การประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สาย

นายศุภเสฏฐ์ ชูชัยศรี

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2551
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

GLOBAL RECURSIVE QUERY PROCESSING FOR WIRELESS SENSOR NETWORKS

Mr.Supasate Choochaisri

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Computer Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2008

Copyright of Chulalongkorn University

หัวข้อวิทยานิพนธ์	การประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สาย
โดย	นายศุภเสฏฐ์ ชูชัยศรี
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.เฉลิมเอก อินทนากรวิวัฒน์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ผู้ช่วยศาสตราจารย์ ดร.เฉลิมเอก อินทนากรวิวัฒน์)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.วิษณุ โคตรจรัส)

..... กรรมการภายนอกมหาวิทยาลัย
(รองศาสตราจารย์ ดร.อนันต์ ผลเพิ่ม)

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ศุภเสฏฐ์ ชูชัยศรี : การประมวลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สาย (GLOBAL RECURSIVE QUERY PROCESSING FOR WIRELESS SENSOR NETWORKS) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ.ดร.เฉลิม เอก อินทนากรวิวัฒน์, 68 หน้า

การประมวลผลข้อคำถามแบบเรียกซ้ำนั้นเป็นการนำข้อดีของการเขียนโปรแกรมเชิงตรรกะมาประยุกต์รวมเข้ากับระบบฐานข้อมูล ทำให้สามารถอนุมานข้อมูลใหม่โดยอาศัยฐานความรู้จากข้อมูลเดิมที่มีอยู่ในระบบได้โดยการสร้างข้อคำถามที่อยู่ในรูปเพรดิเคตข้อเท็จจริง และกฎที่มีอยู่หรือสร้างขึ้นในระบบ ซึ่งในปัจจุบันงานวิจัยทางด้านเครือข่ายตัวรับรู้แบบไร้สายได้มีการออกแบบการส่งข้อคำถามเพื่อเรียกดูข้อมูลในรูปแบบฐานข้อมูลเชิงสัมพันธ์ แต่ยังไม่สามารถรองรับการประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สายได้

วิทยานิพนธ์นี้ได้ทำการออกแบบและสร้างระบบที่เหมาะสมแก่การประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สายโดยเฉพาะ โดยได้ออกแบบแบบจำลองของภาษา วิธีการประมวลผล และสถาปัตยกรรมระบบที่เหมาะสม และมีการสร้างส่วนประมวลผลโลจิกคิวสำหรับตัวรับรู้ไร้สายซึ่งทำหน้าที่ในการรับข้อคำถามเชิงตรรกะ รวมทั้งสร้างระบบเซนส์ทูพีซึ่งเป็นระบบสำหรับการเขียนโปรแกรมมหัพภาพเชิงตรรกะซึ่งมีความสามารถในการประมวลผลข้อคำถามแบบเรียกซ้ำ และมีการใช้เทคนิคที่ช่วยลดการส่งข้อมูลในเครือข่ายตัวรับรู้ไร้สาย เพื่อช่วยให้ตัวอุปกรณ์รับรู้ไร้สายที่มีพลังงานอยู่อย่างจำกัดนั้นใช้พลังงานน้อยลงและสามารถทำงานได้นานขึ้น ในขณะที่ยังสามารถรักษาความถูกต้องและความเกี่ยวข้องของคำตอบได้ครบถ้วน

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....
สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์หลัก .....
ปีการศึกษา 2551

4970612521 : MAJOR COMPUTER ENGINEERING

KEYWORDS : DEDUCTIVE DATABASE/ LOGIC DATABASE/ LOGIC PROGRAMMING/ MACROPROGRAMMING/ RECURSIVE QUERY PROCESSING/ WIRELESS SENSOR NETWORKS

SUPASATE CHOOCHAISRI : RECURSIVE QUERY PROCESSING FOR WIRELESS SENSOR NETWORKS. ADVISOR : ASST. PROF. CHALERMEK INTANAGONWIWAT, Ph.D., 68 pp.

In recent years, wireless sensor networks have been abstracted and programmed as relational database systems for querying environmental data. However, previous works do not support recursive query processing that is specifically designed for wireless sensor networks. Recursive query processing is one of the most powerful features in a deductive database (a relational database that is integrated with logic programming). This feature enables recursive deduction of new data from the existing knowledge base.

This thesis proposes the design and implementation of the global recursive query processing. The design includes the language model, query processing methodology, and system architecture. To realize the proposal, LogicQ and Sense2P are implemented. LogicQ is a subsystem for processing logic queries whereas Sense2P is a system for logic macroprogramming a wireless sensor network as a deductive database. Furthermore, many techniques for prolonging the network lifetime are used in Sense2P. Our simulation results indicates that Sense2P and LogicQ can reduce the communication cost significantly and consume minimal energy as well as maintain completeness and soundness of the answers.

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

Department: Computer Engineering Student's Signature .....
Field of Study: Computer Engineering Advisor's Signature .....
Academic Year: 2008

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้มิอาจสำเร็จสมบูรณ์ได้หากปราศจากความช่วยเหลือ ข้อคิดเห็น คำแนะนำ และข้อคิดในการทำงานวิจัยต่างๆ จากอาจารย์ที่ปรึกษา ผศ.ดร.เฉลิมเอก อินทนากร วิวัฒน์ ซึ่งเป็นอาจารย์ที่ปรึกษาที่เปี่ยมด้วยความสามารถและเป็นแรงผลักดันในการทำวิจัยให้แก่ผู้วิจัยอยู่เสมอมา ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงมา ณ ที่นี้

กราบขอบพระคุณ ศ.ดร.บุญเสริม กิจศิริกุล รศ.ดร.อนันต์ ผลเพิ่ม และ ผศ.ดร.วิษณุ กลับส่ง คณะกรรมการสอบวิทยานิพนธ์ที่สละเวลามาให้ข้อเสนอแนะอันเป็นประโยชน์ต่อการพัฒนาวิทยานิพนธ์ฉบับนี้เป็นอย่างดี

ขอขอบคุณจุฬาลงกรณ์มหาวิทยาลัยที่ได้ทำให้ผู้วิจัยได้รับความรู้ ประสบการณ์ และได้ทำให้ผู้วิจัยได้รู้จักอาจารย์ เพื่อนๆ พี่ๆ น้องๆ ที่แสนดีมากมาย ขอขอบคุณทุกๆคนที่คอยช่วยเหลือ และเติมเต็มประสบการณ์ที่ดีในรั้วมหาวิทยาลัยแก่ผู้วิจัย

ขอบคุณทุกคนในกลุ่มวิจัยยูบิเน็ต ต้น(ศรัณย์) ต้น(วันชัย) พี่อ๊อฟ พี่เต้ พี่เสรี น้องหมิง น้องเบิร์ด น้องโจ๊ก รวมถึง วิช และน้องแท๊ป ที่ช่วยให้งานวิจัยนี้สมบูรณ์เกินกว่าที่ผู้วิจัยได้คาดหวังไว้

ขอขอบคุณ คุณพ่อ คุณแม่ พี่สาว น้องสาว และทุกคนที่บ้านของผู้วิจัยที่คอยสนับสนุนการศึกษาของผู้วิจัยมาโดยตลอด แม้ว่าผู้วิจัยจะทำให้เป็นห่วงอยู่บ่อยครั้ง

และสุดท้ายขอขอบคุณคนที่เข้าใจ และเป็นกำลังใจให้แก่ผู้วิจัยเสมอมา

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

หน้า

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญตาราง

หน้า

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญภาพ

หน้า

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของปัญหา

เครือข่ายตัวรับรู้แบบไร้สาย (Wireless Sensor Networks) นั้นเป็นเทคโนโลยีหนึ่งที่ได้รับความสนใจในการทำวิจัยเป็นอย่างมากในช่วงระยะเวลาหลายปีที่ผ่านมา โดยเครือข่ายตัวรับรู้แบบไร้สายนั้นมีต้นกำเนิดมาจากแนวคิดที่เรียกว่าการประมวลผลที่มีอยู่ทุกหนทุกแห่ง (Ubiquitous Computing) ซึ่งเป็นแนวคิดที่จะนำการประมวลผลของคอมพิวเตอร์หรือหน่วยประมวลผล (Processor) ขนาดเล็กกระจายไปยังทุกๆที่ในสิ่งแวดล้อมรอบๆตัวเรา และเราก็นำข้อมูลต่างๆที่มีอยู่ในสิ่งแวดล้อมเหล่านั้นมาใช้ให้เกิดประโยชน์ โดยที่เราอาจไม่ทันรู้ตัวว่าเรากำลังใช้เทคโนโลยีนี้อยู่ [1]

เครือข่ายตัวรับรู้แบบไร้สายนั้นได้ตอบสนองแนวคิดการประมวลผลทุกหนทุกแห่งเนื่องด้วยเทคโนโลยีนั้นมีลักษณะการทำงานที่สำคัญ คือ เป็นการนำตัวรับรู้หรืออุปกรณ์รับรู้ (Sensor) ซึ่งอาจใช้ในการรับรู้ความเข้มแสง เสียง ความชื้น ฯลฯ ตามแต่ชนิดของตัวรับรู้ ที่มีขนาดเล็กเป็นจำนวนมากไปกระจายไว้ในบริเวณสิ่งแวดล้อมที่เราสนใจ เพื่อที่จะทำการเก็บข้อมูลของสิ่งแวดล้อมที่เราสนใจนั้น [2-5] โดยที่เราไม่จำเป็นที่จะต้องให้ความสำคัญหรือระบุตัวรับรู้ตัวใดตัวหนึ่ง แต่เราจะระบุคุณลักษณะของข้อมูลที่เราต้องการแทน เช่น ข้อมูลว่าบริเวณใดมีฝนตกหนัก เป็นต้น และตัวรับรู้แต่ละตัวที่อยู่ในบริเวณที่เราสนใจนั้นจะช่วยกันเก็บข้อมูลเพื่อรวบรวมและส่งกลับมาให้เรานำไปทำการวิเคราะห์หรือใช้ประโยชน์ต่อไปเอง โดยที่เราไม่ต้องระบุตัวรับรู้เหล่านั้น

จากความสามารถในการเก็บข้อมูลนี้เองทำให้ในปัจจุบันได้มีการมองเครือข่ายตัวรับรู้แบบไร้สายเป็นเสมือนกับฐานข้อมูลขนาดใหญ่ที่เก็บข้อมูลเอาไว้ [6, 7, 8] โดย Yao และ Gehrke ได้นำเสนอการทำระบบฐานข้อมูลบนเครือข่ายตัวรับรู้แบบไร้สายโดยใช้ชื่อว่า COUGAR [9, 10] ซึ่งได้มีการเสนอแนวคิดในการประมวลผลข้อคำถามภายในเครือข่าย (In-Network Query Processing) แต่ถึงอย่างไรก็ตามสำหรับนักพัฒนาที่ต้องการใช้เพียงแค่ข้อมูลในเครือข่ายตัวรับรู้แบบไร้สายนั้น การเขียนโปรแกรมเพื่อเข้าถึงข้อมูลในเครือข่ายตัวรับรู้แบบไร้สายยังเป็นเรื่องที่ต้องใช้ความรู้ในการเขียนโปรแกรมระดับล็ก เช่น จัดการข้อมูลในระดับชั้นเครือข่าย (Network Layer) เป็นต้น ทำให้เป็นเรื่องยากสำหรับนักพัฒนาที่ต้องการใช้เพียงแค่ข้อมูลในเครือข่ายตัวรับรู้แบบไร้สาย COUGAR จึงได้นำเสนอการเขียนโปรแกรมเชิงประกาศ (Declarative Programming) โดยใช้ภาษาที่มีลักษณะเดียวกันกับภาษา SQL ที่ใช้ในระบบฐานข้อมูลเชิงสัมพันธ์ (Relational Database) ทำให้ผู้เขียนโปรแกรมที่ต้องการเรียกดูข้อมูลไม่จำเป็นต้องรู้ถึงการทำงานในระดับล็ก

TAG [11] เป็นงานวิจัยที่เน้นเรื่องการทำประมวลผลภายในเครือข่าย และใช้ภาษาเชิงประกาศเช่นเดียวกันกับ COUGAR แต่ได้เน้นในเรื่องของการทำภาพรวมกลุ่ม (Aggregation) แบบต่างๆ และเน้นที่การทดลองบนอุปกรณ์ของเครือข่ายตัวรับรู้แบบไร้สายจริง เพื่อศึกษาในเรื่องของการใช้พลังงานที่เหมาะสม และใน TinyDB [12] ซึ่งเป็นระบบฐานข้อมูลเชิงสัมพันธ์ (Relational Database) สำหรับเครือข่ายตัวรับรู้แบบไร้สายนั้นก็ได้นำ TAG มาใช้เป็นส่วนหนึ่งในการสร้างระบบฐานข้อมูลบนเครือข่ายตัวรับรู้แบบไร้สาย แต่ถึงอย่างไรก็ตามก็ยังไม่ได้มีการทำการประมวลผลข้อคำถามแบบเรียกซ้ำ (Recursive Query Processing) ขึ้นมาใช้กับเครือข่ายตัวรับรู้แบบไร้สาย ซึ่งการประมวลผลข้อคำถามแบบเรียกซ้ำนั้นจะสามารถเพิ่มประสิทธิภาพในการสอบถามข้อมูลได้มากยิ่งขึ้น และสามารถนำไปประยุกต์ใช้เป็นระบบฐานข้อมูลนิรนัย (Deductive Database) ซึ่งเป็นฐานข้อมูลแบบตรรกะ (Logic Database) ได้อีกด้วย

DSN [13, 14] เป็นผลงานชิ้นแรกๆที่ได้มีการทำภาษาเชิงประกาศ คอมไพเลอร์ (Compiler) และเครื่องประมวลผลเวลาทำงาน (Runtime Engine) ซึ่งสามารถประมวลผลแบบเรียกซ้ำได้ โดยใช้ภาษา Snlog ที่ถูกออกแบบขึ้นมาโดยมีลักษณะคล้ายกับภาษาดาตาล็อก (Datalog) ซึ่งเป็นภาษาเชิงประกาศสำหรับการเรียกดูข้อมูลแบบไร้กระบวนคำสั่ง (Non-Procedural Query Language) โดยประกอบด้วยเซตของกฎ (Rule) และสัญพจน์ที่ใช้ในการเรียกดูข้อมูล (Query Literal) [15] แต่ว่าจุดประสงค์ของ DSN นั้นมีขึ้นเพื่อครอบคลุมถึงการเขียนโปรแกรมในระดับเครือข่าย (Network Layer) โดยกฎที่เขียนขึ้นนั้นจะเป็นในลักษณะพฤติกรรมเฉพาะที่ (Local Behavior) ของตัวรับรู้แต่ละตัวในเครือข่าย รวมทั้งยังเป็นในลักษณะของการติดต่อสื่อสารในระยะ 1 ช่วงกระโดดของการส่งสัญญาณข้อมูล (1-Hop Communication) เท่านั้น

จากงานต่างๆจะเห็นได้ว่ายังไม่มีงานใดที่ได้เน้นการนำการประมวลผลข้อคำถามแบบเรียกซ้ำมาใช้ในการสอบถามเรียกดูข้อมูลโดยใช้ภาษาเชิงประกาศในระบบฐานข้อมูลบนเครือข่ายตัวรับรู้แบบไร้สายโดยมองเครือข่ายตัวรับรู้แบบไร้สายแบบครอบคลุม (Global) นั่นคือผู้เขียนโปรแกรมไม่จำเป็นต้องมีความรู้ในระดับเครือข่าย และไม่จำเป็นต้องเขียนโปรแกรมให้กับพฤติกรรมเฉพาะที่ของแต่ละตัวรับรู้ ซึ่งการสอบถามเรียกดูข้อมูลแบบเรียกซ้ำโดยใช้ภาษาเชิงประกาศนั้นจะสามารถเพิ่มประสิทธิภาพและการมีความหมาย (Expressiveness) ในการสอบถามข้อมูลเพิ่มขึ้นได้มาก และช่วยให้ผู้เขียนโปรแกรมเล็งผลไปที่ผลลัพธ์ของข้อมูลที่ต้องการเพียงอย่างเดียว ไม่ต้องสนใจการทำงานในระดับเครือข่ายหรือการทำงานเกี่ยวกับฮาร์ดแวร์ ดังนั้นในงานวิจัยนี้จึงจะนำเสนอวิธีการที่ใช้ในกระบวนการประมวลผลข้อคำถามแบบเรียกซ้ำที่จะใช้ในเครือข่ายตัวรับรู้แบบไร้สายขึ้นมาโดยมองเครือข่ายตัวรับรู้แบบไร้สายแบบครอบคลุม โดยจะทำการทดลองบนอุปกรณ์ตัวรับรู้แบบไร้สายที่เป็นแพลตฟอร์ม"โหมต" (Mote) ของมหาวิทยาลัยแคลิฟอร์เนีย เบิร์กลีย์ (UC Berkeley) ซึ่งเป็นอุปกรณ์ที่ใช้จริงในเครือข่าย

ตัวรับรู้แบบไร้สาย เพื่อทดสอบการนำไปใช้งานได้จริง และทำการทดลองผลการทำงานในโปรแกรมจำลอง (Simulator) คือ TOSSIM [16] ซึ่งเป็นโปรแกรมจำลองบนระบบปฏิบัติการ TinyOS [17] เพื่อทดสอบการทำงานในสภาวะที่มีโหมดเป็นจำนวนมากเพื่อให้เห็นถึงความสามารถในการรองรับการเพิ่มขนาดของจำนวนโหมดได้ (Scalability)

1.2 วัตถุประสงค์ของการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อนำเสนอการประมวลผลข้อคำถามแบบเรียกซ้ำสำหรับเครือข่ายตัวรับรู้แบบไร้สายโดยมองแบบครอบคลุม ในลักษณะของภาษาเชิงประกาศ ซึ่งสามารถนำไปใช้ในการเขียนโปรแกรมมหภาพเชิงตรรกะ(Logic Macroprogramming) รวมถึงประยุกต์ใช้เป็นระบบฐานข้อมูลนิรนัย (Deductive Database System) สำหรับเครือข่ายตัวรับรู้แบบไร้สายได้

1.3 ขอบเขตของการวิจัย

1) ใช้ทูโปรล็อก (tuProlog) เป็นส่วนต่อประสานกับผู้ใช้ (User Interface) และทำการคอมไพล์ (Compile) โดยทำงานอยู่บนสถานีฐาน (Base Station)

2) อนุประโยคฮอร์นที่ใช้ในระบบจะเป็นแบบตรรกะอันดับที่หนึ่ง (First-Order Logic) และไม่มีสัญลักษณ์ฟังก์ชัน (Function Symbol) อยู่ในอนุประโยค

3) ไม่สนับสนุนข้อคำถามแบบเรียกซ้ำไม่รู้จบ

4) ไม่สนับสนุนเพรดิเคตที่เป็นนิเสธ (Negation Predicate)

5) คำตอบที่ได้จะเป็นในลักษณะได้ครั้งละ 1 ทิวเพิล (Tuple-at-a-time) เช่นเดียวกันกับภาษาโปรล็อก

6) การทดสอบการทำงานของระบบจะทำบนระบบจำลองที่ไม่มีการสูญเสียหรือการผิดพลาดของข้อมูลในช่องสัญญาณไร้สาย (Wireless Channel)

7) การทำงานของระบบจะทำการหาคำตอบ 1 ครั้งต่อ 1 ข้อคำถาม ซึ่งเหมาะกับโปรแกรมประยุกต์ที่ต้องการหาสภาพเครือข่ายหรือสภาพสิ่งแวดล้อมในขณะที่สนใจ ไม่รองรับข้อคำถามที่ส่งไปครั้งเดียวแล้วรอคำตอบส่งกลับมาเป็นช่วงเวลา

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1) ศึกษาวิธีการประมวลผลข้อคำถามที่มีอยู่ในปัจจุบันในเครือข่ายตัวรับรู้แบบไร้สาย

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

2) ศึกษาวิธีการประมวลผลข้อคำถามแบบเรียกซ้ำแบบต่าง ๆศึกษาวิธีการและอัลกอริทึมในการควบคุมความหนาแน่นของโหมตในระบบเครือข่ายตัวรับรู้แบบไร้สาย

3) ออกแบบวิธีการประมวลผลข้อคำถามแบบเรียกซ้ำสำหรับเครือข่ายตัวรับรู้แบบไร้สาย

4) สร้างโปรแกรมเพื่อใช้ในการประมวลผลข้อคำถามแบบเรียกซ้ำตามที่ได้ออกแบบไว้

5) ทดสอบและเก็บข้อมูลการประมวลผล

6) วิเคราะห์ผลการทดลอง

7) ปรับปรุงแก้ไขในส่วนที่ผิดพลาดหรือสามารถเพิ่มเติมประสิทธิภาพได้

8) สรุปผลและเรียบเรียงวิทยานิพนธ์

1.5 คุณค่าทางวิชาการ

1) สามารถทำให้เครือข่ายตัวรับรู้แบบไร้สายมีความสามารถในการประมวลผลข้อคำถามแบบเรียกซ้ำได้ ซึ่งข้อคำถามแบบเรียกซ้ำจะเพิ่มประสิทธิภาพในการสอบถามข้อมูลมากยิ่งขึ้น โดยเฉพาะการนำมาใช้กับภาษาเชิงประกาศที่จะช่วยให้ผู้ต้องการเรียกดูข้อมูลในระบบนั้นไม่จำเป็นต้องมีความรู้ในการเขียนโปรแกรมระดับลึก

2) เกิดระบบฐานข้อมูลนิรนัยสำหรับเครือข่ายตัวรับรู้แบบไร้สายโดยผสมผสาน (integrate) เข้ากับโปรแกรมทูโปรล็อก (tuProlog)

3) เกิดแนวคิดในการเขียนโปรแกรมมหภาพเชิงตรรกะสำหรับเครือข่ายตัวรับรู้แบบไร้สาย

1.6 ผลงานตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของงานวิทยานิพนธ์ได้รับการตีพิมพ์เป็นบทความวิชาการในหัวเรื่อง ”A System for Using Wireless Sensor Networks as Deductive Databases” โดย ศุภเสฏฐ์ ชูชัยศรี และ เฉลิมเอก อินทนากรวิวัฒน์ ในบันทึกการประชุม “The Fourth IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob’08)” ซึ่งจัดขึ้น ณ International Congress Center เมืองอาวิญง (Avignon) ประเทศฝรั่งเศส ระหว่างวันที่ 12-14 ตุลาคม 2551 ดังภาคผนวก ข

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ฐานข้อมูลนิรนัย และภาษาดาตาล็อก

ฐานข้อมูลนิรนัยนั้นเป็นฐานข้อมูลที่นอกจากจะทำการเก็บข้อมูลต่าง ๆในลักษณะเดียวกันกับฐานข้อมูลเชิงสัมพันธ์ (Relational Database) ไว้แล้ว ยังมีกลไกที่สำหรับใช้ในการให้เหตุผลเกี่ยวกับข้อมูลที่มีอยู่ โดยการเก็บกฎที่ใช้สำหรับการอนุมาน (Inference) เพื่อให้เกิดข้อมูลใหม่ขึ้นมา [18]

ซึ่งฐานข้อมูลนิรนัยนั้นมีการนำไปใช้มากในระบบผู้เชี่ยวชาญ (Expert System) หรือระบบสนับสนุนการตัดสินใจ (Decision Support System) เนื่องจากฐานข้อมูลนิรนัยนั้นจะมีการนำการให้เหตุผลด้านตรรกะเข้าไปใช้กับข้อมูลที่มีอยู่ ทำให้ช่วยในการกรองข้อมูลที่ไม่ต้องการออกไปได้ และยังเหมาะกับงานที่มีลักษณะใช้การกระตุ้น (Trigger) เพื่อให้เกิดการทำงาน

จากลักษณะดังกล่าวนี้เองจึงเหมาะกับการนำมาใช้ในเครือข่ายตัวรับรู้แบบไร้สาย เพราะว่ามีการประยุกต์ใช้เครือข่ายตัวรับรู้แบบไร้สายจำนวนมากในการทำระบบการตรวจตรา (Monitoring System) ซึ่งมักจะเกิดการทำงานเฉพาะอย่างขึ้นก็ต่อเมื่อมีเหตุการณ์ (Event) ที่ผิดปกติเข้ามากระตุ้นให้เกิดการทำงานขึ้น และยังช่วยกรองข้อมูลที่ไม่จำเป็นทิ้งไปได้เป็นจำนวนมาก ส่งผลให้ลดการใช้พลังงานที่ไม่จำเป็น เพราะเครือข่ายตัวรับรู้แบบไร้สายนั้นจะมีพลังงานอยู่อย่างจำกัด

ภาษาที่นิยมใช้ในฐานข้อมูลนิรนัยคือภาษาดาตาล็อก (Datalog) [19] ซึ่งเป็นเซตย่อยของภาษาโปรล็อก (Prolog) ที่เป็นภาษาสำหรับการเขียนโปรแกรมเชิงตรรกะ (Logic Programming) ซึ่งประกอบด้วยเซตของข้อเท็จจริง (Fact) และ กฎ(Rule) โดยข้อเท็จจริงนั้นจะบ่งบอกถึงสิ่งที่มีอยู่จริงหรือความสัมพันธ์ที่เกิดขึ้นจริงในโปรแกรม ส่วนกฎนั้นเป็นข้อความซึ่งสามารถใช้ในการนิรนัยหรืออนุมานให้เกิดข้อเท็จจริงใหม่ขึ้นมาโดยอาศัยข้อเท็จจริงเดิมที่มีอยู่ในระบบ โดยตัวอย่างที่เห็นได้ชัดในฐานข้อมูลนิรนัยคือความสัมพันธ์แบบพ่อลูก เช่น มีข้อเท็จจริงคือ “นายแดงเป็นพ่อของนายดำ” กับ “นายขาวเป็นพ่อของนายแดง” และมีกฎอยู่คือ “ถ้านาย X เป็นพ่อของนาย Y และนาย Y เป็นพ่อของนาย Z แล้ว นาย X จะเป็นปู่ของนาย Z” จากกฎนี้ทำให้ได้ข้อเท็จจริงใหม่ซึ่งเกิดจากการนิรนัยคือ “นายขาวเป็นปู่ของนายดำ”

โปรแกรมดาตาล็อกนั้นจะเขียนอยู่ในรูปของอนุประโยคฮอร์น (Horn Clause) ซึ่งเขียนในรูปทั่วไปคือ

L0 :- L1, …, Ln.

แต่ละ Li เป็นสัญพจน์ (Literal) อยู่ในรูปของ pi(t1, …, tki) ซึ่ง pi คือสัญลักษณ์เพรดิเคต (Predicate Symbol) และ tj คือ พจน์ (Term) โดยพจน์นั้นเป็นได้ทั้งค่าคงที่ (Constant) หรือตัวแปร (Variable)

ส่วนที่อยู่ฝั่งซ้ายของข้อความฮอร์นของโปรแกรม Datalog นั้นจะเรียกว่า ส่วนหัว (Head) และส่วนที่อยู่ฝั่งขวาจะเรียกว่า ส่วนตัว (Body) ซึ่งส่วนตัวนั้นสามารถว่างเปล่าได้ โดยข้อความฮอร์นที่มีส่วนตัวว่างเปล่านั้นจะใช้แสดงว่าเป็น “ข้อเท็จจริง” ในขณะที่ข้อความฮอร์นที่มีอย่างน้อย 1 สัญพจน์ในส่วนตัวจะใช้แสดงว่าเป็น “กฎ”

ดังนั้นจากข้อเท็จจริงความสัมพันธ์ “นายแดงเป็นพ่อของนายดำ” สามารถเขียนแทนได้ด้วย father(dang,dum) และกฎ คือ “ถ้านาย X เป็นพ่อของนาย Y และนาย Y เป็นพ่อของนาย Z แล้ว นาย X จะเป็นปู่ของนาย Z” สามารถเขียนแทนได้ด้วย grandfather(Z,X) :- father(Z,Y), father(Y,X). โดยที่ father และ grandfather นั้นเป็นสัญลักษณ์เพรดิเคต สัญลักษณ์ dang กับ dum นั้นเป็นค่าคงที่ และสัญลักษณ์ X, Y และ Z นั้นเป็นตัวแปร และสำหรับสัญพจน์, ข้อเท็จจริง, หรือข้อความฮอร์นใดๆที่ไม่มีตัวแปรอยู่เลยจะเรียกว่ามูล (Ground)

สำหรับโปรแกรมดาตาล็อกนั้นจะมีข้อกำหนดคือ ทุกๆสัญพจน์ที่เป็นสัญลักษณ์เพรดิเคตเดียวกันนั้นจะต้องรับพารามิเตอร์เป็นจำนวนเท่ากันและประเภทเดียวกัน และโปรแกรมดาตาล็อกจะต้องสอดคล้อง (Satisfy) กับเงื่อนไขความปลอดภัย (Safety Condition) ดังนี้

1. ทุกๆข้อเท็จจริงของโปรแกรมดาตาล็อกจะต้องเป็นมูล

2. ทุกๆตัวแปรที่เกิดขึ้นในส่วนหัวของกฎของโปรแกรมดาตาล็อกจะต้องเกิดขึ้นในส่วนตัวของกฎเดียวกัน

เงื่อนไขความปลอดภัยนี้จะรับประกันได้ว่าเซตของข้อเท็จจริงที่จะเกิดจากโปรแกรมดาตาล็อกนั้นจะมีจำนวนจำกัด ยกเว้นในกรณีที่เป็นเพรดิเคตประเมินผล (Evaluator Predicate) ที่มีลักษณะในการบอกขอบเขต เช่น >, < ฯลฯ นั้น ค่าตัวแปรที่อยู่ในเพรดิเคตประเมินผลจะต้องมีการยึดเหนี่ยว (Bind) ไว้กับตัวค่าคงที่หรือตัวแปรที่จะเกิดการแทนด้วยค่าคงที่ในเพรดิเคตอื่น

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาษาดาตาล็อกนี้เป็นที่นิยมในการนำมาใช้กับระบบฐานข้อมูลซึ่งเป็นระบบฐานข้อมูลนิรนัย โดยเหมาะกับข้อมูลที่มีจำนวนข้อเท็จจริงอยู่เป็นจำนวนมาก โดยจะแบ่งเป็นฐานข้อมูลเป็น 2 รูปแบบ คือ

1. เซตของข้อเท็จจริงแบบมูล (Ground Fact) เรียกว่า ฐานข้อมูลแบบแจกแจง (Extensional Database) หรือ EDB ซึ่งจะมีการเก็บอยู่ในฐานข้อมูลจริง โดยเพรดิเคตใดที่ใช้ในการบ่งบอกถึงฐานข้อมูลแบบแจกแจงก็จะเรียกว่าเป็น เพรดิเคต EDB

2. เซตของกฎ เรียกว่า ฐานข้อมูลแบบรวมคุณสมบัติ (Intensional Database) หรือ IDB ซึ่งจะไม่มีการเก็บไว้จริง แต่จะอยู่ในลักษณะของทรรศนะ (View) ชั่วคราว โดยเพรดิเคตใดที่ใช้ในการบ่งบอกถึงฐานข้อมูลแบบรวมคุณสมบัติก็จะเรียกว่าเป็น เพรดิเคต IDB

โดยปกติแล้วนั้นโปรแกรมดาตาล็อกซึ่งประกอบด้วยกฎต่างๆ เมื่อเริ่มทำงานก็จะหาคำตอบทุกคำตอบที่สอดคล้องกับกฎที่เราต้องการ ดังนั้นอาจมองว่าโปรแกรมดาตาล็อกเป็นข้อคำถามเพื่อหาข้อมูลที่ต้องการออกมาจาก ฐานข้อมูลแบบแจกแจงก็ได้ แต่บางครั้งผู้ใช้งานอาจไม่ได้ต้องการทุกคำตอบแต่ต้องการเพียงแค่คำตอบบางส่วนที่สนใจเท่านั้น ดังนั้นโปรแกรมดาตาล็อกจะสามารถระบุเป้าหมาย (Goal) ลงไปได้ โดยเป้าหมายนั้นจะเขียนในรูปของสัญพจน์ที่นำหน้าด้วยเครื่องหมายอัศเจรีย์ (?) และเครื่องหมายยัติภังค์ (-) เช่น ?-grandfather(dang,X) เพื่อจะขอคำตอบเฉพาะคนที่เป็นปู่ของนายแดงเท่านั้น

