกิน 8 โหนดแต่มีจำนวนโหนดเข้าใช้งานจริงเพียง 3 โหนดนั่นคือโหนดหมายเลข 0, 1, และ 3 (จากรูปที่ 0-1 โหนดที่เข้าร่วมกลุ่มแล้วจะถูกแทนด้วยรูปเครื่องคอมพิวเตอร์ ในขณะที่ตำแหน่งของโหนดที่ไม่ได้เข้าร่วมกลุ่มเพียร์จะถูกกำหนดด้วยรูปวงกลม) พร้อมกันนั้นภายในแต่ละโหนดจะมีการเก็บข้อมูลของการค้นหาเส้นทาง (Routing table หรือในที่นี้จะเรียกว่า Finger table) ซึ่งค่าของ finger table ในแต่ละหลักของตารางสามารถคำนวณได้ดังต่อไปนี้

1. Start สามารถคำนวณได้จาก $(n+2^{(k-1)})$ mod 2^m โดยที่ค่าของ n หมายถึงหมายเลขประจำตัวของโหนดนั้น (node ID) ส่วน k คือตำแหน่งแถวที่ต้องการค้นหาภายในตาราง Finger table และ 2^m เป็นจำนวนโหนดทั้งหมดที่คาดว่าจะมีอยู่ในระบบ
2. Inc หรือ Include ได้มาจากการนำช่วงของตัวเลขใน start มาเรียงต่อกัน
3. Succ หรือ Successor node ซึ่งค่าที่ปรากฏใน finger table เป็นค่าของ successor ตัวแรกที่พบในระบบสำหรับค่า start แต่ละตัว เช่น จากรูปที่ 0-1 ที่ finger table ของโหนดหมายเลข 0 เมื่อกำหนดว่าต้องการหาค่า succ ของ start หมายเลข 2 จะหมายถึง โหนดหมายเลข 3 เพราะโหนดหมายเลข 3 เป็นโหนด successor ที่ใกล้เคียงกับค่า start ที่ต้องการที่สุด และค่า succ ของ start หมายเลข 1 หมายถึงตัวโหนดหมายเลข 1 เอง เนื่องจากโหนดหมายเลข 1 เป็น successor ที่ใกล้เคียงกับค่า start มากที่สุดเช่นกัน

สำหรับการค้นหาเส้นทางไปยังปลายทางจะใช้ตาราง Finger table เป็นตัวช่วยในการค้นหา โดยตัวอย่างที่ใช้ในการอธิบายโปรแกรมในครั้งนี้คือระบบการค้นหาไฟล์



รูปที่ 0-2 ตัวอย่างการค้นหาข้อมูลในคอร์ดโดยใช้ DHT

1. หลังจากที่โหนดทั้งหมดเข้ามาร่วมสร้างกลุ่มของคอร์ดขึ้นมาแล้ว โหนดที่ต้องการจะกระจาย (ในที่นี้ให้เป็นโหนดที่ 3) บอกระบบว่าตนเองมีการเก็บไฟล์ที่ชื่อ “Word.txt” เอาไว้ จะนำชื่อไฟล์ไปผ่านกระบวนการเข้ารหัส (hash) แล้วจึงนำมาผ่านกระบวนการ mod เพื่อให้ได้ค่าของโหนดที่จะเป็นผู้รับผิดชอบในการค้นหาไฟล์ “Word.txt” โดยจากรูปที่ 0-2 หลังจากผ่านกระบวนการเข้ารหัสแล้วผลลัพธ์ที่ได้คือโหนดหมายเลข 6 เป็นผู้รับผิดชอบข้อมูลดังกล่าว
2. โหนดนำข้อมูลไปแจ้งยังโหนดที่ 6 ซึ่งเป็นผู้รับผิดชอบไฟล์ “Word.txt” เพื่อให้เก็บข้อมูลลงใน Distributed Hash Table (DHT) แต่จากรูปที่ 0-2 แสดงให้เห็นว่าโหนดหมายเลข 6 ยังไม่ได้เข้าร่วมในระบบนี้ ดังนั้นสัญญาณการสืบค้นจึงถูกส่งต่อไปให้ successor ของโหนดที่ 6 นั่นคือโหนดที่ 0 ตามตาราง finger table ของโหนดที่ 3 ซึ่งแสดงให้เห็นในรูปที่ 0-2 โดยเมื่อโหนดหมายเลข 3 ต้องการส่งข้อมูลไปเก็บยังโหนดหมายเลข 6 โหนดหมายเลข 3 จะมองหาใน finger table ของตนเองในช่องของ inc ซึ่งเป็นช่วงของหมายเลขโหนดปลายทาง โดยโหนดหมายเลข 6 จะอยู่ในแถวที่ start = 5, inc = [5, 7), succ = 0 ดังนั้นจึงทำให้โหนดหมายเลข 3 รู้ว่าข้อมูลของไฟล์ “Word.txt” ควรส่งไปยังโหนดหมายเลข 0 เพราะโหนดหมายเลข 0 เป็น successor ของโหนดหมายเลข 6 นั่นเอง
3. จากนั้นเมื่อโหนดหมายเลข 1 ต้องการค้นหาไฟล์ “Word.txt” โหนดดังกล่าวจะทำการเข้ารหัสชื่อของไฟล์ที่ต้องการ ซึ่งผลลัพธ์ที่ได้จะเป็นโหนดหมายเลข 6 เป็นผู้รับผิดชอบเก็บข้อมูลเกี่ยวกับไฟล์ปลายทางที่ต้องการ
4. โหนดหมายเลข 1 ค้นหาภายใน finger table ของตนเองเพื่อหาเส้นทางไปยังโหนดหมายเลข 6 ซึ่งจากรูปที่ 0-2 ในตารางหลัก Inc จะเห็นว่าโหนดที่รับผิดชอบโหนดหมายเลข 6 จะอยู่ในแถวที่สามนั่นคือโหนดหมายเลข 0 ดังนั้นโหนดหมายเลข 1 จึงส่งข้อมูลการค้นหาไปยังโหนดหมายเลข 0 เพื่อสอบถามข้อมูลของโหนดปลายทางที่เก็บไฟล์ที่ต้องการอยู่
5. หลังจากที่โหนด 0 ได้รับสัญญาณร้องขอมา จะทำการตอบกลับไปยังโหนดหมายเลข 2 ว่าไฟล์ที่ต้องการอยู่ที่โหนด 3 ดังนั้นโหนดหมายเลข 2 จึงส่งสัญญาณร้องขอไฟล์ไปยังโหนดที่ 3 ต่อไป

จากตัวอย่างจะเห็นว่าการค้นหาโดยใช้คอร์ดสามารถกระทำได้อย่างรวดเร็วโดยระยะเวลาสูงสุดที่ใช้ในการค้นหา จะไม่เกิน log(N) ให้ N หมายถึงจำนวนโหนดที่มีอยู่ทั้งหมดในระบบ และข้อมูลที่เก็บอยู่ในแต่ละโหนดเพื่อใช้ในการค้นหาจะมีจำนวนน้อย อย่างไรก็ตามข้อเสียของการค้นหาแบบคอร์ดคือ สามารถค้นหาได้เฉพาะชื่อข้อมูลแบบเฉพาะเจาะจงเท่านั้น ไม่สามารถทำการค้นหากับระบบที่มีการใช้องค์ประกอบอื่น ๆ มาช่วยในการค้นหา เช่น ระบบล่วงรู้

บริบทได้ ดังนั้นการนำระบบเพียร์ทูเพียร์แบบคอร์ดและระบบล่วงรู้บริบทมาทำงานร่วมกัน จึงจำเป็นต้องมีการประยุกต์ใช้กระบวนการทำงานให้เหมาะสม ซึ่งจะกล่าวต่อไปในบทที่ 4

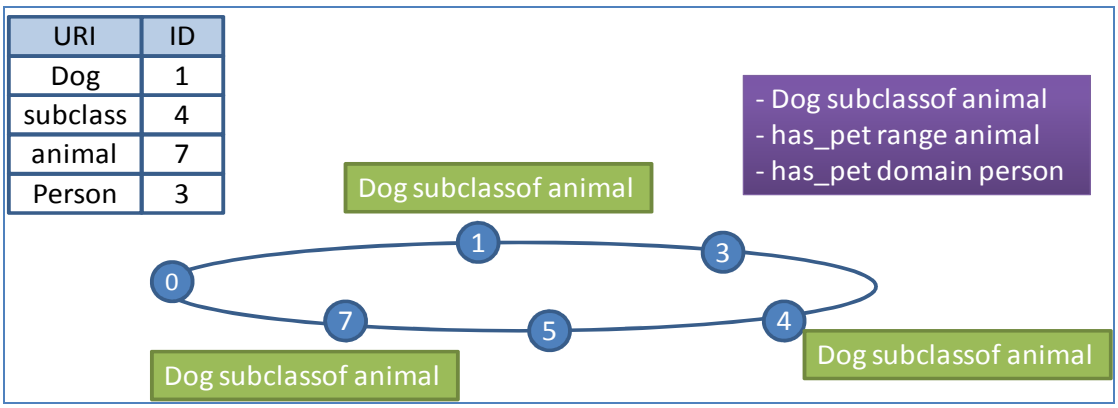
การเก็บข้อมูลออนโทโลยีบนระบบเครือข่ายเพียร์ทูเพียร์

ในหัวข้อนี้จะกล่าวถึงวิธีที่ใช้สำหรับการเก็บข้อมูลที่มีอยู่ในระบบเพียร์ทูเพียร์แบบคอร์ด (Chord) ซึ่งการเก็บข้อมูลภายในคอร์ดโดยทั่วไปจะเก็บในรูปแบบของคู่อันดับแบบกระจาย (Distributed Hash Table หรือ DHT) ซึ่งการเก็บข้อมูลในรูปแบบนี้ จะขาดความน่าเชื่อถือในการเก็บข้อมูล เนื่องจากข้อมูลที่เก็บจะถูกเก็บที่เพียร์ใดเพียร์หนึ่งในระบบอยู่เครื่องเดียว ส่งผลให้เมื่อเพียร์หลุดออกจากระบบจะส่งผลให้ระบบไม่สามารถได้รับข้อมูลที่หล่นหายไปได้ และยังไม่สามารถแก้ไขข้อมูลที่หายไปได้อีกด้วย

สำหรับข้อมูลที่จะใช้ในการเก็บครั้งนี้เป็นข้อมูลออนโทโลยี ซึ่งคุณลักษณะของข้อมูลออนโทโลยีคือ ไม่ค่อยมีการเปลี่ยนแปลงของข้อมูล ข้อมูลจำเป็นจะต้องเข้าถึงได้ตลอดเวลา มีการค้นหาข้อมูลเหล่านั้นอยู่บ่อยครั้ง และการเก็บข้อมูลต้องมีวิธีที่ใช้สำหรับการแก้ไขข้อมูลที่เสียได้ เมื่อมีเพียร์ที่เก็บข้อมูลหลุดออกจากระบบ

ซึ่งระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์ส่วนมาก [36] [37] [38] [39] ที่ได้ถูกนำเสนอไม่ได้คำนึงถึงเรื่องของการเก็บค่าออนโทโลยีภายในระบบเครือข่ายเพียร์ทูเพียร์และใช้เพียงแค่กระบวนการของคอร์ดธรรมดาเท่านั้นในการเก็บข้อมูลในกลุ่มเพียร์ทูเพียร์ อย่างไรก็ตาม ได้มีผู้เสนอกระบวนการเก็บออนโทโลยีที่เหมาะสมสำหรับระบบเครือข่ายเพียร์ทูเพียร์ขึ้นมาโดย [40] ซึ่งวิธีการทำงานจะเป็นการกล่าวถึงการเก็บออนโทโลยีในวิธีที่สอดคล้องและเหมาะสมกับการนำมาประยุกต์ใช้กับระบบที่ได้ออกแบบมา

สำหรับการทำงานของ [40] อยู่บนพื้นฐานที่ว่าออนโทโลยีสามารถแบ่งออกได้เป็นข้อมูลย่อย ๆ เรียกว่า Resource Description Framework (RDF) โดยข้อมูลย่อย RDF นี้ประกอบไปด้วยข้อมูลสามส่วนนั่นคือ subject, predicate, และ object ดังที่ได้กล่าวไว้แล้วในบทที่ 2 และข้อมูลเหล่านั้นจะถูกจัดเก็บอยู่ในกลุ่มคอร์ดโดยผ่านทาง DHT ซึ่งข้อมูลที่ถูกกระจายเก็บอยู่นั้น จะถูกคัดลอกซ้ำและกระจายเก็บอยู่ตามกลุ่มเพียร์ต่าง ๆ



รูปที่ 0-3 วิธีการกระจายเก็บข้อมูลในกลุ่มเพียร์ทูเพียร์ชนิดคอร์ด

จากรูปที่ 0-3 แสดงให้เห็นถึงวิธีการเก็บค่าออนโทโลยีในกลุ่มเพียร์ทูเพียร์ ซึ่งถูกนำเสนอโดย [40] และถูกนำมาใช้โดยตรงในวิทยานิพนธ์ฉบับนี้ ซึ่งจากรูปที่ 0-3 แสดงให้เห็นการเก็บข้อมูลในกลุ่มเพียร์ทูเพียร์แบบคอร์ด โดยกรอบด้านบนขวาเป็นการแสดงให้เห็นถึงออนโทโลยีที่ถูกแบ่งออกเป็น RDF แล้ว ซึ่งประกอบไปด้วย triple สามส่วนนั่นคือ Dog subclass of animal, has_pet range animal, และ has_pet domain person สำหรับวิธีการเก็บข้อมูลออนโทโลยีจะเป็นการนำข้อมูล triple เหล่านั้นมาเข้ากระบวนการเข้ารหัส (hash) ในรูปแบบของคอร์ด ส่งผลให้ได้ข้อมูลดังตารางบนมุมซ้ายในรูปที่ 0-3

หลังจากข้อมูลออนโทโลยีผ่านกระบวนการเข้ารหัสดังตารางข้างต้นแล้ว ข้อมูลที่ถูกเก็บไว้จะนำไปใช้สำหรับการกระจายออนโทโลยี เช่น จากรูปที่ 0-3 ออนโทโลยี triple แรกกล่าวว่า Dog subclass of animal ประกอบไปด้วย Dog ที่ถูกเข้ารหัสเป็นโหนดหมายเลข 1, subclass of ถูกเข้ารหัสเป็นโหนดหมายเลข 4, และ animal ถูกเข้ารหัสเป็นโหนดหมายเลข 7 ดังนั้นข้อมูล triple เหล่านั้นจะถูกกระจายไปตามโหนดต่าง ๆ ตามที่ได้เข้ารหัสไว้นั่นคือโหนดที่ 1, 4, และ 7 ตามลำดับ

ซึ่งวิธีดังกล่าวสามารถสรุปเปรียบเทียบกับการเก็บข้อมูลธรรมดาตามคอร์ดดังตารางที่ 0-1 ซึ่งจะสรุปให้เห็นถึงการเปรียบเทียบในแง่ของ ความเร็วในการหาข้อมูล, ความเหมาะสมในการในมาประยุกต์ใช้กับระบบค้นหาแบบบริบท, ความน่าเชื่อถือของข้อมูล, การใช้ทรัพยากร, และการแก้ไขข้อมูลที่สูญหายเมื่อโหนดหลุดออกจากระบบอย่างกระทันหัน

ตารางที่ 0-1 ตารางเปรียบเทียบการเก็บข้อมูลแบบคอร์ดธรรมดาและแบบ [40]

	วิธีการคอร์ดปกติ	**วิธีการที่นำเสนอใน [40]**
ความรวดเร็วในการค้นหาข้อมูล	log(N)	log(N)
ความเหมาะสมในการค้นหาบริบท	หา keyword ได้เฉพาะ subject, predicate หรือ object	รับ keyword เป็นตัวแปรใดก็ได้ใน triple
ความน่าเชื่อถือของข้อมูล	ข้อมูลถูกเก็บที่โหนดเพียงโหนดเดียว	ข้อมูลถูกคัดลอกเป็นสามชุดและเก็บที่โหนดสามตัว
การใช้ทรัพยากร	จำนวนข้อมูลที่เก็บเท่ากับจำนวนประโยคของ triple	จำนวนข้อมูลที่เก็บมากกว่าแบบธรรมดา 3 เท่า
การแก้ไขข้อมูลที่สูญหายเมื่อโหนดหลุดออกจากคอร์ดอย่างกระทันหัน	โหนดเจ้าของออนโทโลยีเป็นผู้แก้ไขข้อมูลเพียงผู้เดียว	โหนดอื่น ๆ ในระบบสามารถช่วยแก้ไขได้

ดังนั้นในวิทยานิพนธ์ฉบับนี้จะนำวิธีการเก็บข้อมูลตามรูปแบบของ [40] มาใช้งานโดยตรงเนื่องจาก การเก็บข้อมูลแบบดังกล่าว มีข้อดีเหนือกว่าการเก็บข้อมูลแบบคอร์ดนั่นคือสามารถค้นหาข้อมูลได้อย่างรวดเร็วพร้อมทั้งเหมาะสมกว่าในการหาข้อมูลแบบ Triple และสามารถแก้ไขข้อมูลที่เสียได้เมื่อโหนดหลุดออกจากระบบเพียร์อย่างกระทันหัน อย่างไรก็ตามจะมีข้อเสียที่ข้อมูลที่ต้องเก็บ จะเพิ่มจำนวนขึ้นมากเป็นสามเท่าจากปกติ

การประยุกต์ใช้ระบบล่วงรู้บริบทกับระบบเพียร์ทูเพียร์

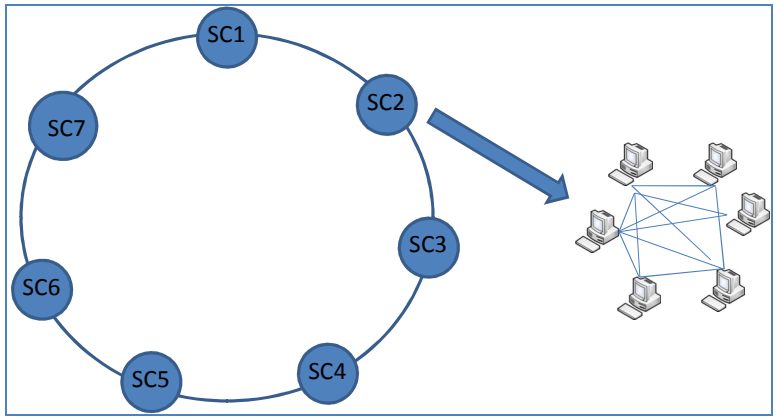
สำหรับในหัวข้อนี้จะเป็นการกล่าวถึงการทบทวนวรรณกรรมในส่วนของการประยุกต์ใช้ระบบรับรู้บริบทกับระบบเพียร์ทูเพียร์ โดยจะแยกกล่าวการทบทวนวรรณกรรม

ออกเป็นสองส่วนนั่นคือการประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง (Unstructured P2P) และการประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง (structured P2P)

การประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง

การออกแบบระบบล่วงรู้บริบทสำหรับการทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้างนั้น แนวความคิดของการส่งสัญญาณค้นหา จะตั้งอยู่บนพื้นฐานของการส่งสัญญาณแบบบรอดคาสท์ (broadcast) นั่นคือการส่งออกไปหาทุกโหนดที่ตนเองเชื่อมต่ออยู่ด้วย เนื่องจากการเชื่อมต่อของเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้างโหนดแต่ละตัวจะไม่มีโอกาสได้รับรู้ถึงเส้นทางการส่งของข้อมูล (routing) อย่างไรก็ตามการเชื่อมต่อแบบไร้โครงสร้างก็ยังมีข้อดีบางอย่าง ดังที่ได้เปรียบเทียบไปแล้วในบทที่ 2

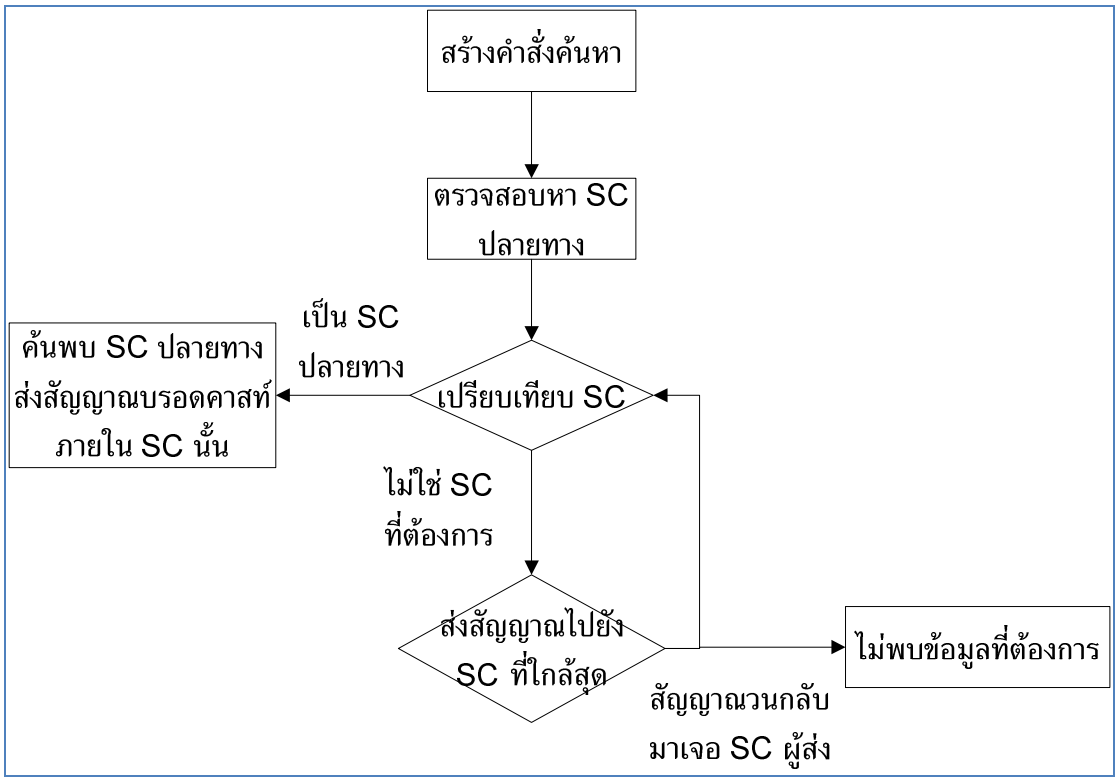
สำหรับตัวอย่างของระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้างได้แก่ [41] [42] [43] [44] จากตัวอย่างดังกล่าวมีการทำงานดังรูปที่ 0-4



รูปที่ 0-4 ระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง

จากรูปที่ 0-4 แสดงให้เห็นถึงการสร้างระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง โดยระบบนี้จะมีการแบ่งกลุ่มของโหนดที่มีบริบทเดียวกันอยู่ด้วยกัน ซึ่งกลุ่มของบริบทที่สร้างขึ้นมาจะถูกเรียกว่า Semantic Cluster (SC) และภายใน SC แต่ละตัวจะเป็นกลุ่มของโหนดที่เชื่อมต่อกันโดยอย่างไร้โครงสร้าง ดังนั้นการค้นหาภายในกลุ่ม SC จะอยู่ในรูปแบบของการส่งสัญญาณแบบบรอดคาสท์

สำหรับกระบวนการค้นหาข้อมูล จะเริ่มจากการที่โหนดหนึ่งต้องการส่งสัญญาณค้นหาออกมาโดยที่อุปกรณ์เหล่านั้น จำเป็นจะต้องรู้ปลายทางที่ใช้สำหรับการค้นหาเป็นชื่ออย่างชัดเจน หลังจากนั้นอุปกรณ์ดังกล่าวจะเปรียบเทียบสัญญาณเพื่อดูว่าโหนดปลายทางอยู่ใน SC เดียวกันหรือไม่ หากอยู่ใน SC เดียวกันก็จะทำการส่งสัญญาณบรอดคาสท์ออกไปเลย แต่หากอยู่ต่าง SC ก็จะส่งสัญญาณการค้นหาออกไปยัง SC ที่อยู่ติดกับตนเองที่สุด หากว่า SC ไม่ใช่ปลายทางก็จะส่งสัญญาณต่อไปยัง SC ตัวถัดไปเรื่อยๆ จนกระทั่งครบทั้งวง หรือค้นหาเจอ ซึ่งการทำงานสามารถอธิบายเป็นแผนภาพการทำงาน (flowchart) ดังแสดงในรูปที่ 0-5



รูปที่ 0-5 แผนภาพการทำงานของระบบล่วงรู้บริบทบนเพียร์ทูเพียร์แบบไร้โครงสร้าง

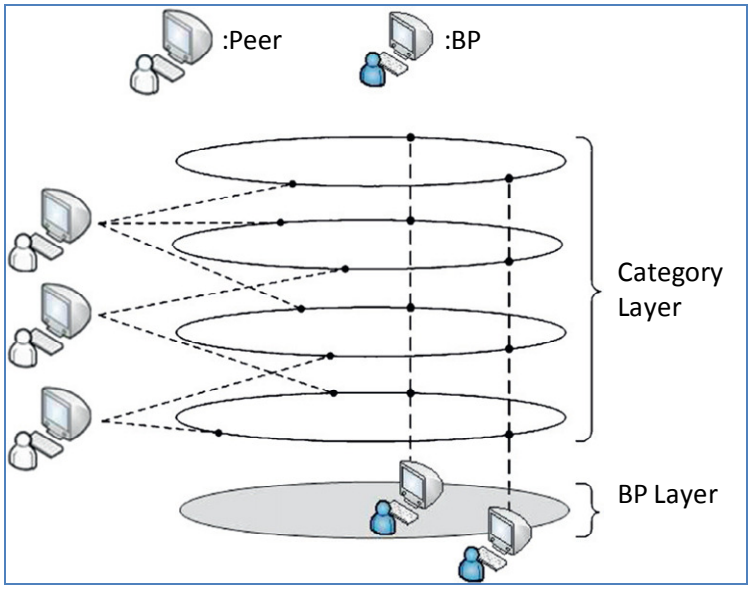
โดยสรุปแล้วการทำงานของระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้างสามารถทำงานได้ไม่รวดเร็วนัก และระบบไม่สามารถบอกกลับมาให้ผู้ส่งสามารถรับรู้ได้ว่าการค้นหาทำได้เสร็จสิ้นหรือไม่ ซึ่งข้อดีข้อเสียของระบบดังกล่าวจะกล่าวในหัวข้อต่อไป เพื่อให้สำหรับการเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนระบบเครือข่ายเพียร์ทูเพียร์แบบไร้โครงสร้าง และการทำงานบนระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง

การประยุกต์ใช้งานระบบล่วงรู้บริบทกับระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง

สำหรับการทำงานของระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้าง ได้ถูกนำเสนอขึ้นมาโดย [45] [46] [47] ซึ่งการทำงานของระบบเหล่านี้จะสามารถทำงานได้อย่างรวดเร็ว และมีการรับประกันการค้นหาว่าจะเจอที่อยู่ปลายทาง อย่างไรก็ตามระบบดังกล่าวจะมีความแตกต่างกันอยู่บ้างในกระบวนการออกแบบและการทำงาน ซึ่งการทำงานของแต่ละวิธีจะถูกกล่าวถึงโดยละเอียดดังนี้

3.2.1.1 การทำงานของระบบ ML-Chord [45]

การทำงานของระบบ [45] จะเป็นการทำงานของระบบที่เรียกว่า ML-Chord ซึ่งหลักการทำงานของระบบดังกล่าว คือการนำคอร์ดมาแบ่งเป็นกลุ่มตามจำนวนบริบทที่ระบบนี้สามารถให้บริการได้



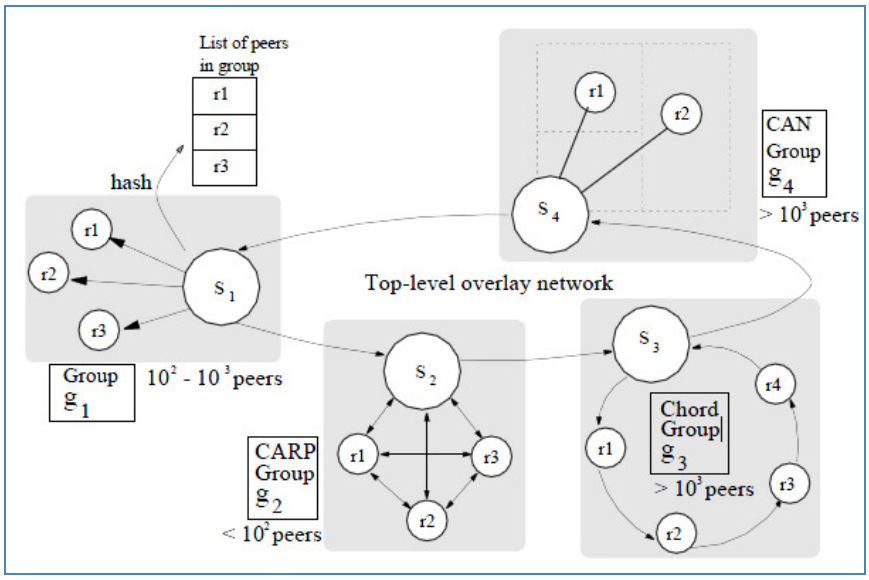
รูปที่ 0-6 การทำงานของระบบ ML-Chord

โดยจากรูปที่ 0-6 แสดงให้เห็นถึงการเชื่อมต่อของระบบ ML-Chord ซึ่งแสดงให้เห็นว่าอุปกรณ์ที่มีอยู่ทั้งหมดจะถูกแบ่งไปเข้าร่วมกลุ่มคอร์ดตามบริบทต่าง ๆ (Category Layer) และข้อมูลบริบททั้งหมดเหล่า จะถูกเชื่อมต่อกันโดยชั้นของซุปเปอร์เพียร์ (BP Layer) ซึ่งทำหน้าที่เป็นตัวผ่านของเส้นทางที่ใช้สำหรับการเชื่อมต่อของข้อมูล โดยหลักการทำงานของระบบจะเป็นการรับค่าของการค้นหาเข้ามาอยู่ในรูปของภาษา SPARQL ซึ่งระบบจะนำค่าการค้นหามาวิเคราะห์ เพื่อดูว่าปลายทางของการค้นหาจะอยู่ที่บริบทเดียวกับผู้ต้องการหาหรือไม่ หากอยู่ใน

บริบทเดียวกันระบบจะส่งสัญญาณไปค้นหาตามวิธีการของคอร์ดธรรมดา แต่หากว่าการค้นหาบ่งชี้ไปยังข้อมูลที่อยู่ในบริบทอื่น ๆ ซึ่งไม่รู้จัก ระบบจะนำข้อมูลที่ได้รับมาส่งต่อไปยังซุปเปอร์โหนด เพื่อนำข้อมูลที่ต้องการค้นหาส่งไปยัง BP Layer แล้วค่อยส่งต่อไปยังกลุ่มบริบทปลายทาง

3.2.1.2 การทำงานของระบบ Hierarchical Peer-to-Peer [46]

ส่วนระบบ [46] ไม่ได้กล่าวถึงการทำงานของระบบล่วงรู้บริบท แต่ก็มีรูปแบบของการเชื่อมต่อของระบบเพียร์ทูเพียร์ในรูปแบบที่น่าสนใจและสามารถนำมาใช้ในการเปรียบเทียบ เพื่อออกแบบระบบล่วงรู้บริบทได้ต่อไป



รูปที่ 0-7 การทำงานของระบบ Hierarchical P2P

การทำงานของระบบ [46] ได้ถูกแสดงไว้ในรูปที่ 0-7 ซึ่งการทำงานของระบบนี้จะเห็นว่ามีการนำระบบเพียร์ทูเพียร์ต่าง ๆ มาทำงานร่วมกันได้โดยผ่านทาง Top-level overlay network โดยระบบดังกล่าวได้นำเสนอการกระจายกลุ่มของข้อมูลออกไปตามจำนวนโหนดที่ต้องการใช้งาน ซึ่งระบบที่มีโหนดจำนวนน้อยก็จะถูกส่งไปยังระบบที่ทำงานด้วย CARP หรือการทำงานของระบบเพียร์ทูเพียร์แบบไร้โครงสร้าง ส่วนกลุ่มของโหนดที่มีอุปกรณ์จำนวนมากขึ้นจะถูกส่งเข้าไปยังกลุ่มเพียร์ทูเพียร์อื่น ๆ เช่น CAN และ Chord เป็นต้น

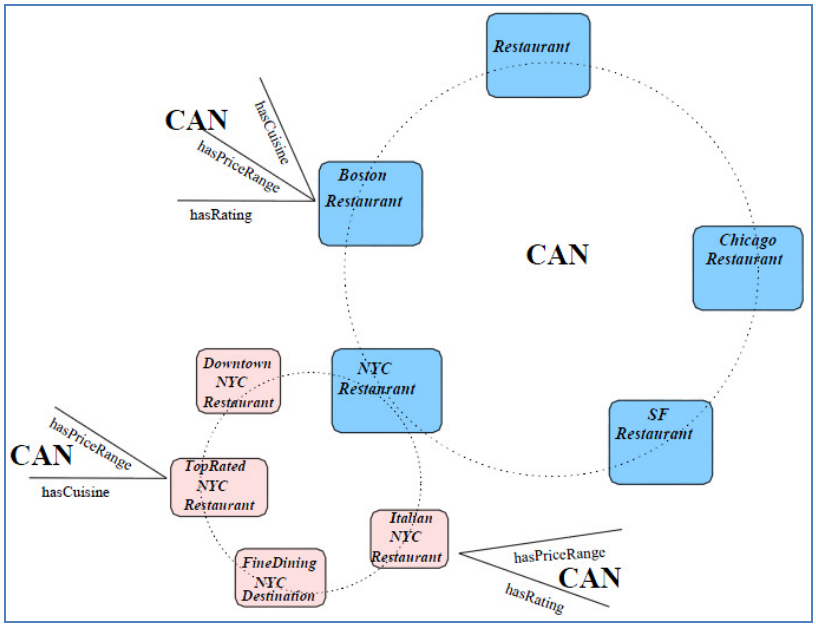
ซึ่งระบบได้แบ่งการค้นหาออกเป็นสองส่วนคือ การค้นหาภายในกลุ่มโหนดเดียวกัน และการค้นหาต่างกลุ่มโหนดกัน โดยการค้นหาภายในกลุ่มโหนดเดียวกันสามารถทำงานได้โดยใช้ระบบการค้นหาแบบปกติที่ระบบได้ถูกออกแบบไว้ ส่วนการค้นหาต่างกลุ่มโหนดนั้นระบบจะกระทำผ่านทาง Top-level overlay network ซึ่งมีการกำหนดให้แต่ละกลุ่มโหนดมีการ

สร้างซุปเปอร์โหนดเอาไว้หลายตัว และแต่ละซุปเปอร์โหนดจะมีการเชื่อมต่อซึ่งกันและกันผ่านทาง Top-level overlay network โดยใช้การเชื่อมต่อแบบ DHT ในการร้อยเรียงซุปเปอร์โหนดเหล่านั้นเอาไว้

โดยในส่วนของรายงาน [46] ไม่ได้กล่าวถึงการทำงานของระบบดังกล่าวกับระบบล่วงรู้บริบท แต่แค่กล่าวถึงในส่วนรายละเอียดของการสร้างกลุ่มเพียร์ทูเพียร์แบบนี้ขึ้นมาเท่านั้น อย่างไรก็ตาม การประยุกต์นำเอาระบบล่วงรู้บริบทเข้ามาใช้กับระบบเพียร์ทูเพียร์รูปแบบนี้สามารถทำได้โดยการนำกลุ่มของโหนดที่สร้างเอาไว้แล้วเป็นกลุ่มของบริบทไปด้วย ซึ่งจะสามารถทำให้ระบบดังกล่าวสามารถทำงานร่วมกับระบบล่วงรู้บริบทได้ทันที

3.2.1.3 การทำงานของระบบ GloServ [47]

สำหรับการทำงานของระบบ [47] หรือระบบ GloServ [48] [49] [50] [51] เป็นระบบที่ใช้เครื่องแม่ข่ายในการทำงานจัดการเกี่ยวกับระบบการตัดสินใจ (Reasoning) ภายในกระบวนการระบบล่วงรู้บริบท และมีการใช้ระบบเพียร์ทูเพียร์ชนิด CAN เข้ามาเพื่อใช้ในการจัดการข้อมูลออนโทโลยีที่อยู่ในระบบ ดังนั้นระบบดังกล่าวจึงไม่ได้เป็นการใช้ระบบเพียร์ทูเพียร์เพื่อช่วยลดปัญหาในเรื่องของการกระจายแบ่งงานออกไป แต่จะใช้เครือข่ายเพียร์ทูเพียร์เข้ามาช่วยในด้านของการจัดการข้อมูลภายในกลุ่มเครื่องแม่ข่าย



รูปที่ 0-8 การทำงานของระบบ GloServ

ดังที่ได้แสดงในรูปที่ 0-8 ซึ่งแสดงให้เห็นถึงการทำงานของระบบ GloServ โดยจากรูปจะเห็นว่าเครื่องแม่ข่ายซึ่งเป็นตัวแทนของแต่ละร้านอาหาร (ในรูปคือ Restaurant) ได้มีการเชื่อมต่อกันผ่านเครือข่ายเพียร์ทูเพียร์ชนิด CAN เพื่อแลกเปลี่ยนข้อมูลเกี่ยวกับออนโทโลยี ส่วนการทำงานของส่วนตัดสินใจหรือ reasoning จะกระทำผ่านทางเครื่องแม่ข่ายเหล่านั้น

ทำให้ระบบ GloServ ยังคงได้รับผลกระทบจากการที่ต้องติดต่ออุปกรณ์แม่ข่ายกลางเพื่อช่วยประมวลผล เช่น ปัญหาคอขวด, ปัญหาความล้มเหลวที่จุดเดียว, ขาดความน่าเชื่อถือในการให้บริการ, และปัญหาการขยายขนาดของระบบ

3.2.1.4 เปรียบเทียบการทำงานของระบบทั้งสาม

จากที่ได้กล่าวถึงระบบล่วงรู้บริบทที่ทำงานบนระบบเครือข่ายเพียร์ทูเพียร์แบบมีโครงสร้างไปทั้งสามแบบแล้ว ต่อไปจะเป็นการนำกระบวนการเหล่านั้นมาเปรียบเทียบการทำงานของระบบทั้งสามในรูปแบบของตารางดังตารางที่ 0-2

ตารางที่ 0-2 ตารางเปรียบเทียบระบบล่วงรู้บริบทที่ทำงานอยู่บนเครือข่ายเพียร์ทูเพียร์ชนิดมีโครงสร้างทั้งสามชนิด

	ML-Chord [45]	**Hierarchical [46]**	**GloServ [47]**
วิธีการเชื่อมต่อระบบเพียร์ทูเพียร์	คอร์ด (Chord)	การเชื่อมต่อรูปแบบได้ก็ได้ เช่น Chord, CAN, และการเชื่อมต่อแบบไร้โครงสร้าง	CAN
การกระจายของการประมวลผลข้อมูล	มีซุปเปอร์เพียร์หนึ่งตัวในแต่ละบริบท	มีซุปเปอร์เพียร์หนึ่งตัวในแต่ละบริบท	มีการใช้เครื่องแม่ข่ายเพื่อประมวลผล
การใช้งานกับระบบล่วงรู้บริบท	มีการประยุกต์ใช้งานกับระบบล่วงรู้บริบทในรูปแบบของ triple พร้อม มีการวัดประสิทธิภาพ	ต้องคิดวิธีการนำระบบล่วงรู้บริบทเข้ามาใช้งานเอง ขาดการวัดประสิทธิภาพ	ใช้สำหรับการกระจายการเก็บข้อมูลภายในกลุ่มเครื่องแม่ข่ายเท่านั้น ไม่ได้กระจายการใช้งานทรัพยากรออกไปในระบบเครือข่าย

โดยสรุปแล้วตารางดังกล่าวแสดงให้เห็นถึงการเปรียบเทียบในด้านต่าง ๆ ของระบบล่วงรู้บริบทที่นำมาประยุกต์ใช้กับระบบเพียร์ทูเพียร์แบบมีโครงสร้าง จะเห็นว่าระบบ GloServ [47] ไม่เหมาะสำหรับการนำมาเป็นตัวอย่างเพื่อใช้ในวิทยานิพนธ์ฉบับนี้ เพราะจุดประสงค์ของระบบ GloServ เนื่องมาจากต้องการกระจายการเก็บข้อมูลภายในกลุ่มเครื่องแม่ข่ายเพียงเท่านั้น ไม่ได้รวมไปถึงการกระจายเก็บข้อมูลไปในเครื่องลูกข่ายอื่น ๆด้วย ซึ่งขัดกับจุดประสงค์ของวิทยานิพนธ์ฉบับนี้ที่ต้องการกระจายข้อมูลไปให้ทุกโหนดในระบบอย่างเท่าเทียมกัน และพยายามกำจัดบทบาทของเครื่องแม่ข่ายออกให้ได้มากที่สุด ดังนั้นการจัดการข้อมูลของระบบล่วงรู้บริบทแบบ GloServ ยังไม่สามารถลดปัญหาการหนาแน่นของข้อมูลที่เครื่องลูกข่าย หรือปัญหาการขยายตัวของระบบ (scalability) ได้อยู่ดี ดังนั้นระบบดังกล่าวจึงไม่เหมาะที่จะนำมาใช้ในการเปรียบเทียบกับระบบที่จะนำเสนอ

สำหรับระบบ ML-Chord [45] และ Hierarchical P2P [46] มีความใกล้เคียงกันในแง่ของการประยุกต์ใช้งานระบบเครือข่ายเพียร์ทูเพียร์นั่นคือมีการสร้างกลุ่มเพียร์ทูเพียร์ย่อย ๆ ขึ้นมาแล้วเชื่อมต่อกันด้วยกลุ่มของซุปเปอร์โหนดที่รวมกลุ่มกันโดยวิธีของคอร์ด แต่ทั้งสองวิธีจะต่างกันในการที่กลุ่มบริบทย่อย ๆ ของ ML-Chord จะประกอบไปด้วยกระบวนการคอร์ดเท่านั้น ในขณะที่ Hierarchical P2P สามารถทำงานร่วมกับกระบวนการเพียร์ทูเพียร์ใดก็ได้เช่น Chord, CAN, และระบบเพียร์ทูเพียร์แบบไร้โครงสร้าง

โดยสรุปแล้วระบบล่วงรู้บริบทที่นำมาประยุกต์ใช้กับระบบเพียร์ทูเพียร์แบบมีโครงสร้าง ซึ่งจะถูกนำมาเปรียบเทียบกับระบบที่นำเสนอโดยวิทยานิพนธ์ฉบับนี้คือระบบ ML-Chord เนื่องจาก

1. ระบบ ML-Chord ลดความซับซ้อนของระบบ Hierarchical P2P ออกไป เนื่องจากระบบ ML-Chord มีการใช้งานคอร์ดเพื่อใช้ในการค้นหาในแต่ละบริบทเพียงอย่างเดียว ซึ่งระบบคอร์ดเป็นการค้นหาแบบมีโครงสร้างที่ใช้เวลาในการค้นหาปลายทางน้อย และมีการเก็บค่าข้อมูลที่ใช้สำหรับการค้นหาเส้นทางน้อยอีกด้วย
2. ระบบ Hierarchical P2P ไม่ได้ถูกออกแบบมาเพื่อระบบล่วงรู้บริบทโดยตรง ทำให้ขาดข้อมูลในด้านการวิเคราะห์เกี่ยวกับการนำมาประยุกต์ใช้กับระบบล่วงรู้บริบทอย่างเหมาะสม
3. ระบบ Gloserv เป็นกระบวนการที่ใช้สำหรับการเชื่อมต่อระหว่างเครื่องแม่ข่ายด้วยกัน มิได้มีเจตนาที่จะใช้สำหรับลดความหนาแน่นของข้อมูลที่ถูกร้องขอมาโดยเครื่องลูกข่าย และระบบ Gloserv ยังใช้การทำงานของ CAN ซึ่งเป็นการจัดกลุ่มโหนดที่อยู่ใกล้เคียงกันให้อยู่ในกลุ่มเดียวกันอีกด้วย ซึ่งวิธีดังกล่าวอาจไม่สามารถใช้งานได้เมื่อนำมาประยุกต์ใช้กับระบบที่ต้องการจะออกแบบได้เนื่องจากโหนดบางตัวในระบบ ไม่สามารถรับรู้ได้ถึงบริบทของตำแหน่งที่ตนเองอยู่

