อัลกอริทึมควบคุมความหนาแน่นบนพื้นฐานเซตครอบคลุม สำหรับปัญหาครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย

นายศรัณย์ เจนจตุรงค์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2550

A SET COVER-BASED DENSITY CONTROL ALGORITHM FOR COVERAGE PROBLEMS IN WIRELESS SENSOR NETWORKS

Mr.Saran Jenjaturong

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 2007

หัวข้อวิทยานิพนธ์	อัลกอริทึมควบคุมความหนาแน่นบนพื้นฐานเซตครอบคลุมสำหรับปัญหาครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย
โดย	นายศรัณย์ เจนจตุรงค์
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.เฉลิมเอก อินทนากรวิวัฒน์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับ
เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

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

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.อรรถสิทธิ์ สุรฤกษ์)

..... อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์ ดร.เฉลิมเอก อินทนากรวิวัฒน์)

..... กรรมการ
(รองศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล)

..... กรรมการ
(รองศาสตราจารย์ ดร.อนันต์ ผลเพิ่ม)

ศรัณย์ เจนจตุรงค์ : อัลกอริทึมควบคุมความหนาแน่นบนพื้นฐานเซตครอบคลุมสำหรับปัญหาครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย (A SET COVER-BASED DENSITY CONTROL ALGORITHM FOR COVERAGE PROBLEMS IN WIRELESS SENSOR NETWORKS)
อ.ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.เฉลิมเอก อินทนากรวิวัฒน์, 106 หน้า

ระบบเครือข่ายตัวรับรู้แบบไร้สายเป็นระบบเครือข่ายชนิดหนึ่งที่ประกอบไปด้วยสถานีตัวรับรู้ไร้สายขนาดเล็กที่มีแหล่งกำเนิดพลังงานสำหรับใช้ในการทำงานอยู่อย่างจำกัด ดังนั้นเพื่อที่จะขยายระยะเวลาการทำงานของระบบให้อยู่ได้นานขึ้นจึงจำเป็นที่จะต้องมีกระบวนการบางอย่างในการช่วยลดปริมาณการใช้พลังงานของระบบให้น้อยลง วิทยานิพนธ์นี้ได้นำเสนออัลกอริทึมควบคุมความหนาแน่นแบบเฉพาะที่สำหรับการประหยัดพลังงาน โดยที่มีจุดประสงค์เพื่อช่วยลดปริมาณจำนวนของสถานีตัวรับรู้ที่จำเป็นต้องเปิดทำงานรวมไปถึงความหนาแน่นของปริมาณการใช้ช่องสัญญาณวิทยุให้น้อยลงในขณะที่ยังสามารถคงสภาพพื้นที่ครอบคลุมการตรวจจับเดิมของระบบเอาไว้ได้

วิทยานิพนธ์นี้ได้แปลงปัญหาพื้นที่ครอบคลุมการตรวจจับไปเป็นปัญหาเซตครอบคลุมแบบถ่วงน้ำหนัก สถานีตัวรับรู้แต่ละตัวจะทำการคำนวณผลเฉลยตามหลักเชิงละโมบจากการแก้ปัญหาเซตครอบคลุมของตัวเอง ซึ่งผลเฉลยนี้จะบ่งชี้ถึงกลุ่มของสถานีตัวรับรู้จากในหมู่สถานีทั้งหมดที่มีตัวมันเองรวมกับสถานีเพื่อนบ้านของมันที่สมควรจะหลับไปเพื่อประหยัดพลังงาน อย่างไรก็ตามผลเฉลยที่ได้จากตัวสถานีตัวรับรู้เทียบกับผลที่ได้จากสถานีเพื่อนบ้านอาจไม่ตรงกัน ดังนั้นในวิทยานิพนธ์นี้ยังได้นำเสนอแผนการลงมติสำหรับการเลือกหลับสถานีตัวรับรู้เพื่อให้มั่นใจได้ว่าสถานีตัวรับรู้ที่ถูกเลือกนั้นเป็นสถานีตัวรับรู้ที่เหมาะสมที่สุดในอาณาบริเวณนั้นที่สมควรแก่การหลับโดยที่ไม่ทำการครอบคลุมพื้นที่ตรวจจับของระบบเดิมแย่ลง

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....
สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษา .....
ปีการศึกษา 2550

4970589821 : MAJOR COMPUTER ENGINEERING
KEY WORD : WIRELESS SENSOR NETWORKS/ SENSING COVERAGE/ DENSITY CONTROL/ SET COVER PROBLEM/ NODE SCHEDULING/ VOTING PROTOCOL

SARAN JENJATURONG : A SET COVER-BASED DENSITY CONTROL ALGORITHM FOR COVERAGE PROBLEMS IN WIRELESS SENSOR NETWORKS. THESIS ADVISOR : ASST. PROF. CHALERMEK INTANAGONWIWAT, Ph.D., 106 pp.

Wireless sensor networks consist of a large number of wireless sensor nodes with limited power and resource. To prolong network lifetime, the energy consumption must be somehow reduced. This work proposes a localized density control algorithm for energy savings. The goals are to maintain a minimal number of active sensor nodes and to reduce radio-traffic intensity while conserving the sensing coverage of the network.

In this thesis, the sensing coverage problem is transformed into a weighted set-cover problem. Each node locally computes a greedy solution of such a problem. This localized greedy solution indicates candidate nodes among this node and its neighbors for sleeping. However, the solution of a node and those of its neighbors might be different. Therefore, this work also includes a voting scheme for selecting inactive nodes to assure that the selected nodes are the most deserving nodes in the area to sleep without worsening the sensing coverage.

Department ....Computer Engineering.... Student's signature .....
Field of study ....Computer Engineering.... Advisor's signature .....
Academic year ....2007....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้จะสำเร็จเรียบร้อยมิได้ หากปราศจากความช่วยเหลือจากอาจารย์ที่ปรึกษา ผศ.ดร. เฉลิมเอก อินทนากรวิวัฒน์ ที่คอยให้คำแนะนำ ความรู้และความคิดเห็นต่างๆอันเป็นประโยชน์ต่องานวิจัยมาโดยตลอด อีกทั้งรวมไปถึงความรู้ในด้านอื่นๆอีกมากมายที่อาจารย์ได้ถ่ายทอดให้อยู่เสมอ สิ่งเหล่านี้ล้วนเป็นสมบัติล้ำค่ายิ่งที่ช่วยผลักดันให้ผู้วิจัยสามารถมาจนถึงจุดนี้ได้และนำมาใช้ประโยชน์ได้โดยไม่มีวันหมด ขอกราบขอบพระคุณเป็นอย่างสูง กราบขอบพระคุณ ผศ.ดร.อรรถสิทธิ์ สุรฤกษ์ รศ.ดร.บุญเสริม กิจศิริกุล และ รศ. ดร. อนันต์ ผลเพิ่ม คณะกรรมการสอบวิทยานิพนธ์ที่สละเวลามาให้ข้อเสนอแนะและเติมเต็มข้อบกพร่องของวิทยานิพนธ์ฉบับนี้ให้ผ่านมาด้วยดี

ขอขอบคุณบิดา มารดา และพี่สาว รวมไปถึงทุกๆคนในครอบครัวของผู้วิจัยที่คอยสนับสนุนและให้กำลังใจมาตลอดระยะเวลา 2 ปีที่ผ่านมานี้ ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย สถานศึกษาที่ได้ประสิทธิประสาทวิชาความรู้อันเป็นประโยชน์ต่อผู้วิจัย โดยเฉพาะเหล่าคณาจารย์ในภาควิชาวิศวกรรมคอมพิวเตอร์ รวมไปถึงเพื่อนๆ พี่ๆ และน้องๆร่วมภาควิชาทุกคนที่คอยให้คำปรึกษาและช่วยเหลือยามที่ผู้วิจัยประสบปัญหาใดๆอยู่เสมอ นอกจากนี้ยังช่วยสร้างบรรยากาศให้ผู้วิจัยรู้สึกไม่เงียบเหงาและสนุกสนานไปกับการเรียน นับเป็นประสบการณ์ชีวิตและความทรงจำที่มีค่ายิ่ง ขอขอบคุณและขอขมาในสิ่งที่ล่วงเกินไว้ ณ โอกาสนี้

สารบัญ

หน้า

สารบัญภาพ

หน้า

สารบัญตาราง

หน้า

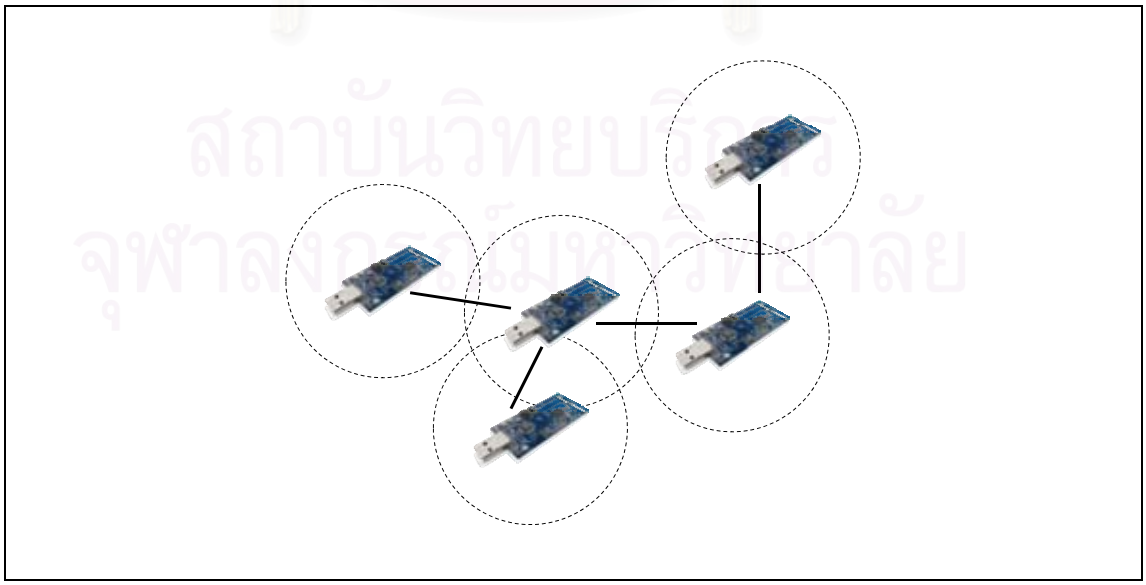
บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ระบบเครือข่ายตัวรับรู้แบบไร้สาย (Wireless sensor networks) เป็นระบบเครือข่ายชนิดหนึ่งที่ประกอบไปด้วยอุปกรณ์อิเลกทรอนิกส์ขนาดเล็กที่เรียกว่า โหมต (Mote) หลายๆตัวมาเชื่อมต่อกัน โดยที่แต่ละโหมตจะมีอุปกรณ์สำคัญที่เรียกว่า ตัวรับรู้ (Sensor) ติดอยู่ ตัวรับรู้จะมีคุณสมบัติในการตรวจจับสิ่งเร้าต่างๆภายในระยะอาณาบริเวณแวดล้อมใกล้ตัวมันแล้วอ่านค่าออกมาเป็นข้อมูล (Data) ตัวอย่างเช่น ระดับอุณหภูมิของอากาศ ระดับความเข้มของแสง ระดับความชื้นของสภาพอากาศ เป็นต้น [1] ข้อมูลเหล่านี้จะถูกเก็บลงในหน่วยความจำและถูกประมวลผลที่หน่วยประมวลผล (Microprocessor) ของโหมตเพื่อทำการวิเคราะห์หรือนำข้อมูลไปใช้ต่อไป ในรูปที่ 1.1 จะแสดงภาพตัวอย่างโครงสร้างของระบบเครือข่ายตัวรับรู้ที่ประกอบไปด้วยโหมตจำนวน 5 ตัวที่มาเชื่อมต่อกัน

เมื่อพิจารณาในแง่โครงสร้างของระบบจะสามารถมองโหมตแต่ละโหมตว่าเป็นสถานีเชื่อมโยงตัวรับรู้ (Sensor node) ซึ่งถูกนำมาวางไว้ในอาณาบริเวณใกล้เคียงกันและมีการเชื่อมต่อสื่อสารแลกเปลี่ยนข้อมูลระหว่างกันแบบไร้สาย (Wireless communication) โดยใช้สัญญาณคลื่นความถี่วิทยุเป็นตัวกลาง โหมตจะมีคุณสมบัติระยะติดต่อสื่อสารกับโหมตตัวอื่นใกล้ไกลต่างกันออกไปตามความแรงของสัญญาณวิทยุที่ปรับได้ อีกทั้งยังขึ้นกับบริษัทผู้ผลิตโหมตเองด้วย



รูปที่ 1.1 โครงสร้างระบบเครือข่ายตัวรับรู้แบบไร้สาย

อุปกรณ์โหมตถูกออกแบบมาให้มีขนาดที่เล็ก ทำให้โหมตมีข้อจำกัดในด้านแหล่งจ่ายพลังงานไฟฟ้าที่จำเป็นต้องมีขนาดเล็กตามไปด้วย ตัวอย่างแหล่งพลังงานเช่น แบตเตอรี่หรือถ่านไฟฉายขนาดเล็ก เป็นต้น ในรูปที่ 1.2 แสดงตัวอย่างของโหมตที่ใช้พลังงานจากถ่านไฟฉายในการทำงาน อายุการใช้งานของโหมตจะสั้นหรือยาวขึ้นอยู่กับขนาดของแหล่งพลังงานที่มีของแต่ละโหมต การที่จะประจุแหล่งพลังงานใหม่ (Recharge) ให้กับโหมตเมื่อพลังงานของโหมตหมดนั้นทำได้ยาก เพราะส่วนใหญ่แล้วระบบเครือข่ายตัวรับรู้แบบไร้สายมักจะนิยมถูกนำไปประยุกต์ใช้เก็บข้อมูลในพื้นที่บริเวณที่ยากแก่การเข้าถึงโดยมนุษย์ เช่น สนามรบ ป่าลึก ทะเลทราย เป็นต้น ดังนั้นการประหยัดทรัพยากรพลังงานอันมีอยู่อย่างจำกัดของโหมตจึงเป็นประเด็นปัญหาที่นักวิจัยมุ่งความสนใจเป็นอย่างมาก

รูปที่ 1.2 ตัวอย่างโหมตที่ใช้พลังงานจากถ่านไฟฉาย

การนำระบบเครือข่ายตัวรับรู้แบบไร้สายมาใช้เก็บข้อมูลในพื้นที่ที่ต้องการบางครั้งอาจจำเป็นต้องอาศัยระบบที่มีความหนาแน่นของโหมตต่อพื้นที่สูงเพื่อเพิ่มความแม่นยำของข้อมูลให้มากขึ้น เนื่องจากโหมตเป็นอุปกรณ์ที่ถูกออกแบบมาเพื่อใช้งานเฉพาะทางและมีระยะการอ่านค่าข้อมูลสภาพแวดล้อมที่จำกัดไม่ไกลนัก ดังนั้นการใช้งานส่วนใหญ่จึงนิยมวางโหมตเป็นจำนวนมากให้อยู่ในระยะที่ใกล้กันเพื่อให้ครอบคลุมพื้นที่ที่ต้องการได้ทั้งหมด เมื่อเปรียบเทียบกับราคาทุนต่อโหมตที่ไม่สูงนัก ผู้ใช้งานระบบส่วนใหญ่มักไม่ค่อยให้ความสำคัญต่อความเสียหายที่อาจเกิดขึ้นกับโหมต เช่น โหมตถูกทำลายด้วยภัยธรรมชาติ เป็นต้น ระบบจึงสามารถมีโหมตจำนวนมากในการทำงานได้ อีกทั้งโหมตยังมีคุณสมบัติที่สามารถติดต่อระหว่างกันแบบไร้สายทำให้การติดตั้งระบบจึงสามารถติดตั้งในลักษณะการกระจายแบบสุ่ม (Random deployment) ได้โดยไม่ต้องคำนึงถึงทอพอโลยีของเครือข่าย (Network topology) มากนัก เช่น อาศัยการโปรยตัวโหมตลงมาจากเครื่องบินที่บินผ่านพื้นที่ที่ต้องการเก็บข้อมูล เป็นต้น การติดตั้งแบบสุ่มอาจต้องระวังในเรื่องของพื้นที่เฝ้าสังเกต (Monitoring area) ของระบบที่ต้องครอบคลุมอาณาบริเวณพื้นที่ที่ต้องการให้ได้ทั้งหมด

เมื่อมองที่ระบบโดยรวม ปริมาณพลังงานไฟฟ้ารวมที่ระบบใช้จะเท่ากับปริมาณพลังงานที่โหมตแต่ละโหมตใช้นำมารวมกัน ซึ่งจะมากหรือน้อยนั้นขึ้นอยู่กับจำนวนโหมตใน

ระบบที่เปิดทำงาน กรณีที่ความหนาแน่นของโหมดในระบบมีสูงย่อมส่งผลให้มีการใช้พลังงานรวมมากตามไปด้วย จากการวิจัยพบว่าหากระบบใดๆสามารถลดจำนวนโหมดที่เปิดทำงานให้น้อยลงกว่าเดิมได้ (เปิดทำงานเฉพาะแค่บางโหมดเท่านั้น) ณ ช่วงเวลาหนึ่งๆ จะสามารถช่วยลดพลังงานรวมที่สิ้นเปลืองของระบบลงได้ อย่างไรก็ตามการที่จะปิดหรือหลับโหมดตัวใดๆในระบบลงไปนั้นไม่สามารถอาศัยการเลือกหลับแบบสุ่มได้ โหมดที่หลับต้องสามารถรับประกันได้ว่าเมื่อหลับไปแล้วไม่ทำให้ประสิทธิภาพการทำงานของระบบลดลงจนเกินรับได้และโหมดที่เปิดทำงานทั้งหมดต้องทำหน้าที่ทดแทนโหมดที่หลับไปได้อย่างสมบูรณ์ในระดับหนึ่ง

ภายใต้ข้อสมมติฐานที่ว่า โหมดมีพื้นที่ตรวจจับ (Sensing area) เป็นลักษณะของจาน (Sensing disk) ในระนาบ 2 มิติ และมีระยะรัศมีตรวจจับ (Sensing range) ค่าหนึ่งเท่ากันหมดทุกโหมด จากปัญหาพื้นที่ครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย [2] [3]กล่าวไว้ว่าเมื่อพิจารณากรณีที่ระบบมีความหนาแน่นสูง อาจเกิดความเป็นไปได้ที่โหมดหลายตัวจะวางตัวอยู่ในตำแหน่งที่ใกล้เคียงกันมากจนพื้นที่ตรวจจับเกิดการทับซ้อนซึ่งกันและกัน ส่งผลให้บริเวณพื้นที่ส่วนที่ทับซ้อนนั้นอยู่ภายใต้อาณาบริเวณตรวจจับของโหมดมากกว่าหนึ่งตัว หากระบบต้องการให้ทุกจุดในพื้นที่ที่ต้องการเก็บข้อมูลถูกตรวจจับโดยโหมดเพียงแค่หนึ่งโหมดก็พอเพียง การมีโหมดมากกว่าหนึ่งตัวมาตรวจจับซ้ำซ้อนในพื้นที่เดียวกันจะถูกมองว่าเป็นการสูญเสียพลังงานโดยไม่จำเป็น โหมดที่ทำหน้าที่ตรวจจับทับซ้อนกับโหมดอื่นถือว่าไม่มีความจำเป็นที่จะต้องเปิดทำงาน ณ ขณะเวลานั้นก็ได้ ระบบสามารถที่จะเลือกปิดการทำงานหรือเปลี่ยนสถานะของโหมดให้เป็นหลับ (Sleep) ลงไปก่อนชั่วคราวได้เพื่อประหยัดพลังงาน

ปัญหาพื้นที่ครอบคลุมสามารถแก้ไขได้ด้วยกระบวนการที่เรียกว่า “การควบคุมความหนาแน่น (Density controlling)” ซึ่งเป็นวิธีการในการจัดกำหนดการและลดจำนวนโหมดที่เปิดทำงานในระบบให้น้อยลงเพื่อประหยัดพลังงาน โดยพยายามค้นหาโหมดที่ไม่จำเป็นต่อการทำงานของระบบแล้วทำการหลับโหมดเหล่านั้นลงชั่วคราวเป็นระยะๆ ระบบที่เหลือยังคงรักษาคุณสมบัติการทำงานเอาไว้ได้ในสภาพที่เปลี่ยนแปลงไปจากเดิมไม่มากเกินกว่าเกณฑ์ที่ยอมรับได้ เมื่อใดโหมดที่ตื่นมาทำงานไปก่อนหน้านี้เหลือพลังงานน้อย กระบวนการจัดกำหนดการตื่น-หลับของโหมดในระบบจะเลือกโหมดตัวที่หลับซึ่งมีพลังงานเหลือมากกว่าขึ้นมาทำงานแทนที่ กระบวนการนี้จะสามารถช่วยประหยัดพลังงานรวมของระบบไปได้เพราะอาศัยโหมดที่น้อยตัวลงในการทำงาน อีกทั้งยังยืดอายุการทำงานของระบบ (Network lifetime) ให้นานขึ้นกว่าเดิมได้ เนื่องจากมีการสลับผลัดเปลี่ยนโหมดขึ้นมาทำงานแทนที่กันอยู่ตลอดเวลา นอกจากนี้ยังช่วยลดปริมาณการใช้ช่องสัญญาณวิทยุ (Traffic intensity) ในพื้นที่ลงได้ด้วย

จากหลักการของอัลกอริทึมควบคุมความหนาแน่นและปัญหาพื้นที่ครอบคลุม การตรวจสอบว่าโหมดใดจำเป็นหรือไม่ในการทำงานของระบบจะอาศัยพิจารณาการทับซ้อนพื้นที่ตรวจจับของโหมดในระบบ โหมดที่มีพื้นที่ตรวจจับทับซ้อนกับโหมดอื่นจะถูกตัดสินว่าเป็นโหมดซ้ำซ้อน (Redundant mote) อย่างไรก็ตามเนื่องจากในระบบมีโหมดซ้อนทับกันอยู่เป็นจำนวน

มาก ทำให้จำนวนโหมดซ้ำซ้อนทั้งหมดอาจมีได้มากกว่าหนึ่งโหมด โหมดซ้ำซ้อนเหล่านี้บางกลุ่มอาจมีอาณาบริเวณพื้นที่ตรวจจับทับซ้อนกันและกัน การตัดสินใจปิดหรือหลับโหมดซ้ำซ้อนเหล่านี้พร้อมๆกันในช่วงเวลาเดียวกันอาจทำให้มีโอกาสที่พื้นที่บอด (Blind area) ซึ่งเป็นพื้นที่ที่ไม่อยู่ภายใต้อาณาบริเวณพื้นที่ตรวจจับของโหมดๆใดเลยเกิดขึ้นมาได้ ระบบจึงต้องมีกระบวนการบางอย่างในการจัดกำหนดการตื่น-หลับโหมดซ้ำซ้อน (Active mote scheduling) เหล่านี้เพื่อป้องกันไม่ให้เกิดพื้นที่บอดขึ้นอันเป็นสาเหตุของการสูญเสียคุณสมบัติครอบคลุม (Coverage property) ของระบบ การจัดกำหนดการนี้ถือได้ว่าเป็นสิ่งสำคัญมากในการควบคุมความหนาแน่น เพราะถึงแม้ว่าระบบจะสามารถลดปริมาณการใช้พลังงานลงได้จากการหลับโหมดซ้ำซ้อน แต่ระบบต้องแลกด้วยการสูญเสียคุณสมบัติครอบคลุมไป อาจก่อให้เกิดผลเสียร้ายแรงมากกว่าผลดีก็เป็นได้

งานวิจัยก่อนหน้านี้ส่วนใหญ่จะพยายามนิยามกฎความซ้ำซ้อนของโหมดออกมาเพื่อใช้ในการตรวจสอบสถานะความซ้ำซ้อนของโหมด อีกทั้งยังนำเสนอกระบวนการจัดกำหนดการตื่น-หลับโหมดซ้ำซ้อนเพื่อรักษาคุณสมบัติพื้นที่ครอบคลุมของระบบเดิมและหลีกเลี่ยงการเกิดพื้นที่บอดขึ้น โดยอาศัยรูปแบบลักษณะการตัดสินใจเลือกตื่น-หลับโหมดจากการตั้งค่าตัวจับเวลาแบบสุ่ม (Randomized timer) ซึ่งโหมดใดที่สุ่มเวลาออกมาแล้วสั้นที่สุดก็จะได้รับโอกาสให้หลับก่อน อย่างไรก็ตามถึงแม้ว่ารูปแบบการเลือกโหมดโดยวิธีสุ่มนี้อาจทำให้มีการกระจายโอกาสให้ทั่วทุกโหมดจริง แต่ปัจจัยอย่างหนึ่งที่ไม่ได้ถูกนำมาพิจารณาในการเลือกด้วยก็คือปัจจัยด้านพลังงาน ทำให้โหมดที่มีพลังงานเหลือใช้งานอยู่น้อยสามารถถูกเลือกขึ้นมาทำงานก่อนแทนที่ควรจะเป็นหน้าที่ของโหมดที่มีพลังงานเหลืออยู่มากกว่าเพื่อยืดระยะเวลาการทำงานของระบบให้ได้นานขึ้น ดังนั้นผลการทำงานที่ได้จากการใช้ตัวจับเวลาแบบสุ่มนี้อาจทำให้การยืดอายุการทำงานของระบบไม่ยาวนานได้เท่าที่ควรจะเป็น

1.2 วัตถุประสงค์ของการวิจัย

นำเสนออัลกอริทึมการควบคุมความหนาแน่นแบบกระจายและเฉพาะที่ (Distributed and localized algorithm) สำหรับระบบเครือข่ายตัวรับรู้แบบไร้สายขนาดกลางและมีช่วงเวลาในการหลับเพื่อประหยัดพลังงานนานมากกว่า 15 นาที โดยอาศัยการถ่วงน้ำหนักด้านปัจจัยพลังงานของโหมดในการพิจารณาเลือกตื่น-หลับโหมดซ้ำซ้อน เพื่อช่วยลดจำนวนโหมดที่จำเป็นต้องเปิดทำงานให้น้อยลงและสามารถประหยัดปริมาณการใช้พลังงานรวมของระบบลงโดยที่ยังคงรักษาคุณสมบัติครอบคลุมของระบบเดิมเอาไว้ได้

1.3 ขอบเขตของการวิจัย

1) อัลกอริทึมนี้ใช้ได้กับระบบที่โหมดทุกตัวมีลักษณะของพื้นที่ตรวจจับเป็นจานและอยู่ในระนาบ 2 มิติเท่านั้น

2) อัลกอริทึมที่นำเสนอมีลักษณะการทำงานแบบกระจายและเฉพาะที่ ดังนั้นโหมตแต่ละโหมตจะอาศัยพื้นฐานความรู้จากแหล่งข้อมูลของโหมตเพื่อนบ้านที่อยู่ใกล้เคียง (Local neighbors) ในการคำนวณ

3) อัลกอริทึมที่นำเสนอจะมี 2 ส่วน ส่วนแรกเป็นการตรวจสอบความซ้ำซ้อนของโหมต ซึ่งนำหลักการของงานวิจัย Tian D. และ Georganas N.D. [4] กับงานของ Jiang J. และ Dou W. [5] เข้ามาประยุกต์ใช้ ส่วนที่สองเป็นการออกแบบและนำเสนอกระบวนการจัดกำหนดการตื่น-หลับของโหมตใหม่ โดยอาศัยวิธีการแก้ปัญหาด้วยอัลกอริทึมเชิงละโมบแบบถ่วงน้ำหนักพลังงาน (Energy-weighted greedy algorithm) สำหรับปัญหาเซตครอบคลุม (Set cover problem) ในการคำนวณคำตอบ

4) ในการทดลองจะทำการเปรียบเทียบอัลกอริทึมที่นำเสนอกับอัลกอริทึมที่มีการรักษาคุณสมบัติครอบคลุมด้วยเหมือนกันเท่านั้น

5) อัลกอริทึมจะถูกทดสอบประสิทธิภาพบนโปรแกรมจำลองทอสซิม (TOSSIM) [6] ซึ่งโปรแกรมที่พัฒนาขึ้นสามารถนำไปใช้งานในระบบจริงได้ (Practical use)

6) อัลกอริทึมนี้จะทดสอบการทำงานบนโปรแกรมจำลองทอสซิมในสภาวะแวดล้อมที่ไม่มีการสูญเสียข้อมูล (Packet loss) ในการรับส่งข้อมูลระหว่างโหมต

7) ระบบที่ทดสอบมีความหนาแน่นของโหมตไม่เกิน 0.03 โหมตต่อตารางหน่วย (3 โหมตต่อ 100 ตารางหน่วยในโปรแกรมจำลอง)

8) ระบบที่ทดสอบมีช่วงเวลาในการหลับเพื่อประหยัดพลังงานนานมากกว่า 15 นาที

9) โหมตทุกโหมตจะไม่มีการเคลื่อนย้ายตำแหน่งตลอดระยะการทำงานของระบบ

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1) ศึกษาความหมายและการทำงานของระบบเครือข่ายตัวรับรู้แบบไร้สายและข้อจำกัดของโหมต

2) ศึกษาปัญหาการประหยัดพลังงานและปัญหาพื้นที่ครอบคลุมสำหรับระบบเครือข่ายตัวรับรู้

3) ศึกษานิยามและทฤษฎีการทับซ้อนพื้นที่ครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย

4) ศึกษาวิธีการและอัลกอริทึมในการควบคุมความหนาแน่นของโหมตในระบบเครือข่ายตัวรับรู้แบบไร้สาย

5) ศึกษาปัญหาเซตครอบคลุม

6) ออกแบบอัลกอริทึมควบคุมความหนาแน่นของโหมดในระบบเครือข่ายตัวรับรู้แบบไร้สาย โดยปรับปรุงแก้ไขจากแนวทางเดิม

7) ทดสอบการทำงานของอัลกอริทึมที่นำเสนอบนโปรแกรมจำลองทอสซิม

8) บันทึกและวิเคราะห์ผลการทดลอง

9) ปรับปรุงแก้ไขหากพบข้อผิดพลาด

10) สรุปผลและเรียบเรียงเป็นวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับจากการวิจัย

อัลกอริทึมควบคุมความหนาแน่นบนระบบเครือข่ายตัวรับรู้แบบไร้สายที่นำเสนอสามารถช่วยลดปริมาณการใช้พลังงานรวมของระบบลงโดยที่อาศัยแค่โหมดบางกลุ่มในการเปิดทำงานในขณะที่สามารถรักษาคุณสมบัติครอบคลุมของระบบเดิมเอาไว้ได้ เนื่องด้วยการทำงานของอัลกอริทึมจะเป็นแบบกระจายและเฉพาะที่ ทำให้สามารถลดความจำเป็นในการเก็บข้อมูลแบบครอบคลุม (Global information) ได้ อีกทั้งอัลกอริทึมนี้ถูกทดสอบใช้งานบนโปรแกรมจำลองซึ่งจำลองการทำงานของระบบปฏิบัติการที่ใช้งานบนตัวอุปกรณ์โหมดจริง ดังนั้นอัลกอริทึมนี้จึงสามารถนำไปประยุกต์ใช้งานได้ในระบบและสภาพแวดล้อมจริง

1.6 ผลงานตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของงานวิทยานิพนธ์ได้รับการตีพิมพ์เป็นบทความวิชาการในหัวเรื่อง “A Set Cover-Based Density Control Algorithm for Sensing Coverage Problems in Wireless Sensor Networks” โดย ศรัณย์ เจนจตุรงค์ และ เฉลิมเอก อินทนากรวิวัฒน์ ในงานประชุมวิชาการ “The Third International Conference on Cognitive Radio Oriented Wireless Networks and Communications” ซึ่งจัดขึ้น ณ โรงแรม Holiday Inn Atrium Singapore เมืองสิงคโปร์ ประเทศสิงคโปร์ ระหว่างวันที่ 15-17 พฤษภาคม 2551 ดังภาคผนวก ค

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 อัลกอริทึมควบคุมความหนาแน่น (Density control algorithm)

โดยปกติแล้วในระบบเครือข่ายตัวรับรู้แบบไร้สาย โหมตแต่ละตัวจะทำหน้าที่ในการตรวจจับสิ่งเร้าภายในพื้นที่ตรวจจับ (Sensing area) รอบ ๆตัวมัน ในรูปที่ 2.1 จะแสดงแบบจำลองโหมตที่มีวงกลมแทนพื้นที่ตรวจจับในระนาบ 2 มิติ โดยมีจุดตรงกลางแทนตำแหน่งของโหมต โดยปกติการทำงานของโหมตจะมีสถานะอยู่ด้วยกัน 4 สถานะ ได้แก่ สถานะส่งข้อมูล (Transmit) สถานะรับข้อมูล (Receive) สถานะเดินเครื่องเปล่า (Idle) และสถานะหลับ (Sleep) สถานะเดินเครื่องเปล่าจะเป็นสถานะที่ตัวโหมตไม่มีการรับและไม่มีการส่งข้อมูลใด ๆกับโหมตอื่น ส่วนสถานะหลับจะเป็นสถานะที่โหมตปิดตัวรับส่งสัญญาณวิทยุ จากการทดสอบวัดปริมาณพลังงานที่ใช้ของโหมต ณ สถานะต่าง ๆ [7] พบว่า พลังงานที่ใช้เมื่อโหมตอยู่ในสถานะรับ ส่ง หรือเดินเครื่องเปล่ามีการใช้พลังงานในปริมาณที่ใกล้เคียงกัน ในขณะที่สถานะหลับจะมีค่าการใช้ปริมาณน้อยกว่าสถานะอื่นถึงเกือบ 10 เท่า เพราะฉะนั้นตีความได้ว่าพลังงานไฟฟ้าส่วนใหญ่ที่โหมตใช้จะหมดไปกับระยะเวลาการเปิดเครื่องขึ้นทำงานของโหมตไม่ว่าจะมีการส่งรับข้อมูลหรือไม่

รูปที่ 2.1 แบบจำลองตำแหน่งและพื้นที่ตรวจจับของโหมต

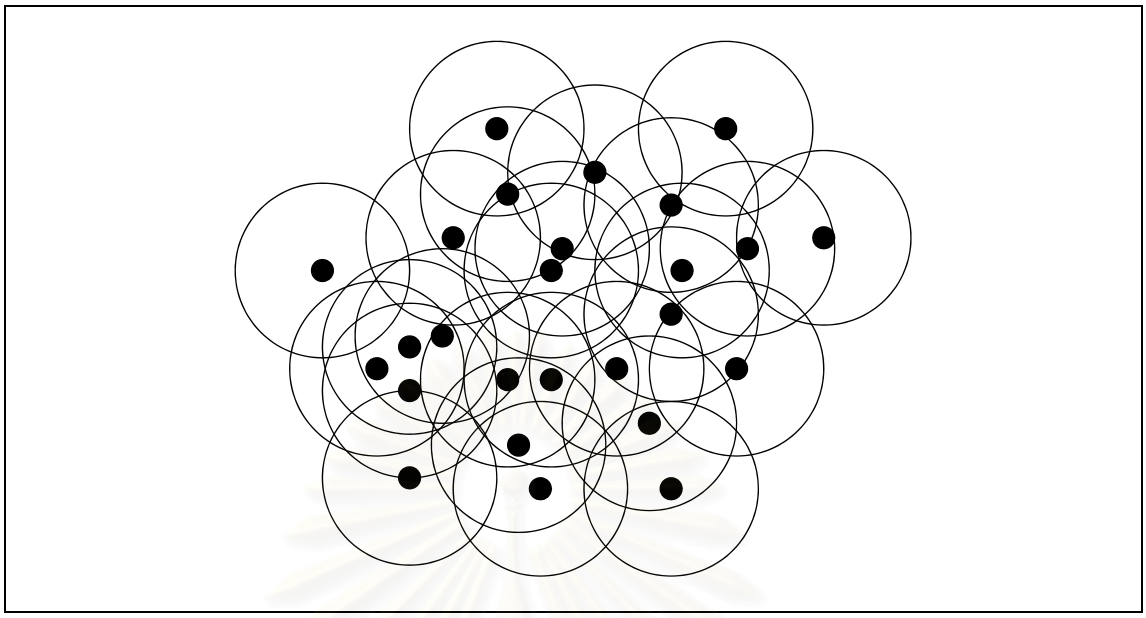
ในระบบเครือข่ายตัวรับรู้หนึ่ง ๆจะประกอบไปด้วยโหมตจำนวนมากมาทำงานอยู่ภายในอาณาบริเวณที่ใกล้เคียงกัน อุปกรณ์โหมตมักจะถูกวางและติดตั้งในวิธีการกระจายแบบ

สุ่มเป็นส่วนใหญ่ ทำให้ผังโครงสร้างการเรียงตัวของโหมดมีลักษณะที่ไม่แน่นอนและยากแก่การควบคุมพื้นที่ตรวจจับของโหมดให้กระจายอย่างสม่ำเสมอทั่วอาณาบริเวณที่ต้องการ ปัญหาครอบคลุมการตรวจจับ (Sensor coverage problem) [8] เป็นปัญหาที่มุ่งเน้นพิจารณาหาคำตอบที่ว่า “ตัวรับรู้ที่วางอยู่ในระบบทั้งหมดนั้น มีประสิทธิภาพในการเฝ้าดูและตรวจจับสภาพแวดล้อมภายในพื้นที่ปริภูมิที่ต้องการดีมากน้อยแค่ไหน” ปัญหานี้เป็นการวัดคุณภาพการบริการตรวจจับ (Quality of sensing service) ของระบบออกมาในรูปแบบของฟังก์ชันของการตรวจจับ (Sensing function) [2] ซึ่งประสิทธิภาพการตรวจจับนี้ถือเป็นมาตรวัดสำคัญที่ผู้ใช้สามารถนำไปบ่งชี้ความดีในการตรวจจับพื้นที่ของระบบได้

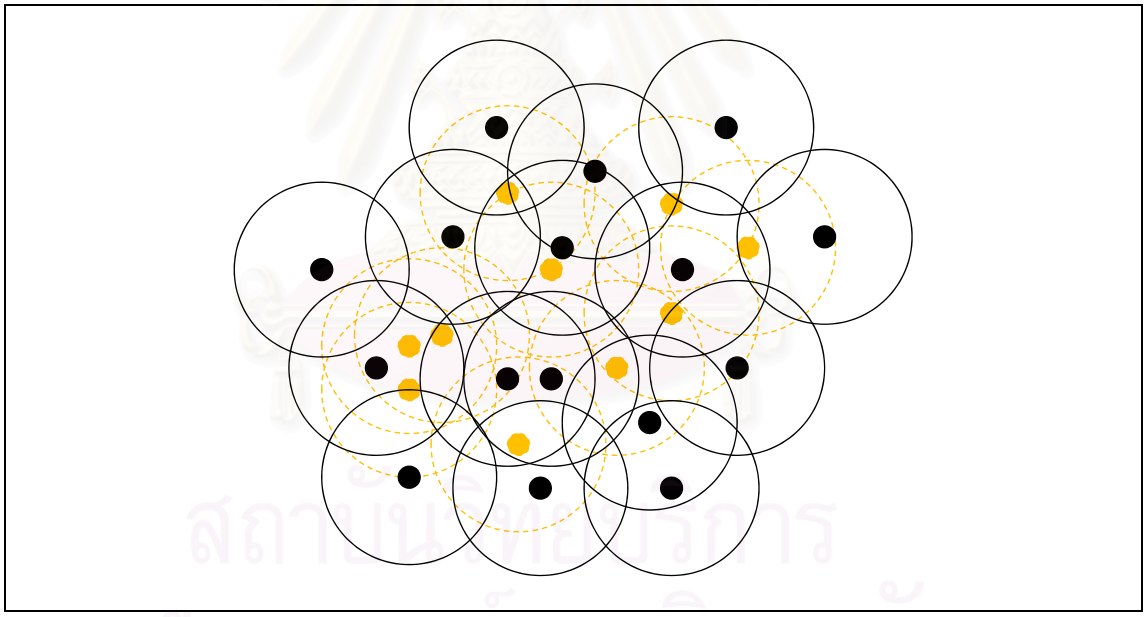
สำหรับระบบเครือข่ายตัวรับรู้หนึ่งๆจะมีคุณภาพการตรวจจับดีมากน้อยเพียงใดจะขึ้นอยู่กับลักษณะการกระจายผังโครงสร้างของโหมดที่ทำงาน หากโหมดมีการกระจายกันทำงานอย่างสม่ำเสมอเท่าๆกันเต็มพื้นที่ย่อมทำให้คุณภาพการตรวจจับของระบบนั้นมีค่าสูง อย่างไรก็ตามเนื่องจากโหมดทุกตัวถูกติดตั้งแบบสุ่มกระจาย ทำให้บางบริเวณเกิดการทับซ้อนกันของพื้นที่ตรวจจับระหว่างโหมดจำนวนมาก ซึ่งหากมองให้แง่ของความจำเป็นและการใช้พลังงานแล้ว ผลของการที่มีโหมดทำงานซ้ำซ้อนในบริเวณใกล้ๆกันจะเป็นการสิ้นเปลืองพลังงานของระบบโดยไม่จำเป็น ซึ่งเกิดขึ้นได้บ่อยสำหรับระบบเครือข่ายตัวรับรู้ที่มีความหนาแน่นของโหมดในปริมาณสูง

อัลกอริทึมควบคุมความหนาแน่น (Density control algorithm) เป็นอัลกอริทึมที่พยายามแก้ปัญหาการใช้พลังงานสิ้นเปลืองของระบบเครือข่ายตัวรับรู้อันเนื่องมาจากการทำงานตรวจจับทับซ้อนพื้นที่เดียวกันระหว่างโหมดมากกว่าหนึ่งตัว อัลกอริทึมควบคุมความหนาแน่นอาศัยกลไกในการจัดการสถานะของโหมดซ้ำซ้อนเหล่านั้นให้มีการตื่นและหลับสลับกันขึ้นมาทำงานแทน ซึ่งเมื่อพิจารณา ณ ช่วงรอบเวลาหนึ่งๆ จากเดิมโหมดในระบบทั้งหมดต้องเปิดทำงานพร้อมกัน จะเหลือเพียงแค่กลุ่มของโหมดที่จำเป็นบางตัวเท่านั้นที่ตื่นมาทำงาน ส่วนโหมดที่เหลือจะถูกทำให้หลับไปชั่วคราว กลุ่มของโหมดที่ตื่นและหลับในแต่ละรอบเวลาจะถูกเลือกให้สลับหน้าที่กันไปในทุกๆโหมด การที่มีโหมดเพียงกลุ่มหนึ่งจากทั้งหมดที่ตื่นมาทำงานทำให้สามารถประหยัดการใช้พลังงานรวมของระบบลงได้แน่นอน

นิยามของโหมดซ้ำซ้อนในที่นี้ ได้แก่ โหมดที่มีพื้นที่ตรวจจับทับซ้อนกับพื้นที่ของโหมดเพื่อนบ้าน ในรูปที่ 2.2 และ 2.3 แสดงภาพโครงสร้างของระบบเครือข่ายตัวรับรู้แบบไร้สายก่อนและหลังการทำงานของอัลกอริทึมควบคุมความหนาแน่น ณ ขณะช่วงเวลาหนึ่งตามลำดับ โหมดซ้ำซ้อนที่ถูกเลือกให้หลับจะแสดงด้วยวงกลมเส้นประ ส่วนโหมดที่อยู่ในสถานะเปิดทำงานจะแสดงด้วยวงกลมเส้นทึบ จะเห็นว่าหลังการทำงานของอัลกอริทึมระบบจะมีจำนวนที่เปิดทำงานน้อยลง เนื่องจากมีโหมดซ้ำซ้อนบางตัวถูกเปลี่ยนสถานะให้หลับไป



รูปที่ 2.2 โครงสร้างระบบก่อนการทำงานของอัลกอริทึมควบคุมความหนาแน่น



รูปที่ 2.3 โครงสร้างระบบหลังการทำงานของอัลกอริทึมควบคุมความหนาแน่น

2.1.2 ปัญหาเซตครอบคลุม (Set cover problem)

ปัญหาเซตครอบคลุม [9] เป็นหนึ่งในปัญหาที่สำคัญทางด้านวิทยาศาสตร์คอมพิวเตอร์ (Computer science) และทฤษฎีความซับซ้อน (Complexity theory) โดยหลักการของปัญหาจะประกอบไปด้วยชุดข้อมูลนำเข้า (Input) ได้แก่ หมู่ของเซต (Collection of sets)

จำนวนหนึ่งที่สมาชิกภายในเซตจะเลือกมาจากข้อมูลที่อยู่ในเซตเอกภพ (Universe) เดียวกัน โดยที่เซตใด ๆอาจจะมีสมาชิกภายในเซตซ้ำหรือไม่ซ้ำกันกับของเซตอื่นก็ได้ ในรูปที่ 2.3 แสดงตัวอย่างของชุดข้อมูลนำเข้าซึ่งประกอบไปด้วยเซตจำนวน 5 เซต ได้แก่ เซต A B C D และ E โดยที่มีข้อมูลสมาชิกในเซตเอกภพทั้งสิ้น 7 ตัว ได้แก่ a b c d e f และ g ข้อมูลเหล่านี้สามารถอยู่ในสมาชิกของเซตได้มากกว่าหนึ่งเซต

$$Universe = \{a,b,c,d,e,f,g\}$$
$$A = \{a,d,g\}$$
$$B = \{a,b,d,e,f\}$$
$$C = \{b,c,g\}$$
$$D = \{d,e,f,g\}$$
$$E = \{e,f,g\}$$

รูปที่ 2.4 ชุดข้อมูลนำเข้าสำหรับปัญหาเซตครอบคลุม

การแก้ปัญหาเซตครอบคลุมจะเป็นการเลือกเซตจำนวนหนึ่งจากเซตข้อมูลนำเข้าทั้งหมด โดยพยายามเลือกออกมาจำนวนน้อยเซตที่สุดเท่าที่จะเป็นไปได้ที่เมื่อยูเนียน (Union) รวมสมาชิกข้อมูลจากทุกเซตที่เลือกแล้วได้เซตผลลัพธ์เทียบเท่ากับเซตเอกภพ เซตที่เป็นคำตอบนี้จะถูกนิยามว่าเป็นเซตครอบคลุม (Set cover) ตัวอย่างเช่น ชุดข้อมูลนำเข้าในรูปที่ 2.4 เซตครอบคลุมที่เป็นคำตอบอาจเป็นเซต B และ C ซึ่งหากยูเนียน 2 เซตนี้เข้าด้วยกันก็จะได้เซตผลลัพธ์เท่ากับเซตเอกภพคือ $\{a,b,c,d,e,f,g\}$ พอดี อย่างไรก็ตามหลักการเลือกเซตมีได้หลากหลายวิธีซึ่งอาจทำให้เซตที่เลือกได้จากคนละวิธีนั้นออกมาไม่ตรงกันได้

ปัญหาเซตครอบคลุมจะแบ่งออกเป็น 2 แบบคือ แบบปัญหาการตัดสินใจ (Set cover decision problem) ที่เป็นการตอบคำถามว่า “จากชุดข้อมูลนำเข้าที่มีและค่าคงที่ k ค่าหนึ่ง สามารถหาหมู่ย่อยของเซตที่เป็นคำตอบของปัญหาเซตครอบคลุมและมีขนาดของหมู่ย่อยเท่ากับหรือน้อยกว่า k ได้หรือไม่?” ซึ่งปัญหาแบบการตัดสินใจนี้จะเป็นหนึ่งในปัญหาเอ็นพีบริบูรณ์ (NP-Complete) ปัญหาเซตครอบคลุมอีกแบบหนึ่งได้แก่ แบบปัญหาการหาค่าเหมาะที่สุด (Set cover optimization problem) ซึ่งเป็นการหาหมู่ย่อยของเซตที่เป็นคำตอบของปัญหาเซตครอบคลุมและมีขนาดของหมู่ย่อยน้อยที่สุดเท่าที่จะเป็นไปได้ด้วย ปัญหาเซตครอบคลุมแบบนี้ถือว่าเป็นปัญหาเอ็นพีแบบยาก (NP-Hard)

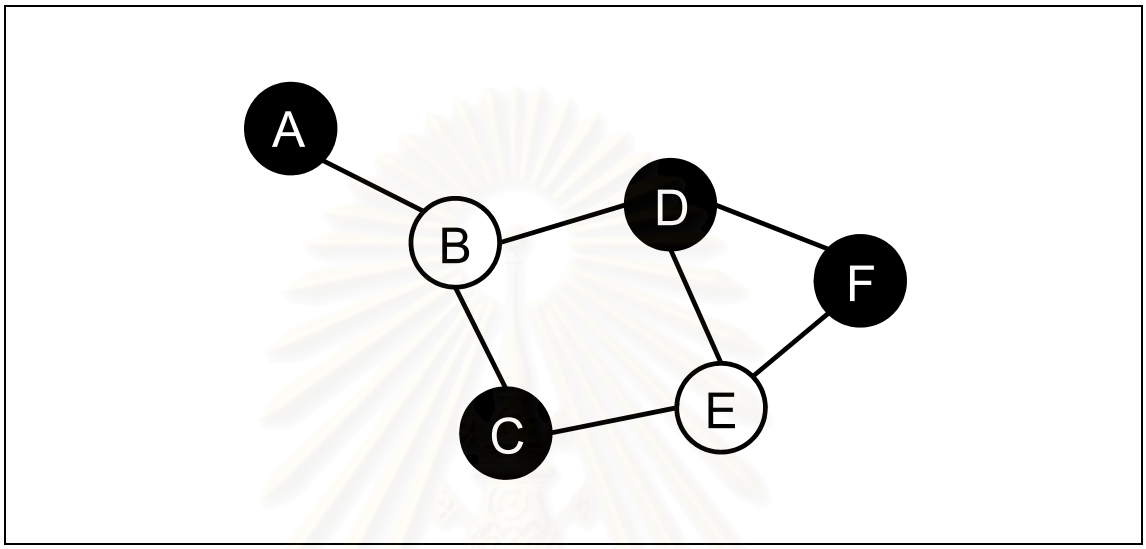
2.1.3 อัลกอริทึมแบบประมาณ (Approximation algorithm)

อัลกอริทึมแบบประมาณ [9] เป็นวิธีการหนึ่งที่ใช้สำหรับแก้ไขปัญหาการหาค่าเหมาะที่สุดประเภทเอ็นพีแบบยาก เนื่องจากเป็นที่เชื่อกันว่าไม่มีอัลกอริทึมใดที่มีประสิทธิภาพ (ทำงานได้รวดเร็ว) ที่สามารถแก้ไขปัญหาเอ็นพีแบบยากได้คำตอบที่เที่ยงตรง จึงได้เกิดความพยายามที่จะหาคำตอบของปัญหาประเภทนี้โดยที่ยอมให้คำตอบของปัญหาอาจไม่ใช่คำตอบที่ถูกต้องที่สุด แต่สามารถหาคำตอบนั้นได้ในเวลาโพลิโนเมียล (Polynomial time) อัลกอริทึมประเภทนี้เรียกว่า อัลกอริทึมแบบประมาณ ซึ่งจะมีข้อแตกต่างกับอัลกอริทึมแบบฮิวริสติก (Heuristic) (ซึ่งมักเป็นการหาคำตอบที่ดีในระดับหนึ่งโดยใช้เวลาไม่นานมากนัก) พอสมควรกล่าวคืออัลกอริทึมแบบประมาณนั้นต้องการหาคำตอบที่สามารถพิสูจน์ได้ว่าดีเพียงใดและมีขอบเขตการใช้เวลาในการคำนวณนานไม่เกินขอบเขตเท่าใด ซึ่งอัลกอริทึมในอุดมคติจะต้องได้คำตอบผิดไปจากคำตอบจริงไม่เกินค่าคงที่ค่าหนึ่ง (เช่น คลาดเคลื่อนไม่เกินร้อยละ 5 เป็นต้น)

ปัญหาเอ็นพีแบบยากมีความหลากหลายอย่างมากในแง่ของการประมาณค่า บางปัญหาสามารถประมาณได้เป็นอัตราส่วนขนาดหนึ่ง (อัลกอริทึมสำหรับประมาณปัญหาเหล่านี้มักเรียกกันว่า แบบแผนการประมาณในเวลาโพลิโนเมียล (Polynomial time approximation scheme หรือ PTAS) ส่วนบางปัญหานั้นก็ไม่สามารถที่จะทำการประมาณได้เลยก็มีเช่นกัน ตัวอย่างของอัลกอริทึมแบบประมาณที่มักกล่าวถึงกัน ได้แก่ อัลกอริทึมสำหรับการคลุมจุดยอดในกราฟ (Vertex cover problem) โจทย์คือมีข้อมูลนำเข้าเป็นกราฟที่ประกอบไปด้วยจุดยอด (Vertex) จำนวนหนึ่ง จุดยอดบางจุดจะมีเส้นเชื่อม (Edge) ระหว่างกัน อัลกอริทึมจะทำการเลือกเซตของจุดยอดจำนวนหนึ่งที่น้อยจุดที่สุดที่ทำให้ทุกๆเส้นเชื่อมในกราฟมีจุดยอดที่ปลายอย่างน้อยข้างหนึ่งถูกเลือก อัลกอริทึมสำหรับประมาณปัญหานี้คือ พยายามหาเส้นเชื่อมที่ยังไม่มีจุดยอดที่ปลายข้างใดถูกเลือก แล้วทำการเลือกจุดยอดที่ปลายทั้งคู่ของด้านนี้ อัลกอริทึมนี้ให้ผลลัพธ์ที่มีขนาดไม่เกิน 2 เท่าของคำตอบที่ดีที่สุด ตัวอย่างในรูปที่ 2.5 แสดงผลลัพธ์ที่ได้จากอัลกอริทึมแบบประมาณในการแก้ปัญหาการครอบคลุมจุดยอด จุดยอดที่มีสีดำจะแทนจุดยอดที่ถูกเลือกและเป็นส่วนหนึ่งของคำตอบ ส่วนจุดยอดที่มีสีขาวจะแทนจุดยอดที่ไม่ได้ถูกเลือก จะได้ว่าขนาดของเซตคำตอบจุดยอดที่ถูกเลือก (Vertex cover) จะเท่ากับ 4 คือ $\{A, C, D, F\}$ ซึ่งจะเห็นได้ว่า จุดยอดที่ปลายข้างใดข้างหนึ่งของทุกเส้นเชื่อมจะเป็นสมาชิกในเซตที่ถูกเลือกเสมอ อย่างไรก็ตามคำตอบที่ได้นี้จะยังไม่ตรงกับคำตอบที่เหมาะสมที่สุดที่มีขนาดของเซตคำตอบเท่ากับ 3 ได้แก่ $\{B, D, E\}$ และ $\{B, E, F\}$

อัลกอริทึมแบบประมาณบางอัลกอริทึมอาจจะไม่เหมาะสมกับงานในทางปฏิบัติ ตัวอย่างเช่น คนส่วนใหญ่มักรู้สึกไม่ค่อยประทับใจกับอัลกอริทึมที่ช่วยให้พวกเขาจ่ายเงินไม่เกิน 20 เท่าของค่าใช้จ่ายที่ถูกที่สุด และเช่นเดียวกันบางอัลกอริทึมอาจมีขอบเขตเวลาในการทำงานที่ไม่ดีนัก (แม้ว่าจะเป็นเวลาโพลิโนเมียลก็ตาม) เช่น $O(n^{2000})$ เป็นต้น สรุปได้ว่าถึงแม้อัลกอริทึมแบบประมาณจะสามารถประมาณคำตอบจริงของปัญหาได้ แต่ขอบเขตของคำตอบ