ในระบบฐานข้อมูลนิรนัยจะมีการตีความ (Interpretation) ของข้อความฮอร์นในระบบในหลายแบบ โดยข้อความฮอร์นนั้นอาจให้ค่าความจริงเป็น"จริง"ในการตีความแบบหนึ่งแต่อาจเป็น"เท็จ"ในอีกรูปแบบหนึ่ง ถ้าข้อความฮอร์นใดให้ค่าความจริงเป็น"จริง"ภายใต้การตีความที่กำหนดมาให้ จะเรียกว่าการตีความที่กำหนดมานั้นสองคล้อง (Satisfy) กับข้อความฮอร์น (โดยการตีความนั้นสามารถมองได้ว่าเป็นการแทนค่าให้กับทุกๆตัวแปรที่ปรากฏอยู่ในกฎก็ได้)

งานวิจัยนี้จะทำการมองเครือข่ายตัวรับรู้แบบไร้สายเป็นเสมือนกับฐานข้อมูลนิรนัย โดยรายละเอียดนั้นอยู่ในหัวข้อ 3.2

2.1.2 กลยุทธ์การประเมินค่า (Evaluation Strategy) ในระบบฐานข้อมูลนิรนัย

ระบบฐานข้อมูลนิรนัยนั้นเป็นการมองระบบฐานข้อมูลให้อยู่รูปฐานข้อมูลแบบตรรกะ และมีความสามารถในการประมวลผลข้อคำถามแบบเรียกซ้ำได้ ซึ่งในการสร้างระบบฐานข้อมูลนิรนัยโดยทั่วไปที่นิยมกันจะสามารถแบ่งได้เป็น 2 วิธี คือ การทำแบบบนลงล่าง และแบบล่างขึ้นบน [15, 18] ซึ่งแต่ละแบบนั้นก็มีข้อดีข้อเสียที่แตกต่างกันไป สำหรับการทำแบบบน

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ลงล่างนั้นจะเริ่มทำจากเป้าหมายที่ต้องการแล้วทำการแบ่งเป็นปัญหาย่อยๆ และทำการแก้ปัญหาย่อยๆเหล่านั้นให้ครบก็จะได้ผลลัพธ์ของเป้าหมายที่ต้องการ

ในขณะที่การทำจากล่างขึ้นบนนั้นจะเริ่มทำจากข้อเท็จจริงที่มีอยู่ แล้วนำข้อเท็จจริงเหล่านั้นมาสร้างข้อเท็จจริงใหม่โดยอาศัยกฎที่ใช้อ้างอิง ที่ต้องการเพื่อให้ได้มาซึ่งเป้าหมายที่ต้องการ แล้วนำข้อเท็จจริงเหล่านั้นมาสร้างข้อเท็จจริงใหม่ขึ้นเรื่อยๆ จนได้เป้าหมายที่ต้องการ

ข้อดีของวิธีการทำจากบนลงล่างนั้นคือสามารถสร้างได้ง่าย และจะมีประสิทธิภาพมากถ้าผู้ใช้ได้ทำการระบุเป้าหมายที่ต้องการมากับข้อคำถาม เช่น มีการกำหนดค่าคงที่ที่ต้องการบางส่วนมาในเป้าหมาย เพราะจะช่วยให้กรองข้อเท็จจริงที่ไม่เกี่ยวข้องทิ้งไปได้มาก ส่วนข้อเสียคือการทำแบบนี้จะเป็นในลักษณะทำครั้งละ 1 ทิวเพิล (tuple at a time) ทำให้ประสิทธิภาพในการทำงานต่ำลง และบางครั้งอาจทำให้โปรแกรมทำงานไม่มีวันจบสิ้นในกรณีที่มีการเกิดการวนซ้ำตัวเองขึ้น ส่วนข้อดีของวิธีการทำจากล่างขึ้นบนคือเป็นการทำครั้งละ 1 เซต (set at a time) ทำให้มีประสิทธิภาพในการทำงานที่มากขึ้น แต่ก็มีข้อเสียคือเนื่องจากว่าการทำแบบล่างขึ้นบนนั้นไม่ใช่การเริ่มต้นจากปัญหาที่ต้องการจริงๆ ดังนั้นจะเกิดการสร้างข้อเท็จจริงที่ไม่เกี่ยวข้องกับเป้าหมายที่ต้องการขึ้นมาเป็นจำนวนมาก และนอกนั้นการตรวจสอบหาข้อผิดพลาดนั้นก็ทำได้ยากกว่าแบบบนลงล่างมาก

2.2 งานวิจัยที่เกี่ยวข้อง

โครงการ COUGAR [7] โครงการวิจัยนี้เป็นงานของกลุ่มนักพัฒนาฐานข้อมูลของมหาวิทยาลัยคอร์เนล (Cornell) ซึ่งเกิดจากการเล็งเห็นว่าเครือข่ายตัวรับรู้ที่มีอยู่แต่เดิมนั้นไม่สามารถที่จะเปลี่ยนพฤติกรรมของระบบได้อย่างไดนามิก นั่นคือไม่สามารถเปลี่ยนพฤติกรรมตามสภาพแวดล้อมได้ทันท่วงทีเพราะว่าเครือข่ายตัวรับรู้ที่มีอยู่นั้นได้อยู่ลงโปรแกรมไว้ตั้งแต่แรก และจะทำการส่งข้อมูลไปยังตัวที่ใช้เชื่อมต่อกับภายนอก (Front End) โดยข้อมูลนั้นก็จะถูกเก็บเอาไว้เป็นกลุ่มเพื่อใช้ในการร้องขอข้อมูลแบบออฟไลน์ (Offline) เพื่อเอาไปใช้ในการวิเคราะห์ต่อไป และปัญหาอีกอย่างหนึ่งที่สนใจก็คือ ระบบการติดต่อสื่อสารของข้อมูลในเครือข่ายนั้นต้องใช้ทั้งทรัพยากรและพลังงานมากกว่าการที่จะมีการคำนวณภายในตัวตัวรับรู้แต่ละตัวเอง ดังนั้นถ้าสามารถทำให้ตัวรับรู้แต่ละตัวรอเก็บข้อมูลให้ครบบางส่วนที่ต้องการและทำการประมวลผลภายในก่อนแล้วจึงค่อยส่งข้อมูลต่อไป ก็จะเป็นการช่วยให้ลดจำนวนการใช้ทรัพยากรในช่องการติดต่อสื่อสารของเครือข่ายลงไปได้อย่างมาก และยังทำให้ยืดอายุการทำงานของเครือข่ายตัวรับรู้ออกไปได้อีกด้วย

โดยในงานวิจัยนี้ได้นำเสนอแนวคิดที่เรียกว่า การประมวลผลข้อคำถามภายในเครือข่าย (In-Network Query Processing) [9, 10] ซึ่งจะเป็นวิธีการประมวลผลข้อคำถามภายในตัวของตัวรับรู้เอง โดยมีหลักการคือ จะมีตัวรับรู้ตัวหนึ่งที่ทำหน้าที่เป็นเกตเวย์

(Gateway) ติดต่อกับผู้ใช้ที่ต้องการส่งข้อคำถามเข้ามา และตัวรับรู้ที่เป็นตัวเกตเวย์นี้เองจะมีสิ่งที่เรียกว่าตัวหาค่าเหมาะที่สุดสำหรับข้อคำถาม (Query Optimizer) คอยทำหน้าที่ในการวางแผนเกี่ยวกับข้อคำถามที่ผู้ใช้ส่งเข้ามาว่าควรมีการเก็บข้อมูลในลักษณะใด แล้วจึงทำการส่งแผนการเก็บข้อมูล (Query Plan) นั้นไปยังตัวรับรู้ตัวอื่นที่เกี่ยวข้อง ส่วนตัวรับรู้ตัวอื่นๆนั้นพอได้รับแผนการเก็บข้อมูลแล้วก็จะทำการเก็บข้อมูลในส่วนที่ตัวมันต้องทำการเก็บ และจะมีการทำการรวบรวมข้อมูลที่ได้จากตัวรับรู้ตัวอื่นที่เกี่ยวข้องกับตัวมัน มาประมวลผลบางส่วน ก่อนที่จะส่งผลลัพธ์ที่ได้ต่อไปสู่ตัวที่เป็นเกตเวย์ และตัวที่เป็นเกตเวย์นั้นก็จะทำการรวบรวมผลลัพธ์ย่อยๆที่ได้ทั้งหมดนำมาประมวลผลสุดท้ายก่อนที่จะส่งผลลัพธ์กลับไปยังผู้ใช้ที่ทำการส่งข้อคำถามเข้ามา เช่น มีข้อคำถามให้นับจำนวนตัวรับรู้ทั้งหมดที่มีอยู่ในระบบว่ามีทั้งหมดกี่ตัว ถ้าเป็นวิธีปกติที่ไม่ได้ทำการประมวลผลข้อคำถามภายในเครือข่าย ตัวรับรู้แต่ละตัวนั้นก็จะทำการส่งข้อมูลของตัวเองไปบอกตัวรับรู้เพื่อนบ้านและตัวรับรู้เพื่อนบ้านก็จะทำการส่งค่านั้นต่อขึ้นไปเรื่อยๆจนถึงตัวรับรู้ที่ทำหน้าที่เป็นเกตเวย์ ซึ่งทุกๆตัวรับรู้ก็จะทำเช่นนี้ สุดท้ายตัวรับรู้ที่เป็นเกตเวย์จึงทำหน้าที่ประมวลผลโดยนับว่ามีข้อมูลที่ส่งมาบอกตัวมันนั้นทั้งหมดเป็นจำนวนเท่าไร แล้วจึงส่งผลลัพธ์นั้นกลับไปบอกผู้ใช้ที่ส่งข้อคำถามเข้า แต่ถ้าในการประมวลผลข้อคำถามภายในเครือข่ายแล้วนั้น เมื่อตัวรับรู้เพื่อนบ้านแต่ละตัวได้รับข้อมูลมาจะยังไม่ทำการส่งต่อข้อมูลไปในทันที แต่จะทำการนำข้อมูลที่ได้รับนั้นมาบวกเข้ากับข้อมูลของตัวมันเองทำให้เกิดเป็นข้อมูลก้อนใหม่ขึ้นมาก่อน (ในกรณีของข้อคำถามเพื่อนับจำนวนตัวรับรู้ทั้งหมด คือนำจำนวนตัวรับรู้ที่ได้รับมาบวกด้วยหนึ่งคือตัวมันเอง) แล้วจึงค่อยส่งข้อมูลก้อนใหม่นั้นออกไปทีเดียว

ซึ่งในกระบวนการประมวลผลภายในเครือข่ายนั้นจะเป็นหนึ่งในกระบวนการสำคัญในการออกแบบการประมวลผลข้อคำถามแบบเรียกซ้ำสำหรับเครือข่ายตัวรับรู้แบบไร้สาย เนื่องจากการประมวลผลภายในเครือข่ายนั้นจะช่วยในการลดพลังงานที่ต้องใช้ในการประมวลผล เพราะประเด็นด้านพลังงานนั้นเป็นหนึ่งในประเด็นสำคัญสำหรับเครือข่ายตัวรับรู้แบบไร้สายที่มีพลังงานอยู่อย่างจำกัด

งานวิจัย TAG [11] งานวิจัยนี้เป็นผลงานของมหาวิทยาลัยแห่งแคลิฟอร์เนีย เบิร์กลีย์ (UC Berkeley) ร่วมกับนักวิจัยจากอินเทล (Intel) โดยงานวิจัยนี้ได้ทำการสร้างบริการ (Service) สำหรับการเก็บรวบรวมข้อมูลบนเครือข่ายตัวรับรู้แบบไร้สาย ซึ่งมีหลักการคล้ายๆกับ Cougar แต่ว่าใน Cougar นั้นจะไม่ได้กล่าวถึงอัลกอริทึมที่จะนำมาใช้ในการส่งผ่านข้อมูลอย่างชัดเจน และไม่ได้ลงรายละเอียดในประเด็นการใช้พลังงาน

นอกจากนั้นในงานวิจัยชิ้นนี้ยังได้มีการทำส่วนต่อประสานเชิงประกาศ (Declarative Interface) แบบง่ายๆเพื่อใช้ในการเก็บรวมรวมหรือจัดกลุ่มของข้อมูล โดยใช้หลักการเหมือนกับภาษาที่ใช้ในการร้องขอข้อมูลที่นิยมใช้กันในระบบฐานข้อมูลในปัจจุบัน และที่สำคัญคือในงานวิจัยชิ้นนี้นั้นได้มีการทำอัลกอริทึมที่จะใช้ในการกระจายหรือทำคำสั่งที่ได้ร้องขอเข้าไปในเครือข่ายตัวรับรู้โดยที่อัลกอริทึมนี้จะช่วยในการประหยัดทั้งพลังงานและเวลาในการประมวลผล

และยังเหมาะสำหรับสภาพแวดล้อมของเครือข่ายตัวรับรู้ที่มีทรัพยากรอยู่อย่างจำกัด รวมถึงการติดต่อสื่อสารที่สามารถขาดการติดต่อได้อยู่อย่างเสมอ

โดยลักษณะการทำงานของ TAG นั้นจะมีตัวรับรู้ตัวหนึ่งที่ทำหน้าที่เป็นสถานีฐาน (Base Station) ซึ่งมีพลังงานมากและสามารถเก็บข้อมูลได้มาก ชุดคำสั่งที่จะใช้ในการเรียกดูข้อมูลนั้นจะทำการกระจายเข้าไปยังเครือข่ายโดยพ่วงติดกันไปกับโพรโทคอลที่ใช้ในเครือข่ายเฉพาะกิจ (Ad-Hoc Network) ที่มีอยู่แล้ว ซึ่งลักษณะการกระจายจะเป็นลักษณะของการใช้ต้นไม้จัดการเส้นทาง (Routing Tree) ในการกระจายข้อมูล และข้อมูลที่ตัวรับรู้อ่านได้นั้นจะถูกส่งกลับไปยังสถานีฐานที่อยู่ที่รากของต้นไม้ โดยข้อมูลที่จะทำการส่งกลับไปนั้นจะยังไม่ใช่สิ่งที่ต้องการแต่เป็นเพียงแค่ส่วนหนึ่งของคำตอบที่ต้องการเท่านั้น ซึ่งต้องมีการประมวลผลที่ตัวรับรู้แต่ละตัวก่อนที่จะส่งผลขึ้นไปต่อเป็นทอดๆ เช่นเดียวกันกับการประมวลผลภายในเครือข่ายของ COUGAR ที่ได้ทำเอาไว้

งานวิจัย TAG นี้ได้ทำภาษาเชิงประกาศให้มีลักษณะเหมือนกับภาษา SQL ที่ใช้ในระบบฐานข้อมูลที่นิยมกันในปัจจุบัน สำหรับการร้องขอข้อมูลซึ่งข้อมูลที่ได้นั้นจะเป็นลักษณะของกระแส (Stream) ของข้อมูลที่อ่านได้ และยังได้มีการเปรียบเทียบข้อดีที่ทำการประมวลผลภายในแต่ละตัวตัวรับรู้ เทียบกับการส่งข้อมูลทั้งหมดผ่านไปรวมกันที่ สถานีฐาน แล้วค่อยประมวลผลทีเดียวที่ศูนย์กลางว่ามีข้อดีมากกว่าอย่างไรบ้าง และสุดท้ายยังได้มีการแสดงให้เห็นถึงความทนต่อความผิดพร่อง (Fault Tolerance) ของเครือข่ายที่มักจะมีความบกพร่องเกิดขึ้นได้อยู่อย่างเสมอๆ

และใน TinyDB [12] เป็นงานที่พัฒนาต่อจากงานวิจัยเรื่อง TAG ดังที่ได้กล่าวข้างต้น โดยได้นำมาทำเป็นระบบประมวลผลข้อคำถามสำหรับใช้ในการดึงข้อมูลออกมาจากเครือข่ายตัวรับรู้ที่เป็นตัวตัวรับรู้ได้ใช้ระบบปฏิบัติการ TinyOS [17] โดยที่ TinyDB นั้นไม่จำเป็นที่ต้องมีการเขียนภาษา C แบบฝังตัว (Embedded C) เพื่อที่จะเรียกดูข้อมูล แต่ TinyDB นั้นได้จัดเตรียมการเชื่อมต่อด้วยภาษาที่เหมือนกับภาษา SQL ทำให้เราเพียงแค่ระบุสิ่งที่เราต้องการเท่านั้นว่าต้องการข้อมูลอะไร และให้มีการส่งข้อมูลใหม่มาในทุกๆช่วงเวลาเท่าไร

โดยจุดประสงค์ของ TinyDB นั้นคือการทำให้โปรแกรมเมอร์เรียกดูข้อมูลในเครือข่ายตัวรับรู้ได้อย่างสะดวกและง่ายดายยิ่งขึ้น ทำให้สามารถสร้างแอพลิเคชันได้อย่างรวดเร็วยิ่งขึ้น ซึ่งเป็นหนึ่งในแนวคิดเกี่ยวกับการทำภาษาในการเรียกดูข้อมูลเชิงประกาศ (Declarative Language) ที่ผู้ใช้ระบบไม่จำเป็นต้องรู้ถึงการทำงานภายในของเครือข่ายว่าจะไปนำข้อมูลมาอย่างไร และจะต้องมีการขนส่งข้อมูลอย่างไร

ภาษาเชิงประกาศนั้นเป็นหนึ่งในประเด็นที่นักวิจัยในด้านเครือข่ายตัวรับรู้แบบไร้สายได้เล็งเห็นและเริ่มมีงานวิจัยในด้านนี้เช่น FACTS [20, 21] และ DSN [13, 14, 22] โดยทั้ง

สองงานนี้ได้ออกแบบระบบในการเขียนโปรแกรมเชิงประกาศโดยใช้รูปแบบการเขียนโปรแกรมบนพื้นฐานของกฎ (Rule-Based Programming) โดยใน FACTS นั้นได้นำเสนอภาษาในรูปแบบของภาษา Haskell ซึ่งเป็นภาษาเขียนโปรแกรมแบบหน้าที่การทำงาน (Functional Programming Language) ในขณะที่ DSN ได้นำเสนอภาษาในรูปแบบของภาษา Datalog โดยใช้ชื่อว่าภาษา Snlog ซึ่งเป็นภาษาเขียนโปรแกรมแบบตรรกะ (Logic Programming Language) แต่ว่า DSN นั้นได้ออกแบบมาเพื่อเป็นภาษาระดับสูง (High-Level Language) สำหรับการเขียนโปรแกรมทั้งในส่วนของการจัดการข้อมูล และการทำงานในระดับชั้นเครือข่าย ซึ่งผู้ที่เขียนโปรแกรมจำเป็นต้องเขียนการทำงานในชั้นเครือข่ายว่าจะให้มีการส่งข้อมูลกันเช่นไรก่อนที่จะเขียนโปรแกรมเพื่อรวบรวมข้อมูลในระดับชั้นโปรแกรมประยุกต์ (Application Layer) นอกจากนั้นยังเป็นการเขียนเพื่อระบุการทำงานเฉพาะที่ (Local Behavior) สำหรับแต่ละตัวรับรู้อีกด้วย

ถึงแม้ว่าจะมีการวิจัยที่พยายามทำระบบฐานข้อมูลหรือการใช้ภาษาเชิงประกาศบนเครือข่ายตัวรับรู้แบบไร้สาย แต่ว่ายังไม่มีงานใดที่ได้นำภาษาเชิงประกาศมาใช้ในการสอบถามข้อมูลแบบเรียกซ้ำโดยมองเครือข่ายตัวรับรู้แบบไร้สายเป็นฐานข้อมูลนิรนัย โดยมองแบบครอบคลุม (Global) นั่นคือเป้าหมายย่อยแต่ละเป้าหมายในส่วนตัวของกฎนั้นไม่ได้มีความสัมพันธ์กันเฉพาะภายในตัวรับรู้ตัวเดียวหรือตัวรับรู้เพื่อนบ้าน แต่มีความสัมพันธ์กับตัวรับรู้ใดๆก็ได้ในระบบซึ่งเป็นประเด็นสำคัญในงานวิจัยชิ้นนี้

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 3

การออกแบบการประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุม สำหรับเครือข่ายตัวรับรู้แบบไร้สาย

การออกแบบการประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สายในงานวิทยาพนธ์นี้ได้ออกแบบโดยคำนึงปัจจัยสำคัญ 3 ประการด้วยกัน คือ 1) พลังงานที่ใช้ไปในการหาคำตอบสำหรับข้อคำถาม (Energy Consumption) ซึ่งเป็นปัจจัยสำคัญสำหรับเครือข่ายตัวรับรู้แบบไร้สายที่มีพลังงานอยู่อย่างจำกัด 2) ความสมบูรณ์ของคำตอบ (Completeness) คือเซตคำตอบที่ได้จากการส่งข้อคำถามไปนั้น ควรประกอบด้วยคำตอบทั้งหมดที่มีอยู่จริงในระบบ 3) ความสมเหตุสมผลของคำตอบ (Soundness) คือในเซตคำตอบที่ได้จากการส่งข้อคำถามไปนั้น ต้องไม่มีคำตอบที่ขัดแย้งกับกฎหรือทำให้ตรรกะไม่เป็นจริงเจือปนมากับเซตคำตอบ

3.1 ข้อสมมติฐาน (Assumptions)

การออกแบบการประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้แบบไร้สายนี้จะอยู่บนข้อสมมติฐาน ดังนี้

1) ระบบทำงานอยู่บนช่องสัญญาณไร้สาย (Wireless Channel) ซึ่งไม่มีทั้งการสูญเสียแพ็กเก็ต (Packet Loss) และความผิดพลาดในการส่งข้อมูลระดับบิต (Bit Error)

2) ระบบมีกระบวนการส่งข้อมูลแบบเชื่อถือได้ (Reliable Communication)

ข้อสมมติฐานเหล่านี้เพื่อหลีกเลี่ยงปัญหาที่ทำให้เกิดข้อมูลสูญหายในระบบอันเนื่องมาจากสัญญาณวิทยุที่ใช้ในการส่งข้อมูล ซึ่งการป้องกันหรือแก้ไขข้อมูลสูญหายเหล่านี้อยู่นอกเหนือขอบเขตของงานวิทยาพนธ์นี้

3.2 แบบจำลองภาษา

แบบจำลองภาษาที่ใช้การสร้างกฎ หรือข้อคำถามนั้น จะใช้แบบเดียวกันกับภาษาดาตาล็อก (หัวข้อ 2.1.1) ซึ่งเป็นเซตย่อยของภาษาโปรล็อกโดยจะเขียนอยู่ในรูปของข้อความฮอร์น (Horn Clause) โดยมีรายละเอียดของแต่ละสัญลักษณ์ดังนี้

3.2.1 เพรดิเคต (Predicate)

เพรดิเคตนั้นเป็นสัญลักษณ์ที่ใช้ในการแสดงความสัมพันธ์ของข้อมูล เช่น เพรดิเคต temperature(NodeID, Temperature) แสดงความสัมพันธ์ระหว่างหมายเลขของสถานีตัวรับรู้เชื่อมโยงคือ NodeID กับค่าอุณหภูมิที่สถานีตัวรับรู้เชื่อมโยงนั้นอ่านค่าได้คือ Temperature,

เพรดิเคต location(NodeID, X, Y) แสดงความสัมพันธ์ระหว่างหมายเลขของสถานีตัวรับรู้เชื่อมโยง Node ID กับค่าพิกัดแกน X และค่าพิกัดแกน Y ของสถานีตัวรับรู้เชื่อมโยงนั้น เป็นต้น

สัญลักษณ์ที่อยู่ในวงเล็บด้านหลังของชื่อเพรดิตเคต เช่น NodeID, Temperature, X, Y นั้นเรียกว่าอาร์กิวเมนต์ (Argument) ของเพรดิเคต ซึ่งถ้าเขียนอาร์กิวเมนต์ขึ้นต้นด้วยตัวอักษรใหญ่จะหมายความว่าอาร์กิวเมนต์นั้นเป็น"ตัวแปร" ซึ่งสามารถถูกแทนที่ด้วยค่าคงที่ได้ ในขณะนี้ถ้าเขียนอาร์กิวเมนต์ขึ้นต้นด้วยตัวอักษรเล็กนั้นหมายความว่าอาร์กิวเมนต์นั้นเป็น "ค่าคงที่"

ระบบสำหรับการประมวลผลข้อคำถามแบบเรียกซ้ำในวิทยาพนธ์นี้ได้ทำการแบ่งประเภทของเพรดิเคตออกเป็น 3 ประเภท คือ

1) เพรดิเคตที่ผู้ใช้เป็นผู้กำหนดขึ้นมาเอง (User-Defined Predicate)

เพรดิเคตประเภทนี้จะถูกกำหนดขึ้นมาโดยโปรแกรมเมอร์ผู้ใช้ระบบ ซึ่งสามารถกำหนดเป็นความสัมพันธ์ใดๆที่สื่อความหมายตามแต่ที่ผู้ใช้ระบบต้องการ

2) เพรดิเคตทั่วไปที่มีมากับระบบ (General Built-In Predicate)

เพรดิเคตประเภทนี้เป็นเพรดิเคตทั่วไปที่ระบบได้จัดเตรียมไว้ให้โปรแกรมเมอร์ผู้ใช้ระบบได้ทำการใช้โดยไม่ต้องกำหนดเองเพิ่มเติม

ตัวอย่างของเพรดิเคตประเภทนี้ ได้แก่

เพรดิเคต “sum(A, B, S)” ซึ่งใช้สำหรับการคำนวณผลรวมของค่า A และ B โดยให้คำตอบของผลรวมเป็น S หรือ เพรดิเคต “abs(A, Z)” ซึ่งใช้สำหรับการคำนวณค่าสัมบูรณ์ (Absolute Value) ของค่า A โดยให้คำตอบเป็น Z ซึ่งมีค่าเท่ากับ |A| หรือเพรดิเคตประเภท “>”, “<”, “=”, “!=” ซึ่งใช้ในการเปรียบเทียบค่าสองค่า เป็นต้น

3) เพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ (Sensor-Specific Built-In Predicate)

เพรดิเคตประเภทนี้จะเป็นเพรดิเคตที่อิงอยู่บนความสามารถของตัวรับรู้ เช่น เพรดิเคต temperature(NodeID, Temperature) ที่ได้กล่าวไว้ข้างต้นนั้น จะขึ้นอยู่กับความสามารถในการตรวจจับอุณหภูมิของตัวรับรู้ เพรดิเคตอื่นๆที่อยู่ในประเภทนี้ ได้แก่ เพรดิเคต humidity(NodeID, Humidity) เพื่อใช้ในการอ่านค่าความชื้นสัมพัทธ์, เพรดิเคต light(NodeID, Light) เพื่อใช้ในการอ่านค่าความเข้มแสง เป็นต้น

3.2.2 ข้อเท็จจริง (Fact)

ในงานวิทยาพนธ์นี้ข้อเท็จจริงจะหมายถึงข้อเท็จจริงแบบมูล (Ground Fact) เท่านั้น ซึ่งข้อเท็จจริงคือเพรดิเคตที่ทุกๆอาร์กิวเมนต์ถูกแทนด้วยค่าคงที่ ซึ่งข้อเท็จจริงจะเป็นค่าที่แสดงถึงค่าที่ตัวรับรู้ตรวจจับได้จริงที่เกิดจากเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ เช่น temperature(5, 25) หมายถึงค่าอุณหภูมิที่สถานีเชื่อมโยงหมายเลข 5 ตรวจจับได้เท่ากับ 25 องศาเซลเซียส เป็นต้น หรือเป็นค่าที่ผู้ใช้ระบบกำหนดไว้ตายตัวในระบบที่เกิดจากเพรดิเคตที่ผู้ใช้เป็นผู้กำหนดขึ้นมาเอง เช่น location(1, 33, 45) หมายถึงตำแหน่งที่สถานีเชื่อมโยงหมายเลข 1 ตั้งอยู่คือพิกัดแกน X เท่ากับ 33 และพิกัดแกน Y เท่ากับ 45 ในกรณีที่ผู้ใช้ทำการกำหนดตำแหน่งพิกัดตายตัวให้กับสถานีเชื่อมโยงในเครือข่าย

3.2.3 กฎ (Rule)

กฎมีไว้เพื่ออนุมาน (Deduce) ให้เกิดข้อเท็จจริงใหม่ขึ้นมาโดยอาศัยฐานความรู้จากข้อเท็จจริงเดิมที่มีอยู่ภายในระบบ ซึ่งการสร้างกฎนั้นจะเขียนไว้อยู่ในรูปของอนุประโยคฮอร์น (Horn Clause) ซึ่งประกอบด้วยส่วนหัว (Head) และส่วนตัว (Body) เช่นเดียวกับภาษาโปรแกรมดาตาล็อกในหัวข้อ 2.1.1

ตัวอย่างของกฎในโปรแกรมประยุกต์ของเครือข่ายตัวรับรู้แบบไร้สาย เช่น ถ้าเราต้องการรู้ว่าสถานีเชื่อมโยงตัวใดสามารถตรวจจับค่าอุณหภูมิได้มากกว่า 50 องศาเซลเซียส จะสามารถเขียนเป็นกฎได้ดังรูปที่ 3.1

```
hotNode(NodeID) :- temperature(NodeID, Temp), Temp > 50.
```

รูปที่ 3.1 กฎตรวจสอบสถานีเชื่อมโยงที่ร้อน

คำตอบที่ได้รับจากกฎนี้จะเป็นค่าหมายเลขประจำตัวสถานีเชื่อมโยง (NodeID) เฉพาะที่ตรวจจับอุณหภูมิได้เกิน 50 องศาเซลเซียสเท่านั้น

จะเห็นได้ว่ากฎนี้เกิดจากการใช้งานเพรดิเคต 2 ตัวด้วยกัน โดยตัวแรกคือเพรดิเคต “temperature(NodeID, Temp)” ซึ่งเป็น เพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ และเพรดิเคต “>” ซึ่งเป็นเพรดิเคตทั่วไปที่มีมากับระบบ และมีการสร้างเพรดิเคตใหม่ที่ชื่อว่า hotNode(NodeID) ซึ่งเป็นเพรดิเคตที่ผู้ใช้เป็นผู้กำหนดขึ้นมาเอง

ในส่วนตัวของกฎนั้นนอกจากจะเป็นเพรดิเคตประเภทต่างๆแล้ว ยังสามารถเป็นกฎย่อยซ้อนอยู่ด้านในก็ได้เช่นเดียวกัน เช่น ถ้าเราต้องการหาสถานีเชื่อมโยงที่ตรวจจับความเข้ม

แสงได้น้อยกว่าค่าที่เราต้องการ แต่มีเงื่อนไขเพิ่มเติมว่าสถานีเชื่อมโยงนั้นต้องเป็นสถานีเชื่อมโยงที่ตรวจจับค่าอุณหภูมิเกินกว่าค่าที่เรากำหนดไว้ด้วย ซึ่งถ้าเราเขียนกฎตามปกตินั้นจะเขียนได้ดังรูปที่ 3.2

```
interestingNode(NodeID) :- temperature(NodeID,Temp), Temp > 50,
                           light(NodeID,Light), Light < 20.
```

รูปที่ 3.2 กฎตรวจสอบสถานีเชื่อมโยงที่ร้อนและอยู่ในที่มืด

แต่จะเห็นได้ว่าเงื่อนไข temperature(NodeID,Temp), Temp > 50 นั้นเหมือนกับกฎในตัวอย่างก่อนหน้าซึ่งเราได้ทำการสร้างเอาไว้แล้ว ดังนั้นเราสามารถสร้างกฎใหม่โดยอาศัยกฎเดิมที่มีอยู่ได้ดังรูปที่ 3.3

```
interestingNode(NodeID) :- hotNode(NodeID), light(NodeID,Light), Light < 20.
```

รูปที่ 3.3 การสร้างกฎใหม่โดยอาศัยกฎเดิมที่มีอยู่ก่อน

ซึ่งจะเห็นได้ว่าการเขียนกฎในลักษณะนี้จะช่วยให้เราเขียนกฎได้สั้นลงและทำให้คนเรียกใช้งานกฎเข้าใจได้ง่ายขึ้น

นอกจากการเขียนกฎเพื่อให้ได้ข้อเท็จจริงใหม่แล้วนั้น เราอาจใช้กฎในการสร้างสมนาม (Alias) ได้ เช่น ถ้าระบบมีเพรดิเคต coordinate(X,Y) เพื่อบอกพิกัดที่สถานีเชื่อมโยงอยู่นั้น เราสามารถเปลี่ยนสมนามเป็น location(X,Y) เพื่อให้ผู้ใช้งานเข้าใจได้มากขึ้น ดังรูปที่ 3.4

```
location(X,Y) :- coordinate(X,Y).
```

รูปที่ 3.4 การใช้กฎสร้างสมนาม

3.2.4 ข้อคำถาม (Query)

ข้อคำถามนั้นใช้เมื่อผู้ใช้งานต้องการคำตอบกลับคืนมาจาระบบ โดยลักษณะของข้อคำถามนั้นจะขึ้นต้นด้วยเครื่องหมาย ?- แล้วตามด้วยเพรดิเคตที่ต้องการถาม และปิดท้ายด้วย

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

เครื่องหมายจุด เช่น ?-hotNode(NodeID). เพื่อใช้ร้องขอคำตอบที่เป็นหมายเลขประจำตัวสถานีเชื่อมโยงทั้งหมดที่เป็นไปตามกฎ hotNode(NodeID):- tempearature(NodeID,Temp), T > 50.