4. ระบบ ML-Chord มีการประเมินผลการทำงานในรูปแบบของการเปรียบเทียบประสิทธิภาพด้านต่าง ๆ กับระบบล่วงรู้บริบทอื่น ๆ ที่มีอยู่แล้ว ซึ่งผลลัพธ์ที่ได้นั้น แสดงให้เห็นถึงการทำงานของระบบ ML-Chord ว่าความสามารถในการทำงานนั้นเหนือกว่ารูปแบบอื่นอย่างเห็นได้ชัด

การออกแบบระบบล่วงรู้บริบทด้วยเครือข่ายเพียร์ทูเพียร์แบบคอร์ด

จากข้อมูลในบทที่ 2 ซึ่งอธิบายพื้นฐานการทำงานของระบบเครือข่ายแบบเพียร์ทูเพียร์ (Peer-to-Peer) และการทำงานของระบบล่วงรู้บริบท (Context awareness) โดยทั่วไป ประกอบกับข้อมูลในบทที่ 3 ซึ่งกล่าวถึงการนำระบบเพียร์ทูเพียร์มาประยุกต์ใช้กับระบบล่วงรู้บริบท ซึ่งมีทั้งการประยุกต์ใช้การเก็บข้อมูลออนโทโลยี (Ontology) และการเปรียบเทียบระบบล่วงรู้บริบทซึ่งถูกประยุกต์ใช้กับระบบเพียร์ทูเพียร์ต่าง ๆ ส่วนในบทนี้จะเป็นการนำเสนอถึงแนวทางและกระบวนการที่ใช้สำหรับการออกแบบระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด รวมทั้งสรุปผลการออกแบบและอธิบายถึงการทำงานของระบบที่ได้ออกแบบไว้

แนวคิดที่ใช้ในการออกแบบ

หัวข้อนี้จะเป็นการอธิบายถึงความจำเป็นที่จะต้องมีการสร้างระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์ และเหตุผลสำหรับการปรับปรุงระบบเครือข่ายเพียร์ทูเพียร์ให้ดีขึ้น ซึ่งหัวข้อนี้จะเป็นการสรุปข้อมูลดังที่ได้กล่าวมาแล้วในบทที่ 2 และ 3 เพื่อให้ง่ายต่อการทำความเข้าใจการออกแบบระบบที่สร้างขึ้นมาใหม่ต่อไป โดยเริ่มจากปัญหาที่จำเป็นจะต้องมีการสร้างระบบล่วงรู้บริบทขึ้นบนเครือข่ายเพียร์ทูเพียร์มีดังต่อไปนี้

1. ระบบล่วงรู้บริบทแบบเดิมการทำงานทั้งหมดจะกระทำผ่านทางเครื่องแม่ข่าย ซึ่งส่งผลให้เกิดปัญหาต่าง ๆ มากมาย เช่น ปัญหาคอขวด (Bottle neck) และปัญหาความล้มเหลวจากจุดเดียว (Single point of failure) เป็นต้น
2. การทำงานของระบบล่วงรู้บริบทในสถานการณ์ที่มีการเปลี่ยนแปลงของข้อมูลบริบทเป็นจำนวนมากจะส่งผลกระทบต่อระบบเครือข่ายและเครื่องแม่ข่ายอย่างมาก เนื่องจากเมื่อเครื่องลูกข่ายในระบบมีการเปลี่ยนแปลงข้อมูลบริบท ข้อมูลที่เปลี่ยนแปลงเหล่านั้นจะต้องถูกส่งไปยังเครื่องแม่ข่ายอยู่ตลอดเวลา ทำให้ปริมาณของข้อมูลที่วิ่งผ่านระบบและการทำงานของเครื่องแม่ข่ายต้องรับภาระหนักไปด้วย

3. จากปัญหาสองข้อข้างต้นส่งผลให้เกิดปัญหาเกี่ยวกับความน่าเชื่อถือของระบบตามมา โดยระบบที่มีการเชื่อมต่อแบบเพียร์ทูเพียร์จะยิ่งมีความน่าเชื่อถือของระบบเพิ่มขึ้นเมื่อจำนวนโหนดในระบบเพิ่มขึ้น ซึ่งสวนทางกับระบบการเชื่อมต่อแบบแม่ข่ายและลูกข่ายซึ่งความน่าเชื่อถือของระบบจะน้อยลงเมื่อระบบมีเครื่องลูกข่ายเพิ่มมากขึ้น

เทคโนโลยีที่ถูกนำมาใช้

สำหรับเทคโนโลยีที่ใช้ในระบบเพียร์ทูเพียร์ที่เลือกใช้จะเป็นระบบเพียร์ทูเพียร์แบบ Pure P2P ชนิดคอร์ด เนื่องจาก

1. ระบบเพียร์ทูเพียร์รูปแบบ Hybrid P2P จะส่งผลให้ระบบมีปัญหาเช่นเดียวกันกับปัญหาที่เจอในระบบเครือข่ายแบบแม่ข่ายและลูกข่าย
2. ส่วนระบบเพียร์ทูเพียร์รูปแบบ Super P2P มีการกระจายงานในระบบไปให้โหนดแต่ละตัวไม่เท่าเทียมกัน ส่งผลให้เกิดปัญหาในการเลือกซูเปอร์เพียร์ในระบบ และโหนดที่ทำตัวเป็นซูเปอร์เพียร์จะรับภาระหนักในการทำงาน
3. ในระบบเพียร์ทูเพียร์รูปแบบ Pure P2P โหนดทั้งหมดในระบบจะถูกมองว่ามีทรัพยากรเท่าเทียมกัน และสามารถทำงานได้เหมือนกันดังนั้นในระบบดังกล่าวจึงไม่ต้องกังวลว่าจะมีการใช้งานโหนดใดโหนดหนึ่งมากกว่าปกติ
4. สำหรับการค้นหาใน Pure P2P แบบไร้โครงสร้างจะเป็นการค้นหาโดยที่ระบบไม่รับรู้ถึงเส้นทางการค้นหาซึ่งจะใช้วิธีการบรอดคาสท์ในการค้นหาข้อมูล ส่งผลให้ Pure P2P แบบไร้โครงสร้างไม่ถูกนำมาใช้ในวิทยานิพนธ์ฉบับนี้

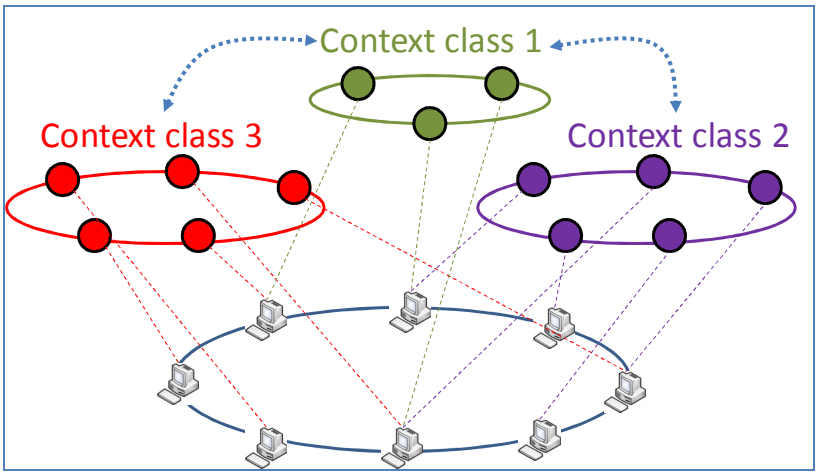
ส่วน Pure P2P แบบมีโครงสร้างมีการรับรู้ถึงเส้นทางการทำงานของโหนดรอบข้างส่งผลให้โหนดเหล่านั้นส่งข้อมูลออกไปโดยมีเส้นทางที่แน่นอน ทำให้ลดความหนาแน่นของข้อมูลในระบบ และสามารถรับประกันได้ว่าการค้นหาจะถึงปลายทางอย่างแน่นอนอีกด้วย

ส่วนระบบล่วงรู้บริบทจะไม่คำนึงถึงการออกแบบออนโทโลยีที่ดี และกระบวนการทำตัวให้เหตุผลมากนักเนื่องจากระบบที่นำเสนอในวิทยานิพนธ์ฉบับนี้เน้นการออกแบบระบบเพื่อให้โหนดต่างๆ สามารถทำงานของระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์ได้เช่นเดียวกับระบบล่วงรู้บริบทซึ่งทำงานอยู่บนระบบแม่ข่ายและลูกข่าย และการทำงานของระบบล่วงรู้บริบทที่เลือกมา จะจำกัดอยู่ที่ภาษา RDF และ OWL-Lite เนื่องจากการทำงานของ OWL-DL, OWL-Full มีรูปแบบที่ซับซ้อนและไม่มีรูปแบบที่ตายตัว จึงไม่สามารถเขียนออกมาในรูปแบบของ triple ได้ในบางกรณี

ดังนั้นเทคโนโลยีที่นำมาใช้สำหรับระบบล่วงรู้บริบทจะเป็นเทคโนโลยีที่จำกัดอยู่ที่การใช้งาน Triple เป็นหลัก เนื่องจากภาษา triple มีการเขียนคำอธิบายอย่างมีระบบทำให้ง่ายต่อการจัดการ ซึ่งภาษาที่ได้กล่าวมาได้แก่ภาษา RDF และ OWL-Lite ซึ่งใช้อธิบายออนโทโลยี ส่วนภาษาที่ใช้ค้นหาข้อมูลภายในออนโทโลยีเป็นภาษา SPARQL

สถาปัตยกรรมของระบบ MF-P2P

ระบบที่ได้ออกแบบใหม่จะใช้ชื่อว่า MF-P2P หรือ Multi finger table Peer-to-Peer เนื่องจากระบบที่ได้ออกแบบไว้เป็นการนำระบบเพียร์ทูเพียร์แบบคอร์ดมาประยุกต์โดยการเพิ่มจำนวน Finger table ขึ้นมากกว่าเดิม เพื่อให้สนับสนุนการทำงานในหลายบริบทจึงสามารถทำงานได้อย่างสอดคล้องกับระบบล่วงรู้บริบททั่วไป ซึ่งเมื่อมีการแบ่งโหนดให้มีการสร้าง finger table ขึ้นมากกว่าหนึ่งตารางแล้ว จะส่งผลให้โหนดในระบบสามารถเข้าร่วมการทำงานในหลายบริบทได้พร้อมกันโดยที่แต่ละโหนดยังเข้าร่วมกลุ่มคอร์ดเดียวกันอยู่ด้วย ดังแสดงในรูปที่ 0-1 ภาพรวมของระบบ MF-P2P



รูปที่ 0-1 ภาพรวมของระบบ MF-P2P

จากรูปที่ 0-1 โหนดทั้งหมดจะถูกนำเข้ามาร่วมกลุ่มเพียร์ทูเพียร์แบบคอร์ดที่บริเวณด้านล่างซึ่งแสดงให้เห็นด้วยรูปเครื่องคอมพิวเตอร์ หลังจากนั้นโหนดแต่ละตัวจะเข้าร่วมกลุ่มคอร์ดด้านบนตามบริบทที่โหนดแต่ละตัวนั้นได้ใช้งานอยู่ ซึ่งการทำงานของระบบ MF-P2P นี้จะมีการดัดแปลงการทำงานของคอร์ดเพื่อให้เอื้ออำนวยต่อการทำงานในระบบล่วงรู้บริบทมากยิ่งขึ้น และการแบ่งโหนดออกเป็นกลุ่มย่อย ๆ ดังที่แสดงในรูปที่ 0-1 นั้นให้โหนดแต่ละตัวเพิ่มตาราง Finger table มากขึ้นตามจำนวนบริบทที่โหนดแต่ละตัวจะรับได้ โดยการแบ่งระดับชั้นของ MF-P2P จะกระทำโดยผ่านทางการให้โหนดแต่ละตัวมีการสร้างตารางต่าง ๆ ขึ้นมาหลายตารางตามจำนวนบริบทที่ต้องการเข้าร่วม จึงส่งผลให้การทำงานของ MF-P2P นั้นมีการแยกโหนดออกไปเป็นกลุ่มตามบริบทของตนเอง อย่างไรก็ตามการแยกไปเข้ากลุ่มบริบทต่าง ๆ นั้นไม่ได้กระทำโดยการสร้างการเชื่อมต่อใหม่เพื่อเข้าร่วมกลุ่มบริบท

สืบเนื่องมาจากการที่ระบบ MF-P2P ไม่ได้มีการแยกกลุ่มโหนดเพื่อไปเข้าร่วมบริบทต่าง ๆโดยการสร้างการเชื่อมต่อใหม่ แต่เป็นการแยกโดยการเพิ่มตาราง finger table ของกลุ่มบริบทปลายทางเข้าไปยังโหนดที่ต้องการเข้าร่วมกลุ่มดังกล่าวแทน ซึ่งส่งผลให้โหนดสามารถแยกไปเข้ากลุ่มบริบทที่ต้องการ ในขณะเดียวกันโหนดเหล่านั้นก็ยังสามารถเชื่อมต่อกันได้อยู่โดยผ่านทางการเชื่อมต่อในชั้นพื้นฐานหรือที่เรียกว่า Global Chord ซึ่งมีผลกระทบสำคัญต่อการสร้างระบบล่วงรู้บริบทบนกลุ่มเครือข่ายเพียร์ทูเพียร์ เนื่องจากโหนดแต่ละตัวในกลุ่มล่วงรู้บริบทจำเป็นจะต้องมีปฏิสัมพันธ์ถึงกันเสมอ

โดยการอธิบายระบบ MF-P2P นี้จะมีการแยกอธิบายเป็นสองส่วนใหญ่นั่นคือการทำงานของ MF-P2P ในระดับชั้น Global Chord ซึ่งเป็นกลุ่มที่โหนดทั้งหมดจำเป็นจะต้องเข้าร่วมกลุ่มดังกล่าวก่อนเสมอและการทำงานในระดับชั้น Context-aware Chord ซึ่งเป็นกลุ่มการเชื่อมต่อที่โหนดซึ่งมีบริบทเดียวกันจะมาเข้าร่วมกลุ่มนี้พร้อมกันหลังจากนั้นจะเป็นการอธิบายตัวอย่างการทำงานของระบบโดยการยกตัวอย่างสถานการณ์จำลอง และสุดท้ายจะเป็นการสรุปการทำงานของระบบ MF-P2P ทั้งหมด

4.2.0.1 การสร้างการเชื่อมต่อในระดับชั้น Global Chord

ระดับชั้น Global Chord หมายถึงระดับชั้นการเชื่อมต่อพื้นฐานที่โหนดทุกโหนดจำเป็นจะต้องเข้าร่วมกลุ่มดังกล่าว เพื่อแลกเปลี่ยนข้อมูลพื้นฐานกันซึ่งในที่นี้ได้แก่ ข้อมูลของ schematic ontology และข้อมูลเกี่ยวกับบริบทของโหนดตนเองผ่านทาง finger table ซึ่งถูกประยุกต์ขึ้นมาใหม่ ดังนั้นการเชื่อมต่อในระดับชั้นของ Global Chord มีประโยชน์ เพื่อให้โหนดทั้งหมดในระบบสามารถเข้าถึงออนโทโลยีในระดับของ schematic ontology ได้ และยังเป็นระดับชั้นซึ่งใช้สำหรับเชื่อมต่อโหนดที่อยู่ต่าง Context-aware Chord สามารถแลกเปลี่ยนข้อมูลกันได้อีกด้วย

กระบวนการเข้าร่วมระบบเครือข่าย MF-P2P สามารถทำได้โดยผ่านทางการใช้กระบวนการของคอร์ดทั่วไป ซึ่งการทำงานของระบบดังกล่าวนั้นโหนดจำเป็นจะต้องรับรู้ถึงโหนดที่จะเข้าร่วมการทำงานด้วยโหนดแรกซึ่งถูกเรียกว่า bootstrap peer นั่นเอง ซึ่งการค้นหา bootstrap peer อยู่นอกเหนือขอบเขตของวิทยานิพนธ์ฉบับนี้เนื่องจากยังไม่มีการกล่าวไว้อย่างชัดเจนถึงการค้นหา bootstrap peer ซึ่งกระบวนการตัวอย่างที่ได้ถูกนำเสนอมาได้แก่

1. การใช้ SAP (Service Announcement Protocol) มีความสามารถในการกระจายตำแหน่งของ bootstrap peer ออกไปในระบบเครือข่าย เพื่อให้อุปกรณ์ที่อยู่ในระบบทั้งหมดสามารถเข้าถึงระบบเพียร์ทูเพียร์ได้โดยผ่านทางโหนดดังกล่าว อย่างไรก็ตาม SAP มีข้อเสียคือโหนดที่จะกระจายไปบอกในระบบเครือข่ายจะรับภาระหนักในการเป็นช่องทางเชื่อมต่อระหว่างโหนดอื่น ๆ

2. การกระจายข้อมูลของ bootstrap peer โดยผ่านทางเครื่องแม่ข่ายกลาง ซึ่งระบบจะมีการกำหนดเครื่องแม่ข่ายกลางที่เก็บตำแหน่งของโหนดในระบบไว้เพื่อใช้สำหรับการเข้าร่วมกลุ่ม ซึ่งวิธีดังกล่าวถูกใช้อย่างแพร่หลายแต่อย่างไรก็ตามจะมีปัญหาเกี่ยวกับความปลอดภัยของกลุ่มเพียร์ เนื่องจากเครื่องแม่ข่ายมีโอกาสที่จะถูกโจมตีเพื่อรับค่าของโหนดทั้งหมดในกลุ่มเพียร์ทูเพียร์ และเครื่องลูกข่ายไม่มีโอกาสรับรู้ได้เมื่อเครื่องแม่ข่ายมีการเปลี่ยนหมายเลข IP
3. การส่งข้อมูลของเพียร์ที่อยู่ในระบบโดยผ่านทางรายชื่อผู้ต้องการติดต่อ (Contact list) ที่จะถูกส่งต่อมาโดยผ่านทางโปรแกรมประยุกต์ต่าง ๆ ซึ่งวิธีดังกล่าวไม่ถูกใช้อย่างแพร่หลายมากนัก เนื่องจาก bootstrap peer ถูกกำหนดโดยตัวผู้ใช้เองทำให้ในบางครั้งที่โหนดใน contact list ไม่ได้เข้าร่วมกลุ่มเพียร์ทูเพียร์ก็จะส่งผลให้ไม่สามารถได้รับข้อมูลของ bootstrap peer ทั้งหมดได้

โดยวิทยานิพนธ์ฉบับนี้จะไม่คำนึงถึงวิธีการได้มาซึ่ง Bootstrap peer แต่จะสมมติว่าอุปกรณ์ที่จะเข้าร่วมกลุ่มเพียร์ทูเพียร์จำเป็นจะต้องรู้จักโหนดที่เป็น bootstrap peer มาก่อนแล้วไม่ว่าจะด้วยวิธีใดก็ตาม

สำหรับการเข้าร่วมกลุ่มของอุปกรณ์ที่ต้องการเชื่อมต่อกับระบบจะใช้วิธีการเข้าร่วมกลุ่มคอร์ดโดยปกติ ซึ่งการเข้าร่วมจะมีกระบวนการดังต่อไปนี้

1. โหนดที่ต้องการเข้าร่วมกลุ่ม MF-P2P ต้องทำการคำนวณหาค่าหมายเลขประจำตัว (Node ID) เสียก่อน ซึ่งค่าดังกล่าวได้จากการนำหมายเลข IP ไปเข้ารหัสด้วยกระบวนการ hash หลังจากนั้นจึงนำค่าที่ได้มา mod กับจำนวนโหนดทั้งหมดที่คาดว่าจะถูกสร้างขึ้นในระบบ
2. หลังจากที่ได้ Node ID แล้ว โหนดที่ต้องการเข้าร่วมกลุ่มจะติดต่อไปยัง bootstrap peer เพื่อขอเข้าร่วมระบบในชั้นของ Based Chord ซึ่ง bootstrap peer จะทำการค้นหา successor ของโหนดที่ร้องขอเพื่อเข้าร่วมระบบ แล้วตอบกลับโหนดที่ต้องการเข้าร่วมกลุ่มด้วยตำแหน่งของ successor นั้น
3. โหนดที่ต้องการเข้าร่วมส่งสัญญาณเข้าร่วมระบบไปยัง successor ที่ได้รับมา ซึ่งโหนดที่ต้องการเข้าร่วมกลุ่มจะตอบกลับมาเพื่อให้โหนดสามารถเข้าร่วมระบบได้
4. หลังจากเข้าร่วมในกลุ่ม Global Chord แล้วโหนดที่ต้องการเข้าร่วมกลุ่มจะทำการคำนวณ finger table ของตนเอง พร้อมทั้งส่งสัญญาณไปแก้ไขค่า finger table ของ predecessor อื่น ๆ

ภายหลังจากที่โหนดที่ต้องการเข้าร่วมกลุ่มได้เข้าร่วมกลุ่ม Global Chord แล้วโหนดดังกล่าวต้องมีการแจกจ่ายบริบทของตนเองไปให้กับโหนดอื่น ๆ ในระบบได้รับรู้โดยผ่าน

ทาง finger table ซึ่งถูกประยุกต์ขึ้นมาใหม่ และทำการเข้าร่วมกลุ่มในชั้นของ Context-aware Chord ต่อไป

4.2.0.2 ส่วนประกอบของ Global Chord

การเชื่อมต่อในชั้นของ Global Chord มีการเชื่อมต่อที่แตกต่างจากการเชื่อมต่อของระบบ Chord ธรรมดาโดยมีการเพิ่มข้อมูลบางส่วนเข้าไป เพื่อให้สอดคล้องต่อการนำไปใช้งานในระบบล่วงรู้บริบท ซึ่งการเพิ่มของข้อมูลนั้นจะยึดหลักการว่า Global chord มีหน้าที่ในการเป็นส่วนที่ใช้สำหรับการค้นหาออนโทโลยีสำหรับระบบทั้งหมด, หน้าที่สำหรับการเป็นตัวเชื่อมเมื่อเกิดการค้นหาจากโหนดที่เข้าร่วม Context-aware Chord หนึ่งไปยังอีก Context-aware Chord หนึ่งด้วย โดยรายละเอียดส่วนประกอบของ Global Chord มีดังต่อไปนี้

4.2.0.3 Finger table

Finger table จากที่ได้กล่าวไปในบทที่ 3 แล้วถึงกระบวนการทำงานและการสร้าง finger table ซึ่งในปกติแล้ว finger table จะทำหน้าที่ในการเก็บค่าของ โหนดที่เป็น successor ให้ ซึ่งการได้มาซึ่งตารางดังกล่าวจำเป็นต้องใช้การเชื่อมต่อไปยังโหนดเหล่านั้นโดยตรงเพื่อร้องขอ Node ID ของโหนดเหล่านั้นดังแสดงในรูปที่ 0-2

Finger Table of Node 0

Start	Inc	Succ.
1	[1,2)	1
2	[2,4)	3
4	[4,0)	0

รูปที่ 0-2 Finger table ตามปกติของคอร์ดสำหรับโหนดหมายเลข 0

จากรูปที่ 0-2 ที่แสดงให้เห็นถึง finger table สามารถอธิบายได้ว่า การเก็บข้อมูล finger table ของโหนดหมายเลข 0 จำเป็นจะต้องติดต่อกับโหนดหมายเลข 1, 3, 0 เพื่อค้นหาว่าโหนดใดเป็น successor ของโหนด 1, 2, 4 บ้าง ซึ่งระบบ MF-P2P ในวิทยานิพนธ์ฉบับนี้จะใช้ประโยชน์จากการเชื่อมต่อของโหนดเหล่านั้น คือนอกจากข้อมูล Node ID หรือหมายเลขประจำโหนดแล้ว โหนดที่ร้องขอไปจะส่งสัญญาณไปถามโหนดปลายทางด้วยเช่นกันว่าโหนดดังกล่าวมีการเชื่อมต่อกับกลุ่มบริบทหรือ Context-aware Chord ใดอยู่บ้าง ซึ่งตัวอย่างที่ได้จากการทำงานแสดงในรูปที่ 0-3

Finger Table of Node 0

Start	Inc	Succ.	Context
1	[1,2)	1	PC
2	[2,4)	3	Person
4	[4,0)	0	Organizer

รูปที่ 0-3 Finger table ที่ใช้งานใน MF-P2P สำหรับหรับโหนดหมายเลข 0

จากรูปที่ 0-3 แสดงให้เห็นถึง Finger table เมื่อมีการเปลี่ยนแปลงค่าเพื่อทำให้ Global Chord ของระบบสามารถเชื่อมโยง Context-aware Chord ถึงกันได้ ซึ่งจากรูปเป็น Finger table ของโหนดหมายเลข 0 โดยเมื่อโหนดดังกล่าวได้คำนวณค่า Start ใน Finger table และส่งสัญญาณเพื่อไปค้นหา Successor แล้ว สัญญาณที่ตอบกลับมาจาก Successor จะรวมข้อมูลบริบทของ Successor ตัวนั้น ๆ มาด้วย เช่นจากรูปแสดงให้เห็นว่า Successor หมายเลข 1 มีการเข้าร่วมบริบทของ PC, ส่วน Successor ของ 2 ซึ่งในที่นี้คือโหนดหมายเลข 3 ได้เข้าร่วมบริบท Person, และโหนดหมายเลข 0 ซึ่งเป็นโหนดตัวเองเข้าร่วมกลุ่ม Organizer

จะเห็นว่าข้อมูลที่เก็บเพิ่มขึ้นมาได้มาจากการส่งสัญญาณไปถาม Successor ในแต่ละโหนดซึ่งเป็นกระบวนการโดยพื้นฐานของคอร์ดอยู่แล้ว ดังนั้นวิธีการดังกล่าวจะไม่ส่งผลกระทบต่อประสิทธิภาพโดยรวมของระบบคอร์ดมากนักเนื่องจากกระบวนการดังกล่าวเป็นเพียงการเพิ่มข้อมูลบริบทตอบกลับมาในระบบเท่านั้น และข้อมูลจะถูกนำไปใช้ในการช่วยค้นหาเพียร์ปลายทางเมื่อมีการค้นหาข้ามบริบทกัน เช่น เมื่อโหนดที่อยู่ในบริบทของ Person ซึ่งเป็นมนุษย์ต้องการค้นหาข้อมูลที่อยู่ในบริบทของ Organizer ซึ่งเป็นโหนดที่เก็บข้อมูลเกี่ยวกับตารางเวลานัดหมายและการจดจำข้อมูลในชีวิตประจำวันอื่น ๆ เป็นต้น

ในกรณีที่โหนดปลายทางมีความสามารถในการเข้าร่วมกลุ่มบริบทหลายกลุ่ม โหนดปลายทางจะตอบกลับมาในรูปแบบของกลุ่มข้อมูล (Collection) ซึ่งแสดงถึงข้อมูลบริบททั้งหมดที่โหนด successor ได้เข้าร่วมอยู่นั่นเอง

4.2.0.4 Distributed hash table (DHT)

สำหรับหน้าที่ของ DHT ในระบบของคอร์ดโดยทั่วไป จะทำหน้าที่ในการเก็บข้อมูลทั้งหมดในระบบที่แต่ละโหนดต้องการนำเสนอออกมา ซึ่งการทำงานของ DHT ของระบบ MF-P2P นี้จะมีการทำงานเช่นเดียวกันกับระบบการทำงานของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ดทั่วไป นั่นคือหลักการทำงานจะอยู่ในรูปแบบของ key, value ซึ่งเมื่อมีข้อมูลที่ต้องการกระจายในกลุ่มเพียร์ทูเพียร์เกิดขึ้น ชื่อของข้อมูลจะถูกนำไปเข้ากระบวนการเข้ารหัส (hash) แล้วจึงนำไป mod กับจำนวนของโหนดทั้งหมดที่คาดว่าจะมีอยู่ในระบบเช่นเดียวกันกับการเข้ารหัส

ของการค้นหา Node ID เพื่อเข้าร่วมกลุ่มคอร์ดนั่นเอง โดยหมายเลขที่ได้จากการเข้ารหัสมาแล้วจะเทียบได้กับหมายเลข start ในตาราง finger table ซึ่งมีไว้สำหรับกำหนดโหนดที่จะเป็นผู้รับผิดชอบข้อมูลที่เกิดขึ้นในระบบ

ในส่วนของระบบ MF-P2P จะไม่มีการเปลี่ยนแปลงรูปแบบการค้นหาภายใน DHT นั่นคือกระบวนการค้นหาและการกระจายข้อมูลใน DHT จะยังคงยึดหลักเดิม เหมือนที่ใช้ในระบบคอร์ด แต่จะมีการจัดระบบข้อมูลที่ใช้สำหรับการกระจายใน DHT ของ Global Chord นั่นคือ ข้อมูลที่ใช้กระจายใน DHT จะเป็นข้อมูลของ schematic ontology เท่านั้น เพื่อให้โหนดทั้งหมดในระบบสามารถเข้าถึง schematic ontology ได้ตลอดเวลาอย่างรวดเร็ว

สำหรับวิธีการกระจาย Schematic ontology ได้กล่าวถึงแล้วในบทที่ 3 หัวข้อ 3.2 การเก็บข้อมูลออนโทโลยีบนระบบเครือข่ายเพียร์ทูเพียร์ ซึ่งจะเห็นได้ว่า schematic ontology ได้ถูกคัดลอกออกมาสามชุดและกระจายออกไปในกลุ่ม DHT ตามความสามารถของ triple ดังที่ได้กล่าวมาแล้วในบทที่ 2

4.2.0.5 Contextual table

ในระบบล่วงรู้บริบท บริบทส่วนมากในระบบจะมีความสัมพันธ์กันกับข้อมูลบริบทอื่น ๆ ซึ่งอาจต้องมีการนำข้อมูลจากแหล่งบริบทอื่น ๆ มาใช้ร่วมในการแสดงผลออกมาแก่ผู้ใช้ เช่น การค้นหาอุปกรณ์ที่ใช้สำหรับการปรับอุณหภูมิในห้อง อาจเกิดขึ้นได้กับอุปกรณ์หลายชนิด เช่น เครื่องปรับอากาศ, พัดลม, หรือ หน้าต่าง แต่การสั่งงานอุปกรณ์เหล่านั้นให้ทำงานได้จำเป็นจะต้องมีการใช้ข้อมูลจากบริบทอื่น ๆ มาประกอบด้วย เช่น อุณหภูมิภายนอกห้อง, อุณหภูมิปัจจุบันในห้อง, อัตราค่าบริการใช้ไฟฟ้า, หรือตำแหน่งของบุคคลต่าง ๆ ที่อยู่ในห้อง เป็นต้น ซึ่งจะเห็นว่าข้อมูลเหล่านี้เป็นข้อมูลที่มาจากบริบทอื่น แต่จำเป็นต่อการตัดสินใจตามสถานการณ์ต่าง ๆ ซึ่งเหตุการณ์ดังกล่าวมักเกิดขึ้นบ่อยครั้งในระบบล่วงรู้บริบทเนื่องจากอุปกรณ์ที่อยู่ภายในระบบล่วงรู้บริบทมักจะมีความสัมพันธ์ซึ่งกันและกันเสมอ

ดังนั้น Contextual table จึงถูกนำเสนอขึ้นมาในระบบ MF-P2P เพื่อเพิ่มประสิทธิภาพการทำงานของระบบ โดยที่ contextual table เป็นตารางซึ่งใช้สำหรับเก็บความสัมพันธ์ของบริบทที่อยู่ในโหนดปัจจุบันที่ต้องเกี่ยวข้องกับบริบทอื่น ๆ รอบข้าง ซึ่งการเก็บค่าของ contextual table จะมีการเก็บคล้ายกับตาราง DHT นั่นคือมีการเก็บชื่อของบริบทรอบข้างที่โหนดจำเป็นจะต้องมีการติดต่อด้วย คู่ไปกับการเก็บตำแหน่งของเพียร์ปลายทางที่ค้นหาเจอดังแสดงในรูปที่ 0-4 ตาราง contextual table ของโหนดนี้ได้มีการเชื่อมต่อกับบริบทอื่นอีกสามบริบทนั่นคือบริบท Temperature, Calendar, และ Calculator

Contextual table

Context	Gateway Node ID
Temperature	3
Calendar	6
Calculator	1

รูปที่ 0-4 ตัวอย่างของ Contextual table

การเก็บค่าของตาราง Contextual table จะทำงานเป็นโพรเซสเบื้องหลัง (background process) นั่นหมายถึง หลังจากที่โหนดเข้าร่วมกลุ่มบริบทไปแล้ว โหนดดังกล่าวจะทำการค้นหาโหนดที่มีบริบทใกล้เคียงโดยอัตโนมัติเพื่อค้นหาโหนดที่มีบริบทตามที่ต้องการไปเรื่อย ๆ และพยายามตรวจสอบความสถานะของโหนดบริบทข้างเคียงอยู่เป็นระยะ ส่งผลให้การทำงานของ contextual table ไม่ไปขัดขวางการทำงานเดิมของระบบ

ประโยชน์ของการใช้งาน Contextual table คือโหนดที่เข้าร่วมกลุ่มไม่จำเป็นจะต้องเสียเวลาในการค้นหาโหนดปลายทางเมื่อมีการค้นหาแบบลูกโซ่ (chain lookup) หรือการค้นหาแบบที่จำเป็นต้องอ้างอิงถึงข้อมูลที่ขึ้นอยู่กับบริบทอื่นนั่นเอง ซึ่ง contextual table จะช่วยเพิ่มความเร็วได้กับการค้นหาได้เป็นสองเท่า เนื่องจากการค้นหาข้อมูลของโหนดข้างเคียงจะถูกเก็บบันทึกไว้ใน contextual table เรียบร้อยแล้ว และการค้นหาจะเป็นเพียงการค้นหาภายในตารางซึ่งถูกเก็บค่าไว้ก่อนแล้ว

การสร้างการเชื่อมต่อในระดับชั้น Context-aware Chord

ในการค้นหาของ MF-P2P ของการเชื่อมต่อในระดับชั้น Context-aware Chord โดยส่วนมากจะเป็นไปตามการทำงานของคอร์ดโดยทั่วไปโดยไม่มีการดัดแปลงเพิ่มเติมโดยส่วนประกอบของระดับชั้น Context-aware Chord ประกอบไปด้วย

4.2.1.1 Distributed Hash Table (DHT)

ข้อมูลที่จะถูกกระจายอยู่ใน DHT ประกอบไปด้วยข้อมูลของ Instance ontology ซึ่งข้อมูลดังกล่าวนั้นเป็นข้อมูลที่ใช้สำหรับบรรยายความสามารถต่าง ๆ ของโหนดที่อยู่ในบริบทดังกล่าว ซึ่งข้อมูล Instance ontology จะถูกกระจายเก็บอยู่ภายใน DHT ในรูปแบบเดียวกันกับที่เก็บข้อมูล Schematic ontology ในชั้นของ Global Chord กล่าวคือการกระจายเก็บค่าของ Instance ontology จะมีการแบ่งออนโทโลยีออกเป็น triple ย่อย ๆ เสียก่อน หลังจากนั้นโหนดซึ่งต้องการจะกระจายออนโทโลยีจะทำการเข้ารหัส (hash) ออนโทโลยีออกเป็นสามค่า (แบ่งเข้ารหัส

ตามค่าของ triple) หลังจากนั้นโหนดจึงกระจาย Instance ontology ออกไปในกลุ่มเพียร์โดยใช้ค่าที่เข้ารหัสได้ช่วยในการเป็นตัวบ่งชี้ (key) ค่าของออนโทโลยีที่ต้องการ

4.2.1.2 Finger table

ในส่วนการทำงานของ Finger table ซึ่งอยู่ในกลุ่มของ Context-aware Chord จะทำงานเหมือนกับการทำงานของคอร์ดปกติโดยไม่มีการเปลี่ยนแปลงใด ๆ ทั้งสิ้น ระบบจะมีการสร้าง finger table ซึ่งประกอบไปด้วยรายการของโหนดปลายทาง และ successor ของกลุ่มโหนดเหล่านั้น

การค้นหาข้อมูลในระบบ MF-P2P

การค้นหาในระบบ MF-P2P เป็นการค้นหาในระบบล่วงรู้บริบทในรูปแบบของการส่งคำสั่งสืบค้นข้อมูล ซึ่งจะไม่คำนึงถึงการค้นหาภายในระบบล่วงรู้บริบทโดยผ่านทางรูปแบบของการทำ rule set ดังนั้นการค้นหาของระบบล่วงรู้บริบทซึ่งมีการเลือกใช้ระบบการส่งคำสั่งสืบค้นข้อมูลจะใช้ภาษา SPARQL (SPARQL Protocol and RDF Query Language) ในการค้นหาข้อมูล ดังนั้นการทำงานของระบบที่ได้ออกแบบไว้จะรองรับออนโทโลยีซึ่งทำงานอยู่ในรูปแบบของ RDF และ OWL-Lite สำหรับในส่วนของ OWL-DL และ OWL-Full จะไม่กล่าวถึงการทำงานในรูปแบบของการใช้ rule set แต่จะทำงานโดยมีสมมุติฐานที่ว่าออนโทโลยีที่ถูกออกแบบมามีความสมบูรณ์มากพอที่จะไม่จำเป็นต้องใช้ rule set เข้าช่วยในการตัดสินใจ ซึ่งการค้นหาในระบบ MF-P2P จะถูกแบ่งออกเป็นสองส่วนนั่นคือ

1. การค้นหาภายในกลุ่มบริบทเดียวกัน โดยการค้นหานี้จะเป็นการค้นหาแบบพื้นฐานที่สุด นั่นคือผู้ร้องขอและผู้ถูกร้องขอข้อมูลอยู่ในบริบทเดียวกันหรืออาจกล่าวได้ว่าโหนดทั้งสองเข้าร่วมกลุ่ม Context-aware Chord เดียวกันนั่นเอง ยิ่งไปกว่านั้นการค้นหาในรูปแบบนี้ยังถูกนำไปใช้สำหรับการค้นหาในรูปแบบที่สอง เมื่อคำสั่งสืบค้นข้อมูล ถูกส่งมาถึงกลุ่ม Context-aware Chord ที่ต้องการแล้วอีกด้วย
2. การค้นหาข้ามกลุ่มบริบท หมายถึงการค้นหาเมื่อโหนดผู้ร้องขอ และผู้ถูกร้องขออยู่ต่างบริบทกัน ซึ่งในความเป็นจริงมีโอกาสเกิดขึ้นได้มากกว่าการค้นหาในแบบแรก และเมื่อข้อมูลที่ใช้สำหรับการค้นหาเดินทางไปถึง Context-aware Chord ปลายทางแล้วจะใช้วิธีการในรูปแบบที่ 1 ในการค้นหาภายในกลุ่ม Context-aware Chord อีกตามเดิม

หลังจากทำการค้นหาจนพบโหนดปลายทางที่ต้องการแล้ว โหนดปลายทางจะตอบกลับไปยังโหนดที่ส่งข้อมูลที่ร้องขอมาโดยตรงเพื่อให้โหนดทำการเชื่อมต่อกลับไปยังโหนดที่เก็บข้อมูลที่ต้องการเอาไว้โดยตรง

4.2.2.1 คำสั่งการสืบค้นข้อมูล (Query)

สำหรับคำสั่งการสืบค้นข้อมูลที่จะถูกนำมาใช้ในระบบ MF-P2P นี้เป็นระบบการประยุกต์ใช้ภาษา SPARQL เพื่อเข้ามาช่วยค้นหาโหนดปลายทางซึ่งเก็บข้อมูลของบริบทปลายทางที่ต้องการไว้ได้ โดยโครงสร้างของภาษา SPARQL เป็นดังที่ได้กล่าวไว้แล้วในบทที่ 2 ซึ่งแสดงให้เห็นในรูปที่ 0-5

```
PREFIX abc: <http://example.com/exampleOntology#>
SELECT ?capital ?country
WHERE {
  ?x abc:cityname ?capital ;
     abc:isCapitalOf ?y .
  ?y abc:countryname ?country ;
     abc:isInContinent abc:Africa .
}
```

รูปที่ 0-5 ตัวอย่างของภาษา SPARQL

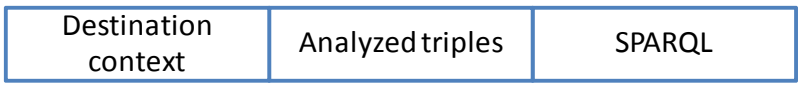
รูปแบบทั่วไปของภาษา SPARQL จะเป็นการผสมผสานระหว่างภาษา SQL และภาษา RDF หรือ OWL-Lite ซึ่งเมื่อนำภาษา SPARQL มาใช้เป็นภาษากลางสำหรับการสืบค้นข้อมูลในระบบ MF-P2P จะต้องมีการทำงานดังต่อไปนี้

1. เมื่อโหนดได้รับคำสั่งสืบค้นข้อมูลจากผู้ใช้เป็นครั้งแรกในรูปของ SPARQL แล้วโหนดจะทำการวิเคราะห์คำสั่งสืบค้นโดยเริ่มจากการจัดกลุ่มของ RDF โดยดูจากตัวแปรที่ต้องการซึ่งตัวแปรที่มีการค้นหาค่าเดียวกันจะถูกจัดกลุ่มไว้ด้วยกัน ซึ่งจากในรูปที่ 0-5 สามารถจัดกลุ่มได้เป็นสองส่วนนั่นคือ ส่วนของตัวแปร x และ y
2. คำสั่งสืบค้นข้อมูลจะถูกสร้างขึ้นมาจาก SPARQL โดยจะมีการเก็บ SPARQL ไว้ภายในคำสั่งสืบค้นข้อมูล และจะมีการกำหนดค่าของ header เพิ่มขึ้นมาเพื่อเก็บ triple ที่จะเป็นเป้าหมายปลายทาง ซึ่งในที่นี้คำสั่งสืบค้นจะถูกเก็บ triple ออกเป็นสองส่วนนั่นคือ triple แรกจะเก็บข้อมูลของ predicate isInContinent และมีการเก็บ object Africa และสำหรับ triple ที่สองจะเป็นการเก็บ predicate isCapitalOf เพียงอย่างเดียว
3. เมื่อมีการค้นหาโหนดปลายทาง โหนดที่ทำการสร้างคำสั่งสืบค้นจะนำ header มาพิจารณาว่าบริบทปลายทางคืออะไร แล้วจะเก็บข้อมูลของบริบทปลายทางไว้ใน header ใหม่ แล้ว

หลังจากนั้นจึงส่งข้อมูลไปยังกระบวนการค้นหาต่อไป ซึ่งกระบวนการค้นหาเหล่านั้นจะกล่าวถึงในบทต่อไป

โดยสรุปแล้วคำสั่งสืบค้นที่ใช้ในระบบ MF-P2P จะมีการเก็บค่าดังแสดงในรูปที่ 0-6 ซึ่งจะประกอบไปด้วยส่วนประกอบสำคัญสามส่วนนั่นคือ