และเวลาในการคำนวณที่ได้อาจอยู่ในช่วงเกินกว่าเกณฑ์ที่รับได้ก็เป็นได้ ข้อจำกัดอีกอย่างหนึ่งของอัลกอริทึมแบบประมาณคือ ใช้ได้กับปัญหาการหาค่าเหมาะที่สุด (Optimization problem) เท่านั้น ไม่สามารถใช้กับปัญหาการตัดสินใจ“แท้ๆ”ได้ เช่น ปัญหาความสอดคล้องแบบบูล (Boolean satisfiability problem) เป็นต้น



รูปที่ 2.5 ผลลัพธ์การทำงานของอัลกอริทึมประมาณสำหรับการคลุมจุดยอดในกราฟ

2.1.4 อัลกอริทึมเชิงละโมบ (Greedy algorithm)

อัลกอริทึมเชิงละโมบเป็นหนึ่งในอัลกอริทึมแบบประมาณที่นิยมใช้ในการแก้ปัญหาการหาค่าเหมาะที่สุด โดยที่แต่ละขั้นตอนการค้นหาคำตอบที่ได้ออกมานั้นจะเป็นในลักษณะของการหาคำตอบที่ดีที่สุดเฉพาะที่ (Local optimum) ภายใต้การคาดหวังว่า เมื่อหาคำตอบที่ดีที่สุดเฉพาะที่ไปได้เรื่อยๆแล้วสุดท้ายคำตอบที่ได้ก็จะเป็นคำตอบที่ดีที่สุดของปัญหา (Global optimum) ที่ต้องการ ตัวอย่างการนำอัลกอริทึมเชิงละโมบไปใช้แก้ปัญหา ได้แก่ การแก้ปัญหาการจัดเส้นทางเดินรถของพนักงานขาย (Traveling Salesman Problem) ปัญหานี้จะเป็นการหาเส้นทางที่สั้นที่สุดที่สามารถเดินทางไปยังทุกๆจุดในสถานที่ต่างๆได้ครบทุกจุด โดยที่แต่ละจุดจะเดินทางผ่านได้แค่ครั้งเดียวเท่านั้น จากปัญหาจะได้ว่า ฟังก์ชันที่ใช้ในการเลือกเส้นทางจะเป็นฟังก์ชันของระยะทาง ซึ่งเส้นทางคำตอบที่ต้องการจะเป็นเส้นทางที่มีค่าฟังก์ชันหรือระยะทางการเดินที่น้อยที่สุด ตามหลักการเชิงละโมบของอัลกอริทึมจะพยายามหาคำตอบที่ดีที่สุดเฉพาะที่ก่อน เพราะฉะนั้นเมื่อพิจารณาที่จุดใดๆในเส้นทาง จุดต่อไปที่จะถูกเลือกเดินไปจะต้องเป็นจุดที่ยังไม่เคยเดินผ่านและอยู่ใกล้จุดปัจจุบันที่ยืนอยู่มากที่สุดก่อนเสมอ โดยคาดหวังว่าสุดท้ายแล้วเมื่อนับรวมระยะทางทั้งหมดที่เดินจะได้ระยะทางที่สั้นที่สุดและสามารถวนครบรอบได้ทุกจุดพอดี

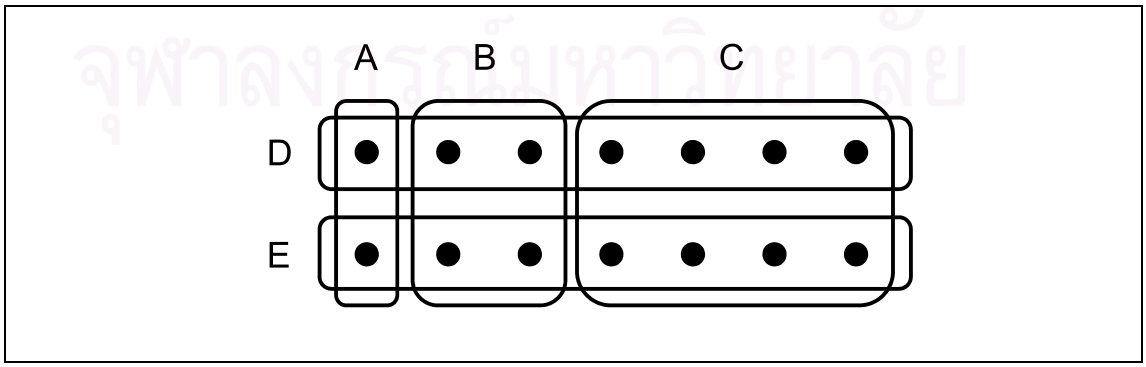
อย่างไรก็ตาม คำตอบที่ได้จากอัลกอริทึมเชิงละโมบอาจจะไม่ใช่คำตอบที่ดีเสมอไป เพราะหลักการของอัลกอริทึมเชิงละโมบจะยึดคำตอบที่ดีที่สุดเฉพาะที่ไว้ก่อนเสมอ ซึ่งในความเป็นจริงแล้วคำตอบที่ดีที่สุดของปัญหาอาจไม่ได้เกิดจากผลรวมของคำตอบที่ดีที่สุดเฉพาะที่ทั้งหมดเสมอไป ดังเช่นตัวอย่างปัญหาการจัดเส้นทางเดินรถของพนักงานขายที่ได้กล่าวไปนั้น อาจจะมีการเลือกเดินไปยังจุดที่ไกลกว่าออกไปบ้างในบางจุดเพื่อให้จุดถัด ๆไปสามารถเดินด้วยระยะทางที่สั้นลง ดังนั้นอัลกอริทึมเชิงละโมบจึงใช้ในการแก้หาคำตอบค่าเหมาะที่สุดได้ดีแต่อาจได้คำตอบที่ไม่ดีที่สุด

2.1.5 อัลกอริทึมเชิงละโมบสำหรับประมาณคำตอบปัญหาเซตครอบคลุม

อัลกอริทึมเชิงละโมบสำหรับใช้แก้ปัญหาเซตครอบคลุมจะมีหลักการเลือกหมู่ย่อยของเซตคำตอบตามกฎคือ พยายามเลือกเซตที่มีสมาชิกภายในเซตอยู่ในยูนิเวอร์สและยังไม่ถูกครอบคลุมจำนวนมากที่สุดก่อนเสมอ ซึ่งการใช้อัลกอริทึมเชิงละโมบวิธีนี้สามารถพิสูจน์ออกมาได้ว่า อัตราส่วนการประมาณ (Approximation ratio) มีค่าเป็น $H(s)$ เมื่อ s แทนขนาดของเซตในหมู่ของเซตนำเข้าที่ใหญ่สุด และ $H(n)$ แทนเลขฮาร์โมนิกที่ n (n-th harmonic number) ดังแสดงในสมการที่ 2.1

$$H(n) = \sum_{k=1}^{n} \frac{1}{k} \leq \ln n + 1 \qquad (2.1)$$

อัลกอริทึมเชิงละโมบอาจจะยังไม่สามารถหาคำตอบหมู่ย่อยเซตที่มีขนาดน้อยที่สุดได้ เนื่องจากเป็นการประมาณ ดังเช่นตัวอย่างในรูปที่ 2.6 จุดแทนสมาชิกในยูนิเวอร์ส วงกลมล้อมรอบแต่ละวงจะแทนเซตนำเข้าที่มีสมาชิกคือจุดภายในวง จะเห็นได้ว่า หากใช้อัลกอริทึมเชิงละโมบในการเลือกเซตคำตอบจะได้คำตอบออกมาเป็น $\{A, B, C\}$ ซึ่งมีขนาดของคำตอบเป็น 3 ในความจริงแล้วคำตอบที่ดีที่สุดควรจะได้เป็น $\{D, E\}$ ซึ่งเพียงพอที่จะครอบคลุมทุกจุดได้และมีขนาดของคำตอบเพียงแค่ 2 เท่านั้น



รูปที่ 2.6 ชุดข้อมูลนำเข้าสำหรับการประมาณคำตอบด้วยอัลกอริทึมเชิงละโมบ

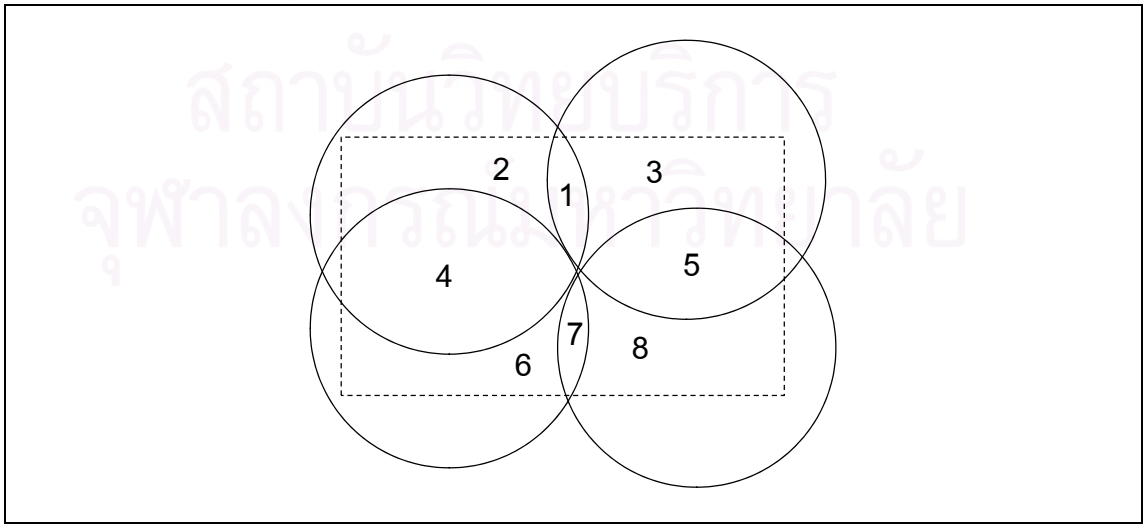
2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัยด้านการตรวจสอบความซ้ำซ้อนของโหมต

งานวิจัยหลายงานได้พยายามนำเสนอนิยามและวิธีการในการตรวจสอบความซ้ำซ้อนของโหมต ตัวอย่างเช่น งานวิจัยของ Slijepcevic S. และ Potkonjak M. [10] ได้นำเสนอวิธีการประหยัดการใช้พลังงานรวมของระบบเครือข่ายตัวรับรู้ โดยพยายามค้นหาความซ้ำซ้อนของโหมตที่เกิดขึ้นในระบบจากการพิจารณาว่า ภายในพื้นที่เฝ้าสังเกตทั้งหมดของระบบจะประกอบไปด้วยเซตของโหมตที่ทำหน้าที่ตรวจจับซ้ำซ้อนกันอยู่หลายเซตด้วยกัน เซตแต่ละเซตจะมีสมาชิกประกอบไปด้วยโหมตบางตัวในระบบซึ่งเมื่อยูเนียนพื้นที่ตรวจจับของโหมตทุกตัวในเซตแล้วจะสามารถครอบคลุมบริเวณเฝ้าสังเกตได้ทั้งหมด เซตของโหมตเหล่านี้จะเรียกว่า “โคเวอร์” (Cover)

งานวิจัยของ Slijepcevic S. และ Potkonjak M. [10] ได้นำเสนออัลกอริทึมที่ใช้ในการค้นหาจำนวนโคเวอร์ที่มากที่สุด โดยอาศัยพื้นฐานของปัญหาที่ชื่อว่าปัญหาเซตครอบคลุมขนาด k (Set k-Cover Problem) ซึ่งพยายามที่จะหาคำตอบของคำถามที่ว่า “มีโคเวอร์ของโหมตจำนวนมากถึง k โคเวอร์ในระบบหรือไม่?” โดยขั้นตอนของอัลกอริทึมในการค้นหาโคเวอร์จะมีดังนี้

1) แบ่งเขตพื้นที่ (Area field) ในพื้นที่เฝ้าสังเกตทั้งหมดตามการตัดกันของเส้นรอบวงกลมแทนพื้นที่ตรวจจับของโหมต ในรูปที่ 2.7 แสดงตัวอย่างการแบ่งเขตพื้นที่ภายในพื้นที่เฝ้าสังเกต (แสดงด้วยสี่เหลี่ยมเส้นประ) ซึ่งสามารถแบ่งออกได้ทั้งสิ้น 8 เขตพื้นที่ย่อย ได้แก่ เขตพื้นที่ย่อย 1 2 3 4 5 6 7 และ 8



รูปที่ 2.7 การแบ่งเขตพื้นที่ตามการตัดกันของเส้นรอบวงกลม

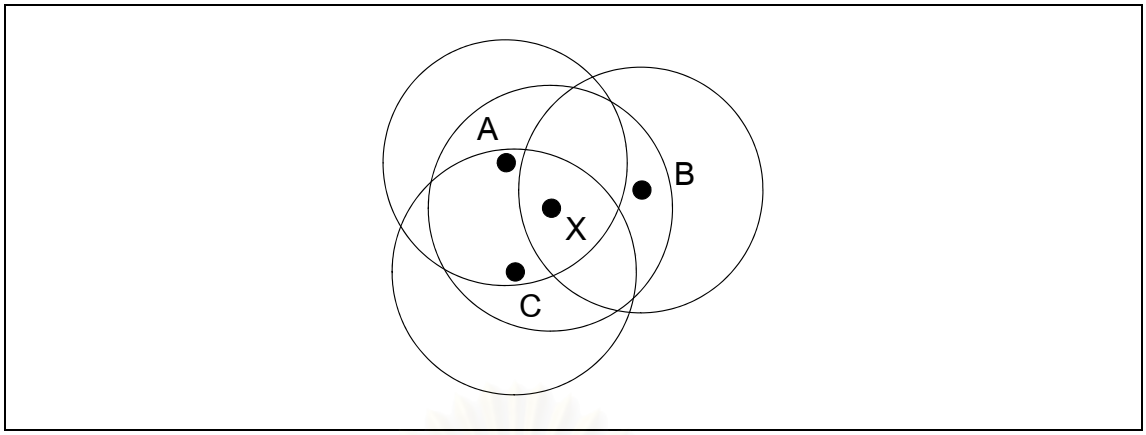
2) หาส่วนย่อยวิกฤต (Critical element) ได้แก่ เขตพื้นที่ที่อยู่ภายในพื้นที่ตรวจจับของโหมตจำนวนน้อยตัวที่สุด

3) นำโหมตทั้งหมดที่มีพื้นที่ครอบคลุมส่วนย่อยวิกฤตมาคำนวณหาค่าฮิวริสติก (Heuristic value) โหมตใดที่มีค่าฮิวริสติกมากสุดให้เลือกโหมตนั้นเก็บลงในโคเวอร์ จากนั้นตรวจสอบดูว่าทุกเขตพื้นที่ย่อยถูกครอบคลุมโดยโหมตใด ๆในโคเวอร์ที่หาได้หมดทุกเขตแล้วหรือไม่ ถ้าหากยังไม่ครบ ให้วนกลับไปหาส่วนย่อยวิกฤตใหม่ในข้อ 2 แล้วทำต่อไปเรื่อย ๆจนกว่าจะครบ

4) เมื่อทุกเขตข้อมูลถูกครอบคลุมโดยโหมตทั้งหมดในโคเวอร์แล้วแสดงว่าหาโคเวอร์ได้ครบสมบูรณ์ 1 โคเวอร์ โหมตทั้งหมดในโคเวอร์นั้นจะถูกตัดออกจากการพิจารณาในการค้นหารอบถัดไป โหมตที่เหลือจะนำไปเริ่มหาโคเวอร์ใหม่ กระบวนการจะทำงานวนซ้ำไปจนกว่าจะพบว่ามีเขตข้อมูลบางเขตไม่สามารถหาโหมตใด ๆมาครอบคลุมได้แล้วจึงจะเป็นอันเสร็จสิ้นกระบวนการและได้จำนวนโคเวอร์ทั้งหมดที่เป็นไปได้ตามต้องการ

อัลกอริทึมนี้สามารถหาจำนวนโคเวอร์ออกมาเกือบจะมากที่สุดเท่าที่จะเป็นไปได้ โดยโคเวอร์แต่ละชุดจะถือว่ามีความซ้ำซ้อนกันกับชุดอื่น จากผลการทดสอบเทียบกับการค้นหาแบบจำลองการอบเหนียว (Simulated annealing) พบว่าอัลกอริทึมนี้สามารถหาจำนวนโคเวอร์ได้มากกว่า อย่างไรก็ตามอัลกอริทึมนี้มีความจำเป็นที่ต้องใช้ข้อมูลแบบศูนย์รวมในการคำนวณและมีการปรับให้เป็นปัจจุบัน (Update) อยู่เสมอ ทำให้เสียโอเวอร์เฮดในการคำนวณสูงและไม่เหมาะต่อการทำงานของระบบเครือข่ายตัวรับรู้แบบไร้สายมากนัก

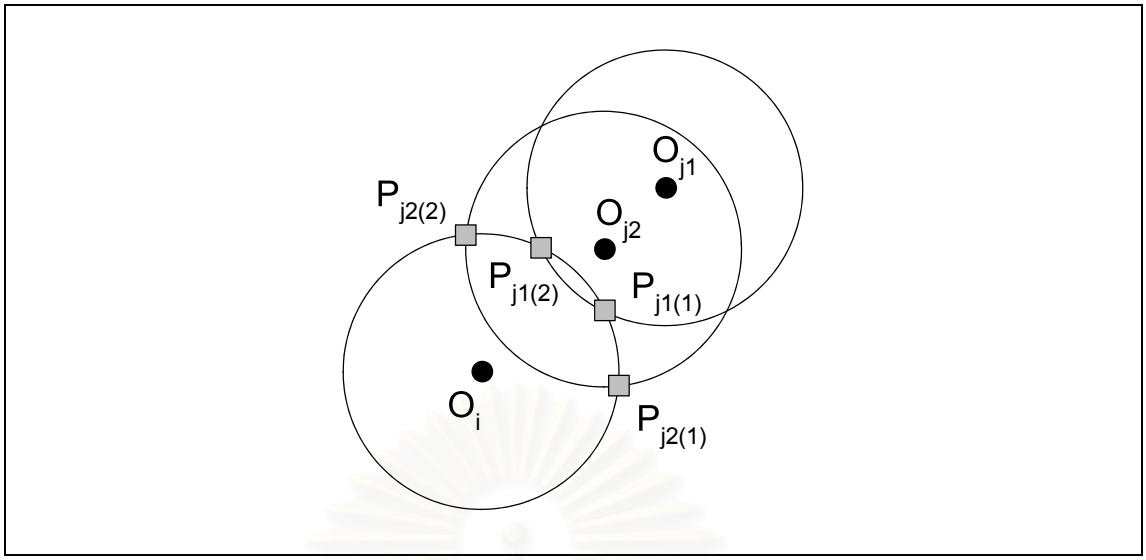
งานวิจัยของ Tian D. และ Georganas N.D. [4] มีการนำเสนอกฎที่ใช้ในการบ่งบอกความซ้ำซ้อนของโหมตโดยอาศัยหลักการคำนวณการครอบคลุมเส้นรอบวงกลมแทนการพิจารณาจากพื้นที่โดยตรง จากกฎกล่าวไว้ว่า ถ้าโหมตใดที่มีเส้นรอบวงของวงกลมแทนพื้นที่ตรวจจับอยู่ภายในพื้นที่ตรวจจับของโหมตเพื่อนบ้าน (Neighboring mote) ครบทั้งเส้น (360 องศา) แล้ว จะได้ว่าพื้นที่ตรวจจับของโหมตนั้นถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมตเพื่อนบ้านหมดแล้วเช่นกันและเรียกโหมตนั้นว่าเป็นโหมตซ้ำซ้อน (Redundant mote) นิยามความหมายของคำว่า โหมตเพื่อนบ้าน หรือเรียกอีกอย่างว่าโหมตสนับสนุน (Sponsoring mote) โดยหมายถึงโหมตที่มีตำแหน่งอยู่ห่างจากตัวโหมตอ้างอิงที่พิจารณาออกไปเป็นระยะไม่เกินหนึ่งช่วงรัศมีตรวจจับ (Sensing radius) ในรูปที่ 2.8 แสดงโหมต X ซึ่งเป็นโหมตซ้ำซ้อนและมีโหมต A B และ C เป็นโหมตเพื่อนบ้านซึ่งอยู่ห่างจากตำแหน่งของโหมต X ออกไปไม่เกินหนึ่งระยะรัศมีตรวจจับ จากรูปจะสังเกตได้ว่า เส้นรอบวงทั้งเส้นของวงกลมแทนพื้นที่ตรวจจับของโหมต X จะอยู่ภายใต้พื้นที่ตรวจจับของโหมตเพื่อนบ้าน A B และ C หมดทั้งเส้นแล้ว



รูปที่ 2.8 โหมดซ้ำซ้อนที่มีเส้นรอบวงแทนพื้นที่ถูกครอบคลุมโดยโหมดเพื่อนบ้าน

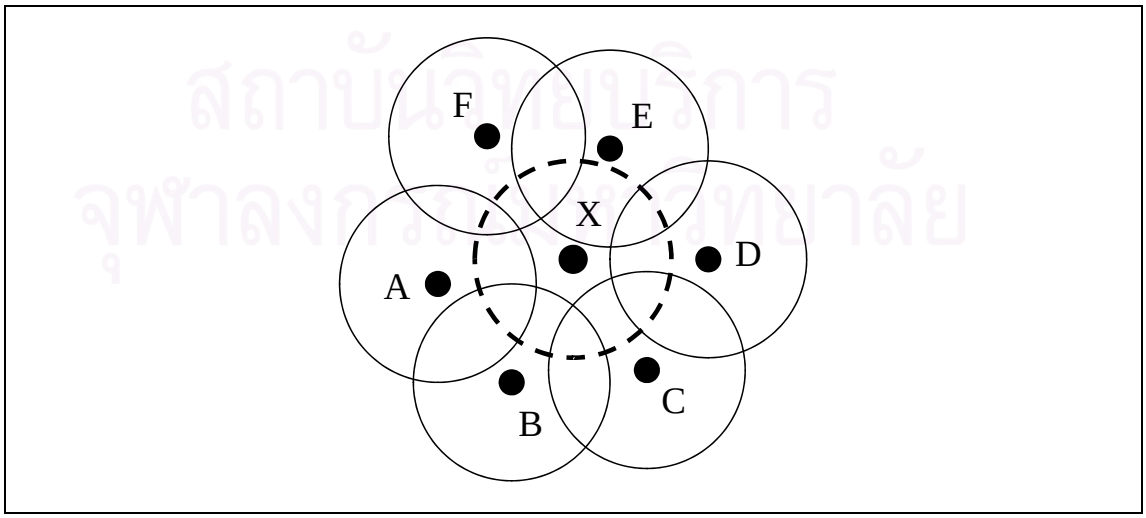
งานวิจัยของ Tian D. และ Georganas N.D. [4] ถูกนำมาพัฒนาให้ดีขึ้นโดยงานของ Jiang J. และ Dou W. [5] ซึ่งมีนำเสนอจุดเปลี่ยนสำคัญอยู่ 2 จุดด้วยกัน ได้แก่ จุดแรกคือการขยายขอบเขตของนิยามโหมดเพื่อนบ้าน จากความจริงที่ว่าการซ้อนทับกันของพื้นที่ตรวจจับระหว่าง 2 โหมดใด ๆไม่ได้เกิดขึ้นเฉพาะกับโหมดที่อยู่ห่างกันไม่เกินหนึ่งระยะรัศมีตรวจจับเพียงเท่านั้น โหมดที่อยู่ห่างออกไปไกลกว่าหนึ่งระยะรัศมีตรวจจับ (แต่ไม่เกิน 2 ระยะรัศมี) สามารถที่จะมีพื้นที่ตรวจจับทับซ้อนกันได้เช่นเดียวกัน การจำกัดนิยามของโหมดเพื่อนบ้านให้อยู่ห่างเพียงแค่ระยะหนึ่งรัศมีตรวจจับจะไม่สามารถครอบคลุมและค้นหาโหมดซ้ำซ้อนตามจำนวนที่มีจริงในระบบได้ทั้งหมด เพราะฉะนั้น Jiang J. และ Dou W. จึงได้ขยายนิยามของโหมดเพื่อนบ้านใหม่ จากเดิมที่จำกัดให้ต้องอยู่ห่างออกไปไม่เกินหนึ่งระยะรัศมีตรวจจับให้เป็นไม่เกิน 2 เท่าของระยะรัศมีตรวจจับแทน ทำให้โหมดแต่ละโหมดจะมีจำนวนโหมดเพื่อนบ้านเพิ่มมากขึ้นซึ่งเป็นการเพิ่มโอกาสที่จะสอบพบโหมดซ้ำซ้อนในระบบให้มากขึ้น

จุดที่สองคือนิยามใหม่ของกฎความซ้ำซ้อน จากเดิมที่อาศัยการครอบคลุมเส้นรอบวงทั้งหมดให้เปลี่ยนมาหันพิจารณาที่ส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมดเพื่อนบ้านแบบมีประสิทธิผล (Effective neighbor) แทน จากกฎกล่าวว่า ถ้าโหมดใดมีส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมดเพื่อนบ้านแบบมีประสิทธิผลที่อยู่ภายในพื้นที่ตรวจจับของโหมดที่กำลังตรวจสอบอยู่ภายในพื้นที่ตรวจจับของโหมดเพื่อนบ้านแบบมีประสิทธิผลตัวอื่นหมดทั้งส่วนแล้ว จะได้ว่าพื้นที่ตรวจจับของโหมดนั้นถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมดเพื่อนบ้านหมดแล้วเช่นกันและเรียกโหมดนั้นว่าเป็นโหมดซ้ำซ้อน นิยามของโหมดเพื่อนบ้านแบบมีประสิทธิผลคือโหมดเพื่อนบ้านที่ไม่ได้มีพื้นที่ตรวจจับทับซ้อนเป็นส่วนย่อย (Subset) ของพื้นที่ตรวจจับทับซ้อนจากโหมดเพื่อนบ้านอื่น ดังเช่นในรูปที่ 2.9 แสดงโหมดเพื่อนบ้านแบบมีประสิทธิผล O_{j2} ซึ่งมีพื้นที่ตรวจจับทับซ้อน (พื้นที่ระหว่างจุดตัด $P_{j2(1)}$ และ $P_{j2(2)}$) ครอบคลุมส่วนพื้นที่ตรวจจับทับซ้อนของโหมดเพื่อนบ้าน O_{j1} (พื้นที่ระหว่างจุดตัด $P_{j1(1)}$ และ $P_{j1(2)}$) เมื่อโหมดที่กำลังพิจารณาคือโหมด O_i



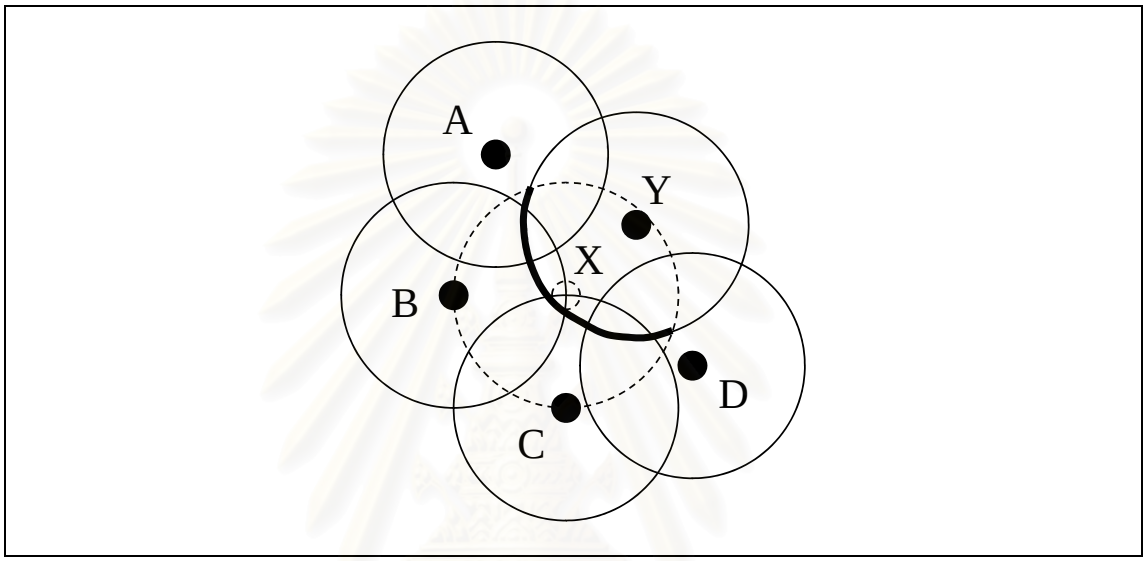
รูปที่ 2.9 โหมตเพื่อนบ้านแบบมีประสิทธิผล

กฎความซ้ำซ้อนที่พิจารณาการครอบคลุมส่วนของเส้นรอบวงของโหมตเพื่อนบ้านแบบมีประสิทธิผลนี้สามารถตรวจสอบพบกรณีการเกิดพื้นที่บอดบริเวณกึ่งกลางพื้นที่ตรวจจับของโหมตได้ด้วย ซึ่งกฎเดิมที่อาศัยการครอบคลุมเส้นรอบวงไม่สามารถตรวจสอบเจอได้ ดังเช่นแสดงให้เห็นในรูปที่ 2.10 เส้นรอบวงแทนพื้นที่ตรวจจับของโหมต X ถูกครอบคลุมหมดโดยพื้นที่ของโหมตเพื่อนบ้าน แต่โหมต X กลับไม่ใช่โหมตซ้ำซ้อนเนื่องจากว่ามีส่วนของพื้นที่ตรวจจับบริเวณตรงกลางที่ยังไม่ได้อยู่ในพื้นที่ตรวจจับของโหมตเพื่อนบ้านตัวใดเลย ซึ่งจากกฎความซ้ำซ้อนที่อาศัยการครอบคลุมส่วนของเส้นรอบวงของโหมตเพื่อนบ้านแบบมีประสิทธิผลจะสามารถบอกได้ว่า โหมต X นี้ไม่ใช่โหมตซ้ำซ้อน



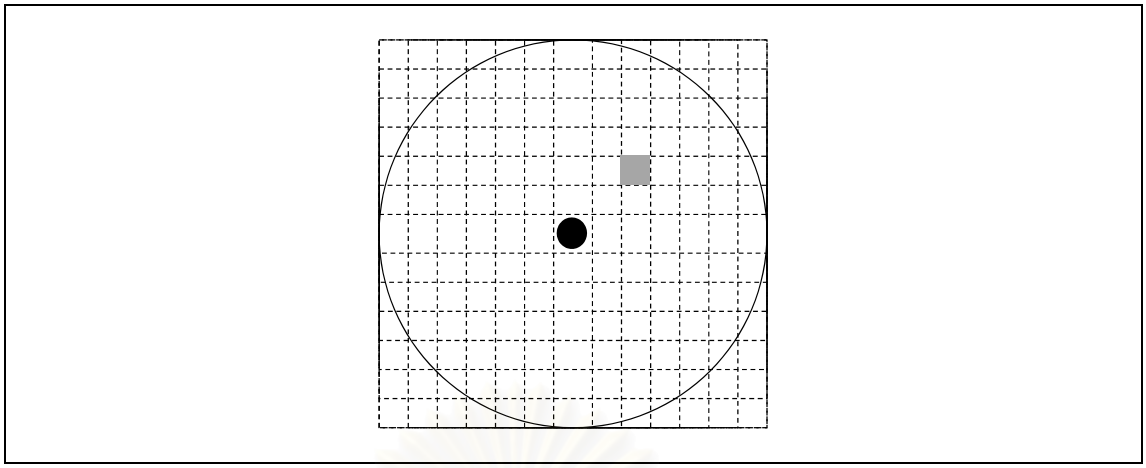
รูปที่ 2.10 การครอบคลุมเส้นรอบวงแทนพื้นที่ของโหมต

ในรูปที่ 2.11 จะแสดงกรณีที่เส้นรอบวงแทนพื้นที่ตรวจจับของโหมต X ถูกครอบคลุมหมดรวมทั้งส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมตเพื่อนบ้าน Y ที่อยู่ภายในพื้นที่ตรวจจับของโหมต X (เส้นหนา) ถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมตเพื่อนบ้าน A B C และ D เรียบร้อยแล้ว และหากพิจารณาที่โหมตเพื่อนบ้าน A B C และ D ส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมตเหล่านี้ที่อยู่ภายในพื้นที่ตรวจจับของโหมต X ก็ถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมตเพื่อนบ้านตัวอื่นเช่นเดียวกัน จึงสรุปได้ว่าโหมต X นี้เป็นโหมตซ้ำซ้อน



รูปที่ 2.11 การครอบคลุมส่วนของเส้นรอบวงที่อยู่ภายในพื้นที่ของโหมตซ้ำซ้อน

งานวิจัยที่กล่าวไปก่อนหน้านี้พยายามเลี่ยงการพิจารณาความซ้ำซ้อนจากการคำนวณที่พื้นที่โดยตรง อย่างไรก็ตามนอกเหนือจากนั้นแล้วยังมีงานวิจัยที่ทำการพิจารณาพื้นที่โดยตรงด้วย เช่น งานวิจัยของ Zhang H. และ Hou J.C. [11] ที่นำเสนอวิธีการตรวจสอบความซ้ำซ้อนของโหมตโดยอาศัยการแบ่งพื้นที่ตรวจจับของโหมตออกเป็นตาตาราง (Grid) ดังเช่นแสดงในรูปที่ 2.12 ทั้งนี้พื้นที่ตรวจจับจะถูกมองว่าประกอบไปด้วยช่อง (Cell) หลายๆช่องเรียงกันตามตารางที่แบ่ง การตรวจสอบความซ้ำซ้อนจะค้นหาว่าโหมตใดที่มีจุดกึ่งกลาง (Centroid) ของทุกช่องตารางในพื้นที่ตรวจจับทั้งหมดอยู่ภายในพื้นที่ของโหมตอื่น จะได้ว่าโหมตนั้นเป็นโหมตซ้ำซ้อนทันที อย่างไรก็ตามการทำงานของอัลกอริทึมวิธีนี้ถือได้ว่ามีการสิ้นเปลืองหน่วยความจำที่มากพอสมควร เพราะต้องมีการเก็บข้อมูลของทุกๆช่องตารางเพื่อใช้ในการตรวจสอบ ยิ่งอัตราส่วนละเอียดมากก็ต้องใช้หน่วยความจำมากซึ่งอาจไม่เหมาะนักกับระบบเครือข่ายตัวรับรู้ที่มีหน่วยความจำน้อย นอกจากนี้ความแม่นยำของคำตอบที่ได้ยังขึ้นอยู่กับอัตราส่วนการแบ่งตารางอีกด้วย



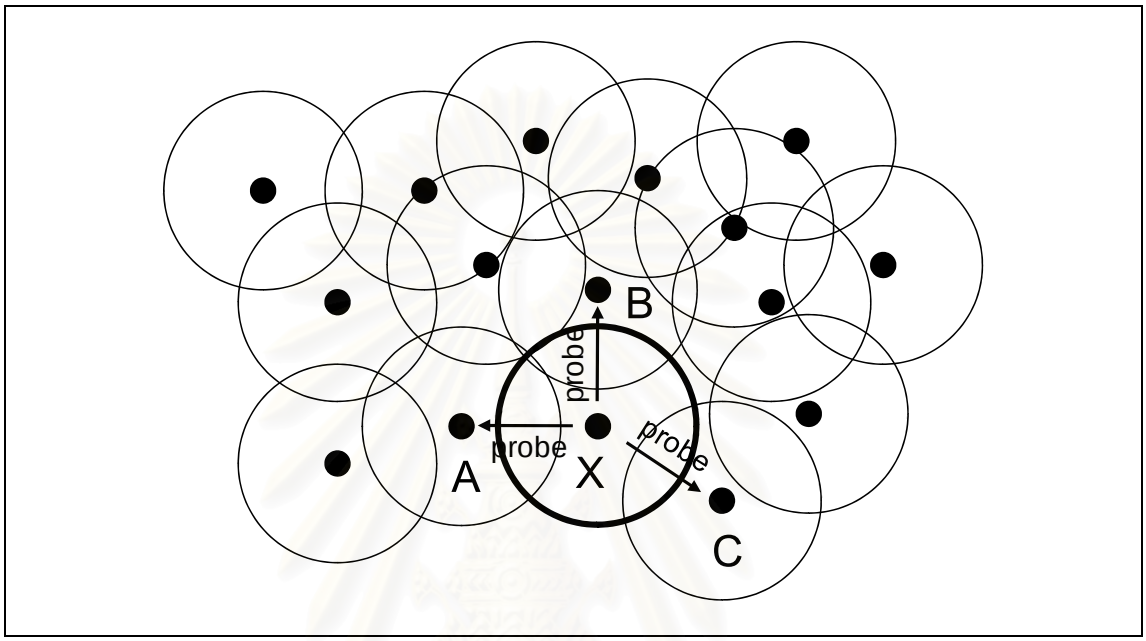
รูปที่ 2.12 การแบ่งตาตารางแทนพื้นที่วงกลม

2.2.2 งานวิจัยด้านการจัดกำหนดการตื่น-หลับของโหมดซ้ำซ้อน

ในระบบเครือข่ายตัวรับรู้ที่มีความหนาแน่นของโหมดต่อพื้นที่สูง อาจทำให้มีโหมดซ้ำซ้อนเกิดขึ้นเป็นจำนวนมากและกระจายตามบริเวณต่างๆในพื้นที่เฝ้าสังเกตได้ โหมดซ้ำซ้อนเหล่านี้ไม่สามารถที่จะเปลี่ยนเป็นสถานะหลับพร้อมกันได้หมดพร้อมๆกัน ณ ช่วงเวลาเดียวกัน เหตุเพราะอาจทำให้เกิดพื้นที่บอดขึ้นมาในระบบได้ เพราะฉะนั้นจำเป็นที่จะต้องมีกระบวนการในการจัดการช่วงเวลาตื่นและหลับของโหมดซ้ำซ้อนเหล่านี้เพื่อไม่ให้มีการเกิดพื้นที่บอดขึ้น

งานวิจัยของ Fan Y. และคณะ [12] ได้นำเสนออัลกอริทึมการจัดกำหนดการที่ชื่อว่า PEAS โดยอาศัยการสลับสับเปลี่ยนโหมดผลัดกันขึ้นมาทำงาน โหมดที่ไม่ได้ทำงานจะถูกเปลี่ยนสถานะให้อยู่ในสถานะหลับ โดยที่มีการตื่นขึ้นมาส่งข้อความถามโหมดเพื่อนบ้านเป็นระยะๆ (Probing) เพื่อตรวจสอบการมีอยู่ของโหมดเพื่อนบ้านรอบๆข้างว่ายังคงทำงานอยู่ดีหรือไม่ หากพบว่าไม่ได้รับข้อความแจ้งตอบกลับ (Reply message) จากโหมดเพื่อนบ้านใดเลย แสดงว่าในบริเวณพื้นที่นั้นอาจมีการสูญเสียโหมดไปบางส่วนจนทำให้อาจมีพื้นที่ที่ไม่ได้รับการตรวจจับเกิดขึ้น โหมดที่ตรวจสอบจะทำการปลุกตัวเองให้ตื่นขึ้นมาทำงานทดแทนทันที ตรงกันข้ามถ้าหากได้รับข้อความตอบรับกลับมาจากโหมดเพื่อนบ้านตัวใดตัวหนึ่ง แสดงว่าโหมดเพื่อนบ้านยังคงทำงานอยู่ปกติ ตัวเองก็จะกลับสู่สถานะหลับต่อไปและรอเวลาที่จะตื่นขึ้นมาตรวจสอบใหม่อีกครั้งในรอบถัดไป นอกจากนี้อัตราการสูญเสียโหมดในระบบกับอัตราการตื่นของโหมดขึ้นมาทำงานทดแทนนั้นจะถูกปรับ (Tune) ให้มีค่าที่ใกล้เคียงกัน ข้อดีของงานวิจัยนี้คือ ณ ช่วงเวลาหนึ่งๆ จะมีโหมดจำนวนน้อยมากที่ตื่นมาทำงาน ทำให้สามารถประหยัดพลังงานได้มากพอสมควร อัลกอริทึมเองมีขั้นตอนการทำงานที่ไม่ซับซ้อนและเสียเวลาการคำนวณไม่มากนัก

ในรูปที่ 2.13 แสดงลักษณะการส่งข้อความสอบถามการมีอยู่ของโหมดเพื่อนบ้านของโหมด X ซึ่งหากโหมดเพื่อนบ้าน A B และ C ยังคงเปิดทำงานดีอยู่ โหมด X จะได้รับข้อความตอบกลับ 3 ข้อความทั้งจาก A B และ C อย่างไรก็ตามหากโหมด X ไม่ได้รับข้อความจากโหมดใดเลย โหมด X จะปลุกตัวเองให้ตื่นขึ้นมาทำงานทันที



รูปที่ 2.13 การส่งข้อความสอบถามการมีอยู่ของโหมดเพื่อนบ้านสำหรับอัลกอริทึม PEAS

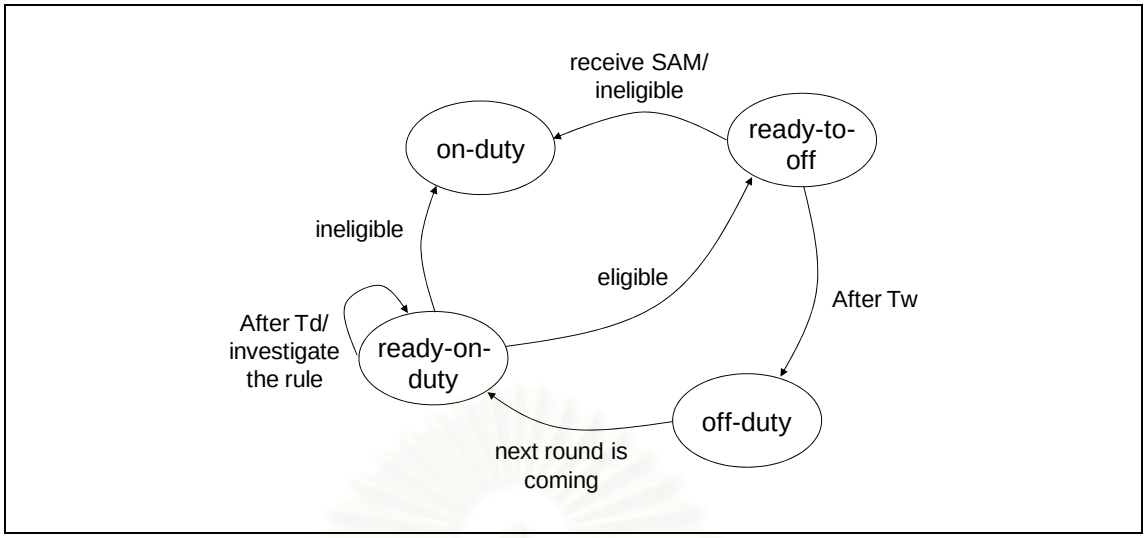
ข้อเสียของอัลกอริทึมนี้คือกรณีที่โหมดเพื่อนบ้านส่วนใหญ่ได้รับความเสียหายและตายไปเกือบหมด เหลือเพียงโหมดแค่ไม่กี่ตัวเท่านั้นที่ยังทำงานอยู่ ซึ่งตามความเป็นจริงแล้วสมควรที่จะปลุกโหมดที่หลับอยู่ขึ้นมาทำงานแทน อย่างไรก็ตามจากเงื่อนไขของการรับข้อความตอบกลับจากโหมดเพื่อนบ้านที่ขอแค่ได้รับข้อความเพียงข้อความเดียวจากโหมดเพื่อนบ้านใดสักตัวก็เพียงพอให้โหมดกลับไปสู่สถานะหลับเช่นเดิมได้ ทำให้โหมดส่วนมากที่หลับอยู่ยังคงไม่ตื่นขึ้นมาทำงานแทนตามที่ควรจะเป็น ตัวอย่างเช่นในรูปที่ 2.13 หากโหมด B และ C เสีย แต่โหมด A ยังคงทำงานดีอยู่ โหมด X จะได้รับข้อความจากโหมด A แล้วทำการหลับต่อไป ทั้งๆที่เกิดพื้นที่บอดจากการหายไปของโหมด B และ C แล้วในระบบ นอกจากนี้อัลกอริทึมยังไม่มีการพิจาณาเรื่องของพื้นที่ครอบคลุมด้วย ทำให้ระบบไม่สามารถที่จะรักษาคุณสมบัติพื้นที่ครอบคลุมเดิมก่อนการทำงานของอัลกอริทึมเอาไว้ได้ ซึ่งไม่เป็นผลดีสำหรับการเก็บข้อมูลในบริเวณพื้นที่ที่ต้องการข้อมูลครอบคลุมจากทุกๆจุด

เนื่องจาก PEAS ไม่สามารถรักษาคุณสมบัติพื้นที่ครอบคลุมของระบบเดิมได้ ดังนั้นงานวิจัยต่อๆมาจึงพยายามที่จะออกแบบการจัดกำหนดการตื่น-หลับของโหมดโดยที่เน้น

พิจารณาการรักษาคุณสมบัติพื้นที่ครอบคลุมของระบบควบคู่กันไปด้วย งานวิจัยของ Tian D. และ Georganas N.D. [4] มีการนำเสนอการจัดกำหนดการในการตื่น-หลับการทำงานของโหมต (Active node scheduling) โดยให้โหมตแต่ละโหมตมีการเปลี่ยนสถานะวนเป็นรอบๆไปตามเวลา โดยสถานะของโหมตจะมีด้วยกันทั้งสิ้น 4 สถานะได้แก่ สถานะ ready-on-duty สถานะ ready-to-off สถานะ on-duty และสถานะ off-duty การทำงานของโหมตในแต่ละสถานะจะมีดังนี้

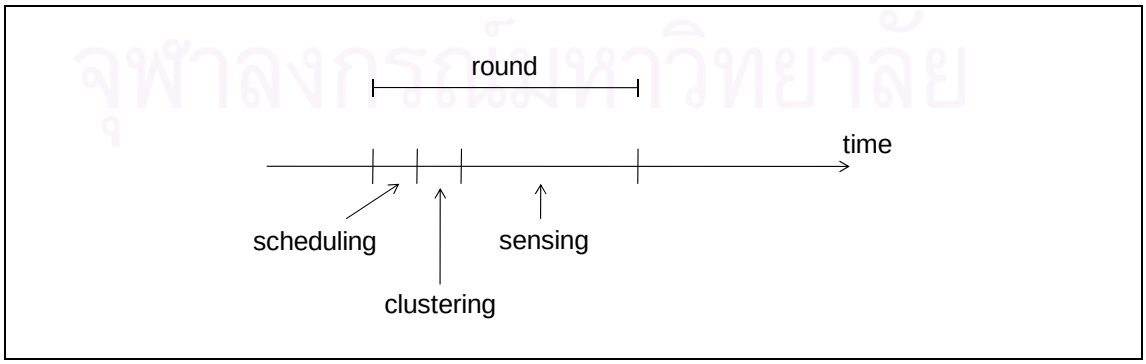
- ready-on-duty: สถานะที่โหมตทำการตรวจสอบความซ้ำซ้อน
- ready-to-off: สถานะที่โหมตซ้ำซ้อนรอและเตรียมตัวที่จะหลับ
- on-duty: สถานะที่โหมตเปิดทำงาน
- off-duty: สถานะที่โหมตเปลี่ยนสถานะเป็นหลับ

ในรูปที่ 2.14 แสดงแผนภาพวงจรการเปลี่ยนแปลงสถานะของโหมต โดยแรกเริ่มโหมตจะอยู่ในสถานะ ready-on-duty และทำการตรวจสอบว่าตัวเองเป็นโหมตซ้ำซ้อนหรือไม่ตามนิยามกฎความซ้ำซ้อนที่อาศัยการครอบคลุมเส้นรอบวงแทนพื้นที่ตรวจจับของโหมต หากพบว่าตัวเองไม่ได้เป็นโหมตซ้ำซ้อน ให้เปลี่ยนไปเป็นสถานะ on-duty เพื่อเปิดทำงานตามปกติ ตรงกันข้ามหากพบว่าตัวเองเป็นโหมตซ้ำซ้อน ให้ทำการเปลี่ยนสถานะไปที่ ready-to-off จากนั้นรอเป็นช่วงระยะเวลาหนึ่งเพื่อรอรับข้อความแจ้งเตือนการเปลี่ยนสถานะเป็นหลับจากโหมตข้างเคียงก่อนที่ตัวเองจะตัดสินใจหลับและเปลี่ยนไปสู่สถานะ off-duty เนื่องจากว่าการคำนวณความซ้ำซ้อนของโหมตใดๆนั้นจะต้องอาศัยอยู่บนสมมติฐานที่ว่าโหมตเพื่อนบ้านข้างเคียงจะต้องเปิดทำงานทุกโหมต ดังนั้นหากระหว่างการทำงานมีโหมตเพื่อนบ้านใดที่หลับไปย่อมส่งผลให้สถานะความซ้ำซ้อนของตัวเองอาจเปลี่ยนแปลงได้ เพราะฉะนั้นในสถานะ ready-to-off จะทำการรอรับข้อความแจ้งเตือนจากเพื่อนเป็นช่วงเวลาขณะหนึ่งเสียก่อน โดยช่วงเวลานี้จะนานมากน้อยแค่ไหนขึ้นกับการสุ่มตัวจับเวลา (Random timer) ของแต่ละโหมตที่ได้ออกมาไม่เท่ากัน หากพบว่าในช่วงเวลานี้ได้รับข้อความแจ้งเตือนจากโหมตเพื่อนบ้านใดๆ ให้ทำการตรวจสอบความซ้ำซ้อนของตัวเองใหม่อีกครั้ง หากพบว่าตัวเองไม่อยู่ในสถานะซ้ำซ้อนอีกต่อไปแล้วก็ให้เปลี่ยนเข้าไปสู่สถานะ on-duty และเปิดทำงานทันที แต่หากว่ายังคงซ้ำซ้อนอยู่ก็ให้วนเข้าสู่สถานะ ready-to-off ใหม่อีกครั้ง



รูปที่ 2.14 วงจรการเปลี่ยนแปลงสถานะของโหมดซ้ำซ้อน

งานวิจัยของ Tian D. และ Georganas N.D. [4] นี้ทำการทดสอบบนพื้นฐานโปรโตคอล LEACH (LEACH protocol) [13] โดยที่ในหนึ่งรอบ (Round) ของกระบวนการทำงานจะถูกแบ่งออกเป็น 3 ขั้นตามแกนเวลา ได้แก่ ส่วนการจัดกำหนดการ (Scheduling) ส่วนการจัดกลุ่ม (Clustering) และส่วนการตรวจจับ (Sensing) ในรูปที่ 2.15 แสดงลำดับการทำงานของ 3 ส่วนนี้ และเมื่อทำการทดสอบแล้วจะเห็นได้ว่า อายุเวลาการทำงานของระบบเครือข่ายจะนานขึ้นและพลังงานรวมที่ระบบใช้มีค่าลดน้อยลง อย่างไรก็ตามข้อจำกัดของงานวิจัยนี้ที่เห็นได้ชัดคือ หลักการเลือกตื่นและหลับโหมดนั้นยังคงต้องอาศัยตัวจับเวลาที่มีการตั้งเวลาแบบสุ่มมาใช้ในการตัดสินใจอยู่ ทำให้โหมดซ้ำซ้อนทุกโหมดมีโอกาสเฉลี่ยที่จะถูกเลือกใกล้เคียงกัน ซึ่งในความเป็นจริงแล้วโหมดบางโหมดนั้นเหลือพลังงานให้ใช้งานอยู่น้อยมากและสมควรที่จะถูกเลือกให้หลับมากกว่าก็ตาม แต่งานวิจัยนี้ไม่ได้ดึงเอาปัจจัยด้านพลังงานที่ว่านี้มาใช้พิจารณาในการคำนวณแต่อย่างใด



รูปที่ 2.15 ขั้นตอนการทำงานของกระบวนการจัดกำหนดการเทียบกับเวลา

บทที่ 3

การออกแบบอัลกอริทึมควบคุมความหนาแน่น

วิทยานิพนธ์นี้ได้นำเสนออัลกอริทึมควบคุมความหนาแน่นบนระบบเครือข่ายตัวรับรู้แบบไร้สาย โดยมีวัตถุประสงค์เพื่อช่วยประหยัดปริมาณการใช้พลังงานรวมของระบบลง โดยที่การทำงานของอัลกอริทึมจะแบ่งได้ออกเป็น 2 ขั้นตอนหลัก คือ ขั้นตอนแรกเป็นการตรวจสอบความซ้ำซ้อนของโหมตในระบบ (Redundancy determination) โดยอาศัยกฎความซ้ำซ้อน (Redundancy rule) และหลักการการครอบคลุมเส้นรอบวงแทนพื้นที่ [4] รวมทั้งนิยามโหมตเพื่อนบ้านแบบมีประสิทธิผล [5] มาใช้ และขั้นตอนที่สองเป็นการนำเสนอกระบวนการจัดกำหนดการตื่น-หลับของโหมตซ้ำซ้อน (Activation scheduling) ซึ่งจะประกอบไปด้วยวิธีการเลือกสถานะตื่น-หลับของโหมตซ้ำซ้อน (Active mote selection) จากพื้นฐานปัญหาพื้นที่ครอบคลุมและเซตครอบคลุม รวมไปถึงเกณฑ์วิธีการลงมติ (Voting protocol) เพื่อตัดสินหาคำตอบสุดท้ายในการเลือกเปลี่ยนสถานะของโหมต กระบวนการทำงานของอัลกอริทึมทั้งหมดจะมีรายละเอียดดังนี้

3.1 ข้อสมมติฐานของระบบเครือข่ายตัวรับรู้

การทำงานของอัลกอริทึมจะอยู่บนข้อสมมติฐานทั้งสิ้น 6 ข้อ ดังนี้

1) โหมตในระบบเครือข่ายตัวรับรู้แบบไร้สายที่พิจารณาจะวางตัวอยู่บนระนาบ 2 มิติ ที่มีพิกัดเป็นแกน X และแกน Y เท่านั้น

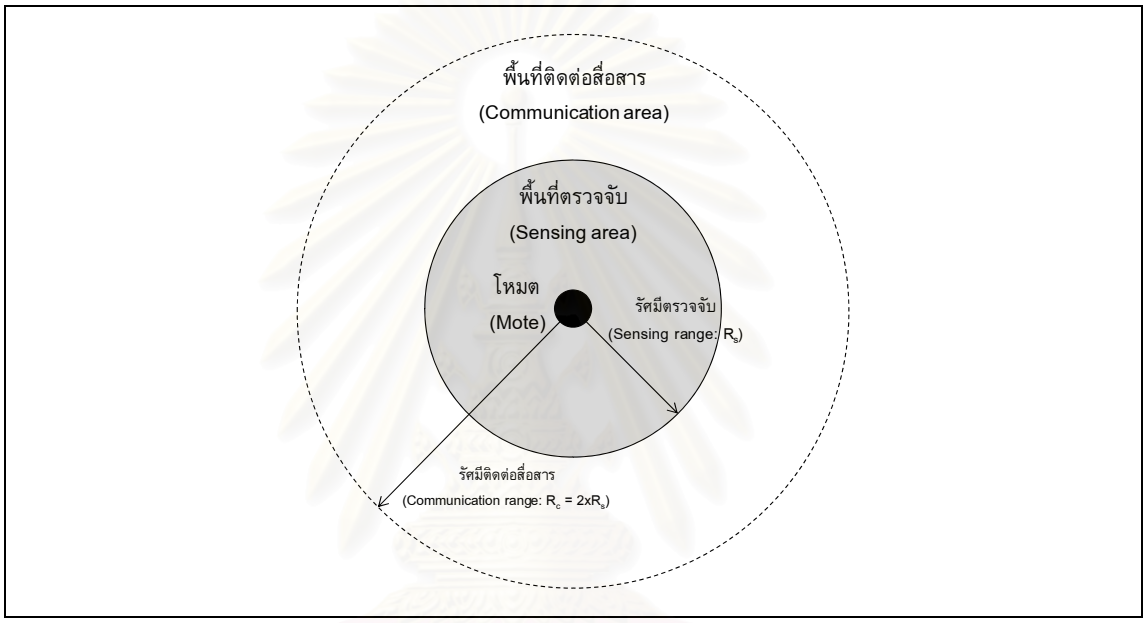
2) โหมตแต่ละโหมตจะรู้ตำแหน่ง (Location) ของตัวเอง ซึ่งได้ออกมาในรูปแบบของคู่ลำดับ (X, Y) เมื่อเทียบกับตำแหน่งอ้างอิงหนึ่งๆที่กำหนดไว้