ข้อคำถามนั้นสามารถแบ่งออกได้เป็น 4 กลุ่มด้วยกัน คือ

1) ข้อคำถามเพื่อใช้ในการตรวจสอบข้อเท็จจริง

ข้อคำถามประเภทนี้จะใช้ในการตรวจสอบว่าข้อเท็จจริงที่ระบุไปนั้นมีอยู่จริงในระบบหรือไม่ โดยทำการระบุอาร์กิวเมนต์ของเพรดิเคตด้วยค่าคงที่ทั้งหมดหรือเป็นเพรดิเคตข้อเท็จจริง เช่น ?- temperature(2, 25). เพื่อใช้ในการตรวจสอบข้อเท็จจริงที่ว่า สถานีเชื่อมโยงหมายเลข 2 นั้นสามารถตรวจจับอุณหภูมิได้เท่ากับ 25 องศาเซลเซียส เป็นจริงหรือไม่

โดยคำตอบของข้อคำถามประเภทนี้นั้นมีเพียง “จริง” หรือ “เท็จ” เท่านั้นไม่ว่าจะมีข้อเท็จจริงนี้อยู่ในระบบกี่คำตอบก็ตาม

2) ข้อคำถามเพื่อใช้ในการรับข้อเท็จจริงทั้งหมด

ข้อคำถามประเภทนี้จะใช้ในการขอคำตอบของเพรดิเคตที่เราระบุทั้งหมดที่มีอยู่ในระบบ ซึ่งคำตอบนั้นสอดคล้องกับเงื่อนไขที่เราระบุไว้ในอาร์กิวเมนต์ของเพรดิเคตในข้อคำถาม เช่น ?- temperature(X,Y). เพื่อใช้ในการรับข้อเท็จจริงทั้งหมดของสถานีเชื่อมโยงใดๆและค่าอุณหภูมิที่สามารถอ่านได้ใดๆ ?- temperature(X,25). เพื่อใช้ในการรับข้อเท็จจริงทั้งหมดของสถานีเชื่อมโยงที่สามารถตรวจจับอุณหภูมิได้ 25 องศาเซลเซียส เป็นต้น

3) ข้อคำถามที่ใช้ในการตรวจสอบกฎ

ข้อคำถามประเภทนี้จะคล้ายกับข้อคำถามที่ใช้ในการตรวจสอบข้อเท็จจริง ยกเว้นแต่ว่าเพรดิเคตที่ใช้ถามนั้นเป็นกฎที่มีอยู่ภายในระบบ เช่น ?- hotNode(5). เพื่อใช้ในการตรวจสอบว่ามีสถานีเชื่อมโยงที่มีหมายเลขประจำตัวเป็น 5 ที่สอดคล้องกับกฎ hotNode(NodeID) :- temperature(NodeID, Temp), Temp > 50. หรือไม่

คำตอบที่ได้จากข้อคำถามที่ใช้ในการตรวจสอบกฎมีเพียง 2 ประเภทเช่นกัน คือ “จริง” และ “เท็จ”

4) ข้อคำถามที่ใช้ในการรับคำตอบทั้งหมดจากกฎ

ข้อคำถามประเภทนี้จะใช้ในการอนุมานข้อเท็จจริงทั้งหมดที่สอดคล้องกับกฎในข้อคำถาม เช่น ?- hotNode(X). เพื่อขอหมายเลขประจำตัวของสถานีเชื่อมโยงทั้งหมดที่มีอยู่ในระบบที่สอดคล้องกับกฎ hotNode(NodeID) :- temperature(NodeID, Temp), Temp > 50.

ซึ่งในข้อคำถามประเภทที่ 3 และ 4 นั้นเราสามารถสร้างข้อคำถามแบบเรียกซ้ำขึ้นมาได้

3.2.5 ข้อคำถามแบบเรียกซ้ำ (Recursive Query)

ข้อคำถามจะเป็นข้อคำถามแบบเรียกซ้ำถ้าเพรดิเคตของข้อคำถามนั้นมีการเรียกกฎที่มีลักษณะของกฎแบบเรียกซ้ำ คือ ในส่วนตัวของกฎนั้นประกอบไปด้วยเพรดิเคตที่มีชื่อเดียวกันกับเพรดิเคตส่วนหัวของกฎอยู่ ดังเช่นในรูปที่ 3.5

```
1) danger(AreaID) :- temperature(NodeID, T), T > 80, area(NodeID, AreaID).

2) danger(AreaID) :- humidity(NodeID, H), H < 40,
                     area(NodeID, AreaID), adjacent(AreaID, AdjAreaID),
                     winddir(AdjAreaID, AreaID), danger(AdjAreaID).

?- danger(X).
```

รูปที่ 3.5 ตัวอย่างกฎสำหรับข้อคำถามแบบเรียกซ้ำ

กฎที่ 1 นั้นเรียกว่าเป็นกรณีฐาน (Base Case) ของกฎแบบเรียกซ้ำ ซึ่งอธิบายคุณลักษณะของพื้นที่ที่อันตราย ซึ่งนิยามตามกฎนี้คือ พื้นที่จะเป็นอันตรายถ้ามีอย่างน้อยหนึ่งสถานีเชื่อมโยงภายในพื้นที่นั้นซึ่งสามารถตรวจจับค่าอุณหภูมิได้เกินกว่า 80 องศาเซลเซียส

กฎที่ 2 นั้นเรียกว่าเป็นกรณีที่มีการเรียกซ้ำ (Recursive Case) นั่นคือในส่วนตัวของกฎจะมีเพรดิเคต danger(AdjAreaID) เป็นเป้าหมายย่อยประกอบอยู่ด้วย ซึ่งถ้าประมวลผลมาถึงเพรดิเคตนี้ก็จะทำการเรียกกฎที่ 1 ซ้ำอีกครั้งโดยส่งอาร์กิวเมนต์ใหม่เข้าไป โดยความหมายของกฎที่ 2 ในตัวอย่างนี้คือ ถ้าพื้นที่ไม่มีสถานีเชื่อมโยงที่ตรวจจับอุณหภูมิถึง 80 องศาเซลเซียส (นั่นคือไม่สอดคล้องกับกฎข้อที่ 1) สถานีเชื่อมโยงนั้นจะถือเป็นพื้นที่อันตรายถ้าพื้นที่ในบริเวณนั้นมีมีสถานีเชื่อมโยงที่ตรวจจับความชื้นสัมพัทธ์ได้ต่ำกว่า 40% (หรืออากาศแห้ง) และมีลมพัดมาจากจากพื้นที่ที่อยู่ในอันตรายเพราะสามารถทำให้ไฟลุกลามมาได้ง่าย

โดยการประมวลผลข้อสอบถามแบบเรียกซ้ำนั้นจะทำการประมวลผลกฎแรกก่อนเสมอเมื่อไม่สอดคล้องจึงไปทำงานกฎต่อไปจนเกิดการเรียกซ้ำ โดยรายละเอียดเกี่ยวกับการประมวลผลนั้นจะกล่าวถึงในหัวข้อ 3.3.1.2

ในระบบที่ได้พัฒนาในงานวิจัยนี้ เมื่อระบบเริ่มทำงาน ผู้ใช้งานระบบสามารถกำหนดข้อเท็จจริงและกฎที่ต้องการใช้ไว้ก่อนที่จะส่งข้อคำถามลงไปยังระบบได้

3.3 สถาปัตยกรรมระบบ (System Architecture)

ระบบที่ออกแบบมาเพื่อรองรับข้อคำถามแบบเรียกซ้ำสำหรับเครือข่ายตัวรับรู้แบบไร้สายในงานวิจัยนี้ได้ทำการแบ่งออกเป็น 2 ส่วนหลัก คือส่วนประมวลผลข้อคำถาม และ ส่วนประมวลผลการเก็บรวบรวมข้อมูล ดังแสดงในรูปที่ 3.6

3.3.1 ส่วนประมวลผลข้อคำถาม (Query Processing Engine)

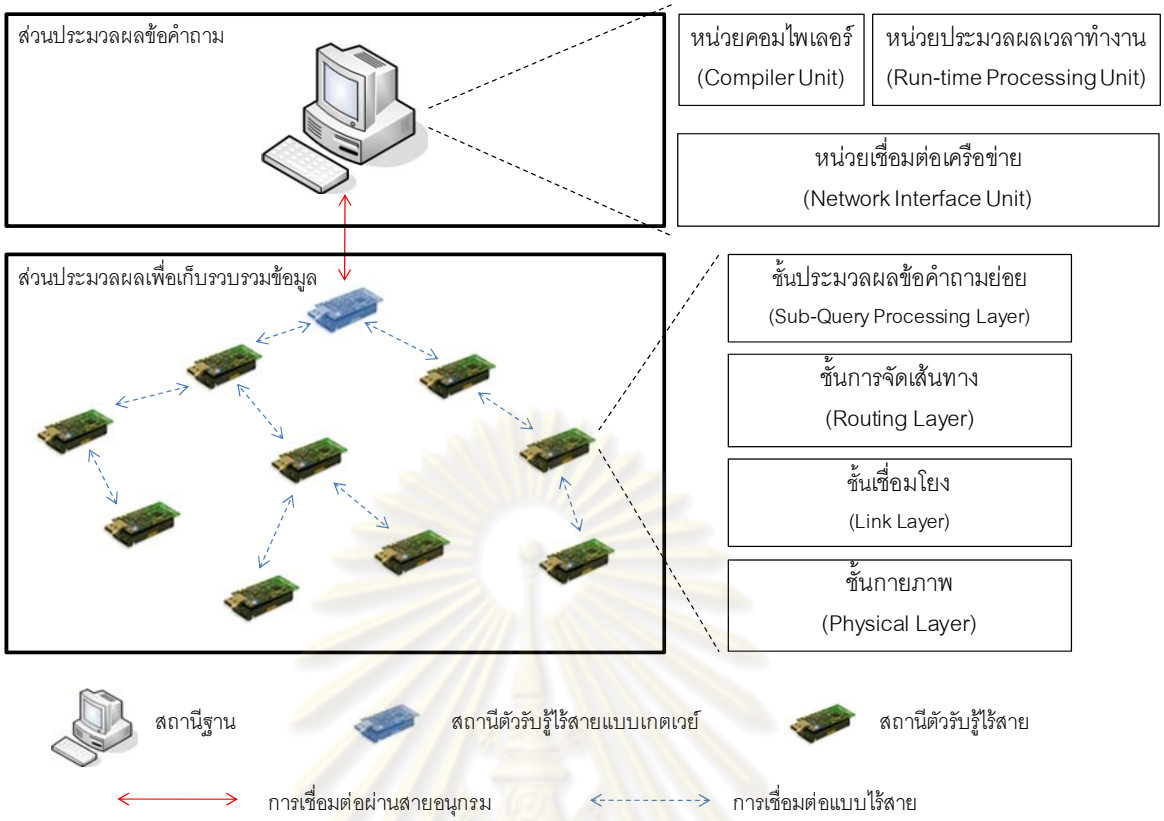
ส่วนนี้เป็นส่วนที่ทำงานอยู่บนสถานีฐาน (Base Station) ซึ่งมีหน้าที่หลักในการติดต่อกับผู้ใช้งานระบบ แปลงโปรแกรมที่ผู้ใช้งานระบบส่งเข้ามาทั้งข้อเท็จจริง กฎ และข้อคำถาม และทำการประมวลผลข้อคำถามนั้นเพื่อส่งคำตอบที่สอดคล้องกับข้อคำถามกลับคืนสู่ผู้ใช้งานระบบ

ในส่วนประมวลผลข้อคำถามนั้นประกอบไปด้วยส่วน 3 ส่วนประกอบสำคัญคือ หน่วยคอมไพเลอร์, หน่วยประมวลผลเวลาทำงาน และหน่วยต่อประสานเครือข่าย โดยจะใช้ระบบทูโปรล็อก (tuProlog) [23] ซึ่งเป็นระบบโปรล็อกที่เขียนขึ้นด้วยภาษาโปรแกรมจาวา (Java) เป็นส่วนประมวลผลฐาน (Base Engine) และครอบคลุมการทำงานในหน่วยคอมไพเลอร์และหน่วยประมวลผลเวลาทำงาน ซึ่งในงานวิจัยนี้จะเน้นไปที่หน่วยประมวลผลเวลาทำงานและหน่วยต่อประสานเครือข่าย

3.3.1.1 หน่วยคอมไพเลอร์ (Compiler Unit)

หน่วยคอมไพเลอร์นั้นมีหน้าที่ในการรับโปรแกรมโปรล็อกที่ประกอบด้วยข้อเท็จจริง กฎ และข้อคำถามมาคอมไพล์ให้อยู่ให้รูปโครงสร้างข้อมูลที่ภาษาเครื่องเข้าใจ และพร้อมที่จะนำไปประมวลผล

เนื่องจากในงานวิจัยนี้ได้ออกแบบภาษาของโปรแกรมโดยใช้ภาษาดาตาล็อกซึ่งเป็นเซตย่อยของภาษาโปรล็อก ดังนั้นจึงไม่ได้เน้นงานวิจัยไปที่หน่วยคอมไพเลอร์ และได้มีการเลือกใช้หน่วยคอมไพเลอร์ของระบบทูโปรล็อกมาใช้ แต่ถึงอย่างไรก็ตามจะต้องมีการแก้ไขในส่วนของหน่วยประมวลผลเวลาทำงานเพื่อให้เหมาะสมกับคุณลักษณะของเครือข่ายตัวรับรู้แบบไร้สายซึ่งจะกล่าวในหัวข้อย่อยถัดไป



รูปที่ 3.6 สถาปัตยกรรมระบบ

3.3.1.2 หน่วยประมวลผลเวลาทำงาน (Run-Time Processing Unit)

ในส่วนหน่วยประมวลผลเวลาทำงานนั้นถือเป็นส่วนที่สำคัญมากส่วนหนึ่งในงานวิจัยนี้ ถึงแม้ว่าการประมวลข้อคำถามแบบเรียกซ้ำนั้นได้มีการศึกษามาเป็นเวลานานแล้ว [15, 18] แต่ว่าในงานวิจัยเหล่านั้นไม่ได้ออกแบบมาเพื่อใช้กับเครือข่ายตัวรับรู้แบบไร้สายโดยเฉพาะ และก่อให้การสูญเสียพลังงานโดยไม่จำเป็น

ในงานวิจัยนี้ได้ทำการพิจารณาถึงกลยุทธ์การประเมินค่าที่ใช้กับข้อคำถามแบบเรียกซ้ำซึ่งเคยมีการวิจัยมาก่อนเพื่อใช้กับฐานข้อมูลนิรนัย (Deductive Database) ซึ่งแบ่งกลยุทธ์การประเมินค่าออกได้เป็น 2 รูปแบบหลักด้วยกัน คือ แบบบนลงล่าง (Top-Down) และแบบล่างขึ้นบน (Bottom-Up) ดังที่ได้กล่าวไว้ในหัวข้อ 2.1.2

งานวิจัยที่ผ่านนั้นได้บอกว่าการใช้กลยุทธ์การประเมินค่าแบบล่างขึ้นบนนั้นได้ผลและประสิทธิภาพที่ดีกว่าแบบบนลงล่าง [15] แต่ในงานวิจัยนี้ได้โต้แย้งให้เห็นว่าสำหรับเครือข่ายตัวรับรู้แบบไร้สายนั้น การใช้กลยุทธ์การประเมินค่าแบบบนลงล่างได้ผลและประสิทธิภาพที่ดีกว่าแบบล่างขึ้นบน

กลยุทธ์การประเมินค่าแบบล่างขึ้นบนนั้นจะต้องเริ่มการทำงานที่ข้อเท็จจริงทั้งหมดที่มีอยู่ในระบบแล้วนำข้อเท็จจริงเหล่านั้นมากรองด้วยกฎทั้งหมดที่กำหนดไว้เพื่อให้เกิดข้อเท็จจริงใหม่ขึ้นมา แล้วนำข้อเท็จจริงใหม่นั้นใส่กลับเข้าไปรวมกับข้อเท็จจริงเดิม แล้วทำกระบวนการเช่นนี้ซ้ำไปเรื่อยๆจนกว่าจะไม่เกิดข้อเท็จจริงใหม่ขึ้นมาในระบบ สุดท้ายจึงเลือกเซตคำตอบจากข้อเท็จจริงทั้งหมดที่สอดคล้องกับกฎที่ได้กำหนดไว้ จะเห็นได้ว่าการประเมินค่าแบบนี้ถ้านำมาใช้กับเครือข่ายตัวรับรู้แบบไร้สายนั้นจะเกิดการสิ้นเปลืองพลังงานเป็นอย่างมาก เนื่องจากข้อเท็จจริงของเครือข่ายตัวรับรู้แบบไร้สายนั้นก็คือค่าต่างๆของสิ่งแวดล้อมที่ต้องใช้ความสามารถในการรับรู้ของสถานีเชื่อมโยงแต่ละตัว ดังนั้นถ้าต้องใช้ข้อเท็จจริงทั้งหมดที่มีอยู่ระบบเป็นค่าเริ่มต้น สถานีเชื่อมโยงทุกตัวจึงต้องทำการส่งข้อมูลข้อเท็จจริงทั้งหมดที่ตนเองมีอยู่กลับขึ้นไปที่สถานีฐานเพื่อประมวลผลเพื่อหาความสัมพันธ์ จึงจะสามารถอนุมานให้เกิดข้อเท็จจริงหรือคำตอบแบบครอบคลุมได้ ซึ่งบางค่านั้นไม่จำเป็นต้องส่งขึ้นไปเนื่องจากไม่มีโอกาสในการสอดคล้องกับกฎ

ถ้าไม่ใช้วิธีการส่งทุกข้อเท็จจริงขึ้นไปยังบนสถานีฐาน การประเมินค่าแบบล่างขึ้นบนอาจทำได้อีกวิธีนึงคือการทำการประมวลทั้งหมดอยู่ในเครือข่าย (In-Network Processing) โดยพยายามหาประมวลผลเพื่อหาคำตอบทั้งหมดในลักษณะแบบกระจาย (Distributed) แต่การประเมินค่าโดยวิธีนี้นั้น ยิ่งมีจำนวนสถานีเชื่อมโยงในเครือข่ายมากเท่าไรอาจทำให้มีการใช้พลังงานที่มากยิ่งขึ้น สาเหตุเนื่องมาจากว่าการเขียนข้อคำถามเรียกซ้ำแบบครอบคลุมนั้น แต่ละเพรดิเคตที่เป็นเป้าหมายย่อยนั้นอาจมีความสัมพันธ์กัน เช่น มีการใช้ค่าตัวแปรตัวเดียวกันในอาร์กิวเมนต์ แต่ว่าข้อเท็จจริงที่เกิดจากเพรดิเคตหนึ่งอาจมาจากสถานีเชื่อมโยงคนละตัวกับข้อเท็จจริงที่เกิดจากเพรดิเคตหนึ่ง ดังนั้นถ้าเราต้องการประเมินค่าภายในเครือข่ายเลยนั้นเพรดิเคตที่มีความสัมพันธ์กันจะต้องค้นหาข้อเท็จจริงที่สัมพันธ์กันที่อยู่ในสถานีเชื่อมโยงใดสถานีหนึ่ง แต่เนื่องจากระบบไม่มีองค์ความรู้ครอบคลุม (Global Knowledge) ทำให้ไม่สามารถรู้ได้ทันทีว่าข้อเท็จจริงที่สัมพันธ์กับของตัวเองเพื่อก่อให้เกิดการสอดคล้องกับกฎนั้นอยู่ที่สถานีเชื่อมโยงใดในเครือข่าย ซึ่งจะเห็นได้ชัดเจนว่าการที่จะหาข้อเท็จจริงนั้นมาได้ค่อนข้างซับซ้อนและต้องใช้พลังงานในการติดต่อสื่อสารเป็นจำนวนมาก และเมื่อยิ่งจำนวนสถานีเชื่อมโยงในเครือข่ายมากขึ้นก็ทำให้การค้นหาข้อเท็จจริงในเครือข่ายยากขึ้นและใช้พลังงานมากขึ้นตามไปด้วย

ในทางกลับกันการเมินค่าแบบบนลงล่างนั้นจะเริ่มทำงานจากข้อคำถามที่ผู้ใช้ส่งเข้ามาและประมวลผลกฎย่อยจากซ้ายไปขวา และตัวแปรในอาร์กิวเมนต์ของกฎที่อยู่ทางขวาจะใช้ค่าที่เกิดจากคำตอบที่สอดคล้องกับกฎทางซ้ายในกรณีที่มีการใช้ตัวแปรร่วมกัน ในกรณีที่ผู้ใช้มีการส่งข้อคำถามที่มีการกำหนดค่าคงที่เป็นอาร์กิวเมนต์ การประเมินผลแบบบนลงล่างจะสามารถนำข้อมูลค่าคงที่นั้นมาใช้ในการกรองข้อเท็จจริงที่แต่ละสถานีเชื่อมโยงจะทำการส่งขึ้นมาได้ แล้วจึงประมวลผลข้อเท็จจริงที่ได้รับทั้งหมดบนสถานีฐาน

ตัวอย่างของการกรองข้อมูลแบบบนลงล่างได้แก่กฎดังรูปที่ 3.7

```
hotObject(ObjectID, AreaID) :- detect(ObjectID, AreaID),
                               temperature(ObjectID, Temp),
                               Temp > 50.

?- hotObjectID(ObjectID, 3).
```

รูปที่ 3.7 ตัวอย่างกฎและข้อคำถามที่ใช้การกรองข้อมูลแบบบนลงล่าง

เมื่อเริ่มกระบวนการประเมินค่า จะเริ่มจากการนำค่าคงที่ที่ระบุไว้ในข้อคำถามคือ AreaID = 3 ไปแทนค่าในเป้าหมายย่อยแรก คือ detect(ObjectID, 3) หลังจากนั้นจึงส่ง detect(ObjectID, 3) นี้ลงไปยังเครือข่ายตัวรับรู้ไร้สาย ซึ่งสถานีเชื่อมโยงแต่ละตัวนั้นจะส่งเฉพาะ ObjectID ซึ่งสามารถตรวจจับได้ใน AreaID เท่ากับ 3 คืนมาเท่านั้น ส่วนที่เหลือไม่จำเป็นต้องส่งขึ้นมาแต่อย่างใด และเมื่อได้เซตคำตอบจากเป้าหมายย่อยแรกแล้ว ระบบจะสามารถนำค่า ObjectID ที่เป็นค่าคงที่ ไปแทนค่าให้เป้าหมายย่อยที่สองคือ temperature เพื่อให้ส่งข้อเท็จจริงเฉพาะที่มีค่า ObjectID ตรงกันเท่านั้น ทำให้ไม่ต้องส่งข้อเท็จจริงทั้งหมดกลับคืนมา ซึ่งช่วยให้ประหยัดพลังงานในการส่งข้อมูลได้มาก

นอกเหนือจากการนำเซตคำตอบของเป้าหมายย่อยก่อนหน้ามาช่วยในการกรองข้อเท็จจริงบนเครือข่ายตัวรับรู้ไร้สายแล้ว ระบบยังสามารถนำค่าที่ถูกกำหนดไว้ด้วยเพรดิเคตทั่วไปที่มีมากับระบบ ได้แก่ =, !=, >, <, >=, <= มาใช้ในการกรองข้อมูลในเครือข่ายตัวรับรู้ไร้สายได้อีกด้วย จากตัวอย่างข้างต้นค่าอุณหภูมิ Temp นั้นถูกกำหนดไว้ด้วย Temp > 50 ซึ่งถ้าเป็นการประเมินค่าแบบที่โปรล็อกทำปกตินั้นจะต้องทำการหาคำตอบทั้งหมดของ temperature(ObjectID, Temp) ออกมาเสียก่อน แล้วจึงค่อยกรองด้วย Temp > 50 ที่สถานีฐาน แต่ในงานวิจัยนี้ได้ทำการผูกเงื่อนไขของค่าคงที่นั้นส่งลงไปยังเครือข่ายตัวรับรู้แบบไร้สายพร้อมกันเลย โดยใช้วิธีในการมองล่วงหน้า (Look Ahead) เพื่อหาค่าที่ถูกกำหนดไว้ด้วยเพรดิเคตทั่วไปที่มีมากับระบบประเภทดังกล่าว แล้วผูกกับค่าตัวแปรที่เกี่ยวข้องที่อยู่ในกฎทั้งหมด ซึ่งวิธีนี้จะช่วยให้ประหยัดพลังงานเพิ่มขึ้นได้อีกมากในกรณีที่มีการกำหนดเงื่อนไขค่าของตัวแปรไว้

ในงานวิจัยนี้ยังได้นำเสนอวิธีการที่ช่วยให้ประหยัดพลังงานเพิ่มเติมด้วยเทคนิคเรียกว่าการแคชแบบซูเปอร์เซต (Superset-Caching) เทคนิคนี้ระบบจะทำการเก็บค่าเซตของคำตอบของเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ ที่เคยได้รับมาจากข้อคำถามก่อนหน้านี้เอาไว้เพื่อเอาไว้ใช้ในโอกาสข้างหน้า และถ้าผู้ใช้ระบบทำการส่งข้อคำถาม

ใหม่เข้ามา แต่ข้อคำถามอันเก่าที่เคยเก็บเอาไว้นั้นเป็นซูเปอร์เซตของข้อคำถามปัจจุบันแล้ว ระบบจะไม่ทำการส่งข้อคำถามลงไปยังเครือข่ายตัวรับรู้ไร้สายเพื่อขอคำตอบเดิมซ้ำ แต่จะใช้คำตอบที่ได้จากการเก็บเอาไว้มาตอบแทน เพราะเซตคำตอบก่อนหน้านั้นย่อมครอบคลุมคำตอบของข้อคำถามปัจจุบัน แต่ถ้าข้อคำถามปัจจุบันไม่ได้อยู่ภายใต้ซูเปอร์เซตที่เคยถามมาแล้วจึงค่อยส่งข้อคำถามใหม่นั้นลงไปยังเครือข่ายตัวรู้ไร้สาย

ตัวอย่างได้แก่ถ้าผู้ใช้เคยส่งข้อคำถามว่า ?- light(X,Y) เข้าไปในระบบและได้คำตอบของข้อคำถามนี้กลับคืนมาแล้ว ถ้าผู้ใช้ส่งข้อคำถามว่า ?- light(3,X) เข้าไปในระบบอีกครั้ง ระบบจะไม่จำเป็นต้องส่งข้อคำถามนี้ลงไปยังเครือข่ายตัวรับรู้แบบไร้สาย เพราะว่าเซตคำตอบของ ?- light(X,Y) นั้นครอบคลุม ?- light(3,X) อยู่แล้ว ดังนั้นระบบจะนำคำตอบเก่าที่เคยเก็บไว้มากรองแล้วตอบไปในทันทีทำให้ไม่สูญเสียพลังงานในการถามคำถามซ้ำ

ถึงแม้ว่าการแคชแบบซูเปอร์เซตจะช่วยประหยัดพลังงานได้มากก็ตามแต่ก็มีข้อเสียอยู่เช่นเดียวกัน นั่นคือในกรณีที่ข้อมูลในสิ่งแวดล้อมมีการเปลี่ยนแปลงบ่อย และผู้ใช้งานต้องการข้อมูลที่ใหม่เสมอทุกครั้ง ซึ่งการทำแคชจะส่งผลให้ผู้ใช้งานระบบได้ข้อมูลเก่าได้ เพราะฉะนั้นจึงต้องมีการใช้ตัวตั้งเวลา (Timer) เพื่อทำการล้างข้อมูลที่เก็บไว้ในแคชทิ้งเพื่อให้ระบบไปร้องขอข้อมูลใหม่จากเครือข่ายตัวรับรู้ไร้สาย ดังนั้นระยะเวลาในการล้างข้อมูลของตัวตั้งเวลานั้นจึงเป็นขอแลกเปลี่ยน (Trade-off) ระหว่างความใหม่ของข้อมูลและพลังงานที่ใช้ ขึ้นอยู่กับการประยุกต์ใช้ของผู้ใช้งานระบบว่าต้องการสิ่งใดมากกว่ากัน เช่น [2] นั้นทำการตรวจจับสิ่งแวดล้อม 1 ครั้งในเวลา 5 นาที ดังนั้นแคชจึงควรถูกล้างข้อมูลทิ้งทุกๆ 5 นาที

3.3.1.3 หน่วยต่อประสานเครือข่าย (Network Interface Unit)

เนื่องจากในการทำงานของระบบจริงนั้นทั้งภาษาและโครงสร้างข้อมูลที่ใช้ในการทำงานของสถานีฐานและเครือข่ายตัวรับรู้ไร้สายนั้นไม่เหมือนกัน ดังนั้นเมื่อต้องมีการส่งข้อมูลแลกเปลี่ยนระหว่างสถานีฐานและเครือข่ายตัวรับรู้ไร้สายนั้น จะต้องมีการทำการแปลงให้อยู่ในรูปที่แต่ละระบบเข้าใจก่อนส่งไปให้ทำงาน และเป็นหน่วยต่อประสานเครือข่ายนี่เองที่จะหน้าที่การแปลงข้อมูลเป็นเส้น (Serialization) และแปลงกลับ (De-serialization) ข้อมูลระหว่างสถานีฐานและเครือข่ายตัวรับรู้ไร้สาย

3.3.2 ส่วนประมวลผลการเก็บรวบรวมข้อมูล (Data Gathering Engine)

ในส่วนนี้จะมีหน้าที่ในการหาคำตอบที่เกี่ยวข้องให้กับแต่ละเป้าหมายย่อยของข้อคำถามที่ผู้ใช้สอบถามเข้ามาในระบบซึ่งจะทำงานอยู่บนอุปกรณ์ตัวรับรู้ไร้สาย โดยส่วนนี้จะประกอบด้วยชั้นประมวลผลข้อคำถาม (Query Processing Layer) ชั้นการจัดเส้นทาง (Routing Layer) ชั้นเชื่อมโยง (Link Layer) และชั้นกายภาพ (Physical Layer) แต่ในงานวิจัยจะเน้นไปที่

การทำงานบนชั้นประมวลผลข้อคำถามและชั้นการจัดเส้นทางเท่านั้น และมีจุดประสงค์เพื่อทำการหาคำตอบที่เป้าหมายย่อยบนสถานีฐานต้องการโดยลดการใช้พลังงานลงให้มากที่สุด