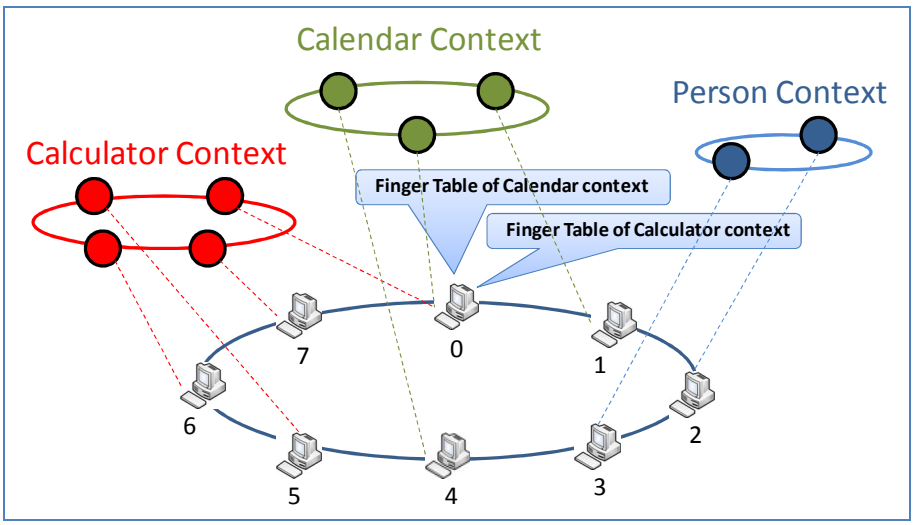
1. Destination context ใช้สำหรับเก็บบริบทปลายทางที่ต้องการค้นหาซึ่งบริบทปลายทางนี้จะเก็บอยู่ในรูปของ หมายเลขที่เข้ารหัสแล้วของบริบทปลายทาง
2. Analyzed triples เป็นส่วนที่ใช้เก็บ triple ซึ่งผ่านการวิเคราะห์มาแล้ว โดยใน header นี้อาจมีการเก็บ triple มากกว่าหนึ่ง triple ได้
3. SPARQL เป็นส่วนที่ใช้สำหรับเก็บคำสั่งเดิมภาษา SPARQL ซึ่งจะถูกนำมาใช้งานเมื่อข้อมูลถูกส่งไปยังโหนดปลายทางแล้ว



รูปที่ 0-6 ส่วนประกอบของคำสั่งสืบค้น (Query)

4.2.2.2 การค้นหาภายในกลุ่มบริบทเดียวกัน

การค้นหาภายในกลุ่มบริบทเดียวกันหมายถึง ค้นหาข้อมูลเมื่อโหนดที่ต้องการร้องขอและโหนดที่ถูกร้องขออยู่ในกลุ่มบริบทเดียวกัน ซึ่งโหนดเหล่านั้นจะรู้ได้ว่าอยู่ในกลุ่มเดียวกันก็ต่อเมื่อคำสั่งสืบค้นข้อมูลระบุถึงบริบทปลายทางที่อยู่ในกลุ่มซึ่งโหนดที่จะส่งคำสั่งสืบค้นข้อมูลมีความรู้เกี่ยวกับบริบทปลายทางที่ถูกเก็บในข้อมูลสืบค้น นั่นคือก่อนการส่งคำสั่งสืบค้นข้อมูลแต่ละครั้งโหนดที่เป็นผู้ส่งจะค้นหาตาราง finger table ของตนเองเสียก่อนว่ามีตาราง finger table ซึ่งเกี่ยวข้องกับบริบทซึ่งอยู่ในข้อมูลที่ต้องการจะสืบค้นหรือไม่



รูปที่ 0-7 ตัวอย่างของการเลือก Finger table

รูปที่ 0-7 แสดงให้เห็นถึงเหตุการณ์ที่โหนดหมายเลข 0 เป็นอุปกรณ์ที่ทำงานอยู่ในบริบทของปฏิทิน (Calendar context) และเครื่องคิดเลข (Calculator context) ซึ่งโหนดดังกล่าวมีการเก็บค่า finger table ของกลุ่มบริบท Context-aware Chord ทั้งของ calendar context และ calculator context ไว้ในโหนดเดียวกัน ดังนั้นเมื่อโหนดได้รับข้อมูลคำสั่งการสืบค้นทั้งข้อมูลที่สร้างขึ้นมาเองและข้อมูลที่ถูกส่งมาจากโหนดข้างเคียง โหนดจะนำข้อมูลเหล่านั้นมาคำนวณว่าบริบทปลายทางควรจะส่งไปที่ใด โดยการคำนวณบริบทปลายทางสามารถทำได้จาก Destination context header ซึ่งปรากฏอยู่ในคำสั่งสืบค้นแต่ละคำสั่ง ซึ่งหลังจากได้บริบทปลายทางแล้วโหนดจะนำคำสั่งมาเปรียบเทียบกับข้อมูลของตาราง finger table ซึ่งถูกเก็บอยู่ในโหนดตนเองหากพบว่าคำสั่งการสืบค้นมีระบุไว้แล้วในกลุ่มของ finger table ที่ตนเองเก็บไว้จะส่งผลให้โหนดส่งคำสั่งการสืบค้นไปค้นหาผ่านทาง finger table ที่เก็บไว้ในตนเองต่อไป

หลังจากที่โหนดรับรู้ว่าบริบทของคำสั่งการสืบค้นข้อมูลตรงกันกับข้อมูลของ Finger table ที่เก็บไว้แล้วคำสั่งการสืบค้นข้อมูลจะถูกส่งผ่าน finger table ที่ถูกต้องหลังจากนั้นคำสั่งสืบค้นข้อมูลจะถูกส่งต่อไปยังโหนดปลายทางด้วยกระบวนการของคอร์ดธรรมดา เนื่องจากการค้นหาเป็นการค้นหาภายในตาราง finger table ของ Context-aware Chord

สำหรับกระบวนการสืบค้นข้อมูลภายใน Context-aware Chord จะมีการแบ่งออกเป็นประเภทตามแต่รูปแบบของ Triple ที่ต้องการจะให้ค้นหาดังต่อไปนี้

1. คำสั่งสืบค้นไม่มีการระบุ subject, predicate, และ object ซึ่งในการค้นหาลักษณะนี้เป็นการค้นหาข้อมูลทั้งหมดที่มีอยู่ในระบบ ดังนั้นการค้นหาจึงกระทำผ่านการส่งสัญญาณ

broadcast ในกลุ่ม Global Chord โดยกรณีดังกล่าวมักไม่เกิดขึ้นในความเป็นจริง และอาจถูกห้ามใช้ในระบบซึ่งนำไปใช้งานจริงได้

2. คำสั่งสืบค้นมีการระบุ subject และ object แต่ไม่ได้มีการระบุ predicate จะเห็นว่ามีการระบุข้อมูลของโหนดปลายทางที่ต้องการอยู่แล้ว เนื่องจาก subject และ object ที่ระบุมาจะเป็นตัวสื่อถึงชื่อของโหนดเป้าหมายปลายทางได้ เช่น ประโยคการค้นหาที่ว่า “ค้นหาเครื่องคอมพิวเตอร์ของนาย ก.” สามารถแปลงเป็น triple ได้คือ ?x เป็นอุปกรณ์คอมพิวเตอร์ และนาย ก. เป็นเจ้าของ ?x ซึ่งจะเห็นว่า subject ของประโยคการค้นหา triple ที่สองจะเป็นตัวกำหนดว่าการค้นหาควรจะกระทำการค้นหาที่โหนดของนาย ก. อย่างไรก็ตามการค้นหาในความเป็นจริงคำสั่งสืบค้นในรูปแบบนี้จำเป็นจะต้องมีการระบุบริบทปลายทางอย่างชัดแจ้งเอาไว้ในคำสั่งสืบค้น เช่น “ค้นหาเครื่องคอมพิวเตอร์ของนาย ก.” เพียงอย่างเดียวไม่พอที่จะใช้ในการสืบค้น ในการค้นหาจริงจะต้องมีข้อมูลเพิ่มเติม เช่น “เครื่องคอมพิวเตอร์ ที่เป็นอุปกรณ์อิเล็กทรอนิกส์” หรือ “นาย ก. ที่เป็นบุคคล” เป็นต้น
3. คำสั่งสืบค้นมีการระบุ predicate ซึ่งการค้นหาดังกล่าวคำสั่งสืบค้นอาจมีการระบุ subject, และ object หรือมีการระบุแค่ subject หรือ object เพียงอย่างเดียวก็ได้ โดยการค้นหาสำหรับระบบดังกล่าว โหนดที่จะส่งคำสั่งการค้นหาจะต้องค้นหาออนโทโลยีภายใน Global Chord เสียก่อนว่า predicate ที่ได้ระบุมาในคำสั่งสืบค้นมีความสัมพันธ์กับบริบทใดบ้าง เมื่อคำสั่งสืบค้นถูกส่งเข้าไปยัง Context-aware Chord ที่ต้องการแล้วคำสั่งสืบค้นจะถูกค้นหาต่อไปยังโหนดซึ่งเก็บ predicate ที่ต้องการไว้

สำหรับในส่วนของการเก็บค่า triple เพื่อใช้สำหรับการค้นหา โหนดจะกระจาย Instance ontology เท่านั้นออกไปในเครือข่ายในระดับชั้น Context-aware Chord ซึ่งค่าเหล่านั้นจะถูกกระจายโดยใช้กระบวนการเดียวกันกับการกระจาย Schematic ontology ที่ได้กล่าวไปแล้วในบทที่ 3 นั่นคือโหนดจะทำการแบ่งออนโทโลยีออกเป็นประโยค triple ย่อย ๆ หลายประโยค หลังจากนั้นแต่ละ triple จะถูกกระจายออกเป็นสามส่วนย่อย ๆ ซึ่งข้อมูลแต่ละส่วนจะถูกเข้ารหัสแล้วกระจายออกไปในกลุ่ม Context-aware Chord

4.2.2.3 การค้นหาข้ามกลุ่มบริบท

สำหรับการค้นหาอีกรูปแบบซึ่งสามารถเกิดขึ้นได้ในระบบ MF-P2P คือการที่โหนดปลายทางที่ต้องการค้นหา และโหนดผู้เริ่มการค้นหาเป็นคนละตัวกัน ซึ่งสถานการณ์ดังกล่าวมีโอกาสเกิดขึ้นได้มากในความเป็นจริง เช่น โหนดที่เป็นเครื่องปรับอากาศซึ่งอยู่ในบริบทของอุปกรณ์อิเล็คโทรนิกส์ที่มีความสามารถในการปรับอากาศ ต้องการรับรู้ถึงข้อมูลอุณหภูมิปัจจุบันที่ได้มาจากโหนดซึ่งทำงานอยู่ในบริบทอุปกรณ์ที่ใช้สำหรับการวัดอุณภูมิ เป็นต้น ซึ่งจากตัวอย่างจะ

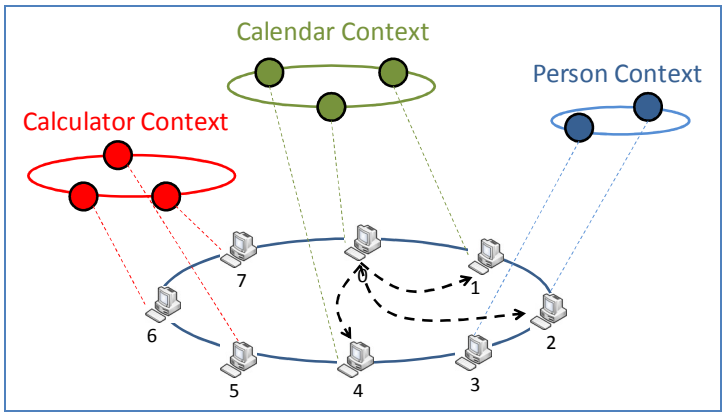
เห็นว่าโหนดที่เป็นผู้ต้องการค้นหาจะเป็นโหนดซึ่งอยู่ในบริบทที่แตกต่างกับโหนดปลายทางที่ต้องการค้นหา

ดังนั้นจะเห็นได้ว่าการค้นหาข้ามกลุ่มบริบทเป็นเหตุการณ์ซึ่งสามารถเกิดขึ้นได้ในความเป็นจริง และในระบบ MF-P2P มีความยืดหยุ่นพอที่จะให้ผู้ใช้งานเลือกกระบวนการทำงานให้เหมาะสมกับระบบที่ต้องการ โดยรูปแบบการค้นหาสำหรับการค้นหาข้ามกลุ่มบริบทมี 4 รูปแบบดังต่อไปนี้

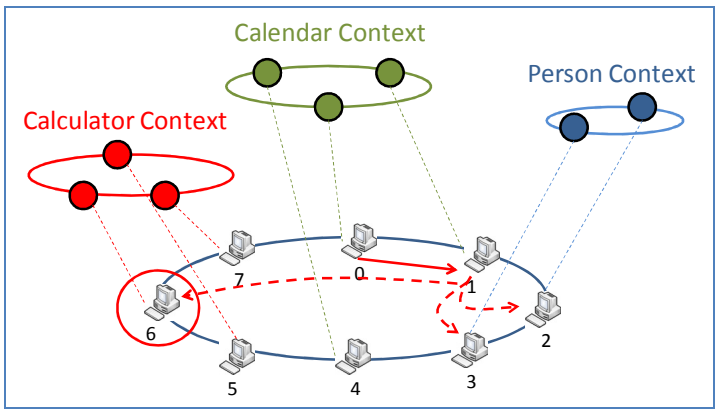
วิธีการค้นหาแบบที่ 1. การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน

การค้นหารูปแบบนี้โหนดแต่ละตัวจะทำงานโดยที่มีความสำคัญเท่าเทียมกันทั้งหมด ดังนั้นการกระจายงานจึงสามารถทำได้โดยเท่าเทียมกัน แต่อย่างไรก็ตามวิธีดังกล่าวจะมีข้อเสียความเร็วที่ใช้ในการค้นหาจะต่ำที่สุดเมื่อเทียบกับวิธีอื่น ๆ (ผลการเปรียบเทียบจะแสดงให้เห็นในบทที่ 5) ในส่วนของการทำงานในการค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน จะอาศัยหลักการของการส่งต่อ (Forward) สัญญาณต่อไปเรื่อย ๆ ในกลุ่มของ Global Chord

ระบบการทำงานของวิธีการค้นหาดังกล่าวเป็นดังที่แสดงให้เห็นในรูปที่ 0-8 และรูปที่ 0-9 ซึ่งรูปเหล่านั้นเป็นตัวอย่างการค้นหาโดยที่โหนดหมายเลข 0 ซึ่งอยู่ในบริบทของ Calendar ต้องการค้นหาข้อมูลของโหนดซึ่งทำงานอยู่ในบริบท Calculator และโหนดดังกล่าวไม่ได้เข้าร่วมบริบท Calculator อยู่ด้วย



รูปที่ 0-8 การค้นหาโดยที่ทุกโหนดมีความสำคัญเท่าเทียมกัน 1



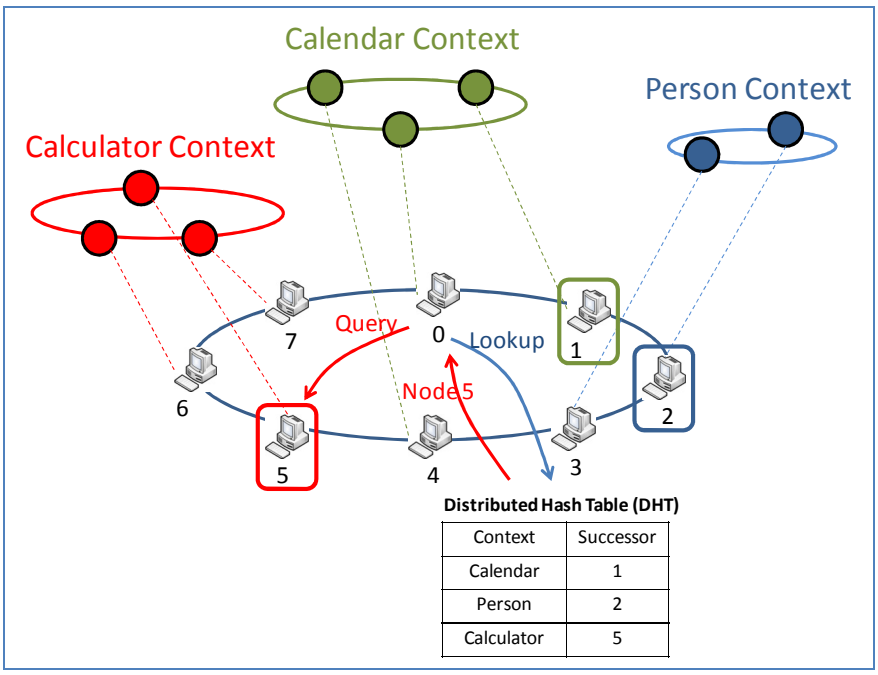
รูปที่ 0-9 การค้นหาโดยที่ทุกโหนดมีความสำคัญเท่าเทียมกัน 2

ซึ่งจากรูปที่ 0-8 แสดงให้เห็นถึงการเริ่มการค้นหาโดยโหนดหมายเลข 0 เมื่อโหนดต้องการค้นหากลุ่มบริบทปลายทาง โหนดจะเริ่มจากการค้นหา finger table ของโหนดตนเองว่ามีโหนดใดบ้างที่รับผิดชอบบริบทที่ต้องการอยู่ ซึ่ง finger table ของโหนดจะชี้ไปยังโหนดหมายเลข 1, 2, และ 4 ตามลำดับซึ่งโหนดเหล่านั้นไม่ได้เข้าร่วมในบริบท Calculator ที่ต้องการ ดังนั้นโหนดหมายเลข 0 จึงส่งสัญญาณต่อไปให้กับโหนดหมายเลข 1 เนื่องจากโหนดหมายเลข 1 เป็นโหนดที่อยู่ใกล้โหนด 0 ที่สุดใน finger table ดังแสดงให้เห็นในรูปที่ 0-9 เมื่อโหนด 1 ได้รับคำร้องขอแล้วจะทำการค้นหาภายใน finger table ของตนเองซึ่งพบว่ามีโหนดหมายเลข 6 ที่อยู่ในบริบท Calculator ดังนั้นโหนดหมายเลข 1 จะส่งสัญญาณร้องขอไปยังโหนดหมายเลข 6 โดยตรง

จะเห็นว่าวิธีการดังกล่าวมีความสามารถในการค้นหาได้โดยไม่ส่งผลกระทบต่อความหนาแน่นของเครือข่ายที่จุดใดจุดหนึ่งมากเกินไปนัก เนื่องจากวิธีดังกล่าวจะกระจายสัญญาณการค้นหาไปยังโหนดข้างเคียงไปเรื่อย ๆ เพื่อค้นหาโหนดที่อยู่ใกล้ที่สุดซึ่งมีหน้าที่รับผิดชอบบริบทปลายทางทำให้การค้นหาเกิดขึ้นอย่างกระจายและแบ่งเบาภาระของการค้นหาไปยังโหนดต่าง ๆ ในระบบ แต่จะมีข้อเสียคือความเร็วในการค้นหาสำหรับการค้นหาจะต่ำมาก

วิธีการค้นหาแบบที่ 2. การค้นหาแบบกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท

การค้นหารูปแบบนี้เป็นการกำหนดซูเปอร์โหนดสำหรับเป็นช่องทางเชื่อมต่อเมื่อมีการค้นหาข้ามบริบท โดยที่ซูเปอร์โหนดเหล่านั้นมีหน้าที่ในการเป็นช่องทางเข้าออกสำหรับรองรับการสัญญาณการค้นหาที่ถูกส่งเข้ามา โดยตำแหน่งของซูเปอร์โหนดเหล่านั้นจะถูกเก็บลงใน DHT ซึ่งจะถูกเก็บคู่กันไปกับชื่อของบริบทปลายทางที่ซูเปอร์โหนดอาศัยอยู่



รูปที่ 0-10 การค้นหาโดยมีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท

รูปที่ 0-10 แสดงให้เห็นถึงการทำงานของการค้นหารูปแบบดังกล่าว โดยในระบบมีการเสนอโหนดซึ่งทำหน้าที่เป็นซูเปอร์โหนดขึ้นมาสามโหนด ได้แก่ โหนดหมายเลข 1 รับผิดชอบบริบท Calendar, โหนดหมายเลข 2 รับผิดชอบบริบท Person, และโหนดหมายเลข 5 รับผิดชอบบริบท Calculator ซึ่งโหนดทั้งสามทำหน้าที่รอรับการเชื่อมต่อจากระบบเพื่อส่งต่อสัญญาณไปยังกลุ่มของโหนดซึ่งอยู่ในบริบทที่ต้องการ

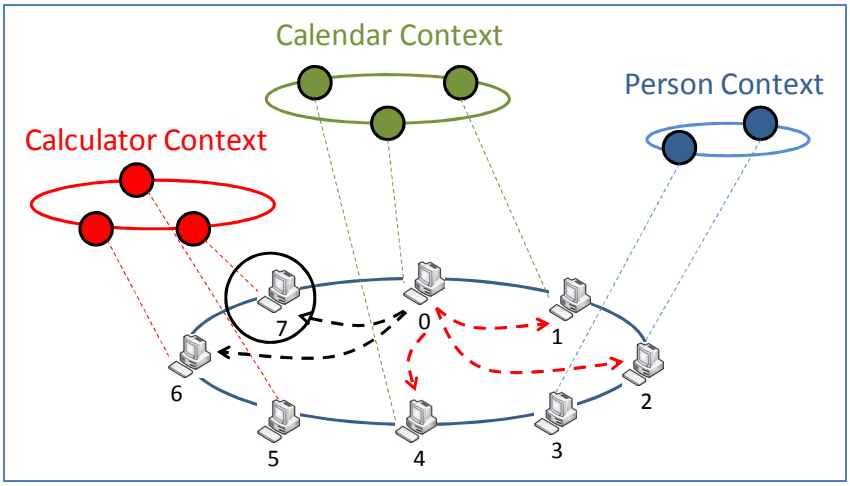
โดยสถานการณ์จำลองที่เกิดขึ้นคือโหนดหมายเลข 0 ซึ่งอยู่ในบริบท Calendar ต้องการทราบถึงข้อมูลบางอย่างที่อยู่ในบริบท Calculator แต่โหนดหมายเลข 0 ไม่อาจทราบได้ว่าต้องส่งสัญญาณการสืบค้น ไปยังโหนดใด ซึ่งในกรณีนี้โหนดหมายเลข 0 จะส่งสัญญาณร้องขอ (สัญญาณ Lookup จากในรูป) เพื่อค้นหาข้อมูลภายใน DHT โดยข้อมูลที่ได้ตอบกลับมาจะเป็นหมายเลขโหนดปลายทางที่เกี่ยวข้องกับบริบทของโหนดที่ต้องการค้นหาซึ่งในที่นี้จะเป็นโหนดหมายเลข 5 หลังจากนั้นโหนดหมายเลข 0 จะส่งสัญญาณการค้นหาต่อไปยังโหนดหมายเลข 5 โดยตรงเพื่อให้โหนดหมายเลข 5 กระจายสัญญาณดังกล่าวในกลุ่มของ Context-aware Chord เป้าหมายต่อไป

การค้นหาในรูปแบบนี้จะมีระยะเวลาสำหรับการค้นหาที่สั้น นั่นคือการค้นหาสามารถทำได้อย่างรวดเร็ว อย่างไรก็ตามการค้นหารูปแบบดังกล่าวจะมีปัญหาที่การเลือกซูเปอร์โหนดซึ่งโหนดที่ถูกเลือกเป็นซูเปอร์โหนดจะต้องรับภาระการทำงานที่หนักมาก ส่งผลให้การหนาแน่นของข้อมูลจะเกิดขึ้นที่บริเวณโหนดซึ่งทำหน้าที่เป็นซูเปอร์โหนดได้ และการค้นหาดังกล่าวยังมีปัญหาการออกจากกลุ่มของซูเปอร์โหนด ซึ่งเมื่อซูเปอร์โหนดออกจากระบบไปแล้วจะ

ส่งผลให้ระบบไม่สามารถทำงานได้อยู่ระยะหนึ่งจนกว่าระบบจะมีการเลือกโหนดใหม่ขึ้นมาทดแทน

วิธีการค้นหาแบบที่ 3. การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

กระบวนการค้นหาในแบบที่สามนี้จะเป็นการค้นหาคล้ายกับแบบที่ 1 นั่นคือโหนดทุกโหนดมีความเสมอภาคกันทั้งระบบ แต่ระบบดังกล่าวจะมีการปรับปรุงเพิ่มเติมในส่วนของการค้นหาให้เร็วขึ้นด้วยการเพิ่มตาราง Finger table ในส่วนของ Global Chord ขึ้นมาอีกตารางหนึ่ง โดยตารางดังกล่าวทำหน้าที่ในการค้นหาโหนดในทิศทางที่ตรงกันข้ามกับตาราง finger table อันแรกดังแสดงในรูปที่ 0-11



รูปที่ 0-11 การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

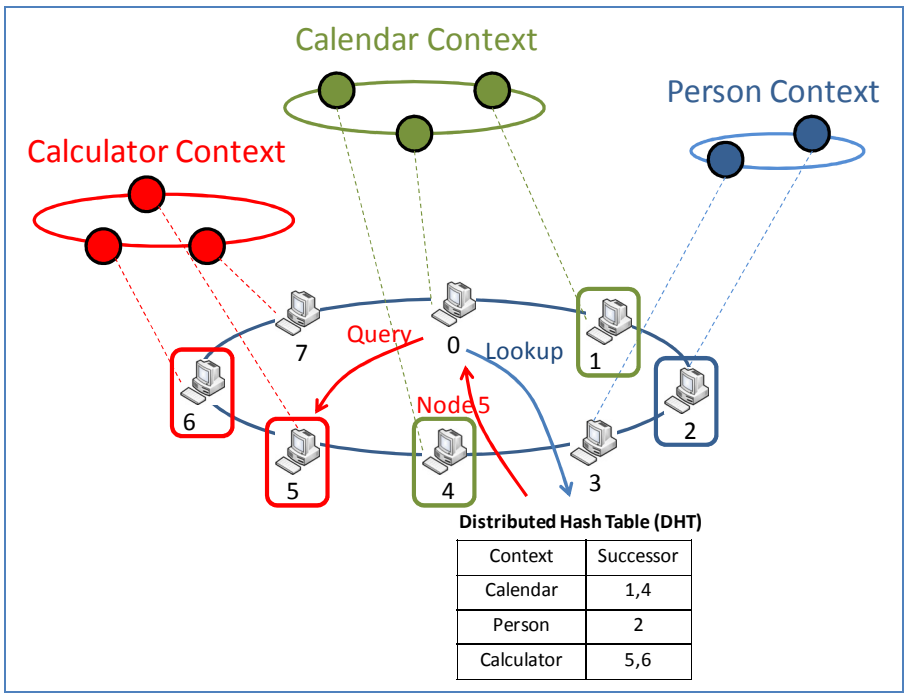
จากรูปที่ 0-11 แสดงสถานการณ์การค้นหาเช่นเดียวกับที่กล่าวไว้ในหัวข้อวิธีการค้นหาแบบที่ 1 ซึ่งจะเป็นการทำงานในส่วนของโหนดหมายเลข 0 ที่อยู่ในบริบท Calendar และต้องการค้นหาข้อมูลที่อยู่ภายในบริบทของ Calculator โหนดหมายเลข 0 จะทำการค้นหาภายใน finger table ของตนเองเพื่อตรวจสอบหาโหนดข้างเคียงที่มีบริบทตามที่ต้องการ ซึ่งในที่นี้ค่าของ finger table ธรรมดาจะเป็นการค้นหาในทิศทางตามเข็มนาฬิกา ซึ่งผลลัพธ์ที่ได้จะเป็นโหนดหมายเลข 1, 2, และ 4 ซึ่งจะเห็นว่าโหนดเหล่านั้นไม่มีการเก็บค่าของบริบทที่ต้องการเอาไว้ ดังนั้นการค้นหาแบบนี้จะมีส่วนช่วยในการเพิ่มความรวดเร็วการค้นหา โดยการค้นหาในทิศทางทวนเข็มนาฬิกาออกไปยังโหนดหมายเลข 7 และ 6 ซึ่งจากตัวอย่างโหนดหมายเลข 0 ค้นพบว่าโหนดทั้งสองตัวมีบริบทที่ต้องการอยู่ภายใน ซึ่งจากรูปที่ 0-11 โหนดหมายเลข 7 จะถูกเลือก

ขึ้นมาเพื่อส่งข้อมูลการสืบค้นออกไปเนื่องจากโหนดหมายเลข 7 เป็นโหนดที่อยู่ใกล้กับโหนดหมายเลข 0 มากที่สุด

จะเห็นว่าวิธีที่นำเสนอไปในรูปแบบนี้จะเป็นการปรับปรุงการค้นหาในรูปแบบที่ 1 ให้ดีขึ้น โดยวิธีดังกล่าวสามารถเพิ่มความเร็วได้เป็นสองเท่าจากความเร็วเดิมที่ได้เสนอไว้ในรูปแบบที่ 1 ซึ่งยังคงการกระจายความหนาแน่นของระบบออกไปยังโหนดปลายทางต่าง ๆ ได้เช่นเดิม แต่กระบวนการดังกล่าวมีข้อเสียสำคัญอยู่สองข้อนั่นคือ โหนดที่อยู่ในระบบจำเป็นจะต้องเก็บตาราง finger table เพิ่มขึ้นมาอีกหนึ่งตาราง ซึ่งตารางดังกล่าวทำหน้าที่สำหรับการค้นหาในทิศทางทวนเข็มนาฬิกา และความเร็วในการค้นหาจะต่ำมากเมื่อเทียบกับการค้นหาในรูปแบบที่ 2 หรือการใช้ซูเปอร์โหนด

วิธีการค้นหาแบบที่ 4. การค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวในแต่ละบริบท

สำหรับการค้นหาในรูปแบบนี้ ระบบจะกำหนดซูเปอร์โหนดขึ้นมาหลายตัวในแต่ละกลุ่ม Context-aware Chord เพื่อใช้สำหรับการกระจายความหนาแน่นของการเชื่อมต่อไปยังโหนดอื่น ๆ ซึ่งค่าของซูเปอร์โหนดที่กำหนดขึ้นมาจะถูกเก็บอยู่ใน DHT ในรูปแบบของกลุ่มของข้อมูล เช่น เซต (set), หรือ ลิส (list) แทนที่จะเก็บเป็นโหนดเพียงตัวเดียว โดยวิธีดังกล่าวมีข้อดีคือโหนดทั้งหมดที่ทำงานใน Global Chord ไม่จำเป็นจะต้องเปลี่ยนแปลงการทำงานมากนัก โดยสามารถใช้กระบวนการของ Chord ธรรมดาได้เลย แต่จะมีการแก้กระบวนการทำงานในระดับชั้นอื่น ซึ่งสามารถแก้ไขได้ง่ายกว่า และความเร็วที่ใช้สำหรับการค้นหาด้วยวิธีนี้ จะสูงมากเมื่อเทียบกับวิธีที่ 1 และ 3 อย่างไรก็ตามการกระจายตัวของข้อมูลอาจจะยังไม่ดีนักหากเทียบกับวิธีก่อนหน้า



รูปที่ 0-12 การค้นหาโดยกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท

รูปที่ 0-12 แสดงให้เห็นว่าในแต่ละกลุ่มบริบทจะมีการเลือกซูเปอร์โหนดขึ้นมามากกว่าหนึ่งตัว เช่น ในบริบท Calendar จะประกอบไปด้วยโหนดหมายเลข 1 และโหนดหมายเลข 4 และในบริบทของ Calculator ประกอบไปด้วยโหนดหมายเลข 5 และ 6 เป็นต้น สำหรับการค้นหาตามตัวอย่างที่แสดงไว้ในรูปที่ 0-12 เมื่อโหนดหมายเลข 0 ซึ่งอยู่ในกลุ่มบริบท Calendar ต้องการข้อมูลจากบริบท Calculator โหนดดังกล่าวจะส่งสัญญาณการค้นหาไปยัง DHT ซึ่งผลลัพธ์ที่ได้มาคือ กลุ่มของโหนดที่เป็นซูเปอร์โหนดสำหรับบริบทที่ต้องการ จากรูปที่ 0-12 โหนดหมายเลข 0 ซึ่งเป็นผู้ส่งสัญญาณค้นหาจะได้รับโหนดหมายเลข 5 และ 6 กลับมาเป็นผลลัพธ์ หลังจากนั้นโหนดหมายเลข 0 จะตัดสินใจเลือกโหนดขึ้นมาตัวหนึ่งแล้วส่งสัญญาณการสืบค้น ไปยังโหนดนั้นต่อไป

วิธีการค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวมีข้อดีคือ การค้นหาสามารถทำได้อย่างรวดเร็วมาก และความหนาแน่นของโหนดที่ต้องรับภาระหนักเพื่อทำตัวเป็นซูเปอร์โหนดจะถูกกระจายออกไป อย่างไรก็ตามวิธีดังกล่าวจะมีข้อเสียอยู่ที่โหนดที่ทำหน้าที่เก็บ DHT จะรับภาระค่อนข้างหนักเนื่องจากโหนดดังกล่าวต้องคอยส่งกลุ่มของบริบทกลับไปให้กับโหนดที่มาร้องขอ และต้องคอยรับคำสั่งเพื่อให้ปรับค่าของกลุ่มโหนดตามแต่ละบริบทอีกด้วย

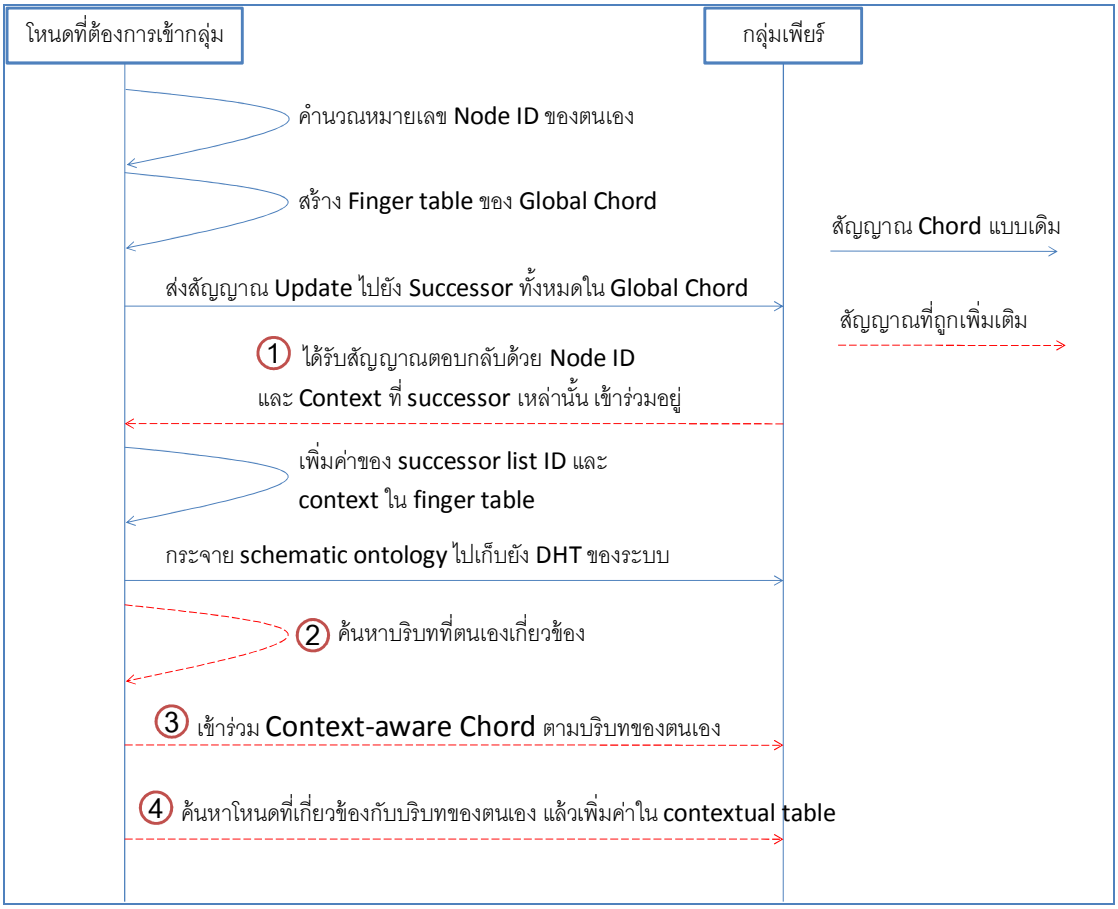
สรุป

ในบทนี้ได้กล่าวถึงการออกแบบระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์ซึ่งใช้ชื่อเรียกว่า MF-P2P โดยการทำงานของระบบนี้จะเป็นการให้แต่ละโหนดเก็บตาราง Finger table และ DHT ไว้มากกว่าหนึ่งตารางซึ่งตารางที่เก็บไว้จะสอดคล้องกับบริบทที่โหนดมีความสัมพันธ์ด้วย โดยจะส่งผลให้โหนดที่เข้าร่วมกลุ่ม MF-P2P เข้าร่วมกลุ่มบริบทเสมือนทำให้เพิ่มความรวดเร็วในการค้นหา และสามารถจัดการระบบค้นหาตามบริบทอย่างมีประสิทธิภาพอีกด้วย

รูปแบบโดยทั่วไปของระบบ MF-P2P โหนดที่ต้องการเข้าร่วมระบบจะต้องส่งสัญญาณร้องขอการเข้าร่วมมายังระบบเสียก่อน ซึ่งเมื่อระบบตอบรับการเข้าร่วมกลุ่มแล้วโหนดจะเข้าร่วมกลุ่ม MF-P2P ในชั้นของ Global Chord ก่อนเสมอ เพื่อทำการกระจาย schematic ontology ให้แก่ระบบได้รับรู้ หลังจากนั้นโหนดจึงเข้าร่วมกลุ่ม Context-aware Chord ต่อไป

สรุปการเข้าร่วมกลุ่ม MF-P2P

ในการเข้าร่วมกลุ่ม MF-P2P สามารถสรุปได้ตามแผนภาพดังแสดงในรูปที่ 0-13



รูปที่ 0-13 แผนภาพสรุปการเข้าร่วมกลุ่ม MF-P2P

รูปที่ 0-13 แสดงให้เห็นถึงการทำงานของระบบ MF-P2P ซึ่งเป็นการแสดงให้เห็นถึงสัญญาณที่ใช้สำหรับเชื่อมต่อระหว่างโหนดที่ต้องการเข้าร่วมระบบและกลุ่มเพียร์ โดยรูปที่ 0-13 ได้แยกเส้นทางการเชื่อมต่อเอาไว้ให้เห็นได้ชัด นั่นคือมีการแสดงการเปรียบเทียบระหว่างการส่งสัญญาณแบบคอร์ดธรรมดาและการส่งสัญญาณในรูปแบบที่ได้ออกแบบเพิ่มเติมขึ้นมาใหม่ (แสดงด้วยเส้นประสีแดง) โดยสัญญาณที่ได้สร้างขึ้นมาใหม่จำเป็นจะต้องมีการแก้ไขกระบวนการ

ของคอร์ดแบบเดิม เพื่อให้ระบบสามารถทำงานได้โดยมีการให้บริการในรูปแบบใหม่เกิดขึ้น ซึ่งสัญญาณที่สร้างขึ้นมาใหม่ประกอบด้วย

- สัญญาณการตอบกลับของ Successor ภายในกลุ่มเพียร์ ซึ่งสัญญาณดังกล่าว จะมีการเพิ่มข้อมูลบริบทของโหนด successor เอาไว้ด้วย
- สัญญาณการเข้าร่วมบริบทอื่น ๆ ซึ่งสัญญาณดังกล่าวเกิดขึ้นหลังจากที่โหนดเข้าร่วม Global Chord เรียบร้อยแล้ว โดยโหนดจะค้นหาข้อมูลบริบทที่ตนเองครอบครองอยู่ (instance ontology) เพื่อประเมินว่าโหนดตนเองควรเข้าร่วมกลุ่มบริบทกลุ่มไหนบ้าง
- โหนดทำการส่งสัญญาณไปยังกลุ่มเพียร์ปลายทางเพื่อ ขอเข้าร่วมกลุ่ม Context-aware Chord ตามบริบทที่ได้คำนวณไว้ก่อนหน้า ซึ่งหลังจากกลุ่มContext-aware Chord ปลายทางรับคำร้องขอแล้ว โหนดผู้ร้องขอจะเข้าร่วมกลุ่มโดยการสร้างตาราง finger table ใหม่ขึ้นมาเพื่อให้สอดคล้อง กับข้อมูลในกลุ่มบริบทใหม่ และทำการกระจาย instance ontology ของตนเองออกไปยัง DHT ของ Context-aware Chord
- หลังจากโหนดได้เข้าร่วมกลุ่ม Global Chord และ Context-aware Chord เรียบร้อยแล้ว โหนดจะค่อย ๆ ค้นหาว่าโหนดของตนเอง จำเป็นต้องมีการเชื่อมต่อกับโหนดปลายทางใดบ้าง จากนั้นโหนดจึงเริ่มค้นหาโหนดที่เกี่ยวข้อง เพื่อเก็บไว้ในตาราง contextual table เพื่อใช้สำหรับเป็นช่องทางลัด (short cut) ที่ช่วยสำหรับการค้นหาต่อไป

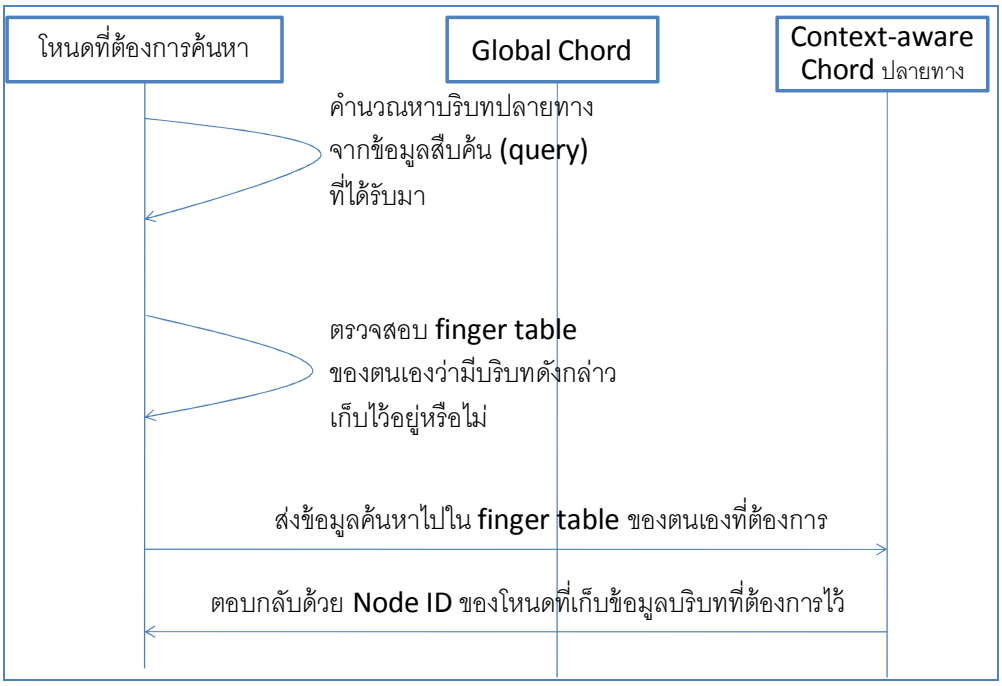
กระบวนการเข้าร่วมกลุ่ม MF-P2P สิ้นสุดลงหลังจากผ่านกระบวนการเข้าร่วม Context-aware Chord แล้ว ซึ่งโหนดจะพร้อมสำหรับการให้บริการรองรับคำร้องขอจากโหนดอื่น ๆ ในระบบ และในขณะเดียวกันสัญญาณซึ่งถูกส่งเพื่อค้นหาข้อมูลมาเติมเต็มตาราง finger table จะทำงานเป็นระบบเบื้องหลัง (background process) เพื่อช่วยปรับปรุงการค้นหาให้เร็วขึ้น

สรุปการค้นหาของ MF-P2P

การค้นหาในกลุ่ม MF-P2P สามารถแบ่งได้เป็นสองส่วนด้วยกัน ได้แก่ การค้นหาภายในกลุ่มบริบทตนเอง และการค้นหาข้ามบริบท โดยการค้นหาข้ามบริบทวิทยานิพนธ์ฉบับนี้ได้เสนอถึงความเป็นไปได้ในการนำไปใช้ 4 กรณี ได้แก่ การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน, การค้นหาแบบที่มีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท, การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง, และการค้นหาโดยกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท

สำหรับการค้นหาภายในกลุ่มบริบทของตนเองสามารถสรุปกระบวนการค้นหาได้ดังรูปที่ 0-14 ซึ่งการค้นหาเริ่มเมื่อโหนดได้รับข้อมูลการสืบค้นมาจากผู้ใช้ จากนั้นระบบจะ