3) พื้นที่ตรวจจับของโหมตจะแทนด้วยรูปแบบจำลองวงกลมระนาบ 2 มิติ โดยมีโหมตอยู่ ณ ตำแหน่งจุดศูนย์กลางของวง ดังแสดงไว้ในรูปที่ 3.1 โดยที่รัศมีการตรวจจับ (Sensing radius) ของทุกโหมตถูกกำหนดให้มีระยะที่เท่ากัน และมีรัศมีการติดต่อสื่อสารกับโหมตข้างเคียง (Communication radius) มีค่าเป็น 2 เท่าของรัศมีตรวจจับ เพื่อรับประกันคุณสมบัติการเชื่อมต่อ (Connectivity) ของระบบ โหมตทุกโหมตสามารถปรับค่าพารามิเตอร์รัศมีทั้ง 2 ค่านี้ได้จากการตั้งค่าความแรงของสัญญาณที่ตัวรับส่งสัญญาณ

4) โหมตทุกโหมตจะอยู่ในตำแหน่งที่ต่างกัน ไม่มีการซ้อนทับกันของโหมตมากกว่า 1 โหมต ณ จุดตำแหน่งเดียวกัน

5) พลังงานไฟฟ้าของโหมตทุกโหมตกำหนดให้มีปริมาณเริ่มต้น (Initial energy) เท่ากัน และมีอัตราการสูญเสียพลังงานไฟฟ้า (Energy consumption rate) เท่ากันเมื่อทำงานอย่างเดียวกัน ณ ช่วงเวลาที่เท่ากัน

6) คุณสมบัติพื้นที่ครอบคุลมของระบบเครือข่ายที่พิจารณาจะมีอันดับของการครอบคลุมเท่ากับ 1 (1-coverage) ทุกๆจุดในพื้นที่เฝ้าสังเกตที่ระบบต้องการเก็บข้อมูลจะต้องอยู่ภายในพื้นที่ตรวจจับของโหมตอย่างน้อย 1 โหมต



รูปที่ 3.1 สมมติฐานแบบจำลองพื้นที่ตรวจจับและพื้นที่ติดต่อสื่อสารของโหมต

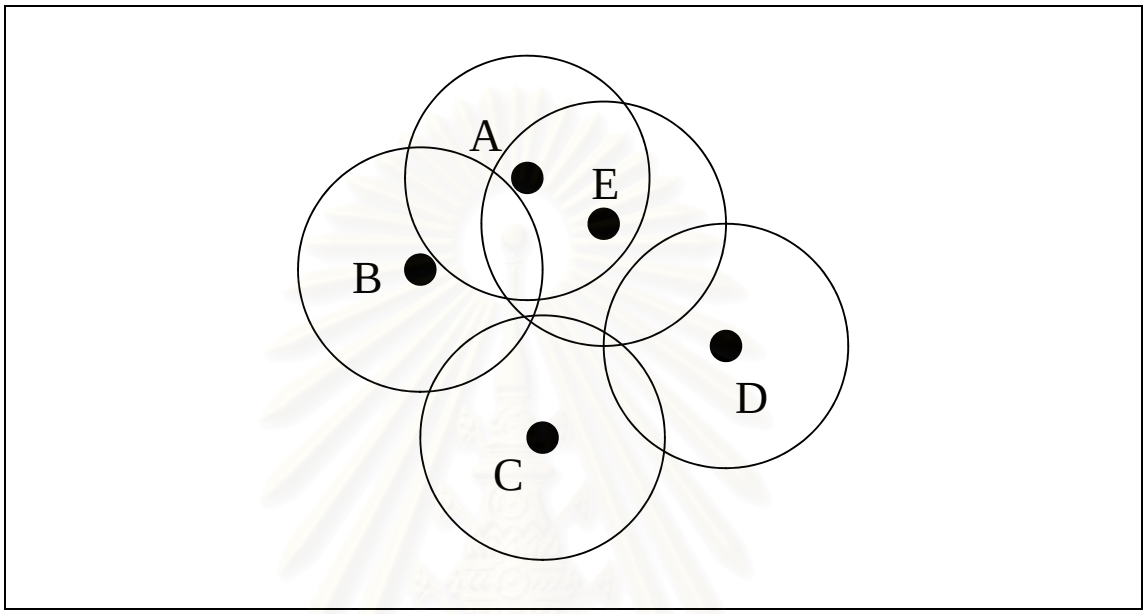
3.2 การตรวจสอบความซ้ำซ้อนของโหมตบนพื้นฐานปัญหาพื้นที่ครอบคลุม

จากปัญหาพื้นที่ครอบคลุม โหมตใดที่มีพื้นที่ตรวจจับครอบคลุมทับซ้อนบริเวณเดียวกันกับโหมตเพื่อนบ้านจะถือว่าโหมตนั้นเป็นโหมตซ้ำซ้อน เพราะฉะนั้นกระบวนการในการตรวจสอบจึงสามารถแบ่งออกได้เป็น 3 ส่วนสำคัญ ได้แก่ การนิยามความหมายของคำว่าโหมตเพื่อนบ้าน กฎความซ้ำซ้อนของโหมตที่ใช้ตัดสินว่าโหมตใดซ้ำซ้อนหรือไม่ และสุดท้ายคือขั้นตอนการทำงานเวลาตรวจสอบจริง โดยทั้ง 3 ส่วนจะมีรายละเอียดดังนี้

3.2.1 นิยามโหมตเพื่อนบ้าน

นิยาม: เมื่อกำหนดให้โหมตมีรัศมีตรวจจับเท่ากัน โหมต A ใดๆในระบบจะถือว่าเป็นโหมตเพื่อนบ้านของโหมต B ก็ต่อเมื่อตำแหน่งของโหมต A อยู่ห่างจากตำแหน่งของโหมต B เป็นระยะทางน้อยกว่าหรือเท่ากับ 2 เท่าของรัศมีตรวจจับ

ในรูปที่ 3.2 แสดงตัวอย่างระบบเครือข่ายที่ประกอบไปด้วยโหมตทั้งสิ้น 5 โหมต ได้แก่ โหมต A B C D และ E ตามนิยามโหมตเพื่อนบ้านแล้ว โหมต A จะมีเซตของโหมตเพื่อนบ้าน คือ {B, E} ในขณะที่โหมต B C D และ E จะมีเซตของโหมตเพื่อนบ้านเป็น {A, C, E} {B, D, E} {C, E} และ {A, B, C, D} ตามลำดับ



รูปที่ 3.2 นิยามโหมตเพื่อนบ้านที่ใช้ในการตรวจสอบความซ้ำซ้อน

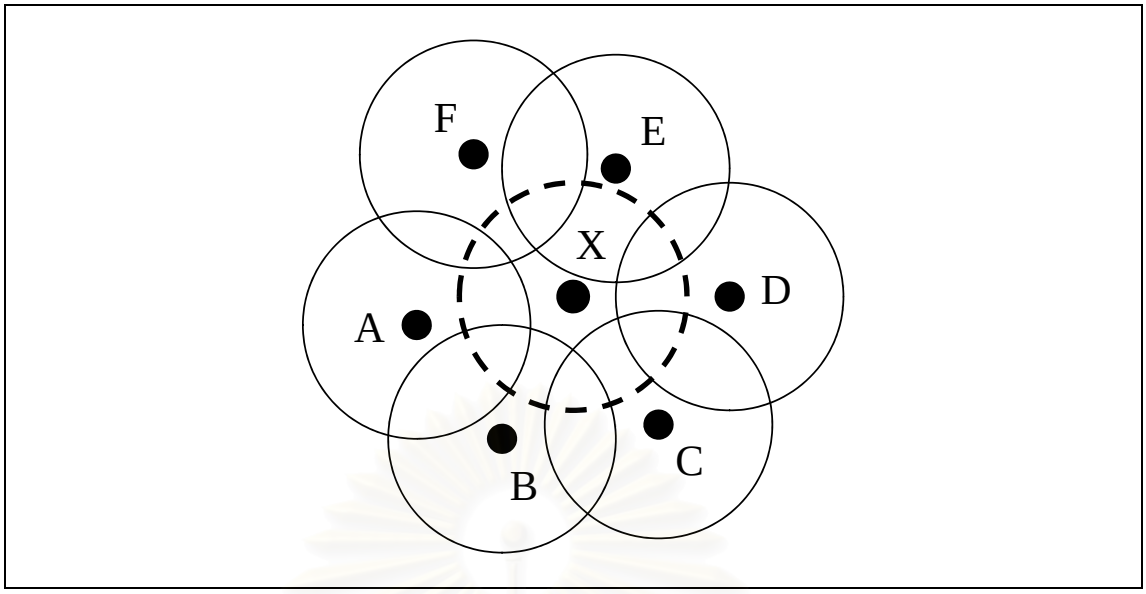
3.2.2 กฎความซ้ำซ้อนของโหมต

เมื่อต้องการตรวจสอบดูว่า โหมตใดถือเป็นโหมตซ้ำซ้อนหรือไม่ สามารถตรวจสอบได้จากกฎความซ้ำซ้อนของโหมตดังนี้

<u>กฎความซ้ำซ้อน</u>: โหมตใดจะเป็นโหมตซ้ำซ้อนก็ต่อเมื่อมีคุณสมบัติตรงตามเงื่อนไขทั้ง 2 ประการดังต่อไปนี้

1) เส้นรอบวงกลมแทนพื้นที่ตรวจจับของโหมต: *"ถ้าโหมตเป็นโหมตซ้ำซ้อนแล้ว จะได้ว่าเส้นรอบวงกลมแทนพื้นที่ตรวจจับของโหมตจะต้องถูกครอบคลุมหรืออยู่ในอาณาบริเวณพื้นที่ตรวจจับของโหมตเพื่อนบ้านข้างเคียงครบทั้งเส้น (360 องศา)"* [4]

ในรูปที่ 3.3 เมื่อพิจารณาตรวจสอบความซ้ำซ้อนของโหมต X จะเห็นว่าเส้นรอบวงแทนพื้นที่ของโหมต X (แทนด้วยเส้นประ) จะอยู่ภายในพื้นที่ครอบคลุมของโหมต A B C D E และ F ครบทั้งเส้น เพราะฉะนั้นจากเงื่อนไขคุณสมบัติเส้นรอบวงของกฎความซ้ำซ้อน โหมต X มีโอกาสที่จะเป็นโหมตซ้ำซ้อนได้



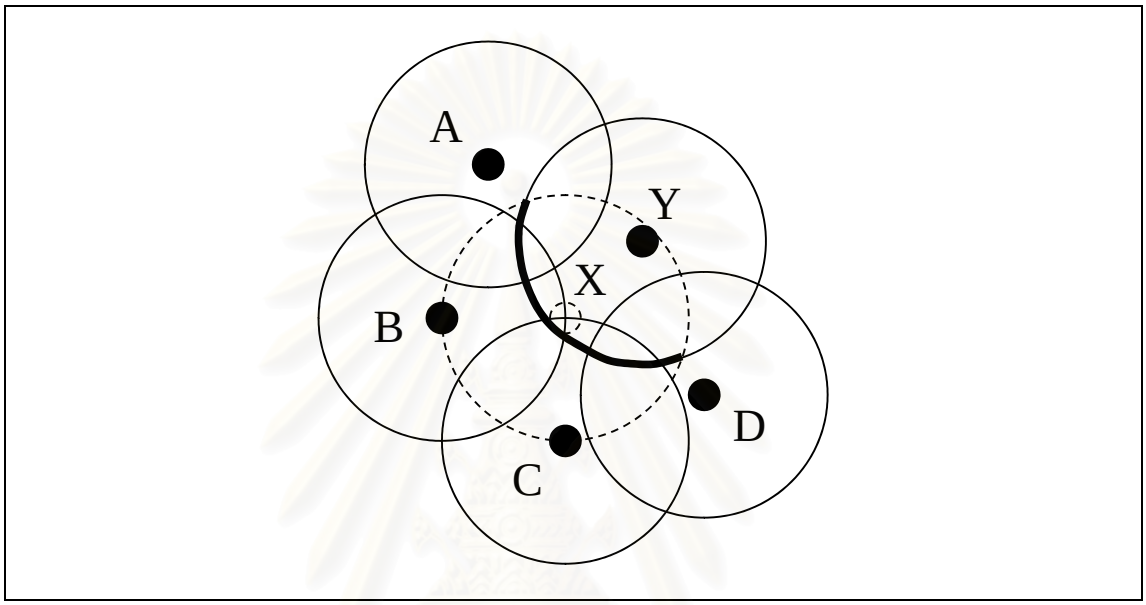
รูปที่ 3.3 ตัวอย่างโหมตที่มีเส้นรอบวงแทนพื้นที่ถูกครอบคลุมครบ 360 องศา

อย่างไรก็ตามเงื่อนไขคุณสมบัติเส้นรอบวงเพียงอย่างเดียวยังไม่เพียงพอในการที่จะสรุปให้โหมตใดเป็นโหมตซ้ำซ้อนได้แน่นอน สังเกตได้จากรูปที่ 3.3 จะเห็นว่า มีพื้นที่บางส่วนบริเวณตรงกลางวงกลมใกล้ตำแหน่งของโหมต X ที่ไม่ได้ถูกครอบคลุมโดยพื้นที่ของโหมตอื่นใดๆเลยซึ่งอาจเกิดเป็นพื้นที่บอดได้หากโหมต X หลับไป ทำให้แม้ว่า โหมต X จะมีเส้นรอบวงถูกครอบคลุมหมดแล้วจริงแต่ก็ยังคงไม่สามารถเป็นโหมตซ้ำซ้อนได้ ดังนั้นจึงต้องมีการตรวจสอบคุณสมบัติในเงื่อนไขที่ 2) ต่อไป

2) ส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับที่อยู่ภายในพื้นที่ตรวจจับของโหมตที่กำลังตรวจสอบ: *“ถ้าโหมตเป็นโหมตซ้ำซ้อนแล้ว จะได้ว่าส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมตเพื่อนบ้านทุกโหมตที่อยู่ภายในบริเวณพื้นที่ตรวจจับของตนจะต้องถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมตเพื่อนบ้านตัวอื่นครบทั้งส่วน”* [5]

จากความจริงที่ว่า ถ้ามีพื้นที่บอดเกิดขึ้นแล้วจะต้องมีโหมตเพื่อนบ้านที่มีส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับทำหน้าที่เป็นขอบเขต (Boundary) ของพื้นที่บอดอยู่อย่างน้อย 1 โหมตเสมอ เพราะฉะนั้นในการตรวจสอบกรณีของพื้นที่บอดที่อาจเกิดขึ้นนั้นสามารถทำได้โดยการตรวจว่า มีส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมตเพื่อนบ้านใด (ที่อยู่ภายในพื้นที่ตรวจจับของโหมตที่กำลังตรวจสอบ) ที่ไม่ถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมตเพื่อนบ้านตัวอื่นหรือไม่ หากไม่มีแสดงว่าพื้นที่โหมตเพื่อนบ้านทุกโหมตมีการเชื่อมซ้อนกันหมด ทำให้หากหลับโหมตที่ตรวจสอบนี้ไปก็ไม่ทำให้มีพื้นที่บอดเกิดขึ้นแน่นอน โหมตนี้จึงเป็นโหมตซ้ำซ้อน ตรงกันข้ามหากมีส่วนของเส้นรอบวงใดสักวงที่ไม่ถูกครอบคลุมจะแสดงว่าการหลับโหมตนี้ไปจะทำให้มีพื้นที่บอดเกิดขึ้นในระบบได้ โหมตนี้จึงไม่สามารถเป็นโหมตซ้ำซ้อนได้

ในรูปที่ 3.4 เมื่อโหมดที่พิจารณาคือโหมด X จะได้ว่า ส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของโหมด Y ที่อยู่ภายในพื้นที่ของ X (แสดงด้วยเส้นทึบ) ถูกครอบคลุมโดยพื้นที่ตรวจจับของโหมดเพื่อนบ้านอื่น ได้แก่ โหมด A B C และ D หมดทั้งส่วน นอกจากนี้โหมด X ยังมีคุณสมบัติการครอบคลุมเส้นรอบวงแทนพื้นที่ตามเงื่อนไขในข้อ 1) ด้วย ดังนั้นจึงสามารถสรุปได้ว่า โหมด X นี้เป็นโหมดซ้ำซ้อน (ในกรณีที่โหมด A B C และ D ทำงานตามปกติ)



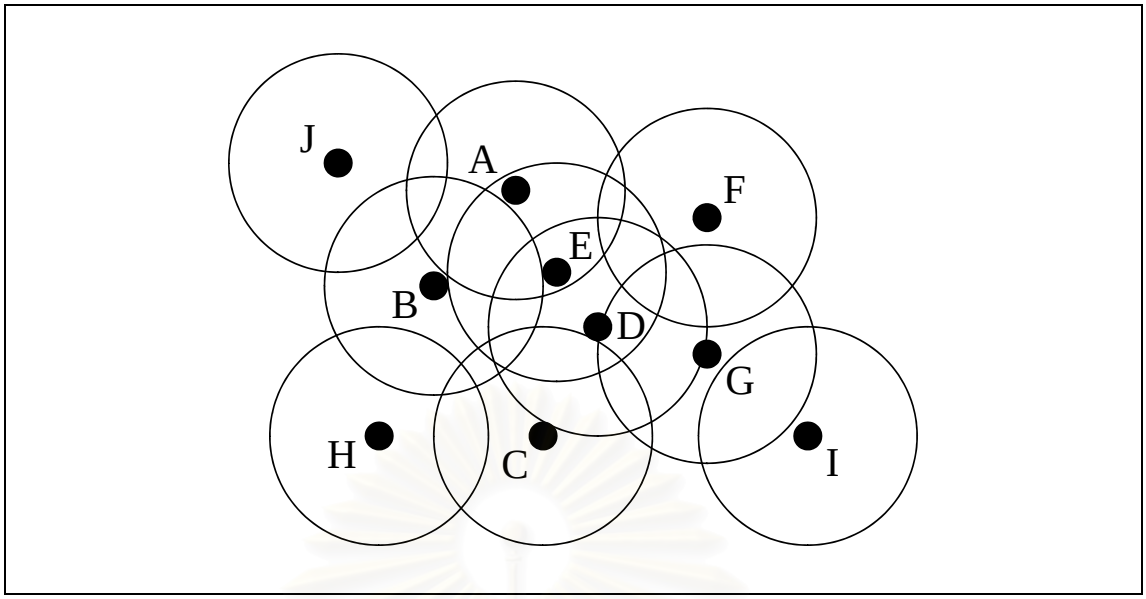
รูปที่ 3.4 ตัวอย่างโหมดที่มีส่วนของเส้นรอบวงแทนโหมดเพื่อนบ้านถูกครอบคลุมหมด

หากตรวจสอบระบบตัวอย่างในรูปที่ 3.3 ตามเงื่อนไขที่ 2) นี้ จะพบว่า โหมด X ไม่สามารถเป็นโหมดซ้ำซ้อนได้ เพราะส่วนของเส้นรอบวงแทนพื้นที่ตรวจจับของทั้งโหมด A B C D E และ F ที่อยู่ภายในพื้นที่ตรวจจับของโหมด X นั้นถูกครอบคลุมแค่เฉพาะบางส่วนของเส้นเท่านั้น เช่น ส่วนเส้นรอบวงจากโหมด A จะถูกครอบคลุมโดยเฉพาะโหมด B และ F เท่านั้น ซึ่งไม่เพียงพอในการที่จะครอบคลุมทั้งส่วนได้หมด ดังนั้นระบบนี้หากหลับโหมด X ไปจะทำให้มีพื้นที่บอดตรงกลางเกิดขึ้นโดยเส้นขอบเขตของพื้นที่บอดจะประกอบไปด้วยส่วนของเส้นรอบวงจากโหมดเพื่อนบ้านทุกโหมดของ X

3.2.3 ขั้นตอนการตรวจสอบโหมดซ้ำซ้อน

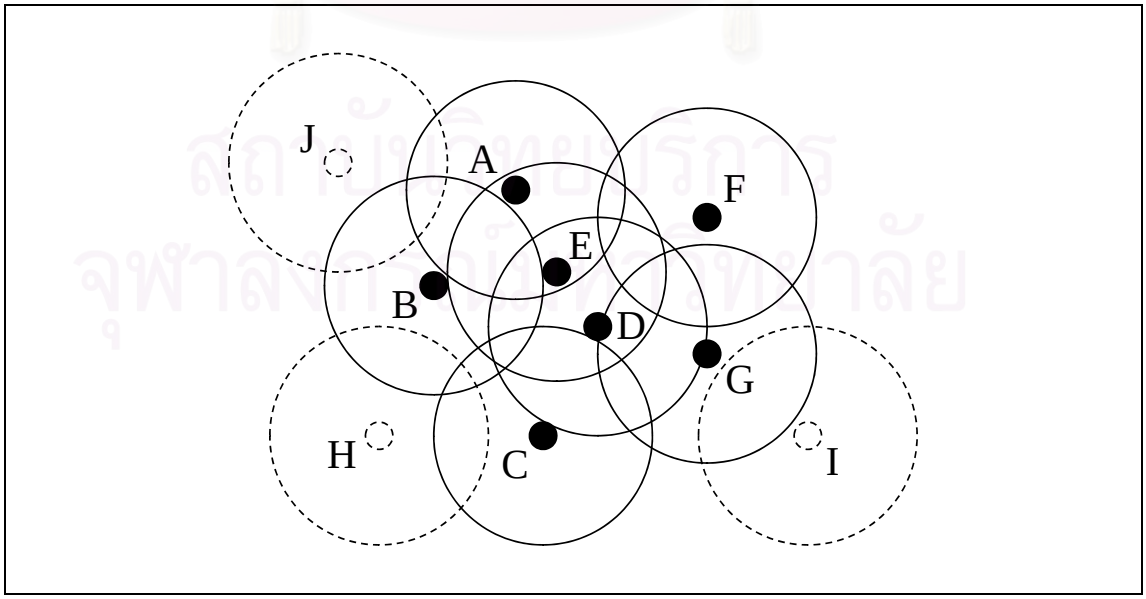
ขั้นตอนการทำงานของกระบวนการตรวจสอบความซ้ำซ้อนจะสามารถแสดงได้จากตัวอย่างอ้างอิงในรูปที่ 3.5 เมื่อให้โหมดที่พิจารณาตรวจสอบคือโหมด E

1) โหมด E ทำการส่งข้อมูลกระจาย (Broadcast) ตำแหน่ง (Location) ของตนไปยังโหมดรอบข้างพร้อมกันนั้นก็รอรับข้อมูลตำแหน่งที่ส่งมาจากโหมดอื่นด้วย



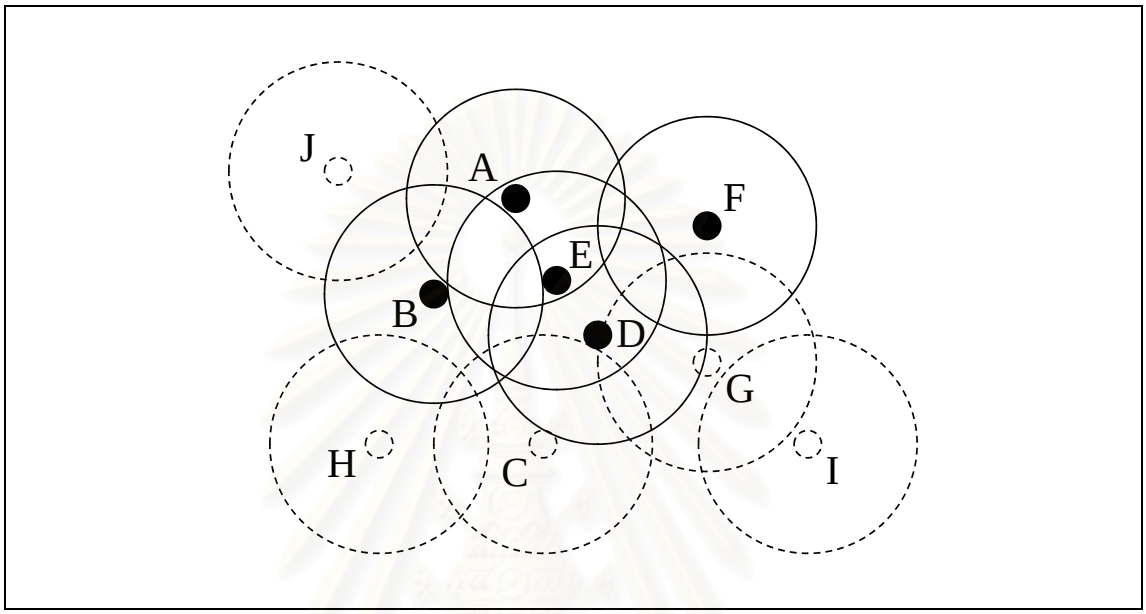
รูปที่ 3.5 ขั้นตอนการตรวจสอบความซ้ำซ้อน(1)

2) จากข้อมูลตำแหน่งของโหมตอื่นๆที่ได้รับ โหมต E สามารถตรวจสอบได้ว่า โหมตรอบข้างตัวใดบ้างที่เป็นโหมตเพื่อนบ้านของตน โดยดูจากระยะทางจากตำแหน่งของโหมตนั้นๆถึงตำแหน่งของตนว่ามากกว่า 2 เท่าของรัศมีตรวจจับหรือไม่ ถ้าไม่แสดงว่าโหมตนั้นเป็นโหมตเพื่อนบ้านของ E ซึ่งจากในรูปที่ 3.5 โหมต E จะมีโหมตเพื่อนบ้านทั้งหมด 6 ตัว ได้แก่ A B C D G และ F โหมตอื่นๆนอกจากนี้จะถือว่าไม่ใช่โหมตเพื่อนบ้านและถูกตัดออกจากพิจารณาไป ดังแสดงให้เห็นด้วยเส้นประในรูปที่ 3.6



รูปที่ 3.6 ขั้นตอนการตรวจสอบความซ้ำซ้อน(2)

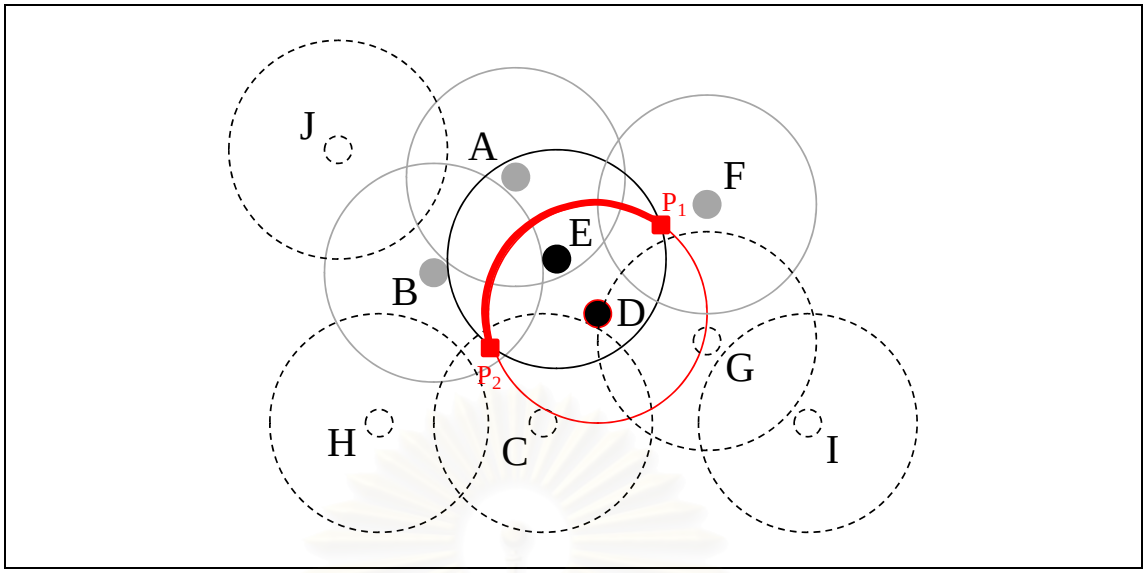
3) จากนิยามโหมตเพื่อนบ้านแบบมีประสิทธิผล [5] โหมตเพื่อนบ้านทั้งหมดจะถูกนำมาคัดเก็บเอาเฉพาะโหมตเพื่อนบ้านแบบมีประสิทธิผลไว้ โหมตที่เหลือให้ตัดออกจากพิจารณาไป จากตัวอย่างในรูปที่ 3.6 โหมตเพื่อนบ้านแบบมีประสิทธิผลของโหมต E มีอยู่ทั้งสิ้น 4 โหมต ได้แก่ โหมต A B D และ F ดังแสดงในรูปที่ 3.7



รูปที่ 3.7 ขั้นตอนการตรวจสอบความซ้ำซ้อน(3)

4) นำตำแหน่งของโหมตเพื่อนบ้านแบบมีประสิทธิผลทั้งหมดมาคำนวณการครอบคลุมเส้นรอบวง ตามหัวข้อ 3.2.2 (1) ว่า เส้นรอบวงของตนถูกครอบคลุมโดยโหมตเพื่อนบ้านแบบมีประสิทธิผลครบทั้งเส้นแล้วหรือไม่ ตัวอย่างในรูปที่ 3.7 จะได้ผลลัพธ์การคำนวณออกมาว่า เส้นรอบวงทั้งเส้นของโหมต E อยู่ในพื้นที่ตรวจจับของโหมต A B D และ F ดังนั้นโหมต E จึงผ่านเข้าไปตรวจสอบในขั้นตอนต่อไป (ดูรายละเอียดการคำนวณการครอบคลุมเส้นรอบวงได้ในภาคผนวก ข)

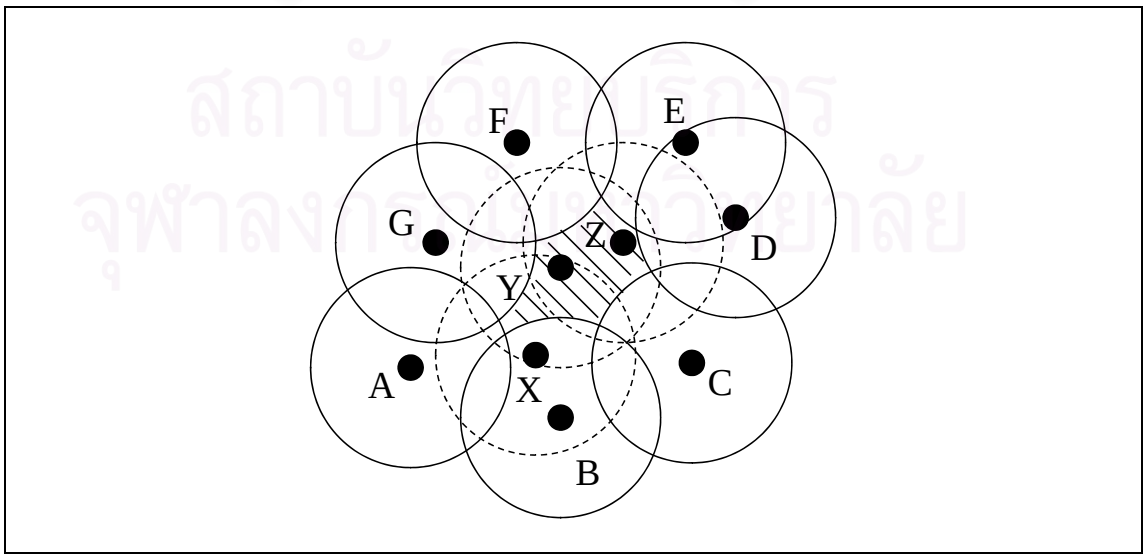
5) เลือกโหมตเพื่อนบ้านแบบมีประสิทธิผลจากทั้งหมดออกมา 1 โหมต โหมตใดก็ได้ จากหัวข้อ 3.2.2 (2) ทำการตรวจสอบดูว่า ส่วนของเส้นรอบวงของโหมตที่เลือกมานั้นถูกครอบคลุมโดยพื้นที่ของโหมตเพื่อนบ้านแบบมีประสิทธิผลตัวอื่นครบทั้งส่วนหรือไม่ ตัวอย่างรูปที่ 3.8 หากโหมตเพื่อนบ้านแบบมีประสิทธิผลที่เลือกมาตรวจสอบคือโหมต D จะได้ว่าส่วนของเส้นรอบวงของโหมต D ที่เชื่อมระหว่างจุดตัด P_1 และ P_2 (แสดงด้วยเส้นทึบหนา) ถูกครอบคลุมโดยโหมต A B และ F หมดทั้งส่วน จึงสรุปได้ว่าโหมต E เป็นโหมตซ้ำซ้อนและสิ้นสุดกระบวนการตรวจสอบ



รูปที่ 3.8 ขั้นตอนการตรวจสอบความซ้ำซ้อน(4)

3.3 การจัดกำหนดการตื่น-หลับของโหมดซ้ำซ้อน

เนื่องจากในระบบที่มีความหนาแน่นของโหมดสูง ทำให้จำนวนโหมดซ้ำซ้อนที่เกิดขึ้นอาจมีได้มากกว่า 1 โหมด โหมดซ้ำซ้อนบางตัวอาจจะอยู่ในตำแหน่งที่ใกล้เคียงกันและมีพื้นที่ตรวจจับทับซ้อนกัน ทำให้ในบางกรณีอาจเกิดพื้นที่บอดขึ้นได้หากมีการปิดหรือหลับโหมดซ้ำซ้อนเหล่านี้ทั้งหมดพร้อมๆกัน ดังเช่นตัวอย่างในรูปที่ 3.9 จะเห็นได้ว่า ทั้งโหมด X Y และ Z เป็นโหมดซ้ำซ้อนทั้งสิ้น แต่ระบบไม่สามารถที่จะหลับโหมดซ้ำซ้อนทั้ง 3 ตัวนี้ได้พร้อมกัน เพราะจะทำให้บริเวณพื้นที่ตรงกลาง (ระบายด้วยเส้นตรงขนาน) มีโอกาสกลายเป็นพื้นที่บอดได้



รูปที่ 3.9 การซ้อนทับพื้นที่ตรวจจับของโหมดซ้ำซ้อนที่อยู่ใกล้กัน

ด้วยสาเหตุดังกล่าวทำให้หลังจากที่ระบบตรวจสอบได้โหมดซ้ำซ้อนออกมากลุ่มหนึ่งแล้วนั้น จำเป็นต้องมีกระบวนการจัดกำหนดการสำหรับการเปลี่ยนสถานะตื่น-หลับของโหมดซ้ำซ้อนเหล่านี้ (Activation scheduling) ด้วย เพื่อรับประกันว่าภายหลังจากที่โหมดซ้ำซ้อนบางส่วนหลับลงไปจะไม่ทำให้ระบบต้องสูญเสียพื้นที่ครอบคลุมเดิมไป

เมื่อพิจารณามองลักษณะของปัญหาการจัดกำหนดการสำหรับการเปลี่ยนสถานะตื่น-หลับของโหมดซ้ำซ้อนนี้จะพบได้ว่า มีรูปแบบของปัญหาที่ใกล้เคียงกันกับปัญหาเซตครอบคลุมเป็นอย่างมาก โดยพื้นที่ครอบคลุมทั้งหมดของระบบจะถูกมองออกเป็นส่วนของพื้นที่ย่อยๆที่เกิดจากการตัดกันของเส้นรอบวงกลมแทนพื้นที่ตรวจจับของโหมด พื้นที่ย่อยๆเหล่านี้จะแทนสมาชิกในเซตเอกภพและโหมดซ้ำซ้อนทั้งหมดจะแทนหมู่ของเซตที่มีสมาชิกในเซตคือพื้นที่ย่อยที่อยู่ภายในพื้นที่ตรวจจับของตน การเลือกตื่นและหลับโหมดซ้ำซ้อนตัวใดในระบบจะเปรียบเสมือนการเลือกและไม่เลือกเซตขึ้นมาเพื่อครอบคลุมสมาชิกในเซตเอกภพตามลำดับ เพราะฉะนั้นในการจัดกำหนดการเปลี่ยนสถานะของโหมดซ้ำซ้อนสำหรับอัลกอริทึมที่นำเสนอนี้จะทำการเทียบปัญหาเข้ากับปัญหาเซตครอบคลุมและนำวิธีการแก้ปัญหาของปัญหาเซตครอบคลุมมาใช้ โดยมีขั้นตอนการทำงานแบ่งออกเป็น 2 ขั้นย่อย ซึ่งมีรายละเอียดดังนี้

1) การเลือกสถานะตื่น-หลับของโหมดซ้ำซ้อน (Active mote selection)

เป็นการค้นหาคำตอบสถานะของโหมดซ้ำซ้อนว่า โหมดตัวใดบ้างที่ถูกตัดสินให้ตื่นมาทำงานและโหมดใดบ้างที่สมควรหลับไป ในส่วนของขั้นตอนนี้จะเริ่มแรกจากการแปลงปัญหาไปเป็นปัญหาเซตครอบคลุมก่อน จากนั้นจึงนำอัลกอริทึมเชิงละโมบแบบถ่วงน้ำหนักที่ใช้ประมาณคำตอบของปัญหาเซตครอบคลุมมาใช้แก้หาคำตอบ

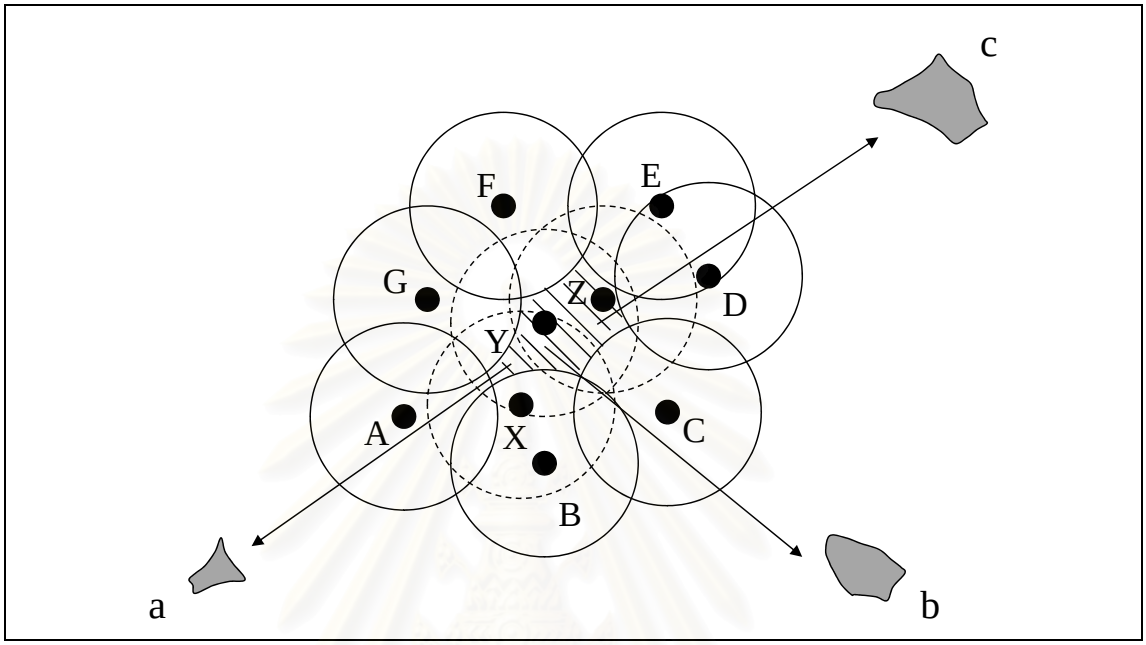
2) เกณฑ์วิธีการลงมติคำตอบ (Voting protocol)

เนื่องจากอัลกอริทึมควบคุมความหนาแน่นที่นำเสนอนี้ถูกออกแบบมาให้มีลักษณะการทำงานแบบเฉพาะที่ ดังนั้นคำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมจากแต่ละโหมดในขั้นตอนแรกอาจได้คำตอบออกมาไม่ตรงกัน เช่น โหมดหนึ่งได้คำตอบตัดสินว่าให้อีกโหมดหนึ่งตื่นมาทำงาน ตรงกันข้ามโหมดอื่นๆกลับได้คำตอบออกมาให้หลับแทน การไม่ลงรอยกันของผลลัพธ์ที่ได้จากหลายๆโหมดทำให้จำเป็นต้องมีเกณฑ์วิธีลงมติคำตอบเข้ามาช่วยในการจัดการตัดสินหาคำตอบสุดท้ายเพื่อเลือกเปลี่ยนสถานะของโหมด

3.3.1 การแปลงปัญหาพื้นที่ครอบคลุมให้เป็นปัญหาเซตครอบคลุม

จากพื้นที่ทั้งหมดที่มีโอกาสเกิดเป็นพื้นที่บอด (พื้นที่ส่วนที่ครอบคลุมโดยโหมดซ้ำซ้อนแต่ไม่ครอบคลุมโดยโหมดไม่ซ้ำซ้อน) จะสามารถแยกออกเป็นพื้นที่ส่วนย่อย (Area section) หลายๆส่วนมาประกอบกันตามการตัดกันของเส้นรอบวงกลมแทนพื้นที่ตรวจจับของ

โหมดซ้ำซ้อน จากตัวอย่างในรูปที่ 3.9 พื้นที่ที่โหมดซ้ำซ้อน X Y และ Z ครอบคลุมและมีโอกาสที่จะเกิดเป็นพื้นที่บอดนั้น (พื้นที่ระบายแรเงา) จะถูกแบ่งออกได้เป็น 3 พื้นที่ส่วนย่อยคือ a b และ c ดังรูปที่ 3.10



รูปที่ 3.10 การแบ่งพื้นที่ส่วนย่อยจากการตัดกันของเส้นรอบวงกลมแทนพื้นที่ตรวจจับ

พื้นที่ส่วนย่อยทั้งหมดที่หาได้จะถูกนำมาใช้แทนข้อมูลนำเข้าของปัญหาเซตครอบคลุม โดยที่พื้นที่ส่วนย่อยแต่ละส่วนจะถูกมองว่าเป็นหนึ่งในสมาชิกของเซตแทนโหมดซ้ำซ้อนที่ครอบคลุมพื้นที่ส่วนนั้นรวมไปถึงเป็นหนึ่งในสมาชิกของเซตเอกภพด้วย จากพื้นที่ส่วนย่อย a b และ c ที่ได้ในรูปที่ 3.10 จะสามารถแปลงเทียบเป็นปัญหาเซตครอบคลุมได้ดังสมการที่ 3.1 3.2 3.3 และ 3.4

$$Universe = \{a, b, c\} \tag{3.1}$$

$$X = \{a, b\} \tag{3.2}$$

$$Y = \{a, b, c\} \tag{3.3}$$

$$Z = \{b, c\} \tag{3.4}$$

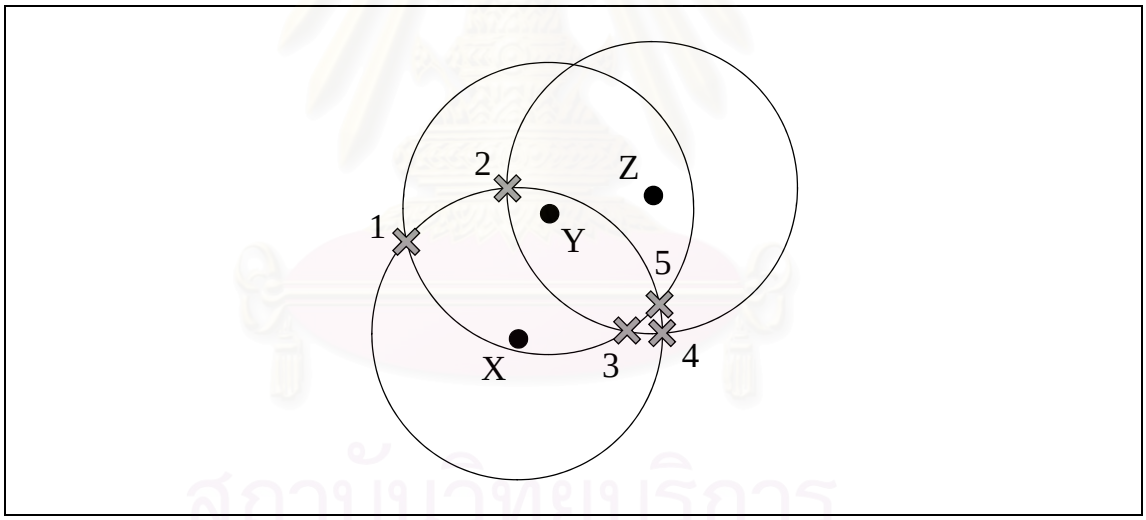
เมื่อได้เซตข้อมูลนำเข้าสำหรับปัญหาเซตครอบคลุมแล้ว ระบบจะสามารถดึงเอาวิธีการแก้ปัญหาสำหรับปัญหาเซตครอบคลุมมาประยุกต์ใช้ในการเลือกโหมดซ้ำซ้อนได้ทันที แต่อย่างไรก็ตามในความเป็นจริงแล้วเนื่องจากตัวโหมดเป็นฮาร์ดแวร์ (Hardware) ที่ไม่สามารถ

มองเห็นภาพการตัดกันของเส้นรอบวงกลมแทนพื้นที่ตรวจจับได้ ทำให้ยากแก่การระบุออกมาได้ว่าโหมตใดปกคลุมพื้นที่ส่วนย่อยใดบ้าง ดังนั้นในการเขียนโปรแกรมเพื่อให้ตัวโหมตทำงานจริงจึงจำเป็นต้องมีกระบวนการที่ใช้ในการค้นหาพื้นที่ส่วนย่อยที่ครอบคลุมสำหรับโหมตใดๆ ด้วย โดยอาศัยเฉพาะข้อมูลนำเข้าจากตำแหน่งของโหมตเพื่อนบ้านมาใช้ในการคำนวณ

3.3.2 การค้นหาพื้นที่ส่วนย่อยภายในบริเวณพื้นที่ตรวจจับของโหมต

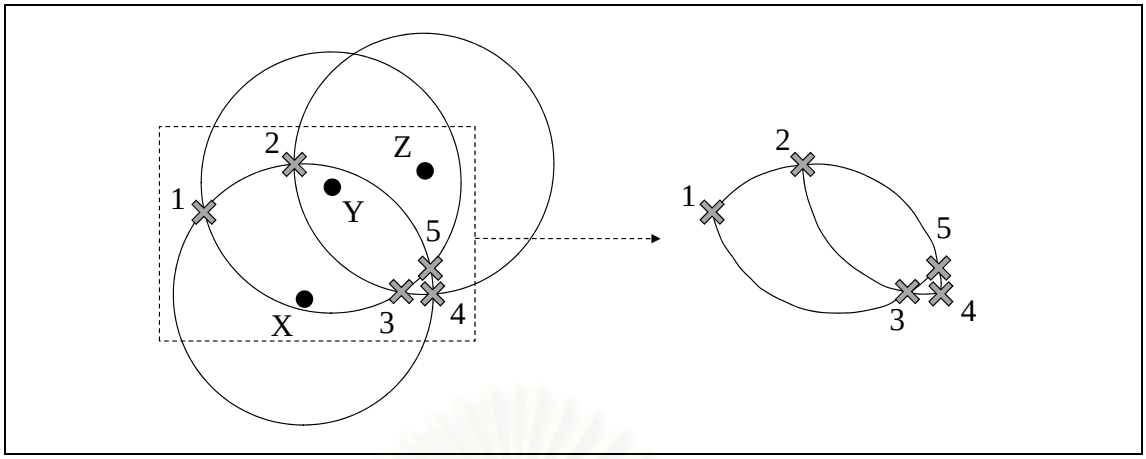
วิทยานิพนธ์นี้ได้นำเสนอกระบวนการในการค้นหาพื้นที่ส่วนย่อยที่ครอบคลุมสำหรับโหมตใดๆ โดยมีขั้นตอนการค้นหาดังนี้

1) จากข้อมูลตำแหน่งของโหมตซ้ำซ้อนเพื่อนบ้านทั้งหมด ทำการเก็บรวบรวมจุดตัดทุกจุดที่เกิดขึ้นจากการตัดกันของเส้นรอบวงกลมและมีตำแหน่งอยู่ภายในพื้นที่ตรวจจับของโหมต รวมทั้งจุดตัดที่อยู่บนเส้นรอบวงด้วย ดังเช่นจากรูปที่ 3.9 เมื่อพิจารณาโหมตซ้ำซ้อน X จะได้ว่าจากข้อมูลตำแหน่งของทั้งโหมตซ้ำซ้อนเพื่อนบ้าน Y และ Z โหมต X จะสามารถหาจุดตัดออกมาได้รวม 5 จุดด้วยกัน ได้แก่ จุด 1 2 3 4 และ 5 ดังรูปที่ 3.11



รูปที่ 3.11 จุดตัดของเส้นรอบวงกลมแทนพื้นที่ตรวจจับสำหรับการค้นหาพื้นที่ส่วนย่อย

2) นำจุดตัดที่ได้ทั้งหมดมาสร้างเป็นกราฟค้นหาพื้นที่ (Area search graph) โดยที่จุดตัดแต่ละจุดจะแทนจุดยอด (Vertex) ในกราฟและส่วนของเส้นรอบวงกลมที่เชื่อมระหว่างจุดตัดจะแทนเส้นเชื่อม (Edge) ระหว่างจุดยอดในกราฟ (จะไม่พิจารณาส่วนเส้นเชื่อมรอบพื้นที่ที่ถูกตรวจจับโดยโหมตตัวเดียว เพราะพื้นที่นั้นไม่มีโอกาสเกิดเป็นพื้นที่บอดจึงไม่จำเป็นต่อการคำนวณ) ในรูปที่ 3.12 แสดงกราฟค้นหาพื้นที่ที่คำนวณได้จากตัวอย่างในรูปที่ 3.11

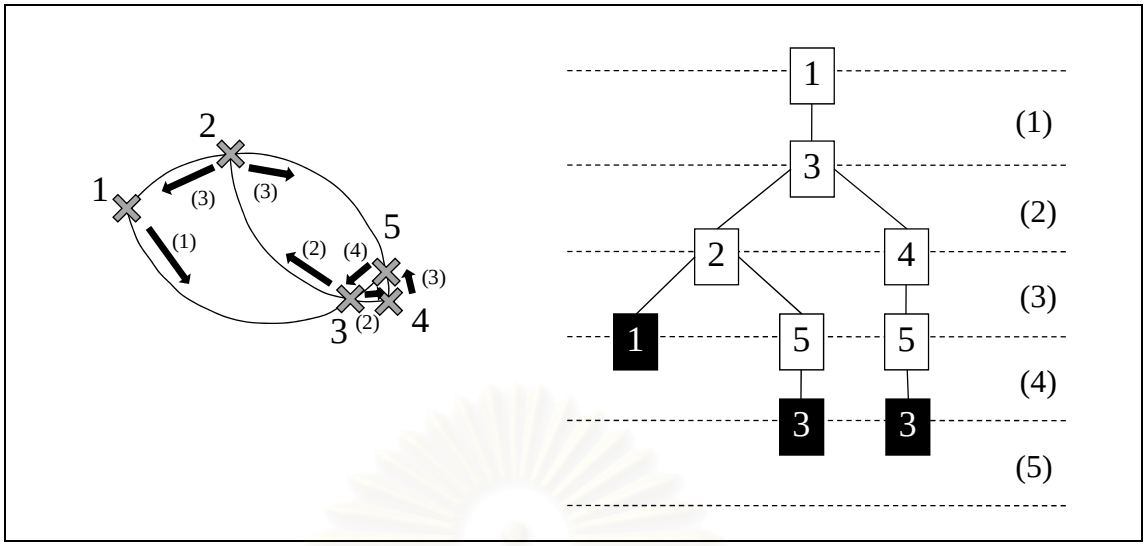


รูปที่ 3.12 กราฟค้นหาพื้นที่

3) จากกราฟค้นหาพื้นที่ ให้เลือกจุดยอดมา 1 จุด จุดใดก็ได้ มาเป็นราก (Root) ของต้นไม้ค้นหาพื้นที่ (Area search tree) จากนั้นเริ่มการไล่ค้นหาพื้นที่โดยหยิบจุดยอดที่เป็นใบ (Leaf node) จากต้นไม้ค้นหาพื้นที่มา 1 จุด (หยิบตามแนวกว้าง (Breadth first search)) จากจุดที่หยิบมานี้ให้ไล่ตามเส้นทางของเส้นเชื่อมในกราฟค้นหาพื้นที่เพื่อหาจุดยอดจุดอื่น โดยมีข้อแม้ว่า จุดยอดใหม่ที่หาได้นั้นจะต้องอยู่บนเส้นรอบวงคนละเส้นกับเส้นรอบวงของเส้นทางเดิมที่ไล่หามาจากจุดยอดก่อนหน้านี้เสมอ จุดยอดใหม่ที่พบจะถูกเพิ่มเข้าไปในต้นไม้ค้นหาพื้นที่ โดยเพิ่มไปในฐานะลูก (Child node) ของจุดยอดก่อนหน้าที่หยิบมา

4) เมื่อไรก็ตามที่พบว่าจุดยอดใหม่ในกราฟที่ไล่เจอนั้นเคยพบไปก่อนหน้านี้แล้ว (เป็นจุดยอดที่มีอยู่ในเส้นทางนับจากระดับรากของต้นไม้ลงมาจนถึงระดับปัจจุบันและอยู่ในระดับถัดขึ้นไปนับจากปัจจุบันไม่เกินจำนวนโหมดซ้ำซ้อน) แสดงว่าเส้นทางที่ค้นหามีการวนมาสิ้นสุดที่จุดซึ่งเคยผ่านไปแล้ว การพบวงวน (Loop) ของเส้นทางนี้แสดงว่ามีการค้นพบพื้นที่ส่วนย่อยใหม่เกิดขึ้นในกราฟ โหมดสามารถที่จะระบุพื้นที่ส่วนย่อยใหม่นี้ได้จากการตามรอยย้อนกลับ (Backtracking) ของเส้นทางจากจุดยอดในต้นไม้ได้ เมื่อพบพื้นที่ส่วนย่อยใหม่แล้วให้ตัดเส้นทางการค้นหาของต้นไม้เส้นนั้นออกจากการพิจารณาได้ทันที จากนั้นกระบวนจะกลับไปที่การหยิบจุดยอดจุดอื่นที่เป็นใบในต้นไม้ออกมาแล้วเริ่มไล่ค้นหาตามวิธีการเดิมไปเรื่อยๆจนกว่าจะไม่มีจุดยอดใดในต้นไม้ให้หยิบมาพิจารณาแล้วเป็นอันจบกระบวนการค้นหาพื้นที่ ในรูปที่ 3.13 แสดงลำดับการไล่ค้นหาตามกราฟและต้นไม้ค้นหาพื้นที่ที่สร้างออกมาได้จากรูปที่ 3.12 โดยมีรายละเอียดและคำอธิบายดังนี้ (ลำดับการไล่ค้นหาจุดยอดจะเป็นไปตามลำดับของเลขในวงเล็บที่กำหนดไว้ในรูป)