3.3.2.1 การจัดการเส้นทาง

ในการจัดเส้นทางนั้นเมื่อระบบเริ่มทำงานเครือข่ายตัวรับรู้ไร้สายจะทำการสร้างต้นไม้จัดการเส้นทางขึ้น (Routing Tree) เพื่อใช้สำหรับการส่งเป้าหมายย่อยหรือข้อคำถามย่อยที่ได้รับจากสถานีฐานไปยังสถานีเชื่อมโยงแต่ละตัวและใช้ต้นไม้จัดการเส้นทางนี้ในการเก็บรวบรวมข้อมูลหรือคำตอบซึ่งสอดคล้องกับข้อคำถามย่อยนั้น แต่ละสถานีเชื่อมโยงในต้นไม้จัดการเส้นทางนั้นจะมีบทบาทเป็นสถานีผู้ปกครอง (Parent Node) โดยจะมีสถานีลูก (Child Node) กี่สถานีก็ได้ ในขณะเดียวกันตัวสถานีเองก็จะทำหน้าที่เป็นสถานีลูกซึ่งมีสถานีผู้ปกครองเพียงหนึ่งเดียว โดยสถานีเชื่อมโยงเกตเวย์ที่ติดต่อกับสถานีฐานจะเรียกว่าเป็นราก (Root) ของต้นไม้จัดการเส้นทาง โดยในการส่งข้อคำถามลงไปในต้นไม้จัดการเส้นทางนั้นจะใช้การส่งต่อข้อคำถามไปยังต้นไม้ที่สร้างขึ้นนี้ยกเว้นข้อคำถามที่ไม่เกิดประโยชน์ที่สามารถระงับไม่ต้องส่งต่อ เช่น เมื่อได้รับคำตอบที่สอดคล้องกับข้อคำถามที่ต้องการแล้วและไม่จำเป็นต้องได้รับคำตอบอื่นอีก เป็นต้น หลังจากนั้นเมื่อแต่ละสถานีเชื่อมโยงมีคำตอบที่สอดคล้องแล้วก็จะทำการส่งคำตอบกลับไปตามต้นไม้จัดการเส้นทางเดิมนี้ โดยส่งต่อขึ้นไปยังสถานีเชื่อมโยงผู้ปกครองของตนเอง

3.3.2.2 การจัดการข้อคำถามย่อยหรือเป้าหมายย่อย

เพรดิเคตที่เป็นเป้าหมายย่อยหรือข้อคำถามย่อยที่จะส่งเข้ามายังเครือข่ายตัวรับรู้ไร้สายนั้นจะเป็นเพรดิเคตประเภทเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ เช่น temperature(NodeID, Temp) สำหรับการตรวจจับอุณหภูมิ connect(NodeID1, NodeID2) สำหรับการตรวจสอบการเชื่อมต่อระหว่างสองสถานีเชื่อมโยง เป็นต้น ซึ่งเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบเหล่านี้จะต้องทำการระบุไว้ก่อนที่ทำการคอมไพล์โปรแกรมเพื่อจะนำระบบไปติดตั้ง และเมื่อเครื่องประมวลผลการอนุมาน (Inference Engine) ของหน่วยประมวลผลเวลาทำงานทำการประมวลผลข้อคำถามย่อยซึ่งต้องการเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบก็จะทำการส่งข้อคำถามย่อยนั้นลงไปยังเครือข่าย

ถึงแม้ว่าข้อคำถามที่ระบบรับเข้ามานั้นจะแบ่งได้ออกเป็น 4 ประเภท (หัวข้อ 3.2.4) แต่ว่าสำหรับมุมมองของส่วนประมวลผลการเก็บรวบรวมข้อมูลที่อยู่บนแต่ละสถานีเชื่อมโยงในเครือข่ายตัวรับรู้ไร้สายนั้น จะสามารถแบ่งข้อคำถามย่อยที่รับมาจากสถานีฐานได้เพียง 2 ประเภทเท่านั้น คือ ข้อคำถามเพื่อใช้ในการตรวจสอบข้อเท็จจริง และข้อคำถามเพื่อใช้ในการรับข้อเท็จจริงทั้งหมด เนื่องจากข้อคำถามที่จะส่งลงมาจากสถานีฐานนั้นจะเป็นเพียงข้อคำถามย่อยที่ประกอบอยู่ในส่วนตัวของกฎหรือเป็นข้อคำถามที่ถามเกี่ยวกับเพรดิเคตตัวเดียวในระบบโดยตรงเพียงเท่านั้น

ในการตรวจสอบข้อเท็จจริงนั้นข้อคำถามที่รับเข้าไปจะมีทุกอาร์กิวเมนต์เป็นค่าคงที่และต้องการคำตอบเพียงแค่ว่า จริง หรือ เท็จ เพื่อบอกว่าข้อเท็จจริงนั้นมีอยู่ในระบบหรือไม่ เช่น detect(oiltank, area70) นั้นจะทำการตรวจสอบว่าสามารถตรวจจับวัตถุรหัส oiltank ได้ในพื้นที่รหัส area70 หรือไม่ ซึ่งถ้าสถานีเชื่อมโยงตัวใดตรวจจับได้ก็จะตอบกลับไปยังสถานีฐานเพียงว่า จริง หรือถ้าตรวจจับไม่ได้ก็จะตอบกลับไปว่า เท็จ ซึ่งข้อคำถามรูปแบบนี้นั้นส่วนประมวลผลการเก็บรวบรวมข้อมูลจะไม่จำเป็นต้องส่งข้อคำถามไปยังทุกสถานีเชื่อมโยงที่อยู่ในเครือข่าย เพราะเพียงแค่มีอย่างน้อยสถานีใดสถานีหนึ่งมีคำตอบที่สอดคล้องแล้ว ระบบก็ไม่จำเป็นต้องส่งข้อคำถามต่อไปเพื่อขอคำตอบจากสถานีเชื่อมโยงตัวอื่นอีก

ในทางกลับกันข้อคำถามเพื่อใช้ในการรับข้อเท็จจริงทั้งหมดนั้นสถานีเชื่อมโยงแต่ละตัวจะทำการส่งต่อข้อมูลไปทั่วทั้งเครือข่าย เช่น detect(oiltank, AreaID) ซึ่ง oiltank เป็นค่าคงที่ และ AreaID เป็นตัวแปร เพื่อส่งไปถามสถานีเชื่อมโยงทั้งหมดว่าสามารถตรวจจับวัตถุรหัส oiltank ได้ที่พื้นที่รหัสใดบ้าง

3.3.2.3 ขั้นตอนการทำงานของอัลกอริทึม

อัลกอริทึมสำหรับการประมวลผลข้อคำถามย่อยเหล่านี้สามารถแสดงเป็นรหัสเทียมได้ดังรูปที่ 3.8 ซึ่งอธิบายการทำงานได้ดังนี้

ถ้ากรณีที่ข้อคำถามย่อยที่ได้รับนั้นเป็นข้อคำถามเพื่อใช้ในการตรวจสอบข้อเท็จจริง (บรรทัดที่ 1) สถานีเชื่อมโยงจะทำการตรวจสอบข้อเท็จจริงนั้นภายในตัวเองก่อนว่ามีข้อเท็จจริงที่สอดคล้องกับข้อคำถามหรือไม่ (บรรทัดที่ 2) ถ้ามีคำตอบที่สอดคล้องอยู่ สถานีเชื่อมโยงนั้นจะทำการส่งคำตอบว่า “จริง” กลับไปให้สถานีผู้ปกครองของตัวเองทันทีโดยไม่ทำการส่งข้อคำถามต่อไปอีกเนื่องจากว่าข้อคำถามลักษณะนี้ขอเพียงแค่คำตอบว่า “จริง” เพียงหนึ่งคำตอบก็เพียงพอแล้ว (บรรทัดที่ 3) แต่ถ้าไม่มีคำตอบที่สอดคล้องอยู่ก็จะทำการส่งข้อคำถามต่อไปยังสถานีลูกของตัวเอง (บรรทัดที่ 4 – 5)

```
1:  if  subquery_type == ask_for_checking_existence
2:      if have_local_satisfied_fact
3:          send_answer_up(true)
4:      else
5:          forward_query()
6:  else if sub subquery_type == ask_for_all_satisfied
7:      forward_query()
8:      if have_local_satisfied_fact
9:          send_answer_up(answerset)
```

รูปที่ 3.8 รหัสเทียมแสดงอัลกอริทึมการทำงานส่วนประมวลผลเก็บรวบรวมข้อมูล

แต่ถ้าเป็นกรณีข้อคำถามย่อยที่ได้รับนั้นเป็นข้อคำถามเพื่อใช้ในการรับข้อเท็จจริงทั้งหมด (บรรทัดที่ 6) สถานีเชื่อมโยงนั้นก็จะทำการส่งต่อข้อคำถามไปยังสถานีลูกโดยทันที เพราะข้อคำถามนี้ต้องการคำตอบทั้งหมดที่มีในเครือข่ายดังนั้นจึงไม่ต้องรอให้ประมวลผลภายในตัวเองเสร็จก่อนซึ่งเป็นการสูญเสียเวลาโดยไม่จำเป็น (บรรทัดที่ 7) หลังจากส่งข้อคำถามต่อไปแล้วจึงค่อยทำการตรวจสอบว่าในตัวเองนั้นมีคำตอบที่สอดคล้องกับข้อคำถามหรือไม่ (บรรทัดที่ 8) และถ้าพบคำตอบที่สอดคล้องจึงค่อยส่งคำตอบนั้นกลับขึ้นไปยังสถานีผู้ปกครอง (บรรทัดที่ 9)

ด้วยกลไกในการหยุดข้อคำถามที่ไม่จำเป็นนี่เองจะช่วยให้ระบบสามารถลดพลังงานจากการส่งข้อคำถามที่ไม่จำเป็นลงไปได้ ในขณะที่ระบบฐานข้อมูลบนเครือข่ายตัวรับรู้ไร้สาย เช่น ไทนี่ดีบี (TinyDB) นั้นจะยังคงต้องส่งต่อข้อคำถามไปให้ครบทุกสถานีเชื่อมโยงต่อไปแม้ว่าจะพบคำตอบที่ต้องการแล้วก็ตาม โดยความซับซ้อน (Complexity) ของจำนวนข้อความที่ใช้ในกระบวนการทำงานในกรณีแย่ที่สุด (Worst-case) นั้นคือ O(n) โดย n เป็นจำนวนของสถานีเชื่อมโยงที่มีอยู่ทั้งหมดในระบบ นั่นคือส่งข้อคำถามไปยังทุกสถานีเพื่อขอคำตอบทั้งหมดในกรณีที่ไม่มีสถานีเชื่อมโยงระหว่างทางใดมีคำตอบอยู่

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 4

การทำให้เกิดผลของระบบสำหรับการส่งข้อคำถามเรียกซ้ำแบบครอบคลุม

เพื่อเป็นการพิสูจน์แนวคิดที่ได้นำเสนอและเพื่อสำหรับการนำไปประยุกต์ใช้ได้จริง จึงได้มีการทำให้เกิดผล (Implementation) ของระบบสำหรับการส่งข้อคำถามเรียกซ้ำแบบครอบคลุมขึ้น โดยการทำให้เกิดผลนั้นได้แบ่งออกเป็น 3 ส่วนตามสถาปัตยกรรมระบบที่ได้ออกแบบไว้ (หัวข้อ 3.3)

4.1 ระบบย่อยโลจิกคิว (LogicQ)

ระบบย่อยโลจิกคิวเป็นระบบที่ทำงานอยู่บนอุปกรณ์เครือข่ายตัวรับรู้ไร้สายโดยถูกทำให้เกิดผลโดยการใช้ภาษาเนสซี (nesC) [24] ซึ่งเป็นภาษาแบบส่วนประกอบ (Component-Based Language)และทำงานบนระบบปฏิบัติการไทนี่โอเอส (TinyOS) รุ่น 1.1.15

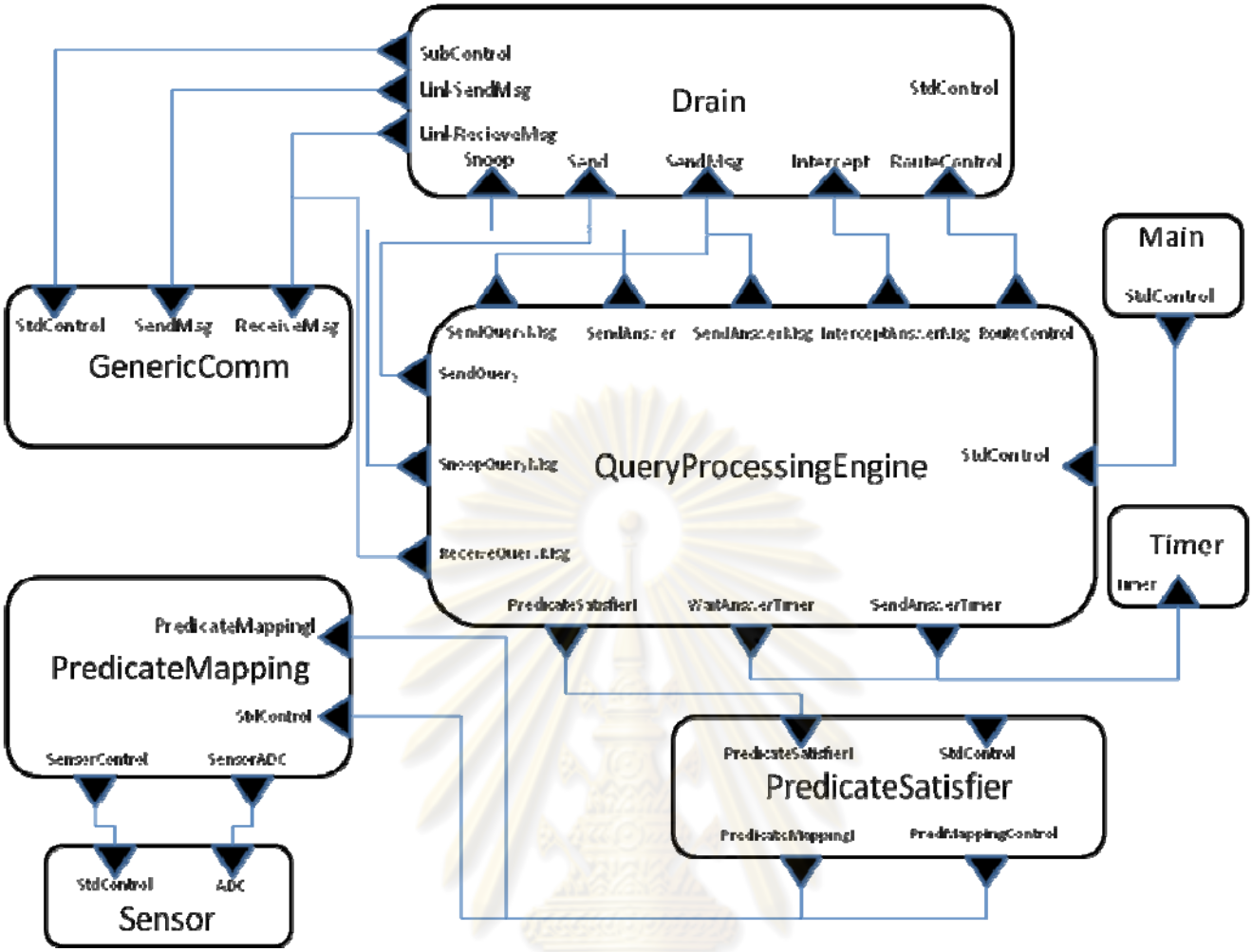
แผนภาพแสดงส่วนประกอบของระบบย่อยโลกจิกคิวแสดงไว้ดังรูปที่ 4.1 ซึ่งประกอบด้วย 8 ส่วนหลักๆด้วยกัน โดยแต่ละส่วนประกอบนั้นมีบริการที่ให้เรียกใช้และเรียกใช้บริการของส่วนประกอบอื่นผ่านทางส่วนต่อประสาน (interface) ของส่วนประกอบนั้นๆ โดยส่วนต่อประสานที่เป็นการให้บริการให้จะแทนด้วยเครื่องหมายลูกศรชี้เข้าหาตัวเอง ในขณะที่ส่วนต่อประสานที่ทำการเรียกใช้บริการของส่วนประกอบอื่นจะแทนด้วยเครื่องหมายลูกศรชี้ออกจากตัวเอง

4.1.1 ส่วนประกอบตัวรับรู้ Sensor

ส่วนประกอบนี้จะทำงานในระดับล่างสุด มีหน้าที่ในการตรวจจับหรือรับรู้ค่าต่างๆจากสิ่งแวดล้อมเมื่อได้รับคำสั่ง ซึ่งในระบบย่อยโลจิกคิวนั้นสามารถมีส่วนประกอบตัวรับรู้ได้มากกว่าหนึ่ง เช่น ส่วนประกอบตัวรับรู้อุณหภูมิ ส่วนประกอบตัวรับรู้ความชื้น ส่วนประกอบตัวรับรู้ตำแหน่ง เป็นต้น โดยส่วนประกอบตัวรับรู้จะมีส่วนต่อประสานชื่อ ADC สำหรับการอ่านค่าที่ได้จากการรับรู้โดยทำการแปลงค่าจากสัญญาณแอนะล็อกเป็นสัญญาณดิจิทัล

4.1.2 ส่วนประกอบการจับคู่เพรดิเคต PredicateMapping

ส่วนประกอบนี้จะให้บริการในการแปลงเพรดิเคตของระบบให้อยู่ในรูปของการทำงานของส่วนรับรู้ที่เกี่ยวข้องกับเพรดิเคตนั้นๆผ่านทางส่วนต่อประสาน PredicateMappingI โดยส่วนประกอบนี้จะทำการเรียกใช้บริการผ่านส่วนต่อประสาน SensorADC ที่เชื่อมต่ออยู่กับส่วนต่อประสาน ADC ของส่วนประกอบตัวรับรู้แต่ละชนิด เช่น เมื่อได้รับเพรดิเคต temperature(X,Y) ก็จะทำการแปลงเพรดิเคตไปเป็นการทำงานของตัวรับรู้ โดยเรียกใช้บริการผ่านส่วนต่อประสาน



รูปที่ 4.1 แผนภูมิส่วนประกอบของระบบย่อยโลจิกคิว

TemperatureADC ที่เชื่อมต่ออยู่กับส่วนต่อประสาน ADC ของส่วนประกอบตัวรับรู้อุณหภูมิ เป็นต้น

4.1.3 ส่วนประกอบการทำให้เพรดิเคตสอดคล้อง PredicateSatisfier

ส่วนประกอบนี้ให้บริการในการตรวจสอบและหาค่าคำตอบของเพรดิเคตที่ทำให้สอดคล้องกับเพรดิเคตข้อคำถามที่กำหนดผ่านทางส่วนต่อประสาน PrecicateSatisfierI ซึ่งจะทำการหาคำตอบเฉพาะภายในสถานีเท่านั้น ส่วนประกอบนี้จะทำการเรียกใช้บริการของส่วนประกอบ PredicateMapping เพื่อขอค่าของแต่ละเพรดิเคตภายในสถานี

4.1.4 ส่วนประกอบติดต่อสื่อสารทั่วไป GenericComm

ส่วนประกอบนี้เป็นส่วนประกอบพื้นฐานที่มีมากับระบบปฏิบัติการไทนี่โอเอส ซึ่งทำหน้าที่ในการส่งและรับข้อมูลผ่านทางช่องสัญญาณวิทยุโดยให้บริการผ่านทางส่วนต่อประสาน SendMsg สำหรับการส่ง และ RecieveMsg สำหรับการรับ

4.1.5 ส่วนประกอบสูบข้อมูล Drain

ส่วนประกอบนี้มีมาพร้อมกับระบบปฏิบัติการไทนี่โอเอสซึ่งทำหน้าที่ในการสร้างต้นไม้จัดการเส้นทาง (Routing Tree) เพื่อใช้ในการสะสมข้อมูลกลับขึ้นสู่สถานีฐาน โดยมีส่วนต่อประสานที่ให้บริการต่างๆ ดังนี้

ส่วนต่อประสาน RouteControl นั้นใช้สำหรับการเรียกดูระดับความลึกของสถานีเชื่อมโยงในต้นไม้จัดการเส้นทาง และสำหรับการตรวจสอบว่าสถานีเชื่อมโยงใดเป็นสถานีเชื่อมโยงผู้ปกครอง

ส่วนต่อประสาน Send ใช้สำหรับการจองบัฟเฟอร์สำหรับการส่งข้อมูลผ่านสัญญาณวิทยุ

ส่วนต่อประสาน SendMsg ใช้สำหรับการส่งข้อมูลที่มีการพักไว้ในบัฟเฟอร์ผ่านทางสัญญาณวิทยุโดยส่งในระยะ 1 ช่วงกระโดด (Hop) ของสัญญาณเท่านั้น ซึ่งเมื่อส่วนต่อประสานนี้ถูกเรียกใช้ จะทำการส่งข้อมูลผ่านทางส่วนต่อประสาน LinkSendMsg เพื่อไปเรียกใช้บริการ SendMsg ของส่วนประกอบ GenericComm

ส่วนต่อประสาน Snoop ใช้สำหรับการรับข้อความที่มีการส่งด้วยส่วนต่อประสาน SendMsg จากสถานีเชื่อมโยงอื่นอันนอกเหนือไปจากข้อความที่เป็นคำตอบที่ส่งกลับสู่สถานีเชื่อมโยง เช่น ข้อความข้อคำถาม เป็นต้น

ส่วนต่อประสาน Intercept ใช้สำหรับการดักจับข้อความที่เป็นคำตอบที่ส่งมาจากสถานีเชื่อมโยงลูก ซึ่งได้ออกแบบเผื่อไว้สำหรับการประหยัดพลังงานโดยการระงับคำตอบบางประเภท หรือการทำภาพรวมกลุ่ม (Aggregation) ของคำตอบก่อนส่งต่อไปยังสถานีเชื่อมโยงผู้ปกครอง เป็นต้น

4.1.6 ส่วนประกอบตัวตั้งเวลา Timer

เป็นส่วนประกอบพื้นฐานของระบบปฏิบัติการไทนี่โอเอส ใช้ประโยชน์ในการตั้งเวลาในการส่งข้อความเพื่อหลีกเลี่ยงการชนกันของข้อความ โดยให้บริการการตั้งเวลาผ่านทางส่วนเชื่อมตัว Timer

4.1.7 ส่วนประกอบหน่วยประมวลผลข้อคำถาม QueryProcessingEngine

เป็นส่วนประกอบที่ทำหน้าที่ในการเชื่อมโยงทุกส่วนประกอบเข้าด้วยกัน และทำการทำงานตามอัลกอริทึมดังรูปที่ 3.8 สำหรับการเก็บรวบรวมข้อมูลสำหรับข้อคำถามย่อยหรือเป้าหมายย่อยที่ส่งมาจากสถานีฐาน

4.1.8 ส่วนประกอบหลัก Main

เป็นส่วนประกอบที่ให้ผู้ใช้เรียกใช้และส่งคำสั่งในการเริ่มต้นหรือหยุดการทำงานของระบบ โดยมีจะมีส่วนต่อประสาน StdControl ที่ทำหน้าที่ในการเชื่อมโยงส่วนต่อประสาน StdControl ของทุกๆส่วนประกอบโดยผ่านทางส่วนประกอบ QueryProcessingEngine ลงไปเป็นลำดับชั้น

4.2 ส่วนต่อประสานเครือข่าย

ส่วนต่อประสานเครือข่ายนั้นเขียนขึ้นด้วยภาษาจาวาเพื่อโดยทำการแปลงข้อความจากระบบไทนี่โอเอสให้อยู่โครงสร้างข้อมูลแบบภาษาจาวา และทำการแปลงข้อความที่จะส่งลงไปยังเครือข่ายตัวรับรู้ไร้สายจากภาษาจาวาให้อยู่ในรูปโครงสร้างข้อมูลของไทนี่โอเอส นอกจากนั้นส่วนต่อประสานเครือข่ายจะทำหน้าที่ในการส่งข้อความที่ใช้ในการสร้างต้นไม้จัดการเส้นทางเมื่อเริ่มต้นระบบ และเป็นตัวกลางในการส่งข้อความข้อคำถามต่างๆลงไปยังเครือข่ายตัวรับรู้ไร้สาย โดยการทำการทั้งหมดนั้นอยู่ภายในคลาส SenseonetBridge โดยมีการ implement MessageListener เพื่อให้สามารถรอรับฟังข้อความที่ส่งกลับมาจากระบบเครือข่ายตัวรู้ไร้สายได้ และทำการส่งข้อความนั้นต่อไปยังระบบที่มาเชื่อมต่อคือเซนส์ทูพี

4.3 ระบบเซนส์ทูพี

ระบบเซนส์ทูพีนั้นทำการอยู่บนสถานีฐานและเป็นการนำระบบทูโปรล็อก (tuProlog) [23] ซึ่งเป็นระบบโปรล็อกที่ประมวลผลจากบนลงล่างมาพัฒนาเพิ่มเติมดังนี้

1) เพิ่มเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ เช่น เพรดิเคต temperature(ID,T), light(ID,L), humidity(ID,H) เป็นต้น

2) แก้ไขกระบวนการทำงานของการประมวลผลขณะทำงานเมื่อเจอเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบให้ทำการติดต่อกับเครือข่ายตัวรับรู้ไร้สายผ่านทางส่วนต่อประสานเครือข่าย

3) ทำการผูกค่าตัวแปรโดยทำการมองหาค่าคงที่ล่วงหน้าในกฎก่อนที่จะประมวลผลเพรดิเคตที่มีตัวแปรนั้นๆ

4) ทำการแคชคำตอบที่เกิดจากเพรดิเคตที่ขึ้นกับความสามารถของตัวรับรู้ที่มีมากับระบบ และเมื่อได้รับข้อคำถามใหม่ ระบบจะทำการตรวจสอบกับแคชที่เคยเก็บไว้ก่อนว่าแคชนั้นเป็นซูเปอร์เซตแคชหรือไม่ และจะส่งลงไปยังเครือข่ายตัวรับรู้ไร้สายถ้าข้อคำถามใหม่นั้นไม่ได้อยู่ในแคช แต่ถ้าอยู่ในแคชจะนำคำตอบเก่าที่เคยถามไปแล้วตอบกลับไปโดยไม่ต้องส่งลงไปถามเครือข่ายตัวรับรู้ไร้สาย

โดยรูปส่วนต่อประสานผู้ใช้ (User Interface) และวิธีการใช้งานระบบเซนส์ทูพีนั้นได้แสดงไว้ในภาคผนวก ก

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 5

ผลการทดลองและวิเคราะห์ผล

ในบทนี้จะทำการวิเคราะห์สมรรถนะของระบบเซนส์ทูพีที่เกิดจากการนำทฤษฎีที่ได้นำเสนองานวิจัยมาทำให้เกิดผล (Implementation) โดยเริ่มจากการกำหนดตัววัดสมรรถนะของระบบ (Performance Metrics) ในแต่ละด้าน กำหนดเครื่องมือที่ใช้และสภาพแวดล้อมในการทดลอง และผลการการทดลองเปรียบเทียบสมรรถนะในแง่มุมต่างๆ คือ การเปรียบเทียบระหว่างการประมวลผลแบบบนลงล่างและล่างขึ้นบน การวัดประสิทธิภาพจากการใช้กระบวนการระงับข้อคำถาม การวัดประสิทธิภาพจากการใช้กระบวนการกรองข้อมูล การวัดประสิทธิภาพจากการใช้ซูเปอร์เซตแคช การวัดความสมบูรณ์ของคำตอบและความเกี่ยวข้องของคำตอบ และการเปรียบเทียบคุณสมบัติของระบบกับระบบฐานข้อมูลอื่น ซึ่งในการทดลองนี้เหล่านี้จะเป็นการแสดงให้เห็นความแตกต่างระหว่างระบบเซนส์ทูพีกับระบบหรือวิธีการอื่นๆ เช่น ระบบไทนี่ดีบี การประมวลผลแบบล่างขึ้นบน หรือระบบเซนส์ทูพีแบบที่ไม่ใส่คุณสมบัติพิเศษที่ช่วยเพิ่มประสิทธิภาพ เป็นต้น

5.1 ตัววัดสมรรถนะของระบบ (Performance Metrics)

ในงานวิจัยนี้ได้ทำการวัดสมรรถนะของระบบใน 3 ด้านด้วยกัน คือ

1) พลังงานที่ใช้ในการติดต่อสื่อสารของระบบ (Energy Consumption)

2) ความสมบูรณ์ของคำตอบ (Completeness)

3) ความเกี่ยวข้องของคำตอบ (Soundness)

สำหรับพลังงานที่ใช้ในการติดต่อสื่อสารของระบบนั้นจะทำการวัดจากจำนวนข้อความ (Message) ที่ส่งทั้งหมดภายในเครือข่าย โดยในระบบใดระบบหนึ่งนั้นจะมีขนาดความยาวของข้อความเท่ากันในทุกๆข้อความที่ทำการส่ง ซึ่งจำนวนข้อความที่ส่งนั้นจะเป็นตัวบ่งชี้ถึงพลังงานที่ใช้ไปในระบบได้เนื่องจากมีความสัมพันธ์กันแบบแปรผันตรงซึ่งกันและกัน ซึ่งระบบจะใช้งานได้เหมาะกับเครือข่ายตัวรับรู้ไร้สายนั้นต้องลดการใช้พลังงานให้มากที่สุด ความสมบูรณ์ของคำตอบและความเกี่ยวข้องของคำตอบนั้นถือเป็นคุณสมบัติที่ต้องมีสำหรับโปรแกรมเชิงตรรกะ โดยความสมบูรณ์ของคำตอบนั้นคืออัตราส่วนระหว่างคำตอบที่ระบบได้รับเทียบกับคำตอบทั้งหมดที่มีอยู่จริงในเครือข่าย ซึ่งจะเป็นตัววัดสมรรถนะที่บอกได้ว่าระบบนี้สามารถรับคำตอบที่มีทั้งหมดได้สมบูรณ์หรือไม่ และเช่นเดียวกัน ความเกี่ยวข้องของคำตอบนั้นวัดจากอัตราส่วนของคำตอบที่เกี่ยวข้องกับข้อคำถามกับจำนวนคำตอบทั้งหมดที่ได้รับ ซึ่งตัววัดสมรรถนะนี้จะเป็นตัวบ่งบอกว่าระบบจะไม่มีคำตอบที่ไม่เกี่ยวข้องกับข้อคำถามเจือปนมาด้วย

5.2 เครื่องมือที่ใช้ในการทดลอง

5.2.1 โปรแกรมจำลองบนไทนี่โอเอส (TinyOS Simulator) หรือ ทอสซิม (TOSSIM)

เนื่องจากอุปกรณ์เครือข่ายตัวรับรู้ไร้สายมีอยู่น้อยทำให้ไม่สามารถวัดประสิทธิภาพเมื่อนำระบบไปใช้จริงซึ่งประกอบด้วยอุปกรณ์จำนวนมากได้ ดังนั้นในการทดลองหลักเพื่อวัดประสิทธิภาพนั้นจะทำการทดลองบนโปรแกรมจำลอง (Simulator) คือทอสซิม

ทอสซิมนั้นเป็นโปรแกรมจำลองที่ถูกออกแบบมาสำหรับจำลองการทำงานของอุปกรณ์บนแพลตฟอร์มโหมต (Mote) ของมหาวิทยาลัยเบิร์กลีย์ (UC Berkeley) ซึ่งทำงานบนระบบปฏิบัติการไทนี่โอเอสโดยเฉพาะและถือเป็นมาตรฐานที่ได้รับการยอมรับและใช้งานกันมากที่สุด (De Facto Standard) โดยทอสซิมนั้นจะสามารถกำหนด จำนวนอุปกรณ์โหมต การเชื่อมต่อทางสัญญาณไร้สาย หรือความน่าจะเป็นในการที่จะเกิดความผิดพลาดในการส่งข้อมูลได้

ในการทดลองของงานวิจัยนี้จะใช้ทอสซิมจำลองให้แต่ละอุปกรณ์โหมตนั้นทำงานโดยใช้ระบบโลจิกคิวที่งานวิจัยนี้ได้นำเสนอเพื่อใช้สำหรับการรวมรวบข้อมูลในเครือข่ายตัวรับรู้ไร้สายเพื่อส่งคำตอบกลับคืนสู่สถานีฐาน (รวมกันเป็นระบบเซนส์ทูพี) และตั้งค่าให้อุปกรณ์หมายเลข 0 นั้นสามารถเชื่อมต่อกับสถานีฐานได้โดยจำลองการเชื่อมต่อแบบผ่านทางพอร์ตอนุกรม (Serial Port)