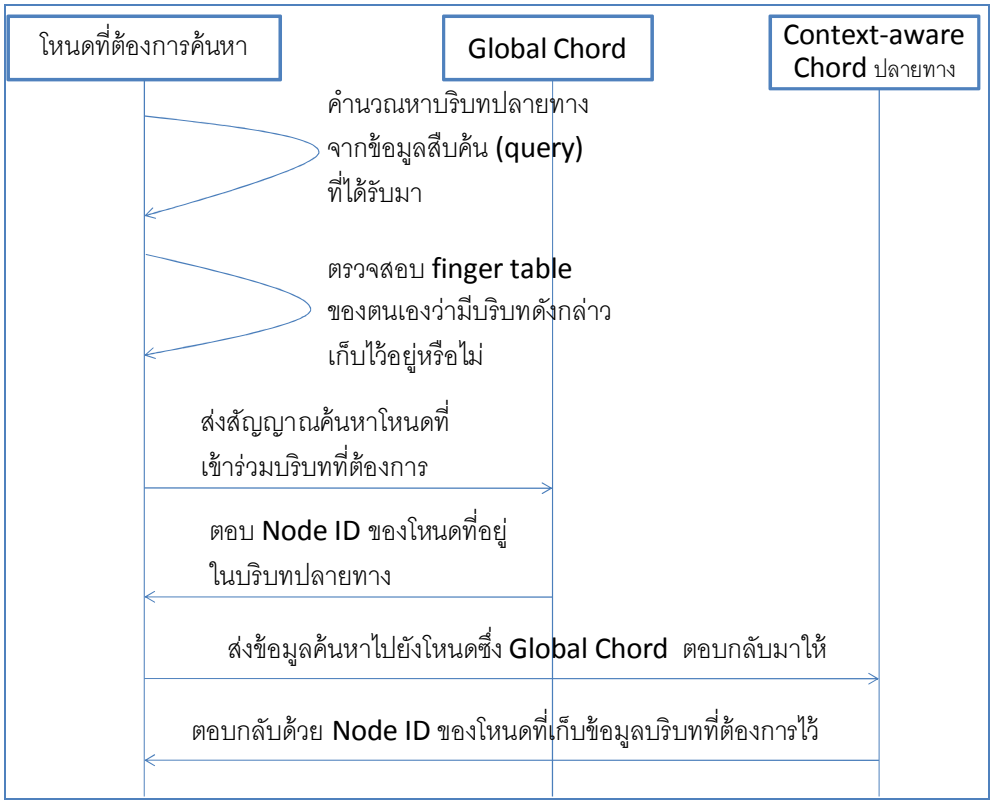
คำนวณเพื่อค้นหาบริบทปลายทางของข้อมูลสืบค้น โดยบริบรทปลายทางจะถูกค้นหาผ่านทางตาราง finger table ของตนเองว่ามีบริบทที่ต้องการอยู่ใน finger table หรือไม่ ซึ่งในการค้นหาที่แสดงในรูปที่ 0-14 เป็นการค้นหาที่โหนดค้นพบบริบทปลายทางใน finger table ของตนเอง นั่นหมายถึงโหนดได้เข้าร่วมบริบทปลายทางอยู่ก่อนแล้ว ดังนั้นโหนดจะส่งสัญญาณการสืบค้นไปค้นหาตามตาราง finger table ของบริบทปลายทาง หลังจากนั้นข้อมูลใน Context-aware Chord ปลายทางจะตอบกลับมาด้วยหมายเลขของโหนดปลายทางที่มีข้อมูลบริบทอยู่ หรือตอบกลับมาว่าหาไม่พบเมื่อการค้นหาไม่เจอข้อมูลในบริบทปลายทางที่ต้องการ



รูปที่ 0-14 แผนภาพแสดงการค้นหาเมื่อโหนดที่ต้องการค้นหาอยู่ในบริบทเดียวกันกับโหนดปลายทาง

สำหรับการค้นหาเมื่อโหนดต้นทางที่ต้องการค้นหาไม่ได้อยู่ในกลุ่ม Context-aware Chord เดียวกันกับโหนดปลายทางที่เก็บข้อมูลที่ต้องการไว้ การค้นหาจำเป็นจะต้องส่งสัญญาณไปยังกลุ่ม Global Chord เสียก่อนเพื่อค้นหาโหนดที่ใกล้ที่สุดซึ่งเข้าร่วมกลุ่มบริบท ที่ต้องการดังรูปที่ 0-15 แสดงให้เห็นถึงการค้นหาข้อมูลบริบทซึ่งโหนดที่ต้องการค้นหาอยู่ต่างบริบทกัน โดยขั้นตอนเริ่มต้นของการค้นหาจะเป็นเช่นเดียวกับขั้นตอนของการค้นหาในรูปแบบก่อนหน้า นั่นคือโหนดจะคำนวณหาบริบทปลายทางจากข้อมูลสืบค้น ซึ่งผลลัพธ์ของข้อมูลนี้จะเป็นข้อมูลของบริบทปลายทางที่ต้องการค้นหา หลังจากนั้นโหนดจะนำข้อมูลมาค้นหาใน finger table ของตนเองว่ามีการเก็บบริบทปลายทางไว้หรือไม่ ซึ่งจากรูปที่ 0-15 โหนดไม่ได้เข้าร่วมกลุ่ม

บริบทปลายทาง ทำให้โหนดต้นทางไม่สามารถค้นหาข้อมูลของบริบทปลายทางพบใน finger table ของตนเอง ดังนั้นโหนดที่ต้องการค้นหาจึงจำเป็นต้องส่งข้อมูลบริบทปลายทางไปบอกกับ Global Chord เพื่อให้กลุ่มเพียร์ค้นหาโหนดใด ๆ ที่เข้าร่วมกลุ่มบริบทปลายทางไว้ได้



รูปที่ 0-15 แผนภาพแสดงการค้นหาเมื่อโหนดที่ต้องการค้นหาอยู่ต่างบริบทกันกับโหนดปลายทาง

สำหรับการค้นหาโหนดที่เก็บบริบทปลายทางของ Global Chord ได้ถูกออกแบบไว้ด้วยกัน 4 กรณีนั่นคือ

วิธีการค้นหาแบบที่ 1. การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน
วิธีการค้นหาแบบที่ 2. การค้นหาแบบกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท
วิธีการค้นหาแบบที่ 3. การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง
วิธีการค้นหาแบบที่ 4. การค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวในแต่ละบริบท

หลังจากที่โหนดได้รับข้อมูลของโหนดที่มีบริบทปลายทางกลับมาจาก Global Chord แล้วโหนดที่ต้องการค้นหาจะส่งข้อมูลต่อไปให้ Node ID ที่ได้รับมา ซึ่งข้อมูลการสืบค้น จะถูกส่งเข้าไปภายในกลุ่มบริบทปลายทางหลังจากนั้นจะใช้กระบวนการค้นหาเช่นเดียวกับที่ใช้เมื่อโหนดที่ต้องการค้นหาอยู่ในบริบทเดียวกันกับโหนดปลายทาง ซึ่งหากการค้นหาสามารถทำได้อย่างสมบูรณ์โหนดปลายทางจะตอบข้อมูลของโหนดปลายทางได้แก่ ข้อมูลบริบทขณะนั้น และ Node ID เพื่อโหนดต้นทางจะสามารถร้องขอการค้นหาไปยังโหนดได้โดยตรงเพื่อร้องขอข้อมูลที่ต้องการ

การทดลองและการวิเคราะห์ผล

ในบทนี้จะเป็นการกล่าวถึงขั้นตอนการวิเคราะห์ผลการทำงานของระบบ MF-P2P ที่ได้กล่าวไปแล้วในบทที่ 4 ซึ่งบทนี้คณิตศาสตร์จะถูกนำมาใช้เป็นเครื่องมือในการเปรียบเทียบระหว่างการค้นหาในรูปแบบ ML-Chord กับ MF-P2P ที่ใช้การค้นหารูปแบบต่าง ๆ ในแง่ของ ความเร็วในการค้นหา, ความหนาแน่นของข้อมูลเมื่อเกิดการค้นหาขึ้น, และปริมาณหน่วยความจำที่ใช้เมื่อมีการสร้างระบบขึ้น สำหรับสาเหตุที่นำการค้นหาแบบ MF-P2P รูปแบบต่าง ๆมาเปรียบเทียบกับระบบล่วงรู้บริบทแบบ ML-Chord เนื่องจากระบบล่วงรู้บริบทรูปแบบ ML-Chord มีความเร็วที่ใกล้เคียงกับ MF-P2P เพราะระบบทั้งสองมีการเลือกใช้ระบบเพียร์ทูเพียร์แบบมีโครงสร้างเช่นเดียวกัน และระบบ ML-Chord ได้ถูกนำเสนอมาแล้วว่ามีประสิทธิภาพสูงกว่าระบบล่วงรู้บริบทแบบเพียร์ทูเพียร์รูปแบบอื่น ๆ

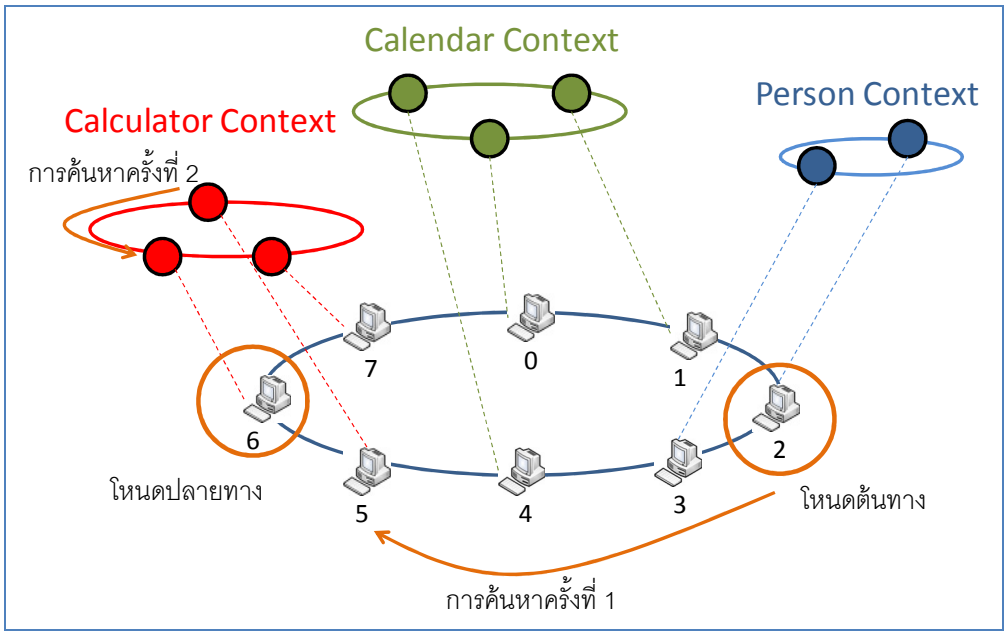
การเปรียบเทียบความเร็วในการค้นหา

หัวข้อนี้จะเป็นการกล่าวถึงการเปรียบเทียบความสามารถของระบบในแง่ของความเร็วในการค้นหา ซึ่งความเร็วในการค้นหาจะถูกคำนวณอยู่ในหน่วยของจำนวนโหนดที่ต้องผ่านระหว่างต้นทางไปยังปลายทาง ซึ่งในการทดลองนี้จะไม่คำนึงถึงเวลาจริง (เช่น เวลาในหน่วยของนาที, หรือวินาที เป็นต้น) และการวิเคราะห์เชิงคณิตศาสตร์นี้จะแสดงให้เห็นถึงความเร็วในการค้นหาโดยการนับจำนวน hop สำหรับกรณีที่แย่ที่สุด (worst case) เท่านั้น และในการค้นหาใน Chord จะถูกกำหนดให้มีความเร็วเท่ากับ log(N) โดยที่ N หมายถึงจำนวนโหนดทั้งหมดในระบบ ซึ่งข้อมูลดังกล่าวถูกพิสูจน์มาแล้วใน [31]

การวิเคราะห์เชิงคณิตศาสตร์ของระบบ MF-P2P

ความเร็วที่ใช้ในการค้นหาจากการนับจำนวน hop ในการค้นหาของระบบ MF-P2P ที่ใช้เป็นกรณีศึกษานี้เป็นกรณีที่แย่ที่สุดสำหรับการค้นหา (worst case) นั่นคือโหนดที่ต้องการค้นหาอยู่ต่างบริบทกับโหนดปลายทาง ซึ่งจะส่งผลให้ความเร็วใช้ในการค้นหาประกอบไปด้วยส่วนที่ใช้ในการหาสำคัญสองส่วนตามรูปที่ 0-1 โดยจำนวน hop ที่ใช้ในการค้นหาในช่วงแรกคือช่วงที่มีการค้นหาข้ามบริบท และจำนวน hop ส่วนที่สองมาจากการค้นหาภายในกลุ่มบริบทปลายทาง

จากรูปที่ 0-1 โหนดต้นทางเป็นโหนดที่อยู่ในบริบทของ Person และต้องการข้อมูลที่อยู่ในบริบท Calculator ดังนั้นความเร็วที่ใช้ในการค้นหาจะเริ่มวัดหลังจากที่โหนดต้นทางมองใน finger table ของตนเองและพบว่าไม่ได้เข้าร่วมบริบทเดียวกับโหนดปลายทาง จึงต้องทำการค้นหาโหนดที่ใกล้ที่สุดที่เป็นสมาชิกของบริบทปลายทาง (การค้นหาครั้งที่ 1) ซึ่งการค้นหาในครั้งนี้จะประกอบไปด้วยกระบวนการที่แตกต่างกัน 4 แบบดังที่ได้กล่าวไปแล้วในบทที่ 4 หัวข้อ 4.3.3.3 การค้นหาข้ามกลุ่มบริบท หลังจากนั้นจะเป็นการค้นหาครั้งที่ 2 ซึ่งเป็นการค้นหาภายในกลุ่มบริบทเดียวกัน



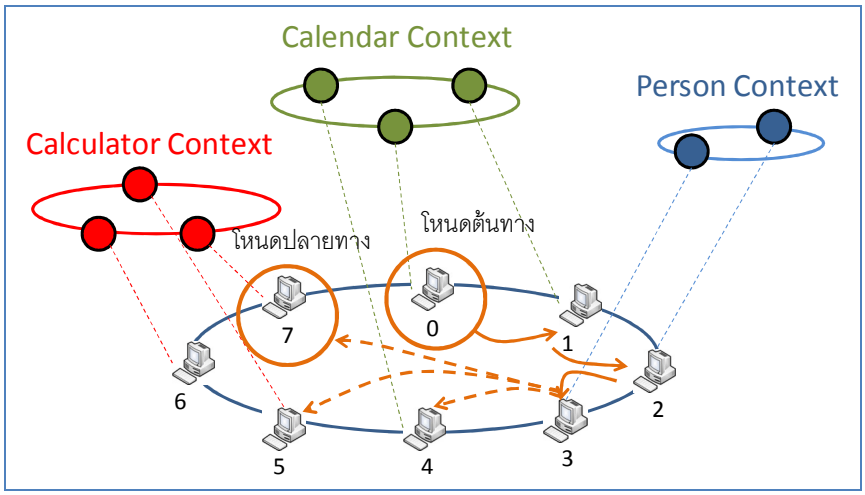
รูปที่ 0-1 ระยะทางที่ใช้ในการค้นหาของ MF-P2P

การวิเคราะห์ทางคณิตศาสตร์ของการค้นหาสำหรับ MF-P2P จะแยกเป็นการวิเคราะห์ออกเป็น 4 กรณีตามรูปแบบการค้นหาข้ามบริบทดังที่กล่าวไปแล้วในบทที่ 4 ดังต่อไปนี้

วิธีการค้นหาแบบที่ 1. การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน

การค้นหารูปแบบนี้โหนดจะค้นหาภายใน Finger table ของตนเองเสียก่อน ว่ามีบริบทที่ต้องการมีเก็บอยู่ใน finger table (โหนดข้างเคียง) หรือไม่ ซึ่งหากค้นหาบริบทที่ต้องการไม่พบในโหนดตนเอง โหนดต้นทางจะส่งสัญญาณคำสั่งสืบค้นไปยังโหนดข้างเคียงที่ใกล้ที่สุด (ในที่นี้คือโหนดตัวแรกใน finger table) ซึ่งโหนดข้างเคียงจะนำสัญญาณคำสั่งสืบค้นมาค้นหาภายใน finger table ของตนเองและส่งต่อไปเรื่อยๆ ดังนั้นจึงส่งผลให้ความเร็ว ในการค้นหาบริบท

ปลายทางใน Global Chord มีค่าเท่ากับ โดยที่ N หมายถึงจำนวนโหนด ทั้งหมดที่มีอยู่ในระบบ (สามารถอ่านเพิ่มเติมได้จากภาคผนวก ข) ซึ่งกรณีที่แย่ที่สุด (worst case) ที่เกิดขึ้นได้คือโหนดปลายทางเป็นโหนดที่อยู่ติดกับโหนดต้นทางทางซ้ายมือสุดดังรูปที่ 0-2 โดยโหนดต้นทางคือโหนดหมายเลข 0 เข้าร่วมอยู่ในบริบท Calendar ต้องการค้นหา โหนดที่เป็นสมาชิกของบริบทปลายทางซึ่งอาศัยอยู่ในบริบท Caculator ซึ่งกระบวนการค้นหาเริ่มต้นโดยโหนดหมายเลข 0 ค้นหาภายใน finger table ของตนเองและพบว่าภายใน finger table ไม่มีข้อมูลของบริบทปลายทางอยู่ ดังนั้นจึงส่งต่อไปให้โหนดข้างเคียงที่ใกล้ที่สุด นั่นคือโหนดหมายเลข 1 หลังจากนั้นโหนดหมายเลข 1 ค้นหา finger table ของตนเองเช่นเดียวกัน โดยการค้นหาจะดำเนินต่อไปเรื่อย ๆ จนกระทั่งถึงโหนดที่รู้จักบริบทปลายทางที่ต้องการภายใน finger table นั่นคือโหนดหมายเลข 3 ตามรูปที่ 0-2



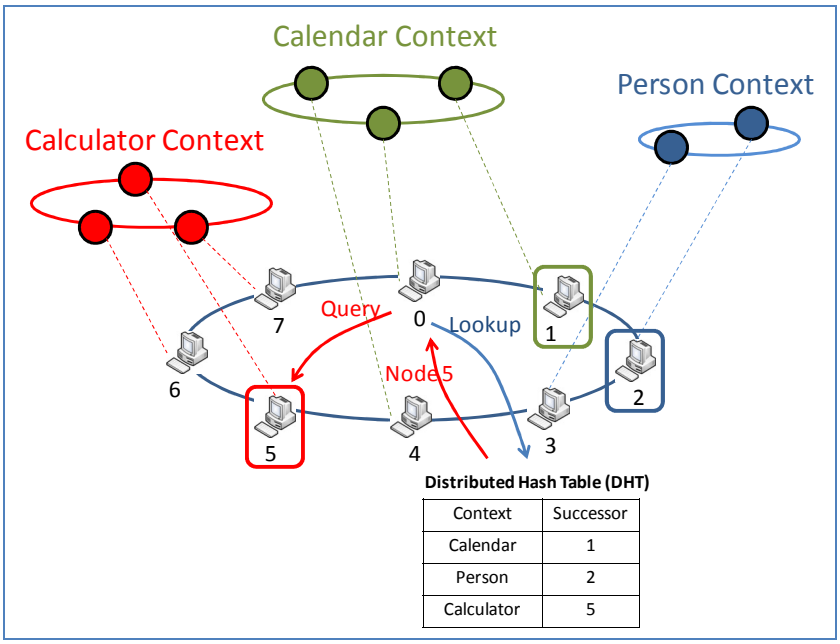
รูปที่ 0-2 ระยะทางที่ใช้ในการค้นหาข้ามบริบทเมื่อทุกโหนดทำงานเท่าเทียมกัน

วิธีการค้นหาแบบที่ 2. การค้นหาแบบกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท

สำหรับการค้นหารูปแบบนี้ กลุ่มบริบทแต่ละกลุ่มจะมีการกำหนดโหนดที่เป็นซูเปอร์โหนดขึ้นมาแล้วทำการกระจายตำแหน่งของโหนดนั้นไปยัง DHT โดยจะเป็นการกำหนดตำแหน่งของโหนดนั้นในรูปแบบของหมายเลขโหนดหรือ Node ID ซึ่งจะถูกจับคู่กับชื่อของบริบทที่เกิดขึ้นในระบบเพื่อให้โหนดอื่น ๆ ที่ต้องการค้นหาบริบทสามารถเข้ามามองหา โหนดได้ผ่านทาง DHT

จากรูปที่ 0-3 แสดงให้เห็นว่าโหนดต้นทางคือโหนดหมายเลข 0 ต้องการค้นหาข้อมูลที่เกี่ยวข้องกับบริบท Calculator จึงใช้การมองหาค่า Calculator ผ่านทาง DHT หลังจากนั้น

DHT จะตอบหมายเลข Node ID ของโหนดที่เข้าร่วมบริบทปลายทางที่ต้องการมาให้ ดังนั้นโหนดต้นทางจึงสามารถส่งข้อมูลการสืบค้น ไปยังบริบทของโหนดปลายทางได้โดยตรงโดยผ่านทางซูเปอร์โหนดซึ่งได้รับกลับมาจาก DHT ซึ่งจะส่งผลให้จำนวน hop ในการค้นหาโหนดปลายทางนั้นคือจำนวน hop ที่ใช้สำหรับการค้นหาภายใน DHT ของคอร์ดปกติเท่านั้น ดังนั้นความเร็วที่ใช้ในการค้นหาของกระบวนการนี้จะเป็น log(N) ซึ่งสามารถอ่านเพิ่มเติมได้จากภาคผนวก ข



รูปที่ 0-3 ระยะทางที่ใช้ในการค้นหาข้ามบริบทเมื่อมีการกำหนดซูเปอร์โหนด

วิธีการค้นหาแบบที่ 3. การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

สำหรับการค้นหาในรูปแบบนี้เป็นการค้นหาโดยที่แต่ละโหนดจะเก็บ Finger table ในชั้นของ Global Chord ออกเป็นสองทาง นั่นคือ Chord ปกติจะมีการเก็บ finger table แค่ทางเดียวเท่านั้น คือ finger table สำหรับโหนดที่อยู่ในทิศทางตามเข็มนาฬิกา ซึ่งกระบวนการการค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง นี้จะเป็นการให้โหนดแต่ละตัวมีการเก็บ finger table สำหรับการค้นหาเพิ่มเติมขึ้นมาอีกทิศทางหนึ่งนั่นคือการค้นหาในทิศทางทวนเข็มนาฬิกา ซึ่งจะส่งผลให้การค้นหาในกลุ่มบริบทมีความเร็วขึ้นอีกสองเท่า แต่ต้องแลกด้วยการเก็บข้อมูลที่เพิ่มมากขึ้น

ในส่วนของกระบวนการค้นหาที่เกิดขึ้น จะใช้กระบวนการค้นหาตามรูปแบบของการค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน แต่จะเป็นการค้นหาในสองทิศทางดังนั้น

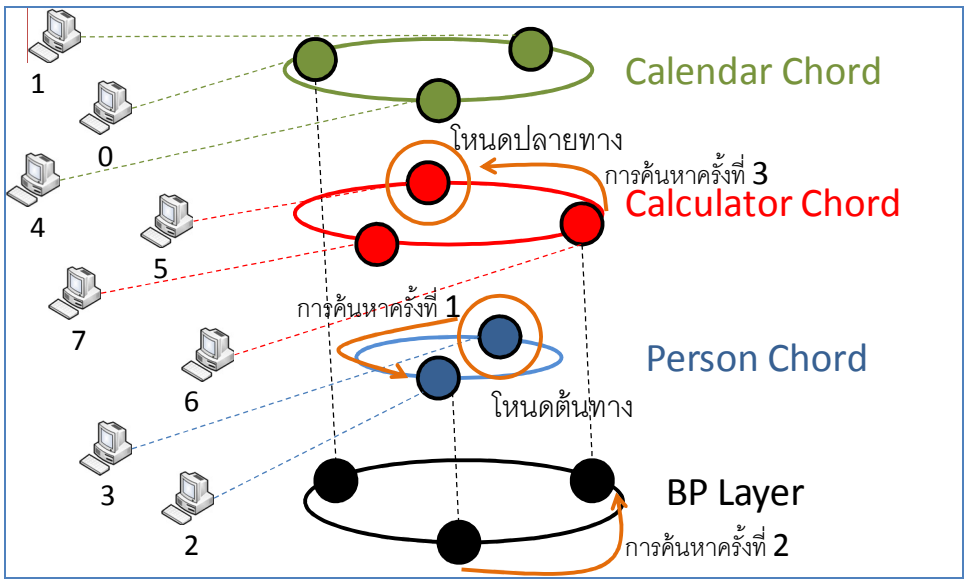
จำนวน hop ที่ใช้ในการเปรียบเทียบความเร็วในการค้นหาจึงน้อยลงมาครึ่งหนึ่ง ส่งผลให้ความเร็วในการค้นหาจากเดิม กลายเป็น นั่นเอง

วิธีการค้นหาแบบที่ 4. การค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวในแต่ละบริบท

รูปแบบการค้นหาในหัวข้อนี้จะเป็นการเพิ่มซูเปอร์โหนดให้แต่ละบริบท โดยการมองว่าซูเปอร์โหนดของแต่ละบริบทไม่จำเป็นต้องมีซูเปอร์โหนดเพียงแค่ตัวเดียว เพื่อลดความหนาแน่นของข้อมูลที่จะเกิดขึ้นที่ซูเปอร์โหนด การค้นหาโดยกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบทจึงถูกออกแบบขึ้น เพื่อให้ซูเปอร์โหนดแต่ละตัวช่วยกระจายความหนาแน่นของข้อมูลออกไป ซึ่งการกระทำดังกล่าวเป็นการเพิ่มข้อมูลภายใน DHT ของแต่ละบริบท ซึ่งข้อมูลที่เก็บเพิ่มขึ้นจะอยู่ในรูปของ กลุ่มของ Node ID คู่กันกับชื่อของ บริบทปลายทางที่โหนดแต่ละตัวเข้าร่วมอยู่ ดังนั้นความเร็วของการค้นหานี้จะมีค่าเท่ากับการค้นหาโดยกำหนดซูเปอร์โหนดสำหรับแต่ละบริบทนั่นคือ log(N) เพราะกระบวนการที่ใช้ในการค้นหาจะเป็นรูปแบบเดิม แต่กระบวนการดังกล่าวจะช่วยลดความหนาแน่นของข้อมูลที่เกิดขึ้นที่โหนดที่ต้องทำหน้าที่เป็นซูเปอร์โหนด

การวิเคราะห์เชิงคณิตศาสตร์ของระบบ ML-Chord

สำหรับการค้นหาโดยระบบ ML-Chord แสดงให้เห็นในรูปที่ 0-4 ซึ่งการค้นหาจะประกอบไปด้วยเส้นทางการค้นหาสามส่วน นั่นคือการค้นหาจะเริ่มหลังจากที่โหนดต้นทางรับรู้ว่าบริบทซึ่งตนเข้าร่วมอยู่กับบริบทปลายทางไม่ได้อยู่ในกลุ่มบริบทเดียวกัน ดังนั้นโหนดต้นทางจึงเริ่มต้นการค้นหาภายในกลุ่มของ Person Chord เพื่อทำการค้นหาโหนดที่เป็นซูเปอร์โหนดของบริบท Person ซึ่งหลังจากค้นพบโหนดแล้ว คำสั่งสืบค้น จะถูกส่งต่อไปยังกลุ่ม BP Layer ซึ่งเป็น Chord ที่เกิดขึ้นจากซูเปอร์โหนดเข้าร่วมกลุ่มกัน จึงเป็นการเริ่มการค้นหาครั้งที่ 2 นั่นคือการค้นหาซูเปอร์โหนดของบริบทปลายทาง เมื่อค้นพบแล้วซูเปอร์โหนดของบริบทปลายทางจะส่งต่อสัญญาณการสืบค้นไปให้กับโหนดปลายทางต่อไป



รูปที่ 0-4 ระยะทางที่ใช้ในการค้นหาของ ML-Chord

ดังนั้นจำนวน hop ที่ใช้ในการค้นหาสำหรับ ML-Chord จึงเป็น 2 log(n) + log(C) โดยที่ n หมายถึงจำนวนโหนดที่อยู่ในบริบทที่ต้องการ ซึ่งในการวิเคราะห์ครั้งนี้จะสมมติว่าแต่ละบริบทมีจำนวนโหนดเท่ากันทั้งหมด เพื่อให้ง่ายต่อการพิจารณา และ C หมายถึงจำนวนบริบทที่ใช้ในการทดลองแต่ละครั้ง ซึ่งค่าของ log(C) เป็นการแสดงให้เห็นถึงจำนวน hop ที่ใช้ในการค้นหาภายใน BP Layer เนื่องจากจำนวนโหนดที่อยู่ใน BP Layer มีจำนวนเท่ากับจำนวนของบริบทที่เกิดขึ้นในระบบ เช่น ในรูปที่ 0-4 จะเห็นว่ามีบริบทอยู่ทั้งหมด 3 บริบท (นั่นคือ Calendar, Caculator, และ Person) ดังนั้นจำนวนของโหนดที่อยู่ใน BP Layer จึงเป็นจำนวนเท่ากับ 3 โหนดไปด้วย

ผลการวิเคราะห์เมื่อกำหนดจำนวนเพียร์ในระบบคงที่

ในการวิเคราะห์ครั้งนี้จะกำหนดให้จำนวนเพียร์ในระบบคงที่อยู่ที่ 100,000 เพียร์ และให้ค่าของจำนวนบริบทเปลี่ยนไปเรื่อย ๆ ตั้งแต่ 500 จนกระทั่งถึง 3500 บริบท แล้วทำการเปรียบเทียบผลการคำนวณในรูปของกราฟ ซึ่งจากข้อกำหนดดังกล่าวสามารถกำหนดค่าของเวลาการค้นหาออกเป็นสมการทางคณิตศาสตร์ดังต่อไปนี้ (ให้ N เป็นจำนวนโหนดทั้งหมดในระบบซึ่งในที่นี้จะมีค่า 100,000 โหนดและ C เป็นจำนวนบริบทที่เพิ่มขึ้นเรื่อย ๆ ซึ่งจำนวน C ที่เพิ่มขึ้นจะส่งผลให้จำนวน table ที่เกิดขึ้นมากขึ้นตามไปด้วย)

- การค้นหาโดยใช้ ML-Chord จะได้สมการเป็นดังต่อไปนี้

$$2 \log \left(\frac{N}{C}\right) + \log(C) \qquad (0\text{-}1)$$

- การค้นหาโดยใช้ MF-P2P 1: แบบที่ทุกโหนดมีการทำงานเท่าเทียมกัน จะได้สมการเป็น

$$\log \left(\frac{N}{C}\right) + \frac{N}{2} \qquad (0\text{-}2)$$

- การค้นหาโดยใช้ MF-P2P 2: แบบที่มีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท จะได้สมการเป็น

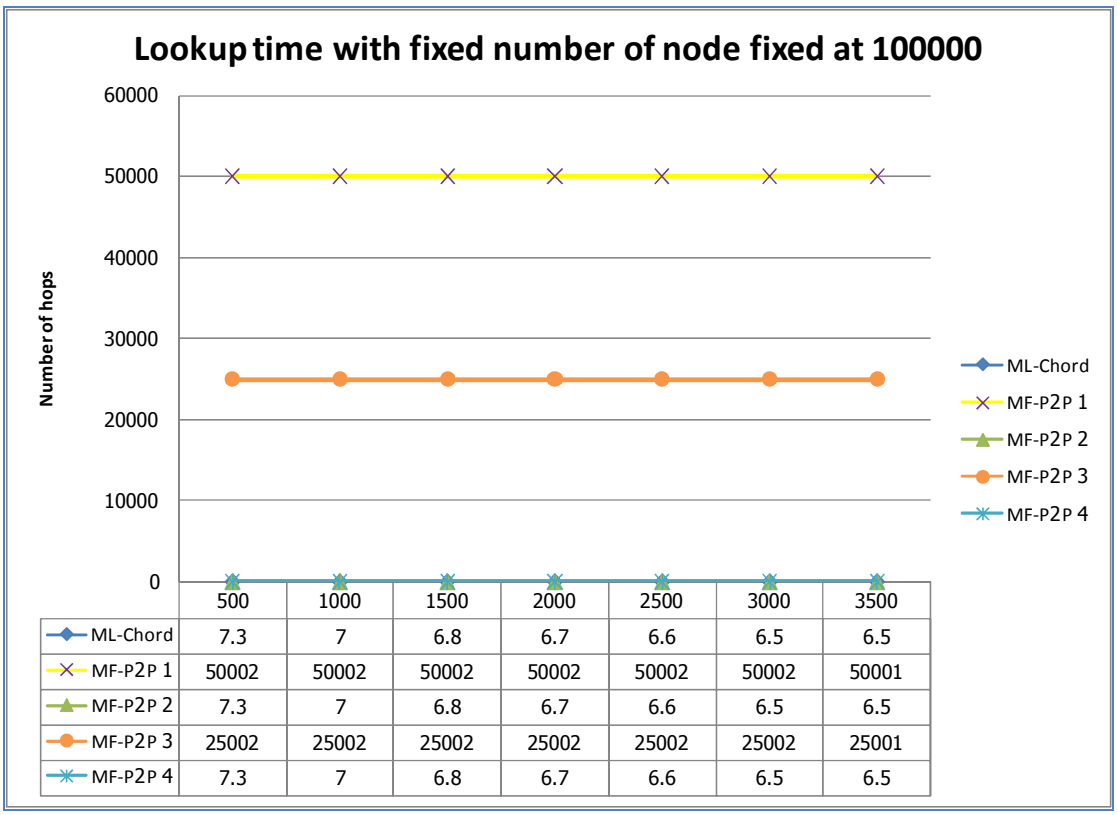
$$\log (\quad) + \log (N) \qquad (0\text{-}3)$$

- การค้นหาโดยใช้ MF-P2P 3: แบบที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง จะได้สมการเป็น

$$\log (\quad) + \qquad (0\text{-}4)$$

- การค้นหาโดยใช้ MF-P2P 4: แบบกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท จะได้ความเร็วเท่ากับแบบที่ MF-P2P 2 นั่นคือ

$$\log (\quad) + \log (N) \qquad (0\text{-}5)$$



	500	1000	1500	2000	2500	3000	3500
ML-Chord	7.3	7	6.8	6.7	6.6	6.5	6.5
MF-P2P 1	50002	50002	50002	50002	50002	50002	50001
MF-P2P 2	7.3	7	6.8	6.7	6.6	6.5	6.5
MF-P2P 3	25002	25002	25002	25002	25002	25002	25001
MF-P2P 4	7.3	7	6.8	6.7	6.6	6.5	6.5

รูปที่ 0-5 กราฟเปรียบเทียบความเร็วในการค้นหาเมื่อจำนวนโหนดในระบบมีค่าคงที่ โดยแกน X แสดงจำนวนบริบทที่เพิ่มขึ้น และแกน Y แสดงให้เห็นจำนวนโหนดที่ใช้ในการสืบค้น

จากรูปที่ 0-5 แสดงให้เห็นถึงกราฟที่ใช้เปรียบเทียบเมื่อเวลาผ่านไป โดยที่แกน X แสดงให้เห็นถึงจำนวนบริบท และแกน Y แสดงให้เห็นถึงจำนวนโหนดที่ข้อมูลต้องส่งจากต้นทางไปยังปลายทาง ซึ่งจำนวนโหนดนี้จะใช้เป็นหน่วยของความเร็วในการวิเคราะห์การทดลองครั้งนี้ ซึ่งจากกราฟจะประกอบไปด้วยข้อมูลของส่วนประกอบดังต่อไปนี้

- ML-Chord เป็นการแสดงให้เห็นถึงความเร็วในการค้นหาสำหรับกระบวนการ ML-Chord ซึ่งกราฟดังกล่าวจะถูกทับด้วยกราฟของ MF-P2P 2 เนื่องจากการค้นหาทั้งสองมีความเร็วที่เท่ากัน
- MF-P2P 1: เป็นความเร็วกระบวนการค้นหาแบบที่ทุกโหนดมีการทำงานเท่าเทียมกัน
- MF-P2P 2: และ MF-P2P 4: เป็นความเร็วในการค้นหาที่มีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท และแบบกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท เนื่องจากทั้งสองรูปแบบมีความเร็วเท่ากัน และจากรูปที่ 0-5 จะเห็นว่าความเร็วของวิธีการค้นหาดังกล่าวยังเท่ากับแบบ ML-Chord อีกด้วย

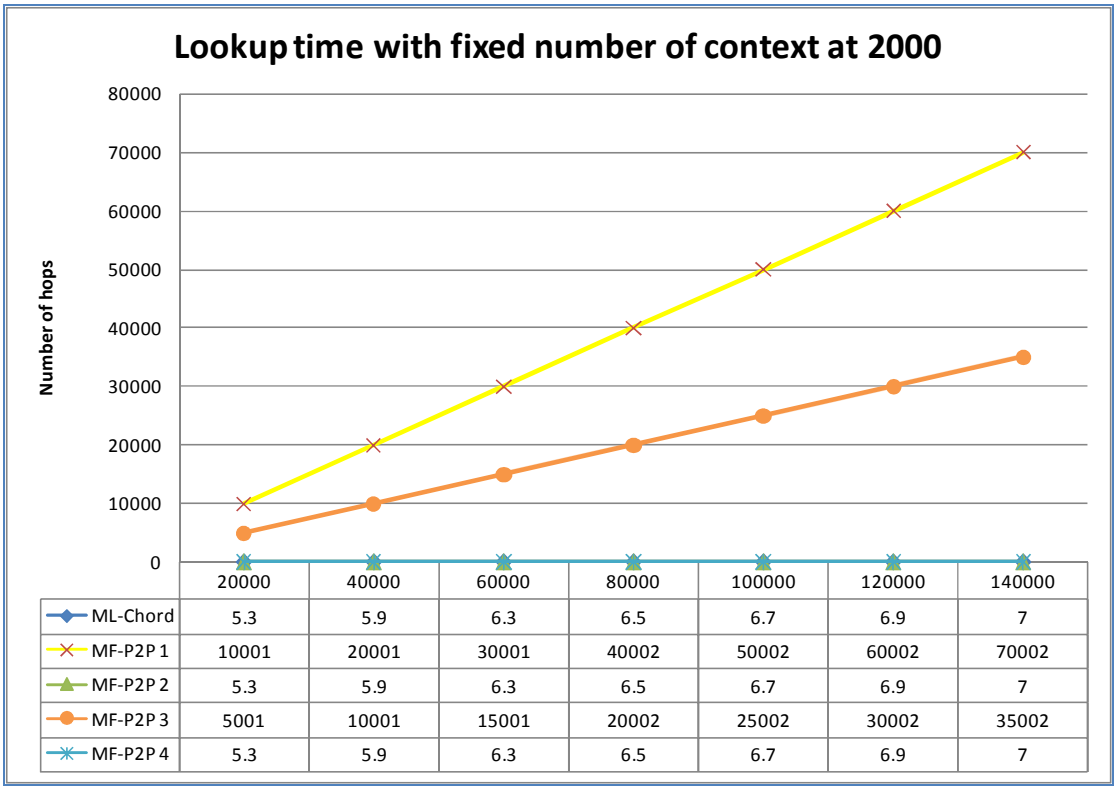
- MF-P2P 3: เป็นความเร็วในการค้นหาแบบที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

ซึ่งจากรูปที่ 0-5 จะเห็นว่ากระบวนการที่ใช้เวลามากที่สุดคือการค้นหาโดยที่ทุกโหนดมีการทำงานเท่าเทียมกัน (MF-P2P 1) เนื่องจากการค้นหาดังกล่าวจะเป็นการส่งสัญญาณไปยังโหนดข้างเคียงเรื่อย ๆ ทำให้ใช้เวลาในการค้นหามากที่สุด สำหรับกระบวนการค้นหาที่มีการใช้เวลาน้อยลงมาครึ่งหนึ่งคือการค้นหาแบบที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง (MF-P2P 3) ซึ่งกระบวนการดังกล่าวมีการสร้าง finger table ขึ้นมาเพิ่มอีกทิศทางหนึ่ง ทำให้ลดเวลาในการค้นหาไปได้ครึ่งหนึ่งอีกด้วย

สำหรับการค้นหาแบบ MF-P2P 3 และ ML-Chord จะให้เวลาในการค้นหา ที่ใกล้เคียงกัน เนื่องจากการค้นหาดังกล่าวมีการกำหนดโหนดหนึ่งขึ้นมาเป็นซูเปอร์โหนดทำหน้าที่ในการรับค่าการค้นหาทั้งหมดที่เกี่ยวข้องกับบริบทปลายทาง ทำให้การค้นหาแต่ละครั้งมีเป้าหมายที่แน่นอน คือโหนดที่ทำตัวเป็นซูเปอร์โหนดนั่นเอง ดังนั้นการค้นหาจึงใช้จำนวน hop เท่ากับการค้นหาภายในคอร์ดธรรมดา ซึ่งทำให้ความเร็วในการค้นหามีค่าสูงมากนั่นเอง

ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทให้คงที่

ในการวิเคราะห์ครั้งนี้จะเป็นการกำหนดให้จำนวนบริบทที่ระบบได้ถูกสร้างขึ้นมามีค่าคงที่อยู่ที่จำนวน 2000 บริบทและระบบจะทำการเพิ่มจำนวนโหนดขึ้นไปเรื่อย ๆ จากนั้นนำกระบวนการค้นหาแบบต่าง ๆ มาเปรียบเทียบกันโดยมีการเพิ่มจำนวนของเพียร์ในระบบขึ้นเรื่อย ๆ โดยเริ่มจาก 20,000 เพียร์และเพิ่มไปเรื่อย ๆ จนกระทั่งถึง 140,000 เพียร์ โดยสมการทางคณิตศาสตร์ที่ใช้ในการทดลองนี้จะเป็นสมการเดียวกันกับสมการที่ได้ใช้ในหัวข้อ 0



	20000	40000	60000	80000	100000	120000	140000
ML-Chord	5.3	5.9	6.3	6.5	6.7	6.9	7
MF-P2P 1	10001	20001	30001	40002	50002	60002	70002
MF-P2P 2	5.3	5.9	6.3	6.5	6.7	6.9	7
MF-P2P 3	5001	10001	15001	20002	25002	30002	35002
MF-P2P 4	5.3	5.9	6.3	6.5	6.7	6.9	7

รูปที่ 0-6 กราฟเปรียบเทียบความเร็วในการค้นหาเมื่อจำนวนบริบทในระบบมีค่าคงที่ โดยแกน X แสดงจำนวนโหนดที่เพิ่มขึ้น และแกน Y เป็นจำนวนโหนดที่ใช้ในการสืบค้น

จากรูปที่ 0-6 แสดงให้เห็นถึงกราฟที่ใช้ในการเปรียบเทียบความเร็วในการค้นหาเมื่อจำนวนบริบทมีค่าคงที่ และจำนวนของเพียร์ในระบบมีค่าเพิ่มขึ้นเรื่อย ๆ ซึ่งจากรูปกราฟประกอบไปด้วยการเปรียบเทียบกระบวนการ ML-Chord, MF-P2P 1, 2, 3, และ 4 ซึ่งความหมายของเส้นที่ใช้ในการเปรียบเทียบ จะเหมือนกับความหมายที่ได้กล่าวไว้แล้วในหัวข้อ 0 โดยผลการทดลองจะเห็นว่าความเร็วในการค้นหาของ MF-P2P 1 และ 3 มีแนวโน้ม ที่จะเพิ่มมากขึ้นเรื่อย ๆ เมื่อโหนดมีจำนวนเพิ่มมากขึ้น ซึ่งจะเห็นว่าเมื่อเทียบกับกระบวนการค้นหาแบบที่มีการใช้ซูเปอร์โหนด เช่น ML-Chord, MF-P2P 2, MF-P2P 4 แล้วจะเห็นว่าการค้นหาในรูปแบบที่ไม่มีการกำหนดซูเปอร์โหนดโดยตรงจะใช้เวลามากกว่าเป็นอย่างมาก เนื่องจากระบบไม่มีการเก็บตำแหน่งของโหนดที่มีบริบทปลายทางเอาไว้ และระบบต้องการหลีกเลี่ยงการใช้รูปแบบการบรอดคาสท์ ดังนั้นโหนดต้นทางจึงต้องส่งสัญญาณการสืบค้นไปยังโหนดข้างเคียงเรื่อย ๆ เพื่อค้นหาโหนดที่มีบริบทปลายทางที่ต้องการและอยู่ใกล้กับโหนดต้นทางมากที่สุด