4.1) สมมติให้จุด 1 เป็นจุดเริ่มต้นค้นหา โดยถือว่าจุด 1 อยู่บนเส้นรอบวงของโหมด X เพราะฉะนั้นจุดยอดต่อไปที่พบในกราฟจะได้แก่ จุด 3 ซึ่งอยู่บนเส้นรอบวงของโหมด Y ทำการเพิ่มจุด 3 เข้าไปเป็นลูกของจุด 1 ในต้นไม้



รูปที่ 3.13 ลำดับการค้นหาและต้นไม้ค้นหาพื้นที่

4.2) จากจุด 3 ซึ่งอยู่บนเส้นรอบวงของโหมต Y จะได้ว่าจุดต่อไปที่พบในกราฟได้แก่ จุด 2 และจุด 4 ซึ่งอยู่บนเส้นรอบวงของโหมต Z ทำการเพิ่มจุด 2 และจุด 4 เข้าไปเป็นลูกของจุด 3 ในต้นไม้

4.3) จากจุด 2 ซึ่งอยู่บนเส้นรอบวงของโหมต Z จะได้ว่าจุดต่อไปที่พบในกราฟได้แก่ จุด 1 และจุด 5 ซึ่งอยู่บนเส้นรอบวงของโหมต X ทำการเพิ่มจุด 1 และจุด 5 เข้าไปเป็นลูกของจุด 2 ในต้นไม้

จากจุด 4 ซึ่งอยู่บนเส้นรอบวงของโหมต Z จะได้ว่าจุดต่อไปที่พบในกราฟได้แก่ จุด 5 ซึ่งอยู่บนเส้นรอบวงของโหมต X ทำการเพิ่มจุด 5 เข้าไปเป็นลูกของจุด 4 ในต้นไม้

4.4) จุด 1 เป็นจุดที่อยู่บนต้นไม้ถัดขึ้นไป 3 ระดับและอยู่บนเส้นทางย้อนกลับจากจุด 1 ปัจจุบันด้วย ดังนั้นจะได้พื้นที่ใหม่ออกมาคือพื้นที่ที่ครอบด้วยเส้นทาง (1 → 2 → 3 → 1)

จากจุด 5 ซึ่งอยู่บนเส้นรอบวงของโหมต X จะได้ว่าจุดต่อไปที่พบในกราฟได้แก่ จุด 3 ซึ่งอยู่บนเส้นรอบวงของโหมต Y ทำการเพิ่มจุด 3 เข้าไปเป็นลูกของจุด 5 ในต้นไม้

จากจุด 5 ซึ่งอยู่บนเส้นรอบวงของโหมต X จะได้ว่าจุดต่อไปที่พบในกราฟได้แก่ จุด 3 ซึ่งอยู่บนเส้นรอบวงของโหมต Y ทำการเพิ่มจุด 3 เข้าไปเป็นลูกของจุด 5 ในต้นไม้

4.5) จุด 3 เป็นจุดที่อยู่บนต้นไม้ถัดขึ้นไป 3 ระดับและอยู่บนเส้นทางย้อนกลับจากจุด 3 ปัจจุบันด้วย ดังนั้นจะได้พื้นที่ใหม่ออกมาคือพื้นที่ที่ครอบด้วยเส้นทาง (3 → 5 → 2 → 3)

จุด 3 เป็นจุดที่อยู่บนต้นไม้ถัดขึ้นไป 3 ระดับและอยู่บนเส้นทางย้อนกลับจากจุด 3 ปัจจุบันด้วย ดังนั้นจะได้พื้นที่ใหม่ออกมาคือพื้นที่ที่ครอบด้วยเส้นทาง (3 → 5 → 4 → 3)

5) โหมต X จะได้คำตอบพื้นที่ทั้งหมดที่ได้จากการค้นหาทั้งหมด 3 พื้นที่ ได้แก่ พื้นที่ที่ครอบด้วยเส้นทาง (1 → 2 → 3 → 1) (3 → 5 → 2 → 3) และ (3 → 5 → 4 → 3) ซึ่งเมื่อเปรียบเทียบกันกับพื้นที่ในรูปที่ 3.10 จะได้ว่า พื้นที่ที่ครอบด้วยเส้นทาง (1 → 2 → 3 → 1) (3 → 5 → 2 → 3) และ (3 → 5 → 4 → 3) สามารถแทนพื้นที่ย่อย a b และส่วนเกิน d ได้พอดี หรือเขียนเป็นเซตจะได้ว่า X = {a, b, d}

การค้นหาพื้นที่จะสิ้นสุดลงเมื่อพบว่าจุดยอดที่ใบของต้นไม้ทุกจุดมีการซ้ำกับจุดยอดบนต้นไม้ในระดับถัดขึ้นไปหมดแล้ว ซึ่งจากการที่เส้นเชื่อมที่เกิดจากเส้นรอบวงในกราฟนั้นมีจำนวนที่จำกัด ดังนั้นกระบวนการค้นหาพื้นที่ส่วนย่อยนี้จึงสิ้นสุดการทำงานเมื่อได้คำตอบของพื้นที่ส่วนย่อยครบทุกพื้นที่เสมอ อย่างไรก็ตามวิธีการนี้อาจยังมีค่าผิดพลาดจากพื้นที่ส่วนย่อยที่ได้เกินออกมามากกว่าความจำเป็นใช้งานจริงอยู่บ้างเล็กน้อย แต่เนื่องด้วยอัลกอริทึมนี้ไม่ใช่การหาคำตอบที่ดีที่สุด ดังนั้นค่าผิดพลาดที่เกิดขึ้นจึงถือได้ว่าอยู่ในเกณฑ์ที่รับได้

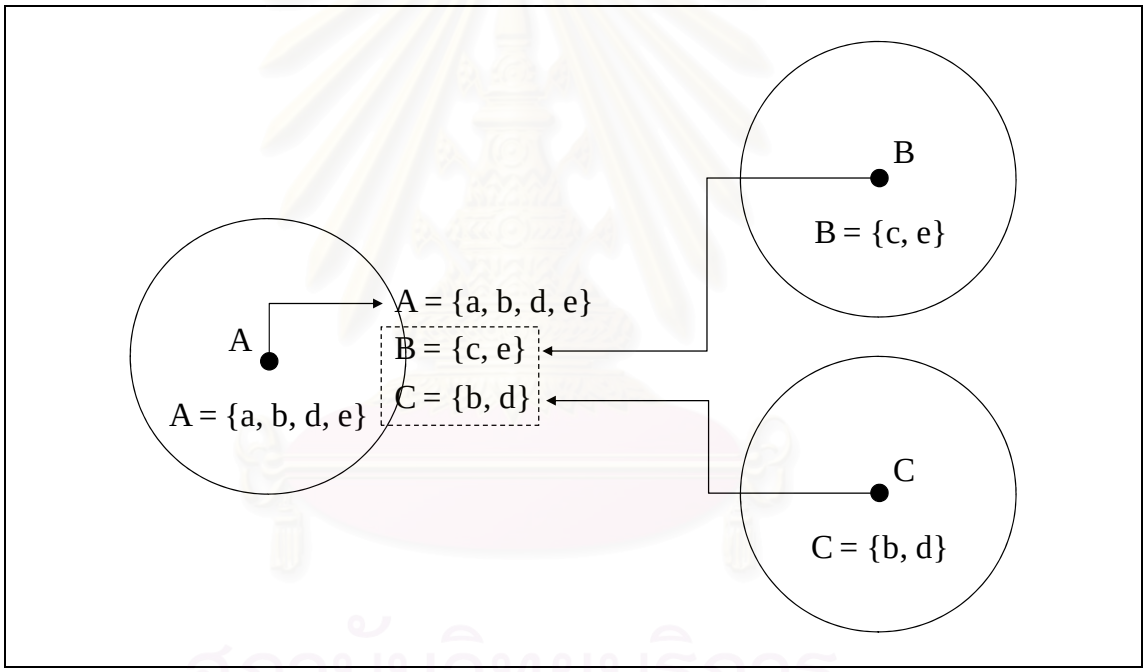
ข้อสังเกตอย่างหนึ่งคือ พื้นที่บอดจะกินอาณาบริเวณอยู่ในส่วนพื้นที่ที่ได้รับการตรวจจับจากโหมตซ้ำซ้อนมากกว่า 1 โหมตเท่านั้น ดังเช่นตัวอย่างในรูปที่ 3.11 พื้นที่ย่อย (1 → 3 → 4 → 1) เป็นพื้นที่ที่ถูกตรวจจับโดยโหมต X เท่านั้น (เทียบกันภายในกลุ่มของโหมตซ้ำซ้อนด้วยกัน) ถ้าเกิดมีพื้นที่บอดกินอาณาบริเวณอยู่ในส่วนพื้นที่นี้ แสดงว่าพื้นที่บอดส่วนนั้นจะถูกครอบคลุมโดยโหมต X เพียงโหมตเดียวเท่านั้น (เทียบกับทั้งระบบ) ซึ่งจะกลายเป็นว่าโหมต X ไม่ใช่โหมตซ้ำซ้อน ซึ่งขัดแย้งกันกับความจริงที่ว่าโหมต X เป็นโหมตซ้ำซ้อน ดังนั้นจึงพิสูจน์ได้ว่าพื้นที่บอดจะไม่มีโอกาสเกิดขึ้นนอกบริเวณพื้นที่ที่ได้รับการตรวจจับจากโหมตซ้ำซ้อนมากกว่า 1 โหมต ระบบสามารถตัดพื้นที่ส่วนย่อยที่ได้รับการตรวจจับจากโหมตซ้ำซ้อนเพียงตัวเดียวออกจากการพิจารณาได้

3.3.3 การเลือกสถานะตื่น-หลับของโหมตซ้ำซ้อน

หลังจากที่โหมตซ้ำซ้อนได้เซตของพื้นที่ส่วนย่อยทั้งหมดออกมาแล้ว โหมตซ้ำซ้อนเหล่านี้จะเริ่มกระบวนการในการเลือกคำตอบว่าจะตัดสินให้โหมตใดตื่นหรือหลับ โดยอาศัยอัลกอริทึมแบบประมาณที่ใช้ในการแก้ปัญหาเซตครอบคลุมที่เรียกว่า อัลกอริทึมเชิงละโมบ เข้ามาช่วย คำตอบที่ได้จากอัลกอริทึมเชิงละโมบจะเป็นเซตคำตอบของโหมตที่ถูกเลือกให้ตื่นมา

ทำงาน ซึ่งตามหลักการทั่วไปของอัลกอริทึมเชิงละโมบแล้วโหมดที่ถูกเลือกเป็นอันดับแรกจะต้องเป็นโหมดที่มีเซตของพื้นที่ส่วนย่อยขนาดใหญ่ที่สุดก่อนเสมอ (เลือกโหมดที่สามารถครอบคลุมพื้นที่ได้กว้างสุดก่อน) จากตัวอย่างในรูปที่ 3.11 คำตอบที่ได้จากอัลกอริทึมเชิงละโมบจะเลือกตื่นโหมดซ้ำซ้อน Y ก่อนอันดับแรก เนื่องจากมีขนาดของเซตพื้นที่ส่วนย่อยครอบคลุมได้มากกว่าทั้งของโหมด X และ Z

การใช้งานอัลกอริทึมเชิงละโมบจำเป็นต้องอาศัยข้อมูลนำเข้าได้แก่เซตพื้นที่ส่วนย่อยจากทั้งของตัวเองที่หาได้และจากโหมดซ้ำซ้อนเพื่อนบ้านโหมดอื่นๆด้วย ดังนั้นในช่วงแรกของกระบวนการในการเลือกคำตอบจะต้องมีการแลกเปลี่ยนข้อมูลเซตพื้นที่ส่วนย่อยระหว่างโหมดซึ่งกันและกันก่อนเสมอ ดังเช่นตัวอย่างในรูปที่ 3.14 สมมติให้โหมดซ้ำซ้อน A B และ C มีเซตของพื้นที่ส่วนย่อยเป็น {a, b, d, e} {c, e} และ {b, d} ตามลำดับ



รูปที่ 3.14 การแลกเปลี่ยนเซตพื้นที่ระหว่างโหมด

โหมด A จะรับเซตจากทางโหมด B และ C เข้ามา เพื่อนำมาใช้เป็นข้อมูลนำเข้าสำหรับการแก้ปัญหาเซตครอบคลุมตามสมการที่ 3.5 3.6 3.7 และ 3.8 จากนั้นทำการแก้ปัญหาโดยใช้อัลกอริทึมเชิงละโมบเพื่อตัดสินเลือกโหมดซ้ำซ้อน (เลือกเซตของพื้นที่ส่วนย่อย) เข้าไปอยู่ในเซตคำตอบ (เซตของโหมดซ้ำซ้อนที่ตื่นมาทำงาน) ทีละโหมดๆ โดยแต่ละครั้งที่มีการเลือกจะตรวจสอบด้วยว่า ยังเหลือพื้นที่ส่วนย่อยในเซตเอกภพที่ยังไม่ถูกครอบคลุมโดยโหมดซ้ำซ้อนที่เป็นสมาชิกในเซตคำตอบหรือไม่ ถ้ายังเหลือก็ให้ทำการเลือกโหมดซ้ำซ้อนตัวอื่นเข้ามาอยู่ในเซตคำตอบไปเรื่อยๆจนกว่าทุกพื้นที่ส่วนย่อยในเซตเอกภพจะถูกครอบคลุมหมด จากตัวอย่าง

ในรูปที่ 3.14 ผลการทำงานของอัลกอริทึมเชิงละโมบจะได้ว่า “โหมต A และ B เป็นโหมตซ้ำซ้อนที่ควรจะตื่นมาทำงาน ส่วนโหมต C ไม่จำเป็นต้องทำงานและสามารถหลับไปได้”

$$Universe = \{a, b, c, d, e\} \tag{3.5}$$

$$A = \{a, b, d, e\} \tag{3.6}$$

$$B = \{c, e\} \tag{3.7}$$

$$C = \{b, d\} \tag{3.8}$$

อย่างไรก็ตามการตัดสินคำตอบในการเลือกตื่นโหมตโดยอาศัยน้ำหนักจากขนาดของเซตพื้นที่ส่วนย่อยที่ครอบคลุมเพียงอย่างเดียวนั้นอาจทำให้ผลออกมาไม่ดีนัก เนื่องจากปัจจัยด้านพลังงานที่เหลืออยู่ของโหมตซึ่งถือได้ว่าเป็นอีกปัจจัยหนึ่งที่สำคัญนั้นไม่ได้ถูกหยิบขึ้นมาพิจารณาในการคำนวณ โหมตแต่ละโหมตมีพลังงานไฟฟ้าหลงเหลือพอให้ใช้งานอยู่ไม่เท่ากันตามเวลาที่ผ่านไป หากโหมตใดที่ถูกเลือกให้ตื่นขึ้นมาทำงานบ่อยจะทำให้โหมตนั้นมีอัตราการสูญเสียพลังงานที่มากกว่าโหมตอื่นทำให้อาจตายไปจากระบบได้ เพราะฉะนั้นในบางครั้งการผลัดเปลี่ยนโหมตอื่นๆที่มีพลังงานเหลือมากกว่าขึ้นมาทำงานทดแทนบ้างในบางครั้งจะส่งผลดีต่ออัตราการสิ้นเปลืองพลังงานระบบได้มากกว่าการเลือกโดยอาศัยพื้นที่ครอบคลุมเพียงอย่างเดียว อัลกอริทึมเชิงละโมบที่นำมาใช้ในการเลือกคำตอบนี้จึงได้มีการถ่วงน้ำหนัก (Weight) ทางด้านปริมาณพลังงานที่เหลืออยู่ของโหมตเสริมเข้าไป โดยที่มีฟังก์ชันที่ใช้เป็นกฎในการตัดสินใจว่าเป็นฟังก์ชันค่า *Cost – Effectiveness* ของโหมต ซึ่งแสดงได้ดังสมการที่ 3.9

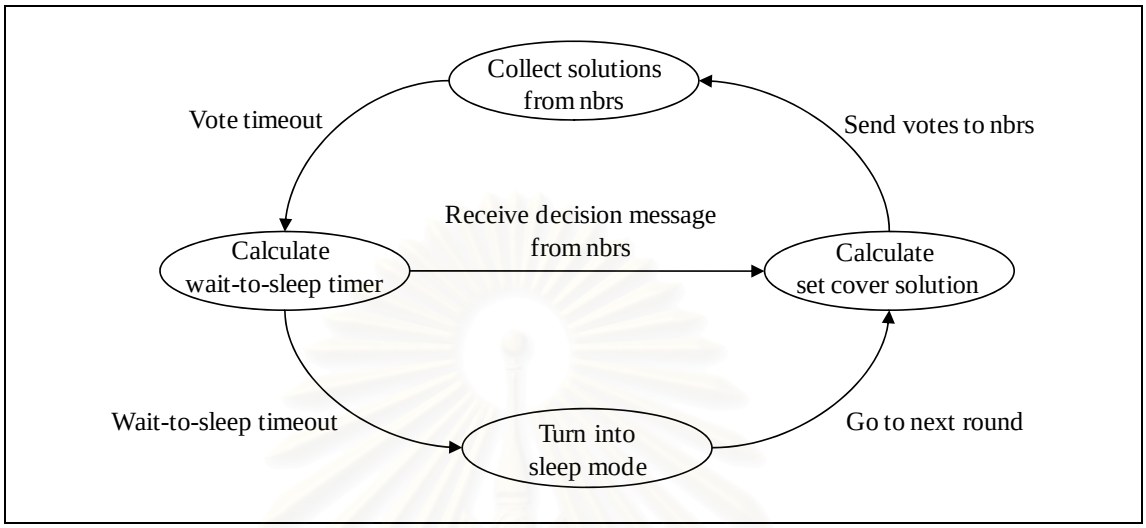
$$Cost - Effectiveness = \mathrm{Re}\,maining\ energy\ x\ Number\ of\ un\mathrm{cov}ered\ area \tag{3.9}$$

จากสมการค่า *Cost – Effectiveness* จะได้ว่า โหมตใดที่เหลือพลังงานใช้งานอยู่มากและมีเซตของพื้นที่ส่วนย่อยครอบคลุมขนาดใหญ่ด้วยแล้ว ค่า *Cost – Effectiveness* จะมีค่าสูงและเหมาะที่จะถูกเลือกขึ้นมาทำงานก่อน ตรงกันข้ามโหมตใดเหลือพลังงานน้อยและครอบคลุมพื้นที่ไม่มากจะมีค่า *Cost – Effectiveness* ที่ต่ำกว่า ดังนั้นในการเลือกคำตอบของอัลกอริทึมเชิงละโมบแบบถ่วงน้ำหนักนี้แต่ละครั้งจะเลือกเปิดโหมตที่มีค่าของ *Cost – Effectiveness* มากที่สุดก่อนเสมอ

3.3.4 เกณฑ์วิธีลงมติคำตอบ

จากการที่คำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมของแต่ละโหมตไม่ลงรอยกันนั้น ทำให้จำเป็นต้องมีกระบวนการในการตัดสินคำตอบสุดท้ายสำหรับโหมตซ้ำซ้อนแต่ละตัวด้วย เกณฑ์วิธีลงมติถือเป็นวิธีการแบบหนึ่งที่ใช้สำหรับการตัดสินหาคำตอบสุดท้ายโดยที่

รายละเอียดของเกณฑ์วิธีลงมติคำตอบนั้นจะมีการวนสถานะของโหมดตามแผนภูมิแสดงการเปลี่ยนแปลงสถานะ (State diagram) ดังรูปที่ 3.15



รูปที่ 3.15 การเปลี่ยนแปลงสถานะของโหมดตามเกณฑ์วิธีลงมติคำตอบ

สถานะแรกเริ่มต้นของโหมดหลังจากที่ได้คำตอบจากการแก้ปัญหาเซตครอบคลุมมาแล้วนั้นจะอยู่ที่สถานะ Calculate set cover solution ในสถานะนี้โหมดแต่ละโหมดจะทำการส่งมติ (Vote) ที่แสดงคำตอบที่หามาได้จากการแก้ปัญหาเซตครอบคลุมให้กับทางโหมดเพื่อนบ้านทุกโหมดพร้อมกันนั้นก็เปลี่ยนสถานะไปสู่สถานะ Collect solutions from nbrs เพื่อรอรับมติจากทางโหมดเพื่อนบ้านด้วยเช่นกัน มติที่ได้จะมี 2 ประเภทคือ มติที่ลงความเห็นให้ตื่น (ACTIVE vote) เมื่อคำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมออกมาว่าให้โหมดนั้นๆตื่น และมติที่ลงความเห็นให้หลับ (INACTIVE vote) เมื่อคำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมออกมาว่าให้โหมดนั้นๆหลับ จากนั้นเมื่อเวลาผ่านไปช่วงหนึ่งให้ทำการปิดรับการลงมติจากโหมดเพื่อนบ้านและเปลี่ยนสถานะไปยังสถานะ Calculate wait-to-sleep timer พร้อมกับนำมติทั้งหมดที่ได้รับมาสรุปผลการลงมติ

ผลการลงมติจะสรุปออกมาในรูปแบบของอัตราส่วนหรือร้อยละของจำนวนมติที่แสดงคำตอบให้โหมดที่พิจารณานี้หลับ (Percentage of INACTIVE votes) โดยผลร้อยละที่ได้นี้จะถูกนำไปใช้ในการตั้งค่าตัวจับเวลา wait-to-sleep เพื่อรอการเปลี่ยนสถานะอีกทีหนึ่ง หลักการตั้งค่าตัวจับเวลานี้จะมีว่า “โหมดใดที่ได้รับการลงมติให้หลับสูง จะถูกตั้งเวลารอการเปลี่ยนสถานะที่สั้น หากโหมดใดที่ได้รับการลงมติให้หลับต่ำ จะถูกตั้งเวลารอการเปลี่ยนสถานะที่นานกว่า” หลักการตั้งเวลาแบบนี้จะเป็นการเปิดโอกาสให้แก่โหมดที่ได้รับการลงมติให้หลับเป็นจำนวนมากได้ทำการหลับก่อน

หากพบว่าภายในช่วงเวลาที่กำลังรอนี้ได้รับข้อความแจ้งการหลับ (Deactivation message) มาจากโหมดเพื่อนบ้านตัวใด แสดงว่ามีการเปลี่ยนแปลงขึ้นในระบบโดยมีโหมดอื่นตัดสินใจหลับไปก่อนหน้านี้แล้ว ซึ่งทำให้สถานะปัจจุบันของโหมดอาจมีการเปลี่ยนแปลงไปแล้วจากเดิม (การหลับโหมดอื่นไปอาจทำให้โหมดเดิมเคยซ้ำซ้อนกลายเป็นไม่ซ้ำซ้อนแทน) ดังนั้นโหมดที่ได้รับข้อความแจ้งการหลับจากโหมดอื่นจะทำการเปลี่ยนสถานะกลับไปยังสถานะ Calculate set cover solution ใหม่อีกครั้งพร้อมกับปรับชุดข้อมูลของโหมดเพื่อนบ้านเดิมที่มีอยู่ให้เป็นปัจจุบัน (Update) เพื่อทำการคำนวณสถานะซ้ำซ้อนและแก้ปัญหาเซตครอบคลุมใหม่ (Recalculate) จากนั้นจึงเข้าสู่กระบวนการลงมติใหม่อีกครั้งหนึ่ง โหมดใดที่คำนวณใหม่แล้วพบว่าตัวเองไม่ใช่โหมดซ้ำซ้อนอีกต่อไปแล้วให้ทำการตื่นตัวเองขึ้นมาทำงานทันที

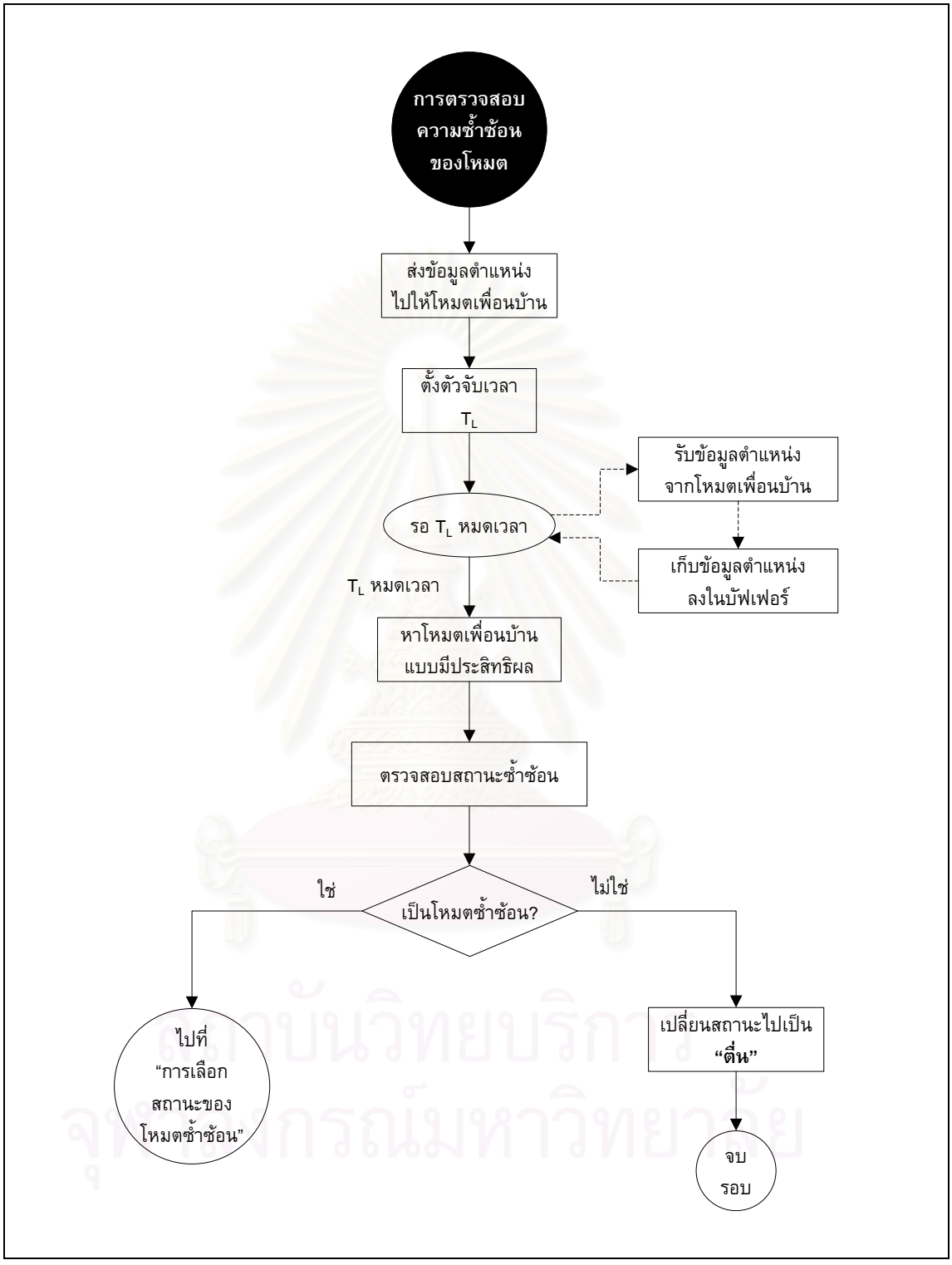
ตรงกันข้ามหากพบว่าภายในช่วงเวลาที่กำลังรอนี้ไม่ได้รับข้อความแจ้งการหลับจากโหมดเพื่อนบ้านใดๆเลย โหมดจะสามารถทำการเปลี่ยนสถานะของตัวเองไปเป็นสถานะ Turn into sleep mode พร้อมกับหลับตัวเองเสีย (sleep) เมื่อเวลาผ่านไปช่วงระยะหนึ่งจึงค่อยทำการปรับข้อมูลพลังงานที่เหลืออยู่ของโหมดแล้วเริ่มกระบวนการทั้งหมดในรอบถัดไปใหม่เช่นนี้เรื่อยๆไป

3.4 การรับประกันการรักษาคุณสมบัติพื้นที่ครอบคลุมภายหลังการหลับโหมด

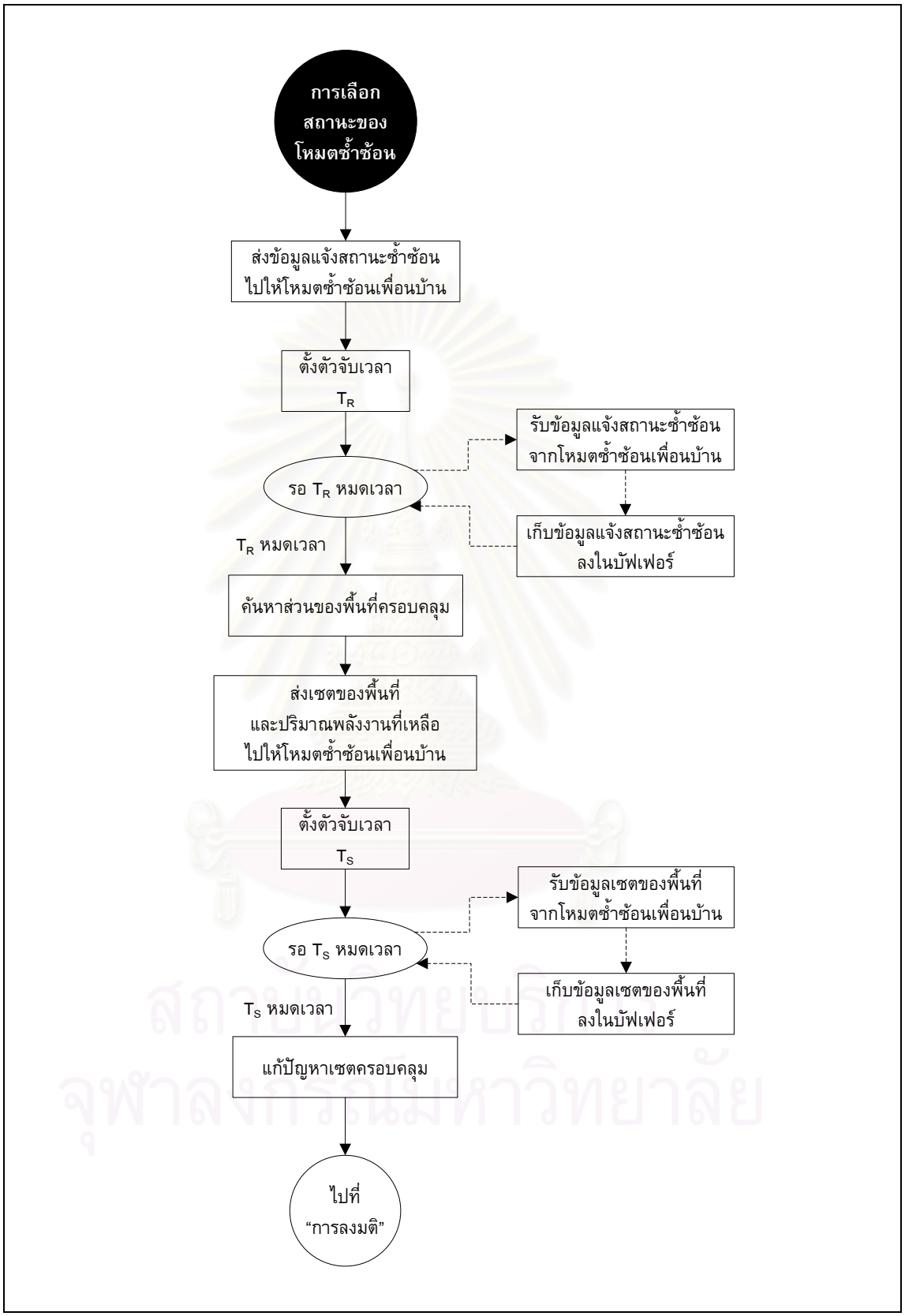
เนื่องด้วยการที่โหมดมีการส่งข้อความแจ้งเตือนแก่โหมดอื่นก่อนที่จะทำการตัดสินใจหลับตัวเองลงไปนั้น ทำให้โหมดอื่นๆสามารถที่จะรับรู้ถึงสถาณภาพการมีอยู่ของโหมดปัจจุบันบริเวณรอบๆตัวมันได้ ส่งผลให้การคำนวณความซ้ำซ้อนและหาคำตอบสำหรับการหลับโหมดใหม่ในแต่ละครั้งที่มีโหมดบางตัวหลับไปก่อนหน้าจะตั้งอยู่บนเซตข้อมูลนำเข้าสำหรับเซตปัญหาที่เป็นปัจจุบันอยู่เสมอ (มีการตัดเอาส่วนพื้นที่เดิมของโหมดที่หลับออกจากเซตข้อมูลนำเข้าและไม่นำไปใช้การพิจารณาในการคำนวณใหม่) การหลับโหมดใดๆแต่ละครั้งจะไม่มีโอกาสที่จะทำให้เกิดพื้นที่บอดขึ้นในพื้นที่เฝ้าสังเกตแน่นอน เพราะฉะนั้นจึงเป็นการพิสูจน์ว่าระบบจะสามารถรับประกันการรักษาคุณสมบัติพื้นที่ครอบคลุมภายหลังการทำงานของอัลกอริทึมเอาไว้ได้

3.5 ขั้นตอนการทำงานของอัลกอริทึม

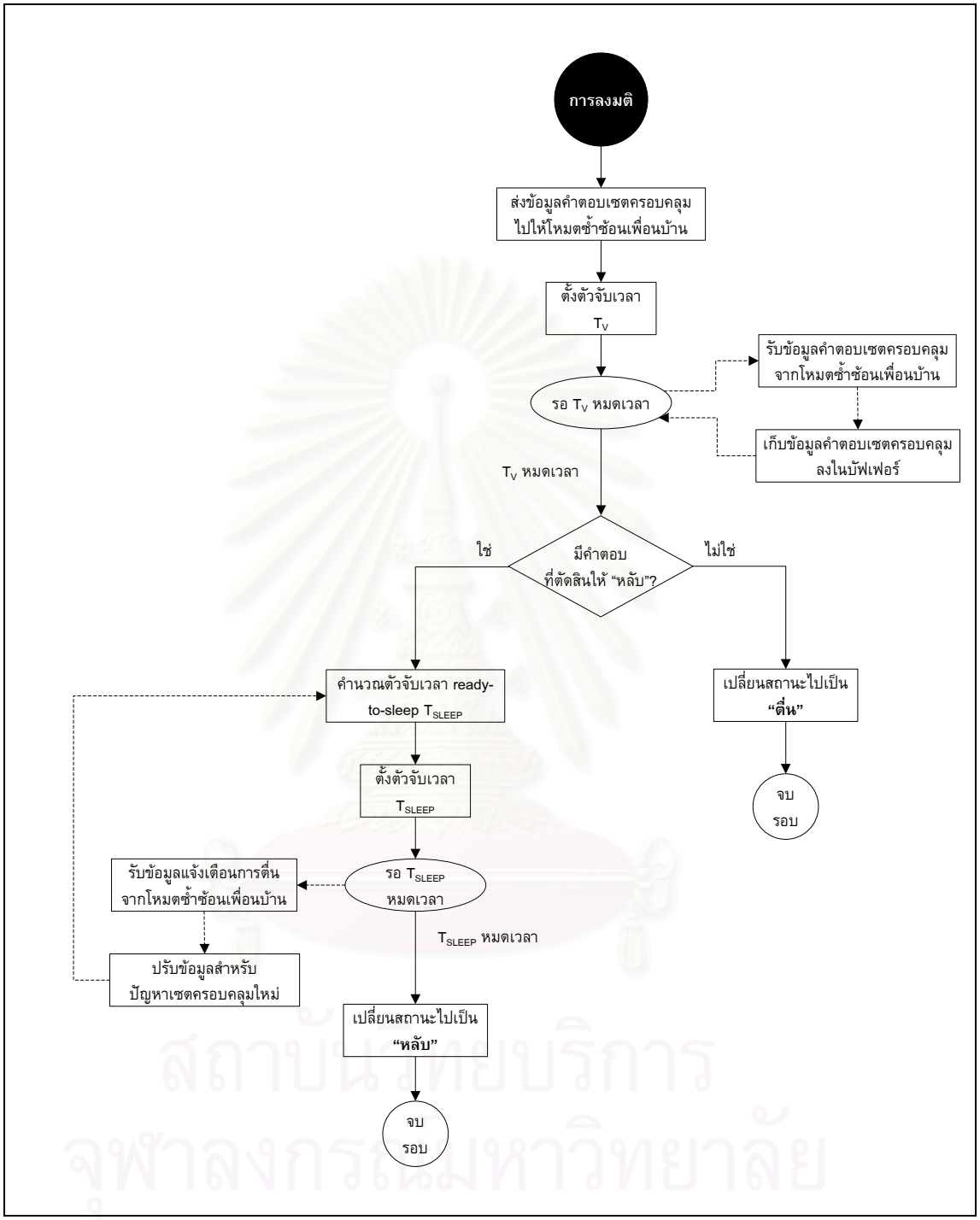
ภาพรวมขั้นตอนการทำงานของอัลกอริทึมจะแบ่งออกเป็น 3 ส่วนย่อย ได้แก่ ส่วนแรกคือการตรวจสอบความซ้ำซ้อนของโหมด รายละเอียดขั้นตอนย่อยจะแสดงไว้ดังแผนภาพในรูปที่ 3.16 ส่วนที่สองเป็นการแก้ปัญหาเซตครอบคลุมเพื่อเลือกโหมดให้ตื่นมาทำงาน จะแสดงแผนภาพขั้นตอนการทำงานได้ดังรูปที่ 3.17 และส่วนสุดท้ายเป็นการลงมติเลือกคำตอบของโหมด จะแสดงแผนภาพขั้นตอนการทำงานได้ดังรูปที่ 3.18



รูปที่ 3.16 แผนภาพแสดงขั้นตอนการทำงานของอัลกอริทึมควบคุมความหนาแน่น(1)



รูปที่ 3.17 แผนภาพแสดงขั้นตอนการทำงานของอัลกอริทึมควบคุมความหนาแน่น(2)



รูปที่ 3.18 แผนภาพแสดงขั้นตอนการทำงานของอัลกอริทึมควบคุมความหนาแน่น(3)

บทที่ 4

ผลการทดลองและวิเคราะห์ผล

เนื่องด้วยในปัจจุบันตัวอุปกรณ์โหมดที่ใช้ทำการทดลองมีราคาจำหน่ายที่ค่อนข้างสูงพอสมควร อีกทั้งในงานวิจัยนี้มีความจำเป็นที่จะต้องใช้โหมดจำนวนมากในการทดสอบการทำงานของอัลกอริทึม ดังนั้นการวัดผลจะกระทำการทดลองบนโปรแกรมจำลอง (Simulator) ที่ชื่อว่าทอสซิม (TOSSIM) [6] แทนการใช้อุปกรณ์จริง โปรแกรมจำลองทอสซิมนี้สามารถจำลองการทำงานของระบบปฏิบัติการไทนีโอเอส (TinyOS) [14] ที่ใช้ทำงานบนแพลตฟอร์มตัวโหมดจริงได้ โปรแกรมประยุกต์ที่ใช้ในการทดสอบจะเขียนขึ้นด้วยภาษาโปรแกรมเนสซี (nesC) [15] ซึ่งเป็นภาษาเดียวกันกับที่ใช้เขียนโปรแกรมเพื่อทำงานบนระบบไทนีโอเอส ดังนั้นในทางปฏิบัติแล้วตัวโปรแกรมทดสอบการทำงานของอัลกอริทึมนี้จะสามารถนำไปใช้งานในอุปกรณ์โหมดรวมไปถึงระบบบนสภาพแวดล้อมจริงได้

การทดสอบประสิทธิภาพการทำงานของอัลกอริทึมตลอดงานวิจัยนี้จะกระทำการจำลองสถานการณ์การสุ่มวางตำแหน่งของโหมดในระบบ โดยมีการกำหนดขนาดของพื้นที่เฝ้าสังเกต (Monitoring area) ที่ทำการทดลองไว้คงที่เท่ากับ 100x100 ตารางหน่วย (หน่วยในที่นี้จะเป็นหน่วยที่ได้จากโปรแกรมจำลองไทนีวิสซึ่งใช้ในการสุ่มตำแหน่ง) โหมดแต่ละโหมดจะมีรัศมีตรวจจับและรัศมีติดต่อสื่อสารคงที่เท่ากับ 10 และ 20 หน่วยตามลำดับ และระบบจะกำหนดให้ไม่มีการสูญเสียข้อมูลในการรับ-ส่งข้อมูลระหว่างโหมดแต่อย่างใด

อัลกอริทึมควบคุมความหนาแน่นที่นำมาทดสอบจะมีทั้งสิ้น 5 อัลกอริทึม โดยที่ 2 อัลกอริทึมแรกจะเป็นอัลกอริทึมจากงานวิจัยก่อนหน้าที่นำมาเปรียบเทียบผล ส่วน 2 อัลกอริทึมหลังจะเป็นอัลกอริทึมใหม่ที่นำเสนอในงานวิจัยนี้ อัลกอริทึมทั้ง 5 แบบจะมีรายละเอียดการทำงานแตกต่างกันดังนี้

1) อัลกอริทึมควบคุมความหนาแน่น PEAS

   มีหลักการทำงานตาม [12]

2) อัลกอริทึมควบคุมความหนาแน่นแบบอาศัยตัวจับเวลาแบบสุ่ม

   มีหลักการทำงานตาม [4]

3) อัลกอริทึมควบคุมความหนาแน่นแบบอาศัยตัวจับเวลาแบบจำกัดขอบเขตสุ่ม

   มีหลักการทำงานเช่นเดียวกับอัลกอริทึมควบคุมความหนาแน่นแบบอาศัยตัวจับเวลาแบบสุ่ม แต่เพิ่มเติมในส่วนการจำกัดขอบเขตของการสุ่มเวลา โดยใช้

ปริมาณพลังงานที่เหลืออยู่ของโหมดเป็นตัวจำกัดขอบเขต โหมดที่มีปริมาณพลังงานเหลืออยู่มาก จะมีขอบเขตการสุ่มเวลาสำหรับตั้งตัวจับเวลาก่อนหลับนานกว่าโหมดที่มีปริมาณพลังงานเหลืออยู่น้อย

4) อัลกอริทึมควบคุมความหนาแน่นแบบอาศัยเซตครอบคลุม

ใช้คำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมแบบถ่วงน้ำหนักของตัวโหมดเองเท่านั้นในการตัดสินใจตื่น-หลับ และใช้ปริมาณพลังงานที่เหลืออยู่ของโหมดในการตั้งตัวจับเวลาก่อนหลับ

5) อัลกอริทึมควบคุมความหนาแน่นแบบอาศัยเซตครอบคลุมผสมการลงมติ

ใช้ผลการลงมติคำตอบที่ได้จากการแก้ปัญหาเซตครอบคลุมแบบถ่วงน้ำหนักของโหมด (เพื่อนบ้าน) หลายๆโหมดในการตัดสินใจตื่น-หลับ และใช้อัตราส่วนของมติที่บอกให้หลับต่อมติที่บอกให้ตื่นในการตั้งตัวจับเวลาก่อนหลับ

ในหัวข้อที่ 4.1 จะกล่าวถึงรายละเอียดเครื่องมือต่างๆที่ใช้ในการทดลอง จากนั้นในหัวข้อที่ 4.2 4.3 4.4 และ 4.5 จะเป็นผลการทดลองเปรียบเทียบจำนวนโหมดที่หลับได้เฉลี่ยต่อจำนวนโหมดทั้งหมด ปริมาณพลังงานรวมที่ระบบใช้จากการทำงานของอัลกอริทึม ณ ขณะเวลาใด (แสดงการลดปริมาณพลังงานที่ใช้) อายุการทำงานเฉลี่ยของโหมด และปริมาณชุดข้อมูลที่มีการรับส่งทั้งหมดตามลำดับ

4.1 เครื่องมือที่ใช้ในการทดลอง

4.1.1 ระบบปฏิบัติการณ์ไทนีโอเอส (TinyOS)

ไทนีโอเอส (TinyOS) [14] เป็นชื่อของระบบปฏิบัติการหนึ่งที่ถูกออกแบบมาเพื่อใช้ทำงานบนตัวโหมดโดยเฉพาะ การทำงานของระบบจะเป็นอยู่บนพื้นฐานการเกิดขึ้นของเหตุการณ์เป็นหลัก (Event-Based) โดยที่ ณ ขณะใดขณะหนึ่งจะมีเพียงแค่ 1 กระบวนการ (Process) เท่านั้นที่จะทำงานในระบบ หน่วยความจำจะถูกจัดสรร ตั้งแต่ตอนแปลโปรแกรม (Compile time) ซึ่งรหัสคำสั่งไทนีโอเอส (TinyOS code) และรหัสคำสั่งตัวโปรแกรมประยุกต์ (Application code) ที่เขียน จะถูกแปลรวมด้วยกันแล้วค่อยทำงาน

ไทนีโอเอสและโปรแกรมประยุกต์ที่ทำงานจะแบ่งส่วนของการทำงานออกเป็นส่วนโปรแกรม (Component) แต่ละส่วนโปรแกรมจะมีความสามารถในการทำงานที่แตกต่างกัน โดยที่ในแต่ละส่วนโปรแกรมจะมีส่วนที่เรียกว่าส่วนต่อประสาน (Interface) ซึ่งประกอบไปด้วยฟังก์ชันให้เรียกใช้เพื่อเชื่อมต่อการทำงานของส่วนโปรแกรมเข้าด้วยกัน ส่วนโปรแกรมที่เรียกใช้

ฟังก์ชันจะอยู่ในฐานะเป็นผู้ใช้บริการฟังก์ชัน ในทางตรงข้ามส่วนโปรแกรมที่กำหนดการทำงานของฟังก์ชันจะเป็นผู้ให้บริการฟังก์ชัน

ส่วนโปรแกรมพื้นฐานที่มีการใช้งานในงานวิจัยนี้สามารถแบ่งตามหน้าที่การทำงานออกเป็น 3 กลุ่มใหญ่ๆคือ

1) ส่วนโปรแกรมสำหรับการสื่อสารระหว่างโหมต

ส่วนโปรแกรมประเภทนี้จะใช้สำหรับการรับ-ส่งข้อมูลระหว่างโหมต ได้แก่ ส่วนโปรแกรม GenericCommPromiscuous โดยมีฟังก์ชั่นให้บริการผ่านส่วนต่อประสานที่ชื่อว่า SendMsg และ ReceiveMsg สำหรับการส่งและรับข้อมูลตามลำดับ

2) ส่วนโปรแกรมสำหรับการบอกตำแหน่งของโหมต

ส่วนโปรแกรมประเภทนี้ใช้ในการระบุตำแหน่งของโหมตปัจจุบัน ซึ่งประกอบไปด้วยส่วนโปรแกรม ADC ที่ทำหน้าที่รับค่ามาจากตัวแปลงข้อมูลจากสัญญาณแอนะล็อก (Analog signal) มาเป็นข้อมูล 16 บิต โดยมีช่องสัญญาณแอนะล็อก (ADC port) ให้ใช้งานทั้งหมด 256 ช่อง ซึ่งช่องสัญญาณที่ใช้ในการบอกตำแหน่ง X และ Y ของโหมตได้แก่ช่องที่ 128 และ 129

3) อื่นๆ

ส่วนโปรแกรม Timer เป็นส่วนโปรแกรมที่ใช้จัดการเรื่องการจับเวลา โดย Timer หนึ่งสามารถให้บริการกับส่วนโปรแกรมอื่นได้มากถึง 256 ตัวในคราวเดียว โดยมีความสามารถในการตั้งเวลาระดับมิลลิวินาทีได้

4.1.2 โปรแกรมจำลองทอสซิม (TOSSIM)

ทอสซิม (TOSSIM) [6] เป็นโปรแกรมจำลองที่ถูกเขียนขึ้นด้วยภาษาซีและเนสซีเพื่อจำลองรูปแบบโครงสร้างการทำงานจริงของระบบไทนีโอเอส โดยการทำงานจะเริ่มตั้งแต่ฟังก์ชัน main() ในแฟ้มชื่อ nido.nc ซึ่งมีขั้นตอนดังนี้

1) กำหนดค่าพารามิเตอร์ (Parameters) ตามที่ได้รับจากบรรทัดคำสั่ง (Command line)

2) กำหนดสถานะเริ่มต้นของทอสซิม

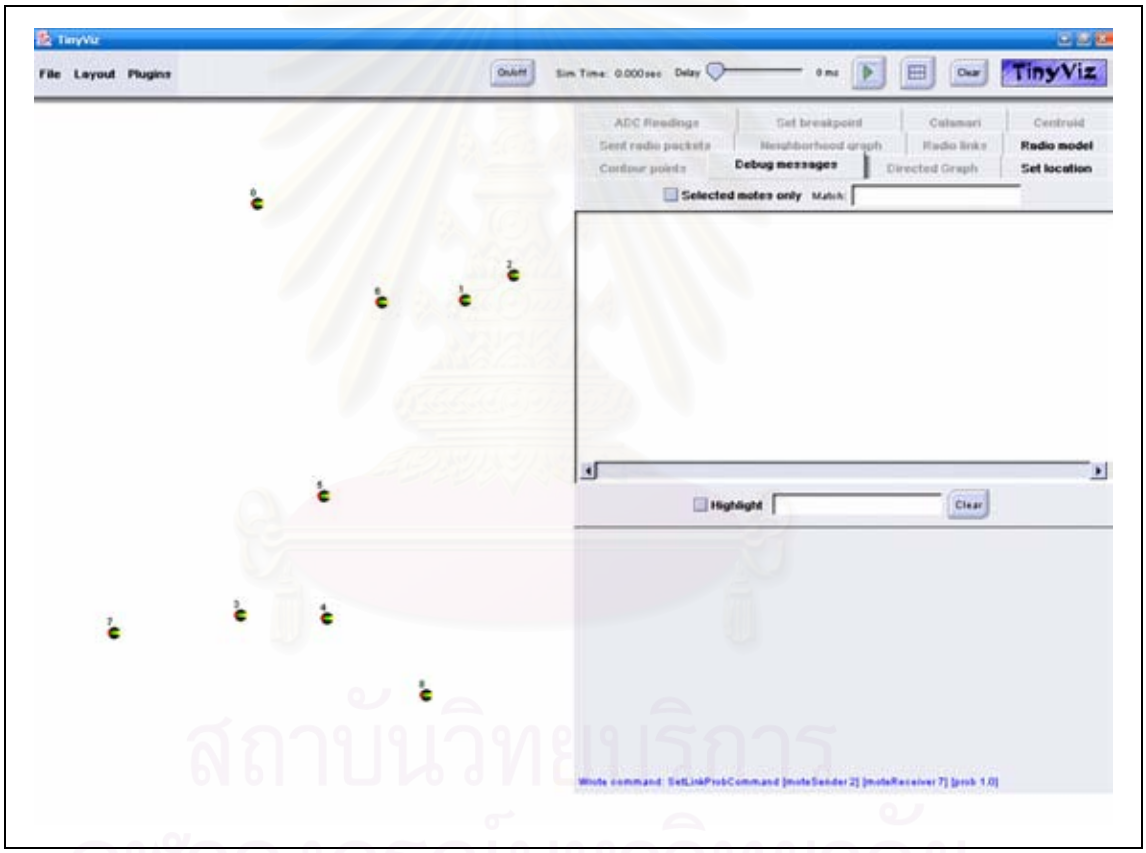
3) กำหนดการทำงานของคำสั่งภายนอก (External command)

4) กำหนดสถานะเริ่มต้นของแต่ละโหมต จากการเรียกฟังก์ชัน StdControl.init() และ StdControl.start()

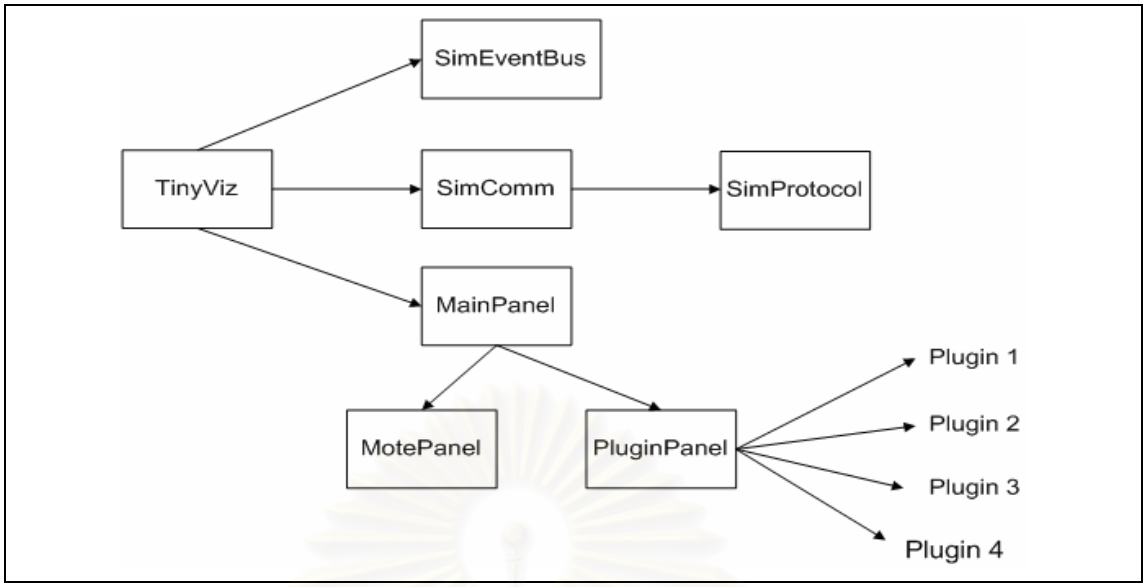
5) เริ่มต้นวงวนเพื่อจัดการเกี่ยวกับเหตุการณ์ (Events) และภารกิจ (Tasks)

4.1.3 โปรแกรมสร้างภาพจำลองไทนีวิส (TinyViz)

ไทนีวิส (TinyViz) จะจำลองการทำงานของโปรแกรมประยุกต์ไทนีโอเอสบนโปรแกรมจำลองทอสซิมให้ออกมาในรูปแบบของภาพ (Graphic user interface) ลักษณะการทำงานจะขึ้นอยู่กับรูปแบบของเหตุการณ์ที่เกิดขึ้น (Event-driven) โดยเหตุการณ์แต่ละเหตุการณ์จะถูกส่งผ่านมายังช่องสัญญาณรับเหตุการณ์ (Event port channel) ในรูปที่ 4.1 และ 4.2 จะแสดงหน้าตาของโปรแกรมไทนีวิสและโครงสร้างทางวัตถุ (Object) ของไทนีวิสที่ทำงานร่วมกันในการจัดการกับเหตุการณ์ต่างๆที่เกิดขึ้นตามลำดับ



รูปที่ 4.1 โปรแกรมสร้างภาพจำลองไทนีวิส



รูปที่ 4.2 โครงสร้างทางวัตถุสำหรับการจัดการกับเหตุการณ์ต่างๆที่เกิดขึ้นของไทนีวิส

4.1.4 โปรแกรมเสริมสำหรับไทนีวิส (TinyViz Plugins)

ไทนีวิสจะมีการแบ่งส่วนการทำงานออกเป็นส่วนย่อยๆที่เรียกว่า โปรแกรมเสริม (Plugin) โดยที่โปรแกรมเสริมแต่ละอันจะทำหน้าที่รับผิดชอบจำลองการทำงานในส่วนที่ต่างกันออกไป เช่น SetLocationPlugin ที่จะเป็นโปรแกรมเสริมเพื่อจัดการเรื่องของการจำลองตำแหน่งของโหมต หรือ RadioModelPlugin เป็นโปรแกรมจำลองที่ใช้สำหรับจำลองระยะการติดต่อระหว่างโหมต ทุกครั้งที่ไทนีวิสเริ่มทำงาน ผู้ใช้จะเห็นว่ามีโปรแกรมเสริมมากมายแสดงออกมาให้เลือกใช้เพื่อการจำลองการทำงานตามที่โปรแกรมประยุกต์ต้องการได้

4.1.5 โปรแกรมจำลองพลังงานสำหรับทอสซิม (PowerTOSSIM)

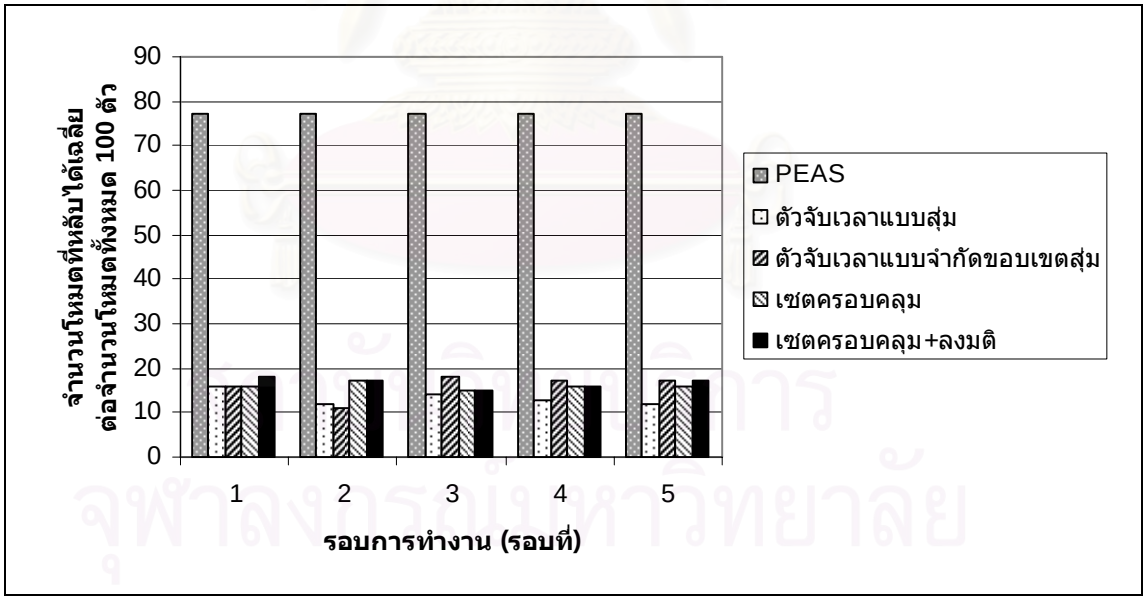
โปรแกรมจำลองพลังงานสำหรับทอสซิมหรือเพาเวอร์ทอสซิม [16] จะเป็นโปรแกรมซึ่งถูกออกแบบมาเพื่อจำลองรูปแบบการใช้พลังงานของโหมตในแพลตฟอร์มต่างๆ โดยสามารถแสดงปริมาณการใช้พลังงานของส่วนอุปกรณ์ที่ทำงานภายในตัวโหมต ณ ขณะเวลาใดๆได้ เช่น ส่วนประมวลผล (CPU) ส่วนรับส่งข้อมูลจากสัญญาณวิทยุ และส่วนหลอดไฟ (LEDs) เป็นต้น ค่าพลังงานที่แสดงจะเป็นค่าพลังงานที่ส่วนอุปกรณ์นั้นใช้เมื่ออยู่ในสถานะการทำงานต่างๆโดยเทียบจำลองมาจากต้นแบบคือโหมตจริงที่ทำงาน ทำให้ผลการจำลองอัตราการใช้พลังงานที่ได้จากเพาเวอร์ทอสซิมนี้ค่อนข้างได้ค่าที่แม่นยำและใกล้เคียงกับความเป็นจริง งานวิจัยนี้จึงนำเอาค่าพลังงานที่วัดได้จากเพาเวอร์ทอสซิมมาใช้เป็นตัววัดอัตราการประหยัดพลังงานของอัลกอริทึม

4.1.6 ภาษาโปรแกรมเนสซี (nesC Language)

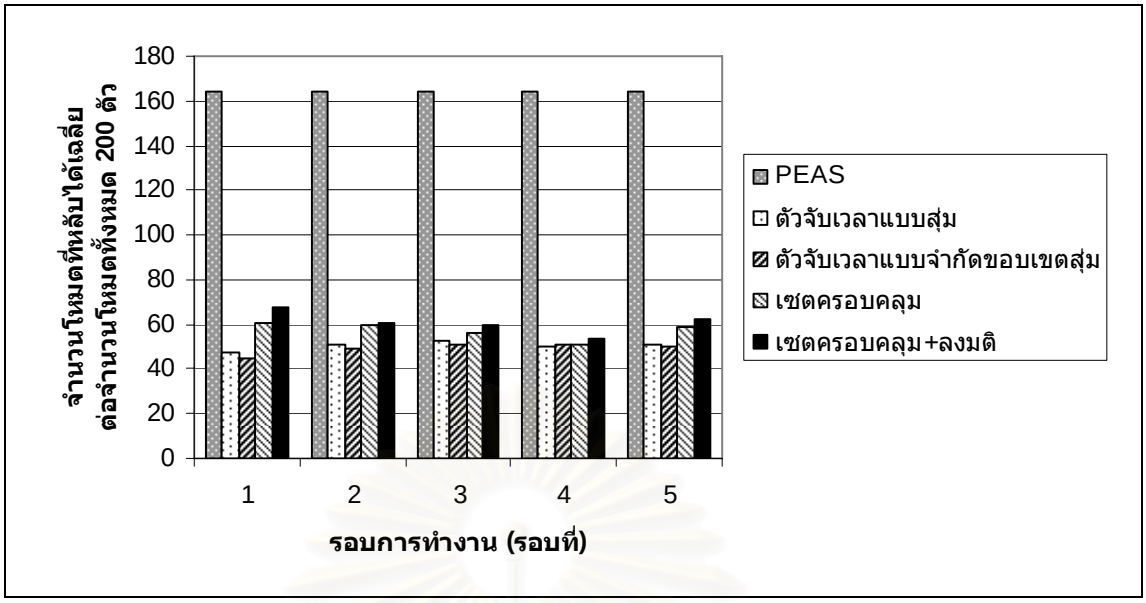
เนสซี (nesC) [15] คือภาษาโปรแกรมที่ใช้ในการเขียนโปรแกรมประยุกต์สำหรับทำงานบนโหมตสำหรับระบบเครือข่ายตัวรับรู้ โดยที่รูปแบบการเขียนจะเป็นลักษณะการนำเอาส่วนของโปรแกรมหลาย ๆส่วนมาเชื่อมต่อกันเพื่อทำงานตามที่โปรแกรมประยุกต์ต้องการ ลักษณะการเขียนแบบนี้จะเรียกว่าลักษณะการเขียนโปรแกรมโดยอาศัยส่วนโปรแกรมประกอบ (Component-Based) นอกจากนี้เนสซียังถูกออกแบบมาให้มีการทำงานแบบเน้นประหยัดพลังงานด้วย ซึ่งเหมาะกับคุณลักษณะของโหมตที่มีพลังงานจำกัด

4.2 ผลการเปรียบเทียบจำนวนโหมตที่หลับได้เฉลี่ย

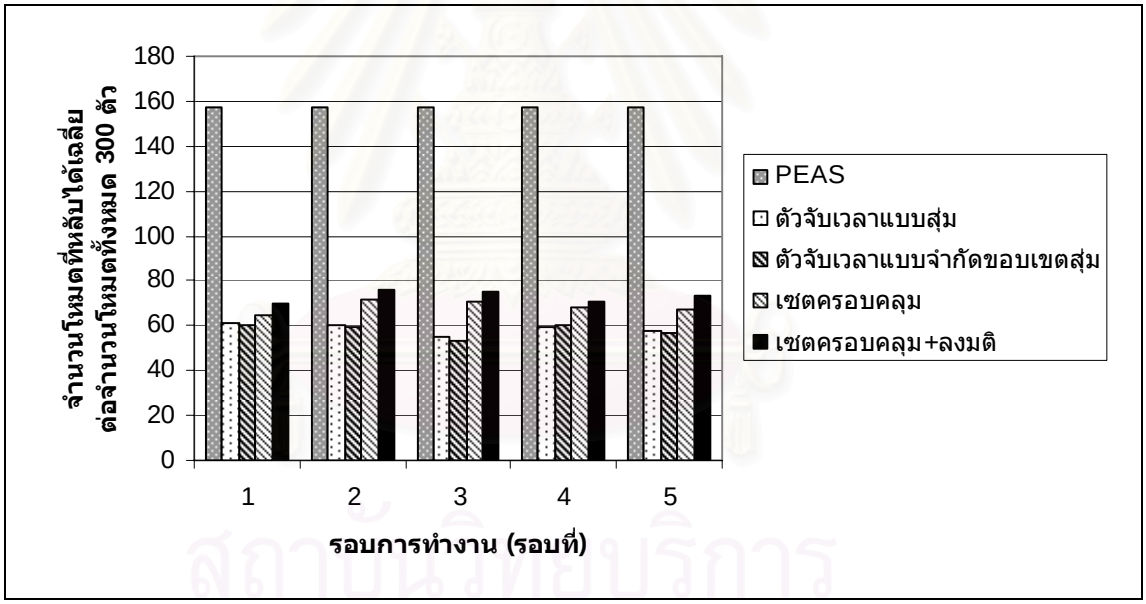
หลังจากที่อัลกอริทึมควบคุมความหนาแน่นทำการตรวจสอบความซ้ำซ้อนของโหมตและผ่านกระบวนการจัดการการตื่น-หลับของโหมตเรียบร้อยแล้ว โหมตที่มีทั้งหมดในระบบจะถูกแบ่งออกเป็น 2 พวกคือ โหมตที่ตื่นขึ้นมาทำงานตามปกติและโหมตซ้ำซ้อนที่หลับตัวเองลงไปเพื่อประหยัดพลังงาน การทดลองส่วนนี้จะทำการเปรียบเทียบจำนวนโหมตที่สามารถหลับได้เฉลี่ยต่อจำนวนโหมตทั้งหมดในระบบเท่ากับ 100 200 และ 300 ตัว ผลการเปรียบเทียบจะแสดงได้ดังแผนภูมิแท่งในรูปที่ 4.3 4.4 และ 4.5 ตามลำดับ



รูปที่ 4.3 กราฟแสดงจำนวนโหมตที่หลับได้เฉลี่ยต่อจำนวนโหมตทั้งหมด 100 ตัว



รูปที่ 4.4 กราฟแสดงจำนวนโหมดที่หลับได้เฉลี่ยต่อจำนวนโหมดทั้งหมด 200 ตัว



รูปที่ 4.5 กราฟแสดงจำนวนโหมดที่หลับได้เฉลี่ยต่อจำนวนโหมดทั้งหมด 300 ตัว

จากรูปที่ 4.3 4.4 และ 4.5 จะเห็นได้ว่า เมื่อเปรียบเทียบที่จำนวนโหมดในระบบมีค่าเท่ากัน จำนวนโหมดที่หลับได้เฉลี่ยต่อหนึ่งรอบการทำงานของอัลกอริทึม PEAS จะมีค่ามากที่สุดเสมอ เนื่องจากว่าในอัลกอริทึม PEAS นั้นจะอาศัยหลักการสุ่มตรวจสอบโดยขอแค่เพียงตรวจพบว่ามีโหมดเพื่อนบ้านตัวใดตัวหนึ่งที่ยังเปิดทำงานอยู่ก็จะทำให้ตัวมันตัดสินใจเปลี่ยนสถานะเป็นหลับทันที ซึ่งหลักการที่ว่านี้สามารถทำให้โหมดส่วนใหญ่หลับลงไปได้เป็นจำนวนมากจริง แต่อย่างไรก็ตาม PEAS ไม่ได้นำเอาปัจจัยพื้นที่ครอบคลุมของระบบมาใช้พิจารณาใน

การเลือกหลับโหมดแต่อย่างใด ทำให้โหมดที่ตื่นทำงานนั้นไม่อาจรับประกันว่าจะสามารถครอบคลุมพื้นที่เฝ้าสังเกตทั้งหมดทดแทนโหมดที่หลับไปได้ ต่างจากแบบวิธีการของอีก 4 อัลกอริทึมที่เหลือซึ่งถึงแม้ว่าจะมีจำนวนโหมดที่หลับได้เฉลี่ยออกมาน้อยกว่า PEAS แต่ระบบสามารถรับประกันการรักษาพื้นที่ครอบคลุมภายหลังการทำงานของอัลกอริทึมได้

เมื่อพิจารณาเปรียบเทียบระหว่างอัลกอริทึมควบคุมความหนาแน่นแบบอาศัยตัวจับเวลาแบบสุ่มและตัวจับเวลาแบบจำกัดขอบเขตสุ่ม จะเห็นว่าจำนวนโหมดที่หลับได้เฉลี่ยจากทั้ง 2 อัลกอริทึมนี้จะมีค่าใกล้เคียงกันเสมอไม่ว่าจำนวนโหมดที่มีทั้งหมดในระบบจะมากหรือน้อย เนื่องจากว่าการจำกัดขอบเขตของการสุ่มเวลาเป็นเพียงการช่วยผลัดเปลี่ยนโหมดที่มีพลังงานเหลือมากให้มีโอกาสในการถูกสุ่มให้ตื่นมาทำงานก่อนเท่านั้น ไม่ได้ส่งผลต่อการเพิ่มปริมาณโหมดที่หลับได้แต่อย่างใด ทำให้แนวโน้มของจำนวนโหมดที่หลับได้เฉลี่ยจะยังคงเป็นไปตามแบบวิธีการสุ่มเช่นเดิมไม่ว่าจะมีจำกัดขอบเขตหรือไม่ก็ตาม ต่างจากผลการเปรียบเทียบระหว่างอัลกอริทึมควบคุมความหนาแน่นแบบอาศัยเซตครอบคลุมและเซตครอบคลุมผสมการลงมติที่จะได้ผลจำนวนโหมดที่หลับได้เฉลี่ยออกมาต่างกัน เนื่องจากว่าการลงมติจะมีส่วนช่วยในการเลือกหลับโหมดให้สอดคล้องและเหมาะสมต่อโหมดส่วนมากในระบบ ทำให้จำนวนโหมดที่หลับได้เฉลี่ยรวมของระบบจะมีค่าสูงกว่ากรณีไม่มีการลงมติที่อาศัยคำตอบจากตัวโหมดเองตัวเดียวเท่านั้น

ในรูปที่ 4.3 แสดงผลการเปรียบเทียบจำนวนโหมดที่หลับได้เฉลี่ยเมื่อจำนวนโหมดในระบบมีค่าเป็น 100 ตัว จากผลที่ออกมาจะเห็นว่า วิธีการใช้ตัวจับเวลาแบบสุ่ม (รวมไปถึงแบบจำกัดขอบเขตสุ่ม) ในการเลือกโหมดจะมีจำนวนโหมดที่หลับไปได้เฉลี่ยต่อรอบการทำงานน้อยกว่าวิธีการเซตครอบคลุม (รวมไปถึงแบบเซตครอบคลุมผสมการลงมติ) เมื่อเทียบที่รอบการทำงานเดียวกัน อย่างไรก็ตามผลต่างของจำนวนโหมดที่หลับได้เฉลี่ยระหว่างทั้ง 2 วิธีนี้อาจมีค่าแสดงออกมาเป็นตัวเลขที่ไม่สูงนักคืออยู่ประมาณ 2-3 โหมดเท่านั้น (ประมาณร้อยละ 2-3 ของจำนวนโหมดทั้งหมดในระบบ) เนื่องด้วยระบบยังมีความหนาแน่นของโหมดต่อพื้นที่ต่ำ โหมดส่วนใหญ่มีตำแหน่งกระจายตัวกันอยู่และเกิดการซ้อนทับของพื้นที่ตรวจจับระหว่างกันไม่มากนัก ทำให้โดยรวมแล้วผลจากกระบวนการจัดการตื่นหลับของโหมดจากทั้ง 2 วิธีจะยังไม่แสดงออกมาเด่นชัด

ในรูปที่ 4.4 จะแสดงผลการเปรียบเทียบเมื่อจำนวนโหมดในระบบมีเพิ่มขึ้นเป็น 200 ตัว ซึ่งหมายถึงการเพิ่มโอกาสให้โหมดซ้ำซ้อนมีพื้นที่ตรวจจับซ้อนทับกันในระบบมากขึ้น ผลการทดลองที่ออกมาจะเห็นได้ว่า วิธีการเซตครอบคลุม (รวมไปถึงแบบเซตครอบคลุมผสมการลงมติ) สามารถลดจำนวนโหมดที่หลับลงไปได้มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม (รวมไปถึงแบบจำกัดขอบเขตสุ่ม) อยู่ระหว่างช่วง 10-21 โหมดหรือคิดเป็นร้อยละ 10-21 ของจำนวนโหมดทั้งหมดในระบบ ซึ่งเพิ่มขึ้นจากกรณีที่ระบบมีโหมดทั้งหมด 100 ตัว

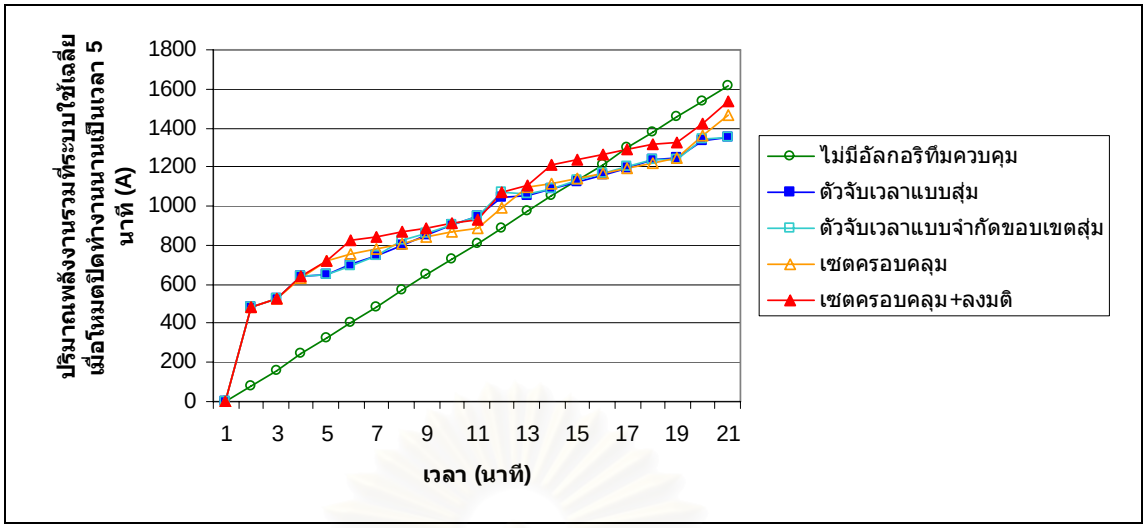
สำหรับกรณีที่โหมดในระบบมีจำนวนมากกว่า 200 ตัวขึ้นไป เช่นกรณี 300 ตัว ในรูปที่ 4.5 ผลความต่างระหว่างจำนวนโหมดที่หลับได้เฉลี่ยจากทั้ง 2 วิธีการจะเพิ่มสูงขึ้นแตกต่างจากกรณีที่มีโหมด 200 ตัวไม่มากนัก เหตุเพราะว่าโหมดที่เพิ่มเข้าไปใหม่นี้เริ่มจะกระจายตำแหน่งครอบคลุมพื้นที่เฝ้าสังเกต (ซึ่งจำกัดไว้คงที่ที่ 100x100 ตารางหน่วย) ซ้ำๆในบริเวณเดิมกับโหมดอื่นๆที่ครอบคลุมอยู่ก่อนหน้าแล้ว ทำให้วิธีการเซตครอบคลุมผสมการลงมติซึ่งพิจารณาที่พื้นที่ครอบคลุมเป็นหลักจะมีโหมดซ้ำซ้อนที่ต้องตื่นขึ้นมาทำงานเพิ่มขึ้นจากกรณีมีโหมด 200 ตัวไม่มากนักและมีแนวโน้มที่จะคงที่หากมีโหมดในระบบมากขึ้น

จากผลการทดลองทั้งหมดสามารถวิเคราะห์ได้ว่า ในวิธีการใช้ตัวจับเวลาแบบสุ่มโหมดซ้ำซ้อนทุกตัวจะถือว่ามีโอกาสถูกเลือกให้ตื่นขึ้นมาทำงานใกล้เคียงกัน (โดยใช้การสุ่มตัวจับเวลา) ซึ่งในความเป็นจริงแล้วโหมดซ้ำซ้อนบางตัวหากได้รับเลือกให้ตื่นมาทำงานแล้วจะส่งผลดีต่อระบบได้มากกว่าโหมดซ้ำซ้อนตัวอื่น (ครอบคลุมพื้นที่ได้มากกว่าหรือมีพลังงานเหลือมากกว่า) ทำให้คำตอบจากการเลือกโหมดที่ได้จากการสุ่มตัวจับเวลานั้นอาจไม่ใช่คำตอบที่เหมาะสมและมีการตื่นโหมดซ้ำซ้อนขึ้นมาทำงานเปลืองมากกว่าที่จำเป็น ต่างจากวิธีการเซตครอบคลุม (รวมไปถึงแบบเซตครอบคลุมผสมการลงมติ) ที่มีการให้น้ำหนักการครอบคลุมพื้นที่ตรวจจับและพลังงานที่เหลือของโหมดถ่วงเข้าไปในการตัดสินใจ ทำให้โหมดซ้ำซ้อนที่มีคุณสมบัติครอบคลุมพื้นที่ได้กว้างกว่าและเหลือพลังงานมากกว่ามีโอกาสถูกเลือกให้ตื่นมาทำงานได้ดีมากกว่าโหมดที่ครอบคลุมพื้นที่ได้น้อยและเหลือพลังงานน้อย เพราะฉะนั้นผลคำตอบที่ได้จากวิธีการเซตครอบคลุมจึงมีจำนวนของโหมดที่หลับได้เฉลี่ยต่อรอบการทำงานเพื่อครอบคลุมพื้นที่เฝ้าสังเกตที่ต้องการทั้งหมดน้อยกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม

4.3 ผลการเปรียบเทียบปริมาณพลังงานรวมที่ระบบใช้เฉลี่ย

เนื่องจากการทำงานของอัลกอริทึมควบคุมความหนาแน่นจะมีการหลับโหมดบางโหมดลงไปเพื่อประหยัดการใช้พลังงาน ดังนั้นในแง่ประสิทธิภาพของอัลกอริทึมจึงสามารถนำปริมาณการใช้พลังงานรวมเฉลี่ยของระบบเทียบกับเวลามาใช้เป็นมาตรวัดในการเปรียบเทียบได้ จากการจำลองทอพอโลยีของระบบเครือข่ายตัวรับรู้ด้วยการสุ่มตำแหน่งจากโปรแกรมเสริมไทนีวิสออกมาต่างกันจำนวน 10 แบบโดยมีจำนวนโหมดในระบบทั้งหมดเป็น 100 ตัว และแต่ละตัวมีค่าพลังงานเริ่มต้นเท่ากับ 20,000 mA ผลการทดลองจะแสดงได้ดังกราฟในรูปที่ 4.6

ในรูปที่ 4.6 จะเป็นการเปรียบเทียบปริมาณพลังงานรวมที่ระบบใช้เฉลี่ยเทียบกับเวลาระหว่างกรณีไม่มีอัลกอริทึมควบคุมความหนาแน่นทำงานและกรณีที่มีอัลกอริทึมควบคุมความหนาแน่นทำงานด้วย โดยกำหนดให้ช่วงเวลาที่โหมดเปลี่ยนสถานะเป็นหลับเพื่อประหยัดพลังงานนานเป็นเวลา 5 นาที เนื่องจากในการทำงานของอัลกอริทึมทั้งวิธีตัวจับเวลาแบบสุ่มและวิธีเซตครอบคลุม มีลักษณะการจัดการวนเป็นรอบๆ ทำให้กราฟที่ได้จึงมีความชันไม่คงที่



รูปที่ 4.6 กราฟแสดงปริมาณพลังงานรวมที่ระบบใช้เฉลี่ยเมื่อโหมดหลับเป็นเวลา 5 นาที

ตลอดเวลา (มีลักษณะคล้ายขั้นบันไดซึ่งหมายถึงแต่ละรอบการทำงาน) โดยที่ในแต่ละรอบของการทำงานจะมีการแบ่งออกเป็น 2 ช่วงย่อยคือช่วงการแปลงปัญหาไปเป็นเซตครอบคลุมและคำนวณหาคำตอบ (ซึ่งเป็นช่วงที่โหมดใช้พลังงานสูงกว่าปกติ ความชันของกราฟจะพุ่งขึ้นสูง) และช่วงเวลาที่โหมดเปลี่ยนสถานะไปเป็นหลับตามคำตอบที่หาได้ (ซึ่งเป็นช่วงที่โหมดประหยัดพลังงาน ทำให้กราฟจะมีความชันต่ำลง)

จากรูปที่ 4.6 จะเห็นได้ว่าแนวโน้มการใช้พลังงานของอัลกอริทึมทั้ง 4 วิธีการจะมีปริมาณที่เพิ่มสูงขึ้นตามเวลาการทำงานที่ผ่านไปเช่นเดียวกับกรณีที่ไม่มีอัลกอริทึมควบคุม แต่เนื่องจากในกระบวนการของอัลกอริทึมควบคุมความหนาแน่นนั้นโหมดมีการดึงเอาพลังงานมากกว่าปกติเพื่อนำไปใช้สำหรับการคำนวณและรับ-ส่งข้อมูลระหว่างโหมด ทำให้ในช่วงแรกของการทำงานเมื่อเทียบ ณ เวลาเดียวกัน ระบบจะมีการสูญเสียพลังงานไปในระดับที่สูงกว่ากรณีที่ไม่มีอัลกอริทึมควบคุมความหนาแน่นทำงาน อย่างไรก็ตามภายหลังจากที่อัลกอริทึมได้คำตอบและหลับโหมดบางส่วนลงไปเพื่อประหยัดพลังงานแล้ว แนวโน้มการใช้พลังงานของระบบจะมีอัตราที่ต่ำลงกว่ากรณีไม่มีอัลกอริทึมควบคุม โดยที่วิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มจะมีเส้นกราฟแสดงปริมาณการใช้พลังงานเทียบกับเวลาใกล้เคียงกัน เนื่องจากมีจำนวนโหมดที่หลับได้เท่าๆกัน ต่างจากวิธีการเซตครอบคลุมผสมการลงมติที่จะมีเส้นกราฟแสดงปริมาณการใช้พลังงานเทียบกับเวลาสูงกว่าวิธีการเซตครอบคลุม เนื่องจากมีการใช้พลังงานเพื่อรับส่งข้อมูลมติระหว่างโหมดเพิ่มเข้ามาด้วย

ในรอบแรกของการทำงาน (ช่วงนาทีที่ 0-10) จะพบว่า ช่วงเวลา 1 นาทีแรกสำหรับทั้งวิธีการใช้ตัวจับเวลาแบบสุ่ม วิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่ม วิธีการเซตครอบคลุม และวิธีการเซตครอบคลุมผสมการลงมติคำตอบ ระบบจะมีการใช้พลังงานรวมเฉลี่ยที่สูงมากถึง

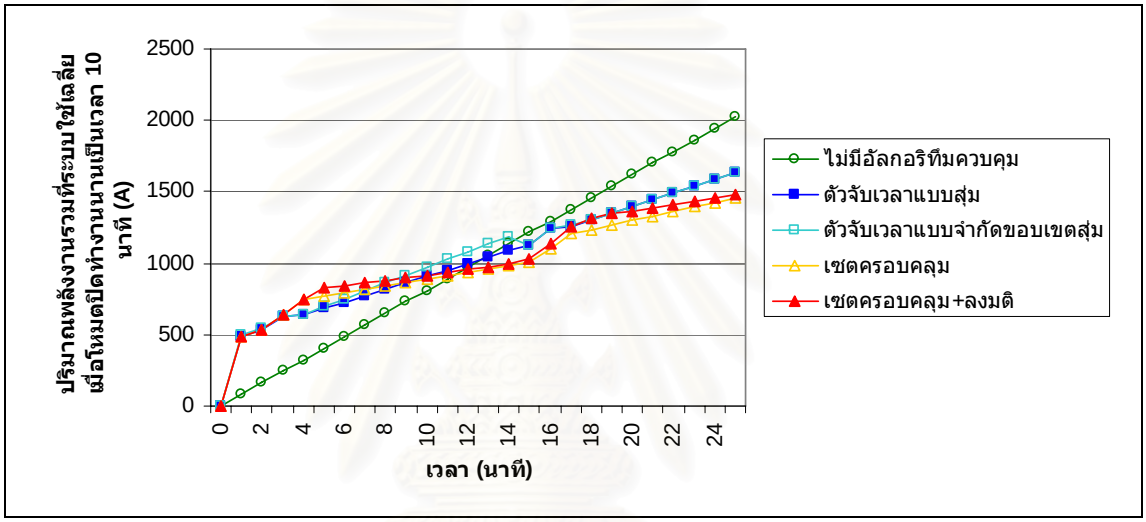
ประมาณ 500 A ในช่วงนี้จะเป็นช่วงที่โหมดมีการแลกเปลี่ยนข้อมูลตำแหน่งของโหมดระหว่างกันเพื่อนำมาใช้ในการคำนวณว่าตนเองเป็นโหมดซ้ำซ้อนหรือไม่ จากการที่อัลกอริทึมทั้ง 4 วิธีใช้หลักการคำนวณความซ้ำซ้อนแบบเดียวกัน ทำให้ปริมาณพลังงานที่ใช้ออกมาในช่วง 1 นาทีแรกนี้มีค่าเท่ากัน อย่างไรก็ตามปริมาณพลังงานที่ใช้จากอัลกอริทึมทั้ง 4 วิธีจะเริ่มแสดงค่าแตกต่างกันชัดเจนขึ้นในช่วง 5 นาทีต่อมาคือช่วงนาทีที่ 1-5 เนื่องจากว่าวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมตินั้นโหมดมีความจำเป็นต้องแลกเปลี่ยนข้อมูลเซตและมติกับโหมดเพื่อนบ้านต่างจากวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มที่ไม่ต้องการข้อมูลใด ๆเพิ่มเติมในการตัดสินคำตอบ ทำให้เส้นกราฟแสดงปริมาณพลังงานที่ใช้ในช่วงนี้ของวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติจะสูงกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มอยู่ประมาณ 100 A

ต่อมาในช่วงนาทีที่ 6-10 ของการทำงาน จะเป็นช่วงเวลาที่มีการหลับโหมดบางส่วนเพื่อประหยัดพลังงาน ในช่วงนี้เส้นกราฟการใช้พลังงานของอัลกอริทึมทั้ง 4 วิธีจะมีความชันหรืออัตราการสูญเสียพลังงานที่คงที่แต่ไม่เท่ากัน เนื่องด้วยวิธีการเซตครอบคลุมผสมการลงมติสามารถหลับโหมดลงไปได้จำนวนมากกว่าทั้งวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่ม (จากหัวข้อ 4.2) ทำให้ระบบมีอัตราการสูญเสียพลังงานที่น้อยกว่า (สังเกตจากความชันของเส้นกราฟจากวิธีการเซตครอบคลุมผสมการลงมติที่มีค่าต่ำกว่า)

อย่างไรก็ตามเนื่องจากว่าในแต่ละรอบการทำงานของวิธีการเซตครอบคลุมผสมการลงมติมีการเสียพลังงานในการคำนวณ ณ ช่วง 5 แรกของการทำงานมาก ทำให้เมื่อวัดปริมาณการใช้พลังงานของระบบรวม ณ นาทีสุดท้ายของแต่ละรอบการทำงานจากวิธีการเซตครอบคลุมผสมการลงมติจะยังมีค่าสูงกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มอยู่ (รวมไปวิธีการเซตครอบคลุม ซึ่งไม่มีการลงมติด้วย) ถึงตัวอย่างเช่น ช่วงนาทีที่ 11-18 ซึ่งเป็นรอบที่ 2 ของการทำงาน วิธีการเซตครอบคลุมผสมการลงมติจะมีปริมาณการใช้พลังงานสุดท้าย ณ นาทีที่ 18 มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มอยู่ประมาณ 100 A

ในรูปที่ 4.7 จะเป็นการทดลองโดยขยายระยะเวลาที่โหมดหลับนานเพิ่มขึ้นจากเดิม 5 นาทีเป็น 10 นาที ผลที่ได้จะเห็นว่า เส้นกราฟแสดงปริมาณพลังงานที่ใช้จากวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติช่วงนาทีที่ 0-11 นั้นจะอยู่สูงกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มเช่นเดียวกับกรณีระยะเวลาที่โหมดหลับนาน 5 นาที อย่างไรก็ตามเมื่อพิจารณาตั้งแต่นาทีที่ 11 เป็นต้นไป จะพบว่าเส้นกราฟของวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติเริ่มจะอยู่ต่ำกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่ม จากจุดนี้สามารถวิเคราะห์ได้ว่าถึงแม้พลังงานที่โหมดใช้ในการคำนวณเพื่อให้ได้มาซึ่งคำตอบในช่วงแรกของการ

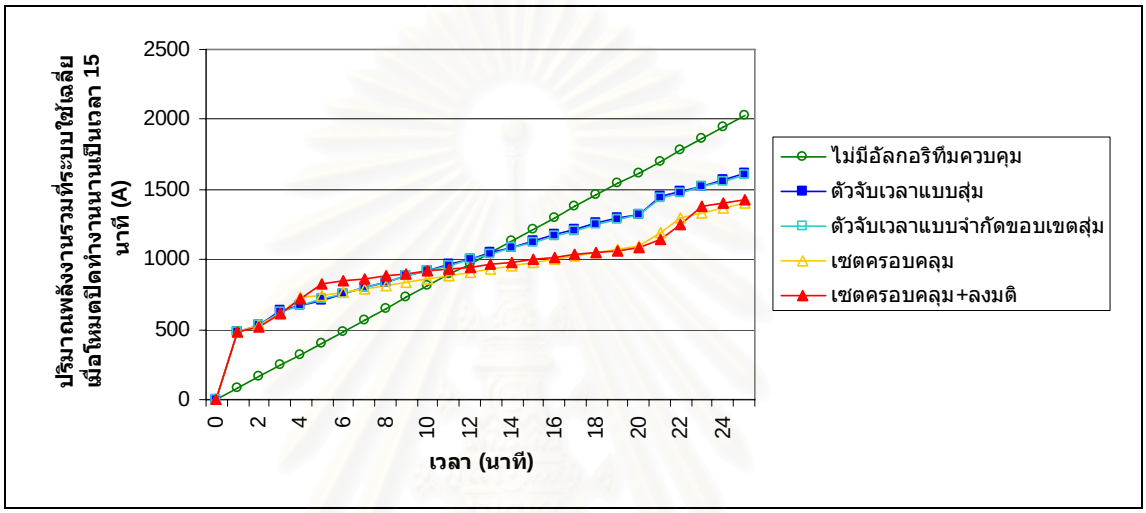
ทำงานจากวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมตินั้นจะสูงกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่ม (ซึ่งทำให้ผลลัพธ์ในช่วงแรกของการทำงานมีการประหยัดพลังงานที่แย่กว่า) แต่เนื่องจากผลลัพธ์ที่ได้จากวิธีการเซตครอบคลุมมีจำนวนโหมดที่หลับลงไปได้มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม ทำให้หากยืดระยะเวลาที่โหมดเหล่านี้หลับเพื่อประหยัดพลังงานออกไปให้นานพอ (ในที่นี้คือ 10 นาที) จะทำให้แนวโน้มการใช้พลังงานเฉลี่ยรวมของระบบในระยะยาวจากวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติจะต่ำกว่าวิวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มเมื่อเทียบ ณ ขณะเวลาการทำงานเดียวกัน



รูปที่ 4.7 กราฟแสดงปริมาณพลังงานรวมที่ระบบใช้เฉลี่ยเมื่อโหมดหลับเป็นเวลา 10 นาที

โดยปกติแล้วการใช้งานอัลกอริทึมควบคุมความหนาแน่นบนสภาพแวดล้อมจริงนั้นจะมีความถี่ของการวนรอบทำงานไม่สูงนัก เนื่องจากการคำนวณหาโหมดที่สามารถหลับได้แต่ละครั้งจำเป็นต้องใช้พลังงานมากพอสมควร หากอัลกอริทึมมีความถี่การวนรอบที่สูงจะทำให้ได้ผลการประหยัดพลังงานที่ได้ภายหลังไม่คุ้มค่ากับพลังงานที่เสียไปในการคำนวณ ดังนั้นระยะเวลาที่โหมดหลับเพื่อประหยัดพลังงานนั้นจึงมักถูกกำหนดให้นานพอสมควร ซึ่งจากผลการทดลองในรูปที่ 4.6 และ 4.7 แสดงให้เห็นว่า อัลกอริทึมควบคุมความหนาแน่นจะสามารถลดปริมาณการใช้พลังงานเฉลี่ยรวมของระบบได้เล็กน้อยในกรณีที่แต่ละรอบการทำงานของอัลกอริทึมใช้เวลาไม่นานนัก แต่หากระบบมีรอบการทำงานหรือระยะเวลาที่โหมดหลับเพื่อประหยัดพลังงานนานขึ้น (เหมาะกับการใช้งานในสภาพแวดล้อมจริงมากขึ้น) วิธีการเซตครอบคลุมผสมการลงมติจะแสดงผลการลดปริมาณการใช้พลังงานรวมของระบบได้มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มเมื่อเทียบ ณ ขณะเวลาการทำงานเดียวกัน

ในรูปที่ 4.8 จะเป็นตัวอย่างกรณีที่มีการกำหนดระยะเวลาที่โหมดหลับนานเพิ่มขึ้นอีกเป็น 15 นาที จะเห็นว่า เมื่อพิจารณานาทีที่ 21 ของการทำงาน อัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติจะสามารถลดปริมาณการใช้พลังงานรวมเฉลี่ยของระบบลงไปได้มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการใช้ตัวจับเวลาแบบจำกัดขอบเขตสุ่มชัดเจนถึง 250 A หรือคิดเป็นร้อยละ 17.37 ของปริมาณพลังงานที่วิธีตัวจับเวลาแบบจำกัดขอบเขตสุ่มชัดเจนใช้ เป็นต้น



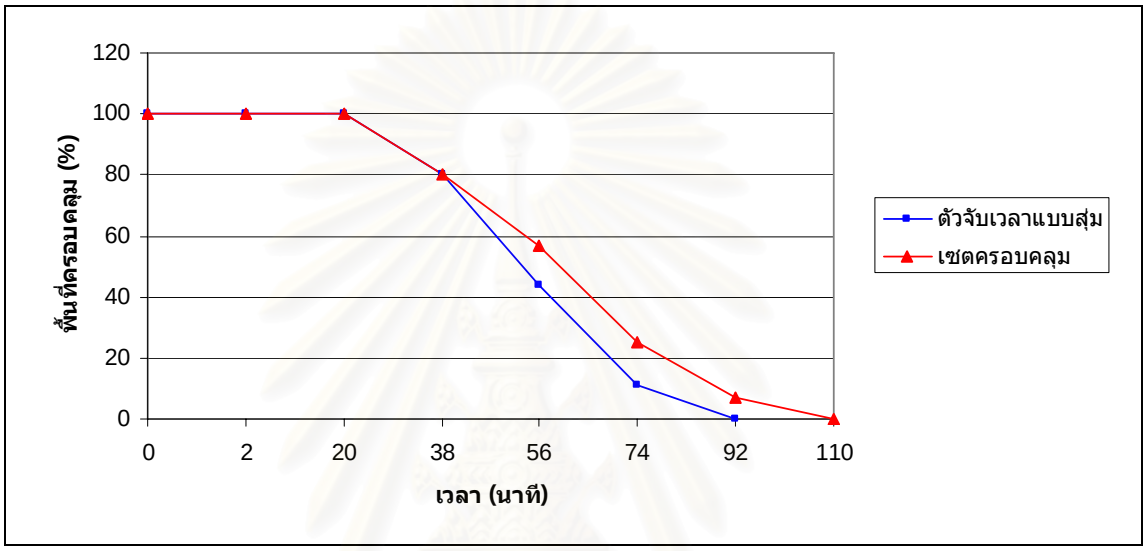
รูปที่ 4.8 กราฟแสดงปริมาณพลังงานรวมที่ระบบใช้เฉลี่ยเมื่อโหมดหลับเป็นเวลา 15 นาที

จากทั้งรูปที่ 4.7 และ 4.8 สามารถวิเคราะห์ได้ว่าการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมติสามารถลดปริมาณพลังงานรวมเฉลี่ยที่ระบบใช้ลงได้มากกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มเมื่อเทียบ ณ เวลาการทำงานเดียวกัน โดยที่ประสิทธิภาพการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมและวิธีการเซตครอบคลุมผสมการลงมตินั้นจะแสดงผลการลดพลังงานออกมาได้ดีชัดเจนขึ้นเมื่อระบบมีระยะเวลาที่โหมดหลับเพื่อประหยัดพลังงานสอดคล้องกับการใช้งานบนสภาพแวดล้อมจริงมากขึ้นได้แก่ 15 นาทีขึ้นไป

4.4 ผลการเปรียบเทียบอายุเวลาการครอบคลุมพื้นที่ตรวจจับ

ในหัวข้อนี้จะเป็นการแสดงถึงปริมาณพื้นที่ตรวจจับที่ระบบครอบคลุมเทียบกับระยะเวลาที่ผ่านไป โดยการวัดปริมาณพื้นที่ครอบคลุมการตรวจจับจะวัดจากการแบ่งพื้นที่เฝ้าสังเกตทั้งหมดออกเป็นตาตาราง แต่ละพื้นที่จัตุรัสย่อยในตารางจะมีขนาดเป็น 1x1 ตารางหน่วย และให้จุดศูนย์กลางของพื้นที่ย่อยเป็นตัวแทนการครอบคลุมพื้นที่นั้นๆ วิธีการวัดจะเทียบออกมาเป็นร้อยละของจำนวนพื้นที่ย่อยทั้งหมดที่โหมดซึ่งเปิดทำงาน ณ ขณะเวลานั้นสามารถ

ครอบคลุมได้เทียบกับจำนวนพื้นที่ย่อยทั้งหมดที่ครอบคลุมได้ในกรณีที่โหมดทุกตัวเปิดทำงาน (ซึ่งคือกรณีที่ระบบไม่มีอัลกอริทึมควบคุมความหนาแน่นทำงาน) การทดลองจะอ้างอิงแบบเดียวกันกับในหัวข้อ 4.3 คือทำการจำลองทอพอโลยีของระบบเครือข่ายตัวรับรู้ด้วยการสุ่มตำแหน่งออกมาต่างกัน 10 แบบโดยมีโหมดในระบบทั้งหมด 100 ตัว แต่ละตัวมีค่าพลังงานเริ่มต้นเท่ากับ 20,000 mA และโหมดมีการหลับเพื่อประหยัดพลังงานนานเป็นระยะเวลา 15 นาที ผลการทดลองจะแสดงได้ดังรูปที่ 4.9



รูปที่ 4.9 กราฟแสดงอายุเวลาการครอบคลุมพื้นที่ตรวจจับ

จากกราฟในรูปที่ 4.9 จะเห็นว่า ในช่วงเวลานาทีที่ 0-25 ปริมาณพื้นที่ครอบคลุมของระบบจะมีค่าเท่ากับร้อยละ 100 คงที่ตลอด ซึ่งสามารถอธิบายได้ว่า ในช่วงเวลา 25 นาทีแรกของการทำงาน โหมดไม่ซ้ำซ้อนซึ่งจำเป็นต้องเปิดทำงานตลอดเวลาจะยังคงมีพลังงานเหลือเพียงพอที่จะทำงานได้ตามปกติอยู่ ในขณะที่ตัวโหมดซ้ำซ้อนเองส่วนหนึ่งซึ่งผ่านกระบวนการเลือกหลับจากอัลกอริทึมควบคุมความหนาแน่นทั้งแบบวิธีเซตครอบคลุมและตัวจับเวลาแบบสุ่มก็สามารถเปิดทำงานเพื่อครอบคลุมพื้นที่ได้เช่นเดียวกัน ดังนั้นในช่วงเวลา 25 นาทีแรกนี้จึงเป็นช่วงเวลาที่ระบบสามารถทำงานอัลกอริทึมควบคุมความหนาแน่นโดยที่ยังรักษาการครอบคลุมพื้นที่เป้าสังเกตได้ในปริมาณเทียบเท่ากับกรณีที่โหมดทุกตัวเปิดทำงาน

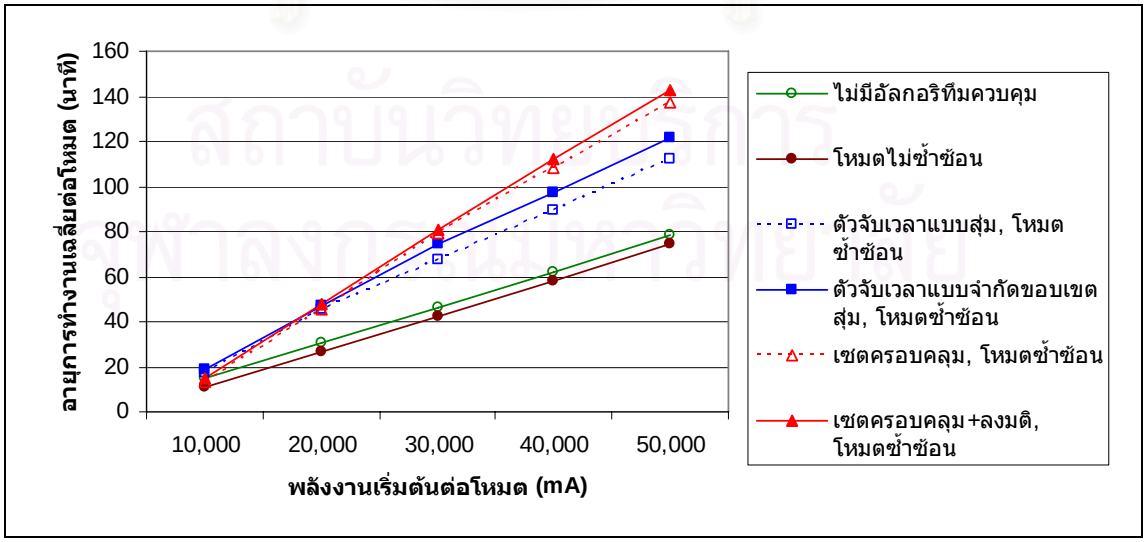
อย่างไรก็ตามเมื่อพิจารณาในเวลาถัดมาคือนาทีที่ 26 จะเป็นเวลาที่อัลกอริทึมควบคุมความหนาแน่นทั้งแบบวิธีเซตครอบคลุมและตัวจับเวลาแบบสุ่มเริ่มไม่สามารถรักษาสภาพพื้นที่ครอบคลุมเดิมของระบบเอาไว้ได้อีกต่อไป เนื่องจากกว่าโหมดไม่ซ้ำซ้อนทุกโหมดได้หมดพลังงานลง ทำให้ระบบสูญเสียการครอบคลุมพื้นที่บางส่วนซึ่งเดิมเคยเป็นพื้นที่ในความดูแลของโหมดไม่ซ้ำซ้อนเหล่านี้ไป ดังนั้นกราฟในรูปที่ 4.9 จึงมีร้อยละของปริมาณพื้นที่

ครอบคลุมลดต่ำลงเหลือแค่เพียงประมาณร้อยละ 80 ของพื้นที่เดิมเท่านั้น ซึ่งร้อยละ 80 ของพื้นที่นี้จะถูกครอบคลุมโดยโหมดส่วนที่เหลือซึ่งยังมีปริมาณพลังงานเหลือพอที่จะทำงานอยู่

ถึงแม้ว่าปริมาณพื้นที่ครอบคลุมของระบบจะเริ่มลดต่ำลงเรื่อยๆเมื่อโหมดบางส่วนได้หมดพลังงานลงไป แต่เนื่องจากการที่อัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมสามารถหลับโหมดลงไปได้ในปริมาณมากกว่าแบบวิธีตัวจับเวลาแบบสุ่มเมื่อเทียบในแต่ละรอบการทำงาน ทำให้โดยเฉลี่ยแล้วอายุการทำงานของโหมดจะอยู่ได้นานขึ้น ดังนั้นเมื่อเวลาผ่านไปโอกาสที่จะหลงเหลือโหมดทำงานอยู่ในระบบตามแบบวิธีเซตครอบคลุมจะมีมากกว่าวิธีตัวจับเวลาแบบสุ่ม ปริมาณพื้นที่ที่ระบบครอบคลุม (ซึ่งส่วนหนึ่งขึ้นอยู่กับจำนวนโหมดที่เหลือในระบบ) เมื่อเทียบ ณ เวลาเดียวกันก็จะมากกว่าตามไปด้วย ดังนั้นเส้นกราฟแสดงปริมาณพื้นที่ตรวจจับที่ระบบครอบคลุมเทียบ ณ นาทีที่ 30 เป็นต้นไปของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมจะอยู่เหนือเฉลี่ยประมาณร้อยละ 10-15 ของแบบวิธีตัวจับเวลาแบบสุ่ม และจะมีค่าสิ้นสุดเป็น 0 (โหมดทุกตัวในระบบพลังงานหมด) ในนาทีที่ 60 ของการทำงาน ซึ่งไกลกว่าวิธีตัวจับเวลาแบบสุ่มที่สิ้นสุด ณ นาทีที่ 40

4.5 ผลการเปรียบเทียบอายุการทำงานเฉลี่ยต่อโหมด

ในการเปรียบเทียบอายุการทำงานเฉลี่ยต่อโหมดจะแยกออกเป็น 2 ประเภทตามสถานะของโหมดได้แก่ โหมดซ้ำซ้อนซึ่งมีการตื่นหลับเป็นรอบๆและโหมดไม่ซ้ำซ้อนที่เปิดทำงานตลอดเวลา อายุการทำงานเฉลี่ยต่อโหมดเมื่อเทียบกับปริมาณพลังงานเริ่มต้นที่โหมดมีจะสามารถแสดงได้ดังกราฟในรูปที่ 4.10 โดยที่มีการกำหนดจำนวนโหมดทั้งหมดในระบบเป็น 100 โหมดและมีระยะเวลาการหลับของโหมดซ้ำซ้อนเป็น 15 นาที



รูปที่ 4.10 กราฟแสดงอายุการทำงานเฉลี่ยต่อโหมด

จากกราฟในรูปที่ 4.10 จะเห็นว่า แนวโน้มของโหมดทั้งแบบซ้ำซ้อนและไม่ซ้ำซ้อนจะมีอายุที่นานขึ้นเมื่อให้ค่าพลังงานเริ่มต้นของโหมดมีสูงขึ้น โดยเมื่อเทียบที่พลังงานเริ่มต้นต่อโหมดเท่ากัน อายุการทำงานของโหมดซ้ำซ้อนที่มีการหลับเพื่อประหยัดพลังงานเป็นระยะจะอยู่ได้ยาวนานกว่าโหมดไม่ซ้ำซ้อนที่เปิดทำงานตลอดเวลา นอกจากนี้โหมดซ้ำซ้อนที่ทำงานอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมจะสามารถยืดอายุการทำงานเฉลี่ยต่อโหมดออกไปได้นานกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มประมาณร้อยละ 2 4 20 และ 25 เมื่อพลังงานเริ่มต้นต่อโหมดเป็น 20,000 mA 30,000 mA 40,000 mA และ 50,000 mA ตามลำดับ และมีแนวโน้มเพิ่มสูงขึ้นหากโหมดมีพลังงานเริ่มต้นมากกว่า 50,000 mA ขึ้นไป ซึ่งจากผลการทดลองนี้สามารถวิเคราะห์ได้ว่า เนื่องจากในแต่ละรอบการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมจะอาศัยโหมดซ้ำซ้อนในจำนวนที่น้อยกว่าวิธีการใช้ตัวจับเวลาแบบสุ่มสำหรับการทำงาน ดังนั้นจำนวนรอบที่โหมดจำเป็นต้องตื่นมาทำงานก็จะน้อยกว่าตามไปด้วย อายุการทำงานเฉลี่ยต่อโหมดจึงมากขึ้นกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม อย่างไรก็ตามหากพิจารณาที่พลังงานเริ่มต้นต่อโหมดเป็น 10,000 mA จะเห็นว่าอายุการทำงานเฉลี่ยต่อโหมดของวิธีเซตครอบคลุมจะสั้นกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม เนื่องจากในวิธีเซตครอบคลุมพลังงานทั้งหมดของโหมด (ซึ่งมีอยู่แค่ 10,000 mA) ถูกใช้ไปกับการคำนวณในรอบแรก ทำให้ไม่มีพลังงานเหลือสำหรับการใช้ในช่วงหลับเพื่อประหยัดพลังงาน อายุจึงสั้นกว่าวิธีการใช้ตัวจับเวลาแบบสุ่ม

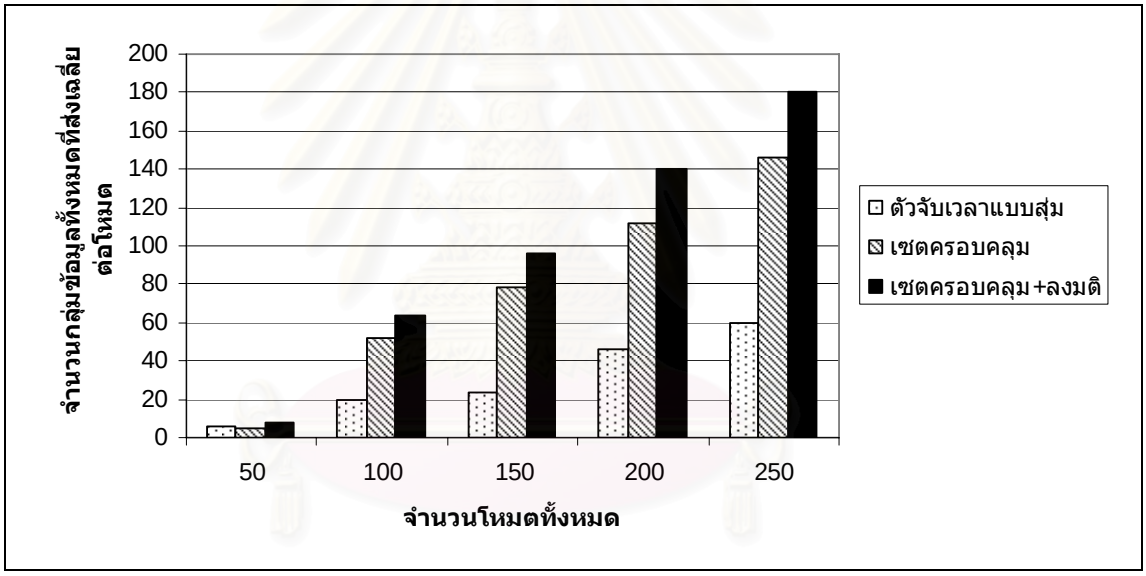
สำหรับวิธีการตัวจับเวลาแบบจำกัดขอบเขตสุ่มจะสามารถช่วยยืดขยายอายุการทำงานของโหมดซ้ำซ้อนขึ้นได้เล็กน้อยประมาณร้อยละ 10 เมื่อเทียบกับวิธีการตัวจับเวลาแบบสุ่ม เนื่องจากมีการกระจายความน่าจะเป็นให้โหมดที่มีปริมาณพลังงานเหลือมากได้มีโอกาสถูกเลือกขึ้นมาทำงานก่อนโหมดที่เหลือพลังงานน้อยเสมอ ทำให้ภาระการทำงานและการใช้พลังงานรวมของระบบสามารถเฉลี่ยกระจายออกไปทุกๆโหมดได้ เช่นเดียวกันกับกรณีที่มีการผสมการลงมติเข้าไปในวิธีการเซตครอบคลุมซึ่งสามารถช่วยหลับโหมดลงได้เพิ่มจากวิธีการเซตครอบคลุมเล็กน้อย ทำให้สามารถยืดขยายอายุการทำงานของโหมดซ้ำซ้อนออกไปได้เพิ่มอีกประมาณร้อยละ 5 เมื่อเทียบกับวิธีการเซตครอบคลุม

ข้อสังเกตอย่างหนึ่งที่ได้จากการทดลองนี้คือเมื่อเทียบที่ปริมาณพลังงานเริ่มต้นเท่ากัน โหมดที่ไม่ได้ทำงานอัลกอริทึมควบคุมความหนาแน่นและตรวจสอบออกมาได้ว่ามีสถานะไม่ซ้ำซ้อนจะมีอายุการทำงานเฉลี่ยออกมาสั้นกว่ากรณีโหมดที่ไม่ได้ทำงานอัลกอริทึมควบคุมความหนาแน่นใดๆ เหตุเพราะว่าในอัลกอริทึมควบคุมความหนาแน่น โหมดจะมีดึงเอาพลังงานส่วนหนึ่งมาใช้ในการตรวจสอบสถานะซ้ำซ้อน ซึ่งตามปกติแล้วสำหรับโหมดซ้ำซ้อน โอเวอร์เฮดด้านพลังงานส่วนนี้จะถูกชดเชยด้วยการหลับเป็นระยะๆ แต่สำหรับโหมดไม่ซ้ำซ้อนที่ต้องเปิดทำงานตลอดเวลาแล้วนั้น พลังงานส่วนนี้จะไม่ถูกชดเชยแต่อย่างใด ทำให้สามารถวิเคราะห์ได้ว่าอัลกอริทึมควบคุมความหนาแน่นจะส่งผลดีในการยืดอายุการทำงานของโหมด

ซ้ำซ้อนได้ แต่อาจส่งผลเสียต่อโหมดที่ไม่ซ้ำซ้อนด้วย อย่างไรก็ตามเมื่อเทียบระหว่างอายุที่ขยายได้ของโหมดซ้ำซ้อน (ประมาณร้อยละ 56-82) กับอายุที่สั้นลงของโหมดไม่ซ้ำซ้อน (ประมาณร้อยละ 28 ของอายุโหมดกรณีไม่มีอัลกอริทึมควบคุม) แล้วถือได้ว่าคุ้มค่าแก่การใช้งาน

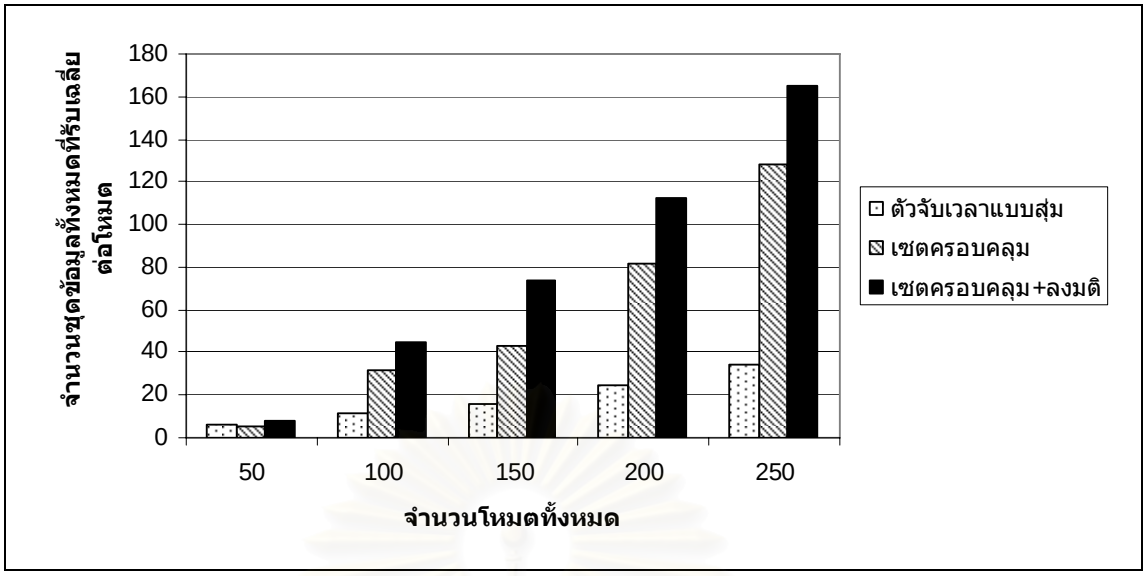
4.6 ผลการเปรียบเทียบจำนวนกลุ่มข้อมูลที่รับ-ส่งเฉลี่ยต่อโหมด

ในหัวข้อนี้จะทำการทดลองเปรียบเทียบโอเวอร์เฮดที่เพิ่มขึ้นจากการทำงานของอัลกอริทึมควบคุมความหนาแน่นจากวิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีการเซตครอบคลุม รวมทั้งวิธีการเซตครอบคลุมผสมการลงมติ โดยพิจารณาที่ปริมาณกลุ่มข้อมูลที่มีการรับและส่งรวมเฉลี่ยต่อโหมดหลังสิ้นสุดกระบวนการทำงานของอัลกอริทึมเทียบกับจำนวนโหมดที่มีในระบบทั้งหมด โดยผลที่ออกมาจะแสดงได้ดังแผนภูมิในรูปที่ 4.11 และ 4.12 ตามลำดับ



รูปที่ 4.11 กราฟแสดงจำนวนกลุ่มข้อมูลที่รับเฉลี่ยต่อโหมด

จากกราฟในรูปที่ 4.11 และ 4.12 จะเห็นว่า โอเวอร์เฮดด้านปริมาณของกลุ่มข้อมูลที่รับและส่งเฉลี่ยต่อโหมดจากการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมผสมการลงมติเมื่อเทียบที่จำนวนโหมดทั้งหมดเท่ากันจะมีค่าสูงกว่าวิธีการตัวจับเวลาแบบสุ่มอยู่ประมาณ 2-4 เท่า และสูงกว่าวิธีการเซตครอบคลุมที่ไม่มีการลงมติด้วย เนื่องจากว่าในการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีการเซตครอบคลุมผสมการลงมตินั้นจะมีโอเวอร์เฮดจากการรับและส่งกลุ่มข้อมูลเซตรวมทั้งผลมติระหว่างโหมดเพื่อนำไปใช้ในการคำนวณคำตอบ ซึ่งต่างจากวิธีการใช้ตัวจับเวลาแบบสุ่มที่ไม่มีโอเวอร์เฮดในส่วนนี้ จากกราฟเมื่อ



รูปที่ 4.12 กราฟแสดงจำนวนกลุ่มข้อมูลที่ส่งเฉลี่ยต่อโหมต

พิจารณาจากแนวโน้มที่โอเวอร์เฮดด้านปริมาณกลุ่มข้อมูลที่รับส่งมีจำนวนเพิ่มสูงขึ้นอย่างมาก เมื่อโหมตในระบบมีจำนวนเพิ่มขึ้น ทำให้สามารถวิเคราะห์ได้ว่าอัลกอริทึมแบบวิธีเซตครอบคลุมนี้อาจสามารถปรับขนาดได้ (Scalable) ตามจำนวนโหมตในระบบที่เพิ่มขึ้นไม่ดีเท่ากับวิธีใช้ตัวจับเวลาแบบสุ่ม

อย่างไรก็ตามสำหรับในกรณีที่ระบบมีการสูญเสียข้อมูลในการรับส่งระหว่างโหมต วิธีการใช้ตัวจับเวลาแบบสุ่มและวิธีเซตครอบคลุมอาจพบปัญหาเช่นเดียวกันในแง่ที่ว่า หากมีการหลับโหมตไปแล้วปรากฏว่าโหมตเพื่อนบ้านไม่รับรู้ เป็นต้น ทำให้อาจก่อให้เกิดช่องโหว่ของพื้นที่ครอบคลุมขึ้นในระบบก็เป็นได้

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

งานวิจัยนี้ได้นำเสนออัลกอริทึมควบคุมความหนาแน่นของโหมดสำหรับปัญหาพื้นที่ครอบคลุมในระบบเครือข่ายตัวรับรู้แบบไร้สาย เพื่อช่วยลดปริมาณการใช้พลังงานรวมของระบบ โดยอาศัยการเทียบปัญหาเข้ากับปัญหาเซตครอบคลุมพร้อมนำหลักวิธีการแก้ปัญหาของปัญหาเซตครอบคลุมมาประยุกต์ใช้ในอัลกอริทึม ซึ่งอัลกอริทึมที่ออกแบบจะมีการทำงานเป็นแบบกระจายเฉพาะที่และสามารถนำไปใช้งานได้ในระบบและสภาพแวดล้อมจริง จากผลการทดลองและวิจัยสามารถสรุปได้ดังนี้

5.1 สรุปผลการวิจัย

1) อัลกอริทึมควบคุมความหนาแน่นแบบวิธีตัวจับเวลาแบบสุ่มมีการสูญเสียพลังงานรวมของระบบไปมาก เนื่องจากว่าไม่มีระเบียบวิธีการในการเลือกหลับโหมด ทำให้จำนวนโหมดที่หลับได้มีปริมาณน้อยเกินไป ต่างจากแบบวิธีเซตครอบคลุมที่มีการเลือกหลับโหมดอย่างมีระเบียบตามกฎเกณฑ์ ซึ่งทำให้คำตอบของจำนวนโหมดที่หลับได้มีโอกาสเข้าใกล้จำนวนที่มากที่สุดที่เป็นไปได้ จึงมีปริมาณพลังงานรวมที่ระบบใช้น้อยกว่าแบบวิธีตัวจับเวลาแบบสุ่ม

2) การคำนวณและการแลกเปลี่ยนข้อมูลเพื่อให้ได้มาซึ่งคำตอบจากการแก้ปัญหาเซตครอบคลุมจำเป็นต้องมีการสูญเสียพลังงานไปในปริมาณมาก ดังนั้นระยะเวลาที่ระบบทำการหลับโหมดเพื่อประหยัดพลังงานควรที่จะยาวนานมากพอ เพื่อที่จะให้ประสิทธิภาพการทำงานของอัลกอริทึมสามารถแสดงผลการประหยัดพลังงานออกมาได้ชัดเจน

3) เนื่องจากในอัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุม โหมดซ้ำซ้อนมีความถี่การวนรอบหลับเพื่อประหยัดพลังงานในอัตราที่สูงกว่าแบบวิธีตัวจับเวลาแบบสุ่ม ส่งผลให้อายุการทำงานเฉลี่ยต่อโหมดในระบบจะยาวนานขึ้น รวมไปถึงตัวระบบเองสามารถรักษาการมีอยู่ของโหมดได้ในจำนวนมากขึ้นด้วยเช่นกัน

4) การรักษาพื้นที่ครอบคลุมของระบบจะแบ่งพิจาณาออกเป็น 2 ช่วง คือช่วงแรกที่โหมดทุกตัวยังมีพลังงานเหลือเพียงพอในการทำงาน ช่วงนี้อัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมจะสามารถรักษาพื้นที่ครอบคลุมเดิมเอาไว้ได้ทั้งหมด (ร้อยละ 100) เช่นเดียวกันกับวิธีตัวจับเวลาแบบสุ่ม อย่างไรก็ตามในช่วงต่อมาคือช่วงหลังจากที่โหมดไม่ซ้ำซ้อนบางตัวเริ่มตายไปจนเกิดพื้นที่บอดขึ้นในระบบ เมื่อเทียบที่เวลาเดียวกัน อัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมจะสามารถรักษาปริมาณพื้นที่ครอบคลุมเดิมของระบบไว้ได้มากกว่าแบบวิธีตัวจับเวลาแบบสุ่ม เนื่องจากโหมดมีอายุการทำงานที่นานขึ้น จึงมีจำนวนโหมดหลงเหลืออยู่ในระบบในปริมาณที่มากกว่า

5) เนื่องจากอัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมมีการนำใช้หลักการลงมติระหว่างโหมตข้างเคียงมาใช้ในการตัดสินคำตอบ ทำให้โอเวอร์เฮดด้านปริมาณกลุ่มข้อมูลที่รับส่งในระบบจะมีเพิ่มสูงขึ้นกว่าปกติและมากกว่ากรณีแบบวิธีตัวจับเวลาแบบสุ่ม

5.2 ข้อจำกัดและข้อเสนอแนะ

1) อัลกอริทึมที่นำเสนอเหมาะสำหรับการทำงานบนระบบเครือข่ายตัวรับรู้ที่มีความหนาแน่นในระดับกลาง เนื่องจากโอเวอร์เฮดด้านปริมาณข้อมูลที่จำเป็นต้องมีการรับ-ส่งเพื่อใช้ในการคำนวณนั้นแปรผันตามจำนวนโหมตเพื่อนบ้าน ทำให้อัลกอริทึมนี้อาจยังไม่เหมาะสำหรับการใช้งานในระบบที่มีความหนาแน่นสูง

2) อัลกอริทึมที่นำเสนอเหมาะสำหรับการทำงานบนระบบเครือข่ายตัวรับรู้ที่มีความถี่การวนรอบทำงานของอัลกอริทึมที่ไม่สูงนัก เนื่องด้วยปริมาณพลังงานที่ใช้ในการคำนวณคำตอบแต่ละรอบการทำงานนั้นยังคงมีค่าสูง ทำให้ไม่เหมาะสำหรับระบบที่มีการคำนวณใหม่อยู่บ่อยเกินไป

3) เนื่องด้วยในวิทยานิพนธ์นี้ โหมตแต่ละโหมตมีค่าพลังงานเริ่มต้นเท่ากันทุกโหมต ทำให้เมื่อพิจารณาที่การรักษาสภาพพื้นที่ครอบคลุมของระบบจะขึ้นอยู่กับโหมตที่ไม่ซ้ำซ้อนและเปิดทำงานตลอดเวลาเป็นสำคัญ ดังนั้นในการทดลองจะเห็นได้ว่าอัลกอริทึมไม่สามารถที่จะรักษาสภาพคงเดิมของพื้นที่ครอบคลุมเอาไว้ได้ตลอดเวลา โดยระบบจะเริ่มมีการสูญเสียพื้นที่ครอบคลุมไปภายหลังจากที่โหมตไม่ซ้ำซ้อนได้ตายไป อย่างไรก็ตามถึงแม้ว่าระบบจะไม่สามารถรักษาพื้นที่เอาไว้ได้ แต่ก็มีกระบวนการบางอย่างที่สามารถนำเข้ามาใช้ช่วยในการรักษาปริมาณพื้นที่เดิมเอาไว้ได้ให้มากที่สุดเท่าที่จะทำได้ด้วย เช่น ในระหว่างรอบการทำงานของอัลกอริทึมปกติ โหมตแต่ละโหมตที่หลับจะทราบว่าโหมตเพื่อนบ้านที่เปิดทำงานรอบ ๆตัวมันแต่ละตัวมีค่าพลังงานเหลือมากน้อยเพียงใดบ้าง ซึ่งจากการที่รู้ปริมาณพลังงานที่เหลือนี้ทำให้โหมตที่หลับสามารถที่จะประมาณอายุเวลาที่เหลืออยู่ของโหมตเพื่อนบ้านแต่ละตัวได้ หากพบว่ามีโหมตเพื่อนบ้านตัวใดที่เหลือพลังงานน้อยมากจนไม่สามารถที่จะอยู่ทำงานได้ครบตามกำหนดเวลาในแต่ละรอบ โหมตที่หลับนี้จะทำการตั้งค่าเวลาเพื่อปลุกตัวเองขึ้นมาทำงานครอบคลุมพื้นที่ทดแทนในจังหวะเวลาที่โหมตเพื่อนบ้านได้ตายไปได้ หรืออีกวิธีหนึ่งคือเปลี่ยนการทำงานของอัลกอริทึมจากเดิมที่เป็นรอบ ๆให้เหลือแค่รอบเดียว โหมตที่ถูกเลือกให้เปิดทำงานจะทำงานไปจนกว่าจะหมดพลังงานแล้วค่อยแจ้งเตือนโหมตอื่นที่หลับอยู่ให้ตื่นมาทำงานแทน เป็นต้น

4) อัลกอริทึมที่นำเสนอจะอาศัยหลักการประมาณคำตอบของปัญหาเซตครอบคลุมมาช่วยในการทำงาน ดังนั้นอัลกอริทึมนี้จึงยังคงไม่ใช่อัลกอริทึมที่ดีที่สุด (Optimized algorithm) ที่จะ

สามารถลดพลังงานลงได้มากที่สุด แต่สามารถที่จะนำไปใช้เป็นต้นแบบแนวคิดในการพัฒนางานวิจัยอื่นๆต่อไปได้

5) ผลการตรวจสอบความซ้ำซ้อนของโหมดที่ได้จากโปรแกรมสำหรับทดสอบอัลกอริทึมอาจมีค่าผิดพลาดบ้างบางกรณี เนื่องจากข้อจำกัดของภาษาที่ใช้เขียนโปรแกรมที่ยังไม่มีฟังก์ชันสนับสนุนให้ใช้ในการคำนวณทางคณิตศาสตร์ ค่าที่ได้จากการคำนวณจึงเป็นค่าประมาณเชิงตัวเลขที่ยังคงมีร้อยละของความคลาดเคลื่อนจากทางทฤษฎีอยู่เล็กน้อย

6) กระบวนการทำงานของอัลกอริทึมควบคุมความหนาแน่นแบบวิธีเซตครอบคลุมจำเป็นต้องมีการส่งข้อมูลแบบกระจายระหว่างโหมดในปริมาณมาก ซึ่งเมื่อพิจารณาเวลาที่นำเอาไปใช้งานในสภาวะแวดล้อมจริงที่มีสื่อรบกวนอยู่ตลอดเวลาแล้ว กลุ่มข้อมูลบางส่วนอาจเกิดการสูญเสียหายได้ ดังนั้นสำหรับในกรณีที่ระบบมีการสูญหายของข้อมูลในระหว่างรับส่งจะเป็นสิ่งจำเป็นที่ต้องพิจารณาแก้ไขต่อไปสำหรับงานวิจัยในอนาคต

5.3 ปัญหาและอุปสรรค

1) เนื่องด้วยเทคโนโลยีระบบเครือข่ายไร้สายถือได้ว่าเป็นเทคโนโลยีใหม่ที่เกิดขึ้นมาไม่นานนัก สิ่งตีพิมพ์หรือข้อมูลสำคัญที่มีให้ศึกษาจึงยังคงมีอยู่น้อยในประเทศไทย อีกทั้งวิธีการและกระบวนการทำงานของระบบบางส่วนยังอยู่ในขั้นตอนกระบวนการวิจัย ข้อมูลส่วนใหญ่จะเป็นข้อมูลที่ได้รับทางอินเทอร์เน็ตหรือรายงานการวิจัยก่อนหน้าเกือบทั้งหมด ทำให้การศึกษาให้เข้าใจและดำเนินการวิจัยนั้นทำได้ยาก

2) การขาดแคลนตัวอุปกรณ์จริงที่ใช้ทำการทดลองจริง เนื่องจากราคาที่ค่อนข้างสูงและยังไม่มีจำหน่ายภายในประเทศไทย ในการทดลองเบื้องต้นจึงยังคงต้องทดลองบนโปรแกรมจำลองทำให้ยังไม่สามารถสรุปผลจากการทำงานบนระบบจริงได้

รายการอ้างอิง

[1] Estrin, D., Govindan, R., Heidemann, J., and Kumar, S. (1999). Next Century Challenges: Scalable Coordination in Sensor Networks. Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, Seattle, Washington, United States.

[2] Meguerdichian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M.B. (2001). Coverage Problems in Wireless Ad-hoc Sensor Networks. INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, pp. 1380-1387.

[3] Huang, C.F., and Tseng, Y.C. (2003). The Coverage Problem in a Wireless Sensor Network. Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications, San Diego, CA, USA.

[4] Tian, D., and Georganas, N.D. (2002). A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks. Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, Georgia, USA.

[5] Jiang, J., and Dou, W. (2004). A Coverage-Preserving Density Control Algorithm for Wireless Sensor Networks. Ad-Hoc, Mobile, and Wireless Networks, pp. 42-55.

[6] 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, Los Angeles, California, USA.

[7] Raghunathan, V., Schurgers, C., Park, S., and Srivastava, M.B. (2002). Energy-Aware Wireless Microsensor Networks. Signal Processing Magazine, IEEE 19: pp. 40-50.

[8] Ilyas, M., and Mahgoub, I. (2005). Handbook of Sensor Networks: Compact Wireless and Wired Sesing Systems. CRC Press.

[9] Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2001). Introduction to Algorithms, Second Edition. The MIT Press.

[10] Slijepcevic, S., and Potkonjak, M. (2001). Power Efficient Organization of Wireless Sensor Networks. Communications, 2001. ICC 2001. IEEE International Conference on, pp. 472-476.

[11] Zhang, H., and Hou, J.C. (2005). Maintaining Sensing Coverage and Connectivity in Large Sensor Networks. International Journal of Wireless Ad Hoc and Sensor Networks 1: pp. 89-124.

[12] Ye, F., Zhong, G., Cheng, J., Lu, S., and Zhang, L. (2003). PEAS: A Robust Energy Conserving Protocol for Long-lived Sensor Networks. Proceedings of the 23rd International Conference on Distributed Computing Systems.

[13] Heinzelman, W.R., Chandrakasan, A., and Balakrishnan, H. (2000). Energy-Efficient Communication Protocol for Wireless Microsensor Networks. System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on.

[14] Levis, P. (2003). TinyOS Tutorial, The tutorial introduces the basic concepts of TinyOS through a set of simple applications and exercises [Online]. Available from: http://www.tinyos.net/tinyos-1.x/doc/tutorial/index.html

[15] 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, San Diego, California, USA.

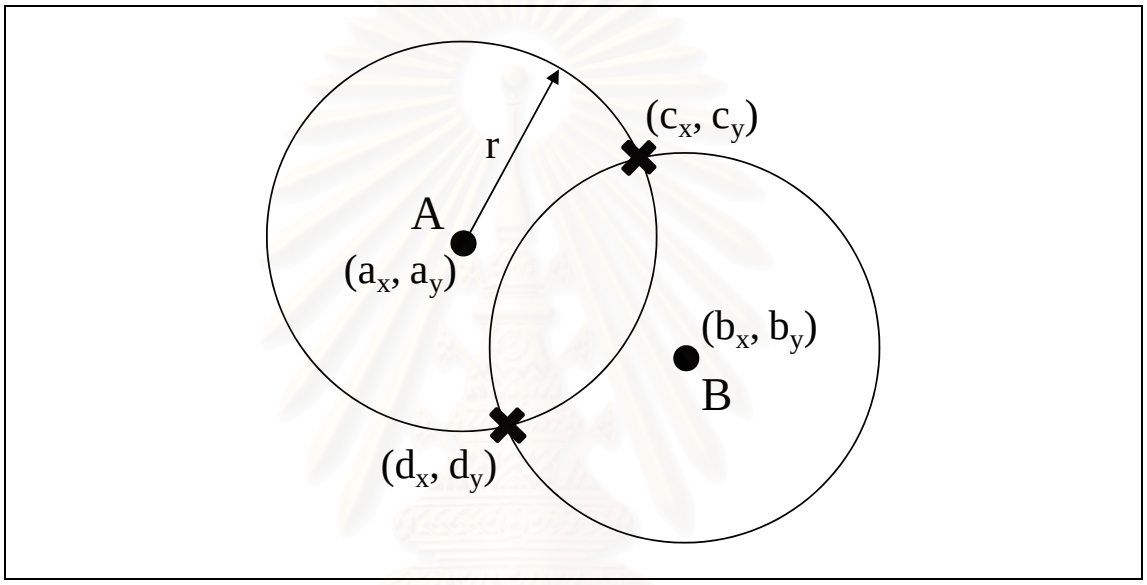
[16] Shnayder, V., Hempstead, M., Chen, B.R., Allen, G.W., and Welsh, M. (2004). Simulating the Power Consumption of Large-Scale Sensor Network Applications. Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, Baltimore, MD, USA.

ภาคผนวก

ภาคผนวก ก

การคำนวณจุดตัดของวงกลม

จากวงกลม 2 วงได้แก่วงกลม A และวงกลม B ที่มีรัศมีเท่ากันเท่ากับ r มีตำแหน่งของคู่ลำดับ ณ จุดศูนย์กลางวงกลม A และ B อยู่ที่ (a_x, a_y) และ (b_x, b_y) ตามลำดับ วงกลม 2 วงนี้มีการตัดกันที่จุด 2 จุดได้แก่จุดที่ตำแหน่ง (c_x, c_y) และ (d_x, d_y) ดังรูปที่ ก.1



รูปที่ ก.1 การตัดกันของวงกลม 2 วง

จากสมการวงกลมในสมการที่ ก.1

$$(x - x_c)^2 + (y - y_c)^2 = r^2 \qquad \text{(ก.1)}$$

เมื่อ x_c แทนตำแหน่งในแนวแกน X ของจุดศูนย์กลางวงกลม

y_c แทนตำแหน่งในแนวแกน Y ของจุดศูนย์กลางวงกลม

r แทนรัศมีวงกลม

จะได้ว่า สมการวงกลมของวงกลม A ในรูปที่ ก.1 เป็นดังสมการ ก.2

$$(x - a_x)^2 + (y - a_y)^2 = r^2 \qquad \text{(ก.2)}$$

และสมการวงกลมของวงกลม B ในรูปที่ ก.1 เป็นดังสมการ ก.3

$$(x-b_x)^2+(y-b_y)^2=r^2 \qquad \text{(ก.3)}$$

เนื่องจากสมการที่ ก.2 เท่ากับสมการที่ ก.3 จึงได้ว่า

$$(x-a_x)^2+(y-a_y)^2=(x-b_x)^2+(y-b_y)^2$$

$$x^2-2xa_x+a_x^{\,2}+y^2-2ya_y+a_y^{\,2}=x^2-2xb_x+b_x^{\,2}+y^2-2yb_y+b_y^{\,2}$$

$$-2xa_x+a_x^{\,2}-2ya_y+a_y^{\,2}=-2xb_x+b_x^{\,2}-2yb_y+b_y^{\,2}$$

$$(-2a_x+2b_x)x=(-2b_y+2a_y)y-a_x^{\,2}-a_y^{\,2}+b_x^{\,2}+b_y^{\,2}$$

$$x=\frac{(-2b_y+2a_y)y-a_x^{\,2}-a_y^{\,2}+b_x^{\,2}+b_y^{\,2}}{-2a_x+2b_x}$$

$$x=\frac{a_y-b_y}{b_x-a_x}y+\frac{b_x^{\,2}+b_y^{\,2}-a_x^{\,2}-a_y^{\,2}}{2(b_x-a_x)} \qquad \text{(ก.4)}$$

หรือหากเขียนในรูปแบบสมการเส้นตรง จะได้เป็นสมการที่ ก.5

$$x=my+t \qquad \text{(ก.5)}$$

เมื่อ m แทน $\dfrac{a_y-b_y}{b_x-a_x}$

t แทน $\dfrac{b_x^{\,2}+b_y^{\,2}-a_x^{\,2}-a_y^{\,2}}{2(b_x-a_x)}$

นำสมการที่ ก.5 ไปแทนค่าในสมการที่ ก.2

จะได้ว่า

$$(my+t-a_x)^2+(y-a_y)^2=r^2$$

$$m^2y^2+2my(t-a_x)+(t-a_x)^2+y^2-2ya_y+a_y^2=r^2$$

$$(m^2+1)y^2+2(m(t-a_x)-a_y)y+(t-a_x)^2+a_y^2-r^2=0$$

จากสูตรการแก้สมการ 2 ตัวแปรในสมการที่ ก.6

$$y=\frac{-b\pm\sqrt{b^2-4ac}}{2a} \qquad \text{เมื่อ} \qquad ay^2+by+c=0 \qquad \text{(ก.6)}$$

จะได้ว่า

$$y = \frac{-2(m(t-a_x)-a_y) \pm \sqrt{(2(m(t-a_x)-a_y))^2 - 4(m^2+1)((t-a_x)^2 + a_y^2 - r^2)}}{2(m^2+1)}$$

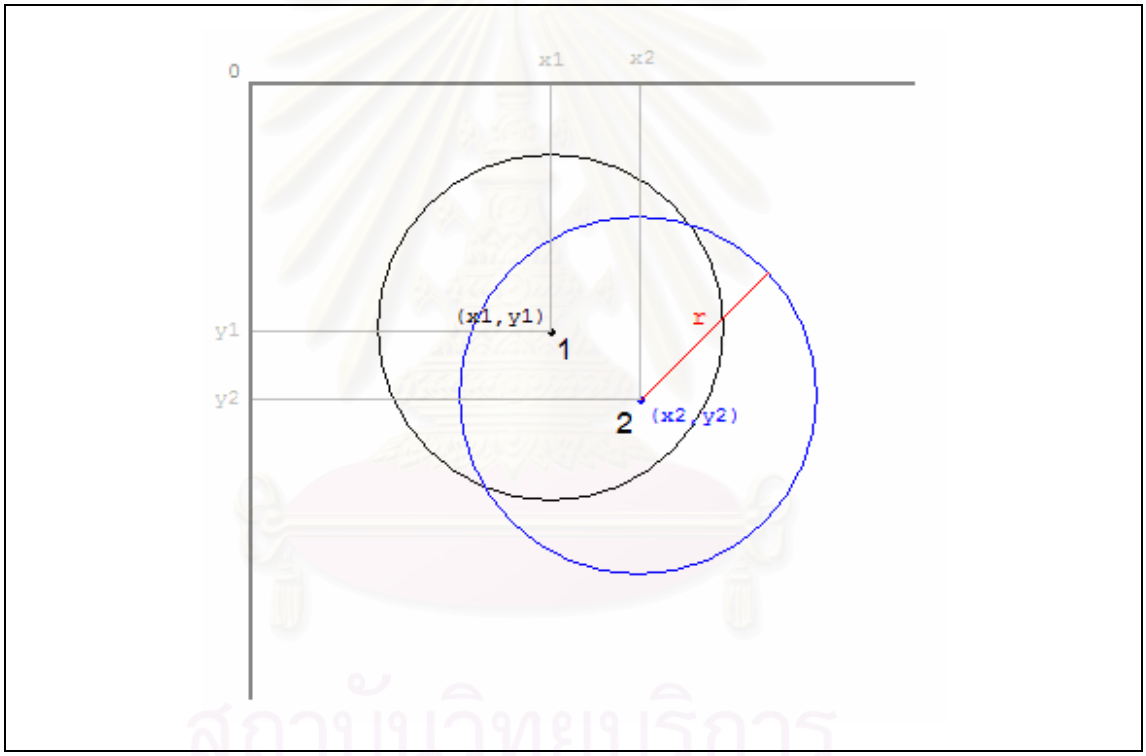
ซึ่งค่าของ y ที่ได้จะมี 2 ค่าด้วยกัน นั่นคือค่าหนึ่งจะเป็นค่า c_y และอีกค่าจะเป็น d_y นำค่า c_y และ d_y ไปแทนกลับในสมการที่ ก.5 จะได้ค่า x ออกมา 2 ค่าเช่นกัน ซึ่งค่าหนึ่งจะเป็นค่า c_x และอีกค่าจะเป็น d_x นั่นเอง ทำให้เราได้ตำแหน่งจุดตัดทั้ง 2 จุดเป็น (a_x, a_y) และ (b_x, b_y) เรียบร้อยแล้ว

ภาคผนวก ข

การคำนวณส่วนทับซ้อนของเส้นรอบวงกลมแทนพื้นที่ตรวจจับ

การคำนวณส่วนทับซ้อนของเส้นรอบวงกลมแทนพื้นที่ตรวจจับระหว่าง 2 โหมตใดๆ จะมีขั้นตอนทำงานพร้อมตัวอย่างดังนี้

1) กำหนดให้โหมต 1 เป็นโหมตอ้างอิง มีตำแหน่งอยู่ที่จุด (x_1, y_1) และโหมต 2 แทนโหมตเพื่อนบ้านของโหมต 1 ที่มีอาณาบริเวณพื้นที่ตรวจจับทับซ้อนกับพื้นที่ของโหมต 1 มีตำแหน่งอยู่ที่จุด (x_2, y_2) ทั้ง 2 โหมตมีค่ารัศมีการตรวจจับเท่ากับ r ดังรูปที่ ข.1



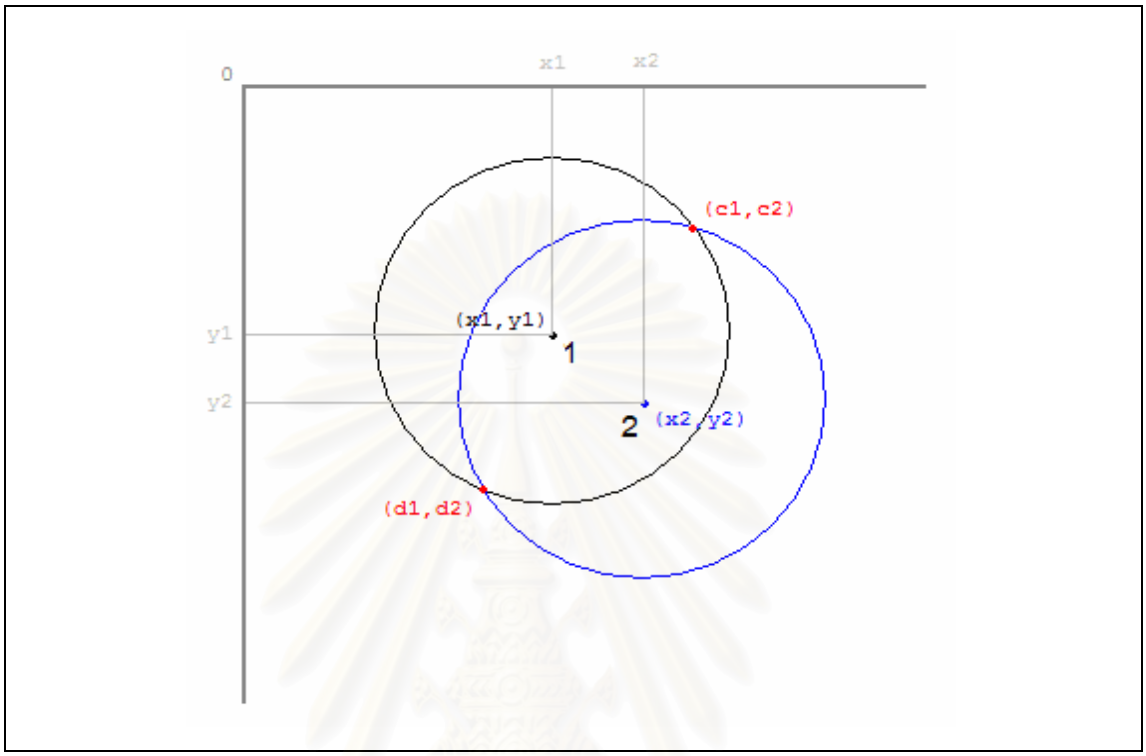
รูปที่ ข.1 การซ้อนทับพื้นที่ของวงกลมในระนาบ 2 มิติ

2) จากตำแหน่งจุดศูนย์กลางวงกลมทั้งสองและค่ารัศมีวงกลม จะสามารถคำนวณหาสมการวงกลมแทนพื้นที่ของโหมต 1 และ 2 ออกมาได้ดังสมการที่ ข.1 และ ข.2 ตามลำดับ

$$(x - x_1)^2 + (y - y_1)^2 = r^2 \qquad \text{(ข.1)}$$

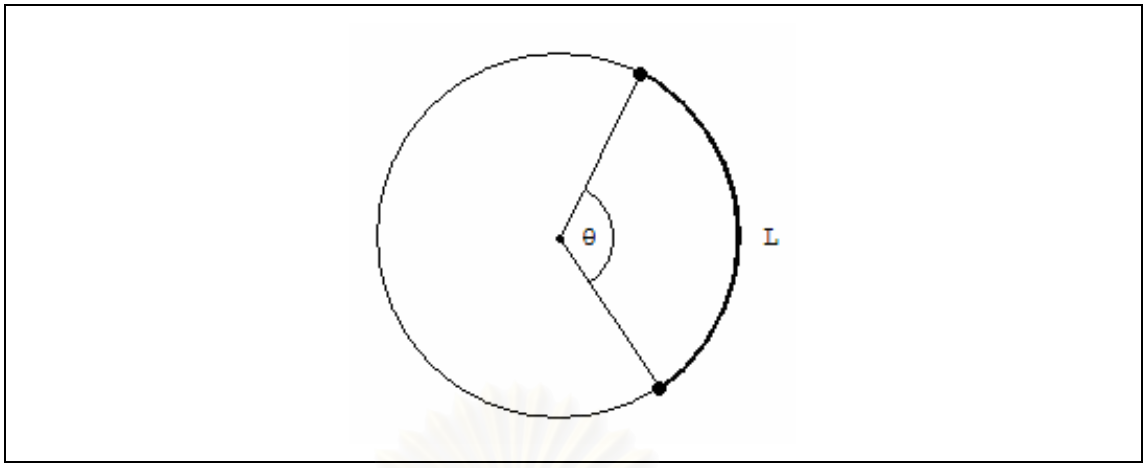
$$(x - x_2)^2 + (y - y_2)^2 = r^2 \qquad \text{(ข.2)}$$

3) จากสมการที่ ข.1 และ ข.2 ทำการหาจุดตัดของวงกลม (c_1, c_2) และ (d_1, d_2) ดังรูปที่ ข.2 โดยอาศัยการแก้สมการวงกลม 2 ตัวแปร ดูรายละเอียดได้ในภาคผนวก ก)



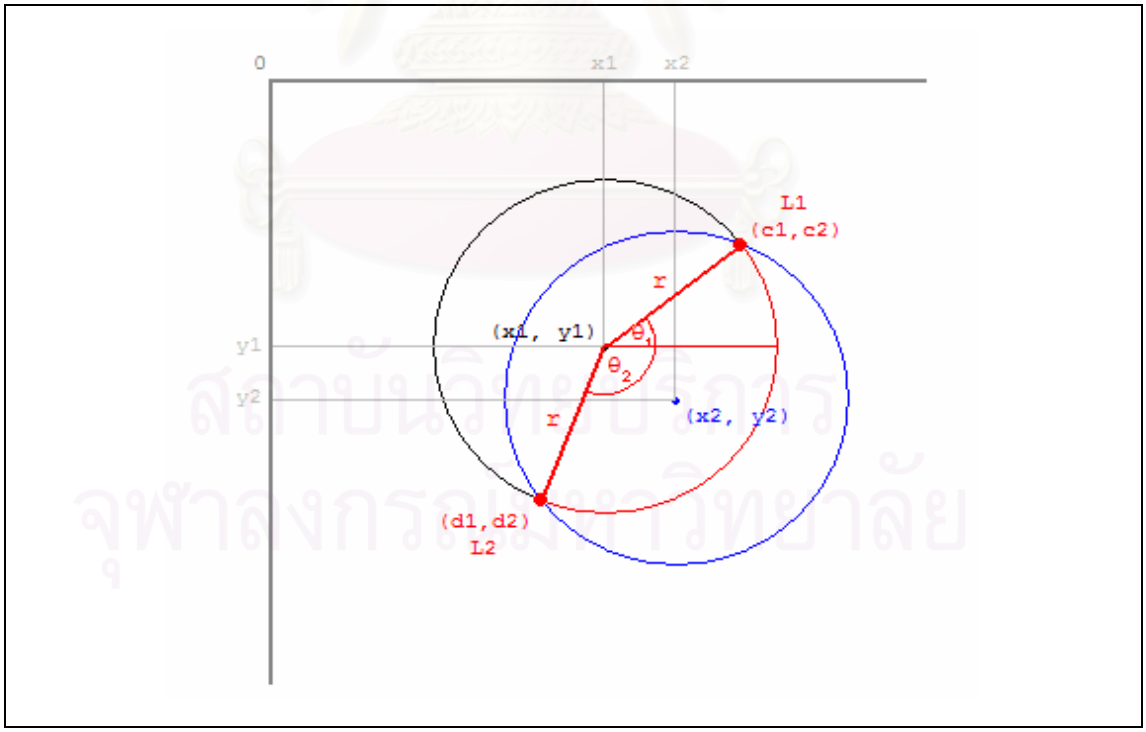
รูปที่ ข.2 ตำแหน่งจุดตัดของวงกลม 2 วง

4) ส่วนของเส้นรอบวงโหมต 1 จากจุดตัด (c_1, c_2) ถึง (d_1, d_2) จะเป็นส่วนของเส้นรอบวงที่โหมต 2 ครอบคลุมโหมต 1 เนื่องจากในการคำนวณจริงการนำส่วนของเส้นรอบวงมาคิดโดยตรงจะค่อนข้างยุ่งยากมาก ดังนั้นจะต้องมีการแทนส่วนของเส้นรอบวงนี้ด้วยตัวแปรอื่นที่สามารถนำมาคิดคำนวณได้สะดวกกว่า ซึ่งจากความสัมพันธ์ระหว่างเส้นรอบวงกับมุม 360 องศา ส่วนของเส้นรอบวงจะถูกแทนด้วยมุมที่ทำระหว่างจุดตัดทั้งสอง ดังเช่นตัวอย่างในรูปที่ ข.3 ส่วนของเส้นรอบวง L จะถูกแสดงแทนได้ด้วยมุม θ



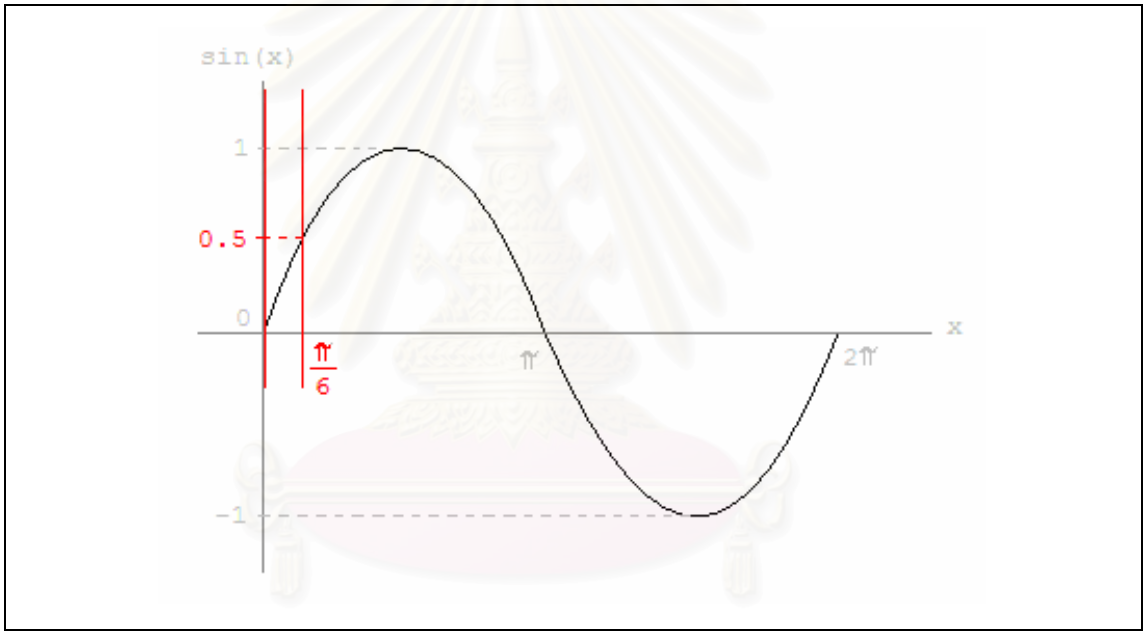
รูปที่ ข.3 ความสัมพันธ์ระหว่างส่วนของเส้นรอบวง L และมุมแทนส่วนของเส้นรอบวง θ

จากรูปที่ ข.2 จะได้ว่าส่วนของเส้นรอบวงที่โหมต 2 ครอบคลุมเส้นรอบวงของโหมต 1 จะถูกแทนด้วยมุม $\theta_1 + \theta_2$ เมื่อมุม θ_1 และ θ_2 แทนมุมที่ทำระหว่างเส้นตรงที่ลากจากจุด (x_1, y_1) ไปยังจุดตัด (c_1, c_2) กับแนวแกน x และมุมที่ทำระหว่างเส้นตรงที่ลากจาก (x_1, y_1) ไปยังจุดตัด (d_1, d_2) กับแนวแกน x ตามลำดับ ดังเช่นแสดงไว้ในรูปที่ ข.4



รูปที่ ข.4 มุมแทนส่วนของเส้นรอบวง θ_1 และ θ_2 กรณีมีจุดตัดวงกลม 2 จุด

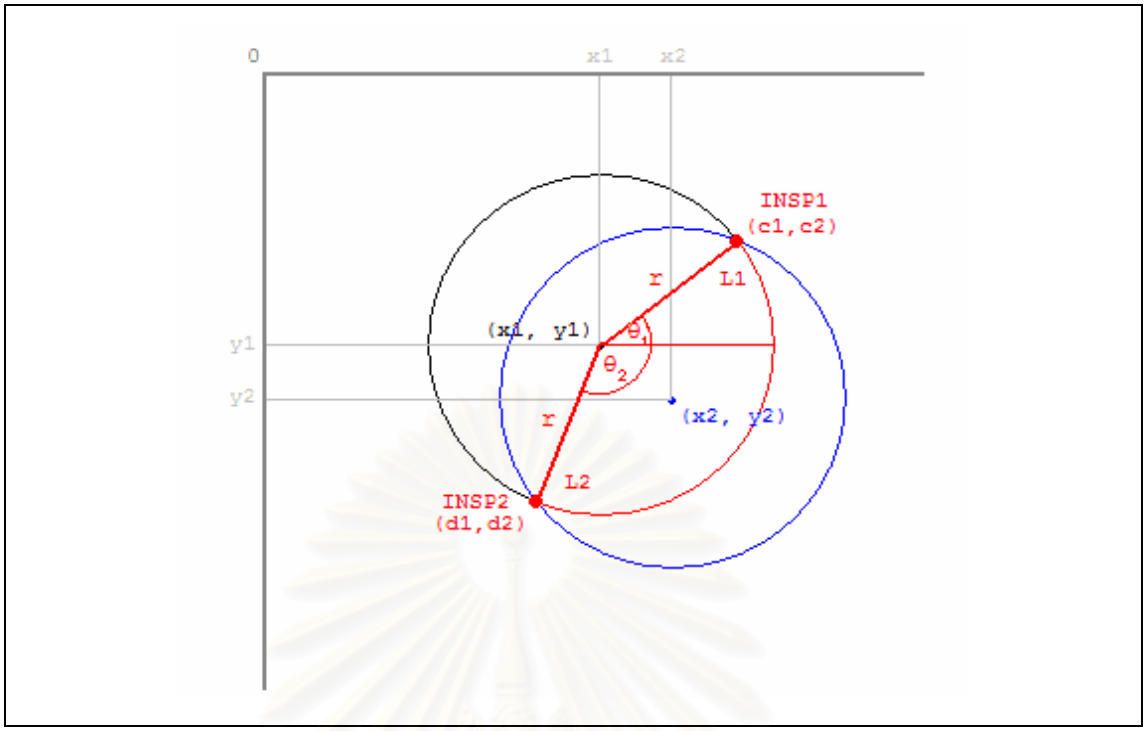
โดยทั่วไปแล้วค่ามุม θ แทนส่วนของเส้นรอบวงนี้จะสามารถหาได้จากค่า arctan ของความชันเส้นตรงที่ลากระหว่างจุดศูนย์กลางวงกลมกับจุดตัด แต่เนื่องจากในคลังฟังก์ชัน (Function library) ของภาษาเนสซีนั้นไม่มีฟังก์ชัน arctan ที่ใช้คำนวณทางด้านตรีโกณมิติให้เรียกใช้ ดังนั้นแทนที่จะพยายามใช้ค่ามุม θ ในการแสดงส่วนของเส้นรอบวงโดยตรง จะนำช่วงของค่า sin แทนมุมนั้นๆมาใช้แทน โดยที่จากมุม $[0,2\pi]$ ของวงกลมหนึ่งวง จะสามารถแทนด้วยช่วงของค่า $\sin[0,1]$ $\sin[1,0]$ $\sin[0,-1]$ และ $\sin[-1,0]$ ตามจตุภาค (Quadrant) ของวง ยกตัวอย่างเช่น มุมที่แทนส่วนของเส้นรอบวงเป็น $[0,\frac{\pi}{6}]$ จะสามารถแทนได้ด้วยช่วงค่า $\sin(0)$ ถึง $\sin(\frac{\pi}{6})$ ซึ่งเท่ากับ $\sin[0,0.5]$ เมื่อนำมาเขียนเป็นกราฟเปรียบเทียบจะได้กราฟดังรูปที่ ข.5



รูปที่ ข.5 กราฟเปรียบเทียบความสัมพันธ์ระหว่างช่วงค่า sin กับมุม θ

5) เมื่อพิจารณาที่การทับซ้อนวงกลมจากโหมดเพื่อนบ้านตัวหนึ่ง จะพบว่าจุดตัดจากวงกลม 2 วงที่เกิดขึ้นจะมีได้ไม่เกิน 2 จุดเสมอ เพราะฉะนั้นช่วงของค่า sin ที่แทนส่วนของเส้นรอบวงที่ถูกครอบคลุมนั้นจะต้องอยู่ภายในช่วงของค่า sin 2 ค่าเสมอ โดยค่า sin ค่าหนึ่งจะคู่กับจุดตัด 1 จุด จากตัวอย่างในรูปที่ ข.4 จะหาค่า sin จากจุดตัดได้ดังนี้

5.1) ทำการลากเส้น L_1 และ L_2 เชื่อมระหว่างจุด (x_1,y_1) กับจุดตัด (c_1,c_2) และ (x_1,y_1) กับจุดตัด (d_1,d_2) ตามลำดับ ดังเช่นในรูปที่ ข.6

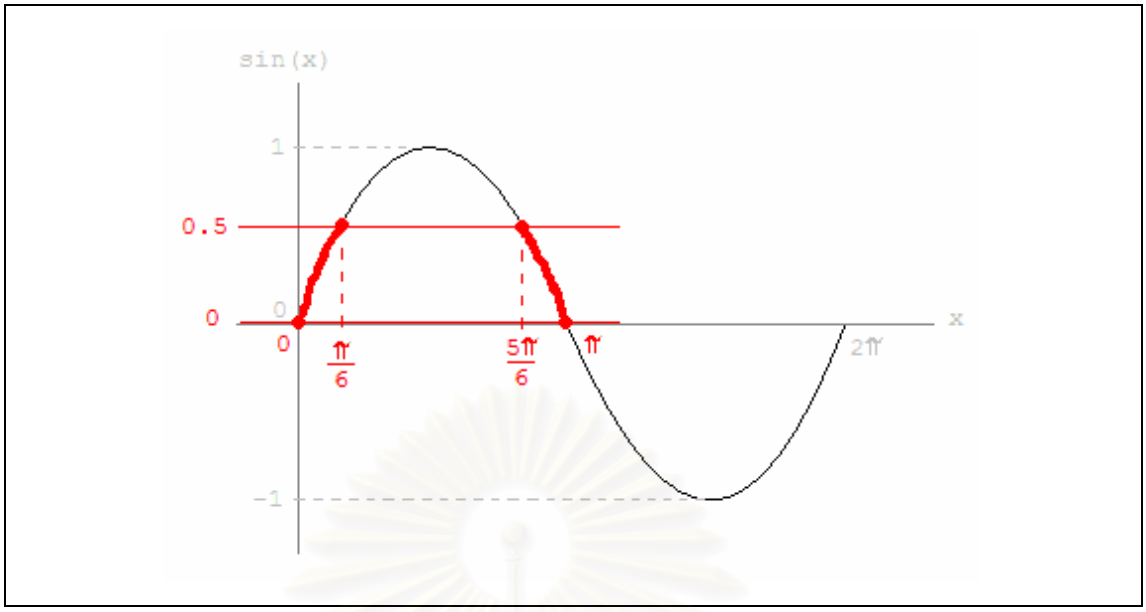


รูปที่ ข.6 เส้นตรงที่ลากระหว่างจุดตัดกับศูนย์กลางวงกลม

5.2) มุม θ_1 และ θ_2 จะแทนมุมที่ L_1 และ L_2 ทำกับแนวแกน x ตามลำดับ จากสูตรการหาค่า sin จะได้ว่า มุม $\theta_1 + \theta_2$ นี้จะสามารถแทนได้ด้วยช่วงของค่า sin ตั้งแต่ $\sin(\theta_1)$ ถึง $\sin(\theta_2)$ ซึ่งเท่ากับ $\sin[\frac{c_2 - y_1}{r}, \frac{d_2 - y_1}{r}]$

5.3) ผลลัพธ์ที่ได้จะบอกว่า ส่วนของเส้นรอบวงที่โหมต 1 โดนโหมต 2 ทับ จะคือช่วงของค่า $\sin[\frac{c_2 - y_1}{r}, \frac{d_2 - y_1}{r}]$

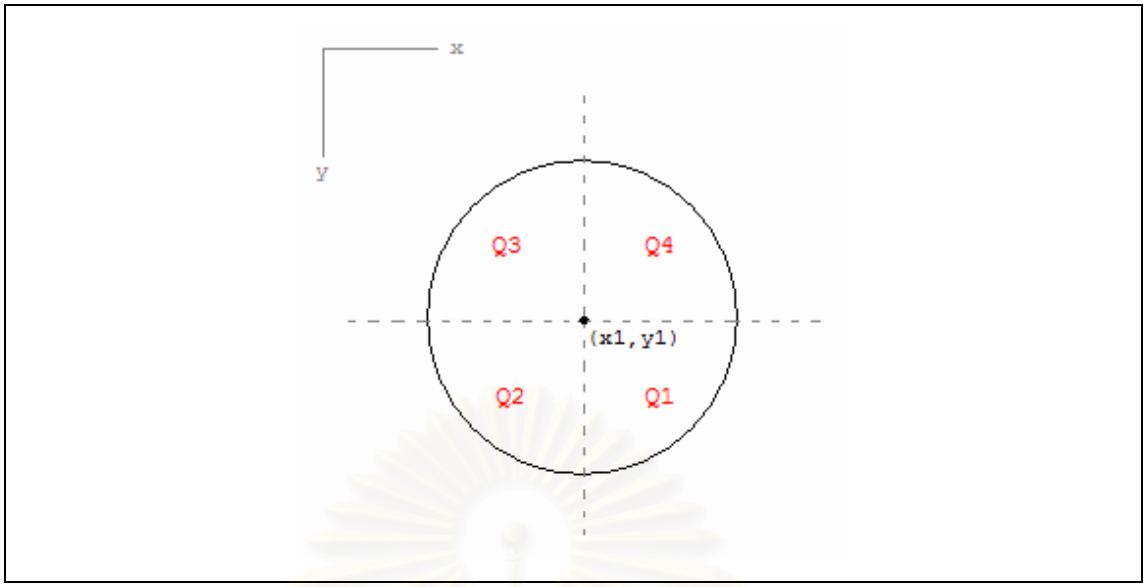
อย่างไรก็ตามเนื่องจากช่วงของค่า sin หนึ่งๆ อาจสามารถถูกตีความหมายแปลงกลับไปเป็นมุมได้หลายมุมที่ต่างกัน ส่งผลให้อาจเกิดข้อผิดพลาดในการแทนส่วนของเส้นรอบวงได้ ตัวอย่างเช่น ช่วงของค่า $\sin[0, 0.5]$ จะสามารถตีความออกมาได้เป็น 2 มุมที่ต่างกัน คือ $[0, \frac{\pi}{6}]$ และ $[\frac{5\pi}{6}, \pi]$ ตามรูปที่ ข.7



รูปที่ ข.7 ช่วงของค่า sin ที่สามารถแปลงเป็นมุมได้มากกว่า 1 มุม

เพราะฉะนั้นช่วงของค่า sin เพียงอย่างเดียวจึงไม่เพียงพอในการแทนส่วนของเส้นรอบวงของโหมต จะต้องมีข้อมูลอื่นเข้ามาช่วยด้วย ได้แก่ จตุภาคของตำแหน่งโหมตเพื่อนบ้านและจุดตัดของวงกลมทั้งสอง เมื่อใช้จุดศูนย์กลางของโหมตเป็นจุดอ้างอิง ลักษณะการวางตำแหน่งของโหมตเพื่อนบ้านและจุดตัดจะสามารถบ่งชี้มุมที่แทนส่วนของเส้นรอบวงได้แม่นยำขึ้น ในรูปที่ ข.8 แสดงจตุภาคทั้ง 4 ภาคของพื้นที่วงกลมและมี Q1 Q2 Q3 และ Q4 แทนจตุภาคที่ 1 2 3 และ 4 ตามลำดับ โดยที่แต่ละภาคจะมีมุมและช่วงของค่า sin ดังนี้

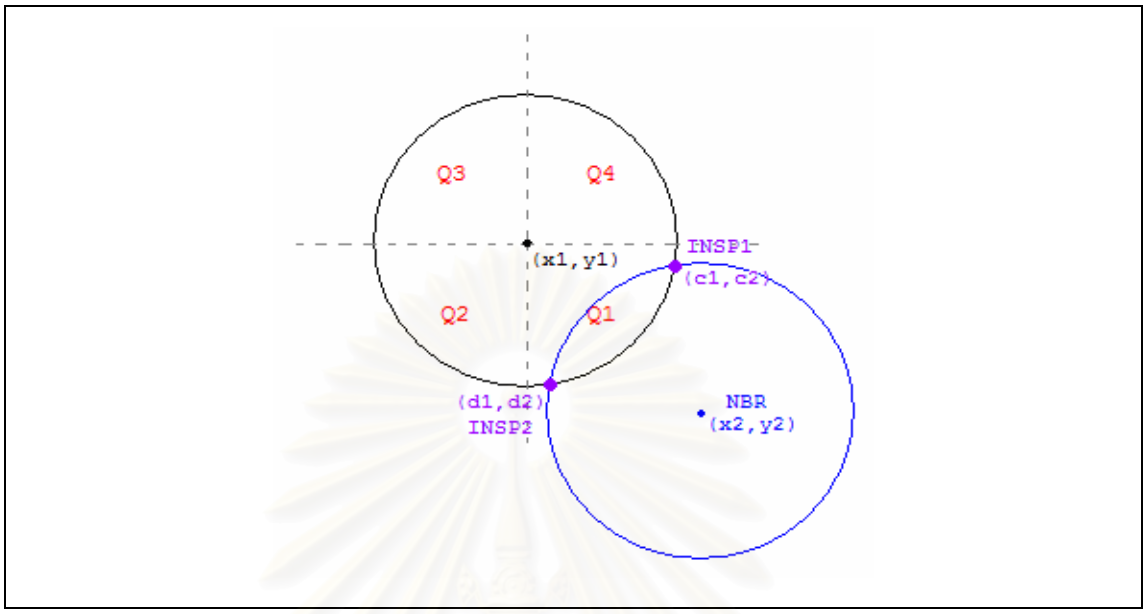
- Q1 : มุม $[0,\frac{\pi}{2}]$ ค่า $\sin[0,1]$
- Q2 : มุม $[\frac{\pi}{2},\pi]$ ค่า $\sin[1,0]$
- Q3 : มุม $[\pi,\frac{3\pi}{2}]$ ค่า $\sin[0,-1]$
- Q4 : มุม $[\frac{3\pi}{2},0]$ ค่า $\sin[-1,0]$



รูปที่ ข.8 การแบ่งขอบเขตของจตุภาคในระนาบ 2 มิติ

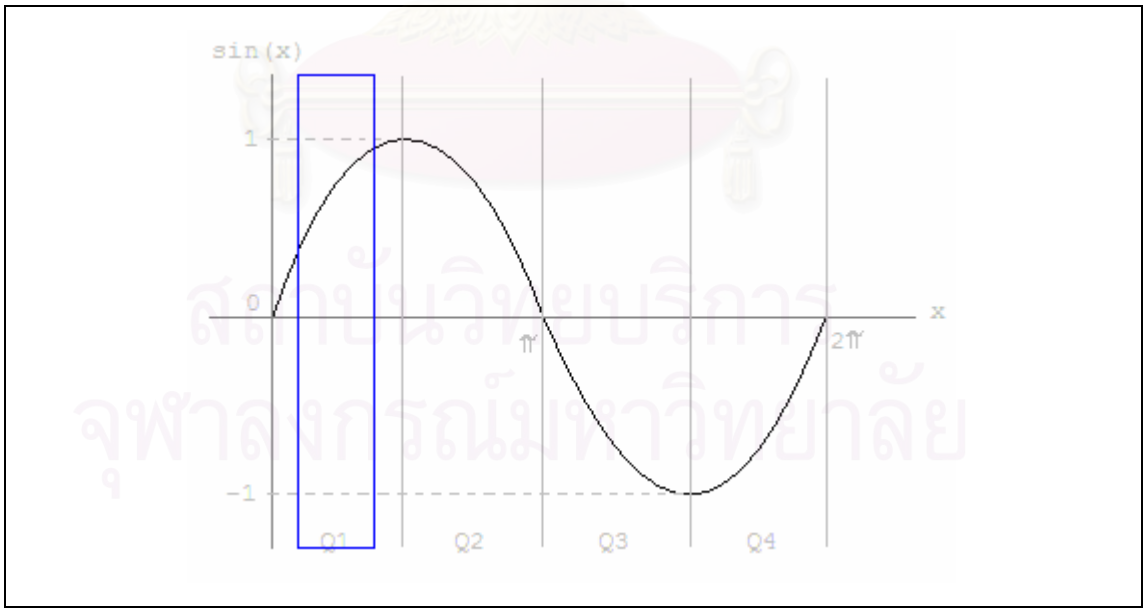
เมื่อลองพิจารณาทุกกรณีที่เป็นไปได้ของตำแหน่งโหมตเพื่อนบ้านที่เข้ามาทับซ้อนพื้นที่กับโหมตอ้างอิงจะได้ว่ามีทั้งสิ้น 16 กรณีดังนี้ โดยกำหนดให้โหมตเพื่อนบ้านแทนด้วยสัญลักษณ์ NBR จุดตัดที่ 1 แทนด้วย INSP1 และจุดตัดที่ 2 แทนด้วย INSP2

I. NBR อยู่ Q1 INSP1 อยู่ Q1 และ INSP2 อยู่ Q1 ดังรูปที่ ข.9



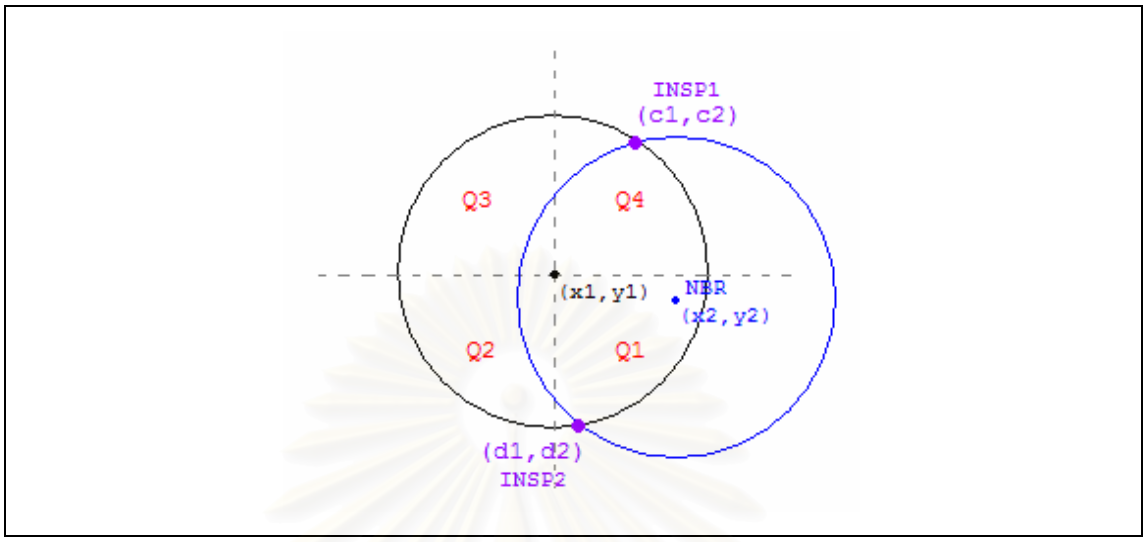
รูปที่ ข.9 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 1 และ 1

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 ของ sin curve ดังรูปที่ ข.10



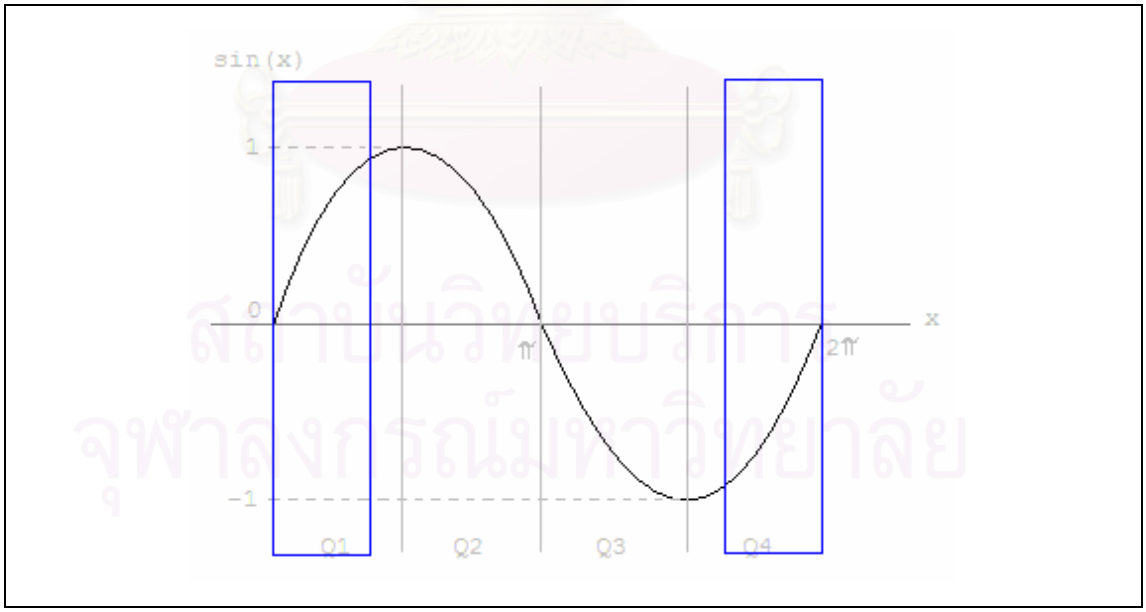
รูปที่ ข.10 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 1 และ 1

II. NBR อยู่ Q1 INSP1 อยู่ Q4 และ INSP2 อยู่ Q1 ดังรูปที่ ข.11



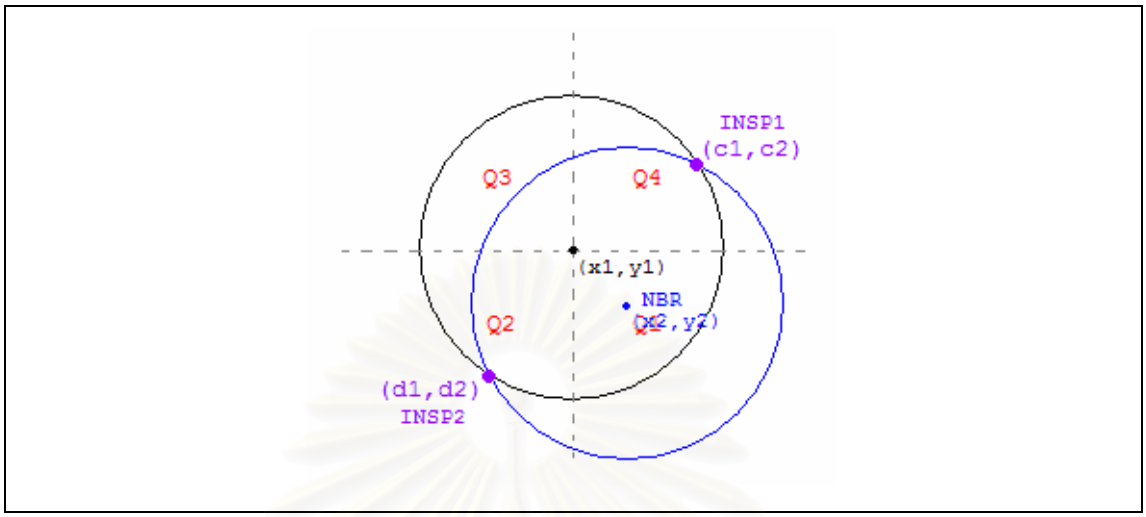
รูปที่ ข.11 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 4 และ 1

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 และ Q4 ของ sin curve ดังรูปที่ ข.12



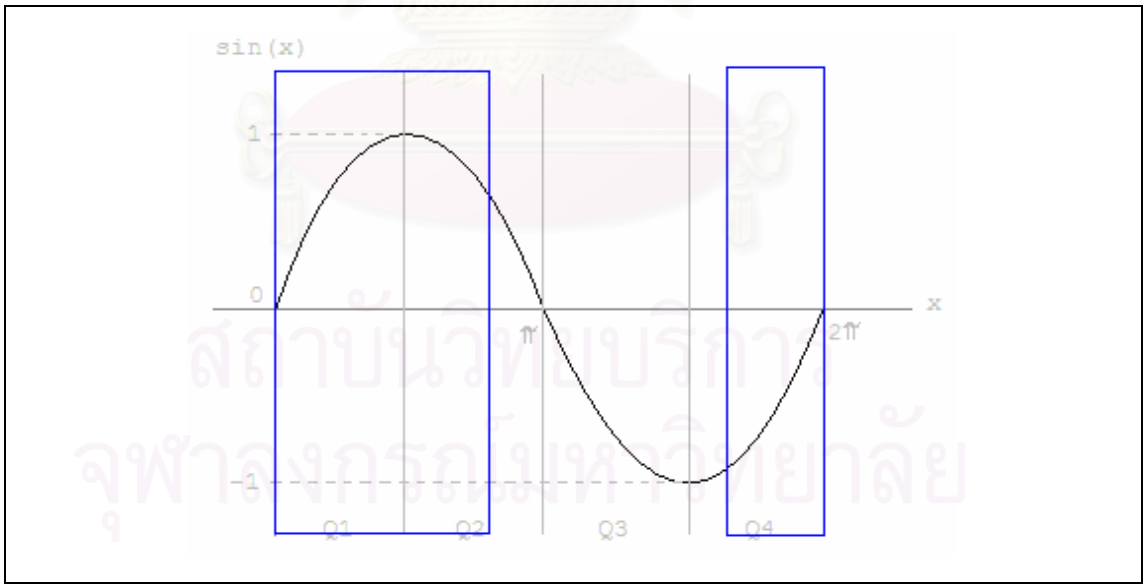
รูปที่ ข.12 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 4 และ 1

III. NBR อยู่ Q1 INSP1 อยู่ Q4 และ INSP2 อยู่ Q2 ดังรูปที่ ข.13



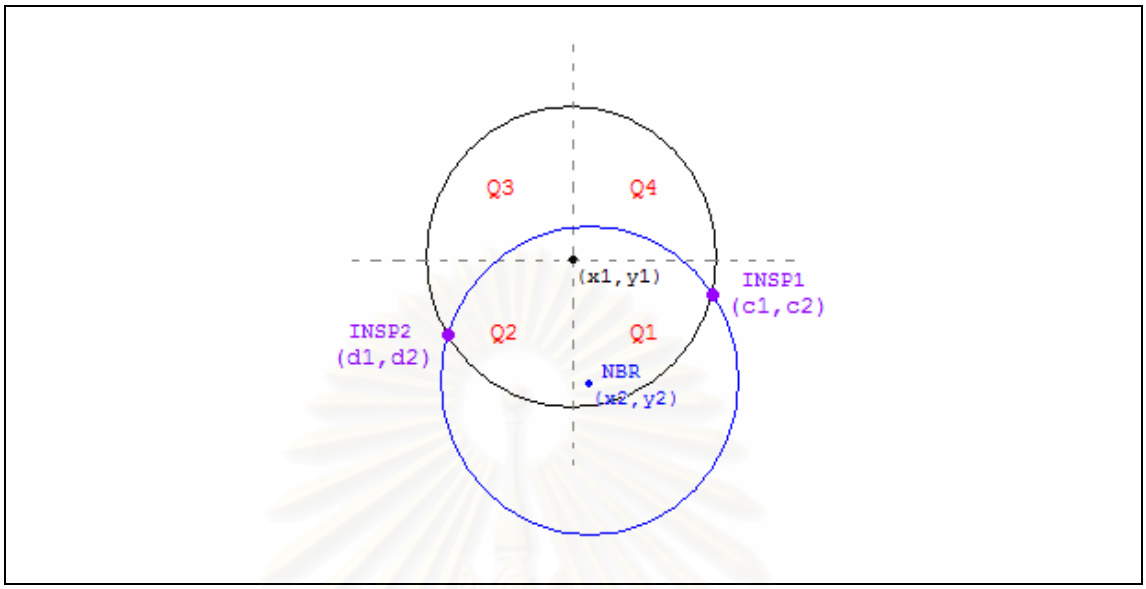
รูปที่ ข.13 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 4 และ 2

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 Q2 และ Q4 ของ sin curve ดังรูปที่ ข.14



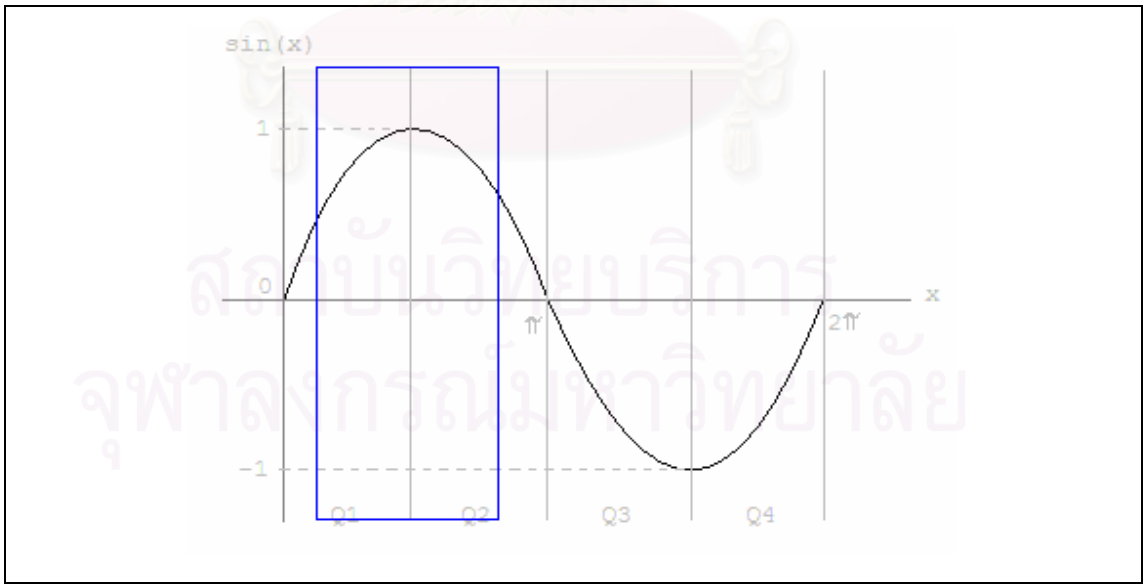
รูปที่ ข.14 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 4 และ 2

IV. NBR อยู่ Q1 INSP1 อยู่ Q1 และ INSP2 อยู่ Q2 ดังรูปที่ ข.15



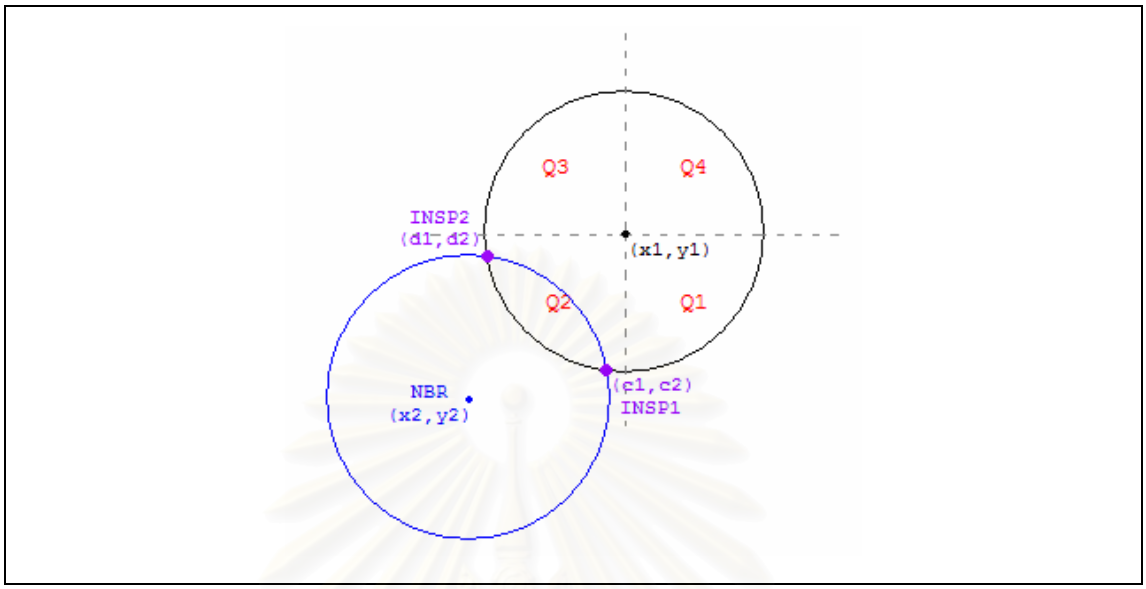
รูปที่ ข.15 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 1 และ 2

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 และ Q2 ของ sin curve ดังรูปที่ ข.16



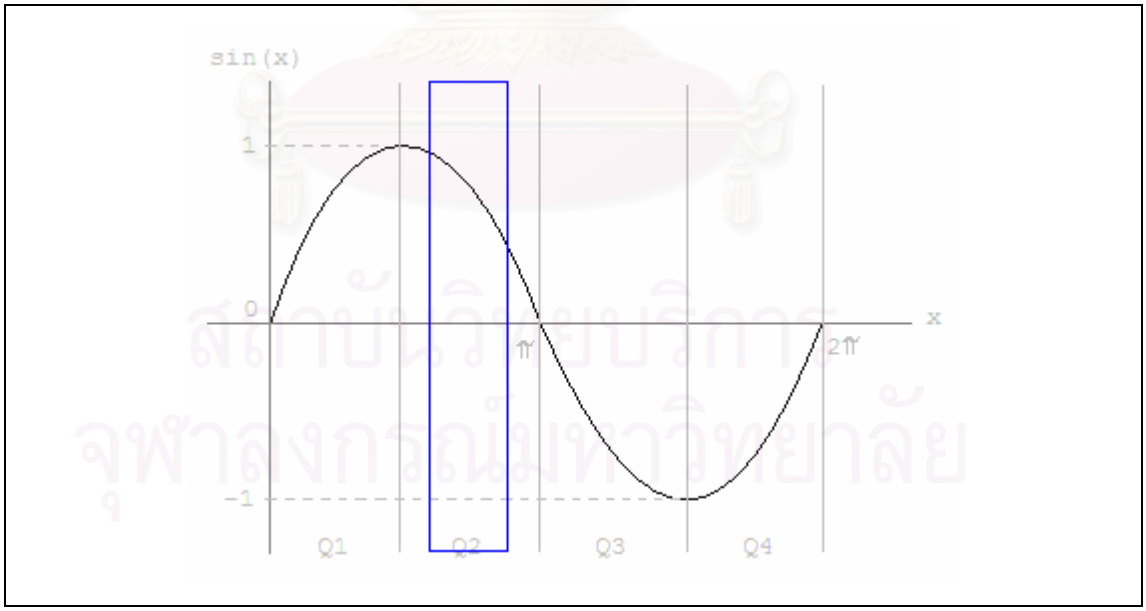
รูปที่ ข.16 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 1 1 และ 2

V. NBR อยู่ Q2 INSP1 อยู่ Q2 และ INSP2 อยู่ Q2 ดังรูปที่ ข.17



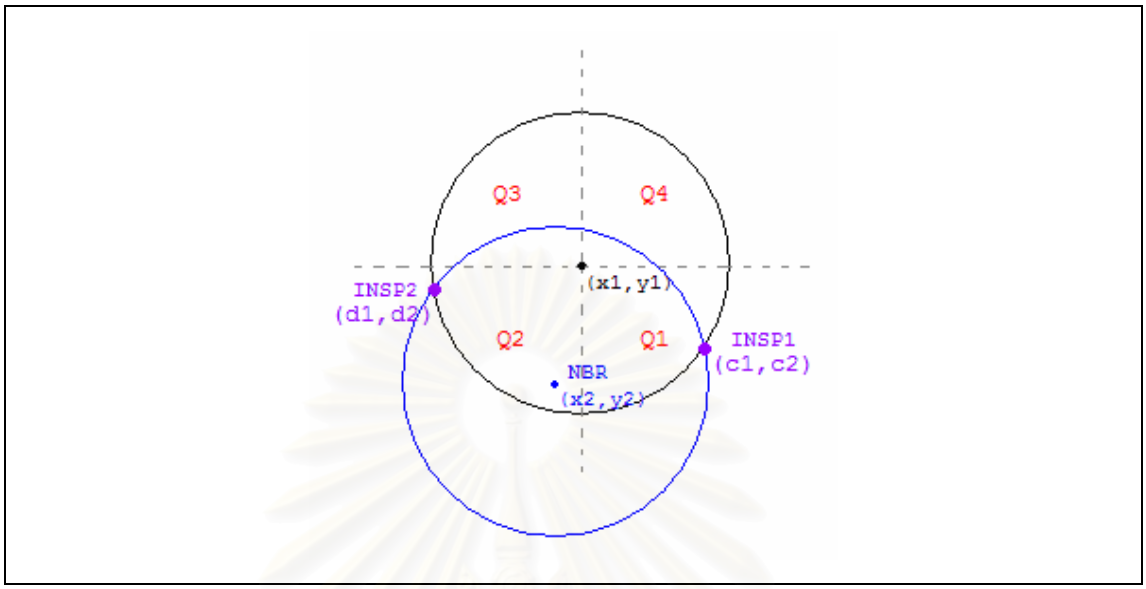
รูปที่ ข.17 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 2 และ 2

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q2 ของ sin curve ดังรูปที่ ข.18



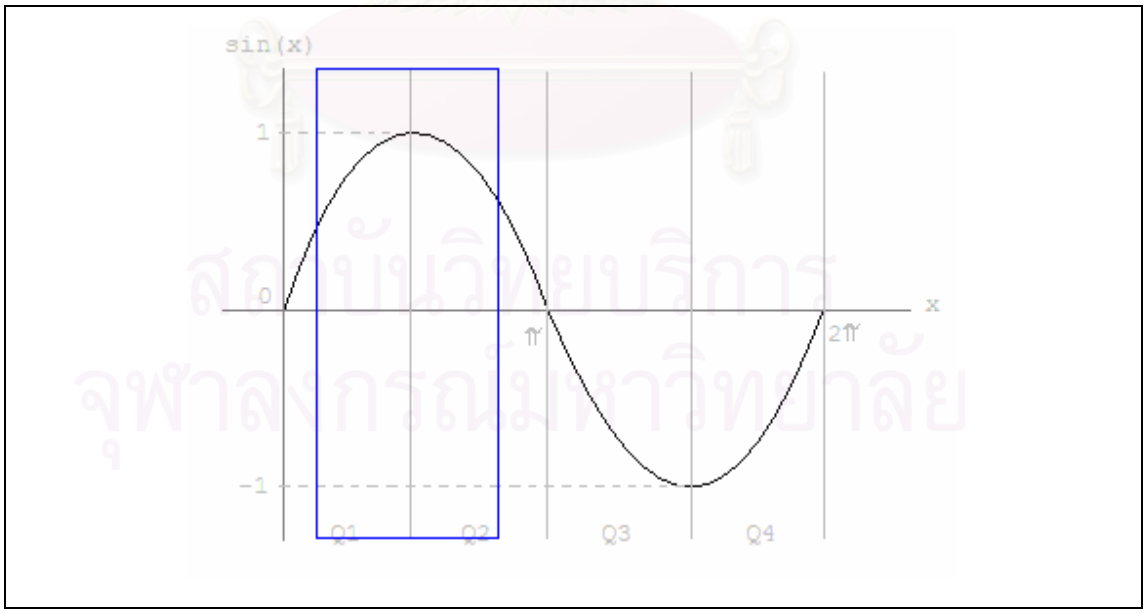
รูปที่ ข.18 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 2 และ 2

VI. NBR อยู่ Q2 INSP1 อยู่ Q1 และ INSP2 อยู่ Q2 ดังรูปที่ ข.19



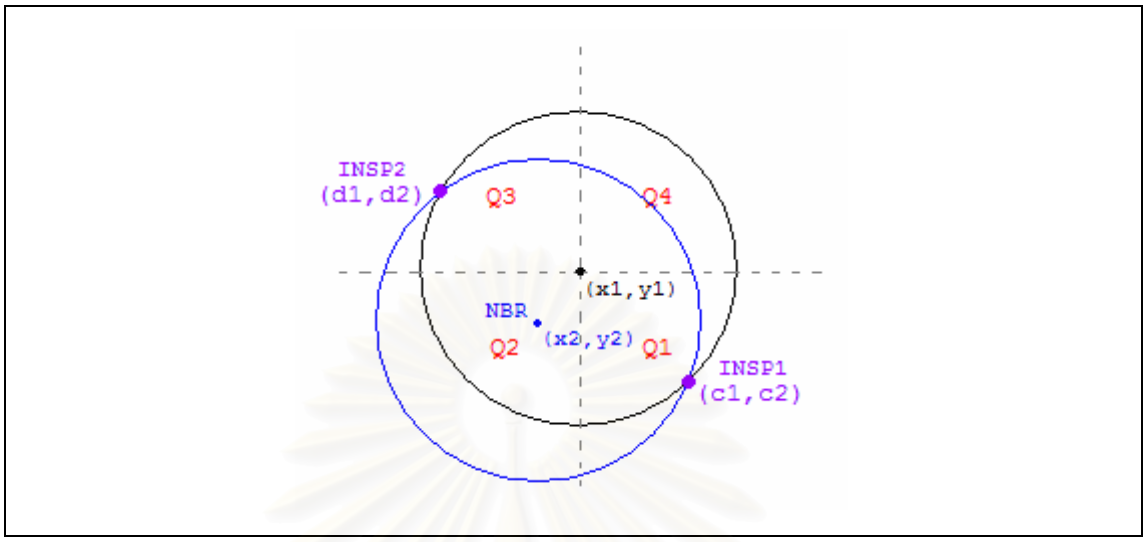
รูปที่ ข.19 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 1 และ 2

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 และ Q2 ของ sin curve ดังรูปที่ ข.20



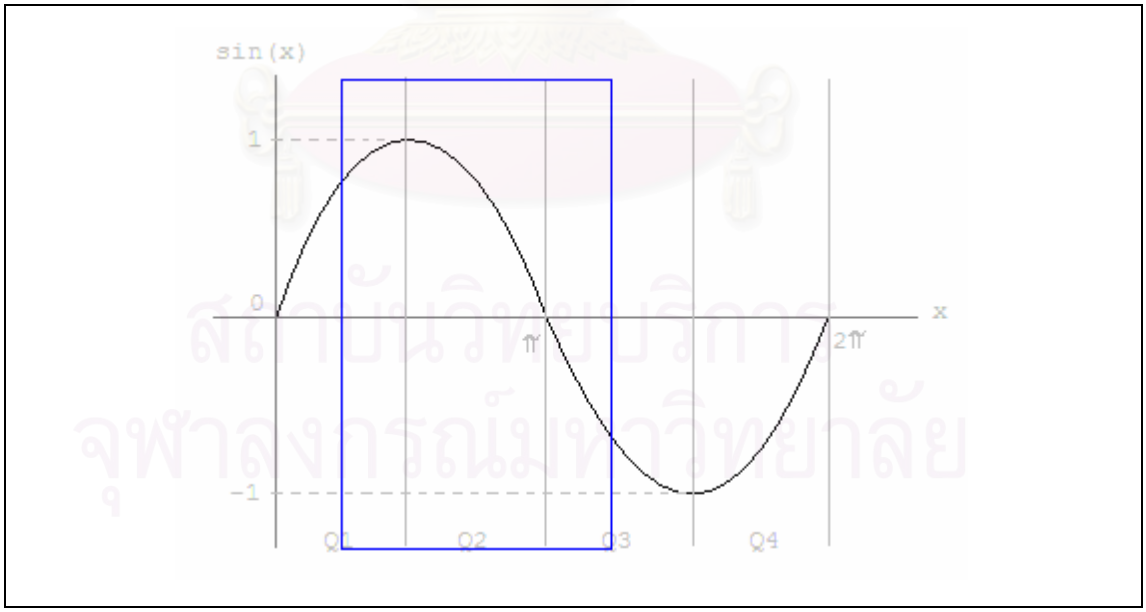
รูปที่ ข.20 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 1 และ 2

VII. NBR อยู่ Q2 INSP1 อยู่ Q1 และ INSP2 อยู่ Q3 ดังรูปที่ ข.21



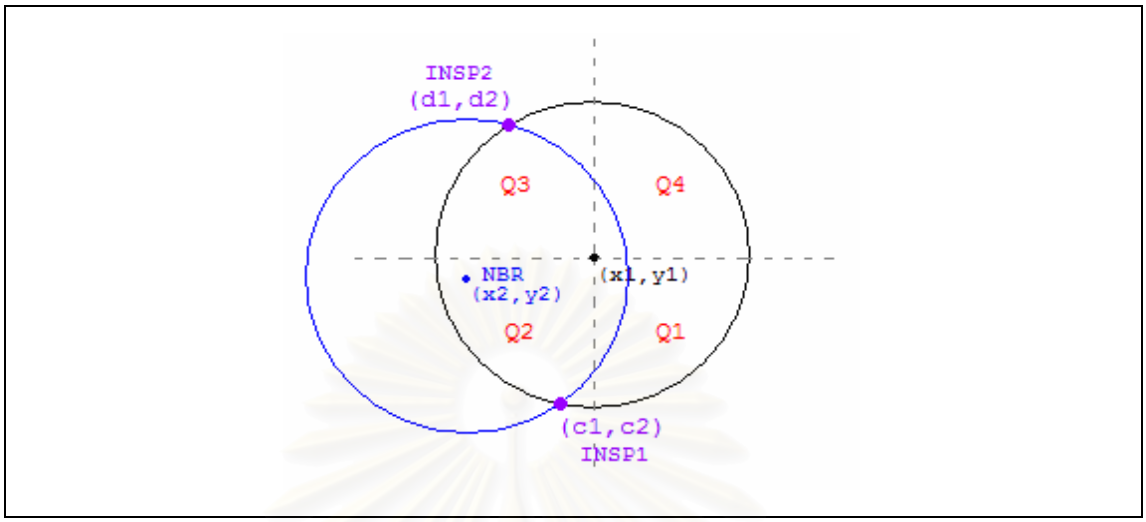
รูปที่ ข.21 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 1 และ 3

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 Q2 และ Q3 ของ sin curve ดังรูปที่ ข.22



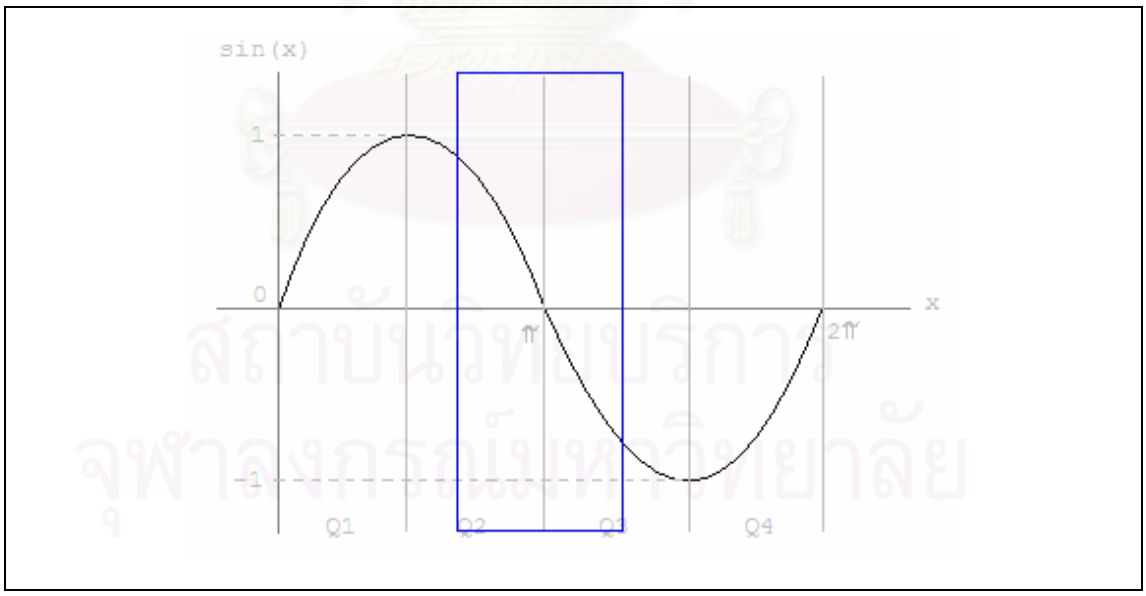
รูปที่ ข.22 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 1 และ 3

VIII. NBR อยู่ Q2 INSP1 อยู่ Q2 และ INSP2 อยู่ Q3 ดังรูปที่ ข.23



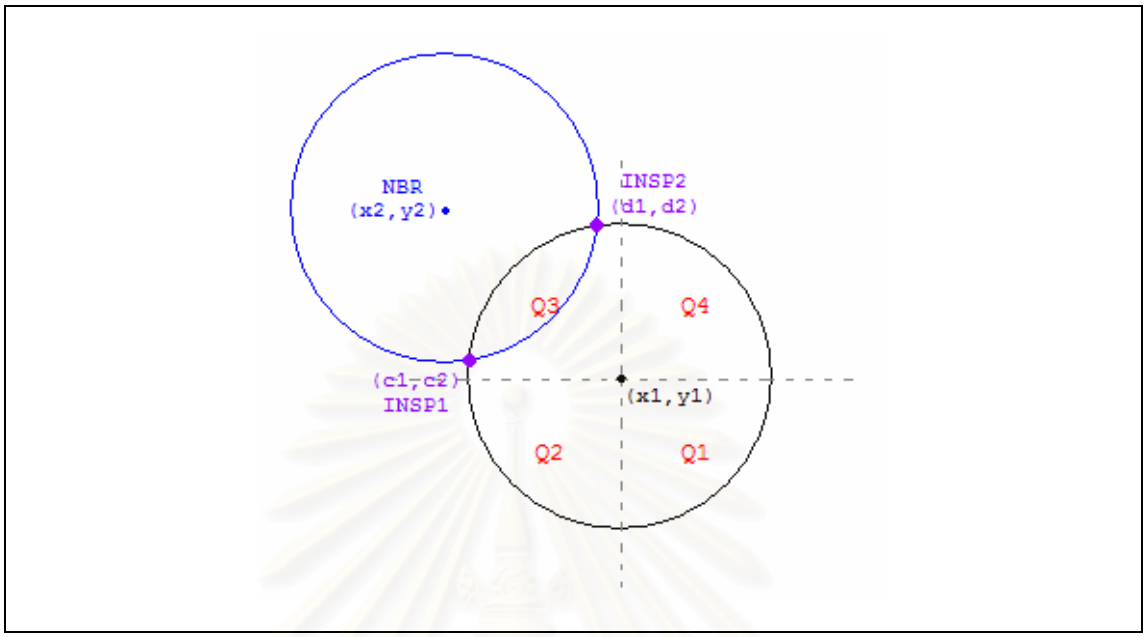
รูปที่ ข.23 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 2 และ 3

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q2 และ Q3 ของ sin curve ดังรูปที่ ข.24



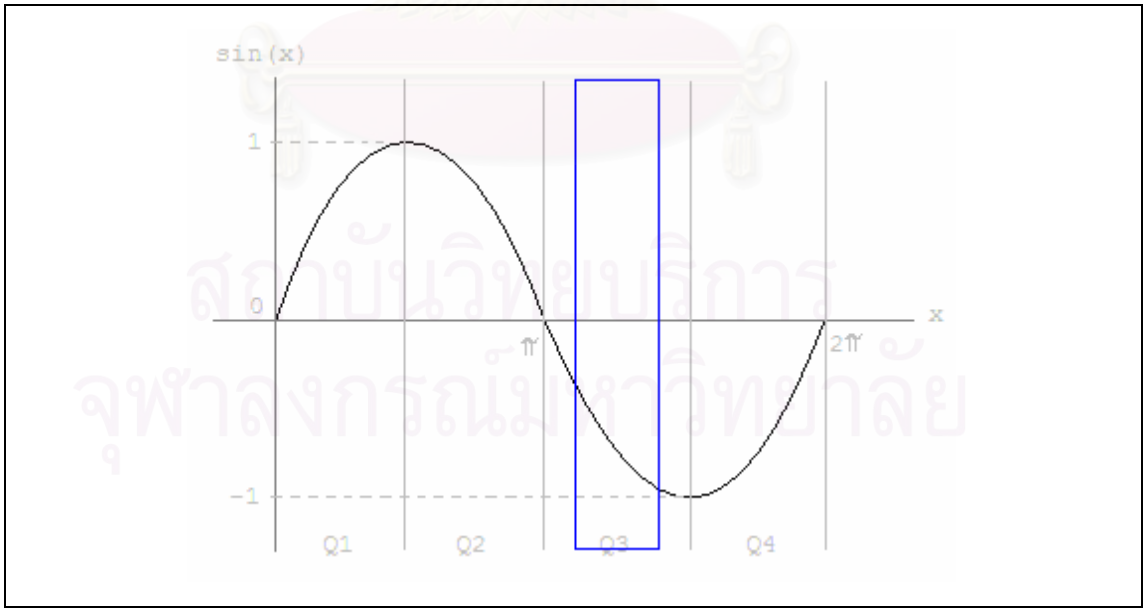
รูปที่ ข.24 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 2 2 และ 3

IX. NBR อยู่ Q3 INSP1 อยู่ Q3 และ INSP2 อยู่ Q3 ดังรูปที่ ข.25



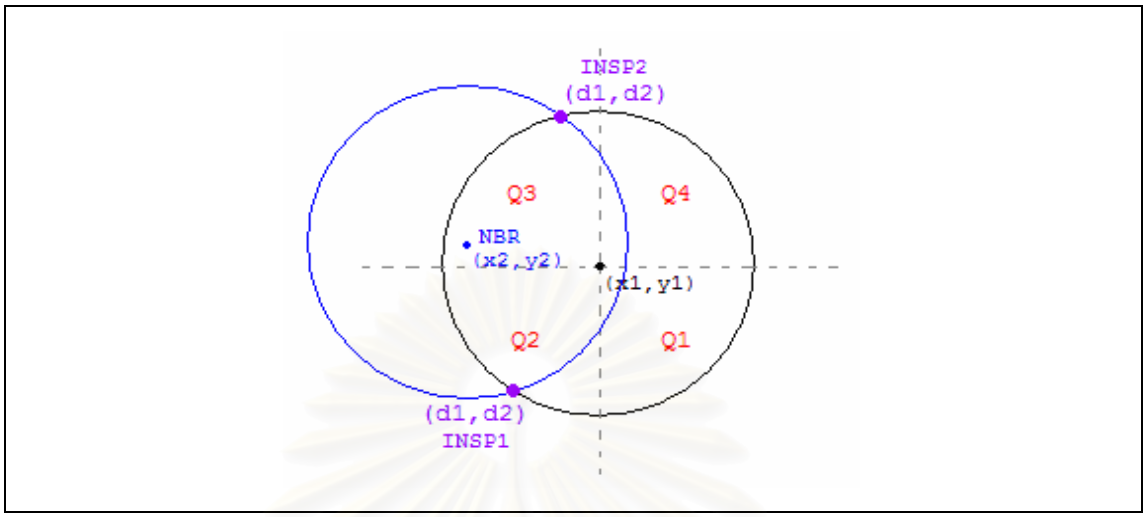
รูปที่ ข.25 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 3 และ 3

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q3 ของ sin curve ดังรูปที่ ข.26



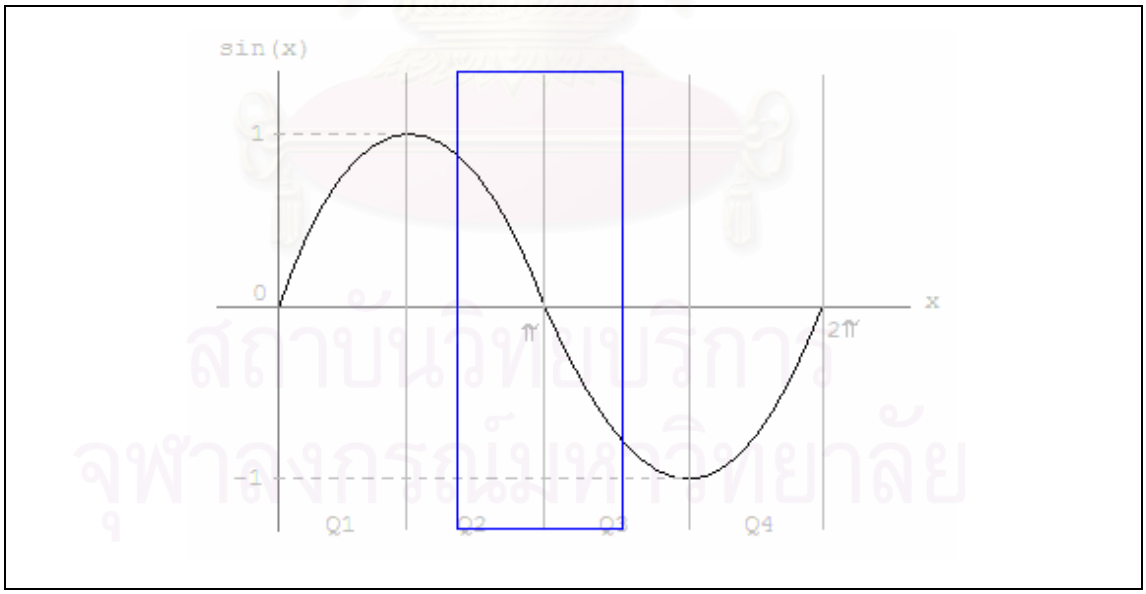
รูปที่ ข.26 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 3 และ 3

X. NBR อยู่ Q3 INSP1 อยู่ Q2 และ INSP2 อยู่ Q3 ดังรูปที่ ข.27



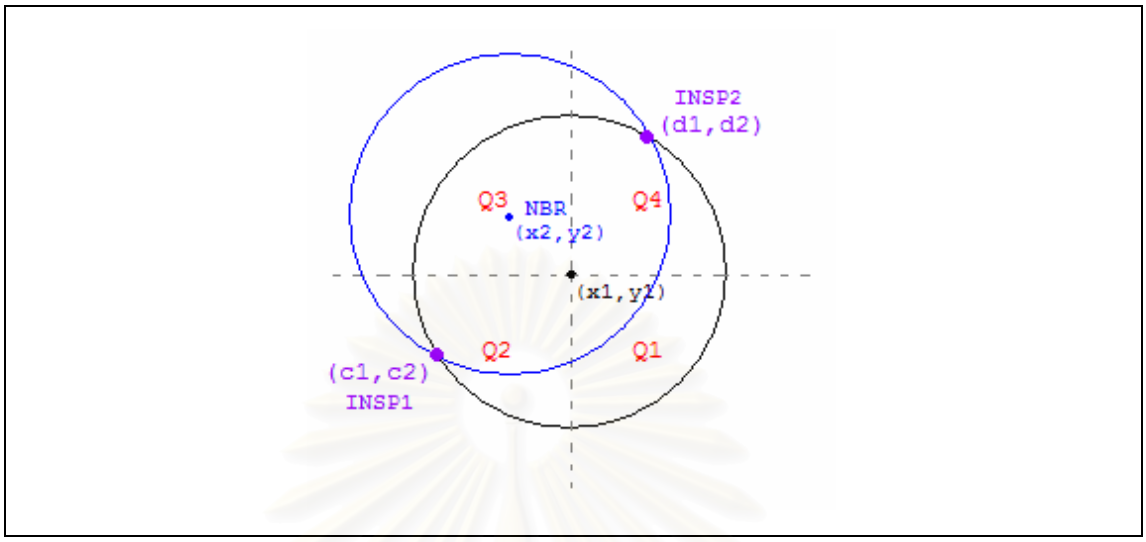
รูปที่ ข.27 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 2 และ 3

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q2 และ Q3 ของ sin curve ดังรูปที่ ข.28



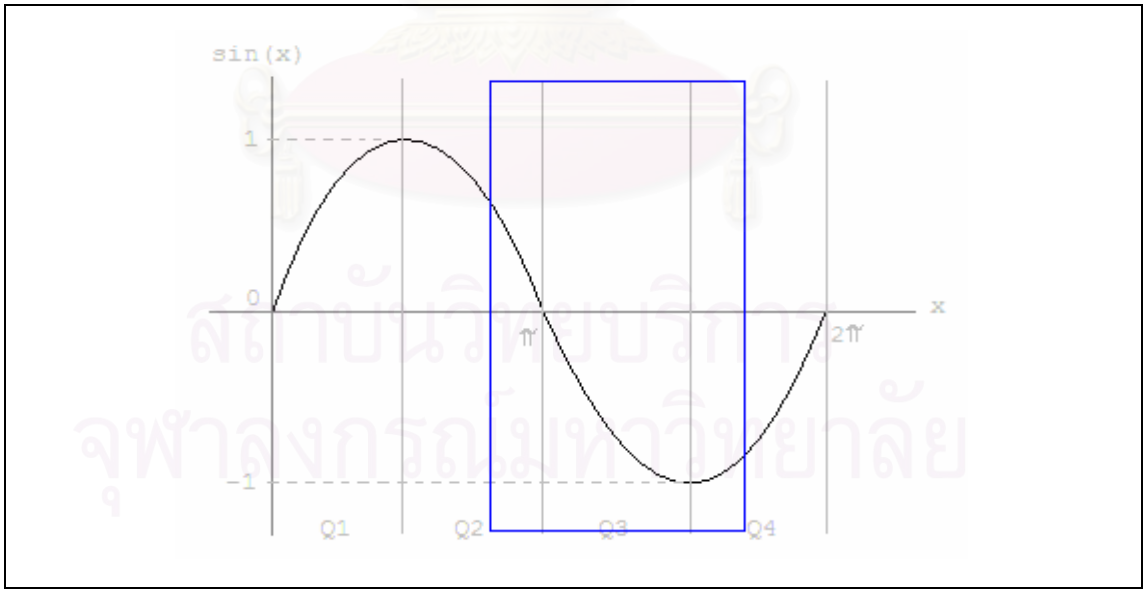
รูปที่ ข.28 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 2 และ 3

XI. NBR อยู่ Q3 INSP1 อยู่ Q2 และ INSP2 อยู่ Q4 ดังรูปที่ ข.29



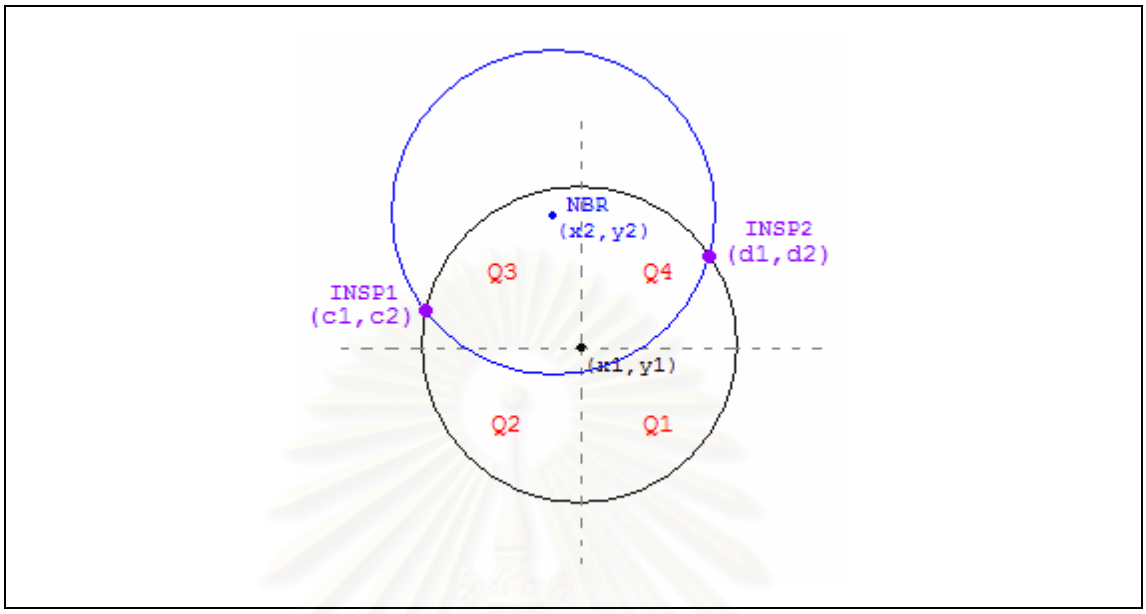
รูปที่ ข.29 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 2 และ 4

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q2 Q3 และ Q4 ของ sin curve ดังรูปที่ ข.30



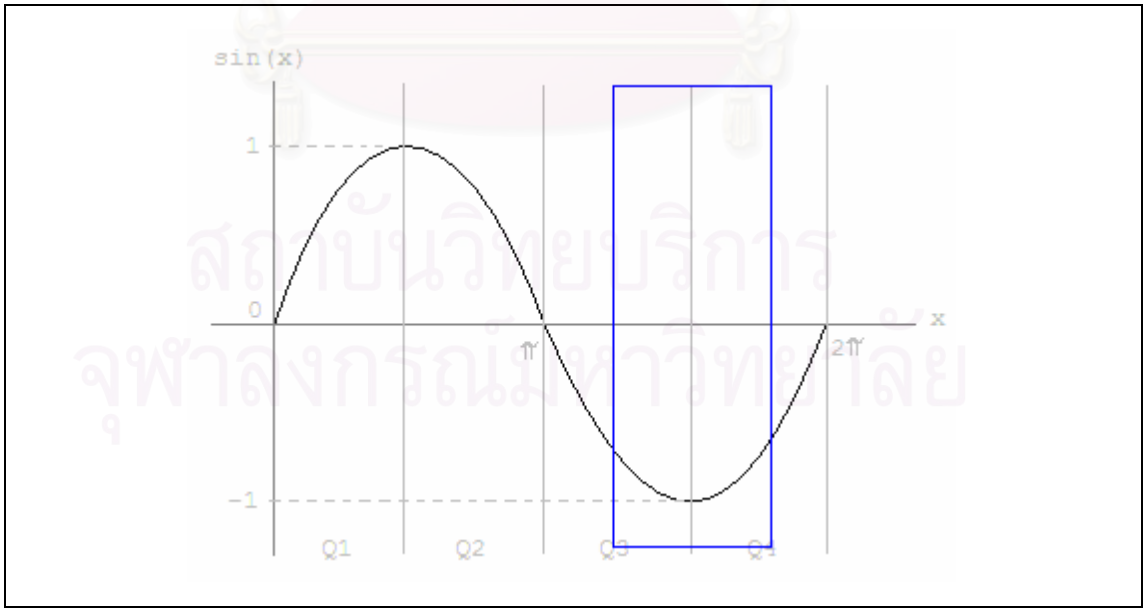
รูปที่ ข.30 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 2 และ 4

XII. NBR อยู่ Q3 INSP1 อยู่ Q3 และ INSP2 อยู่ Q4 ดังรูปที่ ข.31



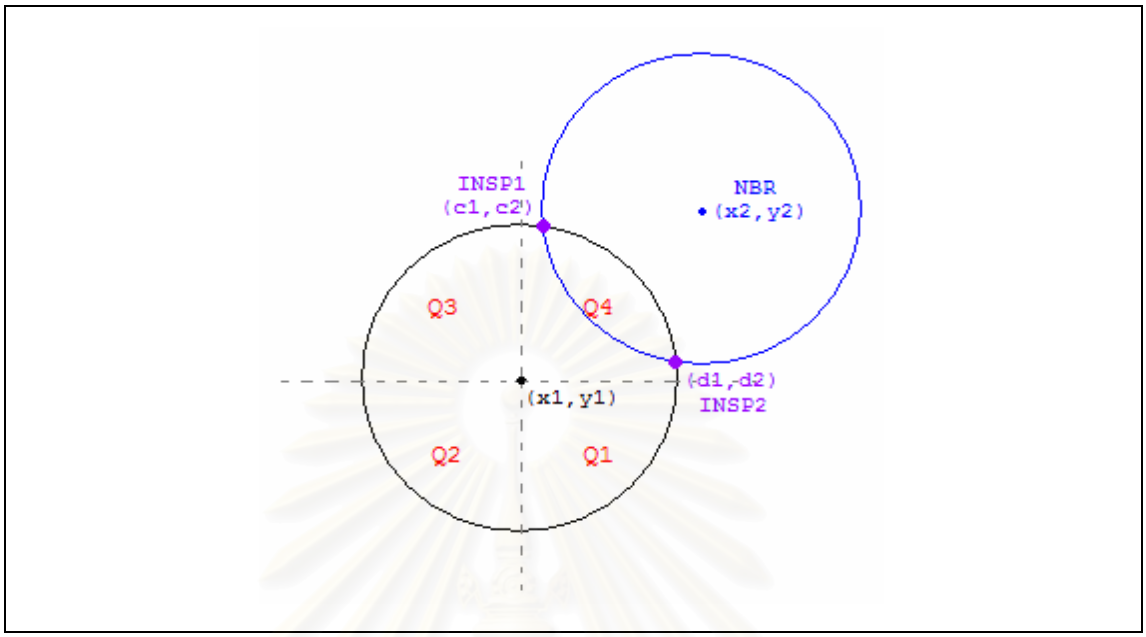
รูปที่ ข.31 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 3 และ 4

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q3 และ Q4 ของ sin curve ดังรูปที่ ข.32



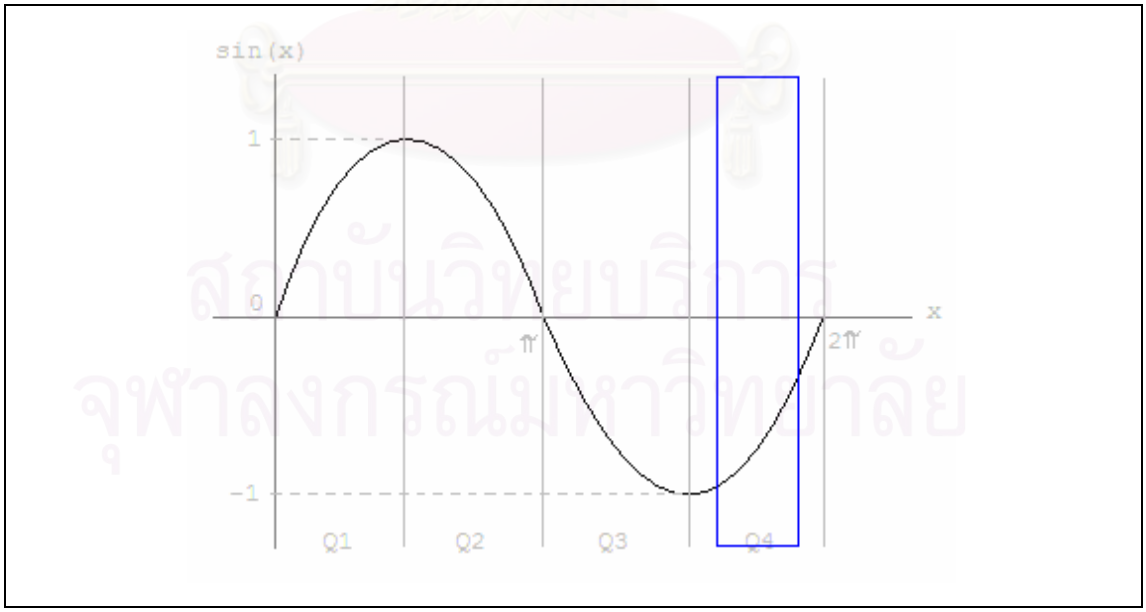
รูปที่ ข.32 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 3 3 และ 4

XIII. NBR อยู่ Q4 INSP1 อยู่ Q4 และ INSP2 อยู่ Q4 ดังรูปที่ ข.33



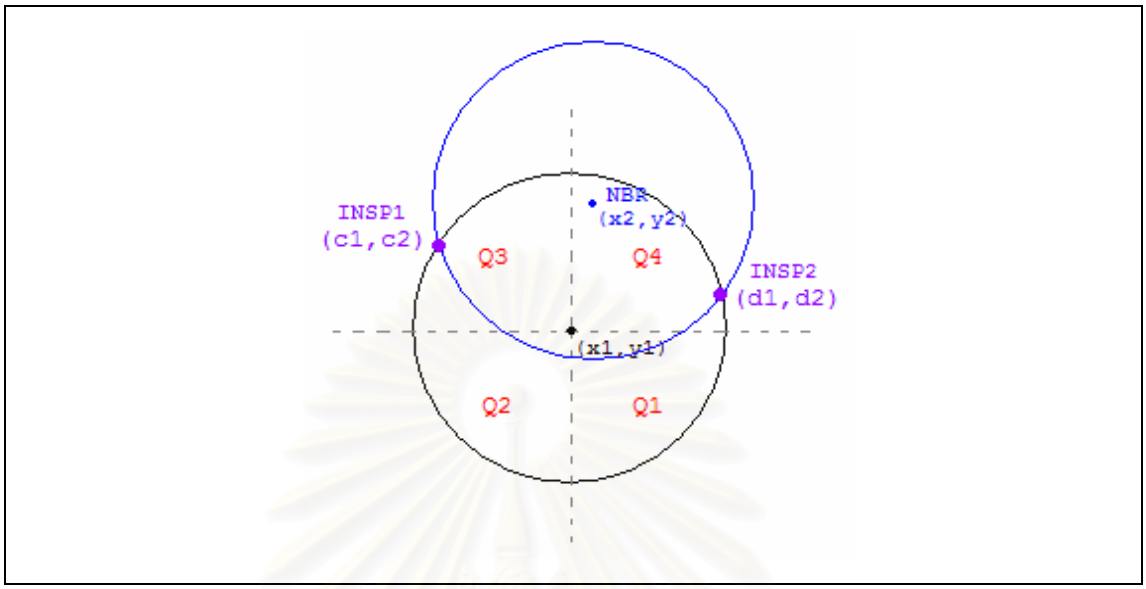
รูปที่ ข.33 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 4 และ 4

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q4 ของ sin curve ดังรูปที่ ข.34



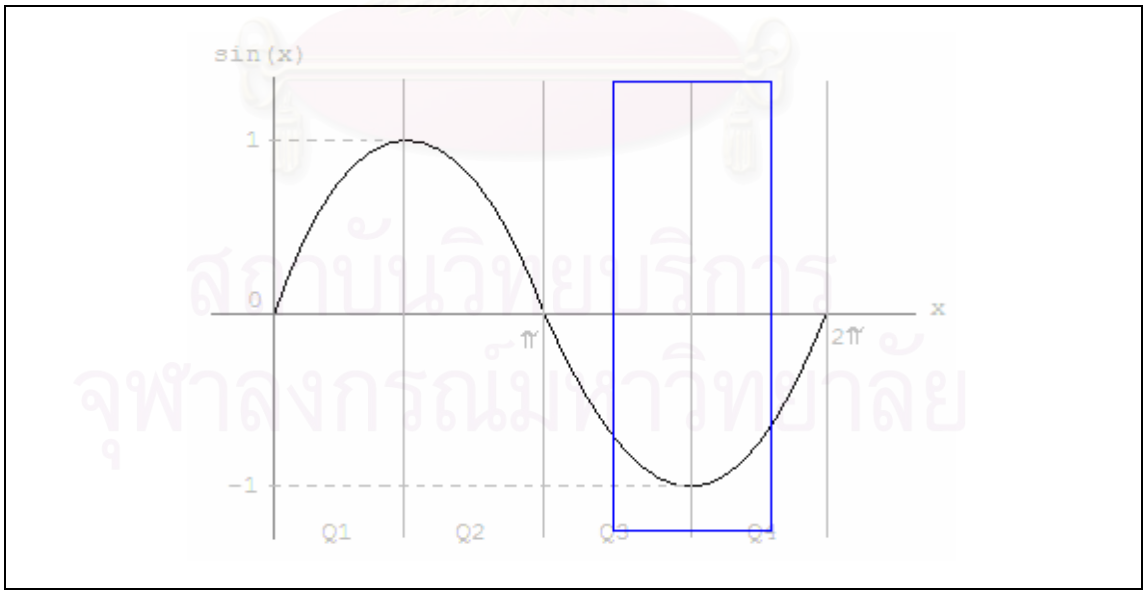
รูปที่ ข.34 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 4 และ 4

XIV. NBR อยู่ Q4 INSP1 อยู่ Q3 และ INSP2 อยู่ Q4 ดังรูปที่ ข.35



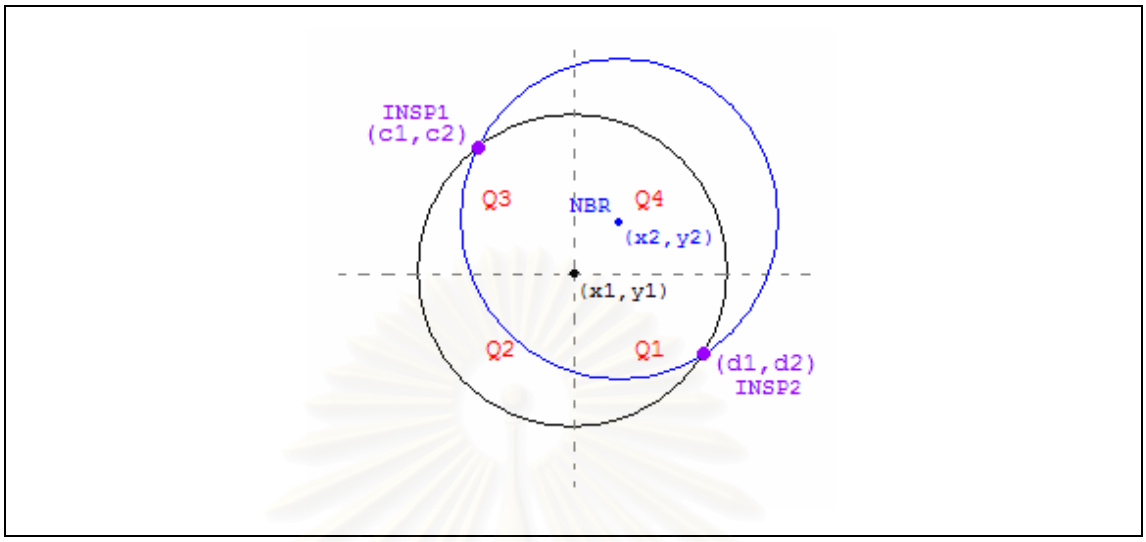
รูปที่ ข.35 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 3 และ 4

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q3 และ Q4 ของ sin curve ดังรูปที่ ข.36



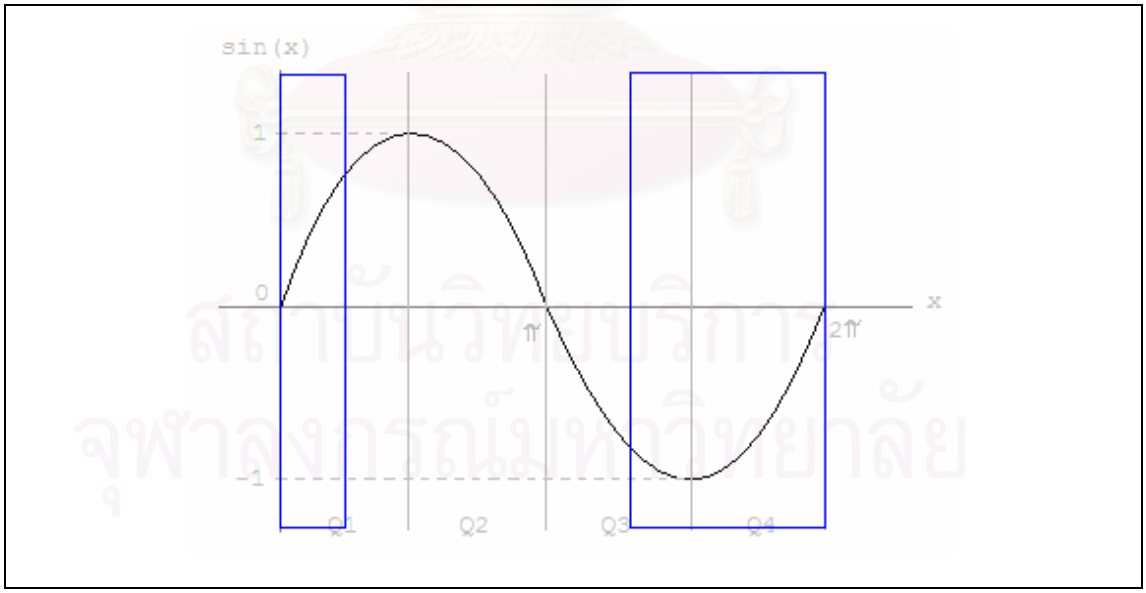
รูปที่ ข.36 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 3 และ 4

XV. NBR อยู่ Q4 INSP1 อยู่ Q3 และ INSP2 อยู่ Q1 ดังรูปที่ ข.37



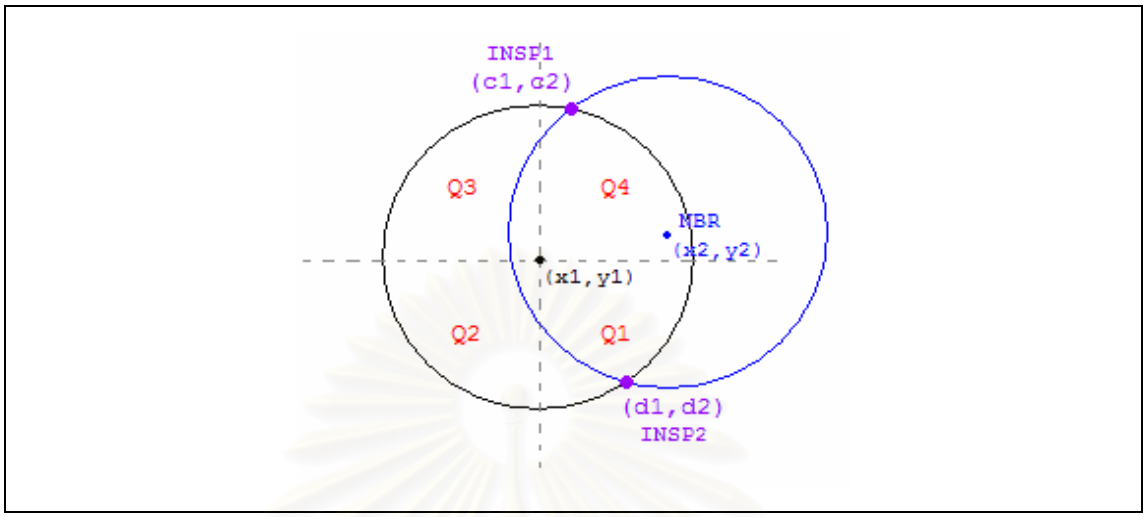
รูปที่ ข.37 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 3 และ 1

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 Q3 และ Q4 ของ sin curve ดังรูปที่ ข.38



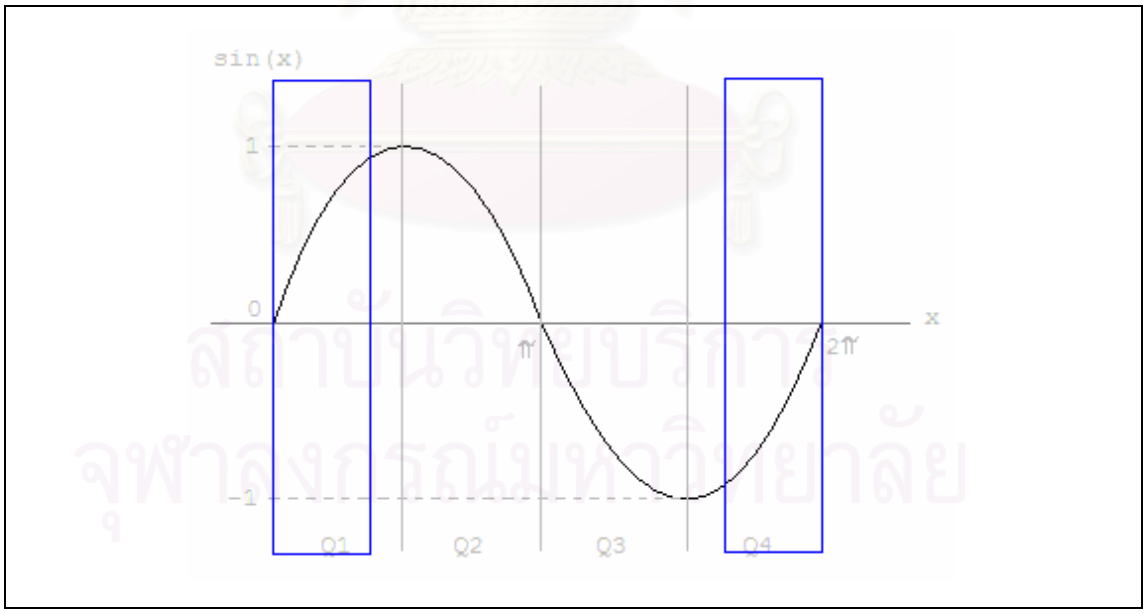
รูปที่ ข.38 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 3 และ 1

XVI. NBR อยู่ Q4 INSP1 อยู่ Q4 และ INSP2 อยู่ Q1 ดังรูปที่ ข.39



รูปที่ ข.39 ภาพแสดงกรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 4 และ 1

กรณีนี้ช่วงของค่า sin แทนมุมจะอยู่ใน Q1 และ Q4 ของ sin curve ดังรูปที่ ข.40



รูปที่ ข.40 กราฟแสดงช่วงค่า sin แทนมุม
กรณีตำแหน่ง NBR INSP1 และ INSP2 อยู่ในจตุภาคที่ 4 4 และ 1

6) จากค่า sin ที่หาได้จากจุดตัดทั้ง 2 จุด รวมเข้ากับจุตภาคที่อยู่ของโหมตเพื่อนบ้านและจุดตัดทั้งสอง จะให้เราสามารถรู้ช่วงของค่า sin ที่ส่วนของเส้นรอบวงที่ถูกครอบคลุมโดยโหมตเพื่อนบ้านได้ดังนี้

6.1) กำหนดให้ ค่า sin ที่ได้จากจุดตัดที่ 1 (INSP1) มีค่าเป็น SIN_{P1}

ค่า sin ที่ได้จากจุดตัดที่ 2 (INSP2) มีค่าเป็น SIN_{P2}

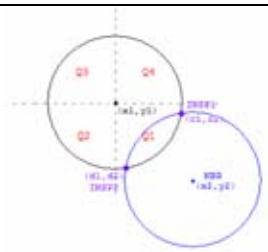
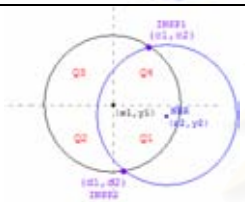
โหมตเพื่อนบ้านอยู่ในจตุภาค Q_N

จุดตัดที่ 1 อยู่ในจตุภาค Q_{P1}

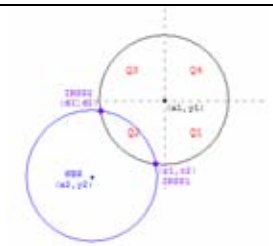
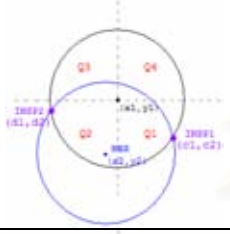
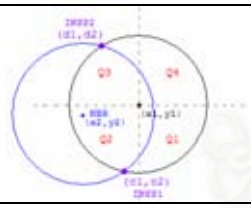
จุดตัดที่ 2 อยู่ในจตุภาค Q_{P2}

6.2) เทียบค่า Q_N Q_{P1} และ Q_{P2} กับตารางที่ ข.1 ข.2 ข.3 และ ข.4 เพื่อบอกช่วงของค่า sin ทั้ง 4 จตุภาค

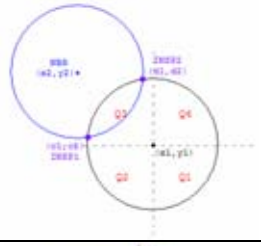
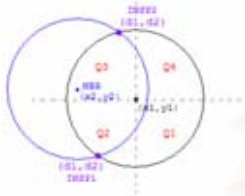
ตารางที่ ข.1 ความสัมพันธ์ระหว่างจตุภาคของ NBR INSP1 และ INSP2 กับช่วงค่าของ sin เมื่อ NBR อยู่ในจตุภาคที่ 1

รูปประกอบ	Q_n	Q_{P1}	Q_{P2}	ค่า sin ใน Q1	ค่า sin ใน Q2	ค่า sin ใน Q3	ค่า sin ใน Q4
	1	1	1	$[SIN_{P1}, SIN_{P2}]$	-	-	-
	1	4	1	$[0, SIN_{P2}]$	-	-	$[SIN_{P1}, 0]$
	1	4	2	[0, 1]	$[1, SIN_{P2}]$	-	$[SIN_{P1}, 0]$
	1	1	2	$[SIN_{P1}, 1]$	$[1, SIN_{P2}]$	-	-

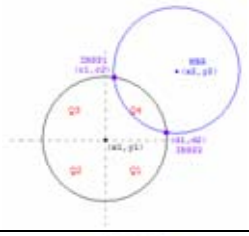
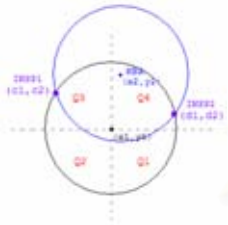
ตารางที่ ข.2 ความสัมพันธ์ระหว่างจตุภาคของ NBR INSP1 และ INSP2 กับช่วงค่าของ sin เมื่อ NBR อยู่ในจตุภาคที่ 2

รูปประกอบ	Q_n	Q_{P1}	Q_{P2}	ค่า sin ใน Q1	ค่า sin ใน Q2	ค่า sin ใน Q3	ค่า sin ใน Q4
	2	2	2	-	$[SIN_{P1}, SIN_{P2}]$	-	-
	2	1	2	$[SIN_{P1}, 1]$	$[1, SIN_{P2}]$	-	-
	2	1	3	$[SIN_{P1}, 1]$	$[1, 0]$	$[0, SIN_{P2}]$	-
	2	2	3	-	$[SIN_{P1}, 0]$	$[0, SIN_{P2}]$	-

ตารางที่ ข.3 ความสัมพันธ์ระหว่างจตุภาคของ NBR INSP1 และ INSP2 กับช่วงค่าของ sin เมื่อ NBR อยู่ในจตุภาคที่ 3

รูปประกอบ	Q_n	Q_{P1}	Q_{P2}	ค่า sin ใน Q1	ค่า sin ใน Q2	ค่า sin ใน Q3	ค่า sin ใน Q4
	3	3	3	-	-	$[SIN_{P1}, SIN_{P2}]$	-
	3	2	3	-	$[SIN_{P1}, 0]$	$[0, SIN_{P2}]$	-
	3	2	4	-	$[SIN_{P1}, 0]$	[0, -1]	$[-1, SIN_{P2}]$
	3	3	4	-	-	$[SIN_{P1}, -1]$	$[-1, SIN_{P2}]$

ตารางที่ ข.4 ความสัมพันธ์ระหว่างจตุภาคของ NBR INSP1 และ INSP2 กับช่วงค่าของ sin เมื่อ NBR อยู่ในจตุภาคที่ 4

รูปประกอบ	Q_n	Q_{P1}	Q_{P2}	ค่า sin ใน Q1	ค่า sin ใน Q2	ค่า sin ใน Q3	ค่า sin ใน Q4
	4	4	4	-	-	-	$[SIN_{P1}, SIN_{P2}]$
	4	4	3	-	-	$[SIN_{P1}, -1]$	$[-1, SIN_{P2}]$
	4	1	3	$[0, SIN_{P2}]$	-	$[SIN_{P1}, -1]$	[-1, 0]
	4	1	4	$[0, SIN_{P2}]$	-	-	$[SIN_{P1}, 0]$

ภาคผนวก ค

บทความทางวิชาการ

ส่วนหนึ่งของงานวิทยานิพนธ์ได้รับการตีพิมพ์เป็นบทความวิชาการในหัวเรื่อง “A Set Cover-Based Density Control Algorithm for Sensing Coverage Problems in Wireless Sensor Networks” โดย ศรัณย์ เจนจตุรงค์ และ เฉลิมเอก อินทนากรวิวัฒน์ ในงานประชุมวิชาการ “The Third International Conference on Cognitive Radio Oriented Wireless Networks and Communications” ซึ่งจัดขึ้น ณ โรงแรม Holiday Inn Atrium Singapore เมืองสิงคโปร์ ประเทศสิงคโปร์ ระหว่างวันที่ 15-17 พฤษภาคม 2551

A Set Cover-Based Density Control Algorithm for Sensing Coverage Problems in Wireless Sensor Networks

Saran Jenjaturong
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
g49sjn@cp.eng.chula.ac.th

Chalermek Intanagonwiwat
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
intanago@cp.eng.chula.ac.th

***Abstract*—Wireless sensor networks consist of a large number of sensor nodes with limited power and resource. To prolong network lifetime, the energy consumption must be somehow reduced. In this paper, we propose a localized density control algorithm for energy savings. The goals are to maintain a minimal number of active sensor nodes and to reduce radio-traffic intensity while conserving the sensing coverage of the network. Our localized algorithm is based on a greedy solution of a weighted set-cover problem. Each node locally computes whether to sleep or to stay active. Given that the local decision might worsen the sensing coverage, we also introduce a voting scheme for selecting active nodes to assure that a node can sleep if and only if its sensing area is completely covered by its active neighbors. We have implemented our localized algorithm and voting scheme on Tiny OS and evaluated on TOSSIM. The result indicates that our algorithm is efficient and viable for practical use.**

Keywords; wireless sensor networks, coverage problem, density control algorithm, set cover problem, redundancy, node scheduling

I. INTRODUCTION

Wireless sensor networks are becoming available for effective monitoring and collecting data in an interested area. Sensor networks are composed of a large number of small, low-power devices called a sensor node. Each sensor node has ability to communicate and transfer data through the wireless on-board radio transceiver. For monitoring an environment, there is a sensing module that can gather some environment information in its sensing range such as light intensity, humidity, and temperature. Each node collects sensing information and forwards data packets to the others in an ad-hoc manner. Due to a small dimension requirement, an energy resource of each sensor is definitely limited. Furthermore, in some monitoring area, it is difficult to replace a new energy resource for all nodes. Consequently, one of the important sensor-network challenges is to extend the network lifetime (time that the sensing coverage property is still maintained) by decreasing overall energy consumption. Most sensor-network applications (e.g., attack detection and environment monitoring [1]) confront the problem of constrained energy because of their high energy consumption in wireless communication.

Under the assumption that each sensor node has a disk sensing area with the same sensing range, a sensing device can monitor data from the environment in two dimension surface. It is possible that sensing areas of any two nodes located in adjacent areas can overlap with one another. In some sensor network applications, high density of sensors is required for more data integrity in monitoring. As a result, there will be several overlapping areas appearing in the whole monitoring area. In the coverage problem [2], [3], to have well monitored area, these overlapping areas are considered to be undesirable because they cause unnecessary energy consumption. For this reason, a managing approach called a density control algorithm is proposed to activate only a subset of sensor nodes. Sensor nodes whose sensing areas are completely covered by other are decided to be redundant. Some of these redundant nodes are considered to be unreasonably active while the others, non redundant nodes, still work. If some of these redundant nodes are turned off, overall energy consumption and radio traffic intensity of the network will be decreased.

Essentially, a density control algorithm requires maintenance of two important networking properties: sensing coverage and network connectivity. After some redundant sensors are deactivated, the sensing coverage area should not be smaller than the original one when all nodes are active. To maintain the network connectivity, it must be possible to successfully forward any data packet between any couple of nodes during the network lifetime.

Most prior density control algorithms address the solution for finding redundant nodes based on overlap of geographical node's sensing areas. They operate under a centralized concept that each node requires to know all nodes' locations in network. The elimination rules of redundancy are provided for checking node's redundancy status. To avoid appearance of blind area (area that is not detected by any sensor), the node scheduling algorithms are proposed for the coverage maintenance. Some redundant nodes are selected to be active in round by using a node selection rule based purely on how large area the node covers. According to this rule, a stimulation of active nodes depends only on a timer set by node randomly. This random timer solution has not considered about node's remaining energy so any node with low energy can be selected to be active instead of the higher. As a result, the network

lifetime extension maybe results poorly. Our work proposes a density control algorithm including a remaining energy factor in a node selection rule.

In this paper, we propose a localized density control algorithm based on weighted set-cover problem. Our algorithm is divided into two main steps. First of all, each sensor determines its redundancy status by using a concept of sponsored areas and a redundancy rule [4], [5] that checks whether its sensing area is completely covered by neighboring nodes or not. Secondly, when a set of redundant nodes has been discovered, these nodes will be put into a redundancy list used by our node scheduling algorithm based on a weighted set cover problem. Because the set cover problem is NP-hard, we apply an approximation algorithm called a greedy algorithm [6] for providing a guarantee on the quality of our solution. Nodes with the maximum number of covered areas and high remaining energy will be selected to be active first. These redundant nodes take turn to be active for ensuring that all nodes run out of energy about the same time. Furthermore, we also provide a voting protocol for node's solution management. From a result, it shows that our algorithm can extend longer network lifetime and use less number of active sensor nodes for monitoring while the coverage is preserved on working. Because only subset of nodes is active, the overall energy consumption and radio-traffic intensity are definitely reduced.

The rest of paper is organized as follows: section 2 reviews the related work in the literature. In section 3, a design of our density control algorithm is described. Performance evaluation and simulation results are shown in section 4 followed by the conclusion in section 5.

II. Related Work

Energy consumption minimization and network lifetime extension become two major challenges for wireless sensor network applications. Because of a limited energy resource, it is necessary to reduce the overall energy consumption for increasing the working lifetime of the network. Most of energy consumption is in wireless communication and microprocessor computation. Turning sensor nodes into a sleep mode can be an efficient solution for saving this energy resource. Many existing methods are presented in centralized and distributed approaches for density control. The main concept is to find a group of sensors that can sleep without losing the coverage property. Slijepcevic and Potkonjak [7] introduce a cover defined as a set of sensor nodes whose sensing areas completely cover the interested monitoring region. They propose a heuristic solution based on a Set k-Cover problem for calculating the number of covers. This number depends on the critical areas, the area detected by few sensor nodes. However, this solution is unclear about an implementation and practical use.

Ye, Zhong, Cheng, Lu, and Zhang [8] present the design of a distributed density control algorithm called PEAS. PEAS is a simple probing-based protocol that can extend system functioning time by keeping only a subset of sensor nodes working while the others are put into a sleep mode. Sleeping nodes wake up occasionally and probe around for the existence of working neighbors. If there is no response from any working neighbors in their local area, probing nodes will change their status to be active and replace the ones disappear; otherwise, they return to sleep again. This means that all of their working neighbors have to run out of power first before the sleepers replace them. This may result in network partitions in some cases. Additionally, this solution cannot ensure the coverage reservation although it may take a fewer number of active nodes, compared with other schemes.

Tian and Georganas [4] present a density control with complete coverage reservation while only a subset of active sensors is required. They propose two new concepts including coverage-based off-duty eligibility rule and backoff-based node-scheduling scheme. They implement their scheme in NS-2 as an extension of the LEACH protocol [9]. The sensing area of a node is defined as a sponsored area and sensor nodes that have sponsored areas overlapping with each other are called sponsored neighbors. The word neighbors are defined as nodes located at equal to or less than one sensing range from current node's location. Each sensor is assumed to have the same sensing range. From the definition of eligibility rule, nodes that have their sensing disk perimeter completely covered by their sponsored neighbors can be concluded that their sensing areas are also completely covered. These nodes are eligible for off duty and can be turned off.

Unfortunately, to avoid appearance of a blind area, these off-duty nodes cannot turn themselves off simultaneously. For this reason, a node scheduling has been proposed to manage their timetable for activation. The operation timeline of each node is divided into rounds and each round consists of three phases including scheduling, clustering, and sensing. Before any off-duty node makes a decision to turn off, it sends others a notification message to inform its status. After a period of time passes, if it does not receive any notification message from neighbors, it is safe to be off; otherwise, it recalculates its covered area again and starts a new message transmission. Due to the definition of neighbors that are only located no further than one sensing radius, the number of discovered off-duty nodes may be fewer than it should be. Although this solution can maintain the sensing coverage of the network, there are too many active nodes.

Jiang and Dou [5] address the above problem of the previous approach by improving the eligibility rule for increasing the number of redundant (or off-duty) nodes. They introduce a new concept of effective neighbors for redundancy determination. Effective neighbors can be located further than one sensing range. Similar to [4], if a node's sensing disk perimeter is completely covered by its effective neighbors, the node will be redundant. For node scheduling, they use the same one as in [4].

Zhang and Hou [10] propose another solution for redundancy determination. Each sensor node divides its own covering area into grid. Each cell in grid is represented by its central point. If all points in the sensing area of a node are located in the sensing areas of its neighbors, the node will be redundant. Even though the solution is rather simple, it requires excessive memory capacity for grid buffering.

Carbunar, Grama, Vitek, and Carbunar [11] study the problem of redundancy detection and elimination in a sensor network. They solve by reducing the problem to Voronoi

diagrams computation. The whole monitoring area is divided into a large number of Voronoi cells for coverage detection. Due to the limited resources of sensor device, Voronoi-based solution has high complexity in computation and maybe not suitable for sensor networks.

According to the previous works, the selection rule in node scheduling depends only on timer set randomly. This kind of method makes any redundant node has the same probability to be selected for activation. The remaining energy of node has not been considered in selection. It can result that node with low remaining energy maybe selected to be active before the higher. In our work, we emphasize the remaining energy factor in our node selection and apply a set-cover problem for node scheduling instead of random timer.

III. SET COVER-BASED DENSITY CONTROL ALGORITHM

In our localized density control algorithm, each node uses only their local information from neighbors for checking redundancy status. To spread sensing duty to nodes with high remaining energy, we weight our active node selection rule with the current remaining energy of nodes. This weighted rule can make our scheduling scheme more balanced while the sensing coverage is maintained.

There are some assumptions of sensor networks in our work as follow.

- Nodes are deployed in a two-dimensioned plane.
- Each node knows its own location represented by a coordinate (x, y).
- A sensing area is a disk and each node has the same communication range twice as far as the sensing range.
- Every node has the same initial energy quantity and consumption rate.

Our density control algorithm has been divided into two phases. In the first phase, each node determines its redundancy status by using the concept of perimeter coverage [4] and effective neighbors [5]. All redundant nodes will be taken into the second phase for node scheduling. This scheduling includes an active node selection (using a greedy solution for a weighted set-cover problem) and notification protocol based on voting. The design of our algorithm will be described in details as follows.

A. Redundancy determination

To have neighboring nodes' locations, each node exchanges some messages containing its own location with local neighbors. All locations received from neighbors are used as information for checking node's redundancy status. From the concept of perimeter coverage and effective neighbors, every redundant node has two main characteristics.

1) Perimeter coverage: All perimeter of a sensing area must be covered by the sensing area of its neighbors. Figure 1(a) shows an example of sensor node X whose perimeter (represented by a circle with node inside) has already been completely covered by seven neighboring nodes.

2) Overlapping segment of perimeter: For each effective neighbor, its segment of perimeter inside the considered node's sensing area must be covered by the sensing area of the other effective neighbors. Figure 1(b) shows that sensor node Y's segment of perimeter inside X's sensing area has already been covered by A's, B's, C's and D's sensing areas.

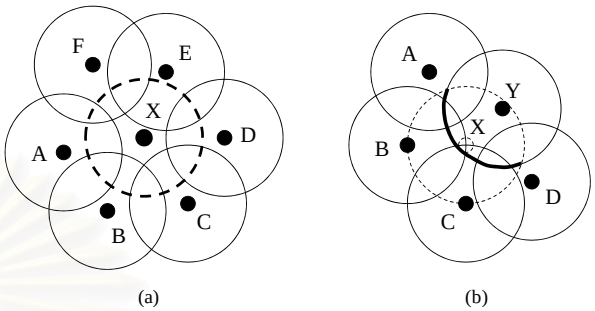


Figure 1. Redundancy charasteristics: (a) Perimeter coverage and (b) Overlapping segment of perimeter

Both of these characteristics are used as main conditions for redundancy determination. Consequently, sensor nodes which have completely these two redundancy characteristics are decided to be redundant.

B. Activation node scheduling

Because of high-density deployment, there is definitely more than one redundant node in the network. Some of these may be located at a nearby area. Figure 2(a) shows ten sensor nodes deployed. From redundancy determination, node X, Y and Z are found to be redundant and should be turned into a sleep mode. However, if all of these three redundant nodes decide to sleep at the same time, a blind area that is not detected by any sensor will exist. For this reason, we cannot turn every redundant node into a sleep mode simultaneously.

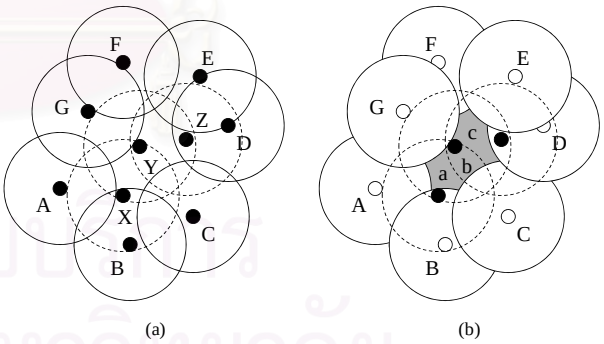


Figure 2. (a) Redundant nodes located at a nearby area and (b) Blind area appearance which is divided into three small regions

To avoid a blind area appearance, all redundant nodes need to properly schedule their working timeline. For longer network lifetime, the number of active redundant nodes should be minimized while the others can sleep for power savings. The selected ones should cover the largest critical areas.

In this paper, the whole monitoring area is divided into small regions. These small areas are bounded by sensing disk perimeters of some nodes. This problem can be considered a

set cover problem whereby each redundant node is treated as a set owner and the set members consist of those small areas it covers. As in figure 2(b), the whole blind area can be divided into three small areas including area a, b and c. Area a is covered by node X and Y so it will be a member of both X's and Y's set. Likewise, area c will be a member of Y's and Z's set while area b will be a member of all X's, Y's and Z's set. We can form a set cover problem model as (1). In this example, node Y should be chosen to be active since it can cover all critical areas a, b, and c by itself.

$$\begin{aligned} Universe &= \{a,b,c\} \\ X &= \{a,b\} \\ Y &= \{a,b,c\} \\ Z &= \{b,c\} \end{aligned} \tag{1}$$

Unfortunately, in a real environment, each node is not provided with such a nice picture of overlapping areas. In this paper, we propose an area searching protocol for computing a set of covering areas of each node. In this protocol, each node exchanges the location with its neighbors. Then, the node uses its neighboring nodes' locations for calculating crossing points between two nodes' perimeters. These crossing points are viewed as the vertices of an area search graph. According to the previous example, node X calculates its crossing points inside its sensing area by using Y's and Z's location. Figure 3 shows that node X has five crossing points including point 1, 2, 3, 4, and 5. These crossing points and the segments of perimeters (connecting between two crossing points) are considered vertices and edges in our area search graph. This graph will be put into an area searching process.

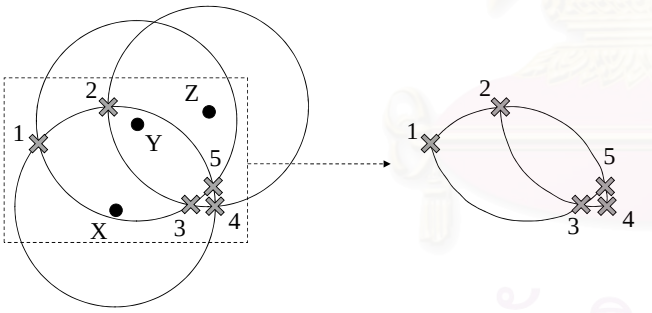


Figure 3. Area search graph using perimeter-crossing points as vertices

Area search graph can be considered as a set of several arrays. Each redundant node in the problem space has its own set of arrays for keeping the crossing points located on nodes' perimeters. All of these points in array are sorted by their positions in clockwise order referring to the center of circle (node's location). As in figure 4, for example, there are three redundant nodes located nearby in the problem space. Node X has its own set of three arrays consist of array X, Y and Z for keeping the crossing points located at node X's, Y's and Z's perimeter respectively. Due to five crossing points in node X's sensing area, array X will have four members consist of point 1, 2, 5 and 4. Same as array X, array Y and Z have point 5, 3, 1 and 4, 3, 2 as their members respectively.

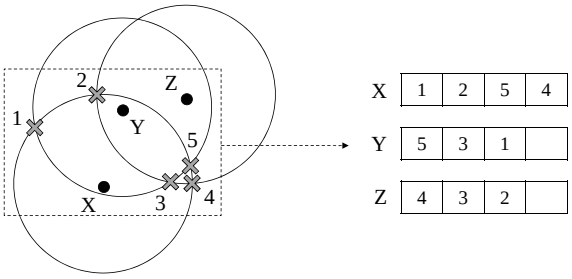


Figure 4. Arrays of crossing points located on nodes' perimeters

If current node searching areas is node P, one of P's vertices in array P is selected as a starting point and as the root of P's area search tree. From this starting point A, we traverse on an edge of the area search graph to the next vertex B. However, this next traversed edge cannot be on the same perimeter which the previous edge is on. This starting point A is considered to be on P's perimeter so the next vertex cannot be on P's perimeter. We can get this vertex B by considering at the vertex located adjacently to vertex A in the other array Q whose vertex A as its member. This vertex B is inserted into the area search tree as a child of vertex A. After the new vertex is inserted, we continue searching by selecting one child in our area search tree (in a breadth-first manner). This selected vertex C will be checked for a loop. If there is another vertex C at the higher level of the tree (vertex C's parent path to the root), we have a loop in our area search graph. As a result, this loop represents an overlapping area found in the computing node's set. The overlapping area can be represented by a list of vertices (a path from the higher vertex C toward the lower vertex C of the area search tree). After the overlapping area is found, we stop searching on this path (represented the overlapping area) in tree.

Otherwise, if there is no other vertex C at the higher level of the tree, we continue traversing from the current vertex C on the next edges (not be on the previous perimeter) of the area search graph and add the new vertices found as a children of vertex C in tree. This searching process will continue searching until there is no leaf vertex in tree has not been traversed (or all edges in area search graph have been traversed). As a result, a set of all node P's covering areas are definitely found.

In figure 5(a), node X chooses point 1 as the starting point. The next point is point 3 on Y's perimeter. From point 3, we cannot continue traversing on the same perimeter. Therefore, we can only traverse on Z's perimeter to point 2 or 4. Figure 5(b) shows an area search tree whose vertices are the crossing points gathered from the searching process. The final result shows that node X have three discovered critical areas including area (1, 3, 2, 1), (3, 2, 5, 3), and (3, 4, 5, 3). However, the areas only covered by one node are not included in our result because they cannot appear to be blind.

When the set of covering areas has been completely discovered, they exchange some messages containing their own sets including remaining energy with their neighbors. All sets received are used to form a set-cover problem. According to the greedy solution, a set with the most members will be selected first. This implies that the owner of the selected set is chosen to be active.

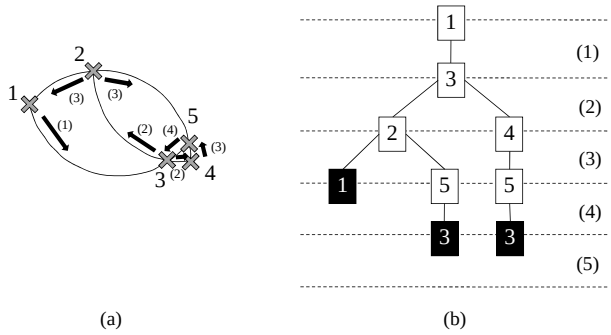


Figure 5. (a) Area searching procedure and (b) Area search tree

As a result, nodes with more covering areas are always selected. Given limited energy, these nodes will run out of energy before others. Consequently, the overall sensing coverage may not be preserved. As in figure 2(a), although node Y has the most covering areas, node X and Z should be activated to replace node Y once in a while. Therefore, we propose that the remaining energy of the node should be factored into the node selection process. Not surprisingly, our greedy rule includes remaining energy as a weight (see Equation (2)).

$$Cost-Effectiveness = Remaining\ energy \times \#Covering\ areas \quad (2)$$

The solution form the set cover problem consists of two node lists: an active and inactive list. From the above equation, higher remaining energy and a larger number of covering areas result in higher cost-effectiveness. Therefore, nodes with such properties will be selected to be active first and put into the active list. This is reasonable because these nodes deserve to work more than the others. After all nodes in the active list have covered the whole blind area already, the selection process stops and the remaining nodes unselected are put into the inactive list.

However, given that neighbors of each node are different, a set cover problem of a node differs from that of the other. (These problems are localized to avoid excessive overhead in forming a centralized problem.) Therefore, their solutions of active nodes may conflict with one another. Some nodes may be decided to be active by one node but the other nodes may prefer them to sleep.

In this paper, we address the above problem using a voting protocol. Figure 6 shows a state diagram of our voting scheme. Once the solution of the set cover problem is obtained in the state "Calculate set cover solution", the node sends a voting message containing its solution to its neighbors. In a sense, the sender votes that nodes on the active list should be activated and nodes on the inactive list should be deactivated. After the votes have been completely sent out, the node goes to the state "Collect solutions from nbrs" and waits for receiving votes from neighbors. Each node focuses on the votes about itself. When voting time is out, node goes to the next state "Calculate wait-to-sleep timer". In this state, the ratio of inactive and active votes is combined with the remaining energy to set up a wait-to-sleep timer. This timer is set up for listening deactivation announcements (receiving deactivation messages from neighbors) in such a way that the timer is shorter for the node with lower remaining energy and more votes to sleep.

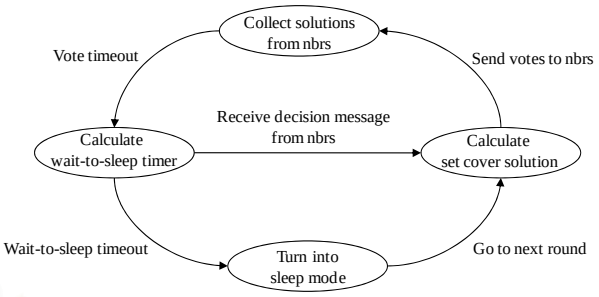


Figure 6. Voting state diagram

Once the wait-to-sleep time is out, the node is eligible to sleep. The node announces to its neighbors about its deactivation and safely goes to sleep in the state "Turn into sleep mode". Other nodes check whether these deactivation events conflict with their current solutions or not. If yes, these nodes go back to the state "Calculate set cover solution". Then, they re-compute their solutions and send new votes to their neighbors again.

IV. Performance Evaluation and Simulation

In this section, we evaluate our algorithm performance on a TinyOS [12] simulator called TOSSIM [13]. Our simulation code is written in nesC [14], a programming language for wireless sensor networks. The monitoring area is a square of 100x100 units. Each sensor has the same sensing range. There is no packet loss in our simulation. In the first part of our evaluation, we compare the performance of four redundancy-determination algorithms. Table I shows a percentage of an average false error (non-identical result between theoretical and simulation) for varying number of simulated sensor nodes. The result indicates that a hybrid approach of perimeter and effective neighbors can reduce false errors and improve the accuracy in redundancy determination.

TABLE I. Results of Simulation for Redundancy Determination

Total number of nodes	**Average false error (%)**			
	Grid partition	***Perimeter coverage***	***Overlapping segment of perimeter***	***Hybrid***
50	4.2	13.8	4.4	3.6
100	12.7	38.6	11.3	9.7
150	18.5	51.3	19.3	15.8
200	25.5	57.3	26.5	22.5
250	29.0	59.8	29.5	25.0
300	29.3	62.0	31.0	27.0
350	32.0	63.0	32.0	24.0

To evaluate our node scheduling algorithm, we compare our set cover approach with PEAS [8] and randomized timer [4] approaches. We measure the number of active nodes required for each approach. As in figure 7, under the assumption of fixed monitoring area through simulation, all approaches can reduce and preserve the number of active nodes varying between 50 and 90. According to the same total node

number, our protocol requires a fewer number of active sensor nodes than the randomized timer approach because it attempts to activate redundant nodes that cover more areas first. Unfortunately, our protocol requires more number of active nodes than PEAS. However, while using PEAS, redundant nodes are turned off without any coverage consideration. Due to our set cover -based algorithm, each part of the monitoring area is covered by at least one sensor node. Therefore, it can guarantee that the original sensing coverage of the network will be conserved.

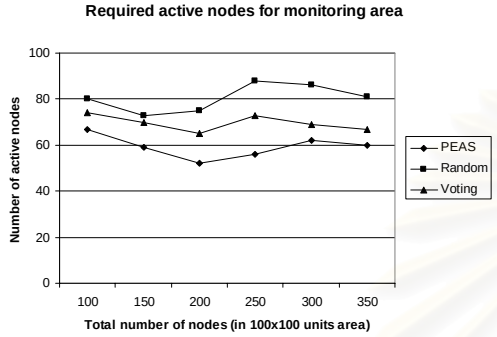


Figure 7. Number of active nodes vs. total number of nodes

In addition, we measure the network lifetime of these approaches (see figure 8). Given that, in our approach, the remaining energy of a node is factored into the node selection rule, nodes with higher energy will likely be selected. Hence, the network lifetime can be extended longer than previous algorithms.

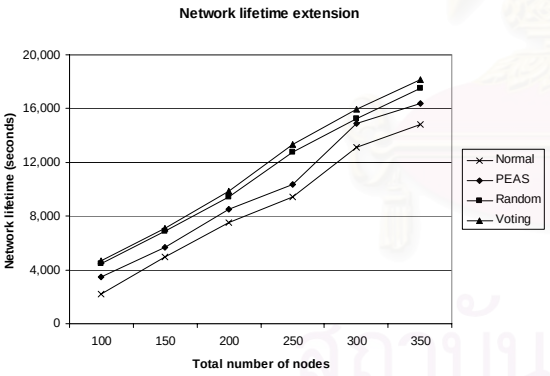


Figure 8. Network lifetime vs. total number of nodes

V. CONCLUSIONS

In wireless sensor networks, several sensor network applications require a high density of sensors to accurately collect data. These networks may consume excessive energy as there are several redundant sensors that cover the same sensing areas. To address this problem, we propose a set cover-based density control algorithm that can reduce the overall energy consumption by putting some subsets of redundant nodes into a sleep mode. This sensing coverage problem is transformed into a weighted set-cover problem. We solve this problem using a greedy approximation algorithm weighted by the remaining energy of nodes.

For practical use, we also introduce an area searching protocol for determining the covering areas of a node. In addition, a voting protocol has been provided for finalizing the active node selection given that our localized solutions may conflict with one another. TinyOS technology and tools (e.g., TOSSIM simulator) have been used in our implementation and evaluation of this work. The result indicates that our approach is efficient, practical, and viable for wireless sensor networks.

REFERENCES

[1] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," in Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking Seattle, Washington, United States: ACM, 1999, pp. 263-270.

[2] C. F. Huang and Y. C. Tseng, "The Coverage Problem in a Wireless Sensor Network," in Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications San Diego, CA, USA: ACM, 2003, pp. 115-121.

[3] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in Wireless Ad-Hoc Sensor Networks," in Proceedings of the 7th Annual International Conference on Mobile Computing and Networking Rome, Italy: ACM, 2001, pp. 139-150.

[4] D. Tian and N. D. Georganas, "A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks," in Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications Atlanta, Georgia, USA: ACM, 2002, pp. 32-41.

[5] J. Jiang and W. Dou, "A Coverage-Preserving Density Control Algorithm for Wireless Sensor Networks," in Ad-Hoc, Mobile, and Wireless Networks, 2004, pp. 42-55.

[6] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to Algorithms, Second Edition: The MIT Press, 2001.

[7] S. Slijepcevic and M. Potkonjak, "Power Efficient Organization of Wireless Sensor Networks," in Communications, 2001. ICC 2001. IEEE International Conference on, 2001, pp. 472-476.

[8] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang, "PEAS: A Robust Energy Conserving Protocol for Long-lived Sensor Networks," in Proceedings of the 23rd International Conference on Distributed Computing Systems: IEEE Computer Society, 2003, pp. 28-37.

[9] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on, 2000.

[10] H. Zhang and J. C. Hou, "Maintaining Sensing Coverage and Connectivity in Large Sensor Networks," International Journal of Wireless Ad Hoc and Sensor Networks, vol. 1, pp. 89-124, 2005.

[11] B. Carbunar, A. Grama, J. Vitek, and O. Carbunar, "Coverage Preserving Redundancy Elimination in Sensor Networks," in Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. First Annual IEEE Communications Society Conference on, 2004, pp. 377-386.

[12] P. Levis, University of California at Berkeley, "TinyOS Tutorial, The tutorial introduces the basic concepts of TinyOS through a set of simple applications and exercises," September 2003, http://www.tinyos.net/tinyos-1.x/doc/tutorial/index.html.

[13] 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: ACM, 2003, pp. 126-137.

[14] D. Gay, P. Levis, R. V. Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC Language: A Holistic Approach to Networked Embedded Systems," in Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation. vol. 38 San Diego, California, USA: ACM, 2003, pp. 1-11.

ประวัติผู้เขียนวิทยานิพนธ์

นายศรัณย์ เจนจตุรงค์ เกิดเมื่อวันที่ 7 มีนาคม พ.ศ. 2527 ที่จังหวัดนนทบุรี สำเร็จการศึกษาระดับมัธยมศึกษาตอนต้นจากโรงเรียนเบญจมราชานุสรณ์ จังหวัดนนทบุรี และมัธยมศึกษาตอนปลายจากโรงเรียนเตรียมอุดมศึกษา กรุงเทพมหานคร สำเร็จการศึกษาปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2548 และเข้าศึกษาในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2549