5.2.2 อุปกรณ์เครือข่ายตัวรับรู้ไร้สายโหมตเทลอสบี (Telos B Sensor Mote)

ถึงแม้ว่าการทดลองบนอุปกรณ์จริงซึ่งมีจำนวนอยู่น้อยนั้นอาจไม่สามารถบอกถึงประสิทธิภาพในการทำงานได้ แต่ใช้เพื่อเป็นข้อพิสูจน์ว่าระบบที่ทำให้เกิดขึ้นจริงนั้นสามารถทำงานได้บนอุปกรณ์จริง โดยอุปกรณ์ที่ใช้ในการทดสอบนั้นเป็นอุปกรณ์ชื่อ เทลอสบี เป็นอุปกรณ์ตัวรับรู้ไร้สายบนแพลตฟอร์มโหมตของมหาวิทยาลัยเบิร์กเลย์ ซึ่งผลิตโดยบริษัทครอสโบว์ (Cross Bow) โดยมีตัวรับรู้ที่ติดมากับอุปกรณ์ 3 ชนิด คือ อุณหภูมิ ความชื้น และความเข้มแสง

เนื่องจากการทดลองบนอุปกรณ์จริงนั้นไม่ได้ทำการวัดผลในตัววัดสมรรถนะต่าง ๆ เนื่องจากอุปกรณ์จริงที่ใช้ทดสอบมีอยู่จำกัด จึงได้ทำการแสดงเพียงการใช้งานบนอุปกรณ์จริงให้ไว้ในภาคผนวก ข ซึ่งแสดงการทดสอบการส่งข้อคำถามแบบเรียกซ้ำบนสถานการณ์จำลอง

5.3 สภาพแวดล้อมที่ใช้ในการทดลอง

การทดลองนั้นได้ใช้ไทนี่โอเอส รุ่น 1.1.15 และจำลองการทำงานของแต่ละสถานีเชื่อมโยงโดยใช้โปรแกรมจำลองทอสซิม โดยการรับส่งข้อมูลนั้นตั้งอยู่บนสมมติฐานคือมีการ

ติดต่อสื่อสารแบบเชื่อถือได้ (Reliable Communication) ได้แก่ ไม่มีแพ็กเก็จสูญหายอันเนื่องมาจากการเกิดความผิดพลาดในการส่งระดับบิตหรือเกิดการชนกันของข้อมูล เพื่อทำการหลีกเลี่ยงปัญหาคำตอบสูญหายที่ไม่ได้เกิดจากการทำงานโดยตรงของระบบ

ในการทดลองได้มีการกำหนดให้สถานีเชื่อมโยงแต่ละตัวสามารถตรวจจับอุณหภูมิและความชื้นของสิ่งแวดล้อม ซึ่งค่าอุณหภูมินั้นจะถูกกำหนดโดยสุ่ม (Random) ให้อยู่ระหว่าง 20 ถึง 80 องศาเซลเซียส และกำหนดให้มีสถานีชื่อมโยงที่สามารถตรวจจับอุณหภูมิได้มากกว่า 50 องศาเซลเซียสอยู่เป็นจำนวนร้อยละ 5 ของจำนวนสถานีเชื่อมโยงทั้งหมดในการทดลองแต่ละครั้ง

5.4 การเปรียบเทียบระหว่างการประมวลผลแบบบนลงล่างและล่างขึ้นบน

การทดลองนี้จะทำการเปรียบเทียบประสิทธิภาพของระบบเซนส์ทูพีซึ่งมีการประมวลผลแบบบนลงล่างดังที่ได้นำเสนอไว้ในงานวิจัยนี้กับวิธีการแบบล่างขึ้นบน โดยในการทดลองแรกนั้นได้ทดลองโดยการส่งข้อคำถามซึ่งเป็นเพรดิเคต temparature เพื่อตรวจสอบหรือขอค่าอุณหภูมิ ซึ่งได้ทดลองกับระบบเซนส์ทูพีจะใช้ข้อคำถาม 3 ประเภทคือ

1) ข้อคำถามประเภทแรกจะเป็นการตรวจสอบการมีอยู่ของข้อเท็จจริง คือทุกๆ อาร์กิวเมนต์ของคำถามนั้นเป็นค่าคงที่

2) ข้อคำถามประเภทที่สองจะเป็นข้อคำถามเพื่อขอคำตอบที่เป็นไปได้ โดยมีการกำหนดให้อาร์กิวเมนต์เป็นค่าคงที่ และเป็นตัวแปร

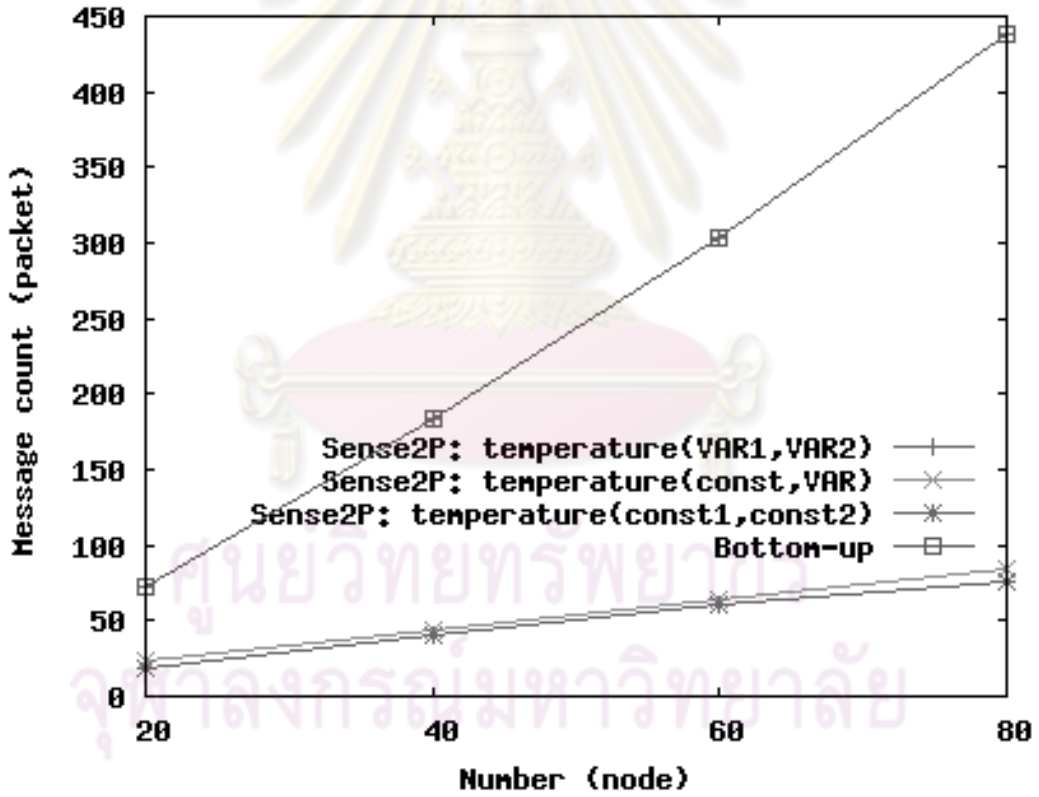
3) ข้อคำถามประเภทที่สามจะเป็นข้อคำถามเพื่อขอคำตอบทั้งหมดที่มีอยู่ในเครือข่าย โดยอาร์กิวเมนต์ทุกตัวนั้นเป็นตัวแปร

ซึ่งในการทดลองนี้เป็นการส่งเพื่อตรวจสอบข้อคำถามแบบง่ายคือเพียงแค่ตรวจสอบกับข้อเท็จจริงที่มีอยู่ในระบบเท่านั้นไม่ได้เป็นการตรวจสอบกฎ โดยจะทำการวัดจำนวนข้อความที่ส่งในเครือข่ายโดยเฉลี่ยเปรียบเทียบกับการประมวลผลแบบล่างขึ้นบน ซึ่งการประมวลผลแบบล่างขึ้นบนนั้นไม่ว่าข้อคำถามจะมีการระบุอาร์กิวเมนต์ในรูปแบบใดก็จะใช้จำนวนข้อมูลในการส่งเท่าเดิมเพราะทุกๆข้อเท็จจริงจะต้องถูกส่งมาไว้ที่สถานีฐานเสมอ

ผลการทดลองได้แสดงไว้ดังรูปที่ 5.1 ซึ่งจะเห็นได้ชัดเจนว่าในระบบเซนส์ทูพีนั้นการระบุค่าคงที่เข้าไปแม้เพียงตัวเดียวนั้นสามารถจะลดจำนวนข้อมูลที่ส่งในเครือข่ายลงได้เป็นอย่างมากเมื่อเปรียบเทียบกับการทำงานแบบล่างขึ้นบนเนื่องจากค่าคงที่นั้นจะไประงับการส่งข้อมูลที่ไม่เกี่ยวข้องกับข้อคำถามของแต่ละสถานีเชื่อมโยง แต่ถึงอย่างไรก็ตามระบบเซนส์ทูพีนั้นจะใช้จำนวนข้อมูลที่ส่งในเครือข่ายเทียบเท่ากับการประมวลผลแบบล่างขึ้นบนในกรณีข้อ

คำถามนั้นต้องการคำตอบที่มีอยู่ทั้งหมดในเครือข่ายเนื่องจากว่าต้องมีการส่งทุกข้อเท็จจริงที่มีอยู่ในเครือข่ายกลับขึ้นมาที่สถานีฐานเช่นเดียวกับการทำงานแบบล่างขึ้นบน

แต่ในการทดลองนั้นได้มีการกำหนดให้ทุกสถานีเชื่อมโยงมีความสามารถในการตรวจจับเหมือนกัน ดังนั้นถ้าในเครือข่ายประกอบด้วยสถานีเชื่อมโยงที่มีความสามารถในการตรวจจับที่แตกต่างกันแล้ว ระบบเซนส์ทูพีนั้นจะสามารถช่วยลดจำนวนข้อมูลที่ส่งในเครือข่ายในมากขึ้นแม้ว่าข้อคำถามนั้นจะเป็นประเภทที่ทุกอาร์กิวเมนต์เป็นตัวแปรก็ตาม เช่น light(X,Y) ก็จะมีเฉพาะสถานีเชื่อมโยงที่ความสามารถในการตรวจจับความเข้มแสงเท่านั้นที่ส่งคำตอบกลับมาให้ ในขณะที่สถานีเชื่อมโยงใดที่ไม่มีความสามารถนี้ก็ไม่จำเป็นต้องส่งข้อมูลกลับมาแต่อย่างใด แต่สำหรับการประมวลผลแบบล่างขึ้นบนแล้วทุกสถานีเชื่อมโยงจำเป็นต้องส่งข้อมูลทั้งหมดในระบบกลับขึ้นมาเพื่อให้หน่วยประมวลผลเวลาทำงานที่อยู่บนสถานีฐานเป็นตัวคัดเลือกข้อเท็จจริงที่เกี่ยวข้องเอง



รูปที่ 5.1 กราฟเปรียบเทียบจำนวนข้อความที่ใช้โดยเฉลี่ยในกรณีที่ส่งข้อคำถามโดยใช้เพรดิเคต temperature(NodeID,T) ในระบบเซนส์ทูพีซึ่งใช้การประมวลผลแบบบนลงล่างและระบบที่ใช้ในการประมวลผลแบบล่างขึ้นบน

สำหรับการทดลองที่สองนั้นจะทำการทดสอบโปรแกรมที่มีการใช้กฎดังรูปที่ 5.2

```
hot(AreaID) :- temperature(NodeID, T), T > 50, area(NodeID, AreaID).
```

รูปที่ 5.2 กฎเพื่อให้ตรวจสอบบริเวณที่สถานีเชื่อมโยงที่ร้อน

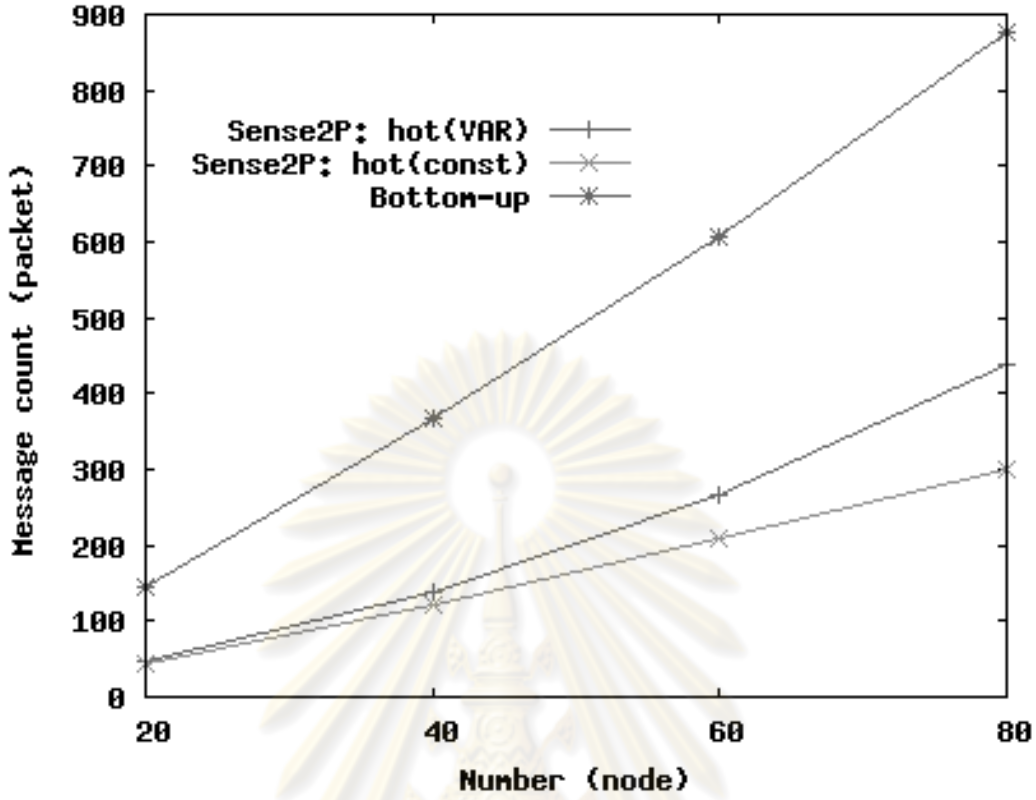
โดยการทดลองจะทำการส่งข้อคำถาม 2 ประเภท

1) ข้อคำถามประเภทแรกจะเป็นการตรวจสอบการมีอยู่ของข้อเท็จจริง คือทุกๆ อาร์กิวเมนต์ของคำถามนั้นเป็นค่าคงที่

2) ข้อคำถามเพื่อขอคำตอบทั้งหมดที่สอดคล้องกับกฎ โดยกำหนดให้ AreaID เป็นตัวแปร

ผลการทดลองได้แสดงในรูปที่ 5.3 ซึ่งเห็นได้ชัดเจนว่าระบบเซนส์ทูพีนั้นมีการใช้งานข้อความในเครือข่ายที่น้อยกว่าการประมวลผลแบบล่างขึ้นบนอย่างเห็นได้ชัด ไม่ว่าจะเป็นข้อคำถามประเภทใดก็ตาม สำหรับกรณีที่ข้อคำถามเพื่อตรวจสอบกฎนั้นเห็นได้ชัดว่าสามารถใช้ค่าคงที่ระบุมาในข้อคำถามเพื่อใช้ในการกรองได้ตั้งแต่ต้น ส่วนกรณีที่เป็นตัวแปรนั้นก็ยังสามารถช่วยกรองคำตอบที่ไม่จำเป็นได้โดยใช้ค่าคงที่ในกฎ คือ T > 50 ในการกรองคำตอบที่ไม่เกี่ยวข้อง ในขณะที่แบบล่างขึ้นบนต้องมีการส่งคำตอบทั้งหมดที่เป็นไปได้กลับขึ้นมาที่สถานีฐานเพื่อประมวลผล

แต่ถึงอย่างไรก็ตามในกรณีที่ข้อคำถามประเภทขอคำตอบทั้งหมดนั้น (ระบุอาร์กิวเมนต์เป็นตัวแปร) ถ้าคำตอบที่ได้จากเป้าหมายย่อยแรกมีมากขึ้นและมีจำนวนสถานีเชื่อมโยงมากขึ้นจะทำให้ข้อมูลที่ใช้ในเครือข่ายมากขึ้นตามไปด้วย เนื่องจากระบบเซนส์ทูพีในการทดลองนี้นั้นจะนำค่าคำตอบที่ได้จากเป้าหมายย่อยแต่ละคำตอบในการส่งไปยังเป้าหมายย่อยถัดไป ดังนั้นจะต้องมีการส่งจำนวนข้อคำถามเพิ่มเป็นจำนวนเท่ากับคำตอบจากข้อคำถามย่อยก่อนหน้า ซึ่งกรณีนี้ในอนาคตสามารถพัฒนาเพิ่มเติมให้ใช้จำนวนข้อมูลในเครือข่ายลดน้อยลงได้โดยการรวมรวมคำตอบที่ได้จากเป้าหมายย่อยก่อนหน้าทั้งหมดเป็นก้อนเดียวกันแล้วจึงนำไปผูก (Bind) กับเป้าหมายย่อยถัดไปแล้วจึงส่งลงไปถามในเครือข่ายตัวรับรู้ไร้สายในคราวเดียว



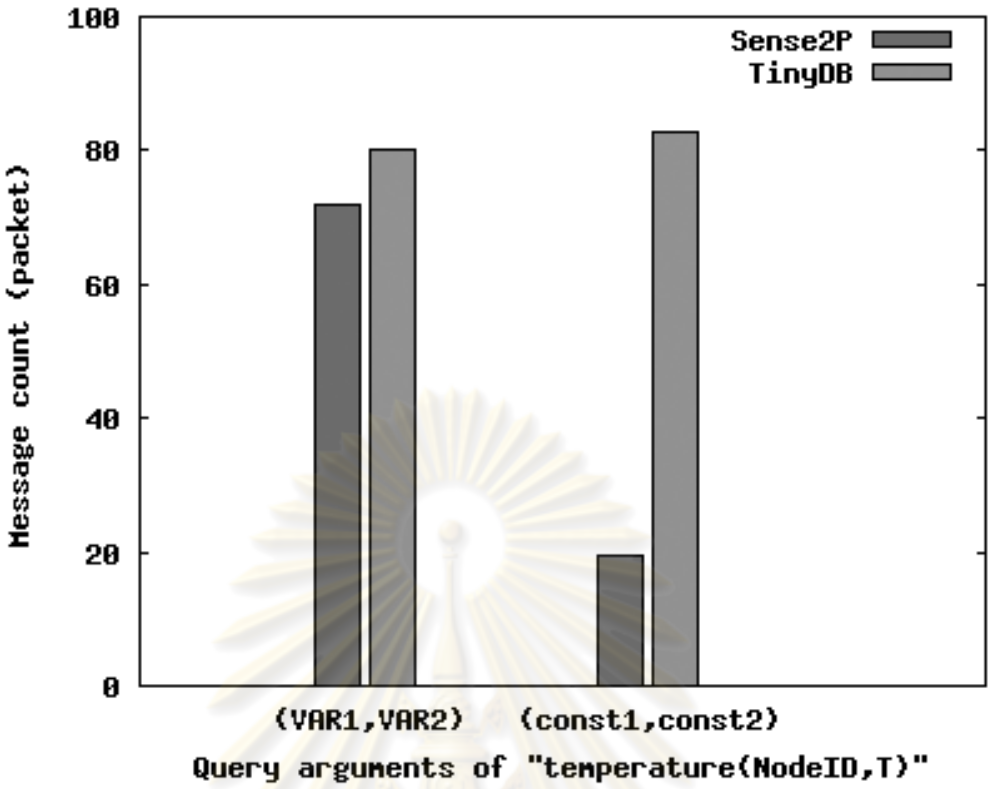
รูปที่ 5.3 กราฟเปรียบเทียบจำนวนข้อความที่ใช้โดยเฉลี่ยในกรณีที่ส่งข้อคำถามโดยอาศัยกฎ hot(AreaID) :- temperature(NodeID,T), T > 50, area(NodeID,AreaID) ในระบบเซนส์ทูพีซึ่งใช้การประมวลผลแบบบนลงล่างและระบบที่ใช้ในการประมวลผลแบบล่างขึ้นบน

5.5 ประสิทธิภาพจากการใช้กระบวนการระงับข้อคำถาม

ในการทดลองนี้จะแสดงให้เห็นผลลัพธ์ของการใช้เทคนิคการระงับข้อคำถาม โดยจะทำการเปรียบเทียบระบบเซนส์ทูพีซึ่งได้ใช้เทคนิคนี้ กับระบบไทนี่ดีบีซึ่งไม่มีการใช้การระงับข้อคำถาม โดยในการทดลองในเครือข่ายจำนวน 60 สถานีเชื่อมโยง และจะทำการส่งข้อคำถาม 2 ประเภท

1) ข้อคำถามเพื่อขอทุกคำตอบที่เป็นไปได้ในเครือข่าย คือกำหนดให้ทุกอาร์กิวเมนต์เป็นตัวแปร

2) ข้อคำถามเพื่อตรวจสอบข้อเท็จจริง คือกำหนดให้ทุกอาร์กิวเมนต์เป็นค่าคงที่



รูปที่ 5.4 กราฟเปรียบเทียบจำนวนข้อความเฉลี่ยที่ใช้ในระบบเซนส์ทูพีซึ่งใช้กระบวนการระงับข้อคำถามและระบบไทนี่ดีบี

ผลการทดลองแสดงในรูปที่ 5.4 ซึ่งเป็นไปตามที่คิดไว้นั่นคือจำนวนข้อความเฉลี่ยที่ใช้ในระบบเครือข่ายของระบบเซนส์ทูพีซึ่งนั้นจะน้อยกว่าของระบบไทนี่ดีบีโดยเฉพาะในกรณีที่ข้อคำถามเป็นประเภทเพื่อตรวจสอบข้อเท็จจริงนั้นจะน้อยกว่ามาก เนื่องจากว่าระบบเซนส์ทูพีซึ่งนั้นเมื่อได้รับข้อคำถามประเภทเพื่อตรวจสอบข้อเท็จจริง และได้ทำการพบสถานีเชื่อมโยงที่มีคำตอบที่สอดคล้องแล้วนั้น สถานีเชื่อมโยงนั้นจะหยุดการส่งต่อข้อคำถามไปยังสถานีลูกทั้งหมด ในขณะที่ระบบไทนี่ดีบีนั้นจะทำการส่งต่อข้อคำถามไปยังทุกสถานีเชื่อมโยงที่มีอยู่ภายในระบบ

แต่เป็นที่น่าสังเกตว่าระบบไทนี่ดีบีนั้นกลับใช้จำนวนข้อความในการส่งข้อคำถามประเภทตรวจสอบข้อเท็จจริง (นั่นคือการกำหนดเงื่อนไข WHERE ให้กับข้อคำถามในภาษา SQL) เป็นจำนวนมากกว่าข้อคำถามเพื่อขอทุกคำตอบที่เป็นไปได้ ซึ่งอาจเกิดจากความหละหลวมในการสร้างระบบของไทนี่ดีบีเอง

5.6 ประสิทธิภาพจากการใช้กระบวนการกรองข้อมูลล่วงหน้า

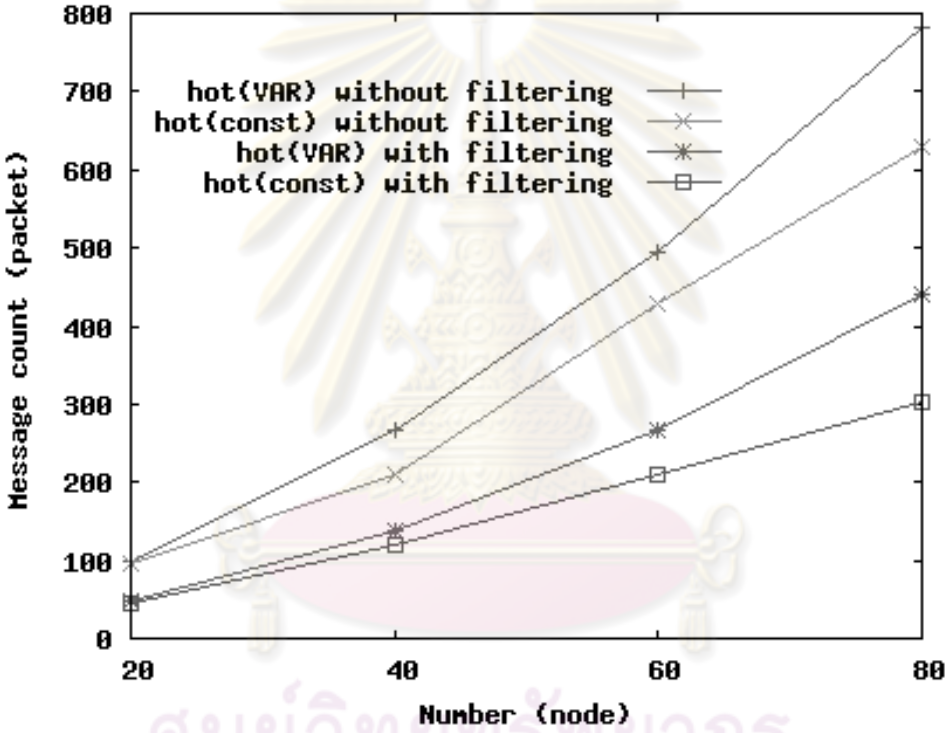
ในการทดลองนี้จะทำการวิเคราะห์ผลจากการใช้กระบวนการกรองข้อมูลล่วงหน้าโดยนำค่าที่สามารถผูกกับตัวแปรในเป้าหมายย่อยได้ส่งลงไปถามในเครือข่าย โดยทำการเปรียบเทียบผลระหว่างเซนส์ทูพีที่มีการใช้กระบวนการกรองข้อมูลนี้ กับเซนส์ทูพีที่ทำการปิด

การใช้งานกระบวนการนี้หรือไม่มีการผูกข้อมูลล่วงหน้า นั่นคือนำระบบโปรล็อกดั้งเดิมมาเชื่อมต่อกับโลจิกคิวโดยตรง และให้มีการประมวลผลจากซ้ายไปขวาตามปกติ

กฎที่ใช้ในการทดลองนี้คือกฎดังรูปที่ 5.2 โดยสำหรับการทดลองแต่ละแบบนั้นจะใช้ข้อคำถาม 2 ประเภท

1) ข้อคำถามเพื่อตรวจสอบกฎ โดยกำหนดให้ AreaID เป็นค่าคงที่

2) ข้อคำถามเพื่อขอคำตอบทั้งหมดที่สอดคล้องกับกฎ โดยกำหนดให้ AreaID เป็นตัวแปร



รูปที่ 5.5 กราฟเปรียบเทียบจำนวนข้อความเฉลี่ยในระบบเซนส์ทูพีที่มีการกรองข้อมูลล่วงหน้า และแบบไม่มีการกรองข้อมูลล่วงหน้า

ผลการทดลองแสดงในรูปที่ 5.5 ซึ่งเป็นไปตามที่คาดการณ์ไว้นั่นคือการกระบวนการกรองข้อมูลล่วงหน้าก่อนหน้าสามารถลดจำนวนข้อความที่ส่งในเครือข่ายได้มาก ถึงแม้ว่าข้อคำถามนั้นจะเป็นข้อคำถามที่ต้องการทุกคำตอบที่สอดคล้องก็ตาม (อาร์กิวเมนต์เป็นตัวแปร) เนื่องจากระบบจะนำค่า T > 50 ซึ่งเป็นเป้าหมายย่อยที่สองนั้นมาผูกค่าเค้ากับตัวแปร T ในเป้าหมายย่อยแรกก่อนที่จะส่งข้อคำถามย่อยของเป้าหมายย่อยแรกลงไปยังเครือข่ายตัวรับรู้ไร้สาย เพื่อกรองข้อมูลข้อเท็จจริง temperature ที่มีค่า T <= 50 ทิ้งทั้งหมดไม่ต้องส่งกลับ

ขึ้นมา แต่ถ้าเป็นในกรณีที่ไม่มีการใช้กระบวนการกรองข้อมูลล่วงหน้านั้น การประมวลจะเริ่มจากเป้าหมายย่อยแรกโดยไม่คำนึงถึงว่าค่าตัวแปร T นั้นมีเงื่อนไขอะไรหรือไม่ แล้วทำการส่งลงขอข้อเท็จจริง temperature ในระบบเครือข่ายตัวรับรู้ไร้สายทันที หลังจากนั้นเมื่อได้คำตอบกลับมาหมดแล้ว จึงค่อยทำการกรองข้อเท็จจริงที่ไม่เกี่ยวข้องทิ้งที่เป้าหมายย่อยที่สอง ส่งผลให้สูญเสียพลังงานในการส่งข้อมูลที่ไม่ได้อยู่ในเซตของคำตอบไปโดยไม่จำเป็น

5.7 ประสิทธิภาพจากการใช้ซูเปอร์เซตแคช

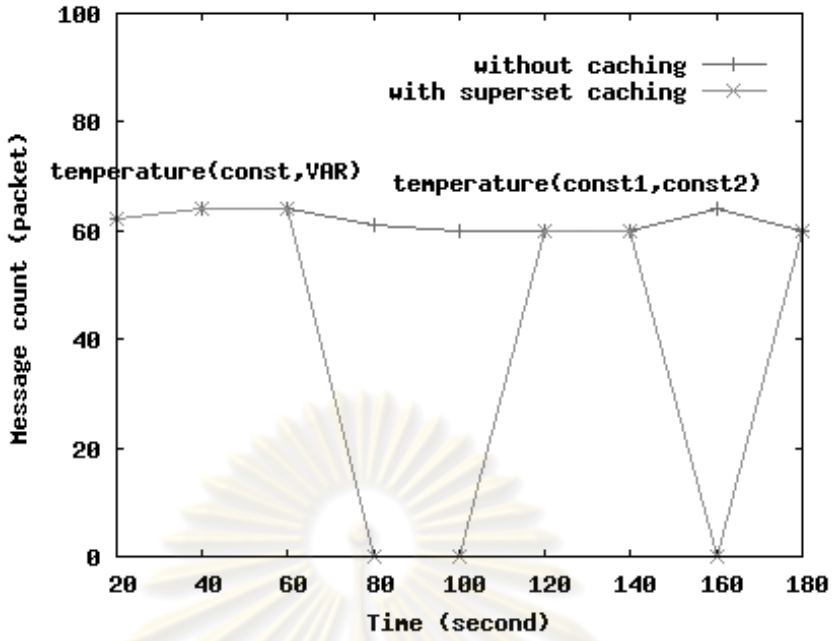
การทดลองนี้ทำเพื่อศึกษาผลลัพธ์จากการใช้เทคนิคซูเปอร์เซตแคช โดยทำการส่งชุดของข้อคำถามเกี่ยวกับเพรดิเคต temperature(NodeID,T) 2 ชุดเข้าไปยังระบบเซนส์ทูพีทั้งแบบเปิดการใช้งานซูเปอร์เซตแคช และแบบปิดซูเปอร์เซตแคช ข้อคำถามแต่ละแบบจะถูกส่งเข้าไปยังระบบทุกๆ 20 วินาที โดยมีจำนวนสถานีเชื่อมโยงในระบบทั้งหมด 60 สถานี

ในชุดการทดลองแรกนั้น 3 ข้อคำถามแรกจะส่งคำถามที่กำหนดให้อาร์กิวเมนต์ของเพรดิเคตตัวแรกเป็นค่าคงที่ และตัวที่สองเป็นตัวแปร เช่น temperature(13,T) เป็นต้น โดยค่าคงที่กำหนดโดยสุ่มแตกต่างกันไปสำหรับแต่ละข้อคำถาม ส่วนข้อคำถามที่เหลือนั้นจะส่งในรูปของการตรวจสอบข้อเท็จจริงที่มีอยู่ในระบบ โดยกำหนดทุกอาร์กิวเมนต์เป็นค่าคงที่