สรุปผล

จากการวิเคราะห์ทางคณิตศาสตร์แสดงให้เห็นว่าระบบที่มีการใช้งานซูเปอร์โหนดทั้งหมด ได้แก่ ML-Chord, MF-P2P 2, MF-P2P 4 จะใช้เวลาในการค้นหาที่น้อยที่สุด สำหรับรูปแบบที่ให้โหนดค้นหาบริบทปลายทางโดยไม่ได้ใช้ซูเปอร์โหนดแต่จะพึ่งพาการค้นหาโหนดข้างเคียงไปเรื่อย ๆ (MF-P2P 1 และ 3) จะมีความเร็วในการค้นหาต่ำที่สุด และจำนวน hop ที่ใช้ยังมากกว่าวิธีอื่น ๆ มาก ตามรูปที่ 0-5และ รูปที่ 0-6

แต่อย่างไรก็ตามวิธีการดังกล่าวจะมีข้อดีอยู่ตรงที่ความหนาแน่นของข้อมูลที่ใช้มีความหนาแน่นน้อยมาก หากเทียบกับวิธีอื่น ๆ ซึ่งวิธีที่ใช้ซูเปอร์โหนดจะส่งผลให้ข้อมูลไปกระจุกตัวอยู่บริเวณที่เป็นซูเปอร์โหนดทำให้ระบบเครือข่ายบริเวณดังกล่าวต้องรับภาระหนัก ส่วนการค้นหาแบบบรอดคาสท์จะทำให้ระบบเครือข่ายทั้งระบบต้องรับภาระหนักมากเช่นเดียวกัน

การเปรียบเทียบความหนาแน่นของข้อมูล

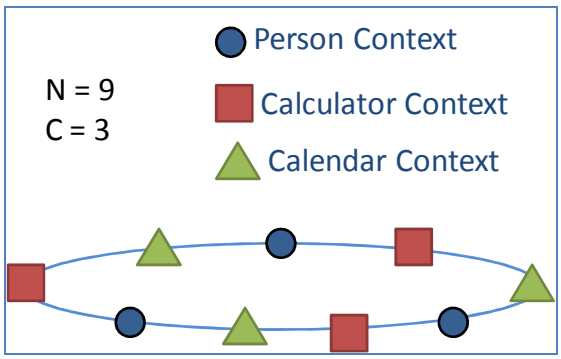
ความหนาแน่นของข้อมูลจะถูกแสดงให้เห็นถึงจำนวนข้อมูลที่วิ่งผ่านโหนดต่าง ๆ ในระบบซึ่งในที่นี้จะเป็นการมุ่งความสนใจไปยังโหนดที่จะต้องรับข้อมูลสืบค้นจำนวนมากที่สุดที่ไหลผ่านไป ซึ่งโหนดดังกล่าวคือโหนดที่จะเป็นโหนดที่ใช้สำหรับเชื่อมต่อบริบทไปยังปลายทางนั่นเอง สำหรับระบบที่ใช้การวิเคราะห์ครั้งนี้ จะสมมติให้โหนดทุกโหนดในระบบส่งสัญญาณการสืบค้นออกมาในอัตรา 1 คำสั่งการสืบค้นต่อ 1 วินาทีและคำสั่งที่ส่งออกมา มีเป้าหมายเข้าไปในบริบทเดียวกันทั้งหมด ซึ่งการวิเคราะห์สามารถแบ่งได้ดังต่อไปนี้

การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P

ในระบบ MF-P2P รูปแบบการค้นหาจะยังคงเดิมเช่นเดียวกับที่แสดงในรูปที่ 0-7 ซึ่งจะเห็นว่าการค้นหาประกอบไปด้วยการค้นหาสองครั้ง แต่อย่างไรก็ตามการวิเคราะห์นี้จะมุ่งเน้นถึงบริเวณที่ระบบเครือข่ายมีความหนาแน่นมากที่สุด นั่นคือบริเวณโหนดที่จะเป็นตัวส่งข้อมูลข้ามบริบท

วิธีการค้นหาแบบที่ 1. การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน

สำหรับการค้นหาในรูปแบบนี้จะสมมติสถานการณ์ว่าโหนดที่เข้าร่วมระบบมีการกระจายตัวอย่างเท่าเทียมกันทั้งหมดดังแสดงในรูปที่ 0-7



รูปที่ 0-7 ภาพแสดงการกระจายตัวของโหนดใน MF-P2P ซึ่งแสดงบริบทแตกต่างกันโดยใช้เครื่องหมายต่าง ๆ

จากรูปที่ 0-7 แสดงให้เห็นถึงโหนดที่อยู่ในระบบ MF-P2P ซึ่งโหนดแต่ละตัวจะเข้าร่วมบริบทที่แตกต่างกันไป และโหนดเหล่านั้นกระจายตัวอยู่ในระบบอย่างสมมาตรตามบริบทต่าง ๆ ซึ่งรูปที่แสดงประกอบไปด้วยโหนดทั้งหมดจำนวน 9 โหนดในระบบ และมีบริบททั้งหมด 3 บริบท ซึ่งแต่ละบริบทจะมีโหนดเป็นสมาชิกอยู่บริบทละ 3 โหนด

กำหนดให้บริบทปลายทางคือบริบท Calculator และทุกโหนดในระบบ ที่ไม่ได้อยู่ในบริบทดังกล่าวต้องการส่งสัญญาณการสืบค้นไปยังบริบทดังกล่าว โหนดเหล่านั้นจะส่งสัญญาณไปยังโหนดที่เป็นสมาชิกของบริบท Calculator ที่ใกล้ที่สุด ดังนั้นจำนวนของข้อมูลที่ไหลผ่านโหนดที่เป็นสมาชิกของบริบท Calculator จึงเป็น C – 1 เนื่องจากจำนวนข้อมูลที่วิ่งผ่านโหนดที่เป็นสมาชิกของบริบทปลายทาง คือจำนวนของโหนดที่อยู่ระหว่างโหนดที่เป็นสมาชิกของบริบทปลายทางดังกล่าว ซึ่งในที่นี้โหนดมีการกระจายตัวอย่างสมมาตรทำให้จำนวนโหนดที่จะส่งข้อมูลมีขนาดเท่ากับจำนวนโหนดอื่น ๆ ที่ไม่ได้อยู่ในบริบทปลายทางและอยู่ระหว่างโหนดจึงทำให้ได้ค่าเป็น C – 1

วิธีการค้นหาแบบที่ 2. การค้นหาแบบกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท

ความหนาแน่นที่เกิดขึ้นเมื่อโหนดทั้งระบบมีการค้นหาพร้อมกัน ไปยังบริบทปลายทางเดียวกันสำหรับระบบที่มีการกำหนดซูเปอร์โหนดสำหรับแต่ละบริบทนั้น จะมีค่าเท่ากับจำนวนของโหนดทั้งหมดในระบบที่ไม่ได้อยู่ในบริบท Calculator ส่งสัญญาณมาที่ซูเปอร์โหนดเพียงโหนดเดียว ดังนั้นจึงส่งผลให้ความหนาแน่นของสัญญาณบริเวณโหนดดังกล่าวกลายเป็น

วิธีการค้นหาแบบที่ 3. การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

สำหรับความหนาแน่นของระบบของรูปแบบนี้จะมีค่าเท่ากันกับระบบที่ทุกโหนดมีความเท่าเทียมกันแบบปกติ เนื่องจากโหนดที่เป็นทางผ่านเข้าไปสู่บริบทปลายทาง (หรือ บริบท Calculator) จะมีจำนวนเท่าเดิม จึงส่งผลให้ความหนาแน่นที่เกิดขึ้นบริเวณโหนดซึ่งเป็นทางเชื่อมต่อไปยังบริบทปลายทางมีค่าเป็น c - 1

วิธีการค้นหาแบบที่ 4. การค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวในแต่ละบริบท

ความหนาแน่นของระบบเมื่อมีการกำหนดซูเปอร์โหนดหลายตัว สามารถคำนวณได้โดย คำนวณต่อจากความหนาแน่นของระบบเมื่อมีการกำหนดซูเปอร์โหนดเพียงตัวเดียว นั่น

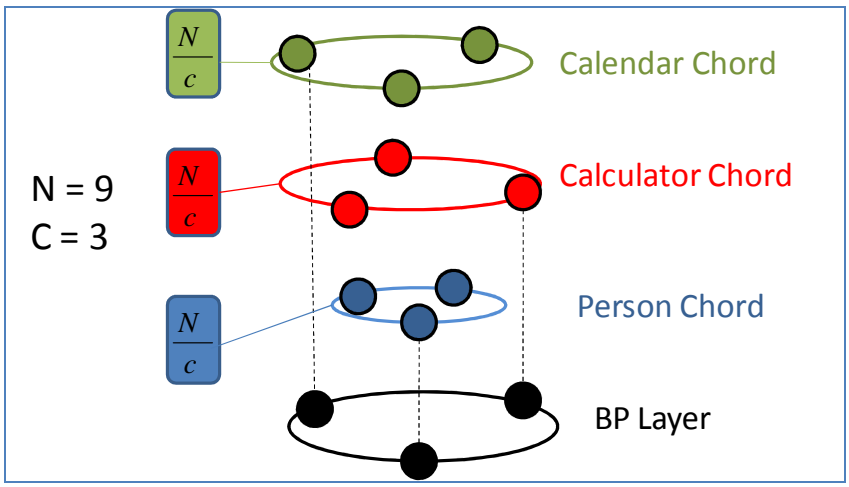
คือจำนวนโหนดทั้งหมดที่ไม่ได้อยู่ในบริบทที่ต้องการมีค่าเป็น ซึ่งแทนที่โหนดเหล่านั้นจะเข้าร่วมกลุ่มบริบทเดียวกันโดยผ่านทางซูเปอร์โหนดตัวเดียว ก็จะถูกกระจายออกไปให้

กลุ่มของซูเปอร์โหนดอย่างเท่าเทียมกัน ดังนั้นค่าความหนาแน่นของระบบจึงกลายเป็น
โดยที่ S หมายถึงจำนวนของซูเปอร์โหนดที่จะถูกเลือกในแต่ละบริบท

การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord

ในระบบ ML-Chord เมื่อแบ่งระบบให้มีจำนวนโหนดคงที่แล้ว จะสังเกตเห็นว่า

จำนวนของโหนดที่มีอยู่ในแต่ละบริบทจะมีจำนวนเท่ากับ (N หมายถึงจำนวนโหนดทั้งหมดในระบบ และ c หมายถึงจำนวนบริบททั้งหมด) ดังนั้นในช่วงเวลาหนึ่ง ๆ จำนวนข้อมูลที่วิ่งผ่านซูเปอร์โหนดเพื่อจะมีจำนวนเท่ากับจำนวนโหนดทั้งหมดในระบบ แต่ไม่รวมโหนดที่อยู่ในบริบทปลายทางเอง ดังนั้นจึงมีความหนาแน่นเกิดขึ้นเป็นจำนวน



รูปที่ 0-8 แผนภาพแสดงให้เห็นถึงสภาพแวดล้อมที่นำมาวิเคราะห์สำหรับ ML-Chord

จากรูปที่ 0-8 แสดงให้เห็นถึงสภาพแวดล้อมที่นำมาใช้วิเคราะห์ จะเห็นว่าในแต่ละบริบทจะมีจำนวนโหนดที่เท่ากันนั่นคือ ดังนั้นเมื่อทำการคำนวณถึงจำนวนทั้งหมดที่จะส่งสัญญาณผ่านทางซูเปอร์โหนดใน BP Layer แล้วจะเห็นว่าโหนดที่จะส่งสัญญาณการสืบค้นคือโหนดอื่น ๆ ที่ไม่ได้เข้าร่วมกลุ่มบริบทปลายทางดังนั้นจำนวนโหนดเหล่านั้นจึงมีค่า เท่ากับ

ผลการวิเคราะห์เมื่อกำหนดจำนวนเพียร์ในระบบคงที่

สำหรับการวิเคราะห์ในหัวข้อนี้เป็นการวิเคราะห์ความหนาแน่นของระบบเมื่อโหนดมีจำนวนคงที่ แต่มีการเปลี่ยนจำนวนบริบทให้เพิ่มขึ้นเรื่อย ๆ ซึ่งระบบจะกำหนดให้มีจำนวนโหนดในระบบอยู่ที่ 100,000 โหนดซึ่งถูกแทนค่าด้วยตัวอักษร N และให้มีจำนวนของบริบทเพิ่มขึ้นตั้งแต่ 10,000 บริบทไปจนกระทั่งถึง 100,000 บริบท และกำหนดให้ค่าจำนวนบริบทแทนด้วยตัวอักษร c และสำหรับกรณี MF-P2P 4 ที่มีการกำหนดจำนวนซูเปอร์โหนดมากกว่าหนึ่งตัวจะถูกแทนค่าจำนวนของซูเปอร์โหนดด้วย s ซึ่งในการทดลองนี้ให้มีค่าคงที่คือ 10 โดยความหนาแน่นของระบบสามารถสรุปเป็นสมการทางคณิตศาสตร์ได้ดังต่อไปนี้

- ML-Chord สำหรับความหนาแน่นของระบบ ML-Chord เมื่อมีการแทนค่าแล้วจะได้สมการสุดท้ายเป็น

(0-6)

- MF-P2P 1: แสดงถึงความหนาแน่นของระบบ MF-P2P แบบที่ทุกโหนดทำงานเท่าเทียมกันทั้งแบบทางเดียวซึ่งมีสมการทางคณิตศาสตร์เป็น

(0-7)

- MF-P2P 2: เป็นความหนาแน่นที่นำไปใช้กับระบบ MF-P2P ที่มีการกำหนดค่าของซูเปอร์โหนดในแต่ละบริบท ซึ่งค่าสมการที่ได้เป็น

(0-8)

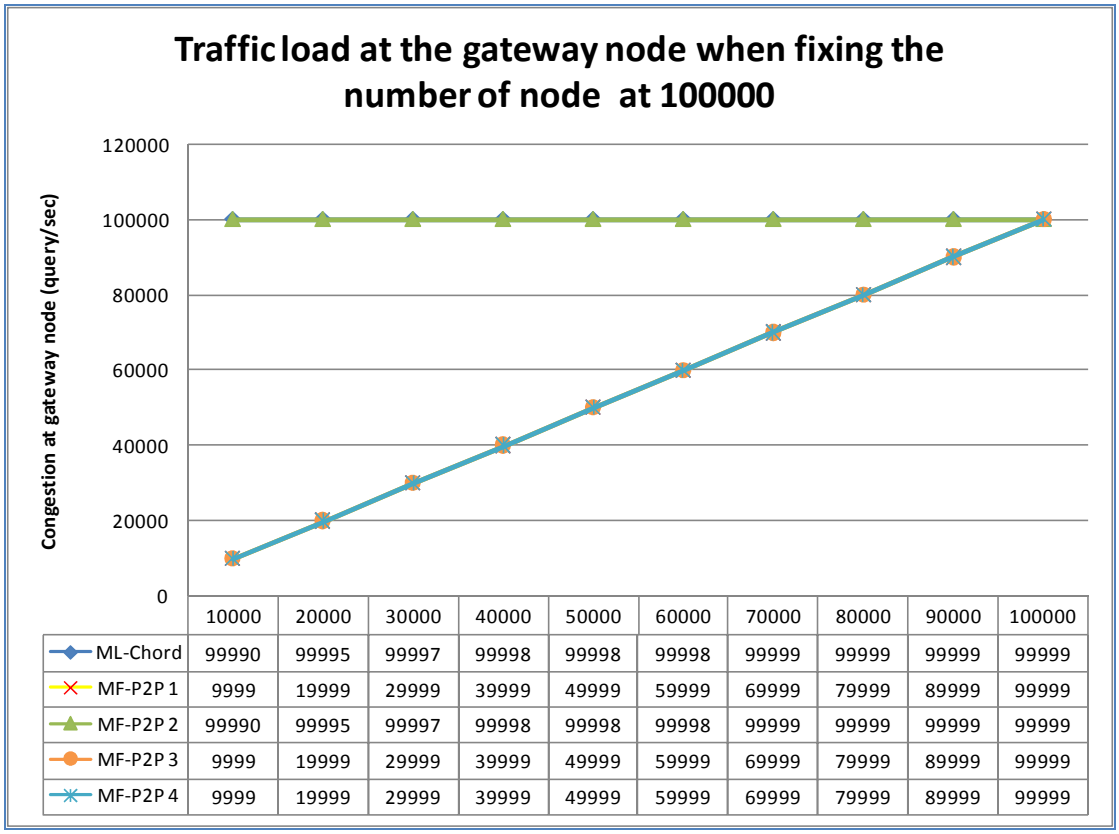
- MF-P2P 3: เป็นความหนาแน่นของระบบ MF-P2P เมื่อโหนดมีการทำงานเท่าเทียวมกันแบบสองทาง ซึ่งสมการของความหนาแน่นคือ

(0-9)

- MF-P2P 4: เป็นความหนาแน่นของระบบ MF-P2P แบบที่มีการกำหนดซูเปอร์โหนดหลายตัวในบริบทเดียวกันมีสมการทางคณิตศาสตร์เป็น

(0-10)

โดยในการทดลองนี้จะมีการกำหนดให้จำนวนของซูเปอร์โหนดที่เก็บไว้มีค่าเท่ากับ 10 ดังนั้นจึงได้สมการสุดท้ายเป็น



	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
ML-Chord	99990	99995	99997	99998	99998	99998	99999	99999	99999	99999
MF-P2P 1	9999	19999	29999	39999	49999	59999	69999	79999	89999	99999
MF-P2P 2	99990	99995	99997	99998	99998	99998	99999	99999	99999	99999
MF-P2P 3	9999	19999	29999	39999	49999	59999	69999	79999	89999	99999
MF-P2P 4	9999	19999	29999	39999	49999	59999	69999	79999	89999	99999

รูปที่ 0-9 กราฟแสดงความหนาแน่นของระบบเมื่อโหนดมีจำนวน 100,000 โหนด โดยแกน X แสดงถึงจำนวนบริบทที่เพิ่มขึ้นและแกน Y แสดงจำนวนข้อมูลที่ไหลผ่านซูเปอร์โหนด

จากรูปที่ 0-9 แสดงให้เห็นถึงกราฟที่แสดงถึงการเปรียบเทียบความหนาแน่นของระบบเมื่อโหนดมีจำนวนโหนดคงที่อยู่ที่ 100,000 โหนด ซึ่งกราฟดังกล่าวสามารถสรุปได้ดังต่อไปนี้

- ความหนาแน่นของ ML-Chord และ MF-P2P 2 จะเห็นว่ามีความหนาแน่นที่สูงที่สุด เนื่องจากการทำงานของระบบเหล่านั้นมีการสร้างซูเปอร์โหนดขึ้นมา เพื่อให้โหนดอื่น ๆ ที่ไม่ได้อยู่ในบริบทดังกล่าวเข้าร่วมบริบทดังกล่าวโดยตรง ดังนั้นความหนาแน่นจึงเริ่มต้นสูงกว่าระบบอื่น ๆ และจะสูงขึ้นเรื่อย ๆ เมื่อมีจำนวนของบริบทเพิ่มมากขึ้น เนื่องจากเมื่อมีจำนวนของบริบทเพิ่มขึ้นเรื่อย ๆ ในขณะที่จำนวนโหนดในระบบมีจำนวนเท่าเดิม จะ

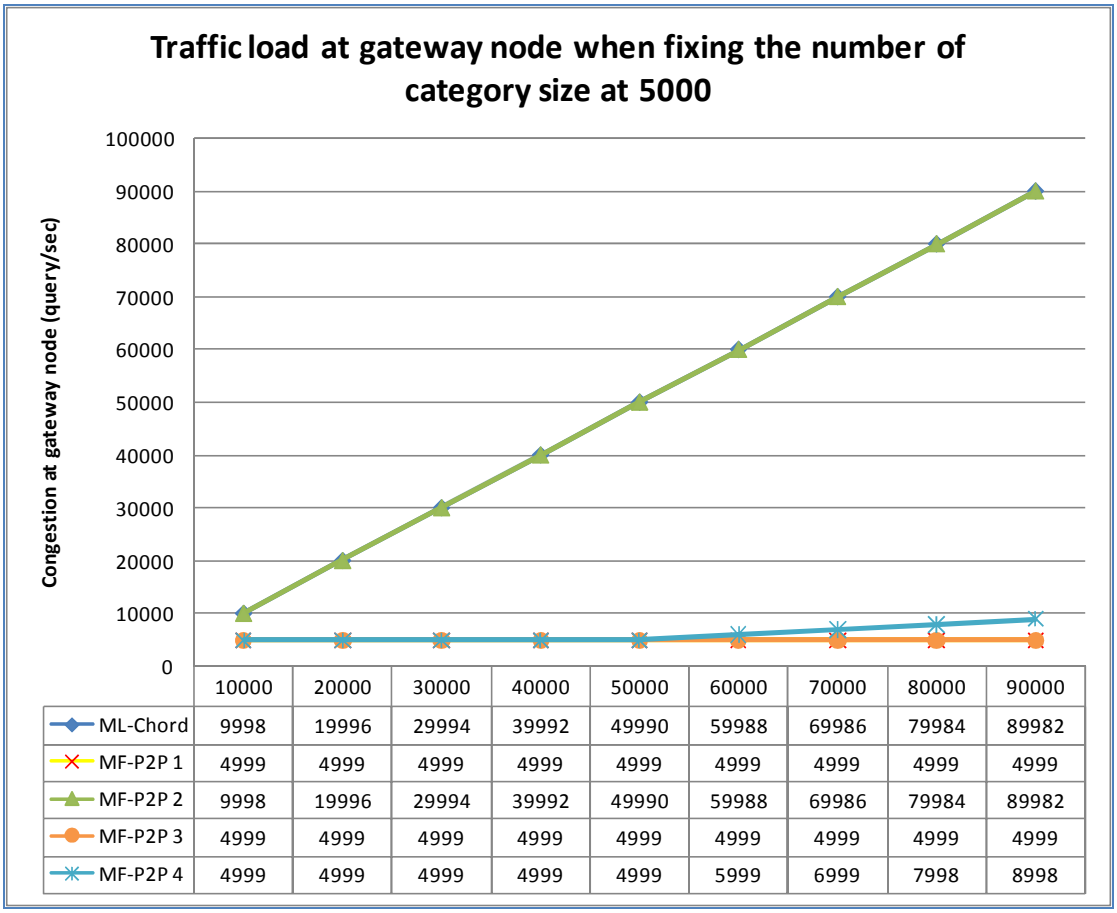
ส่งผลให้จำนวนโหนดที่อยู่ในบริบทปลายทางมีจำนวนน้อยลงเรื่อย ๆ และไปเพิ่มจำนวนให้กับโหนดอื่น ๆ ที่อยู่นอกบริบทไปด้วยเรื่อย ๆ กราฟจึงมีจำนวนเพิ่มขึ้นกว่าเดิม แต่กราฟมีแนวโน้มที่จะมีความหนาแน่นคงที่เมื่อมีจำนวนบริบทเพิ่มมากขึ้นเรื่อย ๆ เพราะค่าความหนาแน่นสูงสุดที่เกิดได้สำหรับกรณีนี้คือจำนวน N หรือขนาดเท่ากับจำนวนโหนดทั้งหมดในระบบนั่นเอง เนื่องจากสถานการณ์ที่ระบบจะมีความหนาแน่นสูงสุดคือ เมื่อโหนดทุกโหนดในระบบไม่มีโหนดใดเลยที่มีบริบทซ้ำกัน จะส่งผลให้เกิดความหนาแน่นสูงสุดนั่นคือ N – 1

- ความหนาแน่นของ MF-P2P 1 และ MF-P2P 3 เป็นความหนาแน่นที่เกิดจากระบบที่ทุกโหนดมีความหนาแน่นเท่าเทียมกันทั้งแบบทางเดียวและสองทาง ซึ่งจากกราฟจะเห็นว่ามีความหนาแน่นน้อยมากในตอนเริ่มต้น และมีแนวโน้มที่จะเพิ่มขึ้นเรื่อย ๆ เมื่อระบบมีการแบ่งเป็นบริบทย่อย ๆ เพิ่มขึ้น เนื่องจากจำนวนความหนาแน่นหมายถึงจำนวนโหนดที่อยู่ระหว่างบริบทปลายทางและบริบทต้นทาง ซึ่งหากยิ่งมีจำนวนบริบทเพิ่มขึ้นจะส่งผลให้ความหนาแน่นในการค้นหาเพิ่มขึ้น แต่การเพิ่มขึ้นของจำนวนโหนดจะไม่ส่งผลกระทบต่อความหนาแน่นเนื่องจากในการทดลองโหนดที่เพิ่มจำนวนมากขึ้นจะถูกกระจายอย่างสมมาตร เพื่อให้เข้าร่วมกลุ่มบริบทอื่น ๆ ได้อย่างเท่าเทียมกัน
- ความหนาแน่นของ MF-P2P 4 ในตามทฤษฎีควรมีจำนวนน้อยกว่า ML-Chord และ MF-P2P 2 อยู่เป็นจำนวน 10 เท่าเสมอ เนื่องจากการทดลองนี้ได้กำหนดให้จำนวนของระบบซูเปอร์โหนดที่แต่ละบริบทมีได้มีค่าเป็น 10 แต่อย่างไรก็ตามเมื่อนำไปทดลองแสดงผลออกมาดังรูปที่ 0-9 แล้วจะเห็นว่าจำนวนของความหนาแน่นกลับเพิ่มขึ้น ซึ่งปรากฏการณ์ดังกล่าวเกิดขึ้นเนื่องจาก จำนวนของโหนดที่มีอยู่ในแต่ละบริบท $(\frac{N}{C})$ มีค่าน้อยกว่า 10 โหนด เช่น ในช่องที่แกน X หรือจำนวนบริบทที่เพิ่มขึ้นมีค่าเท่ากับ 20,000 เมื่อจำนวนโหนดมีค่าคงที่คือ 100,000 โหนด จะส่งผลให้มีจำนวนโหนดอยู่ในแต่ละระบบเพียงแค่ $\frac{100,000}{20,000}$ หรือเท่ากับ 5 โหนดเท่านั้น ดังนั้นจึงนำจำนวนโหนดทั้งหมดที่มีค่าเท่ากับ 5 มาใช้ในการคำนวณแทนที่จะเป็น 10 จึงทำให้ความหนาแน่นของระบบมีความหนาแน่นน้อยเป็น 5 เท่าของเดิม และเป็นเช่นนี้ไปเรื่อย ๆ จึงทำให้กราฟมีค่าใกล้เคียงกันกับ MF-P2P 1 และ MF-P2P 3

ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทให้คงที่

ในหัวข้อนี้จะเป็นการวิเคราะห์ความหนาแน่นของระบบเช่นเดิม แต่จะเน้นการวิเคราะห์เมื่อโหนดมีจำนวนเพิ่มขึ้นดังนั้นสมการที่ใช้จึงเป็นสมการเดิมกับที่ใช้ในหัวข้อก่อนหน้า ส่วนจำนวนของบริบท จะถูกจำกัดให้คงที่โดยให้มีจำนวนบริบทอยู่ทั้งหมด 5,000 บริบท และ

จำนวนของโหนดที่เพิ่มขึ้นเรื่อย ๆ จะถูกแสดงให้เห็นในแกน X ของกราฟ สำหรับแกน Y จะเป็นแกนที่แสดงให้เห็นถึงจำนวนของข้อมูลที่ถูกส่งผ่านโหนดที่มีความหนาแน่นมากที่สุดในระบบ สำหรับสมการที่ใช้ในการวิเคราะห์ครั้งนี้ได้แก่ (ค่า N หมายถึงจำนวนของโหนดที่เพิ่มขึ้นเรื่อย ๆ)



	10000	20000	30000	40000	50000	60000	70000	80000	90000
ML-Chord	9998	19996	29994	39992	49990	59988	69986	79984	89982
MF-P2P 1	4999	4999	4999	4999	4999	4999	4999	4999	4999
MF-P2P 2	9998	19996	29994	39992	49990	59988	69986	79984	89982
MF-P2P 3	4999	4999	4999	4999	4999	4999	4999	4999	4999
MF-P2P 4	4999	4999	4999	4999	4999	5999	6999	7998	8998

รูปที่ 0-10 กราฟแสดงความหนาแน่นของระบบเมื่อมีจำนวนบริบทคงที่คือ 5,000 บริบท โดยแกน X แสดงจำนวนโหนดที่เพิ่มขึ้นและแกน Y แสดงจำนวนข้อมูลที่ไหลผ่านซูเปอร์โหนด

จากรูปที่ 0-10 แสดงให้เห็นถึงความหนาแน่นของระบบเมื่อมีการกำหนดให้จำนวนบริบทในระบบมีค่าคงที่ และเพิ่มจำนวนของโหนดขึ้นเรื่อย ๆ ซึ่งกราฟดังกล่าวสามารถสรุปความหนาแน่นของระบบย่อย ๆ ได้ดังต่อไปนี้

- ML-Chord และ MF-P2P 2 มีความหนาแน่นที่เกิดขึ้นบริเวณซูเปอร์โหนดเท่าเทียมกัน เนื่องจากทั้งสองวิธีมีการใช้งานระบบที่เหมือนกันนั่นคือการใช้งานซูเปอร์โหนดเพียงตัวเดียวในการรองรับคำสั่งสืบค้นจากระบบ ซึ่งจะเห็นว่าระบบมีความหนาแน่นมากขึ้นมากเมื่อเทียบกับการค้นหารูปแบบอื่น ๆ

- MF-P2P 1 และ MF-P2P 3 เป็นกระบวนการที่มีความหนาแน่นน้อยที่สุด ซึ่งจากค่าของกราฟจะถูกแสดงให้เห็นว่าค่าความหนาแน่นไม่เปลี่ยนไปเลยเมื่อโหนดมีจำนวนเพิ่มขึ้น เนื่องจากการทดลองครั้งนี้สมมติให้โหนดในระบบมีการจัดการให้ทุกโหนดจัดเรียงตนเองให้อยู่ในรูปแบบที่สมมาตร ดังนั้นเมื่อมีโหนดใหม่เข้ามาในระบบ โหนดเหล่านั้นจะถูกจัดเรียงให้อยู่ในรูปแบบเดียวกันกับโหนดที่เข้าร่วมในระบบก่อนหน้า ส่งผลให้ความหนาแน่นของระบบมีค่าเท่าเดิมเสมอ
- MF-P2P 4 ตามทฤษฎีแล้วจะมีค่าน้อยกว่า ML-Chord และ MF-P2P 2 อยู่ 10 เท่าเสมอ แต่ตามกราฟที่แสดงออกมาในช่วงแรกกลับมีการเปลี่ยนแปลงที่ไม่มากนัก เนื่องจากจำนวนโหนดที่มีอยู่ในแต่ละบริบท ($\frac{N}{C}$) มีจำนวนน้อยกว่า 10 โหนดนั่นเอง แต่เมื่อแกน X ของกราฟมีจำนวนโหนดเพิ่มขึ้นไปจนถึง 60,000 โหนดจะส่งผลให้มีจำนวนโหนดในแต่ละบริบทมีค่าเท่ากับ $\frac{60,000}{5,000}$ = 12 โหนด ดังนั้นจึงทำให้โหนดเริ่มมีความหนาแน่นสูงขึ้นเรื่อย ๆ

สรุปผล

สำหรับการค้นหาในรูปแบบของ MF-P2P ได้ถูกแบบออกเป็นสามส่วน เพื่อให้ง่ายต่อการวิเคราะห์นั่นคือ การค้นหาแบบกำหนดซูเปอร์โหนดเพียงโหนดเดียวจะถูก แทนที่ด้วย MF-P2P 2 ส่วนการค้นหาแบบเท่าเทียมกันแบบทางเดียวและสองทางส่งผลให้เกิดความหนาแน่นที่เหมือนกันซึ่งจะถูกแทนที่ด้วย MF-P2P 1, 3 และการค้นหาโดยมีการใช้ซูเปอร์โหนดหลายตัวในแต่ละบริบทจะถูกแทนด้วย MF-P2P 4

การทดลองด้านบนแสดงให้เห็นว่าระบบ ML-Chord และ MF-P2P 2 จะมีปัญหาเดียวกันในเรื่องของความหนาแน่นของข้อมูลที่ส่งผ่านซูเปอร์โหนด เนื่องจากเมื่อโหนดทั้งหมดในระบบต้องการค้นหาไปยังบริบทเดียวกันทั้งหมด จะส่งผลให้ทุกโหนดส่งสัญญาณการสืบค้นไปยังบริบทปลายทางพร้อมกัน ซึ่งทางผ่านเข้าไปยังบริบทปลายทางมีเพียงซูเปอร์โหนดเพียงตัวเดียวที่ทำหน้าที่ในการเก็บค่าของบริบทเอาไว้ ดังนั้นจึงส่งผลให้ค่าความหนาแน่นที่ซูเปอร์โหนดดังกล่าวมีค่าที่สูงมากเมื่อเทียบกับการค้นหาในรูปแบบอื่น

สำหรับการค้นหาในรูปแบบ MF-P2P 1 จะส่งผลให้เกิดความหนาแน่น น้อยที่สุดในกรณีที่ระบบมีการเพิ่มค่าจำนวนโหนดขึ้นเรื่อย ๆ โดยมีจำนวนบริบทคงเดิม และความหนาแน่นของระบบมีแนวโน้มจะเพิ่มสูงขึ้น เมื่อระบบมีการกำหนดให้จำนวนโหนดมีค่าคงเดิมในขณะที่มีการเปลี่ยนแปลงค่าของจำนวนบริบทไปเรื่อย ๆ เนื่องจากในการทดลองครั้งนี้จะสมมติให้ทุกโหนดเข้าร่วมบริบทอย่างสมมาตรกัน ซึ่งความหนาแน่นของระบบ MF-P2P 1 นี้ได้มาจากจำนวนโหนดที่อยู่ระหว่างโหนดซึ่งเป็นโหนดที่เข้าร่วมบริบทปลายทางอยู่ ดังที่แสดงให้เห็นในรูป

ที่ 0-7 ซึ่งหากมีการเพิ่มจำนวนบริบทให้มีความหลากหลายขึ้นเรื่อย ๆ จะส่งผลให้จำนวนโหนดที่อยู่ระหว่างโหนดปลายทางดังกล่าวเพิ่มขึ้นเรื่อย ๆ เช่นกัน และเมื่อมีการเพิ่มจำนวนของโหนดในระบบให้เพิ่มมากขึ้น โหนดที่เพิ่มขึ้นมาจะเข้าไปจัดเรียงตัวตามรูปแบบการจัดเรียงตัวของโหนดก่อนหน้า จึงส่งผลให้จำนวนโหนดที่อยู่ระหว่างโหนดที่เข้าร่วมบริบทปลายทางมีจำนวนเท่าเดิมเสมอ การค้นหาแบบ MF-P2P 1 จึงเป็นการค้นหาที่มีความหนาแน่นของ ระบบเกิดขึ้นน้อยที่สุด

ในส่วนของการค้นหาแบบ MF-P2P 4 ที่มีการกำหนดซูเปอร์โหนดหลายตัวสำหรับบริบทเดียวกัน จะมีความหนาแน่นของระบบเกิดขึ้นเป็นจำนวนน้อยกว่าการค้นหาแบบ ML-Chord และ MF-P2P 2 อยู่เป็นจำนวน s เท่า โดยที่ s หมายถึงจำนวนซูเปอร์โหนด ที่กำหนดสำหรับแต่ละบริบท นั่นคือการค้นหาแบบ MF-P2P 4 ยิ่งมีจำนวนของซูเปอร์โหนดมาก ก็จะส่งผลให้ระบบมีความหนาแน่นน้อยลงไปเรื่อย ๆ แต่จำนวนของหน่วยความจำที่ใช้งานก็จะมากขึ้นไปด้วยเช่นกัน ซึ่งการใช้หน่วยความจำสำหรับการค้นหาแบบต่าง ๆ จะแสดงให้เห็นในหัวข้อต่อไป

การเปรียบเทียบความเร็วในการค้นหา เมื่อมีการค้นหาแบบลูกโซ่

ในหัวข้อนี้จะเป็นการเปรียบเทียบความเร็วในการค้นหาเมื่อมีการค้นหาแบบลูกโซ่เกิดขึ้น โดยการค้นหาแบบลูกโซ่หมายถึงการที่เมื่อคำสั่งการสืบค้นถูกส่งไปยังโหนดปลายทางแล้ว โหนดปลายทางไม่สามารถตอบกลับข้อมูลดังกล่าวได้ทันทีหรือข้อมูลที่ตอบกลับไม่เพียงพอต่อความต้องการ ทำให้โหนดปลายทางจำเป็นต้องส่งข้อมูลร้องขอไปยังโหนดอื่น ๆ ข้างเคียงเพื่อขอข้อมูลเพิ่มเติม แล้วนำมาประมวลผลก่อนส่งไปแสดงผลให้แก่ผู้ใช้ หรืออาจเป็นการเก็บข้อมูลเพิ่มเติมเพื่อให้ผู้ใช้ได้รับข้อมูลข้างเคียงที่มากขึ้น ซึ่งการค้นหาแบบลูกโซ่มักจะเกิดขึ้นในระบบการค้นหาแบบล่วงรู้บริบท เนื่องจากในระบบที่ใช้โดยทั่วไป ส่วนมากแล้วโหนดปลายทางไม่สามารถตอบข้อมูลที่ต้องการกลับไปได้ในทันที แต่จำเป็นจะต้องมีการร้องขอไปยังโหนดข้างเคียงเพื่อขอข้อมูลเพิ่มเติม ก่อนจะประมวลผลตัวให้เหตุผล (Reasoner) แล้วจึงส่งข้อมูลตอบกลับไปยังบริบทปลายทางได้อย่างถูกต้อง

การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P

สำหรับการค้นหาแบบ MF-P2P จะมีการเก็บข้อมูลตาราง contextual table ขึ้นมาเป็นพิเศษ เพื่อให้ระบบสามารถรองรับการค้นหาแบบลูกโซ่ได้ ดังนั้นในการค้นหาแบบลูกโซ่ที่เพิ่มขึ้นมาแต่ละครั้งจะส่งผลให้เกิดเวลาในการค้นหาเพียง 1 โหนดเท่านั้นเนื่องจากข้อมูลของตำแหน่งของโหนดปลายทางจะถูกเก็บอยู่ภายใน contextual table ทั้งหมด โดยในการทดลอง

นี้จะกำหนดให้จำนวนครั้งของการค้นหาแบบลูกโซ่แทนด้วยตัวอักษร ch จึงส่งผลให้ได้รับสมการที่ใช้สำหรับการค้นหาดังต่อไปนี้

วิธีการค้นหาแบบที่ 1. การค้นหาโดยที่ทุกโหนดในระบบทำงานเท่าเทียมกัน

$$\log(\ ) + \quad + ch \qquad (0\text{-}11)$$

วิธีการค้นหาแบบที่ 2. การค้นหาแบบกำหนดซูเปอร์โหนดสำหรับแต่ละบริบท

$$\log(\ ) + \log(N) + ch \qquad (0\text{-}12)$$

วิธีการค้นหาแบบที่ 3. การค้นหาโดยที่ทุกโหนดมีความเท่าเทียมกันแบบสองทาง

$$(\ ) + \quad + ch \qquad (0\text{-}13)$$

วิธีการค้นหาแบบที่ 4. การค้นหาแบบกำหนดซูเปอร์โหนดหลายตัวในแต่ละบริบท

$$\log(\ ) + \log(N) + ch \qquad (0\text{-}14)$$

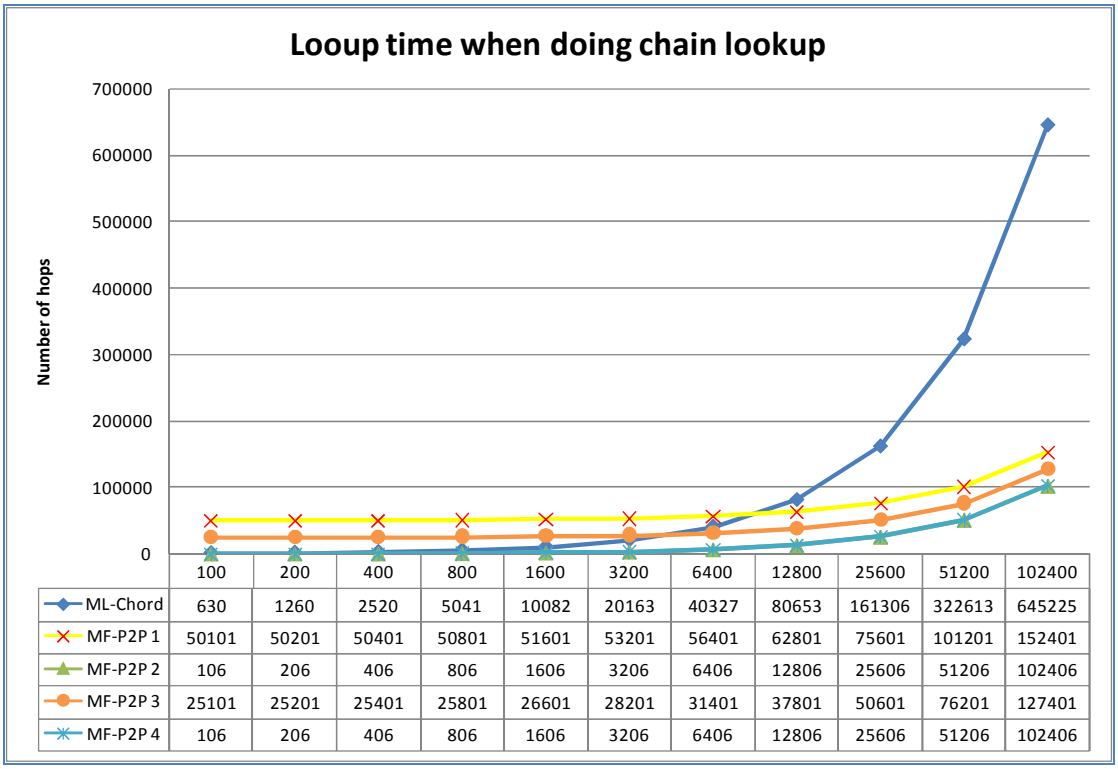
การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord

ในการคำนวณความเร็วสำหรับระบบ ML-Chord เมื่อมีการค้นหาแบบลูกโซ่ ระบบจะใช้ความเร็วเพิ่มขึ้นเป็นจำนวนเท่าตัว สำหรับการค้นหาแต่ละครั้ง เนื่องจากระบบ ML-Chord ไม่มีการเก็บค่าของ contextual table จึงทำให้การค้นหาต้องกระทำใหม่ทุกครั้งที่มีการ ส่งคำร้องขอคำสั่งสืบค้นมา ดังนั้นความเร็วในการค้นหาสำหรับระบบ ML-Chord จึงมีค่าเท่ากับ

$$(\log(\ ) + \ ) * ch \qquad (0\text{-}15)$$

ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทและจำนวนโหนดคงที่

จากรูปที่ 0-11 แสดงความเร็วในการค้นหาโดยที่มีการกำหนดจำนวนบริบทคงที่อยู่ที่ 5,000 บริบท และจำนวนโหนดในระบบมีค่าคงที่คือ 100,000 โหนด โดยในรูปแกน X ใช้สำหรับแสดงจำนวนครั้งของการค้นหาแบบลูกโซ่ ส่วนแกน Y เป็นการแสดงให้เห็นถึงความเร็วที่ใช้ในการค้นหาในระบบดังกล่าว



	100	200	400	800	1600	3200	6400	12800	25600	51200	102400
ML-Chord	630	1260	2520	5041	10082	20163	40327	80653	161306	322613	645225
MF-P2P 1	50101	50201	50401	50801	51601	53201	56401	62801	75601	101201	152401
MF-P2P 2	106	206	406	806	1606	3206	6406	12806	25606	51206	102406
MF-P2P 3	25101	25201	25401	25801	26601	28201	31401	37801	50601	76201	127401
MF-P2P 4	106	206	406	806	1606	3206	6406	12806	25606	51206	102406

รูปที่ 0-11 กราฟแสดงความเร็วในการค้นหาเมื่อมีการค้นหาแบบลูกโซ่ โดยจำนวนโหนดคงที่คือ 100,000 โหนด และจำนวนบริบทคงที่ 5,000 บริบท แต่เพิ่มจำนวนครั้งของการหาแบบลูกโซ่ขึ้นตามข้อมูลในแกน X ส่วนความเร็วในการหาจะแสดงในแกน Y

จากกราฟที่แสดงให้เห็นในรูปที่ 0-11 สามารถสรุปตามข้อมูลในแต่ละกราฟได้ดังต่อไปนี้

- การค้นหาแบบ ML-Chord มีความเร็วในการค้นหาในตอนแรกน้อยมาก แต่เวลาจะเพิ่มขึ้นเป็นจำนวนมากเช่นกันเมื่อเพิ่มจำนวนการค้นหาแบบลูกโซ่ไปเรื่อย ๆ เนื่องจากการค้นหาจะต้องกระทำซ้ำกระบวนการเดิมอยู่เสมอ

- MF-P2P 1 และ MF-P2P 3 เมื่อเริ่มการค้นหาจะมีความเร็วที่ช้ากว่าแบบอื่น ๆ แต่เมื่อมีการค้นหาแบบลูกโซ่มากขึ้นจะมีความเร็วที่สูงกว่า ML-Chord ในที่สุด
- MF-P2P 2 และ MF-P2P 4 แม้ว่าจะเริ่มการทำงานด้วยความเร็วที่ใกล้เคียงกันกับ ML-Chord (ซึ่งมีความเร็วมากกว่า ML-Chord) แต่เมื่อจำนวนของการค้นหาแบบลูกโซ่เพิ่มขึ้นเรื่อย ๆ จำนวนเวลาที่เพิ่มขึ้น กลับน้อยกว่าเวลาที่ ML-Chord ใช้ในการค้นหา

สรุปผลการทดลอง

รูปแบบลูกโซ่แสดงให้เห็นถึง การเปรียบเทียบระบบที่ใช้ในการค้นหา ซึ่งชี้ให้เห็นอย่างชัดเจนว่าระบบที่มีการใช้งาน Contextual table (MF-P2P 1, 2, 3 และ 4) จะมีการใช้เวลาที่น้อยกว่าอย่างเห็นได้ชัด เมื่อมีจำนวนครั้งของการค้นหาแบบลูกโซ่เพิ่มมากขึ้นเรื่อย ๆ

การเปรียบเทียบความหนาแน่นในการค้นหา เมื่อมีการค้นหาแบบลูกโซ่

หัวข้อนี้จะกล่าวถึงการคำนวณหาค่าความหนาแน่นของระบบ เมื่อมีการค้นหาแบบลูกโซ่เกิดขึ้น โดยระบบที่นำมาใช้ในการวิเคราะห์ครั้งนี้จะเป็นระบบที่มีจำนวนของบริบทคงที่ คือ 5,000 บริบท และจำนวนของโหนดในระบบคงที่คือจำนวน 100,000 โหนด ซึ่งค่าที่จะเปลี่ยนแปลงในระบบคือจำนวนของการค้นหาแบบลูกโซ่ ซึ่งจะมีจำนวนเพิ่มขึ้นเรื่อย ๆ โดยผลการทดลองจะบอกถึงจำนวนข้อมูลที่ไหลผ่านโหนดที่จะต้องรับภาระหนักที่สุดในระบบ (ในที่นี้หมายถึง ซูเปอร์โหนดสำหรับ ML-Chord, MF-P2P 1และ 3 และโหนดที่ใกล้ที่สุดที่เป็นสมาชิกของโหนดปลายทางในกรณีที่เป็น MF-P2P 2) ในแต่ละวินาที

การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ MF-P2P

สำหรับความหนาแน่นของระบบ MF-P2P เมื่อมีการทำการค้นหาแบบลูกโซ่ จะส่งผลให้จำนวนข้อมูลที่วิ่งผ่านซูเปอร์โหนด หรือโหนดที่อยู่ใกล้ที่สุดที่เป็นสมาชิกของโหนดปลายทางมีค่าเป็น 0 เสมอ เนื่องจากระบบ MF-P2P ที่ระบบมีความคงที่ โหนดแต่ละตัวจะมีการเก็บ contextual table เอาไว้สำหรับเป็นช่องทางลัดในการสื่อสาร ดังนั้นเมื่อมีการค้นหาแบบลูกโซ่เกิดขึ้นโหนดที่จะต้องทำการค้นหาต่อไปยังบริบทอื่น ๆ สามารถมองหาโหนดปลายทางได้จากข้อมูลที่เก็บอยู่ใน contextual table ได้ทันที โดยไม่ต้องมีการค้นหาข้อมูลใหม่

การวิเคราะห์เชิงคณิตศาสตร์สำหรับระบบ ML-Chord

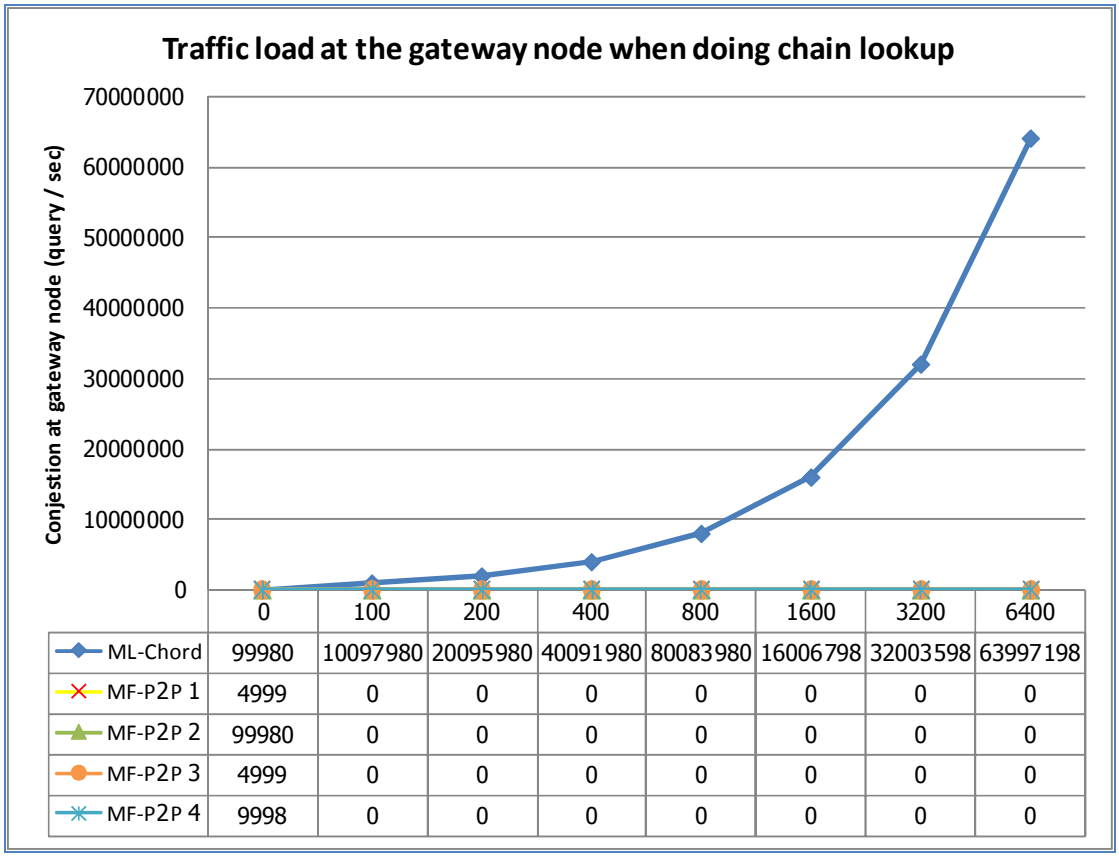
ในการค้นหาแบบ ML-Chord เมื่อโหนดต้องการค้นหาแบบลูกโซ่ขึ้น โหนดจะต้องทำการค้นหาปลายทางใหม่เสมอในทุกการร้องขอ ดังนั้นความหนาแน่นที่เกิดขึ้น บริเวณซูเปอร์โหนดของบริบทปลายทางจึงเพิ่มสูงขึ้นเป็นจำนวนเท่าตัว

ดังนั้นสมการคณิตศาสตร์ที่ใช้ในการคำนวณหาความหนาแน่นของ ML-Chord เมื่อมีการค้นหาแบบลูกโซ่ คือ

(0-16)

โดยที่ ch หมายถึงจำนวนการค้นหาแบบลูกโซ่ที่เกิดขึ้น

ผลการวิเคราะห์เมื่อกำหนดจำนวนบริบทและจำนวนโหนดคงที่



	0	100	200	400	800	1600	3200	6400
ML-Chord	99980	10097980	20095980	40091980	80083980	16006798	32003598	63997198
MF-P2P 1	4999	0	0	0	0	0	0	0
MF-P2P 2	99980	0	0	0	0	0	0	0
MF-P2P 3	4999	0	0	0	0	0	0	0
MF-P2P 4	9998	0	0	0	0	0	0	0

รูปที่ 0-12 กราฟแสดงความหนาแน่นในระบบเมื่อมีการค้นหาแบบลูกโซ่ โดยจำนวนโหนดคงที่คือ 100,000 โหนด และจำนวนบริบทคงที่ 5,000 บริบท แต่เพิ่มจำนวนครั้งของการหาแบบลูกโซ่ขึ้นตามข้อมูลในแกน X ส่วนความหนาแน่นที่เกิดขึ้นจะแสดงในแกน Y

สรุปผล

จากรูปที่ 0-12 จะเห็นว่าเมื่อมีการค้นหาแบบลูกโซ่เกิดขึ้นนั้น

ความหนาแน่นที่เกิดขึ้นบริเวณซูเปอร์โหนดต่าง ๆ ของ ML-Chord จะมีปริมาณสูงมากเนื่องจากโหนดเหล่านั้น ต้องทำการค้นหาใหม่เสมอในทุก ๆ ครั้งของการค้นหาแบบลูกโซ่ ซึ่งแตกต่างกับระบบ MF-P2P รูปแบบต่าง ๆ ที่จะเกิดความหนาแน่นขึ้นในการค้นหาครั้งแรกเพียงครั้งเดียวเท่านั้น และเมื่อมีการค้นหาแบบลูกโซ่เกิดขึ้น โหนดจะส่งสัญญาณการสืบค้นไปยังโหนดปลายทางโดยทันที ซึ่งข้อมูลของโหนดปลายทางได้มาจากการค้นหาภายใน contextual table ในโหนดของตนเอง ดังนั้นจึงไม่จำเป็นต้องทำการค้นหาใหม่ในทุก ๆ การร้องขอของการค้นหาแบบลูกโซ่

การเปรียบเทียบหน่วยความจำที่ต้องใช้ในระบบ

ในหัวข้อนี้จะเป็นการเปรียบเทียบหน่วยความจำที่ใช้ในแต่การค้นหาแต่ละระบบ ซึ่งหน่วยความจำที่ใช้นี้จะนับเป็นจำนวนคู่อันดับที่อยู่ในตารางต่าง ๆ ของแต่ละระบบ โดยในสมการที่สรุปมาจะประกอบไปด้วยตัวแปรดังต่อไปนี้

- C หมายถึง จำนวนบริบทที่แต่ละโหนดเข้าร่วม
- T หมายถึง จำนวนคู่อันดับในตารางสำหรับข้อมูลออนโทโลยี ซึ่งข้อมูลจะถูกเก็บอยู่ในรูปแบบของทริปเปิล (triple) โดยข้อมูลนี้จะเป็นข้อมูลที่ถูกเก็บอยู่ในตาราง DHT
- R หมายถึง จำนวนคู่อันดับในตาราง contextual table
- S หมายถึง จำนวนซูเปอร์โหนดที่ถูกเลือกไว้สำหรับรูปแบบ MF-P2P แบบที่มีการใช้ซูเปอร์โหนดหลายตัวในแต่ละบริบท
- MAX หมายถึง จำนวนคู่อันดับสูงสุดที่เกิดขึ้นในตาราง finger table ซึ่งในที่นี้จะมีค่าเท่ากับ 256 คู่อันดับตามมาตรฐานของ Chord

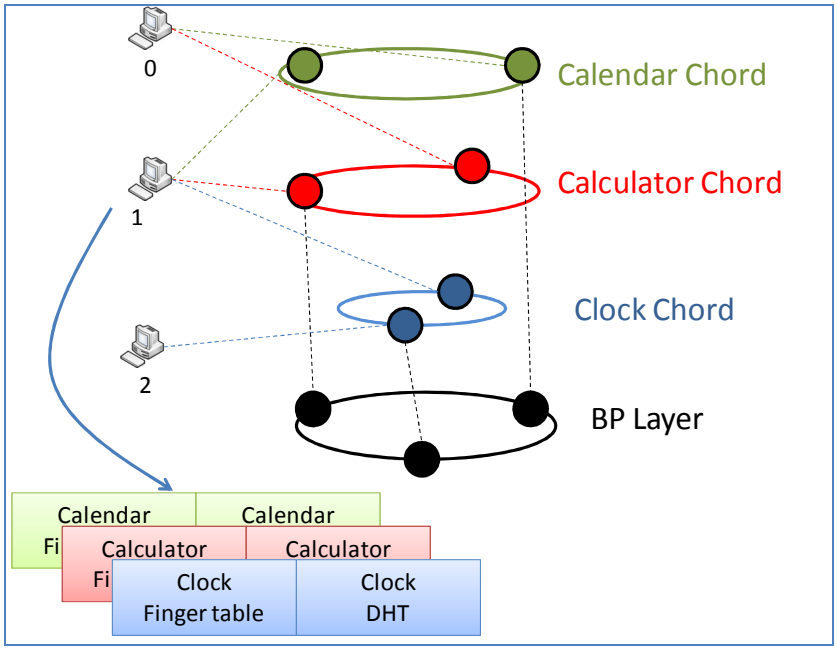
การวิเคราะห์ทางคณิตศาสตร์สำหรับหน่วยความจำที่ต้องใช้ในระบบ

1. ปริมาณหน่วยความจำที่ใช้สำหรับการค้นหารูปแบบ ML-Chord

สำหรับรูปแบบการค้นหาแบบ ML-Chord จะไม่มีการเก็บ contextual table และออนโทโลยีถูกเก็บอยู่ที่แต่ละโหนดอย่างไม่มีรูปแบบ ซึ่งส่งผลให้ออนโทโลยีจำเป็นจะต้องเก็บในทุกคอร์ดที่สร้าง โดยสรุปแล้วการค้นหาแบบ ML-Chord มีตารางข้อมูลที่ต้องคำนึงถึงเพียง

สองตารางนั่นคือ finger table และ distributed hash table (DHT) โดยข้อมูลที่ถูกเก็บอยู่ใน finger table ของ ML-Chord มีค่าสูงสุดอยู่ที่ 256 คู่อันดับ เนื่องจากค่าดังกล่าวเป็นค่าสูงสุดในมาตรฐานของรูปแบบการเชื่อมต่อเพียร์ทูเพียร์แบบมีโครงสร้างแบบคอร์ด และข้อมูลที่ถูกเก็บอยู่ใน DHT จะเป็นข้อมูลออนโทโลยีของตนเอง ดังนั้นข้อมูลที่เก็บจึงมีค่าเท่ากับ T ทริปเปิล

แต่เมื่อระบบ ML-Chord มีการเชื่อมต่อกันเป็นหลายชั้น จะส่งผลให้ระบบดังกล่าวมีการใช้หน่วยความจำเท่ากับ 256 + T ในทุก ๆ คอร์ด ดังนั้นจึงสรุปได้ว่า ปริมาณข้อมูลที่ แต่ละโหนดต้องเก็บจึงมีค่าเท่ากับ C * (MAX + T) ดังรูปที่ 0-13 (โดยที่ MAX หมายถึงจำนวนคู่อันดับสูงสุดที่มีได้ใน DHT ตามมาตรฐาน Chord ซึ่งในที่นี้มีค่าเท่ากับ 256)



รูปที่ 0-13 จำนวนข้อมูลที่ต้องเก็บในแต่ละโหนดที่มีการค้นหาแบบ ML-Chord

ซึ่งรูปที่ 0-13 แสดงให้เห็นถึงข้อมูลที่แต่ละโหนดจำเป็นต้องเก็บ ซึ่งจากรูปจะเห็นว่าโหนดหมายเลข 1 เข้าร่วมบริบท Calendar, Calculator, และ Clock ดังนั้นข้อมูลที่โหนดดังกล่าวจะต้องเก็บจึงมีค่าเป็น 3 * (256+T)

2. ปริมาณหน่วยความจำที่ใช้สำหรับการค้นหาแบบ MF-P2P ในรูปแบบที่มีความสัมพันธ์เท่าเทียมกันแบบทางเดียว (MF-P2P 1)

หน่วยความจำที่ถูกใช้ไปสำหรับการค้นหารูปแบบนี้จะถูกแบ่งออกเป็นสามส่วนใหญ่เช่นเดียวกันกับการค้นหาแบบ MF-P2P แบบมีการกำหนดซูเปอร์โหนด แต่ข้อมูลที่ถูกเก็บ

จำเป็นจะต้องเพิ่มขึ้นในชั้นของ Global Chord เนื่องจาก finger table ของโหนดทุกโหนดในระบบ MF-P2P รูปแบบนี้จะมีการเพิ่มข้อมูลขึ้นมาอีกในส่วนของการเพิ่มข้อมูลบริบทเข้า ไปใน finger table ดังนั้นข้อมูลคู่อันดับที่เพิ่มขึ้นจึงมีปริมาณเท่ากับ หรือ 128 เนื่องจากข้อมูล finger table มีขนาดเท่ากับ 256 ซึ่งในแต่ละแถวของ finger table จะมีข้อมูลเพิ่มขึ้นแถวละ 1 ชุดข้อมูล ดังนั้นข้อมูลคู่อันดับที่เพิ่มขึ้นทั้งหมดจึงมีขนาดเท่ากับ จึงส่งผลให้ finger table ของการค้นหารูปแบบนี้มีขนาดเพิ่มขึ้นจากเดิม 256 คู่อันดับกลายเป็น 384 คู่อันดับ โดยสรุปแล้ว ปริมาณข้อมูลที่เกิดขึ้นของการค้นหารูปแบบนี้จะมีค่าเท่ากับ MAX + + MAX * C + 3T + R

3. ปริมาณหน่วยความจำที่ใช้สำหรับการค้นหาแบบ MF-P2P ในรูปแบบที่มีการกำหนดซูเปอร์โหนดตัวเดียวสำหรับแต่ละบริบท (MF-P2P 2)

สำหรับการค้นหาในรูปแบบนี้โหนดจะมีการเก็บข้อมูลที่สำคัญอยู่สามส่วนนั่นคือ การเก็บข้อมูลที่ Finger table ซึ่งจะมีค่าเท่ากับ 256*(c+ 1) ซึ่งค่าดังกล่าวมาจากการเก็บข้อมูล finger table จำนวน 256 คู่อันดับและข้อมูลของ finger table ที่เก็บอยู่ในแต่ละบริบททำให้มีจำนวนคู่อันดับใน Context Chord ต่าง ๆ เป็น 256c ถัดไปเป็นการเก็บข้อมูลใน DHT จะถูกเก็บแค่ครั้งเดียวไม่ว่าโหนดดังกล่าวจะเข้าร่วมกลุ่มบริบทจำนวนเท่าไรก็ตาม แต่ข้อมูล DHT ที่เก็บ จะถูกกระจายออกเป็นสามส่วนเพื่อให้การค้นหาข้อมูลปลายทาง สามารถทำได้ง่ายมากขึ้นแต่จำนวนของข้อมูลก็จะเพิ่มขึ้นเป็นสามเท่าด้วยเช่นกัน ซึ่งข้อมูลที่เก็บใน DHT มีค่าเท่ากับ 3T และสุดท้ายเป็นข้อมูลของ contextual table ซึ่งเก็บข้อมูลของบริบทที่สัมพันธ์กับโหนดดังกล่าว ยิ่งโหนดดังกล่าวต้องมีความสัมพันธ์กับโหนดข้างเคียงมากก็จะยิ่งมีข้อมูลที่เก็บในตาราง contextual table มากขึ้นด้วยเช่นกัน โดยสรุปแล้วการค้นหารูปแบบดังกล่าวจะมีการใช้ หน่วยความจำเป็น MAX *(C + 1) + 3T + R

4. ปริมาณหน่วยความจำที่ใช้สำหรับการค้นหาแบบ MF-P2P ในรูปแบบที่มีความสัมพันธ์เท่าเทียมกันแบบสองทาง (MF-P2P 3)

หน่วยความจำที่ใช้ในการค้นหาของ MF-P2P ในรูปแบบที่มีความสัมพันธ์เท่าเทียมกันแบบสองทาง จะมีค่าใกล้เคียงกันกับการค้นหาแบบ MF-P2P ในรูปแบบที่มีความสัมพันธ์กันทางเดียว โดยจะมีการเพิ่มข้อมูล finger table ที่ใช้สำหรับการค้นหาข้อมูลในรูปแบบทวนเข็มนาฬิกา ส่งผลให้การเก็บข้อมูลจำเป็นจะต้องเพิ่ม finger table ขึ้นมาอีกหนึ่ง

ตาราง ดังนั้นจึงทำให้ปริมาณข้อมูลที่เก็บเพิ่มขึ้นเป็น ((MAX +)*2) + MAX * C + 3T + R หรือเท่ากับ 768 + 256C + 3T + R

5. ปริมาณหน่วยความจำที่ใช้สำหรับการค้นหาแบบ MF-P2P ในรูปแบบที่มีการกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท (MF-P2P 4)

สำหรับปริมาณหน่วยความจำที่ใช้ในการค้นหา MF-P2P รูปแบบที่มีการกำหนดซูเปอร์โหนดหลายตัว จะมีค่าใกล้เคียงกับการค้นหาโดยมีซูเปอร์โหนดเพียงตัวเดียว โดยข้อมูลที่เพิ่มขึ้นมา เป็นข้อมูลของซูเปอร์โหนดซึ่งถูกเก็บอยู่ใน DHT ซึ่งข้อมูลที่ถูกเก็บเอาไว้ จะมี จำนวนเท่ากับ S เนื่องจากจำนวนซูเปอร์โหนดที่เพิ่มขึ้นมาจะส่งผลต่อ DHT ของ Global Chord เท่านั้น และข้อมูลที่เพิ่มขึ้นมานี้จะมีค่าเท่ากับ จำนวนซูเปอร์โหนดที่ถูกเสนอขึ้นมา คูณกับจำนวนของบริบททั้งหมด ดังนั้นจึงส่งผลให้หน่วยความจำที่ใช้ไปในการค้นหารูปแบบนี้มีค่าเท่ากับ MAX (C + 1) + S + 3T + R

เปรียบเทียบหน่วยความจำที่ใช้ในระบบ

- ML-Chord ใช้หน่วยความจำ

$$C * (MAX + T) \quad (0\text{-}17)$$

- MF-P2P 1 ใช้หน่วยความจำ

$$MAX (\quad + C) + 3T + R \quad (0\text{-}18)$$

- MF-P2P 2 ใช้หน่วยความจำ

$$MAX (C + 1) + 3T + R \quad (0\text{-}19)$$

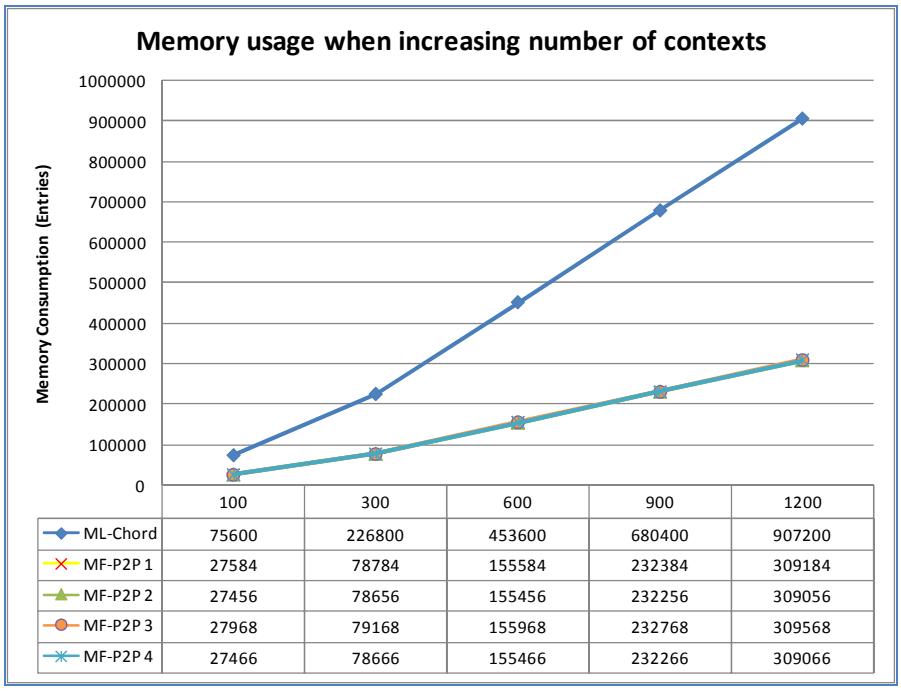
- MF-P2P 3 ใช้หน่วยความจำ

$$MAX (3 + C) + 3T + R \quad (0\text{-}20)$$

- MF-P2P 4 ใช้หน่วยความจำ

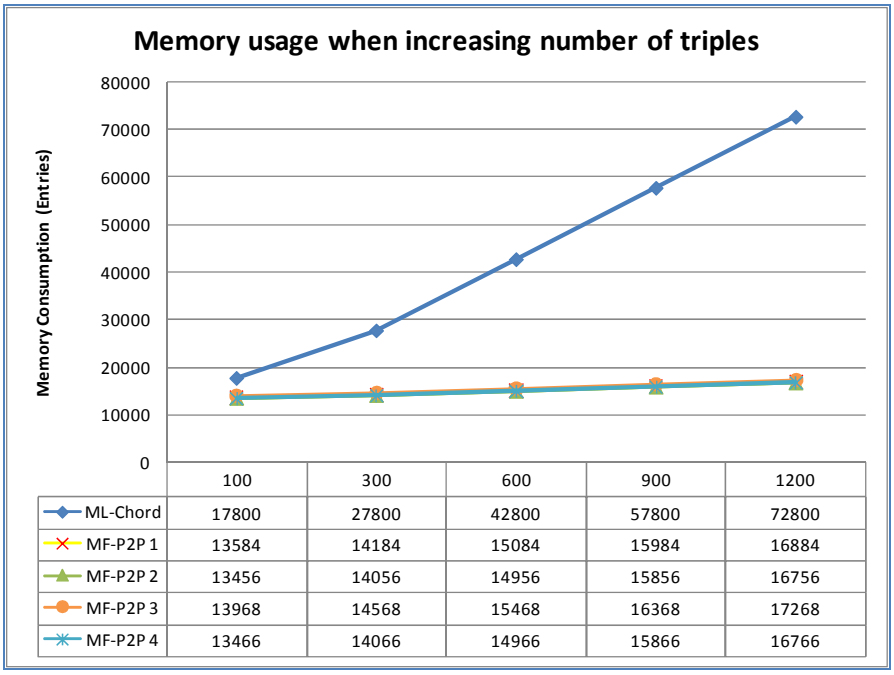
$$\text{MAX}\,(C + 1) + \quad + 3T + R \qquad (0\text{-}21)$$

จะเห็นว่าข้อมูลเหล่านั้นประกอบไปด้วยตัวแปรทั้ง 4 ตัวซึ่งเมื่อทดลองแทนค่าตัวแปรแต่ละตัวเพื่อหาผลกระทบ และตรวจสอบแนวโน้มการใช้หน่วยความจำสำหรับวิธีต่าง ๆซึ่งแสดงออกมาดังรูปที่ 0-14, รูปที่ 0-15, รูปที่ 0-16, และรูปที่ 0-17



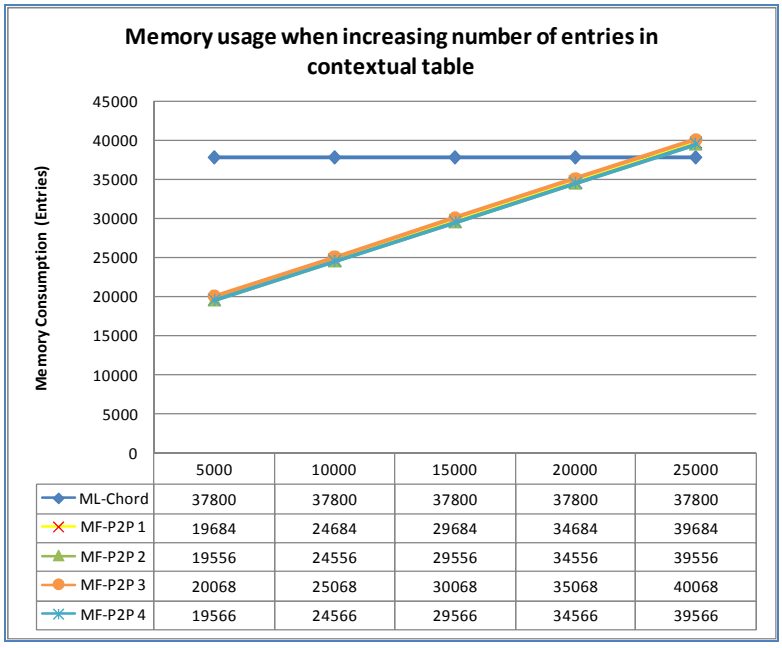
	100	300	600	900	1200
ML-Chord	75600	226800	453600	680400	907200
MF-P2P 1	27584	78784	155584	232384	309184
MF-P2P 2	27456	78656	155456	232256	309056
MF-P2P 3	27968	79168	155968	232768	309568
MF-P2P 4	27466	78666	155466	232266	309066

รูปที่ 0-14 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนบริบทโดยแกน X หมายถึงจำนวนบริบทที่เพิ่มขึ้นในขณะที่แกน Y แสดงจำนวนคู่อันดับที่ใช้ในการเก็บข้อมูลตาราง DHT



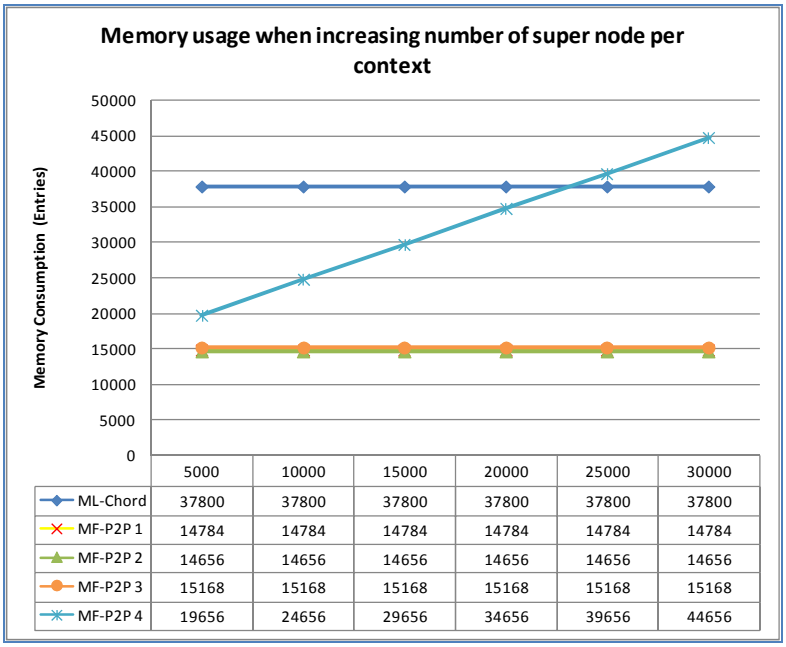
	100	300	600	900	1200
ML-Chord	17800	27800	42800	57800	72800
MF-P2P 1	13584	14184	15084	15984	16884
MF-P2P 2	13456	14056	14956	15856	16756
MF-P2P 3	13968	14568	15468	16368	17268
MF-P2P 4	13466	14066	14966	15866	16766

รูปที่ 0-15 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนทริปเปิลของออนโทโลยีโดยแกน X แสดงจำนวนทริปเปิลที่เพิ่มขึ้นและแกน Y แสดงจำนวนคู่อันดับที่ใช้ในตาราง DHT



	5000	10000	15000	20000	25000
ML-Chord	37800	37800	37800	37800	37800
MF-P2P 1	19684	24684	29684	34684	39684
MF-P2P 2	19556	24556	29556	34556	39556
MF-P2P 3	20068	25068	30068	35068	40068
MF-P2P 4	19566	24566	29566	34566	39566

รูปที่ 0-16 แสดงการใช้หน่วยความจำเมื่อเพิ่มข้อมูลในตาราง Contextual tableโดยแกน X หมายถึงจำนวนคู่อันดับที่เพิ่มขึ้นในตาราง contextual table ส่วนแกน Y แสดงให้เห็นถึงจำนวนคู่อันดับที่ใช้ในตาราง DHT



	5000	10000	15000	20000	25000	30000
ML-Chord	37800	37800	37800	37800	37800	37800
MF-P2P 1	14784	14784	14784	14784	14784	14784
MF-P2P 2	14656	14656	14656	14656	14656	14656
MF-P2P 3	15168	15168	15168	15168	15168	15168
MF-P2P 4	19656	24656	29656	34656	39656	44656

รูปที่ 0-17 แสดงการใช้หน่วยความจำเมื่อเพิ่มจำนวนซูเปอร์โหนดสำหรับ MF-P2P 4 โดยแกน X แสดงให้เห็นถึงจำนวนซูเปอร์โหนดที่ถูกกำหนดในแต่ละบริบทและแกน Y เป็นจำนวนข้อมูลที่ต้องเก็บในตาราง DHT

จากรูปที่ 0-14, รูปที่ 0-15, รูปที่ 0-16, และ รูปที่ 0-17 แสดงให้เห็นถึงการเปรียบเทียบการใช้หน่วยความจำสำหรับระบบการค้นหาแบบต่าง ๆ ซึ่งจากกราฟจะเห็นว่าระบบ ML-Chord มีการใช้หน่วยความจำเยอะที่สุด เนื่องจากระบบดังกล่าวโหนดแต่ละตัวไม่มีการรับรู้ถึงข้อมูลของโหนดตนเองที่อยู่ในบริบทอื่น ๆ ซึ่งทำให้โหนดต้องกระจายข้อมูลออนโทโลยีทั้งหมดในทุกบริบทที่โหนดดังกล่าวได้เข้าร่วม

ส่วนระบบ MF-P2P 4 จะมีการใช้หน่วยความจำมากขึ้นเมื่อโหนดมีการเพิ่มจำนวนซูเปอร์โหนดมากขึ้น สำหรับระบบ MF-P2P อื่น ๆ จากกราฟทั้งสี่ข้างต้น จะเห็นว่าเมื่อมีการเพิ่มค่า C และ T ซึ่งเป็นค่าที่ส่งผลกระทบต่อ ML-Chord ด้วย จะเห็นว่า ML-Chord ได้รับผลกระทบมากกว่าระบบอื่น ๆ เป็นอย่างมาก และเมื่อมีการเพิ่มค่า R ซึ่งเป็นจำนวนข้อมูลของ contextual table ซึ่งมีเฉพาะใน MF-P2P จะเห็นว่าส่งผลกระทบต่อระบบน้อยมาก อย่างไรก็ตามระบบ MF-P2P 4 จะได้รับผลกระทบเมื่อมีการเพิ่มค่า S อย่างมาก สำหรับโหนดที่ต้องเก็บข้อมูลของซูเปอร์โหนดเอาไว้

สรุปผลการทดลอง

ในบทนี้ได้กล่าวถึงการนำวิธีทางคณิตศาสตร์มาใช้ในการคำนวณ เพื่อเปรียบเทียบประสิทธิภาพของการค้นหารูปแบบต่าง ๆ ซึ่งรูปแบบการค้นหาที่สนใจนำมาใช้ในการเปรียบเทียบครั้งนี้ได้แก่ ML-Chord เนื่องจากเป็นระบบที่มีความรวดเร็วในการค้นหามากที่สุด, การค้นหาแบบ MF-P2P รูปแบบที่ทุกโหนดมีความสำคัญเท่าเทียมกัน (MF-P2P 1), รูปแบบการมีซูเปอร์โหนดหนึ่งตัวในแต่ละบริบท (MF-P2P 2), MF-P2P รูปแบบที่ทุกโหนด มีความสำคัญเท่าเทียมกันแบบสองทาง (MF-P2P 3), และ MF-P2P รูปแบบการมีซูเปอร์โหนดหลายตัวในแต่ละบริบท (MF-P2P 4), ซึ่งผลการทดลองทั้งหมดสามารถสรุปได้ดังตารางที่ 0-1

ตารางที่ 0-1 ตารางเปรียบเทียบประสิทธิภาพของระบบต่าง ๆ

	ML-Chord	**MF-P2P 1**	**MF-P2P 2**	**MF-P2P 3**	**MF-P2P 4**
ความเร็วในการค้นหา	เร็วมากที่สุด 2 log ( ) + log(C)	ช้าที่สุด log ( ) +	เร็วมากที่สุด log ( ) + log (N)	ช้า log ( ) +	เร็วมากที่สุด log ( ) + log (N)
ความหนาแน่นของข้อมูล	หนาแน่นมากที่สุด	หนาแน่นน้อยที่สุด	หนาแน่นมากที่สุด	หนาแน่นน้อยที่สุด	หนาแน่นน้อย
ความเร็วในการค้นหา (เมื่อค้นหาแบบลูกโซ่)	ช้าที่สุด (log ( ) +) * ch	เร็วน้อยที่สุด log ( ) + + ch	เร็วมากที่สุด log ( ) + log (N) + ch	เร็วปานกลาง ( ) + + ch	เร็วมากที่สุด log ( ) + log (N) + ch
ความหนาแน่นของข้อมูล (เมื่อค้นหาแบบลูกโซ่)	หนาแน่นมากที่สุด	หนาแน่นน้อยที่สุด 0	หนาแน่นน้อยที่สุด 0	หนาแน่นน้อยที่สุด 0	หนาแน่นน้อยที่สุด 0
ปริมาณหน่วยความจำที่ใช้ในระบบ	ใช้มากที่สุด C * (MAX + T)	ใช้น้อยที่สุด MAX (+ C) + 3T + R	ใช้น้อยที่สุด MAX (C + 1) + 3T + R	ใช้น้อยที่สุด MAX (3 + C) + 3T + R	ใช้มาก MAX (C + 1) + S + 3T + R

จากตารางที่ 0-1 แสดงให้เห็นถึงการเปรียบเทียบข้อมูลสำหรับระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ดทั้งหมด แสดงให้เห็นว่าไม่มีระบบไหนที่สามารถทำงานได้ดีที่สุดในทุกสภาพแวดล้อมซึ่งสามารถสรุปได้ดังต่อไปนี้

- ML-Chord มีการค้นหาได้อย่างรวดเร็วที่สุด แต่จะมีปัญหาความหนาแน่นของข้อมูลบริเวณซูเปอร์โหนด และเมื่อมีการค้นหาแบบลูกโซ่ ระบบจะไม่สามารถทำงานได้ดีมากนัก และหน่วยความจำยังใช้มากที่สุดอีกด้วย เนื่องจากระบบ ML-Chord เป็นระบบที่ให้แต่ละโหนด เข้าร่วมคอร์ดต่างๆ ที่ไม่มีการเชื่อมต่อกัน จึงทำให้โหนดที่อยู่ต่างบริบทกันไม่สามารถแลกเปลี่ยนข้อมูลกันได้ แม้ว่าจะเป็นโหนดเดียวกันก็ตาม
- MF-P2P 1: เป็นระบบที่สามารถลดความหนาแน่นของข้อมูลได้ดีที่สุด และยังลดจำนวนหน่วยความจำที่ต้องใช้ในการเก็บข้อมูลลงได้อีกด้วยดังแสดงให้เห็นในตารางเปรียบเทียบ แต่ระบบมีปัญหาใหญ่คือความเร็วในการค้นหา จะทำได้ช้าที่สุดเมื่อเทียบกับระบบการค้นหาแบบอื่นๆ ระบบ MF-P2P 1 จึงเหมาะสำหรับโหนดที่มีทรัพยากรในระบบต่ำ
- MF-P2P 2: เป็นระบบที่มีการทำงานคล้ายกันกับ ML-Chord แต่จะมีการทำให้โหนดเหล่านั้นสามารถเข้าร่วมกลุ่ม Context-aware Chord อย่างเป็นมโนภาพ จึงทำให้โหนดนั้นสามารถรับรู้ถึงข้อมูลของ ตนเองในบริบทอื่นๆ และโหนดที่เกี่ยวข้อง จึงส่งผลให้ระบบลดข้อมูลที่ต้องเก็บใน DHT และเมื่อมีการใช้งาน contextual table จึงทำให้ระบบสามารถทำงานในการค้นหาแบบลูกโซ่ได้ดี
- MF-P2P 3: ถูกออกแบบมาเพื่อเพิ่มความเร็วในการค้นหาให้กับระบบ MF-P2P 1 ซึ่งจะมีการเก็บข้อมูลเพิ่มขึ้นมาจากระบบ MF-P2P 1 ไม่มากนักดังแสดงในหัวข้อ 5.5 แต่ระบบ MF-P2P 3 จะสามารถเพิ่มความเร็วให้กับการค้นหาสำหรับ MF-P2P 1 ได้เป็นสองเท่าดังที่ได้แสดงให้เห็นในหัวข้อ 5.1
- MF-P2P 4: ถูกออกแบบมาเพื่อลดจุดด้อยของ MF-P2P 2 โดยจะเป็นการลดความหนาแน่นของข้อมูลที่จะเกิดขึ้นบริเวณซูเปอร์โหนด โดยการเพิ่มจำนวนซูเปอร์โหนดเข้าไปในแต่ละบริบท แต่ความหนาแน่นของข้อมูลที่ลดลงจะต้องแลกมาด้วยจำนวนข้อมูลที่ต้องเก็บเพิ่มขึ้น แต่อย่างไรก็ตามจำนวนซูเปอร์โหนด ไม่จำเป็นต้องมีจำนวนมากเพื่อใช้ลดความคั่งของข้อมูลดังข้อมูลที่ได้แสดงไปแล้วในหัวข้อ 5.2

สรุปผลการวิจัย และข้อเสนอแนะ

ในบทนี้จะเป็นการกล่าวถึงการสรุปผลการวิจัยที่ได้ดำเนินการมาแล้วทั้งหมด รวมทั้งการเสนอแนะและแนวทางการนำระบบไปประยุกต์เพื่อปรับปรุงการทำงานให้ดียิ่งขึ้น และเพิ่มความเหมาะสมที่จะนำไปใช้ในสถานการณ์จริง