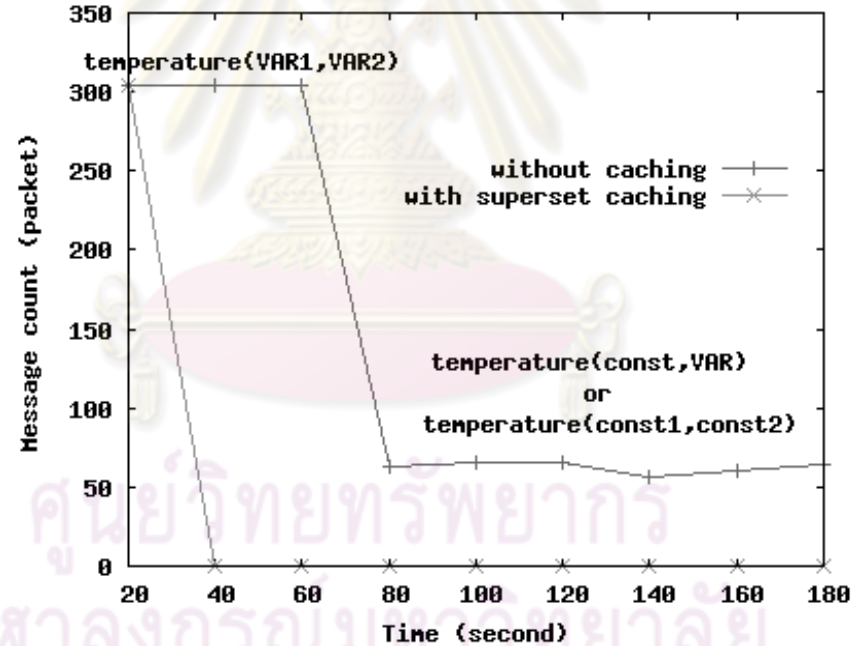
ผลการทดลองของชุดการทดลองแรกนั้นแสดงไว้ในรูปที่ 5.6 ก) ซึ่งจะเห็นว่าระบบเซนส์ทูพีแบบที่ใช้ซูเปอร์เซตแคชนั้นจะไม่มีการส่งข้อคำถามลงไปยังเครือข่าย (จำนวนข้อมูลในเครือข่ายเป็นศูนย์) สำหรับบางข้อคำถามหลังจากได้ส่งข้อคำถาม 3 ข้อคำถามแรกไปแล้ว ซึ่งกรณีนี้เกิดจากการที่การส่งข้อคำถามเพื่อตรวจสอบข้อเท็จจริงนั้นมีค่าคงที่ที่เป็นอาร์กิวเมนต์ตัวแรกตรงกับอาร์กิวเมนต์ตัวแรกในข้อคำถามข้อใดข้อหนึ่งในสามข้อแรกที่ได้ส่งไป ซึ่งข้อคำถามสามข้อแรกจะเป็นซูเปอร์เซตของข้อคำถามนั้น ส่งผลให้ระบบสามารถนำเอาคำตอบที่เคยได้รับแล้วกลับมาใช้ใหม่ได้ ในขณะที่ระบบเซนส์ทูพีแบบที่ปิดการใช้งานซูเปอร์เซตแคชนั้นจะทำการส่งข้อคำถามลงไปยังเครือข่ายตัวรับรู้ไร้สายทุกครั้งแม้ว่าจะเคยมีคำตอบเก็บไว้แล้วก็ตาม

ผลการทดลองชุดที่สองนั้น 3 ข้อคำถามแรกจะเป็นข้อคำถามที่ขอทุกคำตอบที่มีอยู่ในระบบ (ทุกอาร์กิวเมนต์เป็นตัวแปร) หลังจากนั้นจะทำการส่งข้อคำถามที่มีอาร์กิวเมนต์เป็นค่าคงที่อย่างน้อยหนึ่งตัวลงไปยังเครือข่าย

ผลการทดลองชุดที่สองได้แสดงในรูปที่ 5.6 ข) ซึ่งจะเห็นว่าระบบเซนส์ทูพีแบบใช้ซูเปอร์เซตแคชนั้นจะมีการส่งข้อมูลลงไปถามยังเครือข่ายตัวรับรู้ไร้สายเพียงครั้งแรกครั้งเดียว เพราะเซตคำตอบจากข้อคำถามครั้งแรกนั้นครอบคลุมทุกคำตอบที่เป็นไปได้แล้ว ดังนั้นแม้ว่าข้อคำถามที่ตามมาจะเป็นข้อคำถามแบบใด เซตของคำตอบก็จะเป็นเซตย่อยของข้อคำถามแรกเสมอ ส่งผลให้ไม่จำเป็นต้องส่งข้อคำถามเข้าไปยังเครือข่ายตัวรับรู้ไร้สายอีกครั้ง ในขณะที่ระบบ



ก)



ข)

รูปที่ 5.6 กราฟเปรียบเทียบจำนวนข้อความเฉลี่ยในระบบเซนส์ทูพีที่มีการใช้ซูเปอร์เซตแคช และแบบไม่มีการใช้แคช ก) ทดสอบโดยให้ซูเปอร์เซตมีขนาดเล็ก ข) ทดสอบโดยให้ซูเปอร์เซตครอบคลุมทุกคำตอบ

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

เซนส์ทูพีแบบไม่ใช้ซูเปอร์เซตแคชนั้นจะต้องส่งข้อคำถามเข้าไปยังเครือข่ายตัวรับรู้ไร้สายทุกครั้งแม้ว่าจะมีคำตอบที่ครอบคลุมอยู่แล้วก็ตาม

จากการทดลองจะเห็นได้ว่าการใช้ซูเปอร์เซตแคชนั้นจะมีประโยชน์มากขึ้นถ้าซูเปอร์เซตของคำตอบนั้นมีขนาดใหญ่ แต่ถึงอย่างไรก็ตามซูเปอร์เซตแคชก็ไม่ได้มีประโยชน์ในทุกกรณี เช่นในกรณีที่ข้อมูลในสภาพแวดล้อมนั้นมีการเปลี่ยนแปลงบ่อย ส่งผลให้คำตอบที่มีการแคชเก็บเอาไว้นั้นเก่าจนไม่สามารถใช้งานได้ ซึ่งปัญหานี้ก็เป็นปัญหาที่พบในการใช้แคชทั่วไปซึ่งควรต้องมีการกำหนดเวลาหมดอายุของคำตอบตามแต่ลักษณะของการใช้งานของโปรแกรมประยุกต์ว่าสามารถยอมรับคำตอบเก่าได้มากน้อยเพียงใด แต่ต้องการล้างแคชหรือมีการลบข้อมูลในแคชบางส่วนเมื่อใด ซึ่งแน่นอนว่าเวลาที่ใช้ในการล้างข้อมูลนั้นแปรผกผันกับความใหม่ของข้อมูล แต่แปรผันตรงกับการประหยัดพลังงานที่ใช้ในการส่งข้อมูลในระบบ

5.8 ความสมบูรณ์ของคำตอบและความเกี่ยวข้องของคำตอบ

ความสมบูรณ์ของคำตอบและความเกี่ยวข้องของคำตอบนั้นวัดจากการทดลองทั้งหมดที่ได้ทดลองมา โดยวัดจากอัตราส่วนระหว่างคำตอบที่ระบบได้รับเทียบกับคำตอบทั้งหมดที่มีอยู่จริงในเครือข่าย และอัตราส่วนของคำตอบที่เกี่ยวข้องกับข้อคำถามกับจำนวนคำตอบทั้งหมดที่ได้รับตามลำดับ

ด้วยการทดลองบนสิ่งแวดล้อมที่อยู่บนสมมติฐานว่าใช้การติดต่อสื่อสารแบบเชื่อถือได้ (Reliable Communication) ซึ่งไม่มีการเกิดแพ็กเก็จข้อมูลสูญหายบนระบบ ส่งผลให้การทดลองทั้งหมดนั้นมีความสมบูรณ์ของคำตอบ 100% และความเกี่ยวข้องกันของคำตอบ 100% เช่นกัน ซึ่งชี้ให้เห็นว่าถึงแม้จะมีการกรองข้อมูลหรือการยับยั้งข้อคำถามเพื่อประหยัดพลังงานในการติดต่อสื่อสาร แต่อัลกอริทึมการทำงานยังสามารถทำงานได้ถูกต้องเช่นเดิม

5.9 เปรียบเทียบคุณสมบัติของระบบกับระบบฐานข้อมูลอื่น

นอกเหนือไปจากการเปรียบเทียบด้านประสิทธิภาพแล้วอีกส่วนหนึ่งก็คือการเปรียบเทียบคุณสมบัติกับระบบที่มีอยู่ในปัจจุบัน โดยระบบที่นำมาเปรียบเทียบนั้นคือไทนี่ดีบีซึ่งเป็นระบบฐานข้อมูลสำหรับเครือข่ายตัวรับรู้แบบไร้สายโดยเฉพาะ สามารถสรุปความสามารถได้ดังตารางที่ 5.1

ระบบเซนส์ทูพีนั้นมีข้อดีกว่าระบบไทนี่ดีบีที่มีอยู่ปัจจุบันคือสามารถรองรับข้อคำถามแบบตรรกะ (Logic-Based Query) ซึ่งทำให้ระบบรองรับการเขียนโปรแกรมแบบตรรกะได้ ในขณะที่ไทนี่ดีบีนั้นไม่รองรับข้อคำถามลักษณะนี้ และจะคืนคำตอบทั้งหมดที่ต้องการและต้องเขียนโปรแกรมในการประมวลผลคำตอบที่ได้จากไทนี่ดีบีอีกต่อหนึ่ง นอกจากนั้นแล้วระบบเซนส์ทูพียังสามารถรองรับการเขียนโปรแกรมแบบครอบคลุม (Global Query Program) และ

ตารางที่ 5.1 เปรียบเทียบคุณสมบัติระหว่างระบบเซนส์ทูพีและไทนี่ดีบี

	Short-Live Query	Long-Live Query	Logic-Based Query	Event Triggering	Query Program	Column Alias	Constraint
Sense2P	ได้	ไม่ได้	ได้	ไม่ได้	ครอบคลุม	ได้	ครอบคลุม
TinyDB	ได้	ได้	ไม่ได้	ได้	เฉพาะที่	ไม่ได้	เฉพาะที่

การกำหนดเงื่อนไขแบบครอบคลุมได้ (Global Constraint) โดยการทำงานไม่ขึ้นกับสถานีเชื่อมโยงใดสถานีหนึ่ง และสามารถสร้างสมนาม (Alias) ได้โดยการสร้างเป็นกฎที่ประกอบด้วยเพรดิเคตตัวเดียว และด้วยความสามารถเหล่านี้เซนส์ทูพีนอกจากจะใช้สำหรับระบบฐานข้อมูลนิรนัยแล้วเซนส์ทูพียังสามารถใช้ทดแทนฐานข้อมูลเชิงสัมพันธ์ได้ด้วย ในขณะที่ไทนี่ดีบีจะใช้เป็นฐานข้อมูลเชิงสัมพันธ์ได้แต่ใช้เป็นฐานข้อมูลนิรนัยไม่ได้

แต่ถึงอย่างไรก็ตามยังมีบางคุณสมบัติซึ่งเซนส์ทูพีนั้นยังไม่ได้ทำการสร้างขึ้นในงานวิจัยนี้เช่นการทำงานสำหรับข้อคำถามที่ต้องการคำตอบในระยะยาว (Long-Live Query) นั่นคือเมื่อส่งข้อคำถามไปครั้งหนึ่งแล้ว และต้องการให้มีการส่งคำตอบจากข้อคำถามเดิมกลับมาเป็นระยะๆเช่นในไทนี่ดีบี แต่ในเซนส์ทูพีนั้นผู้ใช้จะต้องการส่งข้อมูลใหม่เองทุกครั้งเมื่อต้องการข้อมูล และอีกส่วนหนึ่งคือการส่งคำตอบเมื่อเกิดเหตุการณ์ขึ้น (Event Triggering) คือสามารถกำหนดได้ว่าจะทำการส่งคำตอบเมื่อมีเหตุการณ์หรือค่าที่ต้องการเกิดขึ้นโดยอัตโนมัติ

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 6

สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

งานวิจัยนี้ได้นำเสนอวิธีการประมวลผลข้อคำถามเรียกซ้ำแบบครอบคลุมสำหรับเครือข่ายตัวรับรู้ไร้สาย โดยได้ออกแบบแบบจำลองของภาษาเพื่อใช้สำหรับการสร้างโปรแกรมและข้อคำถาม และสถาปัตยกรรมระบบที่เหมาะสมกับลักษณะของเครือข่ายตัวรับรู้แบบไร้สาย

ในงานวิจัยนี้ได้ทำการกล่าวแย้งงานวิจัยก่อน ๆโดยแสดงให้เห็นว่าการประมวลข้อคำถามแบบเรียกซ้ำนั้น การใช้กลยุทธ์การประเมินผลแบบบนลงล่างจะเหมาะสมและประหยัดพลังงานมากกว่าการประเมินผลแบบล่างขึ้นบนสำหรับเครือข่ายตัวรับรู้ไร้สายอันมีพลังงานและทรัพยากรอยู่อย่างจำกัด นอกจากนั้นยังได้นำเสนอกระบวนการต่าง ๆที่ช่วยให้ระบบทำงานโดยใช้พลังงานอย่างประหยัด ได้แก่ กระบวนการระงับข้อคำถาม กระบวนการกรองข้อมูลล่วงหน้า และการใช้ซูเปอร์เซตแคชช่วยในการจำข้อมูลที่เคยได้สอบถามไปแล้ว

งานวิจัยนี้ยังได้สร้างระบบย่อยโลจิกคิว (LogicQ) ซึ่งเป็นระบบย่อยที่ทำงานบนฝั่งเครือข่ายตัวรับรู้แบบไร้สายเพื่อใช้ในการประมวลผลข้อคำถามตรรกะที่ระบบโปรแกรมเชิงตรรกะจะส่งมาสอบถาม โดยทำให้ผู้สร้างระบบโปรแกรมเชิงตรรกะมีอิสระในการออกแบบส่วนประมวลผลขณะทำงานโดยไม่ต้องสนใจการทำงานในระดับล่าง นอกจากนั้นได้มีการสร้างระบบเซนส์ทูพี (Sense2P) ซึ่งเป็นระบบโปรแกรมมหภาพเชิงตรรกะ (Logic Macroprogramming) ที่เรียกใช้บริการของระบบย่อยโลจิกคิว ซึ่งสร้างขึ้นเพื่อเป็นเครื่องพิสูจน์ทฤษฎีที่ได้ออกแบบไว้และสามารถนำไปใช้เป็นระบบฐานข้อมูลนิรนัย (Deductive Database) ได้

ผลการทดลองได้แสดงเปรียบเทียบให้เห็นว่ากลยุทธ์การประเมินผลแบบบนลงล่างนั้นประหยัดพลังงานมากกว่าแบบล่างขึ้นบนเป็นอย่างมาก นอกจากนั้นยังแสดงให้เห็นถึงผลจากใช้เทคนิคของกระบวนการต่าง ๆที่ช่วยให้ระบบประหยัดพลังงานขึ้นเป็นอย่างมาก ในขณะที่ยังสามารถรักษาความสมบูรณ์ของคำตอบและความเกี่ยวข้องของคำตอบได้ครบถ้วนในสภาพแวดล้อมที่ทำการทดลอง

6.2 ข้อจำกัด

ถึงแม้งานวิจัยนี้จะแสดงให้เห็นประสิทธิภาพของระบบเซนส์ทูพีที่ทำการออกแบบและสร้างขึ้นมาสำหรับเครือข่ายตัวรับรู้แบบไร้สายโดยเฉพาะและมีประสิทธิภาพในการทำงานที่ดีกว่าระบบฐานข้อมูลเชิงสัมพันธ์ที่ใช้กันอยู่ในปัจจุบันสำหรับเครือข่ายตัวรับรู้แบบไร้สาย แต่ระบบนั้นก็ยังมีข้อจำกัดอยู่หลายประการด้วยกัน

ระบบเซนส์ทูพีนั้นยังไม่ได้ทำการสร้างเพื่อรองรับทำงานของข้อคำถามที่ต้องการคำตอบในระยะยาว นั่นคือเมื่อส่งข้อคำถามไปครั้งหนึ่งแล้ว และต้องการให้มีการส่งคำตอบจากข้อคำถามเดิมกลับมาเป็นระยะ ๆ ซึ่งผู้ใช้จะต้องการส่งข้อมูลใหม่เองทุกครั้งเมื่อต้องการข้อมูล และอีกส่วนหนึ่งคือการส่งคำตอบเมื่อเกิดเหตุการณ์ขึ้น คือสามารถกำหนดได้ว่าจะทำการส่งคำตอบเมื่อมีเหตุการณ์หรือค่าที่ต้องการเกิดขึ้นโดยอัตโนมัติ ซึ่งคุณสมบัติทั้งสองนับได้ว่าเป็นคุณสมบัติที่มีการใช้ค่อนข้างบ่อยในระบบเครือข่ายตัวรับรู้แบบไร้สาย

การระงับข้อคำถามนั้นจะช่วยประหยัดพลังงานได้ไม่มากนักในกรณีที่เครือข่ายมีสถานีเชื่อมโยงเป็นจำนวนมาก และสถานีเชื่อมโยงที่มีคำตอบนั้นอยู่ใกล้กับบริเวณใบของต้นไม้จัดการเส้นทาง นั่นคือจะสามารถระงับการส่งข้อมูลได้เพียงไม่กี่สถานีเชื่อมโยงที่อยู่ติดกับบริเวณใบของต้นไม้จัดการเส้นทางเท่านั้น ส่วนการกรองข้อมูลล่วงหน้านั้นจะไม่เกิดประโยชน์อันใดถ้าในกฎนั้นไม่การกำหนดเงื่อนไขค่าคงที่ให้กับตัวแปรในกฎ และการใช้ซูเปอร์เซตนั้นจะเป็นข้อแลกเปลี่ยนระหว่างความใหม่ของข้อมูลกับพลังงานที่ใช้ในระบบ โดยต้องมีกำหนดช่วงเวลาการล้างแคชที่เหมาะสมสำหรับแต่ละโปรแกรมประยุกต์ที่จะนำไปใช้

6.3 ข้อเสนอแนะ

ระบบยังสามารถที่ประหยัดพลังงานได้มากขึ้นกว่าโดยทำการปรับกระบวนการทำงานให้เหมาะสมเพิ่มเติม ได้แก่ การรวบรวมเซตของคำตอบจากเป้าหมายย่อยอันก่อนหน้ารวมเป็นข้อมูลชุดเดียว แล้วส่งลงไปยังเครือข่ายตัวรับรู้ไร้สายในคราวเดียว และในการรับข้อมูลคำตอบในเครือข่ายนั้นถ้าสถานีเชื่อมโยงแต่ละตัวทำการรอรับคำตอบจากสถานีลูกของตัวเองก่อนแล้วนำคำตอบของสถานีลูกรวมกับคำตอบของตนเองแล้วค่อยส่งขึ้นมาไปพร้อมกัน นอกจากนั้นในกรณีที่เป็นเครือข่ายแบบหลากหลาย เช่น แต่ละสถานีเชื่อมโยงนั้นมีความสามารถในการตรวจจับแตกต่างกันนั้น ถ้ามีการส่งความสามารถในการตรวจจับได้ของสถานีตนเองไปบอกกับสถานีผู้ปกครอง สถานีผู้ปกครองจะสามารถใช้ข้อมูลนี้มาช่วยในการระงับการส่งข้อคำถามได้ดียิ่งขึ้น ซึ่งเทคนิคเหล่านี้ล้วนจะช่วยทำให้ระบบประหยัดพลังงานได้มากยิ่งขึ้นแต่ยังไม่ได้ทำการสร้างขึ้นในการวิจัยชิ้นนี้

นอกจากนั้นระบบในงานวิจัยนี้สนับสนุนข้อคำถามแบบเฉพาะกิจ (Ad-hoc query) เพื่อดูสถานะปัจจุบันของเครือข่ายและสิ่งแวดล้อมที่ทำการตรวจจับในขณะนั้น แต่ยังไม่สามารถรองรับข้อคำถามสำหรับการเฝ้าระวังหรือตรวจจับสภาพแวดล้อมแบบกระแส (Stream) ซึ่งต้องมีการปรับปรุงเพิ่มเติมเกี่ยวกับการระบุเวลาเพื่อให้มีการส่งข้อมูลกลับมาเป็นรอบ ๆโดยส่งข้อคำถามลงไปเพียงครั้งเดียว

การป้องกันการเกิดการวนซ้ำไม่รู้จบนั้นอยู่นอกเหนือขอบเขตของงานวิจัยชิ้นนี้ แต่ได้มีงานวิจัยอื่นที่ได้คิดค้นวิธีการป้องกันขึ้น เช่น การใช้วิธีการเต่ากับกระต่าย (Tortoise-and-hare)[25] การใช้กองซ้อน (Stack)[26] เป็นต้น

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

รายการอ้างอิง

[1] Weiser, M. (1999). The Computer for The 21st Century. ACM SIGMOBILE Mobile Computing and Communication Review 3(3): 3-11.

[2] Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R. and Anderson, J. (2002). Wireless Sensor Networks for Habitat Monitoring. Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, pp. 88-97. Atlanta, Georgia, USA.

[3] Gui, C., and Mohapatra, P. (2004). Power Conservation and Quality of Surveillance in Target Tracking Sensor Networks. Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, pp. 129-143. Philadelphia, PA, USA.

[4] Xu, N., Rangwala, S., Chintalapudi, K. K., Ganesan, D., Broad, A., Govindan, R. and Estrin, D. (2004). A Wireless Sensor Network for Structural Monitoring. Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, pp. 13-24. Baltimore, MD, USA.

[5] He, T., Krishnamurthy, S., Stankovic, J. A., Abdelzaher, T., Luo, L., Stoleru, R., Yan, T., Gu, L., Hui, J., and Krogh, B. (2004). Energy-Efficient Surveillance System Using Wireless Sensor Networks. Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services, pp. 270-283. Newyork, NY, USA.

[6] Bonnet, P., Gehrke, J., and Seshadri, P. (2000). Querying the Physical World. IEEE Personal Communications 7(5): 10-15.

[7] Bonnet, P., Gehrke, J. and Seshadri, P. (2001). Towards Sensor Database Systems. Proceedings of the 2nd International Conference on Mobile Data Management, pp. 3-14. London, UK.

[8] Govindan, R., Hellerstein, J. M., Hong, W., Madden, S., Franklin, M., and Shenker, S. (2002). The Sensor Network as a Database. USC Computer Science Department Technical Report, University of Southern California.

[9] Yao, Y., and Gehrke, J. (2002). The Cougar Approach to In-Network Query Processing in Sensor Networks. SIGMOD Rec. 31: 9-18.

[10] Yao, Y., and Gehrke, J. (2003). Query Processing for Sensor Networks. Proceedings of the 1st Biennial Conference on Innovative Data Systems Research, pp. 46-55. Los Alamitos, CA, USA.

[11] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W. (2002). TAG: a Tiny Aggregation Service for Ad-hoc Sensor Networks. SIGOPS Oper. Syst. Rev. 36: 131-146.

[12] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W. (2005). TinyDB: an Acquisitional Query Processing System for Sensor Networks. ACM Trans. Database Syst. 30: 122-173.

[13] Chu, D., Popa, L., Tavakoli, A., Hellerstein, J. M., Levis, P., Shenker, S., and Stoica, I. (2007). The Design and Implementation of a Declarative Sensor Network System. Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, pp. 175-188. Sydney, Australia.

[14] Tavakoli, A., Chu, D., Hellerstein, J. M., Levis, P., and Shenker, S. (2007). A Declarative Sensornet Architecture. SIGBED Rev. 4: 55-60.

[15] Hinz, Y. K. (2002). Datalog Bottom-up is the Trend in the Deductive Database Evaluation Strategy. Technical Report INSS 690, University of Maryland.

[16] Levis, P., Lee, N., Welsh, M., and Culler, D. (2003). TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, pp. 126-137. Los Angeles, California, USA.

[17] Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., and Culler, D. (2004). TinyOS: An Operating System for Wireless Sensor Networks. Ambient Intelligence: Springer-Verlag.

[18] Ramamohanarao, K., and Harland, J. (1994). An Introduction to Deductive Database Languages and Systems. VLDB Journal 3: 107-122.

[19] Ceri, S., Gottlob, G., and Tanca, L. (1989). What You Always Wanted to Know About Datalg (And Never Dared to Ask). IEEE Transactions on Knowledge and Data Engineering 1: 146-166.

[20] Terfloth, K., Wittenburg, G., and Schiller, J. (2006). Rule-Oriented Programming for Wireless Sensor Networks. Proceedings of the International Conference on Distributed Computing in Sensor Networks (DCOSS) / EAWMS Workshop, San Francisco, USA.

[21] Terfloth, K., Wittenburg, G., and Schiller, J. (2006). FACTS - A Rule-Based Middleware Architecture for Wireless Sensor Networks. Proceedings of the 1st IEEE International Conference on Communication System Software and Middleware, pp. 1-8. New Delhi, India.

[22] Chu, D., Tavakoli, A., Popa, L., and Hellerstein, J. (2006). Entirely Declarative Sensor Network Systems. Proceedings of the 32nd International Conference on Very Large Databases, pp. 1203-1206. Seoul, Korea.

[23] Denti, E., Omicini, A., and Ricci, A. (2001). tuProlog: A Light-Weight Prolog for Internet Applications and Infrastructures. Proceedings of the 3rd International Symposium on Practical Aspects of Declarative Languages, pp. 184-198. London, UK.

[24] Gay, D., Levis, P., Behren, R.V., Welsh, M., Brewer, E., and Culler, D. (2003). The nesC Language: A Holistic Approach to Networked Embedded Systems. Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation, pp. 1-11. San Diego, California, USA.

[25] Van Gelder, A. (1987). Efficient Loop Detection in Prolog Using the Tortois-and-Hare Technique. Journal of Logic Programming 4 (1): 23-31.

[26] Nivasch, G. (2004). Cycle Detection Using a Stack. Information Processing Letters 90: 135-140.

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ก

การใช้งานระบบเซนส์ทูพี

การใช้งานระบบเซนส์ทูพีนั้นสามารถทำได้ 2 รูปแบบ คือ ใช้งานบนระบบจำลองทอสซิม และใช้งานระบบกับอุปกรณ์ตัวรับรู้ไร้สายจริง

ขั้นตอนการใช้งานระบบมีดังต่อไปนี้

1. เริ่มการทำงานของระบบย่อยโลจิกคิวบนสถานีเชื่อมโยง

1.1 สำหรับระบบจำลองทอสซิม

1) บน shell ของ Linux หรือบน cygwin ให้ไปที่ไดเรกทอรี่ /opt/tinyos-1.x/apps/RecursiveQueryProcessing แล้วพิมพ์คำสั่ง

make pc

เพื่อเป็นการคอมไพล์โปรแกรมเพื่อใช้กับระบบจำลองทอสซิม

2) พิมพ์คำสั่ง

export DBG=usr1

เพื่อเป็นการสั่งเปิดข้อความดีบักให้เห็นว่าระบบมีการทำงานจริง

2) พิมพ์คำสั่ง

./build/pc/main.exe –rf=*topology_file_name*.nss *num_node*

เพื่อเป็นการเริ่มทำงานระบบย่อยโลจิกคิวที่ทำงานบนแต่ละสถานีเชื่อมโยง

โดย topology_file_name คือ ชื่อไฟล์ที่กำหนดทอพอลอยีของเครือข่าย

num_node คือ จำนวนสถานีเชื่อมโยงทั้งหมดในระบบ

1.2 สำหรับอุปกรณ์ตัวรับรู้ไร้สายเทลอสบี

1) บน shell ของ Linux หรือบน cygwin ให้ไปที่ไดเรกทอรี่ /opt/tinyos-1.x/apps/RecursiveQueryProcessing แล้วพิมพ์คำสั่ง

make telosb

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

เพื่อทำการคอมไพล์โปรแกรมให้ใช้กับอุปกรณ์รับรู้ไร้สายเทลอสบี

2) พิมพ์คำสั่ง

make telosb install.*n*

เพื่อเป็นติดตั้งโปรแกรมโลจิกคิวเข้าสู่อุปกรณ์ตัวรับรู้ไร้สายเทลอสบี

โดย n คือหมายเลข (ID) ของสถานีเชื่อมโยงที่ต้องการ ซึ่งหมายเลข 0 นั้นจะทำหน้าที่เป็นสถานีเชื่อมโยงเกตเวย์เสมอ

3) ทำซ้ำกับทุกๆอุปกรณ์ที่ต้องการ

4) ให้สถานีเชื่อมโยงหมายเลข 0 เชื่อมต่อกับพอร์ตยูเอสบี (USB) ของเครื่องสถานีฐาน

2. เปิดการทำงานส่วนการเชื่อมต่อพอร์ตอนุกรมกับทีซีพีโพรโทคอล

2.1 สำหรับระบบจำลองทอสซิม

1) เปิดเทอร์มินัลใหม่บน Linux หรือเปิดหน้าต่าง cygwin เพิ่มขึ้นอีกหนึ่ง

2) พิมพ์คำสั่ง

java net.tinyos.sf.SerialForwarder –comm tossim-serial

เพื่อเปิดโปรแกรมที่ทำหน้าที่เชื่อมต่อไปยังพอร์ตอนุกรมของระบบจำลองทอสซิมแล้วทำการแปลงให้ผู้เชื่อมต่อภายนอกต่อเข้ามาโดยทีซีพีโพรโทคอลได้

2.2 สำหรับอุปกรณ์ตัวรับรู้ไร้สายเทลอสบี

1) เปิดเทอร์มินัลใหม่บน Linux หรือเปิดหน้าต่าง cygwin เพิ่มขึ้นอีกหนึ่ง

2) พิมพ์คำสั่ง

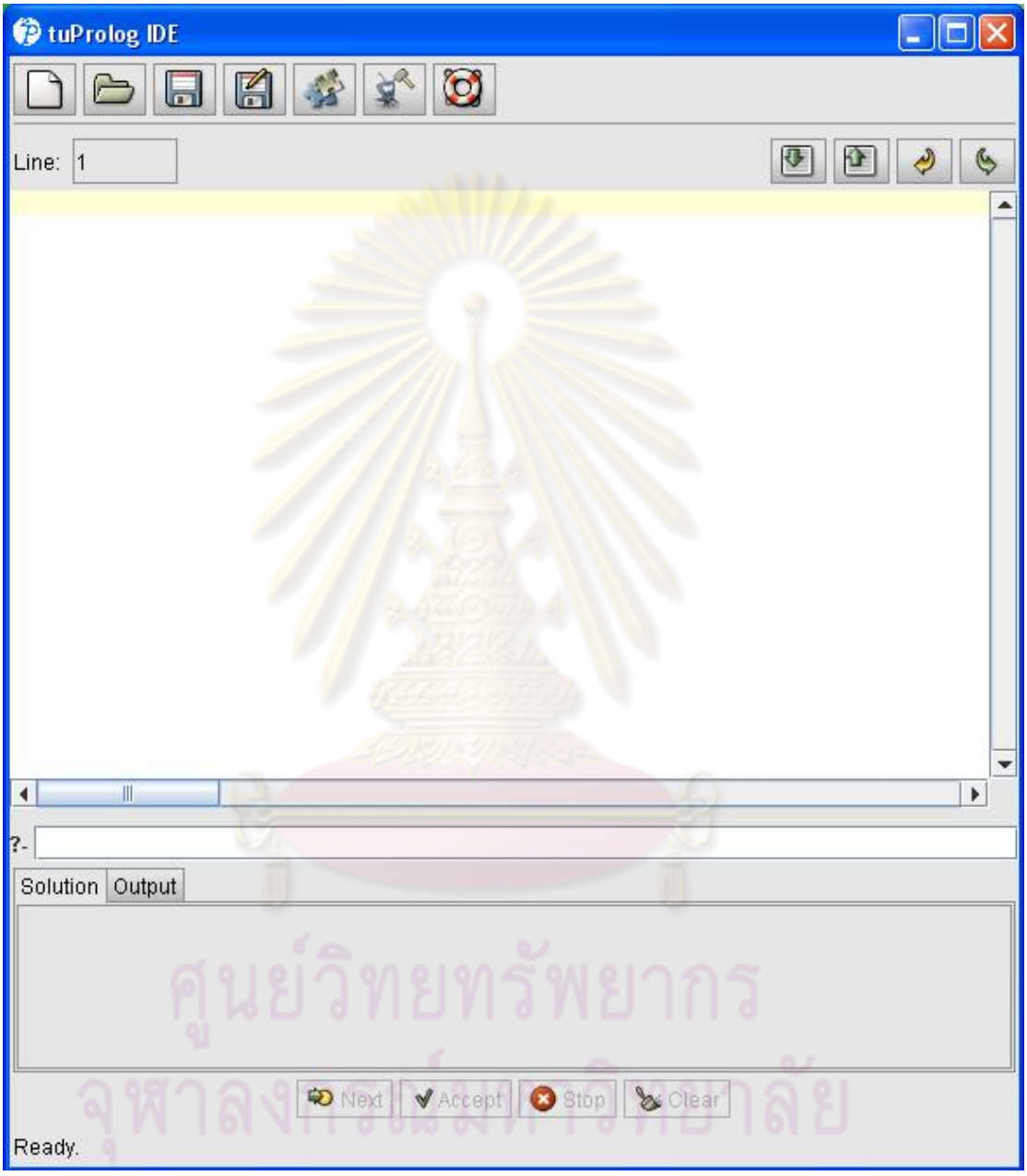
java net.tinyos.sf.SerialForwarder –comm serial@COM*X*:telosb

เพื่อเปิดโปรแกรมที่ทำหน้าที่เชื่อมต่อไปยังพอร์ตอนุกรมของอุปกรณ์ตัวรับรู้ไร้สายเทลอสบีแล้วทำการแปลงให้ผู้เชื่อมต่อภายนอกต่อเข้ามาโดยทีซีพีโพรโทคอลได้

โดย X นั้นเป็นหมายเลข COM port ที่อุปกรณ์ตัวรับรู้ไร้สายเทลอสบีได้เชื่อมต่ออยู่ (ขึ้นกับเครื่อง)

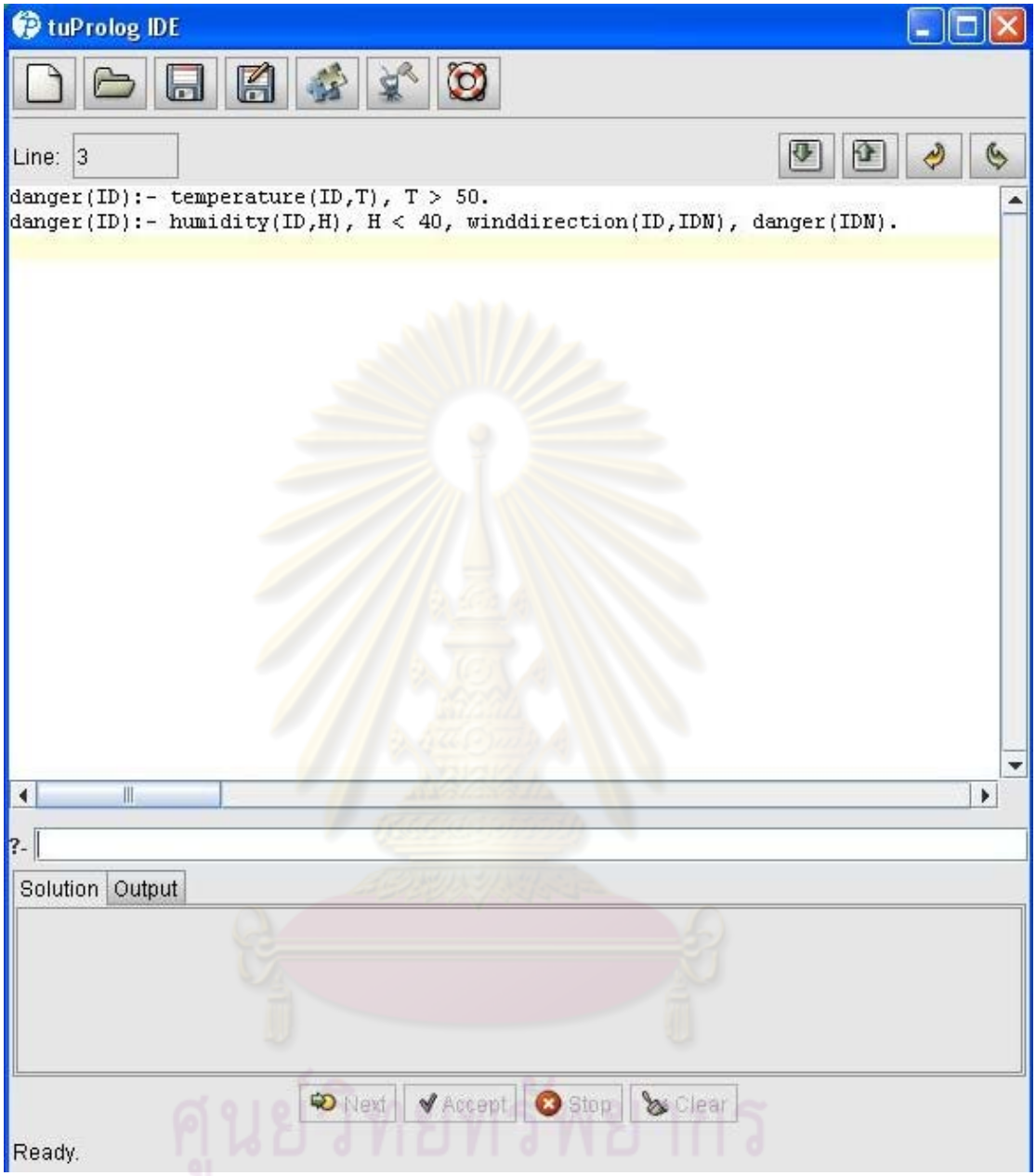
ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

3. เปิดโปรแกรมเซนส์ทูพีเมื่อระบบเริ่มทำงานจะปรากฏหน้าต่างสำหรับการใช้งานขึ้นมา ดังรูปที่ ก.1



รูปที่ ก.1 เริ่มต้นการใช้งานระบบเซนส์ทูพี

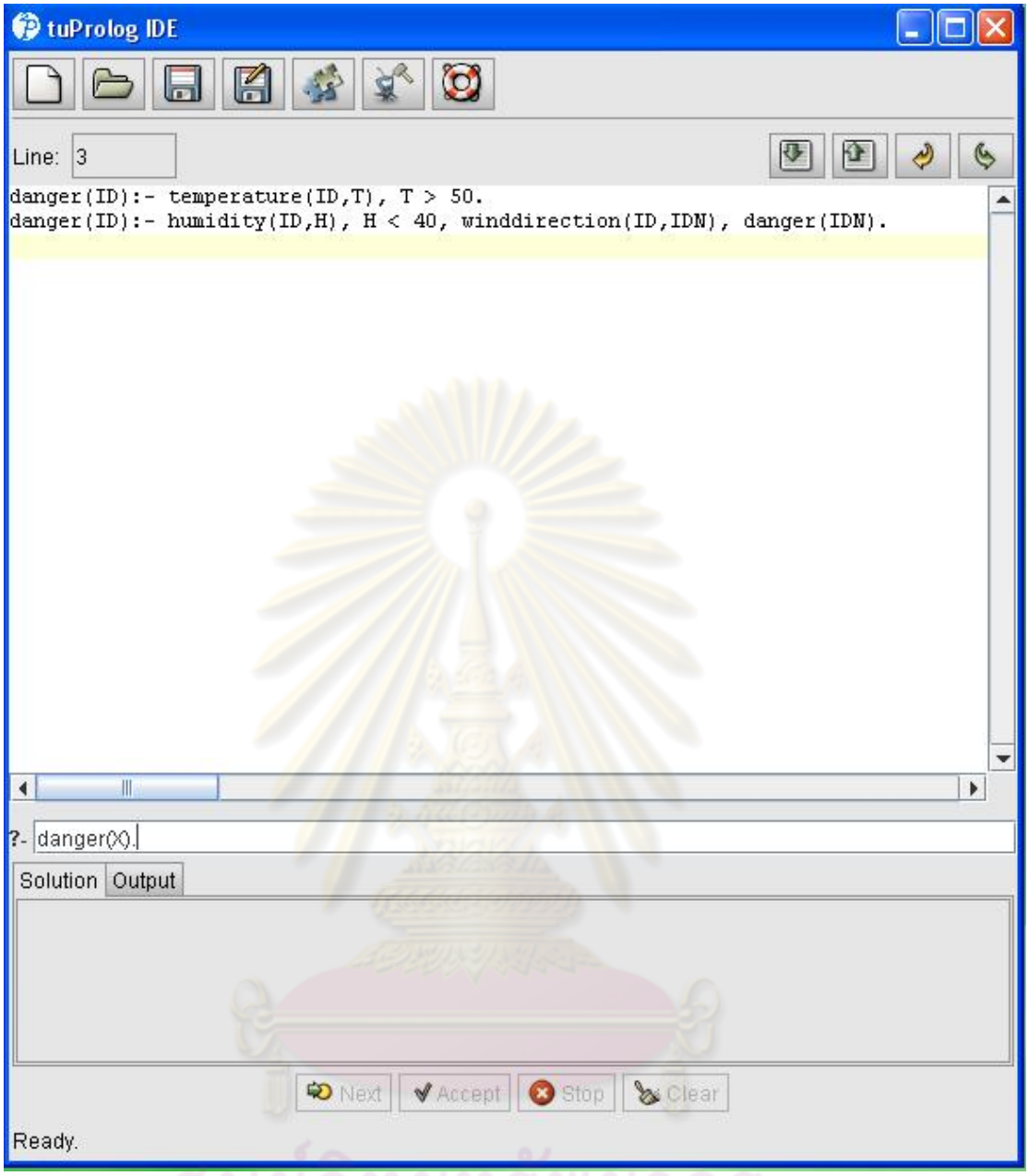
หลังจากนั้นผู้ใช้สามารถกำหนดกฎหรือข้อเท็จจริงลงไปยังระบบได้โดยเพิ่มเข้าไปในพื้นที่บริเวณกรอบสีขาว ดังรูปที่ ก.2



รูปที่ ก.2 การเพิ่มกฎและข้อเท็จจริงลงในระบบเซนส์ทูพี

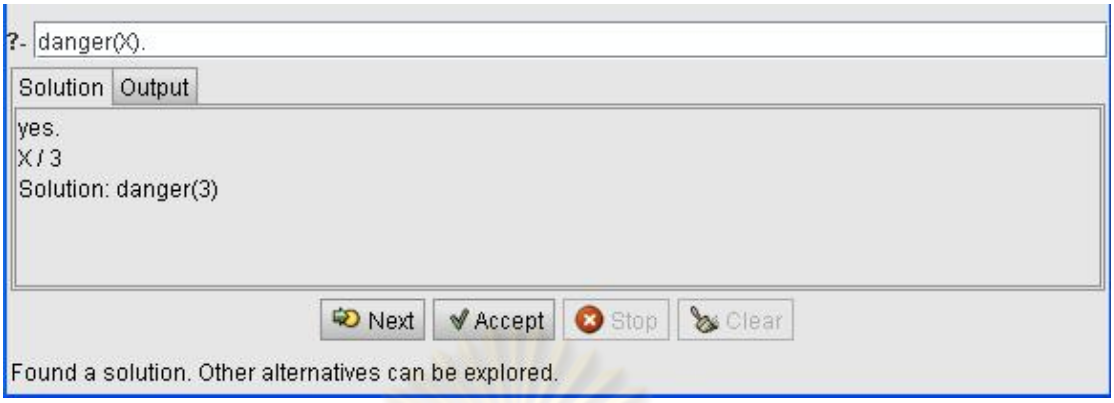
เมื่อต้องการส่งข้อคำถามเข้าไปยังระบบสามารถระบุได้ที่ช่องสีขาวด้านล่างที่มีเครื่องหมาย ?- นำหน้าช่อง ดังรูปที่ ก.3 แล้วทำการกดปุ่ม Enter เพื่อให้ระบบเริ่มค้นหาคำตอบ

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย



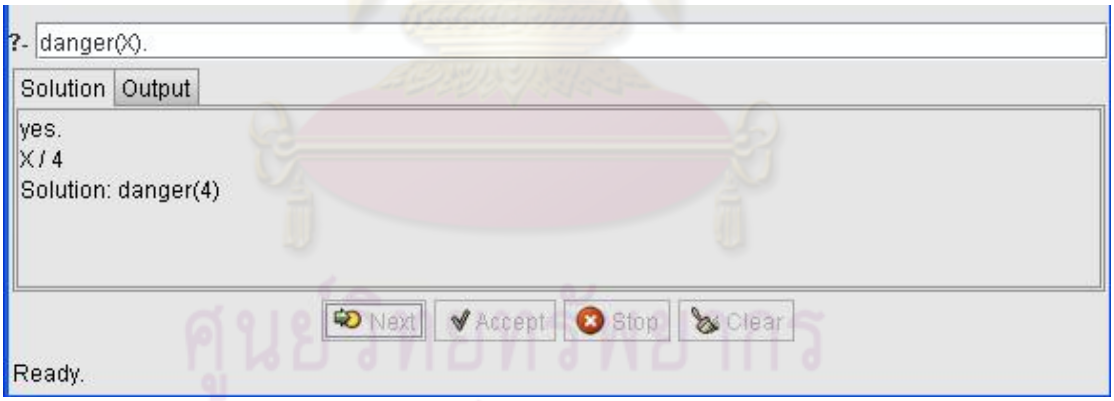
รูปที่ ก.3 การส่งข้อคำถามลงไปยังระบบเซนส์ทูพี

เมื่อระบบสามารถค้นพบคำตอบระบบจากตอบกลับมาว่า yes. แล้วตามด้วยค่าของตัวแปรที่ถูกแทนค่า และคำตอบของข้อคำถามนั้นๆ โดยถ้ายังมีคำตอบอื่นเพิ่มเติมจะขึ้นคำว่า Found a solution. Other alternatives can be explored. ที่ด้านล่าง แต่ถ้าไม่มีคำตอบจะขึ้นว่า No solution found แทน ซึ่งถ้ามีคำตอบอื่นผู้ใช้สามารถกดปุ่ม Next เพื่อขอดูคำตอบอื่น หรือกด Accept เพื่อหยุดการทำงานแต่เพียงเท่านี้ก็ได้ ดังรูปที่ ก.4



รูปที่ ก.4 ระบบเซนส์ทูพีตอบข้อคำถามและยังมีคำตอบอื่นเหลืออยู่

ถ้าผู้ใช้งานระบบกด Next เพื่อขอดูคำตอบอื่นระบบจะทำงานต่อเพื่อหาคำตอบถัดไป แต่ถ้าไม่มีคำตอบใดแล้วระบบจะสิ้นสุดการทำงานและสถานะด้านล่างจะกลับมาเป็น Ready ตามเดิม ดังรูปที่ ก.5



ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

รูปที่ ก.5 ระบบเซนส์แสดงคำตอบครบและหยุดการทำงาน

เมื่อระบบหยุดการทำงานแล้วผู้ใช้ระบบสามารถเปลี่ยนข้อคำถาม เปลี่ยนกฎ หรือเปลี่ยนข้อเท็จจริงและสั่งให้ระบบเริ่มทำงานใหม่ได้ต่อไป

ภาคผนวก ข

การทดสอบการทำงานของโปรแกรมเซนส์ทูพีบนอุปกรณ์จริง

เพื่อเป็นการทดสอบว่าระบบเซนส์ทูพีนั้นสามารถนำไปใช้งานได้จริงจึงได้ทำการทดลองแปลงรหัสโปรแกรม (code) ที่ใช้กับระบบจำลองทอสซิม (TOSSIM) ให้ใช้กับอุปกรณ์จริงคืออุปกรณ์ตัวรับรู้เทลอสบี (TelosB)

โดยทำการทดสอบการจำลองสถานการณ์การตรวจจับหาพื้นที่ที่เป็นอันตรายในการเกิดไฟป่า ซึ่งพื้นที่ที่อาจเกิดไฟป่านั้นมีอยู่ 2 กรณี

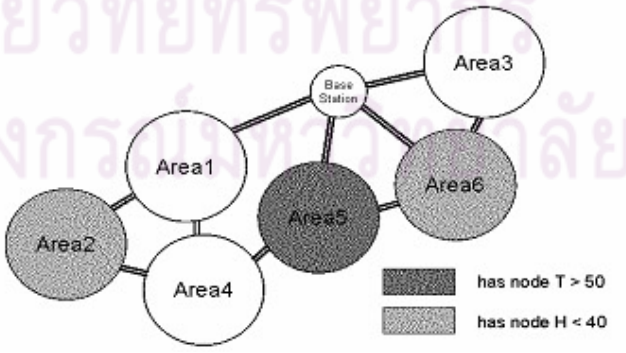
1. มีการตรวจจับอุณหภูมิได้สูงเกินกว่า 50 องศาเซลเซียส หรือเป็นบริเวณที่มีอาจเริ่มมีการติดไฟ
2. เป็นบริเวณที่อากาศแห้งหรือมีความชื้นสัมพัทธ์ต่ำกว่า 40% และมีลมพัดมาจากบริเวณที่อาจมีการติดไฟอยู่ ซึ่งอาจทำให้พื้นที่บริเวณนี้ติดไฟได้เช่นกัน

ซึ่งสามารถนำมาเขียนในรูปของภาษาดาตาล็อกได้ดังนี้

```
danger(ID) :- temperature(ID,T), T > 50.

danger(ID) :- humidity(ID,H), H < 40, winddirection(ID,NID), danger(NID).
```

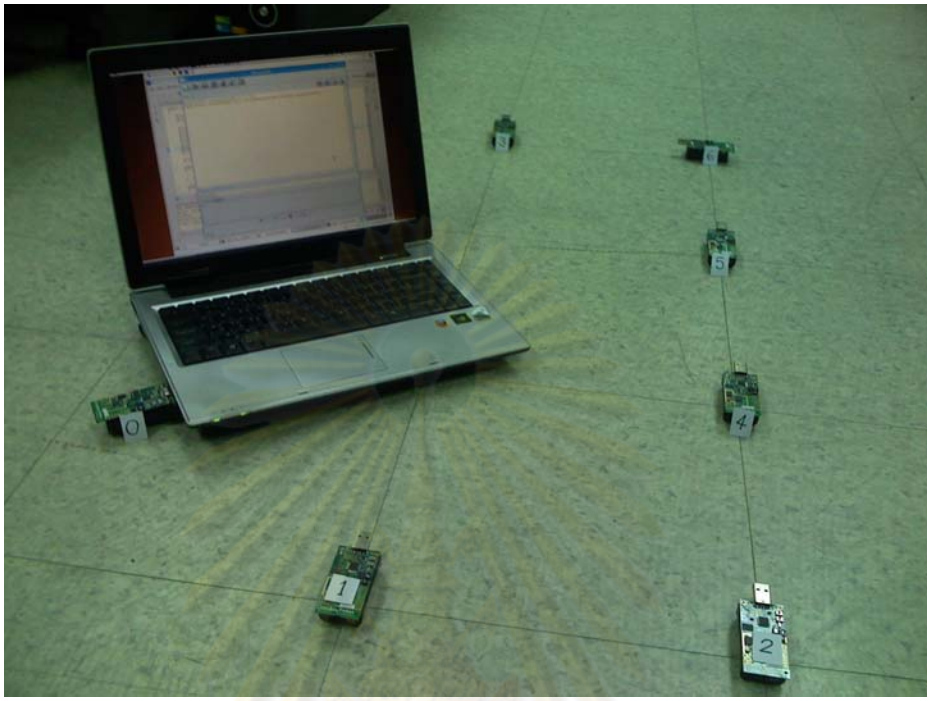
สมมติให้พื้นที่เป็นดังรูปที่ ข.1 คือบริเวณหมายเลข 5 มีการติดไฟอยู่ ส่วนบริเวณหมายเลข 1 และ 6 นั้นเป็นบริเวณที่อากาศแห้ง และมีลมพัดจากบริเวณหมายเลข 5 มายังบริเวณหมายเลข 6



รูปที่ ข.1 สถานการณ์จำลองการตรวจจับไฟป่า

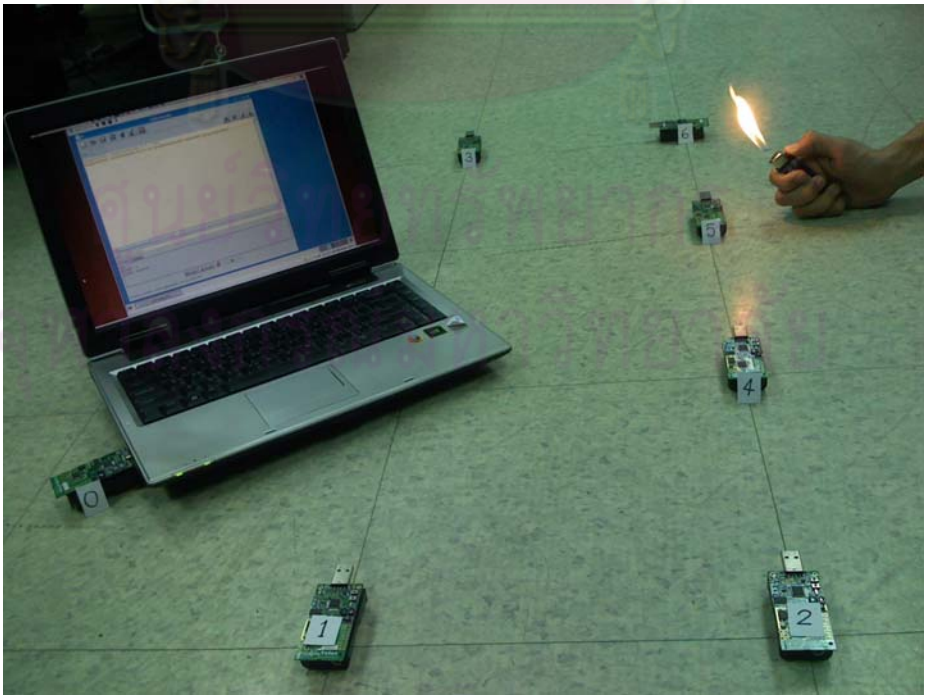
ซึ่งในสถานการณ์นี้พื้นที่ที่เป็นอันตรายในการเกิดไฟป่าคือพื้นที่บริเวณหมายเลข 5 และหมายเลข 6 นั่นเอง

ในการจำลองได้ใช้อุปกรณ์ตัวรับรู้โหมดเทลอสบีจำนวน 7 ตัว แทนการจำลองอาณาบริเวณต่างๆในป่า โดยมีสถานีเชื่อมโยงหมายเลข 0 เป็นสถานีเกตเวย์เชื่อมต่อเข้ากับสถานีฐานซึ่งใช้เป็นเครื่องโน๊ตบุ้ค ดังรูปที่ ข.2



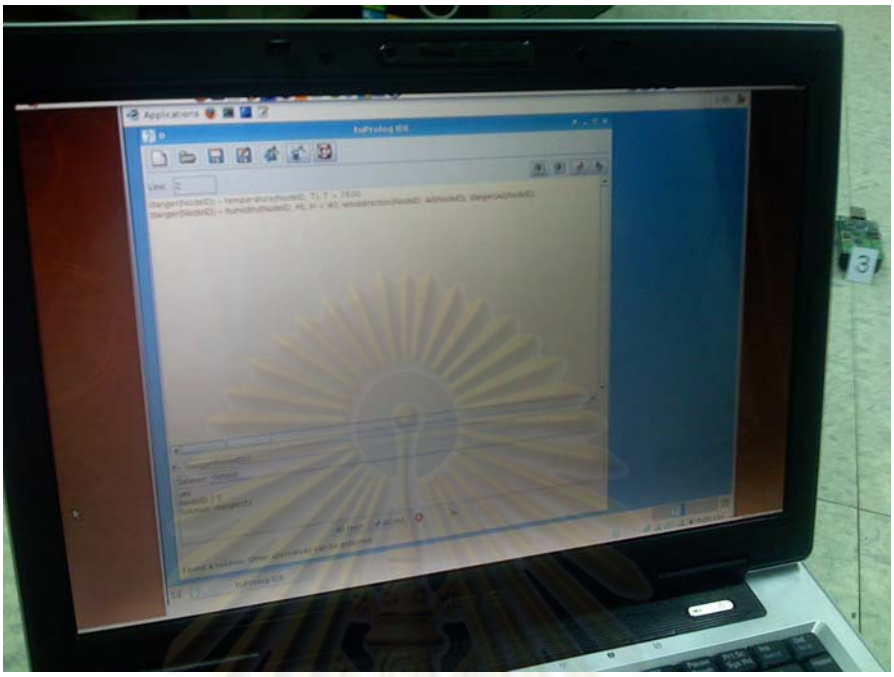
รูปที่ ข.2 การจัดตำแหน่งสถานีเชื่อมโยงเพื่อจำลองสถานการณ์ไฟป่า

หลังจากนั้นจึงนำไฟแช้คมาจุดไฟเพื่อความร้อนในอาณาบริเวณหมายเลข 5 ดังรูป ข.3



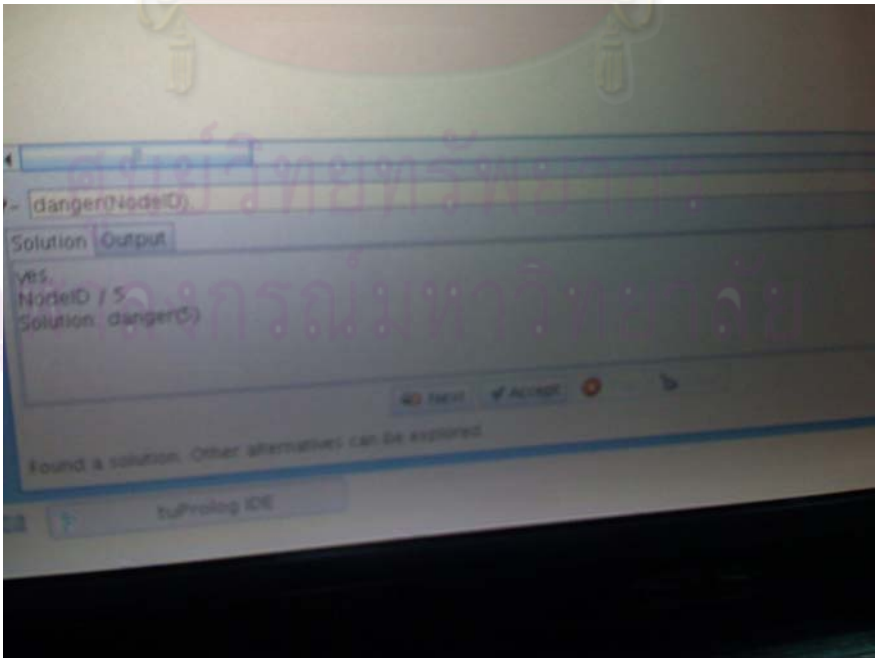
รูปที่ ข.3 ทำการจุดไฟให้ความร้อนแทนบริเวณที่ไฟไหม้ป่า

หลังจากนั้นจึงทำการส่งข้อคำถาม ?-danger(NodeID) ที่สถานีฐานเพื่อขอคำตอบทั้งหมดที่เป็นไปได้ว่าบริเวณใดมีโอกาสเกิดอันตรายจากไฟป่า ดังรูปที่ ข.4



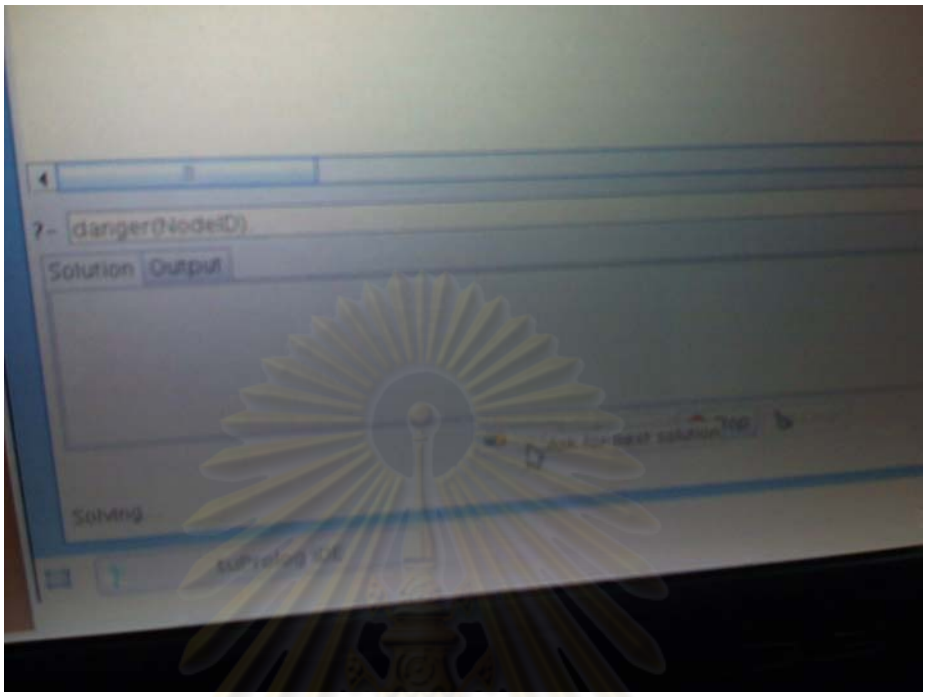
รูปที่ ข.4 หน้าจอโปรแกรมเซนส์ทูพีบนสถานีฐาน

หลังจากนั้นระบบจะทำการหาคำตอบที่เกี่ยวข้องโดยเมื่อมีคำตอบระบบจะแสดงให้เห็นทางด้านล่างของโปรแกรมดังรูปที่ ข.5 โดยจะบอกว่าตัวแปร NodeID จะถูกแทนด้วย 5 ส่วนคำตอบที่ได้คือ danger(5) และในกรณีที่ต้องการดูคำตอบอื่นที่เป็นไปได้ให้กดปุ่ม Next



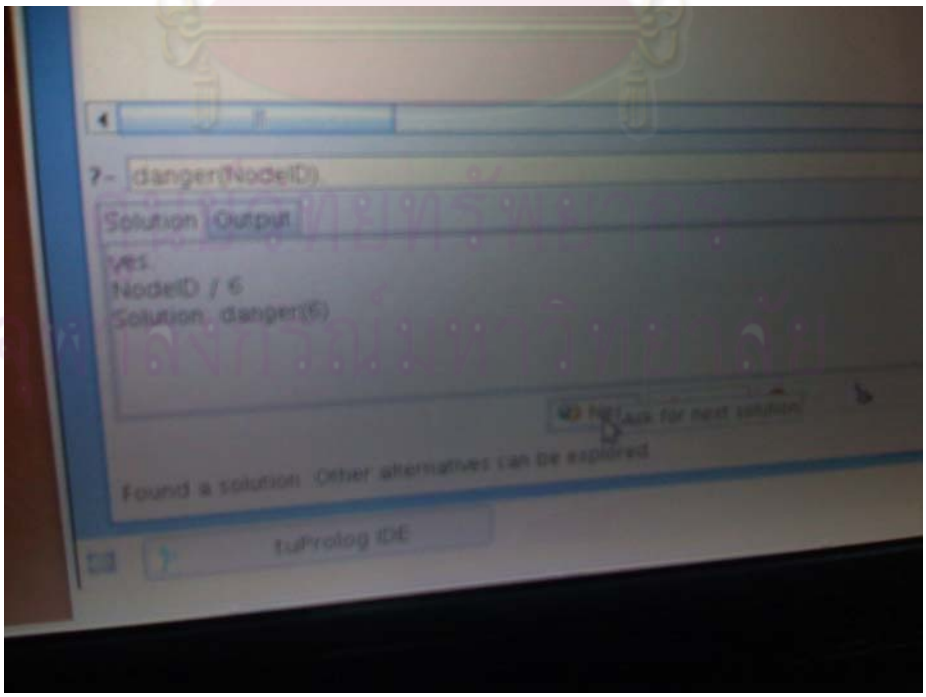
รูปที่ ข.5 การแสดงคำตอบของระบบเซนส์ทูพี

เมื่อกด Next แล้วระบบจะทำการแก้หาคำตอบถัดไปโดยขึ้นสถานะว่า Solving… อยู่ทางด้านล่างดังรูปที่ ข.6