สรุปผลการวิจัย

ในการทำวิจัยในเรื่องการจัดการระบบเครือข่ายเพียร์ทูเพียร์โดยใช้ข้อมูลบริบท หรือ Peer Discovery based on Context Awareness ได้ชี้ให้เห็นถึงปัญหาต่าง ๆ ที่เกิดขึ้นสำหรับระบบล่วงรู้บริบทรูปแบบเดิมเมื่อมีการเชื่อมต่อแบบเครื่องแม่ข่ายและเครื่องลูกข่าย ซึ่งในวิทยานิพนธ์ฉบับนี้ได้กล่าวถึงวิธีการแก้ไขปัญหาดังกล่าวโดยการเปลี่ยนมาใช้ระบบล่วงรู้บริบทบนระบบเครือข่ายแบบเพียร์ทูเพียร์ แต่การใช้งานระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์ยังไม่เป็นที่นิยมนัก เนื่องจากระบบที่นำเสนอมาในปัจจุบันยังไม่เหมาะสำหรับระบบล่วงรู้บริบทอย่างเต็มที่เนื่องจากยังมีปัญหาด้านความเร็ว ความหนาแน่นของข้อมูล รวมไปถึงปริมาณข้อมูลที่ต้องเก็บเป็นจำนวนมากอีกด้วย วิทยานิพนธ์ฉบับนี้จึงได้นำเสนอแนวคิดใหม่ในการใช้งานระบบล่วงรู้บริบทบนเครือข่ายเพียร์ทูเพียร์โดยใช้ชื่อว่าระบบ Multiple Finger table Peer to Peer หรือ MF-P2P ซึ่งรายละเอียดในการออกแบบระบบดังกล่าวแสดงได้ดังต่อไปนี้

การออกแบบระบบล่วงรู้บริบทผ่านทางระบบเครือข่ายเพียร์ทูเพียร์ (MF-P2P)

การออกแบบระบบล่วงรู้บริบทโดยผ่านทางระบบเครือข่ายเพียร์ทูเพียร์จะออกแบบโดยคำนึงถึงข้อจำกัดที่สำคัญนั่นคือ โหนดทั้งหมดเสมือนว่าเป็นโหนดที่ไม่เคลื่อนที่นั่นคือ มีอัตราการเข้าและออกจากระบบน้อย, ต้องเพิ่มความเร็วในการค้นหาให้ได้มากที่สุด, ต้องหาทางลดปริมาณความหนาแน่นของข้อมูลที่ใช้ในระบบเครือข่ายมากที่สุด, การออกแบบจะไม่สนใจการออกแบบออนโทโลยีที่ดีโดยจะสมมติว่าโหนดทั้งหมดมีออนโทโลยีที่ดีอยู่แล้ว และโหนดเหล่านั้นสามารถได้รับข้อมูลบริบทที่ต้องการได้เสมอ ดังนั้นจึงส่งผลให้ระบบที่ออกแบบมามีพื้นฐานอยู่บนระบบเครือข่ายแบบเพียร์ทูเพียร์ในรูปแบบที่มีโครงสร้าง (Structured P2P) แบบคอร์ด (Chord) เนื่องจากระบบดังกล่าวสามารถทำให้เกิดการค้นหาได้อย่างรวดเร็ว และลดความหนาแน่นของข้อมูลในระบบได้เป็นอย่างมาก เนื่องจากในการค้นหาข้อมูลแต่ละครั้ง ข้อมูลจะถูก

ส่งไปตามเส้นทางที่ถูกกำหนดไว้อย่างแน่นอน (finger table) ทำให้ลดความหนาแน่นของข้อมูลในระบบเครือข่ายได้

แนวคิดโดยทั่วไปของระบบ MF-P2P คือให้โหนดแต่ละโหนดในระบบเข้าร่วมระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ด โดยที่แต่ละโหนดมีการเก็บข้อมูลของ finger table มากกว่าหนึ่ง finger table ตามแนวคิดของคอร์ดแบบดั้งเดิม เนื่องจากการแบ่งให้โหนดเก็บตาราง finger table ที่มากกว่าหนึ่งตารางจะส่งผลให้โหนดดังกล่าว เสมือนว่าถูกส่งไปเข้าร่วมกลุ่มคอร์ดแบบมโนภาพได้อย่างไม่จำกัด และข้อมูลที่ถูกเก็บอยู่ในกลุ่มคอร์ดมโนภาพเหล่านั้นยังสามารถถูกส่งต่อถึงกันได้ และจำนวนข้อมูลที่แต่ละโหนดต้องเก็บจะเป็นข้อมูลชุดเดียว ซึ่งจะถูกกระจายเก็บอยู่ในกลุ่มคอร์ดมโนภาพที่แตกต่างกันทำให้ลดข้อมูลที่ต้องกระจายเมื่อมีการเข้าร่วมกลุ่มคอร์ดหลายกลุ่มได้เป็นอย่างมาก ยิ่งไปกว่านั้นการที่ให้โหนดเก็บ finger table หลายตารางยังส่งผลดีต่อการค้นหาของระบบล่วงรู้บริบท เนื่องจากการค้นหาในระบบล่วงรู้บริบทจำเป็นต้องมีการใช้งานออนโทโลยีที่แตกต่างกันสองส่วนนั่นคือ schematic ontology และ instance ontology ซึ่งข้อมูลออนโทโลยีเหล่านั้นจะถูกกระจายอย่างมีระบบในกลุ่ม คอร์ดมโนภาพต่าง ๆ เพื่อให้ง่ายต่อการค้นหา

และระบบ MF-P2P ยังได้มีการนำเสนอว่าควรจะมีการเก็บตารางเพิ่มขึ้นมาอีกหนึ่งตารางนั่นคือ contextual table ซึ่งตารางดังกล่าวนั้นส่งผลให้ระบบมีการค้นหาได้อย่างเร็วขึ้นและลดความหนาแน่นของข้อมูลได้มากเมื่อมีการค้นหาแบบลูกโซ่ ซึ่งการค้นหาแบบลูกโซ่นี้หมายถึงการค้นหาที่เมื่อไปจนเจอปลายทางแล้วโหนดปลายทางจำเป็นต้องได้รับข้อมูลเพิ่มเติมจากโหนดที่อยู่ในบริบทอื่น ๆ โหนดปลายทางนั้นจึงต้องส่งสัญญาณไปยังโหนดอื่น ๆ ต่อไป ซึ่งเป็นเหตุการณ์ที่มีโอกาสจะเกิดขึ้นได้มากในระบบล่วงรู้บริบทจริง โดยการเพิ่ม contextual table นี้ได้แสดงให้เห็นแล้วว่าส่งผลกระทบต่อหน่วยความจำของระบบไม่มากนัก

สำหรับการออกแบบระบบ MF-P2P ได้ถูกออกแบบเอาไว้ในสี่รูปแบบนั่นคือ การค้นหาแบบ MF-P2P เมื่อมีการกำหนดซูเปอร์โหนดเพียงตัวเดียวสำหรับแต่ละบริบท, เมื่อมีการกำหนดซูเปอร์โหนดหลายตัวสำหรับแต่ละบริบท, เมื่อแต่ละโหนดมีความสำคัญเท่าเทียมกันและมีการค้นหาทางเดียว, เมื่อแต่ละโหนดมีความสำคัญเท่าเทียมกัน และมีการค้นหาทั้งสองทาง ซึ่งการออกแบบการทำงานในแต่ละรูปแบบมีความเหมาะสมสำหรับแต่ละสถานการณ์ต่างกันไป

การวิเคราะห์และเปรียบเทียบประสิทธิภาพการทำงานของระบบ MF-P2P

หลังจากวิทยานิพนธ์ฉบับนี้ได้กล่าวถึงการออกแบบระบบ MF-P2P แล้ว ต่อมาเป็นการกล่าวถึงการเปรียบเทียบและวิเคราะห์ประสิทธิภาพการทำงานของระบบ MF-P2P กับระบบ ML-Chord โดยเหตุที่เลือก ML-Chord มาเป็นตัวเปรียบเทียบ เนื่องจากระบบการค้นหาแบบ ML-Chord เป็นระบบที่ถูกออกแบบมาอย่างใกล้เคียงกับระบบ MF-P2P และระบบดังกล่าวยังเป็นระบบล่วงรู้บริบทบนระบบเครือข่ายเพียร์ทูเพียร์เพียงรูปแบบเดียว ที่ถูกออกแบบมาให้ทำงานอยู่ในระบบเพียร์ทูเพียร์แบบมีโครงสร้าง ซึ่งการวิเคราะห์จะเป็นการนำระบบ ML-

Chord คำนวณหาประสิทธิภาพทางด้านต่าง ๆ แล้วนำมาเปรียบเทียบกับค่าที่คำนวณได้จากระบบ MF-P2P ในรูปแบบทั้งสี่ ซึ่งประสิทธิภาพในด้านที่นำมาเปรียบเทียบได้แก่ ความเร็วในการค้นหา , ความหนาแน่นของข้อมูลที่ซูเปอร์โหนด, ความเร็วเมื่อมีการใช้การค้นหาแบบลูกโซ่, ความหนาแน่นของข้อมูลเมื่อมีการค้นหาแบบลูกโซ่, และการใช้หน่วยความจำที่ใช้สำหรับแต่ละระบบ

ซึ่งผลการทดลองแสดงให้เห็นว่าระบบ MF-P2P มีความสามารถในการค้นหาที่รวดเร็วเทียบเท่ากับระบบ ML-Chord และยังแสดงให้เห็นถึงประสิทธิภาพการทำงานที่ เหนือกว่าระบบ ML-Chord เมื่อมีการค้นหาแบบลูกโซ่ และใช้หน่วยความจำน้อยกว่าเมื่อโหนดหนึ่งจำเป็นจะต้องเข้าร่วมหลายบริบทในเวลาเดียวกัน และผลการทดลองยังชี้ให้เห็นถึงข้อดีและข้อเสียสำหรับการทำงานของแต่ละรูปแบบของ MF-P2P เพื่อให้สามารถนำไปประยุกต์ใช้ให้เหมาะสมกับสถานการณ์ต่าง ๆ ได้อย่างเหมาะสม

ข้อเสนอแนะ

สำหรับข้อเสนอแนะและแนวทางที่ใช้เพื่อปรับปรุงระบบให้สามารถนำไปใช้งานในสถานการณ์จริง หรือเพิ่มความสามารถให้สามารถใช้งานกับระบบอื่น ๆ มีข้อเสนอแนะดังต่อไปนี้

ความสอดคล้องกัน และความน่าเชื่อถือของข้อมูลออนโทโลยี

ในวิทยานิพนธ์ฉบับนี้ไม่ได้กล่าวถึงการสอดคล้องกันของออนโทโลยีที่แต่ละโหนดได้ส่งเข้ามาในกลุ่มเครือข่าย และถือว่าโหนดแต่ละตัวมีความน่าเชื่อถือของข้อมูลอยู่แล้ว โดยข้อมูลออนโทโลยีที่แต่ละโหนดส่งเข้ามาในระบบถือว่าเป็นข้อมูลที่สอดคล้องกัน และข้อมูลทั้งหมดน่าเชื่อถือ ดังนั้นแนวทางการเสนอแนะคือการนำวิธีต่าง ๆ เข้ามาตรวจสอบความสอดคล้อง และความน่าเชื่อถือของข้อมูล เช่น โปรแกรมมีการตรวจสอบข้อมูลที่แต่ละโหนดส่งออกมาก่อนที่จะส่งข้อมูลเหล่านั้นเข้าไปในระบบเครือข่ายจริง เป็นต้น

เพิ่มความสามารถในการค้นหาข้อมูลบริบท

สำหรับข้อมูลบริบทที่สามารถค้นหาได้ในวิทยานิพนธ์ฉบับนี้ สามารถทำการค้นหาได้ถึง OWL-Lite และ OWL-DL เท่านั้น ซึ่งระบบที่มีความซับซ้อนของออนโทโลยีมาก จะเป็นการค้นหาโดยใช้ OWL-Full ซึ่งการค้นหาดังกล่าวยังไม่สามารถนำมาใช้งานได้ในระบบ MF-P2P เนื่องจากระบบ MF-P2P อาศัยจะเน้นไปที่การค้นหาโดยใช้คำสั่งสืบค้นแบบ SPARQL เท่านั้น ซึ่งในระบบล่วงรู้บริบทที่มีความซับซ้อนจะต้องสามารถเข้าถึงข้อมูลล่วงรู้บริบทในรูปแบบ

ของการค้นหารูปแบบอื่น ๆ เช่นการใช้ rule based query เพื่อให้ผู้ใช้สามารถ เข้าถึงระบบล่วงรู้บริบทอย่างซับซ้อนได้

ทดสอบการใช้งานในสถานการณ์จริง

ในวิทยานิพนธ์ฉบับนี้ได้แสดงให้เห็นถึงความเป็นไปได้ในการใช้งานระบบ MF-P2P โดยผ่านทางการใช้แบบจำลองทางคณิตศาสตร์ เพื่อช่วยในการคำนวณความสามารถของระบบในด้านต่าง ๆ เช่น ความเร็วในการค้นหาข้อมูล, ความหนาแน่นของข้อมูลที่เกิดขึ้น, และจำนวนหน่วยความจำที่แต่ละโหนดต้องใช้ เมื่อเกิดสถานการณ์ต่าง ๆ ขึ้น ซึ่งผลการทดลองที่ได้แสดงไป ยังไม่ได้ถูกทดสอบบนระบบที่มีการใช้งานจริง ซึ่งผลลัพธ์ที่ได้อาจต่างจากค่าที่คำนวณไว้ได้ และยิ่งไปกว่านั้นการส่งข้อมูลระหว่างโหนดต่าง ๆ จำเป็นจะต้องมีการคำนึงถึงความปลอดภัยของข้อมูลที่ใช้ส่งในระบบอีกด้วย

เอกสารอ้างอิง

[1] Fredrik Arvidsson and Annika Flycht-Eriksson. “Ontologies I”. 2008.

[2] T. Gruber (2001) . “What is an Ontology?”. Online entry. Accessed Nov 9, 2009.

[3] Kore Nordmann. “Standardization of Ontologies”. Online entry. 13 May 2009.

[4] http://www.cs.manchester.ac.uk/~horrocks/ISWC2003/Tutorial/people+pets.owl.rdf. Online entry. Accessed Jan 17, 2010.

[5] Martin Abadi and Luca Cardelli. “A Theory of Objects”. Springer-Verlag New York, Inc. 1996.

[6] http://www.w3.org/Consortium. Online entry. Accessed Jan 19, 2010.

[7] http://www.w3.org/TR/PR-rdf-syntax/ “Resource Description Framework (RDF) Model and Syntax Specification”. Online entry. Accessed Jan 3, 2010.

[8] http://www.ninebynine.org/RDFNotes/RDFContexts.html “Contexts for RDF Information Modelling”. Online entry. Accessed Jan 17, 2010.

[9] http://rdfabout.com/intro/ “What is RDF and what is it good for?”. Online entry. Accessed Jan 19.2010.

[10] Smith, Michael K.; Chris Welty, Deborah L. McGuinness (2004-02-10). ″OWL Web Ontology Language Guide″. W3C. 2008.

[11] http://www.w3.org/TR/owl-features/ “OWL Web Ontology Language Overview”. W3C. 2004.

[12] http://www.w3.org/TR/owl2-direct-semantics/ “OWL2 Web Ontology Language Direct Semantics”. W3C. 2009.

[13] Dean Allemang and Jim Hendler. “Semantic web for the Working Ontologist”. Elsevier, ISBN-13: 978-0-12-373556-0, 2008.

[14] http://protege.stanford.edu/ “The Protégé Ontology Editor and Knowledge Acquisition System”. Online entry. Accessed Feb 3. 2010.

[15] http://jena.sourceforge.net/ “Jena – A Semantic Web Framework for Java”. Online entry. Accessed Feb 2. 2010.

[16] http://clarkparsia.com/pellet “Pellet: The Open Source Java OWL Reasoner”. Online entry. Accessed 2 Feb 2010.

[17] http://www.cs.manchester.ac.uk/~horrocks/FaCT/ “An OWL Description Logic (OWL-DL) Reasoner”. Online entry. Accessed 2 Feb 2010.

[18] http://owl.man.ac.uk/factplusplus/ “A New Generation of the FaCT OWL-DL Reasoner”. Online entry. Accessed 2 Feb 2010.

[19] http://en.wikipedia.org/wiki/Semantic_reasoner "A Semantic Reasoner Encyclopedia". Online entry. Accessed 2 Feb 2010.

[20] Mei, Jing and Harold Boley. "Interpreting SWRL Rules in RDF Graphs". Electronic Notes in Theoretical Computer Science (Elsevier) (151): Page 53-69. 2006.

[21] http://www.w3.org/TR/rdf-sparql-query/ "SPARQL Query Language for RDF". Online entry. Accessed 19 Jan 2010.

[22] http://openjena.org/ARQ/Tutorial/index.html "A Tutorial on SPARQL". Online entry. Accessed 19 Jan 2010.

[23] http://sparql.sourceforge.net/ "A SPARQL Engine for Java". Online entry. Accessed 19 Jan 2010.

[24] http://www.w3schools.com/SQl/default.asp "A Tutorial on SQL". Online entry. Accessed 21 Jan 2010.

[25] http://semanticweb.org/wiki/Main_Page "Semantic Web". Online entry. Accessed 27 Jan 2010.

[26] http://www.w3schools.com/semweb/default.asp "A Tutorial on Semantic Web Technology". Online entry. Accessed 27 Jan 2010.

[27] http://attempto.ifi.uzh.ch/acewiki/ "AceWiki, a semantic wikipedia website". Online entry. Accessed 27 Jan 2010.

[28] http://www.bittorrent.org/beps/bep_0003.html "The BitTorrent Protocol Specification". Online entry. Accessed 1 Feb 2010

[29] Ye Tian; Di Wu; Kam-Wing Ng. "Analyzing Multiple File Downloading in BitTorrent". Peer-to-Peer Computing, 2006. P2P 2006. Sixth IEEE International Conference on 6-8 Sept. 2006, Pages:39 - 48.

[30] Xiaojun Hei, Yong Liu, K. W. Ross. "IPTV over P2P streaming networks: the mesh-pull approach". Communications Magazine, IEEE In Communications Magazine, IEEE, Vol. 46, No. 2, Pages:86 - 92. 31 March 2008

[31] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", SIGCOMM' 01, August 2001.

[32] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, "A Scalable Content Addressable Network", ACM SIGCOMM, San Diego, CA, August 2001.

[33] M. Castro, P. Druschel, Y. C. Hu and A. Rowstron. ″Proximity neighbor selection in tree-based structured peer-to-peer overlays″, Technical report MSR-TR-2003-52, 2003.

[34] Ben Y. Zhao, Ling Huang, et al. "Tapestry: A Resilient Global-Scale Overlay for Service Deployment (2004)", IEEE Journal on Selected Areas in Communications, Vol 22, No. 1, January 2004.

[35] Petar Maymounkov, David Mazières. "Kademlia: A Peer-to-peer Information System Based on the XOR Metric", 2002

[36] Motoyuki Takaai, Hideaki Takeda, and Toyoaki Nishida. "Distributed Ontology Development Environment for Multi-agent Systems", workshop on Quantitative techniques for software agile process, Newport Beach, California, 2004

[37] Peter Becker, Peter Eklund, and Natalyia Roberts. "Peer-to-Peer Based Ontology Editing", International Conference on Next Generation Web Services Practices (NWeSP'05), 2005

[38] M. Ehrig, C. Tempich, and S. Staab. "SWAP: Ontology-Based Knowledge Management with Peer-to-Peer Technology", Proceedings of the second Konferenz Professionelles Wissensmanagement, 2003

[39] Geraldo Xexeo, Adriana Vivacqua, Jano Moreira de Souza, Bruno Braga. "COE: A collaborative ontology editor based on a peer-to-peer framework", Advanced Engineering Informatics Volume 19, Issue 2, April 2005, Pages 113-121

[40] Min Cai, Martin Frank, Baoshi Yan, and Rebert MacGregor. "A Subscribable Peer-to-Peer RDF Repository for Distributed Metadata Management", Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 2, No. 2., December 2004

[41] Tao Gu, Edmond Tan, and Hung Keng Pung. "A Peer-to-Peer Architecture for Context Lookup", Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05), 2005

[42] Wenwei Xue, Hung Keng Pung, Wenlong Ng, and Tao Gu. "Data Management for Context-Aware Computing", IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, Vol. 1, Pages 492-498, 2008

[43] Tao Gu, Hung Keng Pung , Daqui Zang . "A Peer-to-Peer overlay for context information search", Proceedings of the 14th international conference on computer communiation and networks (ICCCN 2005), NY: Wiley; 2005a, Pages 395-400

[44] Flavio De Paoli, and Marco Loregian. “Context-aware Applications with Distributed Ontologies”, Ubiquitous Mobile Information and Collaboration Systems (UMICS 2006)

[45] Eric Jui-Lin Lu, Yung-Fa Huang, Shu-Chiu Lu. “ML-Chord: A multi-layered P2P resource sharing model”, Journal of Network and Computer Applications Volume 32, Issue 3, May 2009, Pages 578-588

[46] L. Garces-Erice, E.W. Biersack, P.A. Felber, et al. “Hierarchical Peer-to-Peer Systems”, Proceedings of ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par), 2003

[47] Knarig Arabshian, and Henning Schulzrinne. “Distributed Context-aware Agent Architecture for Global Service Discovery”, The Second International Workshop on Semantic Web Technology For Ubiquitous and Mobile Applications (SWUMA′06), Trentino, Italy, August 2006

[48] Knarig Arabshian and Henning Schulzrinne. Gloserv: Global service discovery architecture. In First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous), August 2004.

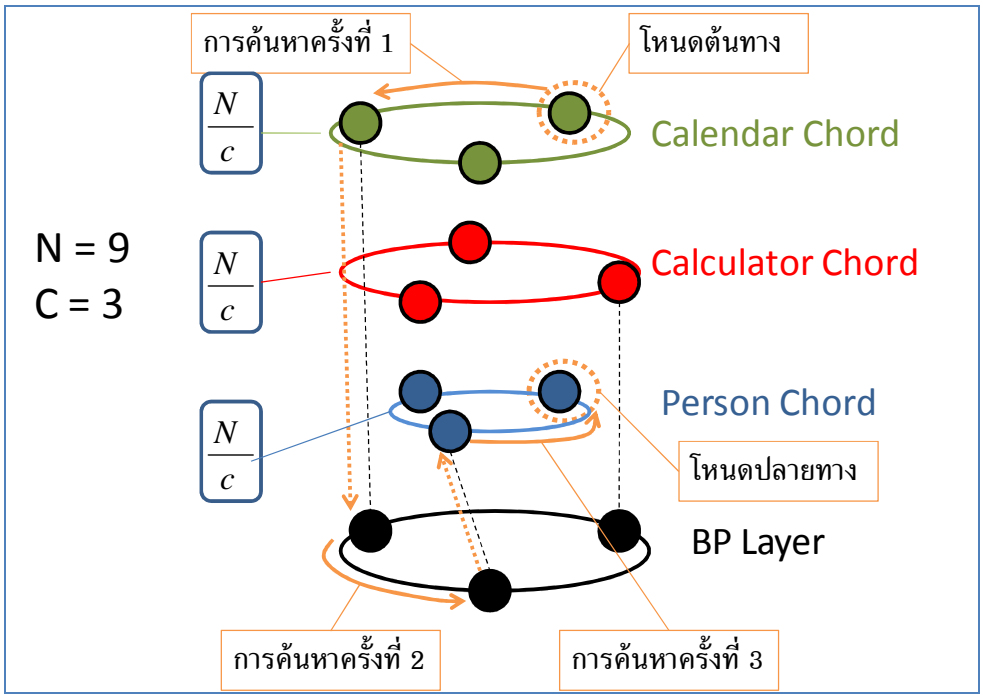
[49] Knarig Arabshian and Henning Schulzrinne. Hybrid hierarchical and peer-to-peer ontology-based global service discovery system. Technical Report CUCS-016-05, Columbia University, April 2005.

[50] Knarig Arabshian and Henning Schulzrinne. An ontology-based hierarchical peer-to-peer global service discovery system. Journal of Ubiquitous Computing and Intelligence (JUCI), 2006.

[51] Knarig Arabshian, Henning Schulzrinne, Dirk Trossen, and Dana Pavel. Gloserv: Global service discovery using the OWL web ontology language. In IEE International Workshop on Intelligent Environments (IE05). IEE, June 2005.

ภาคผนวก ก
การออกแบบสมการทางคณิตศาสตร์สำหรับความเร็วในการค้นหาของระบบ ML-Chord

ในหัวข้อนี้จะเป็นการอธิบายถึงหลักการออกแบบสมการทางคณิตศาสตร์สำหรับการค้นหาความเร็วในการค้นหาของระบบ ML-Chord ซึ่งการออกแบบคำนึงถึงหลักการที่ว่าความเร็วในการค้นหาของระบบเครือข่ายเพียร์ทูเพียร์แบบคอร์ดนั้น มีค่าเท่ากับ log(N) [31] โดยที่ N หมายถึงจำนวนโหนดทั้งหมดที่มีอยู่ในระบบ



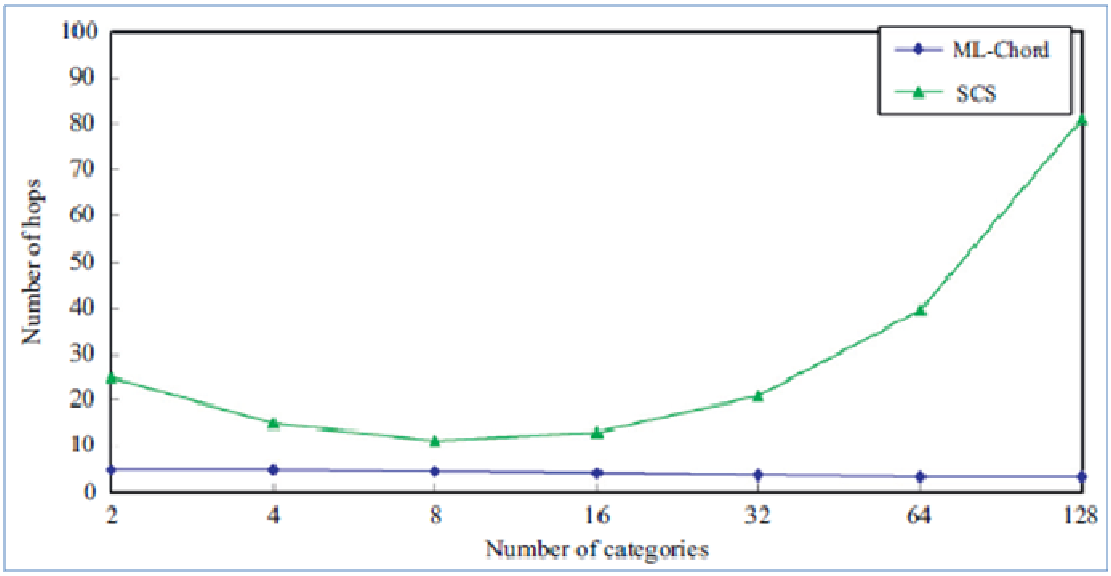
รูปที่ ก-1 การค้นหาของระบบ ML-Chord

จากรูปที่ ก-1 จะเห็นถึงการค้นหาในระบบ ML-Chord ซึ่งเกิดขึ้นทั้งหมดสามครั้งด้วยกัน ซึ่งรูปที่ ก-1 แสดงให้เห็นโหนดต้นทางอยู่ที่บริบท Calendar ซึ่งถูกแสดงด้วยรูปวงกลมที่มีรอยประ และโหนดปลายทางอยู่ที่บริบท Person จำนวนโหนดทั้งหมดในระบบมีจำนวนเท่ากับ 9 ซึ่งถูกแทนค่าด้วยตัวแปร N และมีจำนวนบริบททั้งหมด 3 บริบทได้แก่ Calendar, Calculator, Person โดยจำนวนโหนดทั้งหมดจะกระจายอยู่ในแต่ละบริบทเฉลี่ยเท่ากัน ดังนั้นจาก

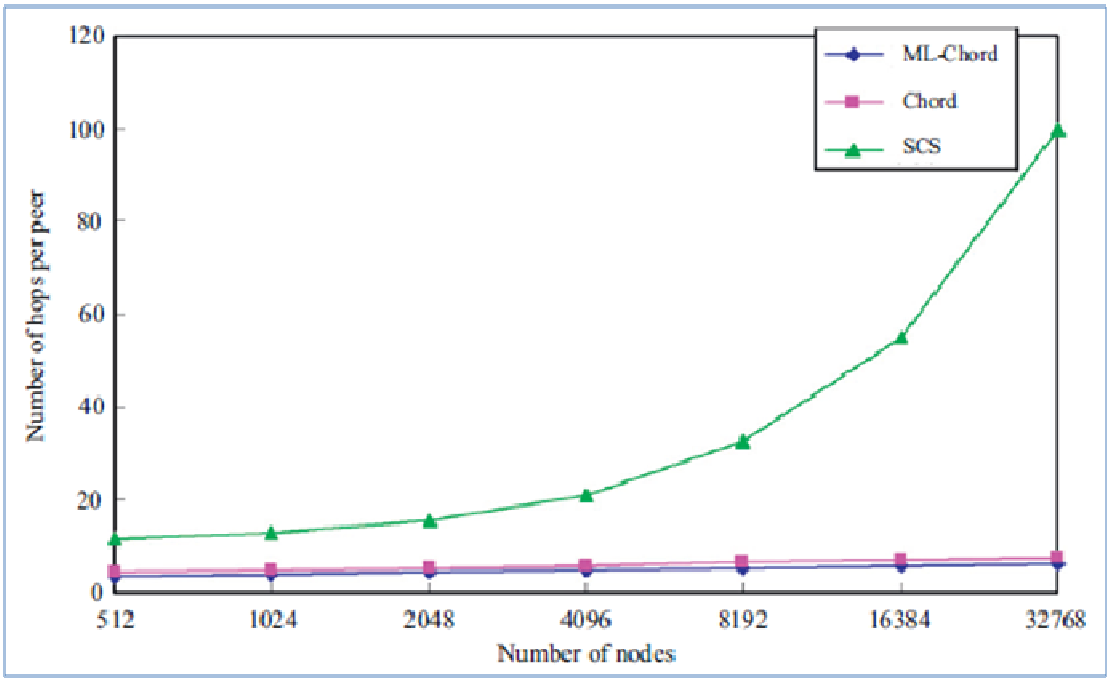
รูปที่ ก-1 จึงแสดงให้เห็นถึงจำนวนโหนดที่กระจายอยู่ทั้งหมด บริบทละ 3 โหนด ดังนั้นจึงสามารถสรุปได้ว่าจำนวนโหนดทั้งหมดที่กระจายอยู่ในแต่ละบริบทมีค่าเท่ากับ

การค้นหาของระบบ ML-Chord แบ่งเป็นสามส่วนได้แก่ การค้นหาส่วนที่ 1 คือ การค้นหาภายในกลุ่มบริบทที่โหนดต้นทางอยู่ เพื่อค้นหาตำแหน่งของซูเปอร์โหนดที่อยู่ภายในกลุ่มบริบทของตนเอง, การค้นหาส่วนที่ 2 เป็นการค้นหาภายในกลุ่มซูเปอร์โหนดด้วยกันเพื่อค้นหาซูเปอร์โหนดของบริบทที่โหนดปลายทาง เป็นสมาชิกอยู่, และการค้นหาส่วนที่ 3 เป็นการค้นหาภายในกลุ่มบริบทที่โหนดปลายทางเป็นสมาชิกอยู่ โดยผู้ที่เริ่มการค้นหาจะเป็นซูเปอร์โหนดของบริบทดังกล่าว ซึ่งจะส่งสัญญาณการค้นหาไปยังโหนดปลายทาง

จาก [45] ได้มีการนำเสนอกราฟที่เกิดจากกระบวนการ simulation โดยมีกราฟที่เกี่ยวข้องสองกราฟได้แก่ รูปที่ ก-2 และรูปที่ ก-3 โดยกราฟเหล่านั้นแสดงให้เห็นถึงความเร็วในการค้นหาในแกน Y โดยมีการเปลี่ยนค่าจำนวนของบริบท และจำนวนของโหนดไปตามแกน X กราฟที่แสดงเป็นกราฟที่นำมาจาก [45] ซึ่งจากรูปที่ ก-2 และรูปที่ ก-3 เป็นกราฟที่ใช้ในการเปรียบเทียบความเร็วในการค้นหาระหว่างระบบ ML-Chord กับระบบอื่น ๆ ได้แก่ ระบบคอร์ดแบบธรรมดา และระบบ SCS อย่างไรก็ตามกราฟที่จะให้ความสนใจ คือกราฟของ ML-Chord เท่านั้นเพื่อเปรียบเทียบแนวโน้มและค่าของกราฟระหว่างระบบ ML-Chord และกราฟที่ได้จากการออกแบบสมการทางคณิตศาสตร์



รูปที่ ก-2 กราฟแสดงความเร็วการค้นหาเมื่อกำหนดให้จำนวนโหนดคงที่ 1024 โหนดแล้วเพิ่มจำนวนบริบทขึ้นเรื่อย ๆ โดยอ้างอิงจาก [45]



รูปที่ ก-3 กราฟแสดงความเร็วการค้นหาเมื่อกำหนดให้จำนวนบริบทคงที่ 16 บริบท แล้วเพิ่มจำนวนโหนดขึ้นเรื่อย ๆ โดยอ้างอิงจาก [45]

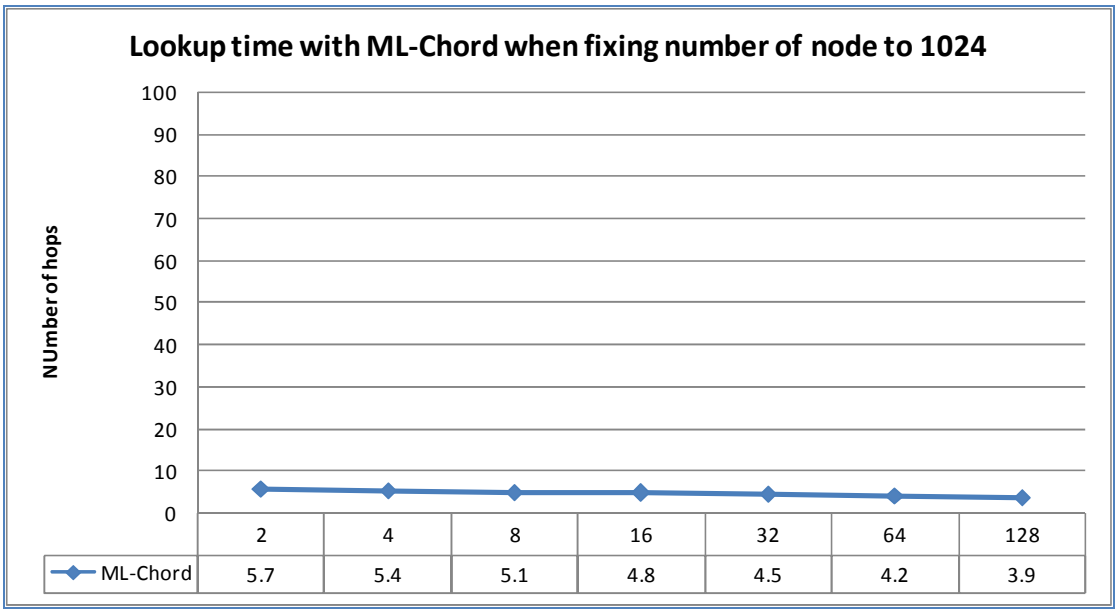
กราฟจากรูปที่ ก-2 และรูปที่ ก-3 จะถูกนำมาเปรียบเทียบกับสมการทางคณิตศาสตร์ ที่จะถูกนำไปสร้างเป็นกราฟเพื่อเปรียบเทียบระหว่างกราฟที่ได้จากการทำ simulation และกราฟที่ได้จากการสร้างสมการทางคณิศาสตร์ ซึ่งในการสร้างสมการทางคณิตศาสตร์ มีการคำนวณความเร็วในการค้นหาขึ้นมาจากส่วนการทำงานทั้งสามส่วนดังต่อไปนี้

การค้นหาส่วนที่ 1 และส่วนที่ 3 มีความเร็วในการค้นหาเท่ากันนั่นคือ มีความเร็วในการค้นหาเป็น log ( ) ซึ่งค่าดังกล่าวได้มาจากสมมติฐานที่ว่าความเร็วที่ใช้ในการค้นหาภายในคอร์ดใด ๆ ก็ตามจะมีค่าเท่ากับ log(N) โดยที่ N มีค่าเท่ากับจำนวนโหนดทั้งหมดในระบบคอร์ดนั้น ๆ ซึ่งในที่นี้ค่าของ N จะมีค่าเท่ากับ เพราะจำนวนโหนดของแต่ละบริบท จะถือว่ากระจายอยู่อย่างเท่าเทียมกันทั้งระบบ โดยตัวอย่างจากรูปที่ ก-1แสดงให้เห็นถึงจำนวนโหนดทั้งหมดที่อยู่ในระบบคือ 9 โหนด และแต่ละบริบทจะถูกโหนดที่กระจายไปอย่างเท่าเทียมกันเป็นบริบทละ 3 โหนดนั่นเอง เพราะฉะนั้นจึงทำให้ได้ความเร็วของการค้นหาส่วนที่ 1 และ ส่วนที่ 3 ออกมาเท่ากับ log ( )

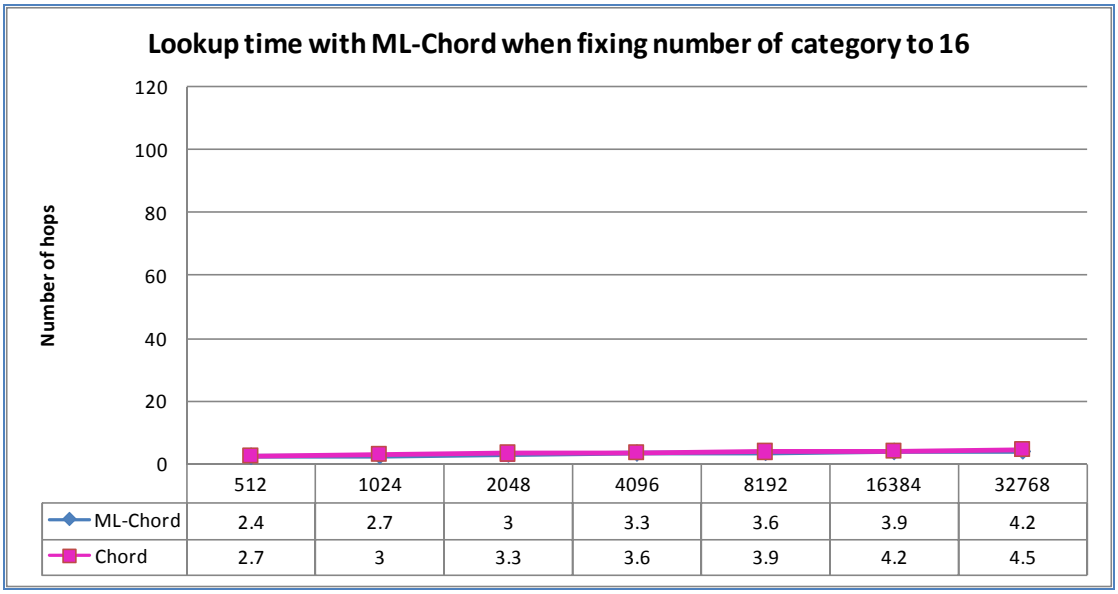
สำหรับการค้นหาส่วนที่ 2 เป็นการค้นหาภายในกลุ่มของโหนดที่เป็นซูเปอร์โหนด โดยมีความเร็วในการค้นหาเป็น log(C) ซึ่งความเร็วได้มาจากสมมติฐานว่าความเร็วในการ

ค้นหาภายในคอร์ดใด ๆมีค่าเท่ากับ log(N) โดยค่า N หมายถึงจำนวนโหนดทั้งหมดในกลุ่มคอร์ด แต่สำหรับการค้นหาภายในกลุ่มของโหนดที่เป็นซูเปอร์โหนดนี้ จำนวนโหนดที่นำมาคำนวณมีค่าเท่ากับ C เนื่องจากโหนดที่อยู่ในกลุ่มของซูเปอร์โหนด เป็นโหนดที่เป็นตัวแทนสำหรับแต่ละบริบท ดังนั้นจึงส่งผลให้จำนวนโหนดที่อยู่ในบริบทมีค่าเท่ากับจำนวนบริบททั้งหมดด้วยเช่นกัน จึงทำให้จำนวนของโหนดเหล่านั้นมีค่าเท่ากับ log(C)

โดยสรุปแล้วความเร็วในการค้นหาของระบบ ML-Chord เกิดจากผลรวมของจำนวน hop ที่เกิดขึ้นจากบริเวณที่ใช้ในการค้นหาทั้งสามส่วนนั่นคือ log ( ) + log ( ) + log(C) ซึ่งสมการดังกล่าวสามารถนำมาเขียนเป็นกราฟได้ดังรูปที่ ก-4 และรูปที่ ก-5



รูปที่ ก-4 ความเร็วในการค้นหาของ ML-Chord เมื่อกำหนดจำนวนโหนดคงที่ 1024 โหนด โดยแกน X เป็นจำนวนบริบทที่เพิ่มขึ้น และแกน Y เป็นความเร็วในการค้นหา



รูปที่ ก-5 ความเร็วในการค้นหาของ ML-Chord และ Chord เมื่อกำหนดจำนวนบริบทคงที่ 16 บริบท โดยแกน X เป็นจำนวนโหนดที่เพิ่มขึ้น และแกน Y เป็นความเร็วในการค้นหา

จากรูปที่ ก-4 และรูปที่ ก-5 แสดงให้เห็นถึงกราฟเมื่อนำสมการที่คำนวณได้มาแทนค่าลงไป โดยกราฟดังกล่าวเป็นการคำนวณเพื่อค้นหาความเร็วในการค้นหา ซึ่งกราฟที่แสดงขึ้นมานี้จะนำมาเปรียบเทียบกับกราฟที่ได้ถูกนำเสนอใน [45] ดังรูปที่ ก-2 และรูปที่ ก-3 โดยค่าที่แสดงทั้งในแกน X และแกน Y จะเป็นค่าเดียวกันทั้ง 4 รูป เนื่องจากต้องการเปรียบเทียบ ให้เห็นถึงแนวโน้มของโหนดเหล่านั้น

เมื่อเปรียบเทียบกราฟรูปที่ ก-4 และรูปที่ ก-5 มีแนวโน้มของกราฟความเร็วในการค้นหาของ ML-Chord สอดคล้องกันกับรูปที่ ก-2และ รูปที่ ก-3 โดยรูปที่ ก-4 และรูปที่ ก-2 เป็นกราฟที่ใช้ในการค้นหาความเร็วของระบบเมื่อระบบกำหนดให้มีจำนวนโหนดในระบบคงที่ และระบบมีการเพิ่มจำนวนบริบทขึ้นเรื่อย ๆ ซึ่งกราฟมีค่าเริ่มต้นที่ใกล้เคียงกันและแนวโน้มของกราฟจะลดลงเรื่อย ๆ ทั้งคู่ และค่าของกราฟที่ได้ใกล้เคียงกันไปตลอด แต่ไม่สามารถบอกได้อย่างชัดเจนนักว่าค่าของกราฟมีความใกล้เคียงกันมากเพียงใดเนื่องจากการทดลองที่กล่าวมาใน [45] ไม่มีการกล่าวตัวเลขอย่างชัดเจนของระบบเหล่านั้นเอาไว้

สำหรับเวลาในการค้นหาที่แสดงให้เห็นในรูปที่ ก-5 และรูปที่ ก-3 ได้แสดงให้เห็นถึงกราฟเมื่อกำหนดให้จำนวนบริบทคงที่อยู่ที่ 16 บริบท และได้มีการเพิ่มจำนวนของจำนวนบริบทขึ้นเรื่อย ๆ และกราฟยังแสดงให้เห็นถึงการเปรียบเทียบระหว่าง ML-Chord และการทำงานของ Chord โดยทั่วไปได้อีกด้วย ซึ่งจะเห็นในรูปที่ ก-5 และรูปที่ ก-3 ว่าแนวโน้มของกราฟ ML-Chord ที่นำมาแสดงมีค่าใกล้เคียงกันมาก

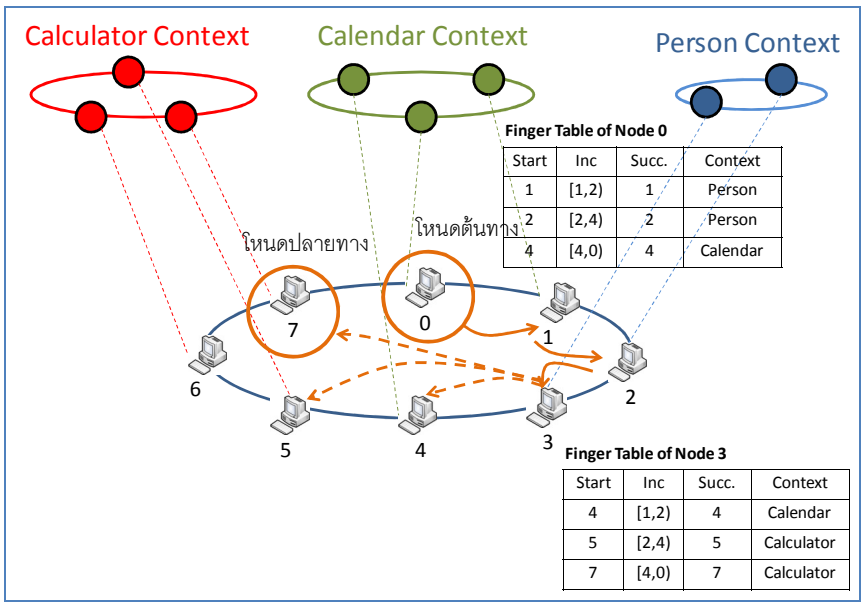
จากการทดลองจะเห็นว่า ค่าที่ได้จากกราฟ เมื่อนำมาเปรียบเทียบกันระหว่างกราฟที่ได้จากสมการทางคณิตศาสตร์ และกราฟที่ได้จากการทดสอบในกระบวนการ Simulation ดังที่ได้แสดงไว้ใน [45] มีค่าใกล้เคียงกันมาก พร้อมทั้งแนวโน้มของกราฟยังไปในทางเดียวกัน ดังนั้นจึงสามารถสรุปได้ว่า สมการทางคณิตศาสตร์ที่ออกแบบไว้มีความน่าเชื่อถือ และสามารถนำแนวคิดดังกล่าวไปประยุกต์ใช้ เพื่อเปรียบเทียบการทำงานของการค้นหาแบบ ML-Chord และการค้นหาแบบ MF-P2P

ภาคผนวก ข
การออกแบบสมการทางคณิตศาสตร์สำหรับความเร็วในการค้นหาระบบ MF-P2P

ระบบ MF-P2P ได้แบ่งรูปแบบการค้นหาภายใน Global Chord ออกได้เป็น 4 รูปแบบดังนี้คือ

- MF-P2P 1 หมายถึง การค้นหาโดยที่ทุกโหนดเป็นซูเปอร์โหนด และการค้นหาจะมีแนวทางการค้นหาไปในทิศทางตามเข็มนาฬิกา เพียงทิศทางเดียว
- MF-P2P 2 หมายถึง การค้นหาโดยที่มีการกำหนดซูเปอร์โหนดขึ้นมาหนึ่งตัว สำหรับบริบทแต่ละบริบท
- MF-P2P 3 หมายถึง การค้นหาโดยที่ทุกโหนดเป็นซูปเปอร์โหนด และการค้นหามีทิศทางได้ทั้งสองทาง นั่นคือทิศทางตามเข็มนาฬิกา, และทิศทางทวนเข็มนาฬิกา
- MF-P2P 4 หมายถึง การค้นหาโดยที่มีการกำหนดซูเปอร์โหนดหลายตัว สำหรับแต่ละบริบท

การค้นหาด้วยระบบ MF-P2P 1 สามารถสรุปสมการทางคณิตศาสตร์ที่ใช้ในการทดลองในวิทยานิพนธ์ฉบับนี้ได้จากรูปที่ ข-1



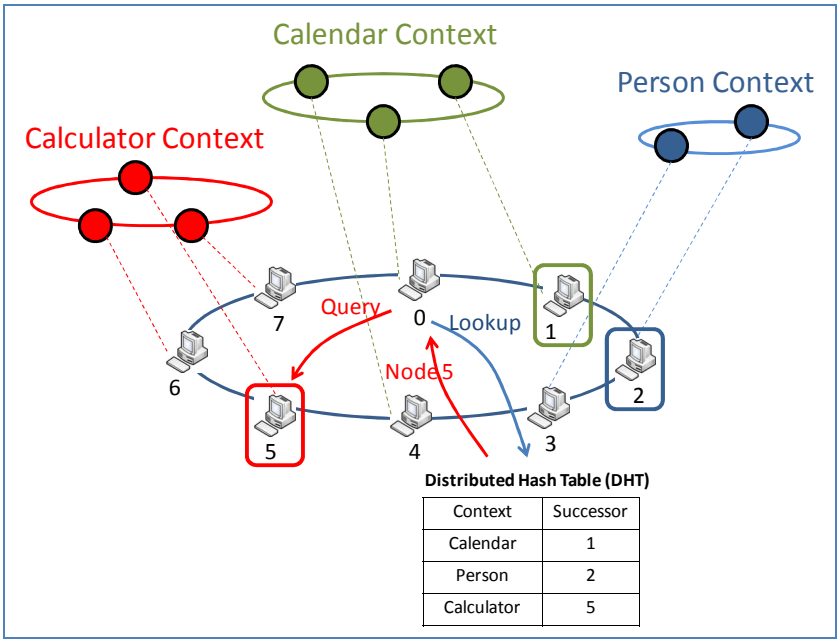
รูปที่ ข-1 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 1

รูปที่ ข-1 แสดงให้เห็นถึงกระบวนการค้นหาเมื่อโหนดมีจำนวนทั้งหมด 8 โหนด และมีจำนวนบริบททั้งหมด 3 บริบท นั่นคือ บริบท Calendar, บริบท Calculator และบริบท Person ซึ่งโหนดที่ต้องการเริ่มการค้นหานั่นคือ โหนดหมายเลข 0 ซึ่งอยู่ในบริบท Calendar ส่วนโหนดปลายทางคือ โหนดหมายเลข 7 ซึ่งอยู่ในบริบท Calculator

การค้นหาเริ่มจากโหนดต้นทาง (โหนดหมายเลข 0) ค้นหาภายใน finger table ของตนเองเพื่อค้นหาว่ามีบริบทปลายทางอยู่หรือไม่ โดยตาราง finger table ของโหนดหมายเลข 0 ที่แสดงในรูปที่ ข-1 รู้จักเพียงโหนดหมายเลข 1, 2, และ 4 ซึ่งโหนดเหล่านั้นไม่ได้อยู่ในบริบทปลายทาง ดังนั้นโหนดหมายเลข 0 จึงส่งสัญญาณการค้นหาไปยังโหนดใกล้เคียงที่สุด นั่นคือโหนดแรกในตาราง finger table ซึ่งเป็นโหนดหมายเลข 1 หลังจากโหนดที่ 1 ได้รับข้อมูลการสืบค้นแล้วจะทำการค้นหา finger table ของตนเองต่อไป โดยจากรูปสมมติว่าสัญญาณการค้นหาถูกส่งไปเรื่อย ๆ จนถึงโหนดหมายเลข 3 ซึ่งตาราง finger table ของโหนดหมายเลข 3 ประกอบไปด้วยโหนดหมายเลข 4, 5, และ 7 โดยโหนดหมายเลข 3 มีความรู้ของโหนดปลายทางซึ่งเป็นโหนดหมายเลข 7 พอดี ดังนั้นโหนดหมายเลข 3 จึงสามารถส่งสัญญาณการค้นหาไปยังโหนดดังกล่าวได้อย่างถูกต้อง

สำหรับระยะเวลาที่ยาวที่สุดที่ใช้ในการค้นหาเท่ากับ เนื่องจากในการค้นหานั้นโหนดจะต้องส่งสัญญาณการค้นหาจากโหนดต้นทางไปยังโหนดข้างเคียงเรื่อย ๆ เพื่อให้สัญญาณการสืบค้นดังกล่าวถูกค้นหาภายใน finger table ของโหนดปลายทางเหล่านั้นต่อไป ซึ่งจำนวนครั้งที่ต้องส่งไปสูงที่สุดคือ หรือเป็นจำนวนเท่ากับครึ่งรอบของทั้งวงคอร์ดนั่นเอง เนื่องจากโหนดแต่ละตัวที่อยู่ในระบบจะมีความรู้ของโหนดข้างเคียงอยู่ด้วยตามที่เก็บไว้ใน finger table เมื่อรวมกับจำนวน hop ที่ใช้ในการค้นหาภายในคอร์ดของบริบทปลายทางแล้วทำให้ได้ ความเร็วในการค้นหาเป็น + log ( )

การค้นหาด้วยระบบ MF-P2P 2 และ 4 สามารถสรุปสมการทางคณิตศาสตร์เพื่อค้นหาความเร็วในวิทยานิพนธ์ฉบับนี้ได้ โดยผ่านทางรูปที่ ข-2

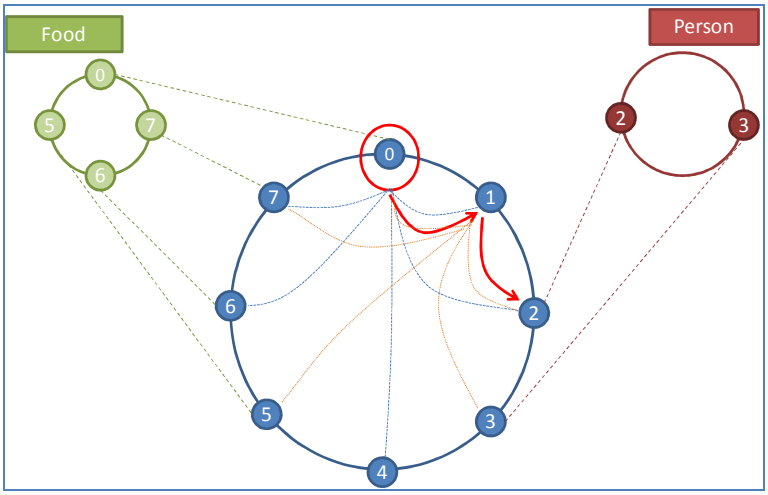


รูปที่ ข-2 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 2 และ MF-P2P 4

ความเร็วที่ใช้ในการค้นหาสำหรับระบบ MF-P2P 2 และ MF-P2P 4 มีความเร็วที่เท่าเทียมกันเนื่องจาก ระบบทั้งสองมีการเลือกซูเปอร์โหนดเช่นเดียวกัน แต่จำนวนซูเปอร์โหนดที่เลือกมีจำนวนที่แตกต่างกัน ดังนั้นจึงไม่ส่งผลต่อความเร็วที่ใช้ในการค้นหา แต่จะมีผลกระทบต่อปริมาณความหนาแน่นของระบบเครือข่าย ซึ่งจะกล่าวถึงในหัวข้อภาคผนวก ค. จากรูปที่ ข-2 เห็นว่าโหนดที่เริ่มต้นการค้นหาคือโหนดหมายเลข 0 และมีโหนดที่ถูกกำหนดให้เป็นซูเปอร์โหนดของแต่ละบริบทเป็นโหนดหมายเลข 1 และโหนดหมายเลข 5 ตามลำดับ ซึ่งจากรูปที่ ข-2 โหนดหมายเลข 0 ต้องการค้นหาโหนดปลายทางที่อยู่ในบริบท Calculator ดังนั้นโหนดหมายเลข 0 จึงส่งสัญญาณการค้นหาไปยัง DHT ได้โดยตรง เนื่องจากซูเปอร์โหนดสำหรับบริบท Calculator ได้ถูกกำหนดไว้ก่อนแล้ว ดังนั้นโดยสรุปแล้วความเร็วที่ใช้ในการค้นหาใน Global Chord สำหรับรูปแบบ MF-P2P 2 และ MF-P2P 4 มีค่าเท่ากับ log(N) ซึ่งส่งผลให้ความเร็วในการค้นหาทั้งหมดเมื่อรวมจำนวน hop ในการค้นหาภายในบริบทปลายทางแล้วมีค่าเท่ากับ

log(N) + log ( )

ความเร็วที่ใช้ในการค้นหาสำหรับระบบ MF-P2P 3 ที่จะถูกนำมาคำนวณเป็นสมการทางคณิตศาสตร์จะถูกแสดงให้เห็นในรูปที่ ข-3



รูปที่ ข-3 เวลาที่ใช้ในการค้นหาใน Global Chord สำหรับระบบ MF-P2P 3

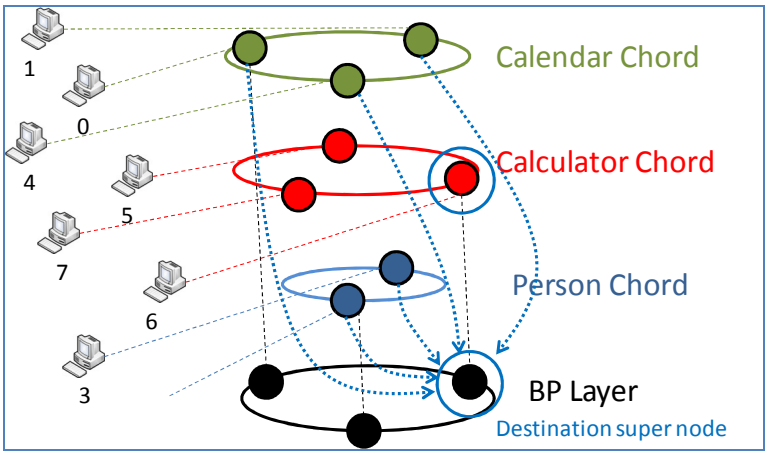
รูปที่ ข-3 แสดงให้เห็นถึงจำนวนความเร็วในการค้นหาภายในกลุ่ม Global Chord ซึ่งจากรูปจะเห็นว่าโหนดแต่ละตัวจะมีการเก็บข้อมูลของโหนดข้างเคียงเพิ่มขึ้น นั่นคือจะมีการเพิ่มจำนวนของ finger table ในระดับชั้น Global Chord ที่แต่ละโหนด จำเป็นจะต้องเก็บเพิ่มขึ้นอีก table หนึ่ง เพื่อใช้ในการค้นหาได้สองทาง ทั้งในทิศทางตามเข็มนาฬิกา และทิศทางทวนเข็มนาฬิกา ดังนั้นจึงทำให้ความเร็วในการค้นหาใน Global Chord ของ MF-P2P 3 มีค่าเป็น

ดังนั้นจึงส่งผลให้ความเร็วในการค้นหาของ MF-P2P 3 มีค่าเท่ากับ + log ( )

ภาคผนวก ค
การออกแบบสมการทางคณิตศาสตร์สำหรับความหนาแน่นของ ML-Chord และ MF-P2P

สมการของความหนาแน่น จะถูกออกแบบโดยมีการสมมติสถานการณ์ว่าโหนดแต่ละตัวจะส่งสัญญาณจากทุกโหนดวินาทีละสัญญาณ และสัญญาณการค้นหาทั้งหมดจะถูกส่งไปยังบริบทปลายทางเดียวกันทั้งหมด ซึ่งความหนาแน่นที่คำนวณออกมาเป็นความหนาแน่นที่บริเวณโหนดซึ่งทำหน้าที่เป็นซูเปอร์โหนดของแต่ละระบบ เพราะโหนดที่ทำหน้าที่เป็นซูเปอร์โหนดจะมีความหนาแน่นของสัญญาณสูงที่สุด

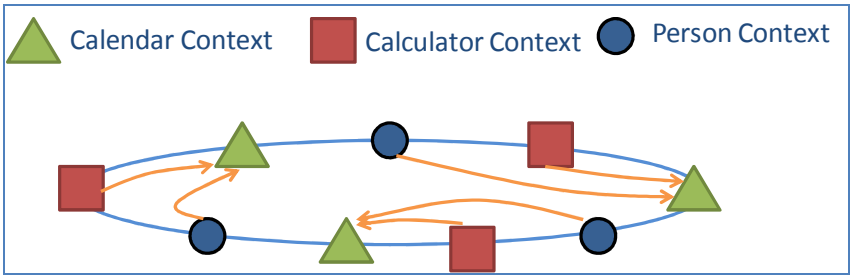
การคำนวณสมการทางคณิตศาสตร์สำหรับระบบ ML-Chord สามารถ คำนวณได้ดังแสดงให้เห็นในรูปที่ ค-1



รูปที่ ค-1 แสดงความหนาแน่นที่ซูเปอร์โหนดของระบบ ML-Chord

รูปที่ ค-1 จะเห็นว่าเมื่อทุกโหนดส่งสัญญาณการร้องขอไปยังบริบทปลายทางเพียงบริบทเดียว ซึ่งสัญญาณทั้งหมดจะถูกส่งผ่านไปยังซูเปอร์โหนดของบริบทเพียงโหนดเดียว ซึ่งรูปที่ ค-1 บริบทปลายทางคือโหนดที่อยู่ในบริบท Calculator โดยความหนาแน่น ของระบบเกิดขึ้นจากโหนดอื่น ๆ ที่ไม่ได้อยู่ในกลุ่มบริบทปลายทางทั้งหมดส่งสัญญาณมายังซูเปอร์โหนดปลายทางเพียงโหนดเดียว ส่งผลให้ความหนาแน่นของระบบเป็น หรือจำนวนของโหนดทั้งหมดที่ไม่อยู่ในบริบทปลายทางนั่นเอง

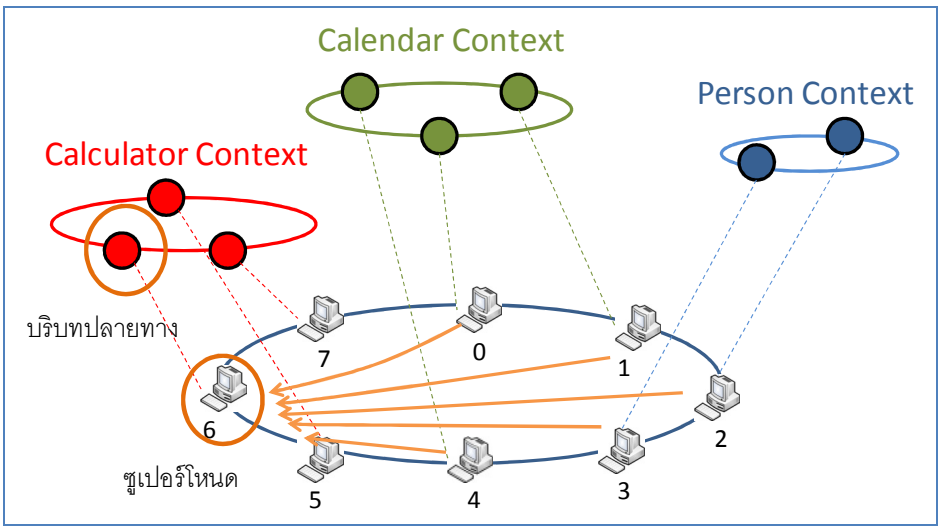
สำหรับสมการความหนาแน่นที่ใช้เพื่อคำนวณระบบ MF-P2P 1 และระบบ MF-P2P 3 หรือระบบที่ทุกโหนดในระบบทำตัวเป็นซูเปอร์โหนดอย่างเท่าเทียมกัน ซึ่งระบบดังกล่าวจะสมมติว่าโหนดทั้งหมดในระบบมีการกระจายตัวอย่างสมมาตรในทุก ๆ บริบทดังแสดงในรูปที่ ค-2



รูปที่ ค-2 การกระจายตัวของโหนดใน Global Chord ของระบบ MF-P2P 1 และ MF-P2P 3

โดยโหนดในระบบ MF-P2P 1 และ MF-P2P 3 กระจายอยู่อย่างสมมาตรกันตามรูปที่ ค-2 ซึ่งในการค้นหาเมื่อกำหนดให้บริบทปลายทางเป็นบริบท Calendar ซึ่งแทนด้วยรูปสามเหลี่ยมในรูปที่ ค-2 และโหนดที่อยู่ในบริบทอื่น ๆ จะส่งสัญญาณการค้นหาไปยังบริบทปลายทางดังกล่าว ดังนั้นจึงส่งผลให้ความหนาแน่นของระบบที่เกิดขึ้นเป็น C – 1 หรือเป็นตามจำนวนของโหนดทั้งหมดที่อยู่ก่อนหน้าบริบท Calendar นั่นเอง

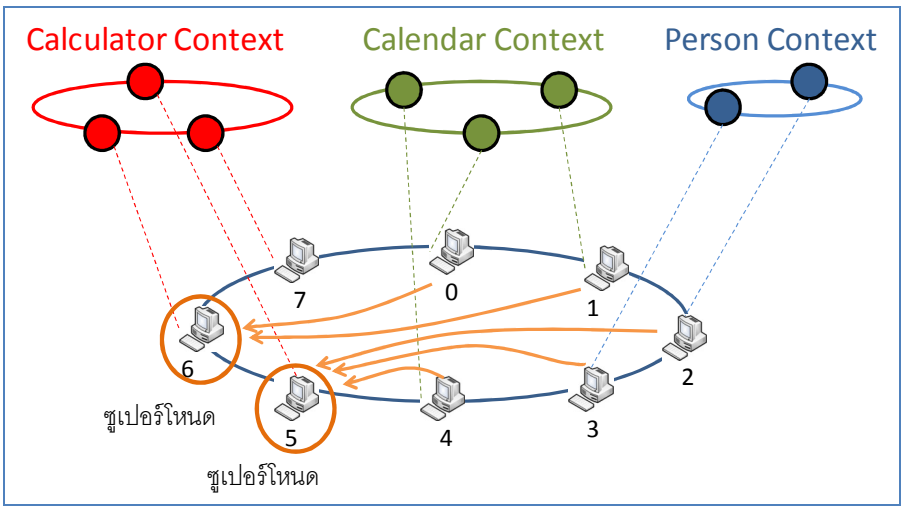
ในส่วนของความหนาแน่นของการค้นหาแบบ MF-P2P 2 ได้มาจากการที่โหนดทั้งหมดในระบบที่ไม่ได้อยู่ในบริบทเดียวกับโหนดปลายทาง ส่งสัญญาณการค้นหาไปยังบริบทปลายทางโดยผ่านทางโหนดปลายทาง



รูปที่ ค-3 ความหนาแน่นที่เกิดขึ้นของ MF-P2P 2

ดังนั้นรูปที่ ค-3 จะเห็นว่าความหนาแน่นที่เกิดขึ้น เกิดจากโหนดทุกตัวในระบบที่ไม่ได้อยู่ในบริบทปลายทาง ซึ่งจากรูปคือบริบท Calculator ทำให้โหนดที่อยู่ในบริบทปลายทางอื่น ๆ ส่งสัญญาณการค้นหาไปยังโหนดเพียงแค่โหนดเดียว ส่งผลให้สมการที่ใช้ในการคำนวณความหนาแน่นของระบบคือ หรือจำนวนโหนดทั้งหมดที่ไม่ได้เข้าร่วมบริบทปลายทางนั่นเอง

สำหรับการค้นหาแบบ MF-P2P 4 ที่โหนดมีการกำหนดซูเปอร์โหนดขึ้นมาหลายตัวสำหรับแต่ละบริบท สามารถสรุปได้ดังรูปที่ ค-4



รูปที่ ค-4 ความหนาแน่นที่เกิดขึ้นของ MF-P2P 4

รูปที่ ค-4 เมื่อโหนดมีการเพิ่มจำนวนของซูเปอร์โหนดขึ้น โหนดที่อยู่ในบริบทอื่น ๆ ที่จะส่งสัญญาณไปยังซูเปอร์โหนด โดยสัญญาณที่ส่งไปจะถูกเฉลี่ยไปให้กับซูเปอร์โหนดแต่ละตัวอย่างเท่ากัน ทำให้ความหนาแน่นที่เกิดขึ้นลดลงเป็นจำนวนซูเปอร์โหนดที่เพิ่มขึ้นเรื่อย ๆ

ดังนั้นทำให้ได้สมการสุดท้ายเป็น โดยที่ N เป็นจำนวนโหนดทั้งหมดใน Global Chord, C เป็นจำนวนบริบทที่เกิดขึ้น, และ s เป็นจำนวนซูเปอร์โหนดที่ถูกเลือกในแต่ละบริบท

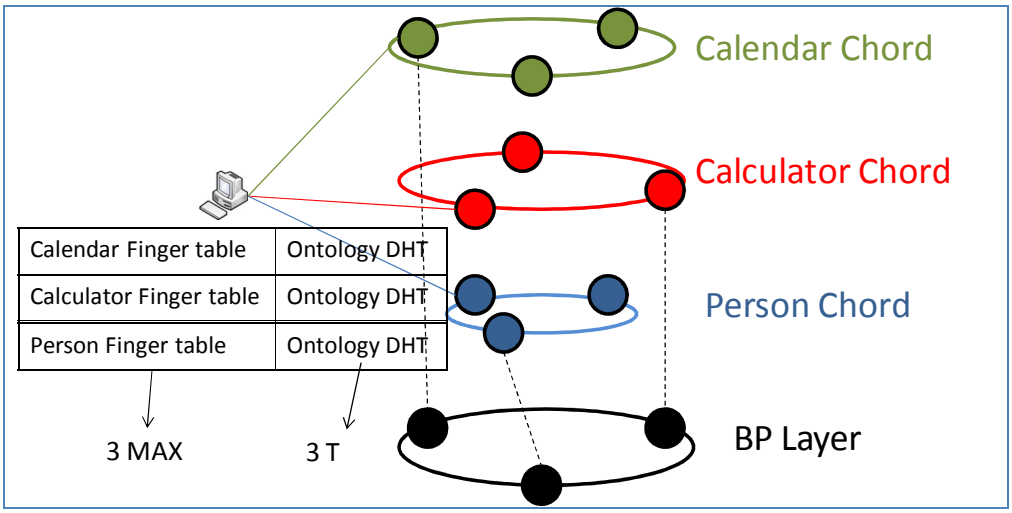
ภาคผนวก ง
การออกแบบสมการทางคณิตศาสตร์สำหรับหน่วยความจำของ ML-Chord และ MF-P2P

ในการออกแบบหน่วยความจำจะไม่ได้มองหน่วยความจำในรูปของ bit หรือ byte ตามการคำนวณหน่วยความจำในระบบคอมพิวเตอร์ทั่วไป แต่จะมองหน่วยความจำอยู่ใน รูปของจำนวนคู่อันดับที่เกิดขึ้นสูงสุดที่เป็นไปได้ในตารางต่าง ๆ เพื่อทำการเปรียบเทียบคู่อันดับเหล่านั้น โดยตัวแปรที่ใช้ในการคำนวณประกอบไปด้วย

- C หมายถึง จำนวนบริบทที่แต่ละโหนดเข้าร่วม
- T หมายถึง จำนวนคู่อันดับในตารางสำหรับข้อมูลออนโทโลยี ซึ่งข้อมูลจะถูกเก็บอยู่ในรูปแบบของทริปเปิล (triple) โดยข้อมูลนี้จะเป็นข้อมูลที่ถูกเก็บอยู่ในตาราง DHT
- R หมายถึง จำนวนคู่อันดับในตาราง contextual table

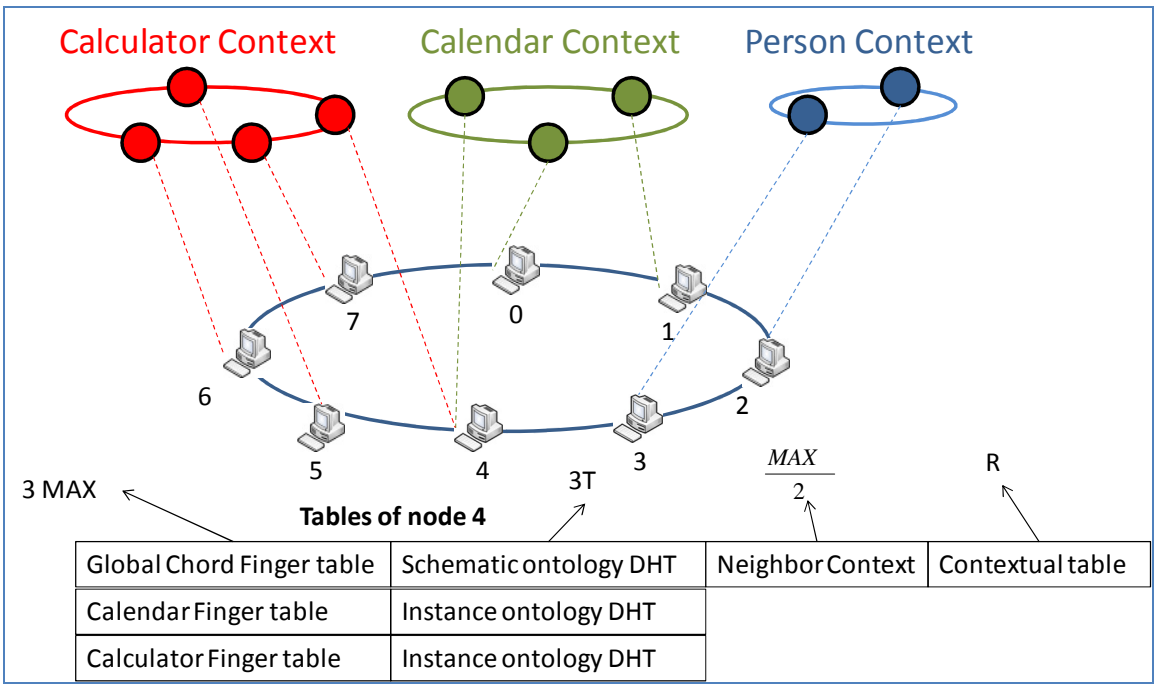
- S หมายถึง จำนวนซูเปอร์โหนดที่ถูกเลือกไว้สำหรับรูปแบบ MF-P2P แบบที่มีการใช้ซูเปอร์โหนดหลายตัวในแต่ละบริบท
- MAX หมายถึง จำนวนคู่อันดับสูงสุดที่เกิดขึ้นในตาราง finger table ซึ่งในที่นี้จะมีค่าเท่ากับ 256 คู่อันดับตามมาตรฐานของ Chord

โดยในการคำนวณหน่วยความจำของระบบ ML-Chord เกิดจากตาราง finger table และตาราง DHT ซึ่งใช้ในการเก็บออนโทโลยี ซึ่งกระจายอยู่ในแต่ละบริบทอย่างเท่ากัน



รูปที่ ง-1 หน่วยความจำที่ใช้สำหรับ ML-Chord

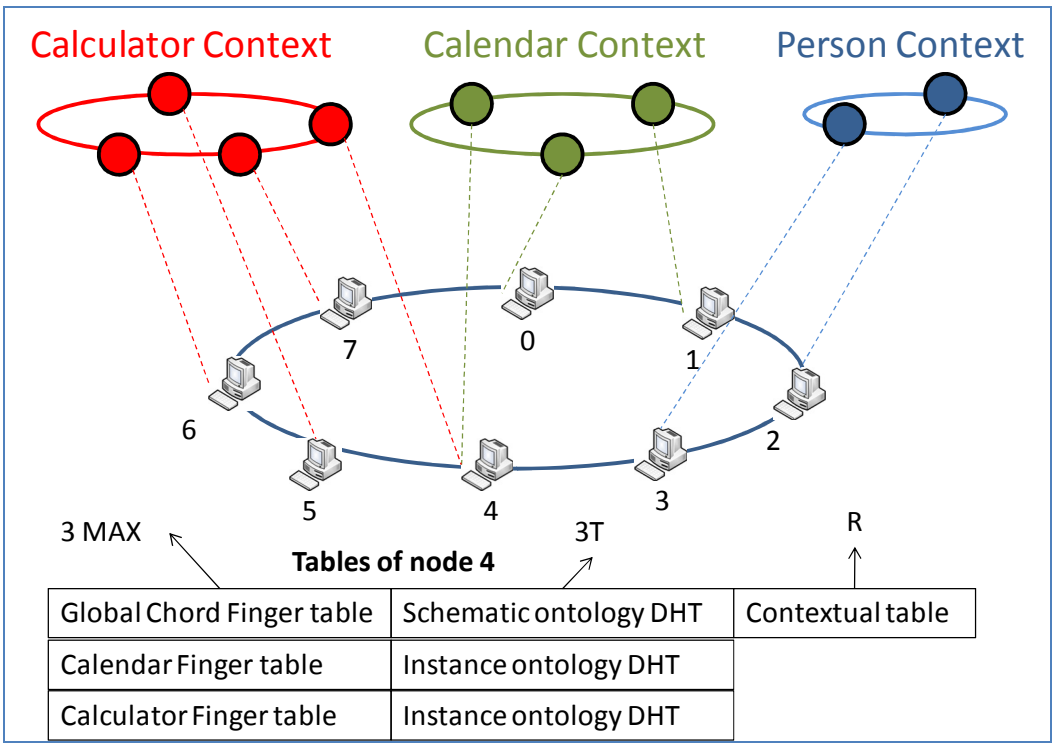
รูปที่ ง-1 แสดงให้เห็นถึงจำนวนตารางทั้งหมดที่จะต้องใช้ สำหรับระบบ ML-Chord ซึ่งจะเห็นว่ามีการใช้ finger table ของแต่ละบริบท และภายในแต่ละบริบทเอง โหนดจำเป็นจะต้องกระจายข้อมูลออนโทโลยีของตนเองทั้งหมดเข้าไปใน DHT ดังนั้นจึงส่งผลให้สมการของหน่วยความจำที่เกิดขึ้นเป็น C * (MAX + T)



รูปที่ ง-2 หน่วยความจำที่ใช้ในระบบ MF-P2P 1

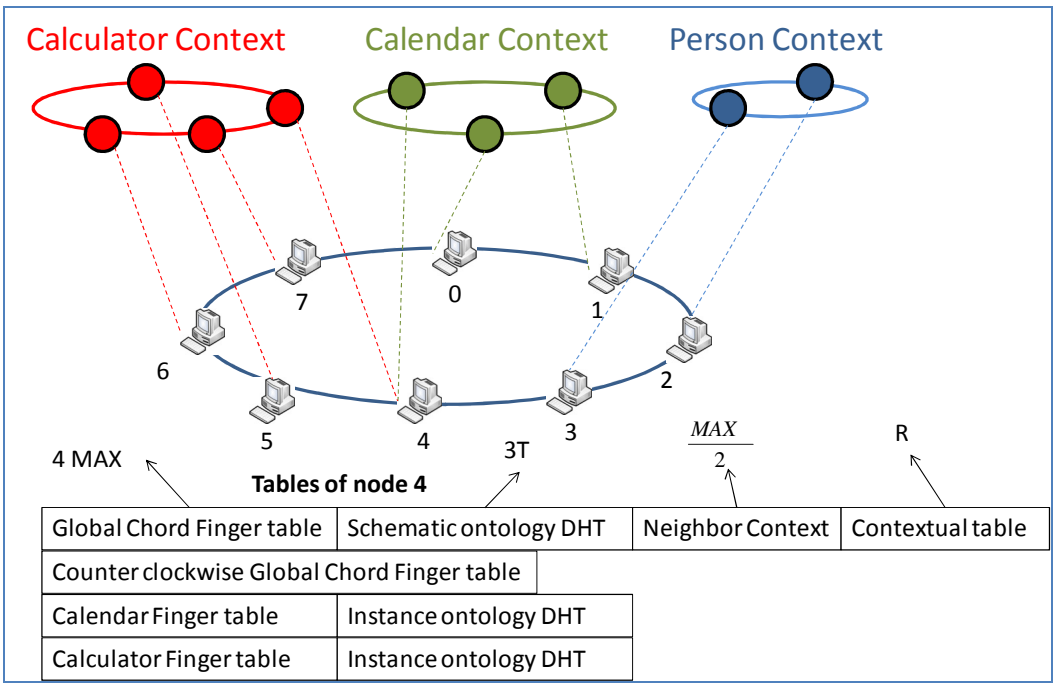
สำหรับหน่วยความจำที่ใช้ในระบบ MF-P2P 1 เกิดจาก Finger table ของแต่ละบริบทรวมทั้งชั้นของ Global Chord ด้วยดังแสดงในรูปที่ ง-2 และออนโทโลยีจะถูกกระจายเข้าไปในระบบเพียงแค่ออนโทโลยีเดียวเท่านั้น เนื่องจากออนโทโลยีที่กระจายใน Global Chord และ Context-aware Chord เกิดจากออนโทโลยีของโหนดมาถูกกระจายให้อยู่ในรูปของ schematic และ instance แล้วกระจายเข้าไปในระบบ MF-P2P แต่อย่างไรก็ตามข้อมูลออนโทโลยีจะถูกกระจายออกไปสามครั้งตามที่ได้ออกแบบการกระจายออนโทโลยี จึงทำให้จำนวนของออนโทโลยีที่ต้องกระจายมีค่าเป็น 3T และข้อมูลที่ต้องถูกเก็บเพิ่มเติมขึ้นมาของระบบ MF-P2P คือข้อมูลบริบทของโหนดข้างเคียง (neighbor context) ซึ่งข้อมูลนี้จะถูกเก็บเป็นปริมาณสูงสุดคงที่คือ

และข้อมูลส่วนสุดท้ายคือข้อมูลของ contextual table ซึ่งจะทำหน้าที่ในการเก็บข้อมูลของบริบทที่เกี่ยวข้องข้างเคียง ซึ่งจำนวนที่เก็บจะถูกติดไว้ในรูปของตัวแปร ส่งผลให้สมการสุดท้ายสำหรับข้อมูลที่ต้องเก็บใน MF-P2P 1 นั่นคือ MAX (+ C) + 3T + R



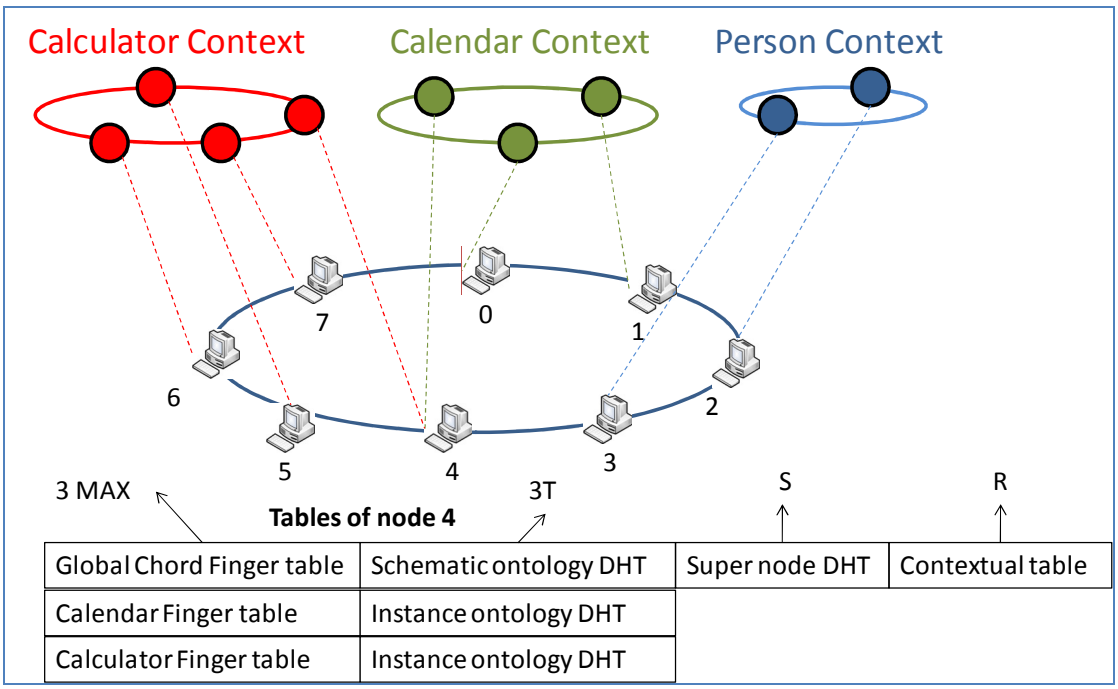
รูปที่ ง-3 หน่วยความจำที่ใช้ในระบบ MF-P2P 2

สำหรับหน่วยความจำที่ใช้ในระบบ MF-P2P 2 เป็นดังที่แสดงในรูปที่ ง-3 ซึ่งจะเห็นว่าข้อมูลทั้งหมดประกอบไปด้วยข้อมูลจาก finger table ของบริบทต่าง ๆ รวมทั้งของ Global Chord เองด้วย และค่าของ DHT ที่เพิ่มขึ้นจากกการเก็บออนโทโลยีซึ่งมีค่าเท่ากับ 3T และข้อมูลของ contextual table ซึ่งใช้ในการเก็บข้อมูลของบริบทข้างเคียง ดังนั้นสมการสุดท้ายที่ใช้ในการเปรียบเทียบหน่วยความจำของระบบ MF-P2P 2 คือ MAX (C + 1) + 3T + R



รูปที่ ง-4 หน่วยความจำที่ใช้ในระบบ MF-P2P 3

สำหรับระบบ MF-P2P 3 จะมีการจัดเก็บข้อมูลที่เท่ากันกับ MF-P2P 1 แต่จะมีข้อมูลเพิ่มเติมขึ้นมาคือข้อมูลของ finger table ของ Global Chord ดังแสดงในรูปที่ ง-4 ซึ่งข้อมูลที่เพิ่มขึ้นมาใช้เพื่อ การเก็บข้อมูลการค้นหาในทิศทางทวนเข็มนาฬิกาด้วย ซึ่งจะทำให้เก็บข้อมูลเพิ่มขึ้น แต่จะเสียเวลาในการค้นหาน้อยลง ดังนั้นจึงทำให้การค้นหาแบบ MF-P2P 3 มีหน่วยความจำที่ต้องเก็บเป็น MAX (3 + C) + 3T + R



รูปที่ ง-5 หน่วยความจำที่ใช้ในระบบ MF-P2P 4

สำหรับการค้นหาแบบ MF-P2P 4 จะมีข้อมูลเพิ่มเติมขึ้นมาจาก MF-P2P 2 นั่นคือข้อมูลของซูเปอร์โหนดที่ต้องเก็บเพิ่มขึ้น มากกว่าเดิมเป็นจำนวน S ดังรูปที่ ง-5 โดยข้อมูลที่เพิ่มขึ้นนี้ไม่ได้เกิดขึ้นในทุกโหนด แต่จะเกิดขึ้นเฉพาะโหนดที่มีการเก็บ DHT ของซูเปอร์โหนดไว้เท่านั้น ซึ่งในระบบทั้งหมดจะมีโหนดที่ต้องเก็บข้อมูลเพิ่มเติมเท่ากับ C โหนด (หรือเป็นจำนวนเท่ากับ จำนวนบริบทที่เกิดขึ้น) และจำนวนข้อมูลที่เก็บเพิ่มมีจำนวนเท่ากับ S คู่อันดับ ซึ่งโหนดที่ต้องเก็บข้อมูลเพิ่มเติมเหล่านั้นจะเก็บข้อมูลเป็นจำนวน เท่ากับ MAX (C + 1) + 3T + S + R

ประวัติผู้เขียน

ชื่อ สกุล นายวศิน เที่ยงคุณากฤต

รหัสประจำตัวนักศึกษา 5010120130

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิศวกรรมศาสตร์บัณฑิต (วิศวกรรมคอมพิวเตอร์)	มหาวิทยาลัยสงขลานครินทร์	2550

ทุนการศึกษา (ที่ได้รับในระหว่างการศึกษา)

ทุนศิษย์ก้นกุฏิ คณะวิศวกรรมศาสตร์ ปี 2550

การตีพิมพ์เผยแพร่ผลงาน

1. W. Thiengkunakrit, S. Kamolphiwong, T. Kamolphiwong, “MF-P2P: A Context Awareness based on structured P2P”, Proceedings of the NECTEC Annual Conference & Exhibitions (NECTEC ACE 2009), Thailand Science Park Convention Center (TSPCC), September 2009, Bangkok, Thailand, pp.
2. W. Thiengkunakrit, S. Kamolphiwong, T. Kamolphiwong, S. Sae-wong, “Enhanced Context Searching based on Structured P2P”, Proceedings of the 2010 International Conference on Communication and Mobile Computing (CMC 2010), April 2010, Shenzhen, China, pp.