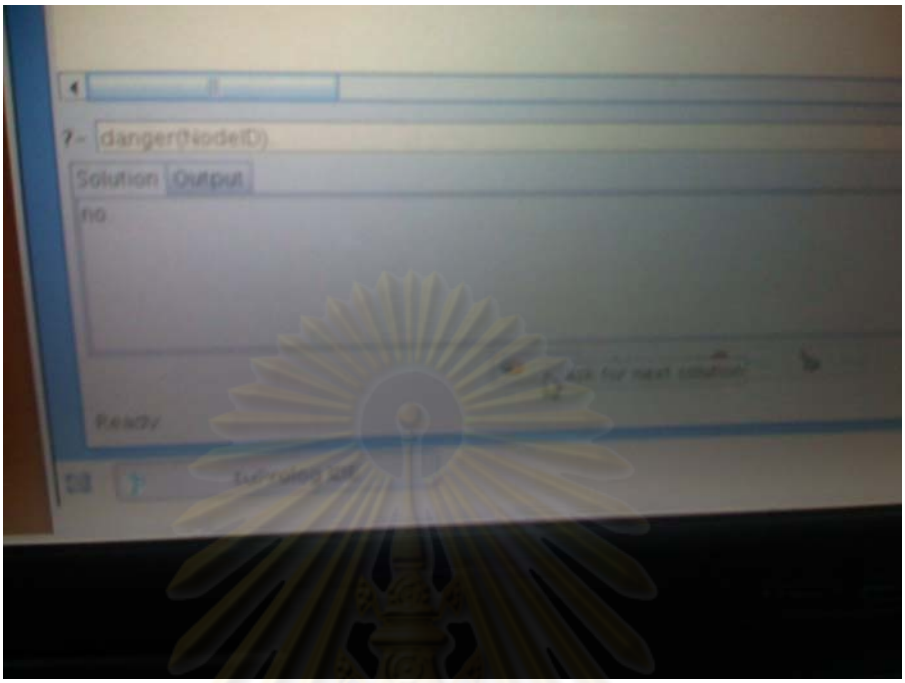
รูปที่ ข.6 ระบบทำการแก้หาคำตอบ

เมื่อพบคำตอบถัดไประบบก็จะทำการแสดงคำตอบออกมา ในที่นี้ก็คือ danger(6) ดังรูปที่ ข.7



รูปที่ ข. 7 รูปแสดงคำตอบถัดไปของระบบ

หลังจากนั้นเมื่อกดเพื่อให้ระบบหาคำตอบถัดไปแต่ไม่มีคำตอบใดที่สอดคล้องกับข้อคำถามแล้ว ระบบจะตอบกลับคืนมาว่า no ดังรูปที่ ข.8 แล้วระบบก็จะสิ้นสุดการประมวลผล



รูปที่ ข.8 ระบบไม่สามารถหาคำตอบที่สอดคล้องเพิ่มได้ สิ้นสุดการทำงาน

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ค

บทความทางวิชาการ

ส่วนหนึ่งของงานวิทยานิพนธ์ได้รับการตีพิมพ์เป็นบทความวิชาการในหัวเรื่อง “A System for Using Wireless Sensor Networks as Globally Deductive Databases” โดย ศุภเสฏฐ์ ชูชัยศรี และ เฉลิมเอก อินทนากรวิวัฒน์ ในงานประชุมเชิงปฏิบัติการ “The First International Workshop on Selected topics in wireless and mobile computing (STWiMob 2008)” ซึ่งเป็นส่วนหนึ่งของงานประชุมวิชาการ “The Fourth IEEE International Conference on Wireless and Mobile Computing, Networking and Communications” ซึ่งจัดขึ้น ณ ศูนย์การประชุมนานาชาติ เมืองอาวิญง ประเทศฝรั่งเศส ระหว่างวันที่ 12-14 ตุลาคม 2551 และได้รับการตีพิมพ์ในรายงานการประชุม “Proceedings of the Fourth IEEE International Conference on Wireless and Mobile Computing, Networking and Communications” หน้า 649 – 654 (DOI = http://doi.ieeecomputersociety.org/10.1109/WiMob.2008.22)

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

IEEE International Conference on Wireless & Mobile Computing, Networking & Communication

A System for Using Wireless Sensor Networks as Globally Deductive Databases

Supasate Choochaisri, and Chalermek Intanagonwiwat, *Chulalongkorn University*

Abstract—**Several research efforts have abstracted wireless sensor networks as relational databases whose data can be easily queried by users. However, none of these works includes recursive query mechanisms and globally logic reasoning techniques necessary for powerful uses as in deductive databases. In this paper, we propose an approach for abstracting wireless sensor networks as deductive databases that consist of rules, predicates, and logic-based queries. Our approach enables globally logic reasoning and data relating on a query that can be recursive or non-recursive. Additionally, the approach is more convenient to users because the users no longer need to be concerned about local behaviors of each sensor node. We also present LogicQ, an underlying system for disseminating and processing logic-based queries as well as collecting data in an energy-efficient manner. LogicQ uses a data filtering and suppressing technique that directly corresponds to an injected logic-based query for energy savings. Our performance analysis indicates that LogicQ can handle logic-based queries efficiently in terms of completeness and soundness while minimizing the energy consumption.**

Index Terms—**Declarative Programming, Deductive Database, Logic Programming, Recursive Query Processing, Wireless Sensor Networks**

I. Introduction

Wireless sensor networks (WSN) have been widely used for collecting data from environments [1-4]. Sensed data types are varied upon a domain in which a user is interested (e.g., temperature, humidity, light intensity). In many domains, different data types are related with each other. For example, habitat monitoring researchers analyze the relation between humidity, temperature, time, and location [1]. Similar to a database, a WSN can be queried for data in relations [5]. Unsurprisingly, researchers have abstracted WSNs as *relational* databases [6, 7].

The relational database has many advantages and has been widely used in traditional database systems. SQL is the de facto standard for querying data from such systems. Due to its *declarative* characteristics, SQL has also been adopted for querying data from WSNs [8, 9]. However, these recent works still have the following limitations (see also Section II).

Manuscript received May 30, 2008.

S. Choochaisri is with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, 10330, Thailand (e-mail: g49sch@cp.eng.chula.ac.th).

C. Intanagonwiwat is with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, 10330, Thailand (e-mail: intanago@yahoo.com).

First, supported queries in the previous works are quite limited. For example, there is only one table accessible at a time. This may not work in networks of heterogeneous sensors. In addition, only conjunctive comparison predicates are supported and arithmetic expressions are limited to operations of an attribute and a constant. As a result, tuple selection is inflexible. Furthermore, sub-queries and column aliases are not allowed either.

Second, each sensed data item is kept as a tuple associated with each node. Constraints in the query are applied only to attributes in the same tuple as well as the same node [8]. Therefore, the constraints are local, not global. It is not designed for deriving data that is related with other data from different nodes.

Third, there is no *recursive* query feature that is designed for an application-level user (instead of a low-level programmer). It is well documented that the recursive query can improve the capability of a database [10, 11]. Even though one can express a recursive query in SQL, the recursive SQL query is rather verbose and unsupported in the previous SQL systems for WSNs. Additionally, other WSN systems with non-SQL recursive features are not convenient to application-level users as they require low-level programming for routing and query processing [12].

Finally, previous systems with a relational abstraction do not support a logic-based query frequently used in deductive databases and expert systems [8, 9, 13].

In this paper, we propose an abstraction of WSNs as deductive databases that enable recursive logic-based queries for application-level users (Section III, and IV). Unlike previous systems, data can be globally deduced from heterogeneous sensors and flexibly selected as needed. Furthermore, we also present LogicQ, an underlying system for processing logic-based sub-queries in an energy-efficient manner.

We have implemented LogicQ in TinyOS [14] (Section V) and evaluated with TOSSIM, the TinyOS simulator [15] (Section VI). Our result indicates that LogicQ completely supports logic-based queries while consuming minimal energy. However, several mechanisms can still be improved and extensive evaluation is needed in order to realize the full potential of our work (Section VII).

978-0-7695-3393-3/08 $25.00 © 2008 IEEE
DOI 10.1109/WiMob.2008.22

IEEE computer society

II. Related Work

Our work has been informed and influenced by a variety of research efforts as follow.

A. Wireless sensor networks as relational databases

The WSN abstraction as a relational database has first been introduced in the COUGAR project [7]. Each sensed data item is represented by a tuple in a table of a backend relational database whose queries are formulated in an SQL variant. An approach for in-network query processing for this system has also been proposed [8, 13].

The combination of COUGAR and directed diffusion [5] is considered a precursor of motes and TinyDB [8], a distributed query processor for database management in WSN. As a part of TinyDB, TAG [13] focuses on query processing in the network and provides a service for users to aggregate sensed data while minimizing power consumption.

Similarly, Govindan et al. [6] have presented that the sensed data item should be stored in a virtual relational table that preserves *location transparency*.

All of the above works are designed as the relational database model. However, unlike our work, none of these systems can process a recursive query at all.

B. Wireless sensor networks as deductive databases

Decades ago, logic programming is combined with relational databases in order to construct deductive database systems that support a powerful formalism and operate quickly even with very large data sets. Their powerful features include a capability to process recursive queries and their superior expressiveness over relational databases. For example, a PROLOG system [17] can be loosely coupled with a relational database system [18] to become a deductive database system (i.e., a relational database system with an inference engine). A frequently-used query language for deductive databases is *Datalog*, a simplified logic programming language [19] that consists of facts and rules (see Section III-A for more details). Our early work in abstracting WSNs as deductive databases has been presented in [20].

Recently, Chu et al [12] have further developed the concept into Snlog, a dialect of Datalog, for programming WSNs and for enabling recursive queries. Snlog, however, is designed to be a *declarative* programming method for low-level programmers, not a declarative query method for application-level users. Unlike users in our paradigm, Snlog programmers must write rules by focusing on local behaviors of each sensor node (instead of the global behavior as a whole). Additionally, the Snlog programmer must deal with networking details and protocols, such as routing, query disseminating, data collecting. Conversely, deductive database users in our systems declare only what data they need, not how to get the data.

III. Wireless Sensor Networks as Globally Deductive Databases

In this section, we describe our architecture and abstraction of WSNs as globally deductive databases. Datalog is used in this explanation as the query language.

In brief, deductive databases consist of *predicates*, *facts*, *rules, and queries*. Predicates are relations of data or tables in the relational database terminology. For example, a predicate *temperature(NodeID, TemperatureValue)* describes a relation between a sensor node id and a reading temperature value.

Facts are already-existing data in a system, such as *temperature(2, 45)*, the sensed temperature of 45 degrees at a node with the identification number 2. Rules are clauses that deduce new facts from existing facts. Rules are represented as *Horn* clauses that contain head and body parts. An example of a rule is shown in Listing 1.

```
hasHotSpotArea(AreaID):- temperature(NodeID, Temp)
                       , Temp > 50
                       , area(NodeID, AreaID).
```

Listing 1. Example of a rule.

Specifically, an area has a hot spot if an arbitrary node in that area senses temperature with a value over 50 degrees. The left hand side of a clause, *hasHotSpotArea(AreaID),* is called *head* and the right hand side, *temperature(NodeID, Temp), Temp > 50, area(NodeID, AreaID)*, is called *body*. A rule will be satisfied only if every predicates in the body are satisfied or substituted by at least one fact.

Queries are questions that a user asks to retrieve data from the system. Queries can be classified into 4 groups. The first group is a fact-checking query that is intended for checking the existences of certain facts. The second group includes fact-retrieving queries that are designed for retrieving all data that satisfy the fact types and constraints in the queries. The third group consists of queries for checking the existences of certain rules. The final group is composed of deductive queries for retrieving all data that satisfy the rules in the queries. A query is represented by *?-* followed by a predicate and a dot. For example, *?- hasHotSpotArea(X).*, is a query to retrieve ids of all areas that has a hot sensor node.

A query will be recursive if its predicate matches with a recursive rule whose body contains the same predicate name as that in its head. For example, one can write recursive rules for detecting sensor nodes in danger as shown in Listing 2.

```
Rule1: danger(AreaID):- temperature(NodeID, T), T > 50
                      , area(NodeID,AreaID).
Rule2: danger(AreaID):- humidity(NodeID, H), H < 40
                      , area(NodeID, AreaID)
                      , adjacent(AreaID, AdjAreaID)
                      , danger(AdjAreaID).
Query: ?-danger(X).
```

Listing 2. Example of recursive rules.

The rule 1 is a base case of a recursive program that describes properties of areas in danger. An area will be in

danger if there is at least one node (in that area) whose sensed temperature is greater than 50 degrees. The rule 2 is a recursive case. Basically, an area will be in danger if there is at least one node (in the area) whose sensed relative humidity is lower than 40 % and its adjacent area is in danger.

Unlike Snlog, our database is *globally* deductive. Specifically, we abstract the entire WSN as one large deductive database. New facts are globally deduced from facts in different nodes rather than locally deduced in each node as in prior works. Therefore, users do not need to construct rules that depending on local behaviors of each node. Specifically, users do not need to program low-level mechanisms such as query dissemination, route computation, data collection, etc.

IV. Query Processing

Query processing schemes for deductive databases has been proposed for many years. However, none of those schemes is designed for WSNs. The mentioned schemes can be categorized into three main groups: top-down, bottom-up, and Prolog-style evaluation approaches [18].

Prolog-style systems (coupled with database systems) are similar to the top-down systems in a sense that their execution starts from the goal and a query can be solved by executing each sub-goal until the deduced facts match the goal. However, Prolog-style systems produce answers one tuple at a time whereas top-down methods produce one set at a time without in-order execution of sub-goals.

Conversely, the bottom-up methods start from existing facts and attempt to deduce new facts from rules that are related to the query. Only facts that match with the goal of the query are selected as the answers. Due to the page limitation, we refer to [18] for more information of each implementation scheme.

Many works suggest that the bottom-up methods have many advantages over top-down methods in traditional deductive database systems [11, 18, 21]. However, in WSNs, we argue that top-down and Prolog-style approaches are more appropriate.

Understandably, facts in a WSN are data that are sensed from an environment. They are locally kept within sensing nodes and sent to the base station only when requested. To process a query in a bottom-up manner, we need facts from all relevant nodes so that new facts can be globally deduced. Therefore, each node may be required to send its data to a rendezvous point (e.g., a base station, an inference engine) for such a deduction. Undoubtedly, the mentioned mechanism consumes excessive energy. To reduce this energy consumption, only relevant data should be delivered.

However, it is not easy to selectively send relevant data because we do not know priori which nodes happen to have certain sensed values or facts that definitely satisfy our rules. A fact in a node may be relevant simply because another fact somewhere else happens to have a certain value.

Conversely, the top-down approach can use information from a query to suppress irrelevant facts from being sent. For example, if the system has a predicate *detect(ObjectID, AreaID)*, which means a sensor node can detect an object with the identification number *ObjectID* in the region *AreaID*. When a user injects a query *?- detect(oiltank,X)*, only sensor nodes that can detect an object named *oiltank* will send answers back. Even though this idea is similar to TinyDB [8], the difference becomes more evident in other queries, such as *?-detect(oiltank, area70)*. In our system, only the first node detecting *oiltank* will reply, although there may be other nodes that detect the same event. This is reasonable, given that one reply about the fact existence is sufficient to satisfy the query. Therefore, the first detecting node does not need to forward the query further. Thus, there is no other replier (see Section V for more details). Conversely, each node in TinyDB still forwards the query to all other nodes in the network; therefore, energy is consumed unnecessarily.

For a recursive query, the top-down strategy can also be used effectively. The inference engine for processing recursive queries should reside on the central server instead of distributed sensor nodes. A reason is that a user usually injects queries from a central server, not from arbitrary sensor nodes. The results must be delivered to the user eventually.

In addition, distributed inference engines may be costly because they still need relevant facts. The relevance of a fact may depend on values of other facts somewhere else. Naively pulling data from all over the network to several inference engines may incur excessive energy consumption.

In our system, most relevant facts are pulled from the network except the persistent ones that do not change over time. The persistent facts can be cached or kept in the backend database of the inference engine for future uses.

Top-down evaluation methods still have an additional advantage. We can use an answer set from the previous sub-goal to filter out (or suppress) the irrelevant facts of the next sub-goal. For example, consider the rule in Listing 3.

```
hotObject(ObjectID, AreaID) :- detect(ObjectID, AreaID)
                             , temperature(ObjectID, Temp)
                             ,Temp > 50.
```

Listing 3. Example of a rule that two predicates related to each other with ObjectID.

When a user injects a query *?-hotObject(X, area70).*, the system will match the query with the above rule. Therefore, the variable *AreaID* in the rule will be bound with the constant *area70*. Then, the system will attempt to match each predicate in the body of the rule. Each body predicate becomes a sub-query that needs to be satisfied.

In this example, the first sub-query is *detect(ObjectID, area70)*. This sub-query is disseminated into the network. Only eligible repliers are nodes with facts or rules that match the sub-query. Others are suppressed. Consequently, only objects in *area70* will be bound to the variable *ObjectID*. Then, each ObjectID will be used to bind the *temperature(ObjectID, Temp)* predicate and can be used to filter out or suppress irrelevant facts from being sent. Furthermore, we can also use a

constraint *Temp > 50* as another filter before injecting a sub-query *temperature(ObjectID, Temp)* to the network.

Due to this filtering technique, this top-down approach can significantly reduce the consumption of energy that is limited in WSNs [5].

V. LogicQ

In this section, we present LogicQ, the underlying system for sub-query processing in WSNs. Running on each sensor node, LogicQ is implemented in TinyOS [14], the de facto operating system for the motes. The functionality of LogicQ is to find answers for each sub-goal that needs data from WSNs.

A. Built-in predicate

Sub-goal predicates that LogicQ is responsible to support are built-in predicates related to sensor nodes. Such predicates include predicates that are related to specific functions of sensor nodes, such as *temperature(NodeID, Temperature)* for sensing the temperature, *connect(NodeID1, NodeID2)* for checking connectivity, etc. These built-in functions are predefined during the system installation. The inference engine will solve the sub-goals that match with these built-in predicates by injecting sub-queries that corresponds to the sub-goals into the network.

B. Routing Tree

When sensor nodes start up, LogicQ constructs a routing tree for disseminating sub-queries from the base station to sensor nodes and for collecting answers that satisfy the sub-queries. We use a *drain* tree of TinyOS as our routing tree. A root of the tree is the gateway node connected to the base station. Each node can have many child nodes but only one parent node. When disseminating the queries, we simply forward queries along the drain tree except suppressible queries (i.e., no longer necessary to be forwarded because the queries have been satisfied).

C. Sub-query type

In Section III-A, we classify queries into 4 groups, however, in a sensor node's view, each sub-query can be classified into two types: one for existence checking and another for retrieving all satisfied predicates. To check an existence of a fact, every argument in this first query type is constant and the answer is only *true* or *false* (e.g., *detect(oiltank, area70)* whereby *oiltank* and *area70* are constant). Therefore, this type of query is not necessarily disseminated to all sensor nodes. If only one node has a fact that satisfies the query, the system does not need answers from other nodes. Conversely, the second query type requires at least one variable as an argument. For example, *detect(ObjectID, area70)* contains a variable *ObjectID* and a constant *area70*. Hence, the system will find every possible answer of *ObjectID* that is detected in *area70*. It is necessary to disseminate this type of queries to all nodes.

D. Processing algorithm

The sub-query processing algorithm for each sensor node can be written as a pseudocode in Listing 4. Once receiving an existence-checking query (Line 1), a sensor node checks its facts locally first whether it has a fact that satisfies the query or not (Line 2). If a sensor node has a satisfying fact, it will send an answer "true" to its parent immediately (Line 3). Given that one answer is sufficient for this query type, the replying node does not further forward the query. Otherwise, it will forward the query to its children (Line 4 - 5).

If the query type requires all satisfied answers (Line 6), a sensor node will forward the query immediately (Line 7). Regardless of the local existence of the satisfying facts, the system still needs satisfying answers from all sensor nodes. After forwarding the query, the node checks for local satisfying answers. If it has one, it will send the answer up to its parent (Line 8 – 9).

```
1: if subquery_type == ask_for_checking_existence
2:     if have_local_satisfied_fact
3:         send_answer_up(true)
4:     else
5:         forward_query()
6: else if sub subquery_type == ask_for_all_satisfied
7:     forward_query()
8:     if have_local_satisfied_fact
9:         send_answer_up(answerset)
```

Listing 4. Sub-query processing algorithm.

Given this query-suppression mechanism, LogicQ can reduce unnecessary transmissions. In addition to our recursive queries, our query dissemination differs from TinyDB's that disseminates every query to all nodes.

VI. Preliminary Evaluation

We conduct our preliminary evaluation of LogicQ on TOSSIM (the TinyOS simulator). We assume the reliable communication (i.e., no packet loss because of bit errors or collisions) to discard the problem caused by radio transmissions. Given that LogicQ is an underlying layer, we also assume that there is an inference engine running on the base station. This engine is responsible for injecting each sub-goal of the rule to the network.

We evaluate LogicQ in 3 metrics: completeness, soundness, and communication cost. Completeness is measured by the ratio of the number of retrieved answers to the total number of existing answers in the networks. We measure the soundness by the ratio of the number of relevant answers retrieved to the total number of retrieved answers. Communication cost indicates the amount of energy consumed by the system. In this evaluation, the communication cost is measured by the number of sent messages in the system.

We compare our proposed top-down method with the bottom-up method to justify the viability of our approach. In the bottom-up method, all facts are sent to the base-station for the inference engine to process. We do not take the cost of routing into account because we use the same routing tree for

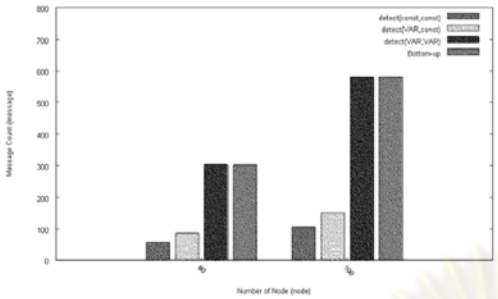


Fig. 1. Communication cost of the query *detect(ObjectID, AreaID)*

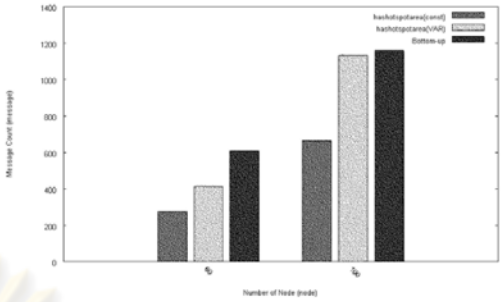


Fig. 2. Communication cost of the query *hashotspotarea(ObjectID, AreaID)*

both methods. Finally, we will also show that LogicQ completely supports recursive queries.

Our preliminary result indicates that LogicQ can preserve nearly 100% completeness of the answer set. Under our investigated scenarios, LogicQ performs slightly better than the bottom-up method does. The missing answers in this simulation are due to the packet loss that is caused by the buffer overflow, not by the radio transmission. Given that our suppressing mechanism can decrease the amount of messages in the system, the probability of buffer overflow is reduced. Thus, LogicQ can achieve better completeness. However, to achieve 100% completeness, a congestion control technique for WSNs may be required.

LogicQ also preserves 100% soundness because of its suppressing technique. The irrelevant answers of each node will be suppressed from sending to its parent as expected.

Furthermore, LogicQ incurs significantly less communication cost than the bottom-up method does in most query types (Figure 1). In this simulation, we count the number of messages sent in the our system in order to answer 3 simple queries using a simple predicate, *detect(ObjectID, AreaID)*. The first query is for checking the existence of a predicate. All arguments in the query are constant. The second query contains one constant argument and one variable argument. In the third query, all arguments are variable.

Our message counts in answering three mentioned queries are compared with that of the bottom-up method. Regardless of the argument types, the bottom-up approach always incurs a certain amount of messages sent because all facts must be delivered to the base station.

LogicQ will significantly reduce the communication cost if the query contains at least one constant argument to suppress irrelevant answers. However, LogicQ will incur the communication cost similar to that of the bottom-up approach if all arguments in the query are variable. Understandably, all facts are required in order to answer such a non-constant query under our investigated scenarios.

Nevertheless, in our simulation, we assume all nodes are equipped with the same sensor type. If the sensor nodes are heterogeneous, LogicQ will still reduce the communication cost significantly even with the non-constant query because only relevant nodes with the matched sensing capability will send back the data, unlike the bottom-up approach that needs all facts and filters out by the inference engine at the base station.

We use the rule in Listing 1 to evaluate our system about its capability to process sub-queries. We simulate our system by injecting each sub-goal of the query *hasHotSpotArea(AreaID)* that corresponds to the above rule. Expectedly, the result (Figure 2) indicates that our system outperforms the bottom-up approach regardless of the query types (including the non-constant type).

However, our savings in non-constant queries is reduced when the number of nodes is increased. This indicates that our system does not scale well for non-constant queries. This problem is due to the naive implementation of our approach. In our implementation, each answer from the previous sub-goal is used to bind the variable in the current sub-goal. The number of sub-queries for the current sub-goal depends on the number of answers from the previous sub-goal because each answer may bind the variable with a different value.

As the number of nodes is increased, the number of answers for the previous sub-goal is also increased. Consequently, the number of sub-queries for the current sub-goal is unavoidably increased. This causes more messages sent into larger networks. However, this problem can be solved by sending only one sub-query for the current sub-goal with a list of different values that are bound with the variable. Nevertheless, we have not yet implemented this optimization in this paper. We intend to further explore this technique and other optimization approaches in our future work.

Finally, the rules in Listing 2 are used for preliminary testing of LogicQ's capability to support the recursive query. In our scenario (Figure 3), each area contains 10 nodes equipped with temperature and humidity sensors. The base station contains static facts of the *adjacent* predicate. For example, the fact *adjacent(2,1)* means that the area 2 is connected to the area 1. Area 5 is the area that has at least one node whose sensed temperature is higher than 50 degree that implies danger. Area 2 and area 6 are the areas where there are at least one node with low humidity that implies possible danger if at least one of their adjacent areas is in danger.

Under this scenario, LogicQ correctly returns area 5 and area 6 as its answers. However, LogicQ sends slightly fewer

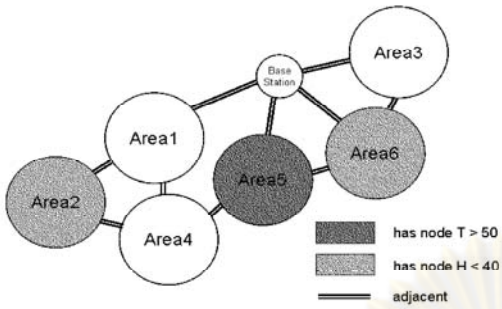


Fig. 3. Simulation scenario for recursive query; each area contains 10 nodes.

messages than the bottom-up method does because, in our simulations, several nodes sense relative humidity lower than 40 % in area 2 and 6. This leads to injecting several sub-queries for the next sub-goal. However, with an optimization previously mentioned, the number of sub-queries and its communication cost will be further reduced.

VII. Future Work

This paper is considered the very first attempt to abstract WSNs as globally deductive databases. Even though our evaluation indicates that LogicQ is reasonably efficient, LogicQ can still be further improved, especially in terms of the communication cost. For example, we aim to include the argument-binding list for each sub-goal variable to further reduce the number of the sub-queries. We also plan to integrate LogicQ into an off-the-shelf Prolog system and extensively analyze the impact of the integration.

VIII. Conclusion

This paper proposes a novel abstraction of WSNs as globally deductive databases. The rule-based approach of the deductive database can empower WSNs with globally logic reasoning. Our global approach enables users to write rules and queries without being concerned with low-level local behaviors of each sensor node.

To efficiently process queries and their sub-queries (either recursive or non-recursive), the top-down approach is more appropriate than the bottom-up approach. This is due to its capability to bind sub-goal arguments that can be used to reduce communication cost by suppressing irrelevant answers and already-satisfied sub-queries from being sent or forwarded.

We also present LogicQ, an underlying system for disseminating and processing logic-based queries. LogicQ can reduce the energy consumption of the system while maintaining 100% soundness and nearly 100% completeness under our investigated scenarios. To achieve 100% completeness, a congestion control may be required.

References

[1] M. Alan, C. David, P. Joseph, S. Robert, and A. John, "Wireless sensor networks for habitat monitoring," in Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, Atlanta, Georgia, USA, 2002.

[2] C. Gui and P. Mohapatra, "Power conservation and quality of surveillance in target tracking sensor networks," in Proceedings of the 10th annual international conference on Mobile computing and networking, Philadelphia, PA, USA, 2004.

[3] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in Proceedings of the 2nd international conference on Embedded networked sensor systems, Baltimore, MD, USA, 2004.

[4] T. He, S. Krishnamurthy, J. Stankovic, A. , T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-efficient surveillance system using wireless sensor networks," in Proceedings of the 2nd international conference on Mobile systems, applications, and services, Boston, MA, USA, 2004.

[5] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," in IEEE/ACM Transactions on Networking (TON). vol. 11, pp. 2-16: IEEE Press, 2003.

[6] R. Govindan, J. M. Hellerstein, W. Hong, S. F. Madden, Michael , and S. Shenker, "The sensor network as a database," in USC Computer Science Department Technical Report, 2002.

[7] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in Proceedings of the Second International Conference on Mobile Data Management: Springer-Verlag, 2001.

[8] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," ACM Trans. Database Syst., vol. 30, pp. 122-173, 2005.

[9] Y. Yao and J. Gehrke, "Query processing for sensor networks," in Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003), 2003.

[10] F. Bancilhon and R. Ramakrishnan, "An amateur's introduction to recursive query processing strategies," in Readings in database systems: Morgan Kaufmann Publishers Inc., pp. 507-555, 1988.

[11] Y. K. Hinz, "Datalog bottom-up is the trend in the deductive database evaluation strategy," in Technical Report INSS 690: University of Maryland, 2002.

[12] D. Chu, L. Popa, A. Tavakoli, J. M. Hellerstein, P. Levis, S. Shenker, and I. Stoica, "The design and implementation of a declarative sensor network system," in Proceedings of the 5th international conference on Embedded networked sensor systems, Sydney, Australia, 2007.

[13] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for ad-hoc sensor networks," SIGOPS Oper. Syst. Rev., vol. 36, pp. 131-146, 2002.

[14] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "TinyOS: an operating system for wireless sensor networks," in Ambient Intelligence: Springer-Verlag, 2004.

[15] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: accurate and scalable simulation of entire TinyOS applications," in Proceedings of the 1st international conference on Embedded networked sensor systems, Los Angeles, California, USA, 2003.

[16] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," SIGMOD Rec., vol. 31, pp. 9-18, 2002.

[17] W. F. Clocksin and C. S. Mellish, Programming in Prolog (2nd ed.): Springer-Verlag, New York, Inc., 1984.

[18] K. Ramamohanarao and J. Harland, "An introduction to deductive database languages and systems," VLDB Journal, vol. 3, pp. 107 - 122, 1994.

[19] S. Ceri, G. Gottlob, and L. Tanca, "What you always wanted to know about datalog (and never dared to ask)," IEEE Transactions on Knowledge and Data Engineering, vol. 01, pp. 146-166, 1989.

[20] S. Choochaisri, "A development of deductive database components for wireless sensor networks," in Bachelor Senior Project, Department of Computer Engineering, Chulalongkorn University, 2005.

[21] R. Ramakrishnan and S. Sudarshan, "Top-down vs. bottom-up revisited," in Proceedings of the International Symposium on Logic Programming (ISLP'91), 1991.

ประวัติผู้เขียนวิทยานิพนธ์

นายศุภเสฏฐ์ ชูชัยศรี เกิดเมื่อวันที่ 19 เมษายน พ.ศ. 2527 ที่จังหวัดกรุงเทพฯ และย้ายภูมิลำเนาไปอยู่ที่จังหวัดปทุมธานี สำเร็จการศึกษาระดับประถมศึกษาจากโรงเรียนอัมพรไพศาล จังหวัดนนทบุรี สำเร็จการศึกษาระดับมัธยมศึกษาจากโรงเรียนสวนกุหลาบวิทยาลัยนนทบุรี จังหวัดนนทบุรี สำเร็จการศึกษาปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2548 (เกียรตินิยมอันดับ 2) และเข้าศึกษาในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2549

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย