โพรโทคอลเครือข่ายสำหรับฟาร์มอัจฉริยะ
HYBRID AD HOC CENTRALIZED ROUTING AND NETWORK PROTOCOL FOR SMART FARM

ธนภัทร สุนทรลิ้มศิริ
ภาธร ไพยรัตน์
ภูกิจ เสือเสนา

ปัญหาพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์)
ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2559

โพรโทคอลเครือข่ายสำหรับฟาร์มอัจฉริยะ

HYBRID AD HOC CENTRALIZED ROUTING AND NETWORK PROTOCOL FOR SMART FARM

ธนภัทร สุนทรลิ้มศิริ

ภาธร ไพยรัตน์

ภูกิจ เสือเสนา

ปัญหาพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์)

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2559

HYBRID AD HOC CENTRALIZED ROUTING AND NETWORK PROTOCOL FOR SMART FARM

TANAPAT SOONTORNLIMSIRI

PATORN PAIYARAT

PHUGIT SUASENA

A SPECIAL PROBLEM SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR

THE DEGREE OF BACHELOR OF SCIENCE (COMPUTER SCIENCE)

DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF SCIENCE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ACADEMIC YEAR 2016

หัวข้อปัญหาพิเศษ	โพรโทคอลเครือข่ายสำหรับฟาร์มอัจฉริยะ Hybrid Adhoc Centralized Routing and Network Protocol for Smart Farm
ชื่อนักศึกษา	นาย ธนภัทร สุนทรลิ้มศิริ รหัสนักศึกษา 56050267 นาย ภาธร ไพยรัตน์ รหัสนักศึกษา 56050341 นาย ภูกิจ เสือเสนา รหัสนักศึกษา 56050343
ปริญญา	วิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์)
ภาควิชา	วิทยาการคอมพิวเตอร์
ปีการศึกษา	2559
อาจารย์ที่ปรึกษา	ดร.ไพรัตน์ ธรเจริญศรี

คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (สจล.) อนุมัติให้ปัญหาพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์) ประจำปีการศึกษา 2559

คณะกรรมการสอบ	ลายมือชื่อ
ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์ ประธานกรรมการ	
รศ.ธีรวัฒน์ ประกอบผล กรรมการ	
ดร.ไพรัตน์ ธรเจริญศรี กรรมการและอาจารย์ที่ปรึกษา	

หัวข้อปัญหาพิเศษ โพรโทคอลเครือข่ายสำหรับฟาร์มอัจฉริยะ
ชื่อนักศึกษา นายธนภัทร สุนทรลิ้มศิริ รหัสนักศึกษา 56050267
นายภาธร ไพยรัตน์ รหัสนักศึกษา 56050341
นายภูกิจ เสือเสนา รหัสนักศึกษา 56050343
ปริญญา วิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์)
ภาควิชา วิทยาการคอมพิวเตอร์
คณะ วิทยาศาสตร์
มหาวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (สจล.)
ปีการศึกษา 2559
อาจารย์ที่ปรึกษา ดร.ไพรัตน์ ธรเจริญศรี

บทคัดย่อ

ในปัจจุบัน เครือข่ายเซ็นเซอร์ไร้สายนั้นได้ถูกนำมาใช้งานอย่างแพร่หลาย ทั้งการเฝ้าติดตามและการจัดการดูแลด้านการเกษตร ช่วยให้เกษตรกรสามารถเข้าถึงการใช้งานเทคโนโลยีได้อย่างมีประสิทธิภาพ แต่เครือข่ายเซ็นเซอร์ส่วนใหญ่เป็นเครือข่ายแบบรวมศูนย์กลาง โดยโหนดทั้งหมดจะมีการติดต่อสื่อสารไปที่โหนดศูนย์กลาง ทำให้ได้พื้นที่การทำงานของเครือข่ายที่น้อย ดังนั้นจึงต้องสร้างเครือข่ายให้มีระยะการทำงานของเครือข่ายที่มากขึ้น ดังนั้นจึงได้นำเสนอเครือข่ายเฉพาะกิจมาช่วยแก้ปัญหานี้ เครือข่ายนี้ไม่จำเป็นต้องมีการวางโครงสร้างเครือข่ายและมีความซับซ้อนน้อย แต่เครือข่ายนี้ยังไม่เป็นที่นิยมทางด้านการเกษตรและโพรโทคอลนี้ยังมีประสิทธิภาพน้อยลง ถ้าเครือข่ายมีขนาดใหญ่มากขึ้น ทำให้ต้องมีการพัฒนาโพรโทรคอลให้มีเหมาะสมและสามารถทำงานได้มีอย่างประสิทธิภาพ

จากปัญหาดังกล่าว งานวิจัยนี้ได้นำเสนอโพรโทคอล Hybrid Ad hoc Centralize protocol for Smart Farm (HAC-SF) ซึ่งถูกพัฒนามาจากโพรโทคอล Ad-hoc On-Demand Distance Vector (AODV) ซึ่งเป็นโพรโทคอลเครือข่ายเฉพาะกิจที่ได้รับความนิยม โพรโทคอล HAC-SF มีการพัฒนาเกี่ยวกับวิธีการติดต่อสื่อสาร อัลกอริทึมในการค้นหาเส้นทางและการซ่อมแซมเส้นทาง โดยโพรโทคอลนี้ถูกนำมาทดสอบด้วยโปรแกรมจำลองเครือข่าย Network Simulation 3 (NS-3) ซึ่งผลลัพธ์จากการจำลองการทำงานคือ สามารถเพิ่มประสิทธิภาพของการทำงานเพิ่มมากขึ้น ได้แก่ ค่าดีเลย์ และ ค่าเปอร์เซ็นต์การรับส่งสำเร็จ เมื่อเทียบกับโพรโทคอล AODV เดิม

คำสำคัญ : เครือข่ายเฉพาะกิจ เครือข่ายเซ็นเซอร์ไร้สาย เครือข่ายแบบรวมศูนย์กลาง
โพรโทคอลเอโอดีวี ฟาร์มอัจฉริยะ

Title Hybrid Ad hoc Centralized Routing and Network Protocol for Smart Farm

Students Mr. Tanapat Soontornlimsiri Student ID 56050267
Mr. Patorn Paiyarat Student ID 56050341
Mr. Phugit Suasena Student ID 56050343

Degree Bachelor of Science (Computer Science)

Department Computer Science

Faculty Science

University King Mongkut's Institute of Technology Ladkrabang (KMITL)

Academic Year 2016

Advisor Dr.Pairat Thorncharoensri

Abstract

Wireless sensor network (WSN) had been used extensively in agriculture. However, most wireless sensor networks are centralized networks that all nodes communicate to their gateways. This architecture can cover only a small area. Therefore, ad hoc network comes to be useful for solving the area size's problem. This ad hoc network needs only one gateway. Also, it has low node density and less redundancy network. Nevertheless, the protocol has less packet delivery success rate and low reliability in large network.

According to these problems, this thesis proposes Hybrid Ad hoc Centralized for Smart Farm protocol (HAC-SF) based on AODV protocol which is a popular protocol for ad hoc network. The results from the simulation using Network Simulator 3 (NS-3) program indicated that HAC-SF is more efficient than the original AODV when the number of nodes has increased in the network.

Keywords : ad hoc network, wireless sensor network, centralized network, AODV protocol, smart farm

กิตติกรรมประกาศ

หัวข้อปัญหาพิเศษนี้มีจุดหมายเพื่อวิจัยเครือข่ายเมชไร้สายให้มีความรวดเร็วและส่งข้อมูลได้มีความถูกต้องมากขึ้น ผู้วิจัยขอขอบคุณภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมกล้าเจ้าคุณทหารลาดกระบังที่ได้สนับสนุนอุปกรณ์และสถานที่สำหรับงานวิจัยในครั้งนี้

การดำเนินงานวิจัยนี้มิอาจสำเร็จลุล่วงไปได้หากปราศจาก ดร.รุ่งรัตน์ เวียงศรีพนาวัลย์ และ ดร.ไพรัตน์ ธรเจริญศรี ที่ได้ให้คำปรึกษา แนะนำขั้นตอนและวิธีจัดทำงานวิจัยให้สำเร็จลุล่วงไปด้วยดี อีกทั้งขอขอบพระคุณเจ้าของเอกสารและงานวิจัยทุกฉบับ ที่ผู้วิจัยได้นำมาอ้างอิงในการทำงานวิจัยจนกระทั่งงานวิจัยสำเร็จลุล่วงไปได้ด้วยดี

ธนภัทร สุนทรลิ้มศิริ
ภาธร ไพยรัตน์
ภูกิจ เสือเสนา

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญตาราง (ต่อ)

ตารางที่ **หน้า**

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในยุคปัจจุบัน เทคโนโลยีได้รับการเปลี่ยนแปลงและพัฒนาไปอย่างรวดเร็ว เพื่ออำนวยความสะดวกสบายให้กับมนุษย์มากที่สุด ทุกสิ่งทุกอย่างจึงต้องใช้เทคโนโลยีเข้ามาช่วยบริหารจัดการระบบให้ระบบมีความรวดเร็ว ปลอดภัย โดยมีการพัฒนาเทคโนโลยีทางด้านอิเล็กทรอนิกส์ การสื่อสารโทรคมนาคม และเครือข่ายคอมพิวเตอร์ให้มีความเจริญก้าวหน้ามากขึ้น จึงเป็นการเข้าสู่ยุคดิจิทัลที่อุปกรณ์อิเล็กทรอนิกส์แต่ละอุปกรณ์สามารถคุยกันเองได้ผ่านการเชื่อมต่ออินเตอร์เน็ต เช่น รถยนต์สามารถคุยกับบ้านได้ การสั่งให้เปิดแอร์ผ่านระบบในรถก่อนที่จะขับรถถึงบ้าน จึงทำให้ทุกอย่างดูสะดวกและเชื่อมโยงถึงกันหมด ด้วยเหตุนี้จะต้องมีระบบมารองรับและจัดการด้านต่าง ๆในชีวิตประจำวันให้เข้าสู่ระบบดิจิทัล

ในงานวิจัยนี้ได้พัฒนาระบบเครือข่ายเมชไร้สายสำหรับการเกษตรกรรม เพื่อให้มีสิ่งอำนวยความสะดวกในการเฝ้าติดตาม และการจัดการดูแลฟาร์มเกษตรกรรมได้มากขึ้นจนเกิดเป็นสิ่งที่เรียกว่า “ระบบฟาร์มอัจฉริยะ (Smart farm system)” ที่จะช่วยให้เกษตรกรสามารถเข้าถึงการใช้งานเทคโนโลยีได้อย่างมีประสิทธิภาพ จนกลายมาเป็นอีกชื่อที่เรียกว่า “เกษตรกรรมความแม่นยำสูง (Precision agriculture)” แต่ในประเทศไทยยังไม่นิยมกับการนำเกษตรกรรมความแม่นยำสูงนี้มาใช้งาน แต่เกษตรกรรมความแม่นยำสูงนี้เป็นที่นิยมกันมากใน ประเทศสหรัฐอเมริกา และ ออสเตรเลีย และเริ่มแพร่หลายเข้าไปในหลายประเทศทั้งยุโรป ญี่ปุ่น หรือแม้กระทั่งประเทศมาเลเซียและอินเดียก็เริ่มมีการทดลองใช้เทคโนโลยีนี้กันอย่างกว้างขวาง ดังนั้นประเทศไทยจึงจำเป็นต้องให้ความสนใจในเรื่องนี้ให้มากขึ้น เพราะในอนาคตเทคโนโลยีนี้จะถูกนำมาใช้ในประเทศเพื่อนบ้านมากขึ้น และอาจทำให้ประเทศไทยสูญเสียโอกาสในการเจริญเติบโตทางเศรษฐกิจทางด้านการเกษตร โดยที่ประเทศไทยนั้นก็มีพื้นที่เกษตรกรรมขนาดใหญ่กว่ามาก จึงจำเป็นต้องมีการวิจัยและพัฒนาเทคโนโลยีนี้ให้ก้าวหน้ามากขึ้นให้ได้

งานวิจัยนี้จึงนำเสนอการพัฒนาโพรโทคอลเครือข่ายไร้สายมาใช้ร่วมกับเกษตรกรรมความแม่นยำสูง เพื่อเป็นการพัฒนาระบบในการติดต่อสื่อสารในภาคการเกษตรให้มีประสิทธิภาพมากขึ้น ซึ่งเครือข่ายที่พัฒนาขึ้นจะสามารถส่งข้อมูลจากโหนดหนึ่งไปสู่โหนดอื่น ๆได้โดยตรงไม่ต้องมีตัวกลางในการสื่อสาร โดยมีการติดตั้งอยู่ในบริเวณพื้นที่การเกษตรที่แตกต่างกันเข้าสู่ศูนย์กลางการควบคุมผ่านเครือข่ายสื่อสารไร้สายและมีโพรโทคอลที่ชื่อว่า “HAC-SF” ย่อมาจาก “Hybrid Ad hoc Centralized Routing and Network Protocol for Smart Farm” โดยโพรโทคอลนี้สามารถนำไป

ประยุกต์ใช้ในการจัดการฟาร์มเกษตรกรรมในรูปแบบของระบบฟาร์มอัจฉริยะ อีกทั้งสามารถนำไปประยุกต์ใช้เพิ่มประสิทธิภาพของการใช้ทรัพยากรให้เป็นไปอย่างแม่นยำและตรงต่อความต้องการของพืช จึงสามารถช่วยในการเพิ่มผลผลิตและช่วยลดการสูญเสียทรัพยากรได้

1.2 วัตถุประสงค์ของงานวิจัย

1. เพื่อศึกษาเทคโนโลยีและการทำงานของเครือข่ายไร้สาย
2. เพื่อศึกษาการออกแบบเครือข่ายไร้สายให้เหมาะสมสำหรับเกษตรกรรมความแม่นยำสูง
3. เพื่อดำเนินการออกแบบและพัฒนาโพรโทคอลสำหรับเครือข่ายไร้สาย ในส่วนการควบคุมการส่งข้อมูลที่ตรวจวัดได้จากอุปกรณ์ตรวจวัดต่าง ๆให้มีประสิทธิภาพมากขึ้นและสามารถนำไปใช้กับเกษตรกรรมความแม่นยำสูงได้
4. เพื่อสร้างองค์ความรู้ใหม่ในการพัฒนาเครือข่ายไร้สายและเพิ่มประสิทธิภาพทางด้านเทคโนโลยีให้กับการทำเกษตรกรรมความแม่นยำสูง

1.3 ขอบเขตของงานวิจัย

1. มีการจำลองเครือข่าย (Network Simulation) เพื่อเปรียบเทียบความสามารถของโพรโทคอล ในการสื่อสารไร้สาย โดยงานวิจัยนี้ได้นำโพรโทคอลมาใช้กับเครือข่ายไร้สายแบบ hybrid ad hoc centralized
2. มีการออกแบบและพัฒนาโพรโทคอลค้นหาเส้นทางและการสื่อสารให้มีประสิทธิภาพมากขึ้น
3. มีการเปรียบเทียบประสิทธิภาพของโพโทคอล AODV และโพรโทคอล HAC-SF ที่พัฒนามาจากโพรโทคอล AODV

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้องค์ความรู้ในด้านการออกแบบเครือข่ายไร้สาย
2. ได้ออกแบบและพัฒนาโพรโทคอลควบคุมการส่งข้อมูลตรวจวัดและการสื่อสารของเครือข่ายเมชไร้สาย
3. ได้องค์ความรู้ใหม่จากการวิจัย เพื่อที่จะสามารถนำไปใช้เพื่อเป็นแนวทางสำหรับงานวิจัยต่อไปในอนาคต
4. ทราบถึงคุณภาพการใช้งานและข้อจำกัดของโพรโทคอล HAC-SF ในเครือข่ายไร้สายที่มีสภาพแวดล้อมและตัวแปรต่างๆ ที่ใช้งานในประเทศไทย เพื่อให้สามารถวางแผน ปรับปรุงและสามารถนำมาใช้ได้จริงในอนาคต

1.5 ขั้นตอนและวิธีการดำเนินงานวิจัย

ขั้นตอนการดำเนินงานงานวิจัยมีทั้งหมด 8 ขั้นตอนเริ่มจากเดือนธันวาคม 2559 และสิ้นสุดเดือนพฤษภาคม 2560 โดยมีรายละเอียดขั้นตอนการดำเนินงานดังนี้

ขั้นที่ 1 : ศึกษาแนวทางและวิธีการดำเนินงานวิจัย

ขั้นที่ 2 : ศึกษาการทำงานของโพรโทคอล AODV

ขั้นที่ 3 : ศึกษาการใช้งานโปรแกรมจำลองระบบเครือข่าย NS-3

ขั้นที่ 4 : ออกแบบ พัฒนาและปรับปรุงโพรโทคอล HAC-SF

ขั้นที่ 5 : เปรียบเทียบการใช้โพรโทคอล HAC-SF กับโพรโทคอล AODV ดังเดิม

ขั้นที่ 6 : ปรับปรุงแก้ไขโพรโทคอล HAC-SF เพื่อให้มีความเหมาะสมมากยิ่งขึ้น

ขั้นที่ 7 : ทดสอบระบบเครือข่ายทั้งระบบ

ขั้นที่ 8 : สรุปผลและจัดทำรายงานฉบับสมบูรณ์

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องที่นำมาใช้ในงานวิจัยนี้ โดยหัวข้อที่ 2.1 จะกล่าวถึงแนวคิดของเกษตรกรรมความแม่นยำสูงและตัวอย่างของเทคโนโลยี หัวข้อที่ 2.2 จะกล่าวถึงเครือข่ายเมชไร้สายในเรื่องการทำงาน โครงสร้างพื้นฐาน โพรโตคอลในการค้นหาเส้นทางเกี่ยวกับองค์ประกอบและประเภทของโพรโตคอล หัวข้อที่ 2.3 จะกล่าวถึงโพรโทคอล AODV การทำงานของโพรโทคอล ในการค้นหาเส้นทางนี้ทั้งหมด หัวข้อที่ 2.4 จะกล่าวถึงมาตรฐาน IEEE 802.11b/g/n หัวข้อที่ 2.5 จะกล่าวถึงโปรแกรมจำลองการทำงานเครือข่ายต่างๆ หัวข้อที่ 2.6 จะกล่าวถึงการหาค่าประสิทธิภาพการทำงานของระบบเครือข่ายที่มีการนำมาใช้เป็นตัวชี้วัดในโปรแกรมจำลองเครือข่าย และหัวข้อสุดท้ายคือ งานวิจัยที่เกี่ยวข้องทั้งหมด

2.1 Precision Agriculture (เกษตรกรรมแม่นยำสูง)

เกษตรกรรมความแม่นยำสูง (precision agriculture) [1] เป็นการทำเกษตรที่ยึดหลัก “ควบคุมการใช้ปัจจัยการผลิตให้ถูกที่และถูกเวลา” โดยอาศัยเทคโนโลยีมาช่วยในการจัดการ ทำให้การเพาะปลูกหรือเลี้ยงสัตว์มีประสิทธิภาพดีขึ้น เพิ่มปริมาณและคุณภาพผลผลิต ตลอดจน ช่วยลดผลกระทบต่อสิ่งแวดล้อมและพลังงานที่ลดลง

โดยทั่วไปแล้วเกษตรกรมักจะใช้ปัจจัยการผลิต (น้ำ ปุ๋ย สารปราบศัตรูพืช) อย่างเท่าเทียมกันทั่วทั้งพื้นที่การเกษตร โดยไม่สนใจตัวแปรอื่นๆ จนบางครั้งมีการใช้ปัจจัยการผลิตมากเกินไป ทำให้เกิดการรั่วไหลของแร่ธาตุและสารอาหารลงดินและแหล่งน้ำ ก่อให้เกิดผลเสียต่อคุณภาพดินและสร้างมลพิษทางน้ำ แต่เกษตรกรรมความแม่นยำสูงจะใช้วิธีสังเกต ตรวจวัด และควบคุมการใช้ปัจจัยการผลิตให้เหมาะสมกับความต้องการของพืชหรือสัตว์ในแต่ละพื้นที่หรือในแต่ละช่วงเวลา จึงช่วยลดความสูญเสียและใช้ทรัพยากรได้อย่างมีประสิทธิภาพมากที่สุด

<u>ตัวอย่างเทคโนโลยีเกษตรกรรมความแม่นยำสูง</u>

- เทคโนโลยีการรับรู้ระยะใกล้ (Proximal sensing) หรือเซ็นเซอร์เพื่อตรวจวัดค่าต่างๆ ใน จุดที่สนใจ เช่น สภาพดิน อากาศ โรคพืช และระดับผลผลิต โดยข้อมูลที่ได้จะถูกนำมาแปลงเป็นแผนที่เชื่อมโยงตัวแปรต่างๆ กับการบริหารจัดการด้านการเกษตรให้เหมาะสม

- ระบบระบุพิกัดจากดาวเทียม (Global Navigation Satellite System: GNSS) เป็นเทคโนโลยีที่ใช้มากที่สุด โดยถูกนำมาต่อยอดใช้กับ Controlled Traffic Farming (CTF) ที่ควบคุมการเคลื่อนไหวของเครื่องจักรแบบอัตโนมัติ และเทคโนโลยี Variable Rate Technology (VRT) ควบคุม

การหว่านเมล็ดพืช ความหนาแน่น การใช้ปุ๋ยและสารปราบศัตรูพืชได้อย่างแม่นยำ จึงช่วยลดต้นทุนค่าปัจจัยการผลิตและลดผลกระทบต่อสิ่งแวดล้อม

- การสำรวจจากระยะทางไกล (Remote Sensing: RS) เป็นการถ่ายภาพจากทางไกลด้วยดาวเทียม เพื่อสำรวจศักยภาพการผลิต ความต้องการหรือขาดแคลนสารอาหารในแต่ละพื้นที่ ซึ่งเหมาะกับ พื้นที่ขนาดใหญ่ ข้อมูลที่ได้มักถูกใช้ในการวางนโยบายระดับรัฐบาลหรือด้านการทหารมากกว่า แต่ปัจจุบัน เทคโนโลยี RS มีราคาถูกลงและใช้งานได้ง่ายขึ้น เช่น อากาศยานไร้คนขับที่เรียกว่า UAV หรือ โดรน จึงถูก นำมาใช้ประโยชน์ในการสำรวจความหลากหลายของพื้นที่เกษตร

- ระบบสารสนเทศภูมิศาสตร์ (Geographic Information Systems: GIS) เป็นการเก็บรวบรวมข้อมูลจากระบบเซ็นเซอร์ต่างๆ และทำเป็นแผนที่แสดงพิกัดภูมิศาสตร์ แต่การแปลข้อมูลจาก GIS ต้องอาศัยความรู้ความชำนาญเพื่อหาความสัมพันธ์ระหว่างอากาศ สภาพดิน และการเติบโตของพืช ข้อมูลจาก GIS สามารถใช้ในการวางแผนควบคุมการใช้ปัจจัยการผลิตให้เหมาะกับความต้องการของพืช

นอกจากนี้ยังมีเทคโนโลยีเกษตรกรรมความแม่นยำสูงอื่นๆ เช่น การเก็บตัวอย่างและตรวจวิเคราะห์คุณภาพและองค์ประกอบในดิน (Sampling Location) ระบบติดตามการเติบโตและระบบควบคุมและคัดแยกคุณภาพผักและผลไม้ (Machine Vision Systems) ระบบติดตามการขนส่งปศุสัตว์ (Tracking Livestock Transporting) และโปรแกรมช่วยบริหารจัดการและตัดสินใจในการทำเกษตร (Farm Management Information Systems) เป็นต้น

ประโยชน์ของเกษตรกรรมความแม่นยำสูงที่มีต่อเกษตรกร เจ้าของฟาร์ม มีดังนี้

- เกิดการลดต้นทุน
- เกิดผลผลิตสูงสุด ทั้งในแง่ปริมาณและคุณภาพที่เหมาะสมกับสภาพพื้นที่แต่ละส่วนในฟาร์ม
- เกิดการใช้ทรัพยากรอย่างคุ้มค่า
- รักษาสภาพแวดล้อม ซึ่งสามารถนำไปสู่กระบวนการผลิตอาหารที่มีคุณภาพและปลอดภัย

รูปที่ 2.1 กรณีตัวอย่างสวนไวน์ใน Napa Valley ซึ่งมีการนำเทคโนโลยี Smart Farm มาใช้กันแพร่หลาย

2.2 Wireless mesh networks (WMNs)

Wireless Mesh Networks (WMNs) [2] เกิดจากความต้องการที่จะแก้ปัญหาในเรื่องของการเดินสายเคเบิลเพื่อเชื่อมต่อจุดเข้าถึง (access point) ทุกตัว ระบบเครือข่ายเมชไร้สายเป็นเทคโนโลยีเครือข่ายไร้สายที่ทำให้จุดเข้าถึงสามารถส่งผ่านข้อมูลได้โดยตรงแบบไร้สาย ไม่ต้องผ่านสายเคเบิล ไม่จำเป็นต้องใช้สถานีฐาน ดังนั้นเมื่ออุปกรณ์สื่อสารไร้สายหรือเรียกว่า โหนด อยู่ในระยะการสื่อสาร โหนดสามารถติดต่อกันได้โดยตรง (Peer to Peer) แต่ในกรณีที่โหนดต้นทาง (Source node) ต้องการส่งข้อมูล แต่ไม่สามารถส่งข้อมูลได้โดยตรงกับโหนดปลายทาง (Destination node) ได้โดยตรง โหนดต้นทางจำเป็นต้องส่งข้อมูลไปยังโหนดอื่นๆในเครือข่ายที่อยู่ใกล้เคียงแทน เพื่อใช้เป็นเส้นทางในการสื่อสาร โดยมีการจัดการเส้นทางการสื่อสารด้วยโพรโทคอลการค้นหาเส้นทาง (Routing Protocol) ทำให้การสร้างและการจัดวางเครือข่ายเมชไร้สาย มีต้นทุนต่ำลง มีความยืดหยุ่นในระบบจึงสามารถขยายหรือปรับเปลี่ยนเครือข่ายได้ จึงสามารถใช้ได้ทั้งเครือข่ายไร้สายแบบภายในและภายนอกอาคาร หรือ อาคารทั้งขนาดเล็กหรือขนาดใหญ่ได้

WMNs จะประกอบไปด้วย Mesh routers และ Mesh clients ส่วนแรกคือ Mesh Routers หรือเรียกว่า gateway ใช้เชื่อมต่อกับสายแลนเพื่อเชื่อมต่ออินเทอร์เน็ตและมีความสามารถในการค้นหาเส้นทาง ซึ่งมีอยู่ใน wireless routing ปกติทั่วไป แต่ Mesh routers จะเพิ่มเติมในส่วนของการสนับสนุนการติดต่อสื่อสารหลายโหนดที่เชื่อมต่อครอบคลุมถึงกันให้สามารถติดต่อสื่อสารกัน

ได้ Mesh routers เป็นอุปกรณ์ที่ใช้ได้กับการเชื่อมต่อแบบไร้สายหลากหลายชนิดบนพื้นฐานหรือเทคโนโลยีไร้สายที่แตกต่างกัน ส่วน Mesh clients เป็นโหนดลูกข่ายของ Mesh routers มีความสามารถของ Mobile node ทำให้มีการสัญจรที่ดี แต่มีข้อจำกัด คือ ต้องใช้แบตเตอรี่ ทำให้พลังงานที่มีใช้งานมีจำกัด

WMNs ได้นำความสามารถและข้อดีของเครือข่าย ad hoc มาปรับใช้คือ ประหยัดการลงทุน บำรุงรักษาได้ง่าย มีความแข็งแรงคงทน มีความน่าเชื่อถือ และมีการให้บริการที่ครอบคลุม เป็นต้น

Wireless mesh networks นั้นมีข้อดีดังต่อไปนี้

- ระบบใช้สายน้อยลง ทำให้ต้นทุนลดน้อยลง โดยเฉพาะเมื่อต้องการวางระบบเครือข่ายที่ครอบคลุมเนื้อที่กว้าง
- การเพิ่มจำนวนโหนดที่ใช้ ทำให้ความเร็วในการส่งข้อมูลภายในเพิ่มขึ้น
- มีความสะดวก เพราะสามารถใช้กับสถานที่ที่ยากต่อการติดตั้งหรือวางสาย เช่น ในสนามกีฬาขนาดใหญ่ โรงงาน การคมนาคม เป็นต้น
- เหมาะอย่างยิ่งสำหรับบางสถานที่ ที่ในบางครั้งสัญญาณจะโดน block เช่น ในสวนสนุกเมื่อบางครั้ง เครื่องเล่นวิ่งผ่าน สัญญาณขาดหายบางช่วง แต่ถ้าเป็น mesh network การสื่อสารจะใช้ node อื่นๆ ข้างเคียงส่งข้อมูลแทนได้
- เครือข่ายสามารถปรับตัวเองได้ ถ้ามีโหนดใหม่เข้ามาในเครือข่ายจะสามารถเชื่อมต่อได้อัตโนมัติ
- เครือข่ายสามารถแก้ไข ปรับปรุงหรือซ่อมแซมตัวเองได้ กรณีที่มีโหนดบางตัวไม่สามารถทำงานได้ จะทำการค้นหาและเปลี่ยนเส้นทางไปยังโหนดข้างเคียงอื่นๆ ได้อย่างอัตโนมัติ
- มีติดตั้งได้ง่าย และขยายระบบได้ง่าย

2.2.1 โครงสร้างของเครือข่ายเมชไร้สาย

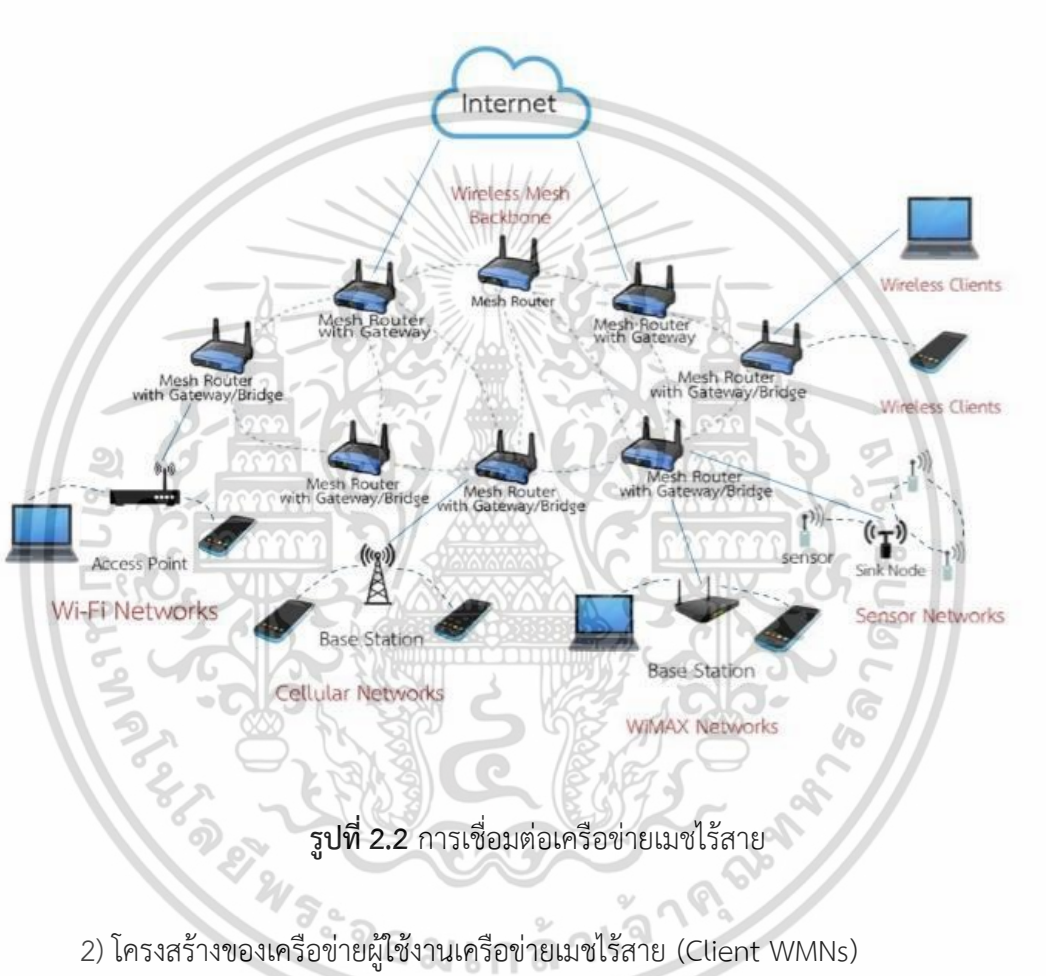
จากงานวิจัยของ Akyildiz, I.F. และ Xudong, W. ในปี 2005 ได้มีการนำเสนอโครงสร้างของ WMNs โดยได้แบ่งเป็น 3 ประเภทหลัก ดังนี้ [3]

1) โครงสร้างแบบโครงสร้างพื้นฐานของเครือข่ายเมชไร้สาย (Infrastructure/Backbone WMNs)

จากรูปที่ 2.2 แสดงรูปแบบโครงสร้างโดยเส้นปะแสดงการเชื่อมต่อแบบไร้สาย และเส้นทึบแสดงการเชื่อมต่อแบบมีสาย

mesh routers ทำหน้าที่เป็นตัวกลางให้ลูกข่ายเชื่อมต่อเข้ากับเครือข่ายและสามารถเข้าใช้บริการอินเทอร์เน็ตได้ ส่วนใหญ่จะอยู่ภายใต้มาตรฐาน IEEE 802.11 ซึ่ง mesh routers ที่อยู่ในเครือข่ายแบบนี้จะสามารถทำการเชื่อมต่อและรักษาเสถียรภาพของระบบได้ด้วยตัวมันเองและเครือข่ายของ mesh routers นั้นสามารถเชื่อมต่ออินเทอร์เน็ตด้วยวิธีที่เรียกว่า Infrastructure meshing คือ จะมีการจัดเตรียมเครือข่ายหลักที่ใช้เป็นจุดรวมในการส่ง

ข้อมูลของไคลเอนต์และเครือข่ายเมชไร้สายสามารถใช้งานร่วมกันผ่านทางฟังก์ชันการทำงานของ gateway/Bridge สำหรับไคลเอนต์ที่ใช้เทคโนโลยีสัญญาณวิทยุเหมือนกับ mesh routers ไคลเอนต์สามารถเชื่อมต่อและติดต่อสื่อสารกันโดยตรงกับ mesh routers ได้ แต่ถ้าไคลเอนต์ใช้เทคโนโลยีสัญญาณวิทยุที่แตกต่างกับ mesh routers ไคลเอนต์จะต้องเชื่อมต่อและติดต่อสื่อสารผ่านเสากระจายสัญญาณ (Base station) และเชื่อมต่อติดต่อจาก Ethernet ไปยัง mesh routers ดังแสดงในรูปที่ 2.2



รูปที่ 2.2 การเชื่อมต่อเครือข่ายเมชไร้สาย

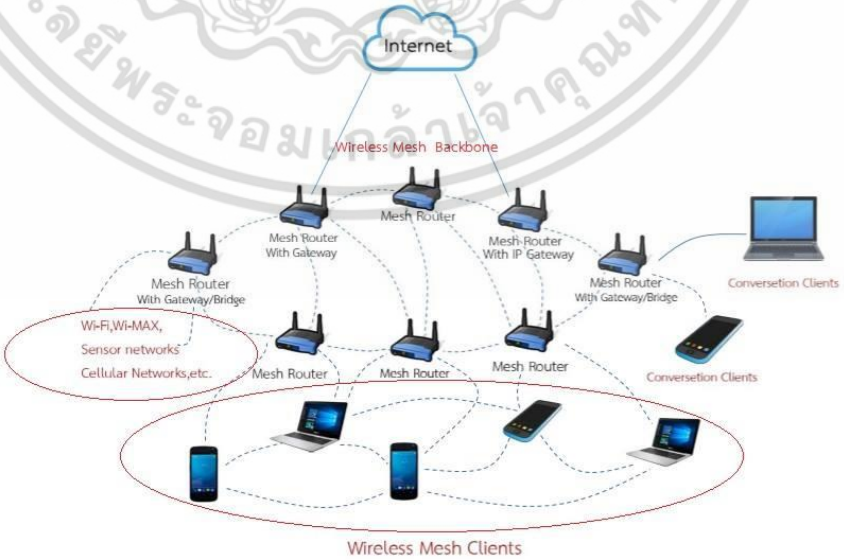
2) โครงสร้างของเครือข่ายผู้ใช้งานเครือข่ายเมชไร้สาย (Client WMNs)

เครือข่ายเมชแบบผู้ใช้งานนี้จะมีการเชื่อมต่อกันเป็นกลุ่มของ Mesh clients โดยโครงสร้างแบบนี้ Mesh clients จะทำหน้าที่จัดการเส้นทางและจัดการระบบให้กับกลุ่ม Mesh clients ดังนั้นในโครงสร้างนี้ไม่มีความจำเป็นต้องใช้ mesh routers โดยการติดต่อสื่อสารในโครงสร้างนี้การส่งข้อมูลจะส่งจากโหนดต้นทางไปยังโหนดปลายทาง โดยส่งผ่านโหนดไปเรื่อยๆจนถึงปลายทาง โดยปกติเครือข่ายแบบนี้จะใช้คลื่นวิทยุในการเชื่อมต่อระหว่าง Mesh clients เท่านั้น โครงสร้างแบบนี้ Mesh clients ต้องมีการเพิ่มฟังก์ชันการค้นหาเส้นทางและการจัดการระบบได้ในตัวเอง แสดงดังรูปที่ 2.3

รูปที่ 2.3 การเชื่อมต่อเครือข่ายผู้ใช้งานเครือข่ายเมชไร้สาย

3) โครงสร้างเครือข่ายเมชไร้สายแบบผสม

โครงข่ายเมชไร้สายประเภทผสม คือการรวมกันของประเภท Infrastructure และ client meshing ดังแสดงในรูปที่ 2.4 โดยไคลเอนต์ในเครือข่ายเมชไร้สายแบบผู้ใช้งานสามารถเข้ามาใช้งานในเครือข่ายโดยผ่าน Mesh routers ซึ่งดีเท่ากับวิธีการ meshing กันโดยตรงกับไคลเอนต์ในเครือข่ายไร้สาย ขณะที่ระบบโครงสร้างพื้นฐานทำหน้าที่จัดเตรียมการเชื่อมต่อไปยังเครือข่ายอื่นๆ โดยทางอินเตอร์เน็ต, Wi-Fi, WiMax, ระบบเครือข่ายมือถือและเครือข่ายเซ็นเซอร์ เป็นต้น



รูปที่ 2.4 การเชื่อมต่อเครือข่ายผู้ใช้งานเครือข่ายเมชไร้สาย

โครงสร้างเครือข่ายเมชไร้สายแบบผสมถูกเลือกให้นำมาใช้กับเครือข่ายเมชไร้สาย โดยลักษณะพิเศษของเครือข่ายเมชไร้สาย มีดังนี้

1) เครือข่ายเมชไร้สายสนับสนุนการทำงานในเครือข่ายแบบ ad hoc และ มีความสามารถในการปรับตั้งค่าด้วยตัวเองได้ ดูแลและตัดสินใจเองได้ แก้ไขหรือรักษาตัวเองได้ มีความแข็งแรงและมีความน่าเชื่อถือ

2) เครือข่ายเมชไร้สายเป็นเครือข่ายไร้สายแบบ multi-hop แต่โครงสร้างพื้นฐานของเครือข่ายไร้สายจะถูกจัดเตรียมโดย mesh routers

3) Mesh routers มีการสัญจรน้อยมากและทำให้ลดภาระในการค้นหาเส้นทางและการตั้งค่ามีน้อยลง ซึ่งลดความสำคัญและภาระของเครื่องลูกข่ายในเครือข่ายเมชไร้สายและโหนดอื่นๆ

4) Mesh routers สามารถทำงานร่วมได้กับเครือข่ายที่แตกต่างกันออกไปได้ รวมทั้งการเชื่อมต่อแบบมีสายและการเชื่อมต่อแบบไร้สาย ดังนั้นเครือข่ายหลายๆประเภทจึงสามารถเข้ามาใช้งานในเครือข่ายเมชไร้สายได้

5) การสูญเสียพลังงาน Mesh routers ไม่ค่อยมีข้อจำกัด แต่ Mesh clients ต้องการโพรโทคอลช่วยในการจัดการด้านพลังงาน

6) เครือข่ายเมชไร้สายไม่สามารถทำงานได้ด้วยตัวคนเดียวและต้องมีการเชื่อมต่อกันระหว่างเครือข่ายไร้สายอื่นๆ

ดังนั้น เครือข่ายเมชไร้สายได้เปลี่ยนความสามารถของเครือข่ายแบบ ad hoc โดยการเพิ่มอัลกอริทึมแบบใหม่และออกแบบหลักการที่จะสามารถนำไปใช้งานได้ในเครือข่ายเมชไร้สายได้จริง

2.2.2 โพรโทคอลที่ใช้ในการหาเส้นทางในเครือข่ายเมชไร้สาย

การค้นหาเส้นทาง (Routing) คือการหาเส้นทางที่จะส่งข้อมูลจากโหนดต้นทางไปยังโหนดปลายทางได้อย่างถูกต้อง

1. ส่วนประกอบของการค้นหาเส้นทาง (Routing Component)

1.1) อัลกอริทึมในการค้นหาเส้นทาง (Routing Algorithm) เป้าหมายในการออกแบบ มีดังนี้

1.1.1) คำนวณหาเส้นทางที่ดีที่สุด

1.1.2) มีประสิทธิภาพและใช้งานส่วน Overhead ได้อย่างเกิดประโยชน์

1.1.3) ทำงานได้ถูกต้องแม้จะเจอเหตุการณ์ผิดปกติ

1.1.4) มีการตอบสนองที่รวดเร็วเมื่อเครือข่ายมีการเปลี่ยนแปลง

1.1.5) ปรับตัวให้เข้ากับเครือข่ายที่มีความหลากหลายได้

1.2) ฐานข้อมูล (Database) คือ ตารางเส้นทาง (Routing table) ใช้เก็บเส้นทางในการส่งข้อมูลไปยังโหนดอื่นๆ

1.3) โพรโทคอลในการค้นหาเส้นทาง (Routing protocol) คือ โพรโทคอลที่ใช้ในการแลกเปลี่ยน
ข้อมูลระหว่างเราเตอร์เกี่ยวกับสถานะปัจจุบันของเครือข่าย มีหน้าที่ดังนี้

1.3.1) สร้างตารางเส้นทางทั้งหมด

1.3.2) ปรับปรุงตารางเส้นทางให้ทันสมัยอยู่เสมอ

1.3.3) คำนวณตัวเลือกที่ดีที่สุดสำหรับฮอป (Hop) ถัดไป

2. ประเภทของโพรโทคอลค้นหาเส้นทาง

โพรโทคอลในการค้นหาเส้นทางแบ่งออกเป็น 3 ประเภท คือ

1) Table-Driven หรือ Proactive Routing Protocol คือ โหนดแต่ละโหนดในเครือข่ายจะมีการสร้างตารางซึ่งมีข้อมูลของเส้นทางที่เหมาะสมที่สุดสำหรับการส่งข้อมูลไปยังโหนดปลายทางอื่นๆในเครือข่ายไว้ล่วงหน้า และมีการปรับปรุงข้อมูลในตารางตามช่วงเวลาที่กำหนด

<u>ข้อดี</u> มีการเตรียมเส้นทางไว้ล่วงหน้า ทำให้เวลาประมวลผลในแต่ละโหนดน้อย ทำให้ค่า end to end delay น้อย

<u>ข้อเสีย</u> ใช้ต้องเสียพื้นที่จัดเก็บตารางข้อมูลเส้นทางของโหนดทั้งหมดในเครือข่าย ซึ่งขนาดของตารางก็จะเพิ่มขึ้นตามจำนวนโหนดที่เพิ่มขึ้น

มีโพรโทคอลการหาเส้นทาง เช่น Optimized Link State Routing (OLSR) และ Destination- Sequenced Distance Vector (DSDV) เป็นต้น

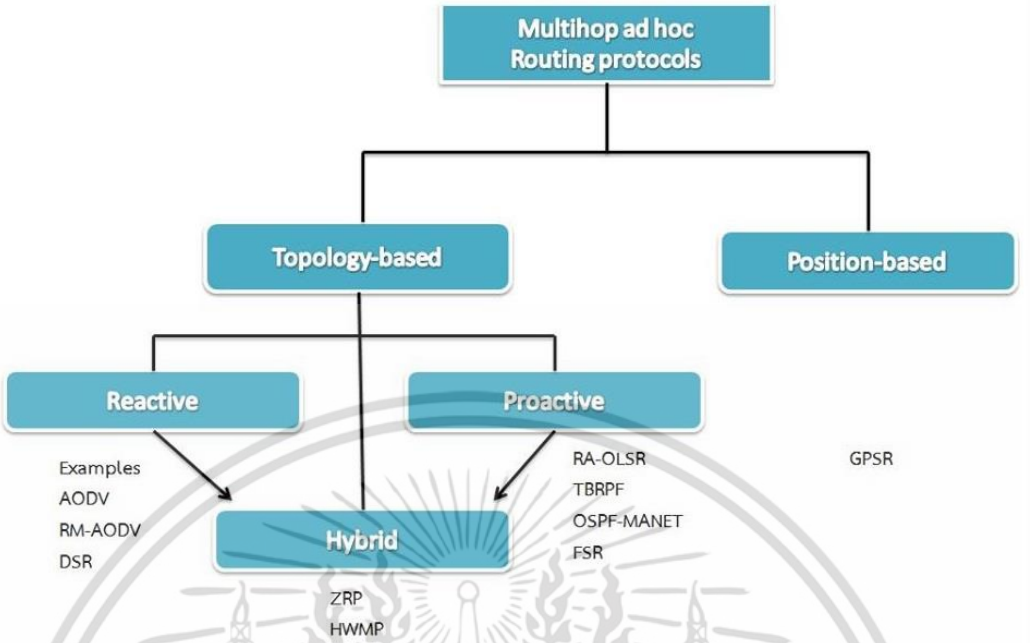
2) On-Demand หรือ Reactive routing protocols เป็นการค้นหาเส้นทางการรับส่งข้อมูลไปยังโหนดปลายทาง ก็ต่อเมื่อมีความต้องการจะส่งข้อมูลไปยังปลายทางนั้นๆ ไม่มีการจัดเตรียมตารางเก็บเส้นทางที่เหมาะสมที่สุดของโหนดปลายทางต่างๆไว้ล่วงหน้า

<u>ข้อดี</u> ค่า Overhead ลดลงเมื่อเทียบกับโพรโทคอลแบบ Table-Driven

<u>ข้อเสีย</u> เสียเวลาในการค้นหาเส้นทางถ้าโหนดปลายทางไม่เคยถูกถามมาก่อน

มีโพรโทคอลการหาเส้นทาง เช่น Ad hoc On-demand Distance Vector (AODV) และ Dynamic Source Routing (DSR) เป็นต้น และได้ถูกนำมาประยุกต์ใช้กับในเครือข่ายเมชไร้สาย

3) Hybrid routing protocols เป็นการรวมข้อดีของทั้ง Proactive ที่ใช้กับโหนดใกล้ๆและใช้เส้นทางที่ใช้บ่อย กับ Reactive ที่ใช้กับโหนดระยะทางไกลและใช้เส้นทางที่ไม่ค่อยใช้ เช่น Zone Routing Protocol (ZRP) และ Hybrid Wireless Mesh Protocol (HWMP) เป็นต้น



รูปที่ 2.5 ประเภทของโพรโทคอลค้นหาเส้นทาง

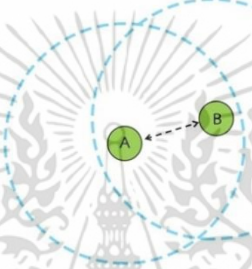
2.3 Ad hoc On-demand Distance Vector Routing (AODV)

Ad hoc On-demand Distance Vector Routing (AODV) [4,7,8] เป็นโพรโทคอลการค้นหาเส้นทางในเครือข่ายเคลื่อนที่ไร้สายแบบ ad hoc มีรูปแบบการสื่อสารที่ไม่จำเป็นต้องมีสถานีฐาน (Base station) ควบคุมการทำงานในส่วนของการสื่อสาร อุปกรณ์แต่ละตัวสามารถทำหน้าที่เป็นได้ทั้งโหนดรับ โหนดส่ง และโหนดเชื่อมต่อได้ โพรโทคอล AODV เหมาะสำหรับเครือข่ายที่มีขนาดใหญ่มีจำนวนโหนดมาก เครือข่ายที่โหนดมีการเคลื่อนที่และโหนดอยู่กับที่

2.3.1 ลักษณะของโพรโทคอล AODV

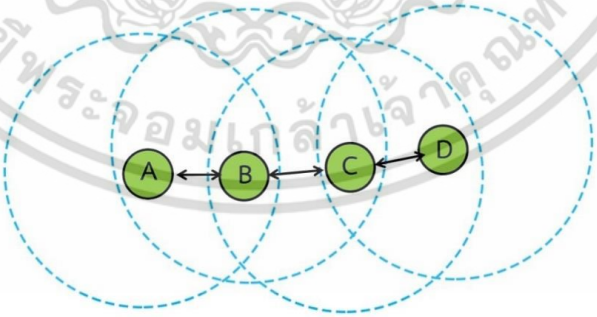
โพรโทคอล AODV เป็นโพรโทคอลการจัดเส้นทางในเครือข่ายไร้สายแบบ ad hoc ทำให้สถานีเชื่อมโยงสามารถติดต่อกันได้ โดยที่เส้นทางอาจมีการเชื่อมต่อหลายช่วง โพรโทคอล AODV จะมีการทำงานเป็นแบบ Reactive กล่าวคือเส้นทางจะถูกสร้างขึ้นเฉพาะเมื่อมีการร้อนขอเส้นทางเท่านั้น (On demand)และสถานีเชื่อมโยงไม่จำเป็นต้องทำการปรับปรุงข้อมูลเส้นทางไปยังสถานีเชื่อมโยงปลายทางที่ยังไม่ใช้งานในขณะนั้น และในขณะการสื่อสารดำเนินอยู่ โดยเส้นทางยังสามารถทำงานได้ ข้อเด่นอย่างหนึ่งของโพรโทคอลนี้คือการค้นหาเส้นทางและเลือกใช้เส้นทางของคู่สถานีเชื่อมโยงต้นทางและปลายทางที่มีอยู่เพื่อให้การส่งข้อมูลนั้นเป็นไปอย่างถูกต้อง การทำงานของโพรโทคอลนี้ คือมีการส่งข้อความควบคุม (Control Messages) เป็นช่วงๆเพื่อใช้ในการกำหนดเส้นทางหรือปรับปรุงข้อมูลเส้นทาง

โพรโทคอล AODV เป็นโพรโทคอลที่เป็นแบบ Routing message ระหว่างเครื่องคอมพิวเตอร์ที่เคลื่อนย้ายได้ (Mobile computer) ในการส่งข้อมูลผ่านไปยังโหนดเพื่อนบ้าน (Neighbor node) เพื่อไปยังโหนดที่ปลายทาง ไม่สามารถติดต่อได้โดยตรง ในระหว่างทางที่ข้อมูลถูกส่งผ่านไปโพรโทคอล AODV ก็จะทำการค้นหาเส้นทางไป โดยจะมั่นใจได้ว่าจะไม่เกิดการวนลูป และพยายามหาเส้นทางที่สั้นที่สุดที่จะเป็นไปได้ อีกทั้งโพรโทคอล AODV ยังสามารถควบคุมการเปลี่ยนแปลงของเส้นทาง (Route) และสามารถสร้างเส้นทางใหม่หากเกิดข้อผิดพลาดได้อีกด้วย



รูปที่ 2.6 ขอบเขตของการสื่อสารแบบ single-hop

จากรูปที่ 2.6 เป็นตัวอย่างการตั้งค่าของโหนด A วงกลมแสดงขอบเขตการติดต่อสื่อสารของโหนด A และเนื่องจากมีการกำหนดขอบเขตการเชื่อมต่อทำให้แต่ละโหนดสามารถติดต่อกับโหนดที่อยู่ถัดจากตัวเองเท่านั้น



รูปที่ 2.7 ขอบเขตของการสื่อสารแบบ multi-hop

จากรูปที่ 2.7 จะเห็นได้ว่าโหนด A สามารถติดต่อกับโหนด B ได้เท่านั้น เพราะโหนด C และโหนด D อยู่นอกขอบเขตการติดต่อของ A จึงไม่สามารถติดต่อกันได้ สำหรับโหนดที่เราสามารถติดต่อ

ได้โดยตรงนั้นเราจะเรียกว่า โหนดเพื่อนบ้าน (Neighbor node) โดยโหนดจะเก็บข้อมูลของโหนดเพื่อนบ้านเมื่อได้รับ HELLO message ที่แต่ละโหนดจะทำการ Broadcast ออกมาตามช่วงเวลาที่กำหนดไว้

เมื่อมีโหนดใดๆต้องการส่งข้อมูลไปยังโหนดอื่นๆที่ไม่ใช่โหนดเพื่อนบ้านจะทำการ Broadcast Route Request (RREQ) ซึ่งใน RREQ นี้จะประกอบไปด้วยฟิลด์ (Field) ของข้อมูลหลายตัว เช่น ต้นทาง (Source), ปลายทาง (Destination) และหมายเลขลำดับของ RREQ (Route Request ID) ซึ่งเป็นหมายเลขที่ไม่ซ้ำกัน (Unique ID)

2.3.2 ตารางเส้นทาง (Routing table)

ตารางเส้นทาง มีหน้าที่ ในการเก็บรายละเอียดเกี่ยวกับเส้นทางในการสื่อสารโดยมีโครงสร้างสำหรับเก็บข้อมูลเพื่อใช้ในการตรวจสอบข้อความ RREQ และ RREP ดังตารางที่ 2.1

ตารางที่ 2.1 ความหมายของส่วนของข้อมูลในตารางเก็บค่าเส้นทาง

ส่วนของข้อมูล	ความหมาย
Destination IP Address	หมายเลขของโหนดปลายทางที่ใช้ในการส่งข้อมูล
Destination Sequence Number	หมายเลขลำดับของโหนดปลายทาง
Valid Destination Sequence Number flag	ตัวแปรที่ใช้บอกว่าข้อมูลเส้นทางนั้นๆมีหมายเลขลำดับของโหนดปลายทางถูกต้องหรือไม่
Hop Count	จำนวนโหนดที่จะต้องการส่งข้อมูลไปจนถึงโหนดปลายทาง
Next Hop	หมายเลขโหนดถัดไปในเส้นทางสื่อสาร
List of Precursors	รายการของหมายเลขโหนดข้างเคียงที่ถูกใช้ในการส่งข้อความ RREP ต่อเพื่อใช้ในการบำรุงรักษา
Life time	เวลาที่โหนดใช้ในการพิจารณาว่าข้อมูลยังใช้งานได้

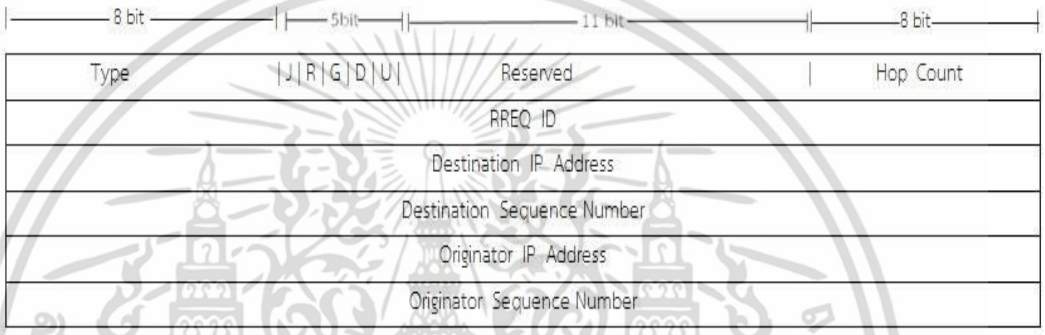
ในตารางเส้นทางของโพรโทคอล AODV แต่ละข้อมูลจะอัพเดทได้ดังกรณีต่อไปนี้

1) เมื่อค่าหมายเลขลำดับปลายทางมีค่าสูงกว่าค่าในตารางเส้นทาง
2) หากค่าหมายเลขลำดับปลายทางมีค่าเท่ากัน ต้องพิจารณาที่ค่า Hop count หากค่า Hop count มีค่าน้อยกว่าก็จะทำการอัพเดทข้อมูลนั้น ซึ่งในแต่ละบรรทัดของตารางเส้นทางจะมีอายุของตัวเองซึ่งในระยะที่เวลายังไม่หมด ก็จะสามารถส่งข้อมูลผ่านเส้นทางนี้ไปได้ตลอด แต่หากหมดอายุก็จะไม่สามารถใช้ได้ และต้องหาเส้นทางใหม่ในการส่งข้อมูลอีกครั้ง

2.3.3 รูปแบบข้อความ (Message format)

คือ ข้อความที่ใช้ในการจัดการเส้นทางที่โหนดต้องการ แบ่งข้อความได้ 3 ประเภทได้แก่

1) RREQ (Route Request message) คือ ข้อความควบคุมที่ใช้สำหรับการสอบถามเส้นทางไปยังปลายทางในกรณีที่ไม่มีเส้นทางข้อมูลนั้นอยู่ในตารางเส้นทาง โดยโหนดจะเริ่มส่งข้อความ RREQ แบบแพร่กระจายไปยังโหนดข้างเคียง และโหนดข้างเคียงจะทำการส่งต่อข้อมูลไปเรื่อยๆไปจนถึงโหนดปลายทาง ซึ่งข้อความ RREQ มีขนาด 24 ไบต์ และมีส่วนประกอบของข้อความดังรูปที่ 2.8

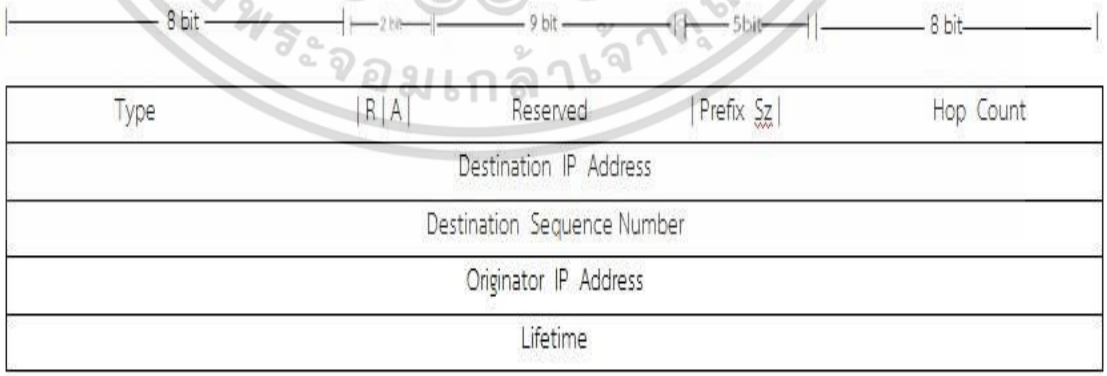


รูปที่ 2.8 ตัวอย่างของข้อความ RREQ

ตารางที่ 2.2 ความหมายของส่วนของข้อมูลในข้อความ RREQ

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความเป็น 1
Hop Count	จำนวนโหนดที่ข้อความ RREQ ผ่านมาจากต้นทางจนถึงโหนดที่ RREQ มาถึง
RREQ ID	หมายเลขลำดับของข้อความ แต่ละชุดที่ทำการ Broadcast
Destination IP Address	หมายเลขไอพีที่อยู่ของโหนดปลายทางที่โหนดต้นทางต้องการส่งข้อมูล
Destination Sequence Number	หมายเลขลำดับล่าสุดจากโหนดต้นทาง
Originator IP Address	หมายเลขไอพีของโหนดต้นทางที่สร้างและส่งข้อความ RREQ เพื่อหาเส้นทางไปยังโหนดปลายทาง
Originator Sequence Number	หมายเลขลำดับ ณ ปัจจุบัน

2) RREP (Route Reply message) คือ message ควบคุมที่ใช้สำหรับตอบเส้นทางที่ถูกร้องขอกลับไปยังโหนดที่ร้องขอ โดยจะใช้เส้นทางย้อนกลับที่ถูกสร้างขึ้นระหว่างขั้นตอนการส่งต่อข้อความ RREQ และเมื่อโหนดถัดไปได้รับข้อความ RREP ก็จะมีกระบวนการสร้างเส้นทางสำหรับการใช้ส่งข้อมูลต่อไป โดยมีส่วนประกอบข้อความดังรูปที่ 2.9

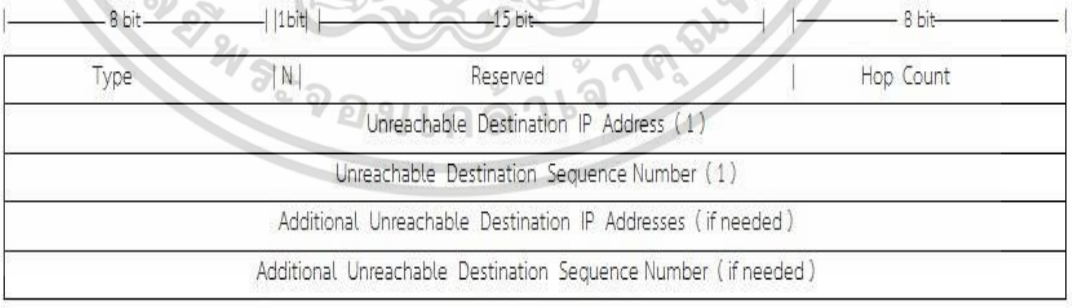


รูปที่ 2.9 ตัวอย่างของข้อความ RREP

ตารางที่ 2.3 ความหมายของส่วนของข้อมูลในข้อความ RREP

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความเป็น 2
Hop Count	จำนวนของ Hop จากต้นทางถึงปลายทาง
Destination IP Address	หมายเลขไอพีของโหนดปลายทางในเส้นทางการส่งข้อมูล
Destination Sequence Number	หมายเลขลำดับในเส้นทาง
Originator IP Address	หมายเลขไอพีของโหนดที่สร้างข้อความ RREQ
Lifetime	เวลาในหน่วยมิลลิวินาที ที่โหนดจะต้องได้รับข้อความ RREP เพื่อยืนยันเส้นทางที่ใช้ในการสื่อสาร

3) RERR (Route Error message) คือ ข้อความควบคุมที่ใช้สำหรับแจ้งว่าปลายทางนั้นหรือโหนดนั้นๆ ไม่สามารถไปถึงได้ เมื่อโหนดได้รับ RERR แล้วก็จะทำการลบปลายทางนั้นๆออกจากตารางเส้นทาง หรือหากมีฮอปถัดไป (Next hop) เป็นโหนดนั้นๆก็จะทำการลบด้วยเช่นกัน ข้อความ RREP มีขนาด 20 ไบต์ โดยมีส่วนประกอบข้อความดังรูปที่ 2.10



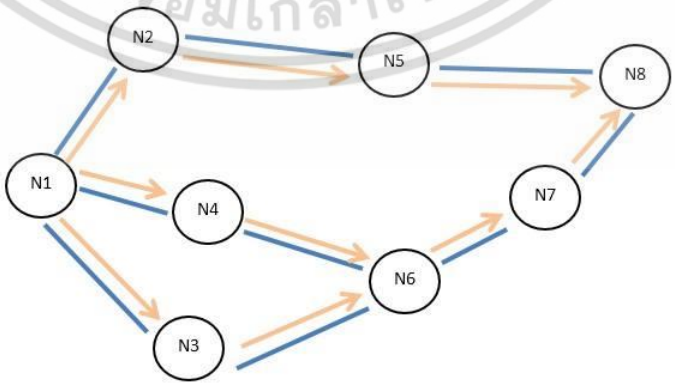
รูปที่ 2.10 ตัวอย่างของข้อความ RERR

ตารางที่ 2.4 ความหมายของส่วนของข้อมูลในข้อความ RRER

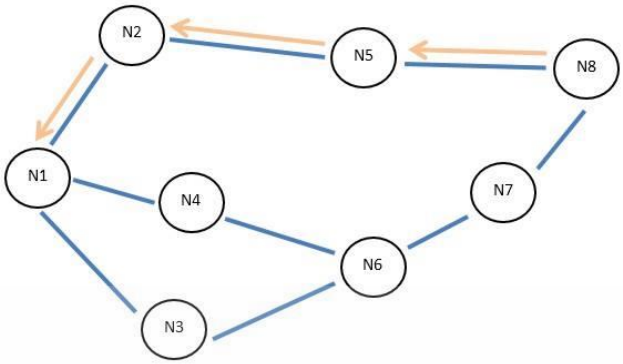
ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความเป็น 3
DestCount	จำนวนของโหนดปลายทางที่ไม่สามารถติดต่อได้
Unreachable Destination IP Address	หมายเลขไอพีที่อยู่ของโหนดปลายทางที่ไม่สามารถติดต่อได้
Destination Sequence Number	หมายเลขลำดับในเส้นทาง
Originator IP Address	หมายเลขไอพีของโหนดที่สร้างข้อความ RREQ
Unreachable Destination Sequence Number	หมายเลขลำดับในตารางเก็บค่าเส้นทางสำหรับปลายทางที่ระบุไว้ในโหนดปลายทางที่ไม่สามารถติดต่อได้

2.3.4 ขั้นตอนการทำงานของโพรโทคอล AODV

การทำงานของโพรโทคอล AODV เป็นดังนี้คือ เมื่อมีโหนดต้นทางที่จะส่งข้อมูลไปยังโหนดปลายทาง โหนดต้นทางจะทำการส่ง RREQ ไปยังโหนดเพื่อนบ้าน และโหนดที่ได้รับก็จะทำการส่งต่อไปยังโหนดที่ใกล้เคียงต่อไป จนถึงโหนดที่ต้นทางต้องการจะติดต่อด้วยหรือโหนดปลายทางนั้นเอง และเมื่อโหนดปลายทางได้รับ RREQ ตัวแรกที่มาถึงที่โหนดปลายทาง โหนดปลายทางก็จะทำการส่ง RREP กลับไปยังโหนดต้นทางที่ทำการส่ง RREQ มาให้โดยจะส่งกลับไปในเส้นทางที่ RREQ ตัวแรกมาถึง เพราะถือว่าใช้เวลาน้อยที่สุดในการส่ง RREQ มาจากต้นทาง ดังรูปที่ 2.11

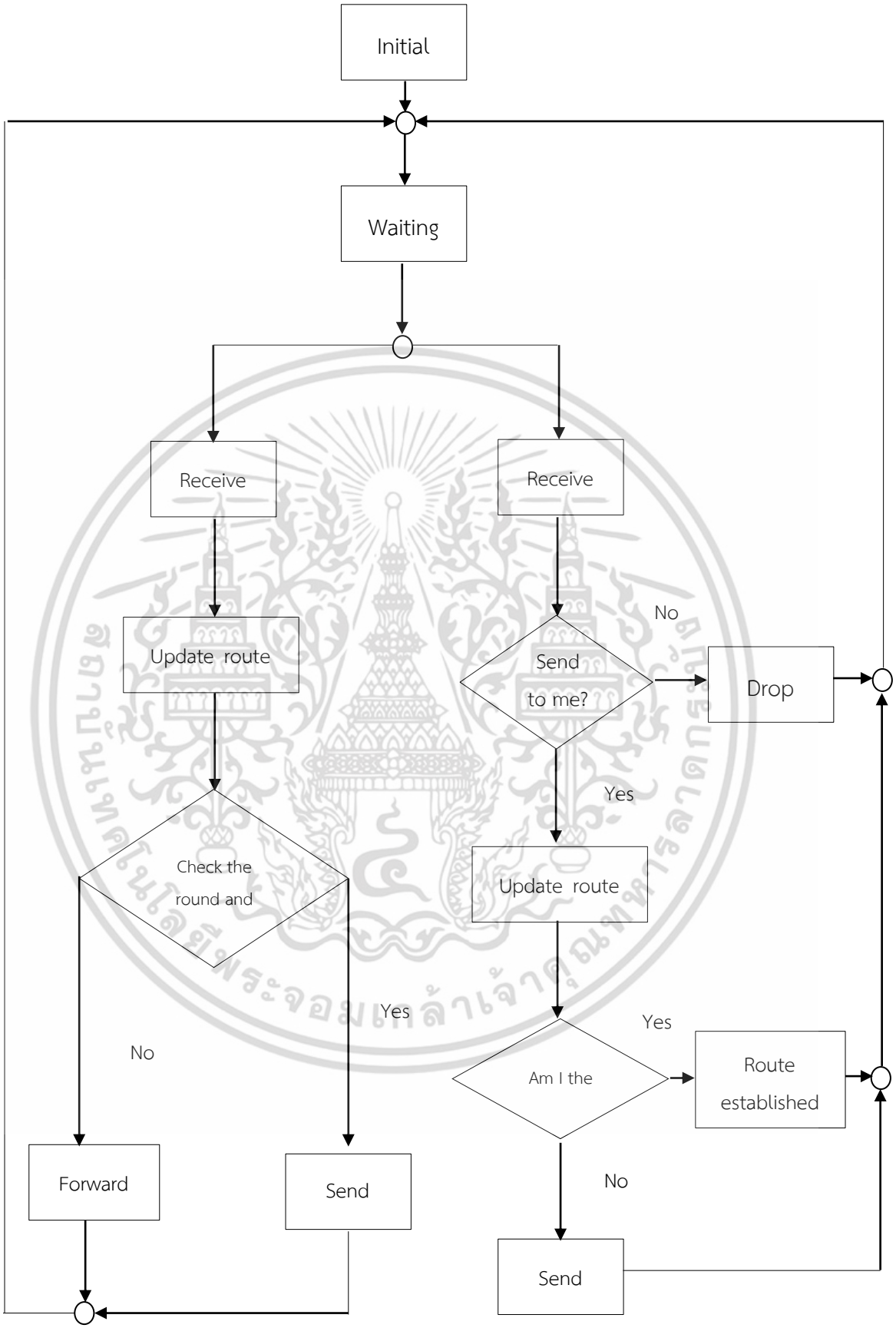


รูปที่ 2.11 การส่ง RREQ



รูปที่ 2.12 เส้นทางการตอบกลับของข้อความ RREP

จากรูปที่ 2.12 เป็นการส่ง RREQ จาก โหนดต้นทาง (N1) ไปยังโหนดปลายทาง (N8) โดยทุกโหนดจะมีตารางบันทึกไว้และจะอัพเดตค่าเฉพาะช่วงเวลาที่จะทำการส่งข้อมูลหรือมีการเคลื่อนที่ของโหนด



รูปที่ 2.13 ตาราง Flowchart การทำงานพื้นฐานของโพรโทคอล AODV

จากรูปที่ 2.13 หลักการทำงานของโพรโทคอล AODV โหนดในระบบจะทำการรอรับข้อความการจัดการเส้นทาง ถ้าหากเป็นข้อความ RREQ โหนดจะเข้าสู่กระบวนการปรับปรุงตารางเส้นทางและตรวจสอบข้อมูลเส้นทางของตัวเองว่ามีข้อมูลที่ต้องการหรือไม่ หากไม่มีจะส่งข้อความ RREQ แต่ถ้าหากมีข้อมูลดังกล่าวนั้นจะตอบกลับด้วยข้อความ RREP และในกรณีโหนดได้รับข้อความ RREP โหนดจะทำการพิจารณาว่าข้อความ RREP ดังกล่าวมีความต้องการส่งมายังตนเองหรือไม่ หากข้อความดังกล่าวไม่ได้ต้องการส่งมายังตนเอง โหนดจะไม่พิจารณาข้อความนั้น แต่หากข้อความดังกล่าวส่งมายังตนเองโหนดจะเข้าสู่กระบวนการปรับปรุงตารางเส้นทางและส่งต่อข้อความ RREP กลับไปยังโหนดที่มีความต้องการเส้นทางนั้นๆต่อไป โดยหลักการทำงานแบบละเอียดของโพรโทคอล AODV สามารถ แบ่งได้เป็น 11 ขั้นตอนดังนี้

1) หมายเลขลำดับ (Maintaining Sequence Numbers)

แต่ละข้อมูลของตารางเส้นทางต้องมีค่าหนึ่งที่บอกหมายเลขลำดับของปลายทางนั้นๆ เพื่อปรับปรุงให้ ตารางเส้นทางให้ทันสมัยอยู่เสมอ ตัวเลขนั้นเรียกว่า หมายเลขลำดับของโหนดปลายทาง (Destination Sequence Number) ซึ่งจะถูกทำการปรับปรุงก็ต่อเมื่อโหนดได้รับหมายเลขลำดับใหม่จากข้อความ RREQ ข้อความ RREP หรือ ข้อความ RRER ที่มีความเกี่ยวข้องกับโหนดปลายทาง โดยในโพรโทคอล AODV ยังใช้หมายเลขลำดับปลายทางเพื่อทำให้ไม่เกิดลูปของการค้นหาเส้นทางทั้งหมดด้วย โหนดแต่ละโหนดจะมีค่าหมายเลขลำดับเป็นของตัวเองและจะเพิ่มค่าดังกล่าวดังกรณีต่อไปนี้

1) ก่อนที่โหนดต้นทางจะค้นหาเส้นทาง โหนดจะเพิ่มค่าหมายเลขลำดับของตัวเองเพื่อหลีกเลี่ยงการซ้ำซ้อนของข้อมูลก่อนหน้าที่เคยถูกส่งและใช้งานในเส้นทางย้อนกลับ
2) ก่อนที่ปลายทางที่ได้รับ RREQ จะทำการส่ง RREP กลับ โหนดนั้นจะต้องเพิ่มหมายเลขลำดับของตัวเองเป็นค่าที่สูงที่สุด ระหว่างค่าหมายเลขลำดับของตัวเองและค่าหมายเลขลำดับปลายทางในข้อความ RREQ และเพิ่มค่าไปอีกหนึ่ง

เมื่อค่าหมายเลขลำดับเพิ่มจนถึงขีดจำกัดของตัวแปรที่เก็บแล้วก็ให้วนมาที่ค่า 0 ใหม่ ในกรณีที่ได้รับ RREP มาแล้วการที่จะเปลี่ยนค่าหมายเลขลำดับปลายทางของปลายทางนั้นๆ ได้ ก็ต่อเมื่อมีค่าหมายเลขลำดับปลายทางสูงกว่าเท่านั้น

สำหรับค่าของหมายเลขลำดับในตารางเส้นทางของโหนดจะมีการเปลี่ยนแปลงได้ 3 กรณีดังต่อไปนี้

1) เมื่อตนเองเป็นโหนดปลายทางและมีเส้นทางใหม่ไปยังตนเอง
2) โหนดดังกล่าวได้รับข้อมูลการจัดการเส้นทางที่มีข้อมูลที่ใหม่กว่าสำหรับปลายทางนั้นๆ
3) เส้นทางที่ใช้ในการส่งข้อมูลไปยังปลายทางหมดอายุหรือเกิดความเสียหาย

2) ข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายและรายการของหมายเลขโหนดที่ถูกใช้ในการส่งข้อความ RREP (Precursor lists)

เมื่อโหนดได้รับข้อความควบคุม (control packet) จากโหนดข้างเคียง หรือมีการสร้างหรือปรับปรุงเส้นทางเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทาง โหนดดังกล่าวจะทำการตรวจสอบว่าตารางเส้นทางของตนเองมีข้อมูลเส้นทางเพื่อใช้ในการส่งไปยังโหนดปลายทางหรือไม่ หากไม่มีข้อมูลเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทาง โหนดดังกล่าวจะทำการสร้างข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายขึ้น โดยที่หมายเลขลำดับจะถูกกำหนดจากข้อมูลที่มีอยู่ในข้อความที่ได้รับมา และข้อมูลเส้นทางจะปรับปรุงให้มีหมายเลขลำดับใหม่ด้วยสาเหตุต่อไปนี้

1) เมื่อได้รับข้อความควบคุมที่มีค่าหมายเลขลำดับสูงกว่าในหมายเลขลำดับของโหนดปลายทางในตารางเส้นทางนั้นๆ
2) เมื่อได้รับข้อความควบคุมที่มีค่าหมายเลขที่เท่ากับหมายเลขลำดับของโหนดปลายทางในตารางเส้นทางแต่มีค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลน้อยกว่า
3) เมื่อโหนดไม่ทราบหมายเลขลำดับที่ใช้ในการส่งข้อมูลไปยังโหนดปลายทาง

สำหรับเวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้ (Life time) ที่จัดเก็บอยู่ในตารางเส้นทางจะมีการกำหนดค่าจากข้อความควบคุมหรือมีการตั้งค่าเริ่มต้นจากตัวแปร ACTIVE_ROUTE_TOMEOUT

สำหรับทุกเส้นทางที่โหนดพิจารณาให้เป็นเส้นทางที่ยังคงทำงานได้ (Valid route) โหนดดังกล่าวจะต้องเก็บรักษารายการของหมายเลขโหนดที่ถูกใช้งานในการส่งข้อความ RREP โดยค่ารายการดังกล่าวจะมีการจัดเก็บโหนดข้างเคียง (Neighboring nodes) ที่ทำการสร้างหรือส่งต่อข้อความ RREP

3) การสร้างข้อความ RREQ

เมื่อโหนดต้นทางต้องการส่งข้อความไปยังโหนดปลายทาง และโหนดต้นทางดังกล่าวไม่มีข้อมูลเส้นทางเพื่อใช้ในการส่ง โหนดดังกล่าวจะสร้างข้อความ RREQ และส่งต่อไปยังโหนดข้างเคียง ในส่วนของหมายเลขลำดับของโหนดปลายทางในข้อความ RREQ จะเป็นค่าสุดท้ายที่โหนดดังกล่าวเคยมีข้อมูลอยู่ แต่ถ้าไม่มีค่าตัวแปร unknown sequence number ในข้อความ RREQ ก็จะถูกใช้งาน ส่วนหมายเลขลำดับของโหนดต้นทาง (Originator sequence number) ในข้อความ RREQ จะเป็นค่าเริ่มต้นหมายเลขลำดับโหนดนั้นๆซึ่งจะถูกเพิ่มครั้งละ 1 เหมือนกับค่าหมายเลขลำดับของ RREQ (RREQ ID) ซึ่งทั้งระบบจะมีการเก็บรักษาหมายเลขลำดับ RREQ ร่วมกัน สำหรับค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลจะถูกตั้งค่าเริ่มต้นให้มีค่าเป็น 0 ก่อนที่จะกระจากข้อความ RREQ

โหนดต้นทางจะเก็บค่าชั่วคราวของข้อมูลหมายเลขลำดับ RREQ และหมายเลขของโหนดต้นทางข้อความ RREQ ในช่วงระยะเวลา PATH_DISCOVERY_TIME และเมื่อโหนดดังกล่าวได้รับ

ข้อความ RREQ จากโหนดใกล้เคียงซึ่งมีค่าหมายเลขลำดับ RREQ และหมายเลขของโหนดต้นทางที่ตนเองเป็นผู้ส่งออกไป โหนดจะไม่พิจารณาข้อมูลดังกล่าว นอกจากนี้โหนดต้นทางจะต้องไม่สร้างข้อความ RREQ มากกว่าค่าที่กำหนดไว้ใน RREQ_RATELIMIT ต่อวินาที ซึ่งหลังจากส่งข้อความ RREQ แบบแพร่กระจายออกไป โหนดจะรอข้อความ RREP ถ้าไม่ได้รับข้อมูลในช่วงเวลาที่กำหนดไว้ในตัวแปร NET_TRAVERSEL_TIME มีหน่วยเป็นมิลลิวินาที โหนดดังกล่าวอาจพยายามค้นหาเส้นทางใหม่อีกครั้ง โดยการเพิ่มค่าหมายเลขลำดับ RREQ และจำนวนครั้งในความพยายามดังกล่าวมีค่ามากที่สุดต้องไม่เกินค่าของ RREQ_RETRIES และเพื่อเป็นการลดความหนาแน่นในระบบ โหนดต้นทางที่ค้นหาเส้นทางใหม่ใช้หลักการทำงานแบบ Binary exponential back off หมายถึงเมื่อโหนดส่ง RREQ แบบแพร่กระจายและรอข้อความ RREP ในช่วงเวลา NET_TRAVERSEL_TIME หากไม่ได้รับข้อความ RREP ที่ต้องการ โหนดต้นทางจะทำการส่งข้อความ RREQ ใหม่โดยการคำนวณเวลาในการรอสำหรับข้อความ RREP ครั้งต่อไปจะมีค่าเป็น 2*NET_TRAVERSEL_TIME มิลลิวินาที ถ้าหากยังคงไม่ได้รับข้อความ RREP ในช่วงเวลาดังกล่าว ข้อความ RREQ จะถูกส่งแบบแพร่กระจายใหม่อีกครั้ง และทุกๆครั้งที่มีความพยายาม ระยะเวลาในการรอข้อความ RREP จะมีค่าเพิ่มขึ้นเป็น 2 เท่าของระยะเวลาในการรอล่าสุด

4) การควบคุมการส่ง Route Request message

เพื่อหลักเลี่ยงการส่งข้อความ RREQ แบบกระจายไปทั่วทั้งระบบโดยไม่จำเป็น โหนดต้นทางจะใช้เทคนิคการค้นหาแบบ Expanding ring search โดยโหนดต้นทางจะเริ่มทำการกำหนดค่าเริ่มต้นของ Time to life (TTL) ให้มีค่าเท่ากับ TTL_START ในข้อความ RREQ โดยค่า TTL หมายถึงค่าจำนวนแพ็กเกจที่สามารถถูกส่งต่อได้ และตั้งค่าระยะเวลาที่กำหนด (Timeout) สำหรับการรอรับข้อความ RREP ให้มีค่าเป็น RING_TRAVERSAL_TIME มิลลิวินาที และถ้าโหนดไม่ได้รับข้อความ RREP ภายในเวลาที่กำหนด โหนดต้นทางจะทำการส่งข้อความ RREQ แบบแพร่กระจายอีกครั้ง และจะเพิ่มค่า TTL ภายในข้อความ RREQ โดยค่าที่เพิ่มขึ้นนั้นมีค่าเท่ากับ TTL_INCREMENT จนกระทั่งค่าของ TTL มีค่าเท่ากับ TTL_THRESHPOLD หลังจากนั้นจะให้ค่า TTL เท่ากับค่า NET_DIAMETER แทน

ในส่วนของจำนวนโหนดที่ใช้ในการส่งข้อมูลที่ถูกบันทึกอยู่ในตารางเส้นทางว่าเป็นเส้นทางที่ไม่สามารถใช้งานได้ (Invalid Rounting table) นั้นหมายถึงค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลล่าสุดที่โหนดดังกล่าวรู้เกี่ยวกับข้อมูลเส้นทางเพื่อใช้ไปยังโหนดปลายทาง และเมื่อมีความต้องการเส้นทางใหม่โดยที่เป็นโหนดปลายทางเดิม ค่าของ TTL ที่ถูกใช้ในการส่งข้อความ RREQ แบบแพร่กระจายจะถูกตั้งค่าเริ่มต้นให้มีค่าเป็นผลรวมของค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลที่มีอยู่ในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย TTL_INCREMENT ในส่วนของตัวแปร Expired rounting table entry ไม่ควรถูกลบก่อนผลบวกของเวลาปัจจุบันกับค่าตัวแปร DELETE_PERIOD และข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย (Rounting table entry) ที่มีการรอข้อความ RREP ไม่ควรถูกลบก่อนเวลาปัจจุบันบวกกับค่า 2*NET_TRAVERSEL_TIME

5) การพิจารณาข้อมูลที่ได้รับและการส่งต่อข้อความ RREQ

เมื่อโหนดได้รับข้อความ RREQ ขั้นแรกโหนดดังกล่าวจะสร้างหรือปรับปรุงค่าของโหนดก่อนหน้า โดยไม่สนใจหมายเลขลำดับ หลังจากนั้นโหนดดังกล่าวจะทำการตรวจสอบว่าเคยได้รับข้อความ RREQ ที่มีค่าหมายเลขของโหนดต้นทางและหมายเลขลำดับ RREQ โหนดดังกล่าวจะไม่กระทำกระบวนการใดๆ แต่หากยังไม่เคยได้รับข้อความ RREQ โหนดจะเริ่มกระบวนการต่อไปนี้ ขั้นแรกโหนดจะทำการเพิ่มค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลที่อยู่ภายในข้อความ RREQ ขึ้นหนึ่งค่า เพื่อใช้ในการส่งต่อไปยังโหนดข้างเคียง หลังจากนั้นโหนดจะทำการค้นหาเส้นทางย้อนกลับ เพื่อส่งค่าไปยังโหนดต้นทางว่ามีหรือไม่ ถ้าหากไม่มีเส้นทางดังกล่าวแล้ว โหนดจะทำการสร้างเส้นทางย้อนกลับ โดยใช้หมายเลขลำดับของโหนดต้นทางที่ได้จากข้อความ RREQ ในตารางเส้นทาง เมื่อเส้นทางย้อนกลับถูกสร้างหรือปรับปรุงแล้วจะทำการเปรียบเทียบค่าหมายเลขลำดับของโหนดต้นทางจากข้อความ RREQ กับหมายเลขลำดับของโหนดปลายทางภายในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย ถ้าหากหมายเลขลำดับของโหนดต้นทางมีค่ามากกว่าก็จะทำการคัดลอกค่าเก็บไว้ในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายแทนที่ค่าเดิมและจะทำการตั้งค่าตัวแปรหมายเลขลำดับถูกต้อง (Valid Sequence number) ในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย ให้มีค่าเป็น TRUE จากนั้นจะทำการตั้งค่าโหนดถัดไปในตารางเส้นทางให้มีค่าเป็นหมายเลขของโหนดที่ทำการส่งข้อความ RREQ ที่ได้รับมา และทำการคัดลอกค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลที่ได้จากข้อความ RREQ

เมื่อใดก็ตามที่โหนดได้รับข้อความ RREQ และทำการปรับปรุงค่าในตารางเส้นทางแล้วพบว่าโหนดดังกล่าวไม่สามารถส่งข้อความ RREP กลับไปได้และหากค่า TTL ในข้อความ RREQ มีค่ามากกว่า 1 โหนดดังกล่าวจะทำการลดค่า TTL ลง 1 และเพิ่มค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลขึ้น 1 จากนั้นจะเข้าสู่กระบวนการส่งข้อความ RREQ แบบแพร่กระจายไปยังโหนดข้างเคียง

6) การสร้างข้อความ RREP

กรณีที่โหนดจะทำการส่งข้อความ RREP กลับมีขั้นตอนดังต่อไปนี้

- โหนดที่ได้รับข้อความ RREQ ดังกล่าวเป็นโหมดปลายทาง ถ้าหากว่าหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ นั้นมีค่ามากกว่าค่าหมายเลขลำดับของตนเอง โหนดจะทำการเพิ่มค่าของหมายเลขลำดับของตนเองขึ้นหนึ่งแต่ถ้าหากว่าค่าหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ ไม่มากกว่าค่าหมายเลขลำดับตนเอง โหนดดังกล่าวจะไม่กระทำกระบวนการเพิ่มค่าหมายเลขลำดับของตนเอง หลังจากนั้นจะทำการตั้งค่าเริ่มต้นให้ค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลมีค่าเป็น 0 และตั้งค่าให้หมายเลขลำดับของโหนดปลายทางภายในข้อความ RREP มีค่าเท่ากับค่าหมายเลขลำดับของตนเอง และทำการส่งข้อความ RREP กลับผ่านทางเส้นทางย้อนกลับ

- โหนดดังกล่าวมีข้อมูลเส้นทางที่ใช้งานได้ (Active route) เพื่อส่งข้อมูลไปยังโหนดปลายทาง โดยที่หมายเลขลำดับของโหนดปลายทางในตารางเส้นทางนั้นมีค่ามากกว่าหรือเท่ากับหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ ที่ได้รับมาและจะทำการคัดลอกค่าหมายเลขลำดับดังกล่าวไปยังหมายเลขลำดับของโหนดปลายทางลงในข้อความ RREP ก่อนทำการส่งกลับผ่านทางเส้นทางย้อนกลับ

ในการสร้างข้อความ RREP โหนดจะทำการคัดลอกค่าของหมายเลขโหนดของโหนดปลายทางและหมายเลขลำดับของโหนดต้นทางที่ได้จากข้อความ RREQ ไปยังข้อความ RREP และการส่งข้อความ RREP จะเป็นการส่งแบบปลายทางเดียวไปยังโหนดตัวถัดไป โดยมีปลายทางเป็นโหนดต้นทางที่ทำการสร้างข้อความ RREQ ขึ้น ส่วนค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลจะถูกเพิ่มขึ้นครั้งละหนึ่ง เมื่อมีการส่งข้อมูลผ่านแต่ละโหนดเช่นเดียวกับการส่งข้อความ RREQ ดังนั้นเมื่อข้อความ RREP ถูกส่งไปถึงโหนดต้นทางที่มีความต้องการเส้นทาง ค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลภายในข้อความ จะหมายถึงระยะทางหรือจำนวนของโหนดที่จำเป็นต้องทำการส่งต่อข้อมูลระหว่างโหนดต้นทางกับโหนดปลายทาง

7) การรับและส่งต่อข้อความ RREP

ขั้นตอนแรกหลังจากเมื่อโหนดได้รับข้อความ RREP จะทำการค้นหาเส้นทางเพื่อทำการส่งข้อมูลไปยังโหนดตัวก่อนหน้า (Previous hop) หลังจากนั้นจะทำการเพิ่มค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลไปยัง RREP ขึ้นหนึ่ง และจะทำการสร้างเส้นทางไปข้างหน้า (Forward route) ไปยังโหนดปลายทางถ้าโหนดไม่มีการสร้างเส้นทางไปข้างหน้า แต่ถ้าเคยมีเส้นทางไปข้างหน้าอยู่โหนดจะทำการเปรียบเทียบค่าหมายเลขลำดับของโหนดปลายทางภายในข้อความกับหมายเลขลำดับของโหนดปลายทางที่มันเคยเก็บไว้ ซึ่งจะถูกปรับปรุงก็ต่อเมื่อมีเหตุการณ์ดังต่อไปนี้

- หมายเลขลำดับภายในตารางเส้นทางที่โหนดเก็บค่าอยู่นั้นถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้ (Invalid Routing table) ภายในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย
- หากหมายเลขลำดับของโหนดปลายทางในข้อความ RREP มีค่ามากกว่าหมายเลขลำดับภายในตารางเส้นทาง ค่าของหมายเลขลำดับของโหนดปลายทางจะถูกคัดลอกเก็บไว้
- หมายเลขลำดับมีค่าเท่ากัน แต่เส้นทางดังกล่าวถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้
- หมายเลขลำดับมีค่าเท่ากัน แต่ค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลใหม่ซึ่งได้รับมาจากข้อความ RREP มีค่าน้อยกว่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลใหม่ที่เคยถูกจัดเก็บไว้ในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายก่อนหน้า

ถ้าหากข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทางถูกสร้างหรือถูกปรับปรุงก็จะดำเนินไปตามขั้นตอนต่อไปนี้

- เส้นทางจะถูกตั้งค่าให้เป็นเส้นทางที่ใช้ได้
- หมายเลขลำดับของโหนดปลายทางจะถูกตั้งค่าให้เป็นค่าที่ใช้งานได้
- โหนดตัวถัดไปในตารางเส้นทางจะถูกตั้งค่าให้เป็นค่าของหมายเลขโหนดที่ได้ทำการส่งข้อความRREP มาสู่โหนดที่ได้รับข้อมูลดังกล่าว
- จำนวนโหนดที่ใช้ในการส่งข้อมูลจะได้จากการเพิ่มค่าขึ้นหนึ่งค่าจากค่าที่ได้จากข้อความ RREP
- ส่วนของค่าเวลาหมดอายุ (Expiry time) จะถูกตั้งค่าให้มีค่าเท่ากับค่าเวลาของเวลาปัจจุบันบวกกับค่าของเวลาที่ใช้พิจารณาว่าเส้นทางดังกล่าวยังใช้งานได้ (Lifetime) ที่ได้จากข้อความ RREP
- หมายเลขลำดับของโหนดปลายทางจะเป็นค่าเดียวกับหมายเลขลำดับของโหนดปลายทางที่อยู่ภายในข้อความ RREP

เมื่อใดก็ตามที่โหนดมีการส่งข้อความ RREP ส่วนของรายการของหมายเลขโหนดที่ถูกใช้ในการส่งข้อความ RREP สำหรับโหนดปลายทางจะถูกปรับปรุงโดยการเพิ่มหมายเลขโหนดถัดไปที่จะทำการส่งข้อความ RREP ต่อและเส้นทางที่ใช้ในการส่งหมายเลข RREP กลับคือเส้นทางย้อนกลับ

8) ข้อความตรวจสอบโหนดข้างเคียง (Hello message)

โหนดอาจใช้วิธีการในการตรวจสอบการเชื่อมต่อกับระหว่างสองโหนดโดยวิธีการส่งข้อความตรวจสอบโหนดข้างเคียงแบบแพร่กระจาย ซึ่งโหนดควรจะมีการใช้งานข้อความตรวจสอบโหนดข้างเคียงเฉพาะในเส้นทางที่ใช้งานเท่านั้น โดยจะทำการส่งแบบแพร่กระจายในทุกๆช่วงเวลา HELLO_INTERVAL มิลลิวินาที ซึ่งข้อความตรวจสอบโหนดข้างเคียงที่ทำการส่งแบบแพร่กระจายนั้นจัดอยู่ในประเภทของข้อความ RREP ที่มีค่า TTL = 1 โดยที่มีการแก้ไขข้อมูลบางส่วนดังนี้

- หมายเลขของโหนดปลายทางคือหมายเลขของโหนดที่ทำการส่งข้อความตรวจสอบโหนดข้างเคียงแพร่กระจาย
- หมายเลขลำดับของโหนดปลายทางคือหมายเลขลำดับล่าสุดของโหนดที่ส่งแบบแพร่กระจาย
- เวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้มีค่าจากการคำนวณของ ALLOWED_HELLO_LOSS*HELLO_INTERVAL

โหนดอาจจะตรวจสอบการเชื่อมต่อโดยคอยรับฟังข้อความที่ได้จากโหนดข้างเคียง ถ้าโหนดดังกล่าวไว้รับข้อความตรวจสอบโหนดข้างเคียงจากโหนดข้างเคียง และภายในช่วงเวลา ALLOWED_HELLO_LOSS*HELLO_INTERVAL มิลลิวินาที โหนดที่ไม่ได้รับข้อความตรวจสอบ

โหนดข้างเคียงจากโหนดข้างเคียงจากโหนดเดิม โหนดดังกล่าวจะสมมุติว่าเกิดการเชื่อมต่อเสียหาย (Link failure) ระหว่างตนเองกับโหนดข้างเคียงขึ้น

เมื่อโหนดได้รับข้อความตรวจสอบโหนดข้างเคียงจากโหนดข้างเคียง และโหนดดังกล่าวมีเส้นทางที่ใช้งานได้และต้องใช้งานโหนดข้างเคียงเป็นโหนดถัดไป โหนดจะเพิ่มค่าเวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้ให้แก้เส้นทางนั้นๆ โดยมีค่าในการเพิ่มอย่างน้อยที่สุดคือผลบวกระหว่างค่าของ ALLOWED_HELLO_LOSS*HELLO_INTERVAL

9) การบำรุงรักษาการเชื่อมต่อระหว่างโหนด (Maintaining Local Connectivity)

โหนดแต่ละตัวที่ทำหน้าที่ในการส่งข้อมูลต่อในระบบ (โหนดที่อยู่ระหว่างเส้นทางที่ใช้งานได้) ควรจะทำการตรวจสอบการเชื่อมต่อสื่อสารไปยังโหนดถัดไปตลอดจนโหนดข้างเคียงที่เคยมีการส่งข้อความตรวจสอบโหนดข้างเคียง ภายในช่วงระยะเวลาของ ALLOWED_HELLO_LOSS*HELLO_INTERVAL ซึ่งโหนดอาจทำการตรวจสอบการเชื่อมต่อกับโหนดถัดไปได้ 2 วิธีคือ

1) Link layer notification
2) กรณีไม่มี Link layer notification โหนดจะใช้วิธีการรอฟังข้อมูลจากช่องสัญญาณ ซึ่งอาจอยู่ในรูปแบบแพ็กเก็ตที่ได้รับจากโหนดถัดไป (รวมไปถึงข้อความตรวจสอบโหนดข้างเคียง) ถ้าหากว่าโหนดไม่ได้รับข้อมูลใดๆจากโหนดถัดไปภายในช่วงเวลา LLOWED_HELLO_LOSS*HELLO_INTERVAL โหนดจะสมมุติว่าเกิดการเชื่อมต่อเสียหายและจะเริ่มทำการส่งข้อความ RERR ในขั้นตอนต่อไป

10) การจัดการข้อความ RERR และการลบข้อมูลเส้นทาง

โดยทั่วไปแล้วข้อความ RERR จะใช้เมื่อการเชื่อมต่อมีการเสียหายโดยจะมีกระบวนกาต่อไปนี้

1) เส้นทางที่มีอยู่เดินเกิดความไม่ถูกต้อง
2) ตรวจสอบโหนดปลายทางที่ได้รับผลกระทบจากเส้นทางที่เสียหาย
3) ทำการกำหนดว่าโหนดข้างเคียงใดบ้างที่ได้รับผลกระทบ
4) ทำการส่งข้อความ RERR ไปยังโหนดข้างเคียง

ซึ่งข้อความ RERR อาจส่งได้ทั้งรูปแบบการแพร่กระจายและแบบปลายทางเดียว และโหนดจะไม่ทำการสร้างข้อความ RERR เกินค่าของ RERR_RATELIMIT ข้อความต่อ 1 วินาที โหนดจะเริ่มทำการสร้างข้อความ RERR ใน 3 กรณีต่อไปนี้

1) ถ้าโหนดตรวจพบได้ว่าการเชื่อมต่อเสียหายระหว่างตนเองกับโหนดถัดไปที่ต้องใช้งานในเส้นทางที่ใช้งานได้ โหนดจะทำการสร้างรายชื่อของโหนดปลายทางที่ตนเองไม่สามารถส่งข้อมูลไปถึงได้ลงในข้อความ RERR

2) ถ้าโหนดได้รับข้อมูลและตนเองไม่มีเส้นทางที่ใช้งานได้ (กรณีที่ไม่มีการทำงาน Local repair) ซึ่งหมายถึงโหนดดังกล่าวจะมีรายชื่อของโหนดปลายทางที่ไม่สามารถส่งข้อมูลได้เพียงแค่โหนดเดียว
3) ถ้าโหนดได้รับข้อความ RERR จากโหนดข้างเคียง ในกรณีดังกล่าวรายชื่อของโหนดปลายทางที่ไม่สามารถส่งข้อมูลได้จะทำการคัดลอกค่าจากข้อความ RERR ที่ได้รับมา

และก่อนจะมีการส่งข้อความ RERR จะต้องทำการปรับปรุงค่าหมายเลขลำดับของโหนดปลายทางในตารางเส้นทางสำหรับโหนดปลายทางที่ไม่สามารถส่งข้อมูลถึงได้ โดยกระบวนการปรับปรุงในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายมีดังต่อไปนี้

1) หมายเลขลำดับของโหนดปลายทางในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย มีค่าถูกต้องจะทำการเพิ่มค่าหมายเลขลำดับขึ้น 1 ในกรณีที่โหนดต้องการเส้นทางเพื่อส่งข้อมูลไปยังปลายทางที่ไม่สามารถส่งข้อมูลถึงได้ ส่วนในกรณีที่ได้รับข้อความ RERR จะทำการคัดลอกค่าหมายเลขลำดับของโหนดปลายทางมาจากข้อความ RERR แทน
2) ข้อมูลเส้นทางภายในตารางเส้นทางจะถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้ นั้นคือจะตั้งค่าให้ข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายดังกล่าวเป็นเส้นทางที่ไม่สามารถใช้งานได้อีก
3) ส่วนค่าเวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้จะมีค่าเท่ากับผลรวมระหว่างเวลาปัจจุบันกับ DELETE_PERIOD เพื่อกำหนดให้ทราบว่าเมื่อเกิดเส้นทางเสียหายดังกล่าวขึ้นโหนดจะเก็บเส้นทางดังกล่าวไว้ระยะหนึ่งก่อนที่จะทำการลบข้อมูลดังกล่าวทิ้ง

11) Local Repair

หากมีการตรวจพบว่าเส้นทางมีการเสียหายโหนดอาจเลือกใช้วิธีซ่อมแซมตัวเองหรือ Local Repair โดยจะดูว่าโหนดปลายทางอยู่ใกล้ตัวเองหรอต้นทางมากกว่า หากอยู่ใกล้โหนดปลายทาย ทาง โหนดที่ตรวจพบว่าเสียหายจะทำการเพิ่มหมายเลขลำดับของโหนดปลายทางและทำการส่งข้อความ RREQ แบบแพร่กระจายไปยังโหนดปลายทางด้วยตัวเอง ซึ่งเมื่อมีการทำงานในการซ่อมแซมเส้นทางดังกล่าว โหนดต้นทางจะไม่ทราบหรือไม่ได้รับข้อความ RREQ ที่โหนดดังกล่าวส่งออกมา เนื่องจากโหนดจะทำการกำหนดค่าข้อความ RREQ ที่ตนเองทำการส่งแบบแพร่กระจายให้มีค่าระหว่าง MIN_REPAIR_TTL ถึงครึ่งหนึ่งของจำนวนโหนดที่ใช้ในการส่งข้อมูลจากตนเองไปยังโหนดต้นทาง และจะทำการเพิ่มค่าขึ้นครั้งละ LOCAL_ADD_TTL และภายหลังจากโหนดที่ทำการตรวจพบว่าเกิดการเชื่อมต่อเสียหายทำการส่งข้อความ RREQ แบบแพร่กระจายออกมา โหนดจะทำการรอข้อความ RREP ซึ่งในระหว่างเวลาดังกล่าวข้อมูลต่างๆที่โหนดได้รับจะถูกเก็บไว้

ชั่วคราว ถ้าภายหลังจากช่วงเวลาการรอ และโหนดไม่ได้รับข้อความ RREP สำหรับใช้ในการส่งข้อมูลไปยังปลายทาง โหนดจะทำการเริ่มกระบวนการในการส่งข้อความ RERR ทันที แต่ถ้าโหนดได้รับข้อความ RREP ในระหว่างช่วงระยะเวลาในการรอที่กำหนดไว้โหนดจะทำการตรวจสอบว่าข้อความ RREP แต่ละข้อความที่ได้รับมามีค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลดีกว่าค่าที่ตนเองเคยบันทึกไว้ในตารางเส้นทางหรือไม่หากค่าภายในข้อความ RREP ที่ได้รับมีค่ามากกว่าที่บันทึกไว้ กระบวนการส่งข้อความ RERR ก็จะเกิดขึ้น ซึ่งกระบวนการทำ Local Repair ดังกล่าวจะเป็นการเพิ่มความสำเร็จในการส่งข้อมูลเนื่องมาจากข้อมูลจะไม่ถูกทิ้งไปในระหว่างส่งข้อความ RERR เมื่อเกิดการเชื่อมต่อเสียหายขึ้น

เมื่อการเชื่อมต่อเสียหายหรือเกิดการเชื่อมต่อเสียหายขึ้นภายในเส้นทางที่ใช้งานได้ซึ่งในบางกรณีอาจทำให้ไม่สามารถส่งข้อมูลไปยังโหนดปลายทางหลายๆตัวได้ เนื่องจากอาจมีการใช้งานโหนดตัวเดียวกันในการส่งข้อมูลไปยังโหนดปลายทางหลายๆตัว โหนดที่ทำการตรวจพบว่าเกิดความเสียหายในการเชื่อมต่อดังกล่าว เมื่อเริ่มกระบวนการทำ Local Repair ขึ้นจะดำเนินการค้นหาเส้นทางใหม่อีกครั้งละหนึ่งโหนดปลายทางเท่านั้น ซึ้งแสดงว่าเส้นทางที่ถูกใช้ในการส่งข้อมูลไปยังโหนดปลายทางอื่นๆจะต้องถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้ แต่ถ้าโหนดอยู่ในกระบวนการทำ Local Repair อาจจะถูกตั้งค่าสถานะว่ากำลังทำ Local Repair อยู่ โดยการตั้งค่าสถานะดังกล่าวจะถูกรีเซ็ตภายในระยะเวลาที่กำหนดไว้ คือหลังจากไม่ได้รับการปรับปรุงภายในเวลา ACTIVE_ROUTE_TIMEOUT ซึ่งก่อนจะหมดเวลาดังกล่าวอาจมีข้อมูลที่ต้องการส่งไปยังโหนดปลายทางอื่นๆ ดังนั้นเส้นทางที่ทำ Local Repair นั้นจะเกิดจากการที่มีข้อมูลที่ต้องการส่งไปยังโหนดปลายทางนั้นๆเท่านั้น หากไม่มีความต้องการในการส่งข้อมูลไปยังโหนดปลายทางกระบวนการทำ Local Repair เพื่อใช้ในการค้นหาเส้นทางในการส่งข้อมูลไปยังโหนดปลายทางดังกล่าวก็จะไม่เกิดขึ้น

2.3.5 การตรวจสอบการเชื่อมต่อ

มีวิธีการตรวจสอบอยู่ 2 วิธีคือ

1) Hello Messages

แต่ละโหนดจะทำการส่ง Hello message ทุกๆช่วงเวลาที่กำหนด หากได้รับ Hello message กลับ โหนดทั้งคู่ก็จะเป็นโหนดเพื่อนบ้านซึ่งกันและกัน Hello message ใช้เพื่อตรวจสอบสถานะการเชื่อมต่อและปรับปรุงตารางเส้นทางเสมอๆ หากโหนดเพื่อนบ้านโหนดใดหายไปและเป็นค่าที่อยู่ในตารางเส้นทางก็จะทำการลบค่าในตารางเส้นทางในข้อมูลนั้นด้วย

เมื่อโหนดใดโหนดหนึ่งไม่ได้รับ Hello message จากเพื่อนบ้าน ตามช่วงเวลาที่กำหนดแล้วโหนดนั้นจะทำการประกาศ RRER กระจายออกไปให้ทุกโหนดทราบว่า โหนดนี้ได้ออกจากรัศมีแล้ว และหากในตารางเส้นทางมีโหนดนี้อยู่ โหนดที่ได้รับ RRER ก็จะทำการทำให้ข้อมูลนั้นหมดอายุไปเพื่อรอการหาเส้นทางใหม่ต่อไป ประโยชน์ของ RRER มีเพื่อไม่ต้องการให้โหนดแต่

ละโหนดนับเวลาถอยหลังเอง หากมีโหนดใดโหนดหนึ่งเสียหรือออกจากรัศมีก็จะสามารถทราบได้โดยเร็ว

2) Link layer feedback

เป็นอีกหนึ่งวิธีในการตรวจสอบสถานการณ์เชื่อมต่อแต่กระบวนการของ Link layer feedback จะได้รับการสนับสนุนจากชั้น Media access control (MAC) ซึ่งสามารถแจ้งเตือนได้เร็วกว่าแบบ Hello Messages โดยวิธี Link layer feedback จะมีวิธีการทำงานคือ เมื่อมีความต้องการจะส่งข้อมูล โหนดจะทำการส่งข้อความต้องการในการส่งข้อมูลไปยังชั้น MAC ซึ่งจะเก็บค่าไว้เพื่อส่งข้อมูลไปยังโหนดที่ต้องการ และเมื่อทำการส่งข้อมูลไปยังโหนดที่ต้องการไม่สำเร็จ โหนดก็จะทราบทันทีว่าการเชื่อมต่อเกิดการเสียหายขึ้น และจะทำการแจ้งเตือนว่าเกิดการเชื่อมต่อเสียหาย

2.4 มาตรฐานการทำงานของระบบเครือข่าย

มาตรฐาน IEEE 802.11b/g/n [5] มีดังนี้

2.4.1 มาตรฐาน IEEE 802.11b

802.11b เป็นมาตรฐานที่ได้รับความนิยมอย่างแพร่หลายรวมทั้งประเทศไทยด้วยเช่นกัน ทำงานที่คลื่นความถี่ 2.4 GHz (คลื่นความถี่นี้สามารถใช้งานแบบสาธารณะในประเทศไทยได้) มีความสามารถในการรับส่งข้อมูลที่ความเร็ว 11 Mbps ผลิตภัณฑ์อุปกรณ์เครือข่ายไวเลสแลนมาตรฐานนี้ได้รับความนิยมจำนวนมากมาก โดยทุกผลิตภัณฑ์ต้องสามารถทำงานร่วมกันได้ อุปกรณ์ทุกยี่ห้อต้องผ่านการตรวจสอบจากสถาบัน Wi-Fi Alliance เพื่อตรวจสอบมาตรฐานของอุปกรณ์และความเข้ากันได้ของแต่ละผู้ผลิต อุปกรณ์ไวเลสแลนที่มาตรฐาน 802.11b ไปใช้ในองค์กรธุรกิจ สถาบันการศึกษา สถานที่สาธารณะ และกำลังแพร่เข้าสู่สถานที่พักอาศัยมากขึ้น และมาตรฐานนี้มีระบบเข้ารหัสข้อมูลแบบ WEP ที่ 128 บิต

2.4.2 มาตรฐาน IEEE 802.11g

มาตรฐาน 802.11g ใช้ความถี่ 2.4 GHz สามารถรับส่งข้อมูลที่ความเร็ว 36 - 54 Mbps ซึ่งเป็นความเร็วที่สูงกว่ามาตรฐาน 802.11b โดยมาตรฐาน 802.11g สามารถปรับระดับความเร็วในการสื่อสารลงเหลือ 2 Mbps ได้ (ตามสภาพแวดล้อมของเครือข่ายที่ใช้งาน) มาตรฐานนี้เป็นที่นิยมของผู้ใช้เป็นจำนวนมากและเข้ามาแทนที่ 802.11b ที่ความเร็วต่ำกว่า

2.4.3 มาตรฐาน IEEE 802.11n

เป็นมาตรฐานที่สามารถทำงานบนคลื่นความถี่ 2.4 และ 5 GHz ได้ รองรับความเร็วตั้งแต่ 300-450 Mbps โดยมีเสาสัญญาณตั้งแต่ 2 - 4 เสา บนตัวอุปกรณ์กระจายสัญญาณไวเลสแลน และหากผู้ใช้ต้องการใช้งานที่ความเร็วสูงสุด เครื่องคอมพิวเตอร์

พกพาหรืออุปกรณ์เคลื่อนที่ต้องรองรับมาตรฐาน 802.11n ด้วยเช่นกัน มาตรฐาน 802.11n สามารถทำงานร่วมกับ 802.11b, g ได้ โดยไม่ทำให้ประสิทธิภาพทั้งระบบลดลงเหมือนมาตรฐาน 802.11g เมื่อมีอุปกรณ์ 802.11b เข้ามาใช้งานร่วมกันสุดท้าย

2.5 Network simulator

Network simulator [6] เป็นโปรแกรมที่ใช้ในการจำลองการทำงานของ Network มีมากมายหลายโปรแกรมมากในปัจจุบัน ซึ่งแต่ละโปรแกรมก็มีความสามารถ ในแต่ละด้านที่เพิ่มขึ้นมาแตกต่างกัน โปรแกรมที่ใช้ในการจำลองการทำงานของ Network ดังต่อไปนี้

2.5.1 NS2

เป็น open-source และสามารถที่จะ run ได้ทั้งบน Linux , FreeBSD, SunOS, Solaris, Window ถูกพัฒนาขึ้นโดย ISI (Information Sciences Institute) NS2 เป็นโปรแกรมที่ใช้ในการ จำลองการทำงานของ network ในแบบที่เป็น discrete event simulator ซึ่งสนับสนุนการจำลองการเลือกเส้นทางในการขนส่ง packet, การจำลองการทำงานของ multicast protocol และ IP protocol เช่น UDP, TCP, RTP, SRM ที่อยู่บนเครือข่ายประเภทที่เป็น wire และ wireless (local และ satellite) ซึ่ง NS2 เป็น tool ที่มีประโยชน์มากทั้งยังสนับสนุน multiple protocol และยังมีความสามารถในการ แสดงรายละเอียดของ network traffic ออกมาในรูปแบบของกราฟิก รวมทั้งยังสนับสนุน algorithm ในการ routing และ queuing เช่น FIFO, round-robin เป็นต้น

2.5.2 OPNET

เป็นโปรแกรมที่เป็นผลิตภัณฑ์ทางพาณิชย์ซึ่งได้รวบรวม Model ต่างๆและครอบคลุมถึงเทคโนโลยีทางด้าน Network เพื่อรองรับให้ผู้ใช้สามารถที่จะออกแบบและจัดการกับระบบเครือข่ายขั้นพื้นฐานพร้อมทั้งยังมีการจำลองอุปกรณ์ทางด้าน network และ network application เพื่อให้ผู้ใช้นั้นได้พัฒนาระบบเครือข่ายเสมือนกับสภาพแวดล้อมในความเป็นจริง

2.5.3 CSim

เป็น Simulation Environment ที่มีพื้นฐานมาจากภาษา C ใน CSim จะมีการทำโมเดลที่มีความเป็นอิสระและมีความสัมพันธ์ระหว่าง entity มีลักษณะการเก็บข้อมูลแบบเป็นลำดับชั้นจึงเป็นระบบที่ช่วยทำให้จัดเก็บแฟ้มข้อมูลเป็นกลุ่ม สามารถเพิ่มประสิทธิภาพในการทำงานในเวลาเดียวกันภายในระบบได้ พร้อมทั้งยังสนับสนุนระบบการทำงานประเภท rates, data, bandwidth, latency และ traffic เป็นต้น ใน CSim นั้นจะมีแบบจำลองของซอฟต์แวร์เป็นเชิงกลศาสตร์ (concurrent process oriented) พร้อมทั้งยังมี Methods ในการช่วยพัฒนาการทำงานทั้งแบบที่เป็น

Graphical และ Textual ในเรื่องของ Graph file นั้นจะมี Graph file เป็น XML ascii-text เพื่อช่วยเพิ่มความสะดวกในเรื่องการทำงานข้าม platform

2.5.4 SWANS (Scalable Wireless As hoc Network Simulator)

เป็น scalable wireless network simulator ซึ่งสร้างอยู่บน JiST platform (Java in Simulation Time) SWANS มีการจัดการระบบในรูปแบบของ independent software component ทำ ให้สามารถที่เปลี่ยนรูปแบบเป็น complete wireless network หรือ sensor network configurations ความสามารถของ SWANS นั้นเหมือนกับ NS2 และ GloMoSim แตกต่างกันที่ SWANS นั้นจะรองรับการจำลองNetwork ขนาดใหญ่ๆได้ซึ่ง SWANS ยังสามารถแสดงผลลัพธ์ของการจำลองในระดับสูง พร้อมทั้งยังช่วยประหยัดหน่วยความจำและสามารถ run Standard Java network application ได้บน simulate network

2.5.5 OMNet++

เป็น discrete event simulation environment ซึ่ง OMNet++ นั้นสามารถที่จะจำลองการสื่อสารของ Network protocol, computer network, traffic modeling, multi-processor และ distributed system เป็นต้น OMNet++ นั้นไม่ได้เป็น Network simulator ซึ่งตัว OMNet++ เป็นแค่ network simulation platform ซึ่ง Models ต่างๆของ OMNet++ นั้นจะเป็น Component architecture ซึ่งคอมโพเนนท์ต่างๆนั้นถูกพัฒนาด้วยภาษา C++ ซึ่งการสร้างคอมโพเนนท์ขนาดใหญ่และโมเดลนั้นจะใช้ภาษาระดับสูง (high-level language) ในการเขียนซึ่งภาษาระดับสูงที่ใช้คือ ภาษา NED (Network Description)

2.5.6 NCTUns3.0

เป็น open-source โปรแกรม NCTUns นั้นพัฒนาขึ้นมาเพื่อใช้เป็น network simulation และ emulation สำหรับ wireless และ mobile network ซึ่ง NCTUns จะสร้างการจำลองสำหรับ wireless ad hoc, sensor, inter-vehicle communication network, GPRS cellular network และ wireless mesh network ซึ่งความสามารถอีกอย่างของ NCTUns คือ สามารถที่จะแบ่งแยก network device และ network protocol ออกเป็นหมวดหมู่ได้

2.5.7 NS3

Network Simulator 3 หรือ NS3 เป็นโปรแกรมที่ใช้ในการจำลองการทำงานของเครือข่ายโดยใช้แบบจำลองทางคณิตศาสตร์ที่ ได้รับการยอมรับอย่างแพร่หลายในเชิงวิชาการงานวิจัยด้านเครือข่ายคอมพิวเตอร์ เนื่องจากสามารถใช้ในการศึกษา พฤติกรรมทางเครือข่ายเพื่อการทำวิจัยได้หลากหลาย โดย NS3 ทำงานภายใต้สภาพแวดล้อมที่

เป็นยูนิกซ์ ซึ่งผู้ใช้งานระบบปฏิบัติการวินโดวส์สามารถจำลองสภาพแวดล้อมการทำงานระบบยูนิกซ์ได้เช่นกันโดยผ่านโปรแกรมจำลองเช่น VMware, Cygwin เป็นต้น

คุณสมบัติของโปรแกรม NS3

- NS3 มีภาษาเดียวคือ C++
- สามารถรองรับการทำงานของภาษา Python
- เนื่องจาก NS3 ใช้ภาษา C++ เป็นภาษาหลักการจัดการกับหน่วยความจำยังคงเหมือนเดิม
- ในแพ็กเก็ตมีหนึ่งบัฟเฟอร์ ซึ่งจะมีข้อมูลแท็กทั้งหมดเก็บไว้ใน meta-data
- มีประสิทธิภาพในด้านการจัดการความจำได้ดีกว่า NS2
- มีแพ็คเก็ต PyViz ซึ่งเป็นภาษา Python ใช้ในการแสดงผล
- NS3 ไม่สามารถเข้ากันได้กับ NS2 เนื่องจากเป็นเวอร์ชั้นที่พัฒนาขึ้นมาใหม่ไม่อิงกับรุ่นเดิม
- NS3 สามารถเขียนโดยภาษา C++ และ Python ซึ่งสามารถเลือกมาใช้ได้ในการแสดงผล
- NS3 มี emulator mode ซึ่งจะสามารถจำลองเครื่องจริงได้ ซึ่งใน NS2 จะไม่มีโหมดนี้

2.6 การหาค่าประสิทธิภาพการทำงานของระบบเครือข่าย

งานวิจัยฉบับนี้ได้มีการจำลองการทำงานของเครือข่ายเพื่อวัดประสิทธิภาพการทำงานของโพรโทคอล AODV ที่ใช้กับเครือข่ายเมชไร้สาย เพื่อนำมาเปรียบเทียบกับโพรโทคอล AODV ในเครือข่ายต่างๆในงานวิจัยที่เกี่ยวข้องที่มีการทดสอบมาก่อนแล้ว ซึ่งค่าที่ใช้วัดประสิทธิภาพการทำงานของโพรโทคอล การทำงานของระบบเครือข่ายประกอบด้วยค่าต่างๆ ดังต่อไปนี้ [7]

2.6.1 ค่าอัตราปริมาณงาน (Throughput)

คือ ค่าของจำนวนข้อมูลที่โหนดปลายทางสามารถรับได้ทั้งหมด ต่อหนึ่งหน่วยเวลาที่เราสนใจ ซึ่งในช่วงเวลาที่สนใจหากข้อมูลจากโหนดต้นทาง ส่งไปไม่ถึงโหนดปลายทางย่อมส่งผลให้ค่าอัตราปริมาณงานมีค่าลดลง โดยสามารถคำนวณหาค่าอัตราปริมาณงานได้จากสมการรูปที่ 2.14

$$\text{Throughput} = \frac{\text{Packets received} \times \text{Packets size} \times 8}{\text{Runtime}} \text{ (bps)} \quad (1)$$

รูปที่ 2.14 สูตรคำนวณค่า Throughput

Throughput คือ ค่าอัตราปริมาณงาน

Packets received คือ ข้อมูลที่รับได้ทั้งหมด

Packets size คือ ขนาดของแพ็กเกจ

Runtime คือ ระยะเวลาที่สิ้นสุดการส่ง – ระยะเวลาที่เริ่มต้นส่ง

2.6.2 ค่าเปอร์เซ็นต์การส่งสำเร็จ (Packet Delivery Ratio)

คือ ค่าที่บอกความสำเร็จในการส่งข้อมูล มีค่าไม่เกิน 100 เปอร์เซ็นต์ ซึ่งในการจำลองการทำงานของระบบเครือข่ายที่มีโหนดจำนวนมาก และมีการเชื่อมต่อพร้อมๆ กันหลายการเชื่อมต่อ หากเกิดการชนกันของข้อมูล ค่าเปอร์เซ็นต์การส่งสำเร็จจะมีค่าลดลงและในระบบเครือข่ายที่มีการกำหนดค่าพลังงานให้กับแต่ละโหนด หากโหนดในเส้นทางสื่อสารพลังงานหมดลง ย่อมส่งผลให้ข้อมูลที่ถูกส่งออกมาจากโหนดต้นทางไม่สามารถไปถึงโหนดปลายทางได้ ดังนั้นค่าเปอร์เซ็นต์การส่งสำเร็จจะมีค่าลดลงเช่นเดียวกัน โดยค่าเปอร์เซ็นต์ การส่งสำเร็จสามารถคำนวณได้จากสมการรูปที่ 2.15

$$PDR = \frac{\text{Total packets received} \times 100}{\text{Total packets sent}} \text{ (Percent)} \quad (2)$$

รูปที่ 2.15 สูตรคำนวณค่า Packet Delivery Ratio

PDR คือ ค่าเปอร์เซ็นต์การส่งสำเร็จ

Total packets received คือ จำนวนข้อมูลที่รับได้ทั้งหมด

Total packets sent คือ จำนวนข้อมูลที่ส่งออกไปทั้งหมด

2.6.3 จำนวนข้อความสื่อสารในระบบเครือข่าย (Overhead)

คือ ข้อความที่ถูกส่งออกมาเพื่อใช้ในการจัดการการสื่อสาร โดยนับจำนวนข้อความสื่อสารในระบบเครือข่าย และนับเฉพาะข้อความสื่อสารจาก โพรโทคอล AODV ได้แก่ ข้อความ RREQ ข้อความ RREP และ ข้อความ RERR ยิ่งมีจำนวนข้อความสื่อสารในระบบน้อยจะส่งผลดีต่อระบบเครือข่าย เนื่องจากในการส่งข้อความสื่อสารออกมาในระบบเครือข่ายจะใช้การแพร่กระจาย (Broadcast) ซึ่งถือเป็นการสิ้นเปลืองแบนด์วิดท์และทรัพยากร ดังนั้นหากสามารถลดจำนวนข้อความสื่อสารที่ส่งออกมาย่อมส่งผลดีต่อระบบเครือข่ายโดยรวม

2.6.4 จำนวนข้อมูลที่ถูกทิ้ง (Packet Drop)

ข้อมูลที่ถูกทิ้ง หมายถึง ข้อมูลที่ส่งออกมาจากโหนดต้นทางแล้วโหนดปลายทางไม่สามารถรับได้ ซึ่งหากมีข้อมูลที่ถูกทิ้งจำนวนมาก จะส่งผลทำให้ค่าเปอร์เซ็นต์การส่งสำเร็จลดลง สาเหตุของการที่ข้อมูลถูกทิ้ง เกิดจากการที่โหนดต้นไม่สามารถหาเส้นทางไปยังโหนดปลายทางได้และเกิดจากการที่ข้อมูลชนกัน (Collision) โดยจะเกิดขึ้นในระดับชั้น MAC เมื่อโหนดในระบบเครือข่ายได้รับข้อมูลจากโหนดอื่นๆ พร้อมกัน ส่งผลให้เกิดการชนกันและไม่สามารถรับข้อมูลได้ ปัจจัยที่ส่งผลต่อการเพิ่มจำนวนข้อมูลที่มีการชนกัน เช่น จำนวนการเชื่อมต่อ โดยยิ่งเพิ่มจำนวนการ เชื่อมต่อมากขึ้น ยิ่งส่งผลให้เกิดการชนกันบ่อยครั้งขึ้น เป็นต้น

2.6.5 ค่าหน่วง (End to End Delay)

เป็นค่าหน่วงของการเดินทางของแพ็คเกจจากต้นทางไปยังปลายทาง ซึ่งมีสาเหตุมาจาก

1) เวลาหน่วงของสื่อนำสัญญาณ (Transmission delay)
2) เวลาหน่วงที่เกิดจากการที่แพ็คเกจต้องรออยู่ในคิวในกรณีที่เครือข่ายเกิดความคับคั่ง (Queuing delay)
3) เวลาหน่วงที่เกิดจากการรวมดาต้าแกรม (Datagram) ให้เป็นแพ็กเก็ต (Packet reassembly delay)
4) เวลาหน่วงที่เกิดจากการพิจารณาของโหนด หรือเกตเวย์ ในการเปลี่ยนข้อมูลในหัวของดาต้าแกรม (Header) เช่น Hop Count หรือ Time-to-Live (Router and other processing delay)
5) เวลาหน่วงที่เกิดจากการบันทึกข้อมูลลงสื่อ และเวลาในการเข้าถึงหน่วยบันทึกของอุปกรณ์จาพวกสวิตซ์และบริดจ์ (Other computer storage delay)

2.6.6 การคำนวณค่าพลังงาน (Energy)

การคำนวณค่าพลังงานที่เซนเซอร์ใช้ในการรับและส่งข้อมูลแต่ละครั้งมีหน่วย เป็นจูล ทำการคำนวณจากค่ากำลังที่ใช้ส่งข้อมูล (txPower) และค่ากำลังที่ใช้รับข้อมูล (rxPower) ทั้งสองค่ามีหน่วยเป็นวัตต์ โดยมีสมการในการคำนวณค่าพลังงานดังสมการรูปที่ 2.16

$$\text{Energy} = \frac{\text{Power} \times 8 \times \text{Packet size}}{\text{Bandwidth}} \text{ (Joule)} \_\_\_\_\_ (3)$$

รูปที่ 2.16 สูตรคำนวณค่าพลังงาน

2.6.7 การคำนวณหาค่าช่วงความเชื่อมั่น 95 เปอร์เซ็นต์ (Confidence interval 95%)

ในการจำลองการทำงานเพื่อวัดค่าประสิทธิภาพการทำงานของโพรโทคอล ในแต่ละการทดลองได้ทำการทดลองซ้ำ โดยการเปลี่ยนตำแหน่งของแต่ละโหนดแบบสุ่มหรือเปลี่ยนโหนดต้นทาง ซึ่งการที่จะบอกได้ว่า ผลลัพธ์จากการจำลองการทำงานเมื่อนำค่ามาเปรียบเทียบกัน แต่ละการทดลองมีค่าต่างกันอย่างมีนัยสำคัญหรือไม่ ต้องคำนวณเพื่อหาช่วงความเชื่อมั่นที่ 95 เปอร์เซ็นต์ โดยมีสูตรการคำนวณดังสมการรูปที่ 2.17

$$CI195\% = \frac{1.96 \times \alpha}{\sqrt{n}} \_\_\_\_\_ (4)$$

รูปที่ 2.17 สูตรคำนวณค่า Confidence interval 95%

α คือ ค่าความแปรปรวน

n คือ จำนวนครั้งในการทดสอบ

ค่าที่อยู่ในช่วงความเชื่อมั่นจะอยู่ระหว่าง ค่าเฉลี่ยจากการทดลอง +CI95% และ - CI95%

ตารางที่ 2.5 ความหมายของค่าพารามิเตอร์ที่ใช้ในการจำลอง

พารามิเตอร์	ความหมาย
MAC layer	โพรโทคอลในระดับชั้น MAC
Network layer	โพรโทคอลในระดับชั้นเครือข่าย
Transport layer	โพรโทคอลในระดับชั้นทรานสปอร์ต
Application layer	ชนิดของข้อมูลที่ทำการส่ง
Number of Nodes	จำนวนโหนดที่ทำการทดลอง
Area	พื้นที่ มีหน่วยเป็นตารางเมตร
Transmission Range	ระยะในการส่งข้อมูล มีหน่วยเป็นเมตร
Packet Size	ขนาดของแพ็กเกจข้อมูล
Data interval	ความถี่ในการส่งข้อมูล
Connection	จำนวนการเชื่อมต่อในระบบเครือข่าย
Initial Energy	การกำหนดค่าพลังงานเริ่มต้น มีหน่วยเป็นจูล
Transmit power (Tx) Receiving power (Rx)	กำหนดค่าการใช้พลังงานในการรับและส่งข้อมูลมีหน่วยเป็นวัตต์
Simulation time	ระยะเวลาในการจำลองการทำงานมีหน่วยเป็นวินาที

ผลลัพธ์ที่ได้จากจำลองการทำงานผ่านโปรแกรม อยู่ในรูปแบบของไฟล์ (.tr) ซึ่งบันทึกเหตุการณ์ที่เกิดขึ้นในระบบเครือข่ายทั้งหมดระหว่างการจำลอง ตั้งแต่เริ่มต้นจนกระทั่งสิ้นสุดเวลาการจำลองการทำงาน ซึ่งข้อมูลจากไฟล์ (.tr) ยังไม่สามารถรู้ถึงค่าประสิทธิภาพการทำงานของโพรโทคอลที่จำลองได้โดยตรง ต้องมีการใช้สคริปต์ในการอ่านค่าและแปรผลลัพธ์ออกมาให้อยู่ในรูปแบบทางสถิติแล้วจึงแสดงผลลัพธ์เป็นตาราง หรือกราฟที่สามารถเข้าใจได้

2.7 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่ผ่านมาในเครือข่ายไร้สายที่เน้นศึกษาเกี่ยวกับโพรโทคอลค้นหาเส้นทางจำนวนมากและมีการศึกษาในหลายลักษณะ ในปี 2002 Zou, Ramamurthy และ Magliveras ได้ทำวิจัยเพื่อแบ่งกลุ่มโพรโทคอลตามคุณลักษณะที่มีความสัมพันธ์และแบ่งตามวิธีการค้นหาเส้นทาง และในปี 2008 Narendra Singh Yadav และ R.P.Yadav ได้ทำวิจัยเพื่อเปรียบเทียบโพรโทคอลค้นหาเส้นทางระหว่างโพรโทคอลในกลุ่ม On-Demand และกลุ่ม Table-Driven โดยใช้โพรโทคอล AODV และ DSDV เป็นตัวแทนสำหรับโพรโทคอลแต่ละกลุ่ม

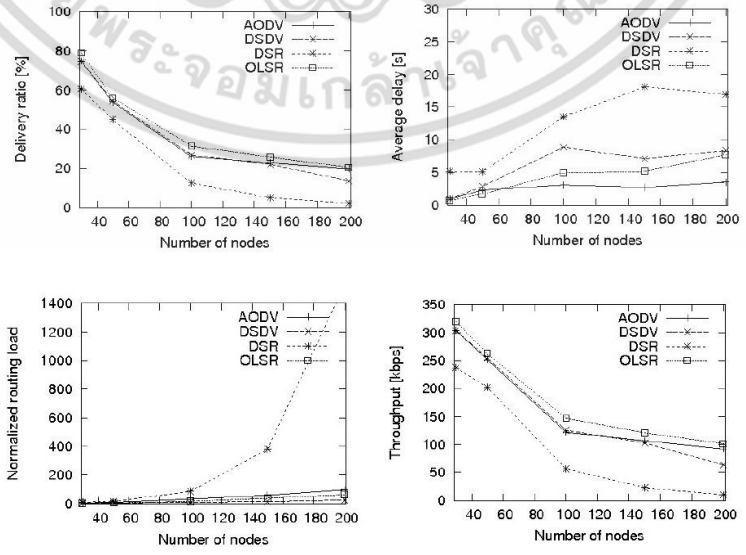
2.7.1 งานวิจัยของ Zakrzewska, Koszałka และ Poźniak-Koszałka

งานวิจัยของ Zakrzewska, Koszałka และ Poźniak-Koszałka ในปี 2008 [10] ได้เปรียบเทียบประสิทธิภาพของโพรโทคอล DSDV, OLSR, AODV และ DSR โดยทดสอบกับโปรแกรมจำลองการทำงานของเครือข่าย NS2 เช่นกัน แต่ทดสอบกับเครือข่ายเมชไร้สาย โดยมีระยะการส่งข้อมูล (Transmission Range)อยู่ที่ 250 m และจำลองเป็นระยะเวลา 900 วินาที ในมาตรฐาน 802.11b ได้แบ่งการทดสอบออกเป็น 3 แบบ คือ

- แบบแรก ทดสอบโดยเพิ่มจำนวนโหนดที่มีในระบบเริ่มจากจำนวนโหนดตั้งแต่ 40-200 โหนด มีความเร็วในการส่งที่ 10 m/s และมีจำนวน data flow ที่ 30 flow
- แบบที่สอง คือ ทดสอบความเร็วในการส่งข้อมูลในหลายๆระดับตั้งแต่ 0-20 m/s มีจำนวนโหนด 50 โหนด และจำนวน data flow ที่ 30 flow
- แบบที่สาม คือ ทดสอบจำนวน data flow ตั้งแต่ 10-50 flow มีจำนวนโหนด 50 โหนด และความเร็วในการส่ง 10 m/s

<u>ผลการทดสอบแบบแรก</u>

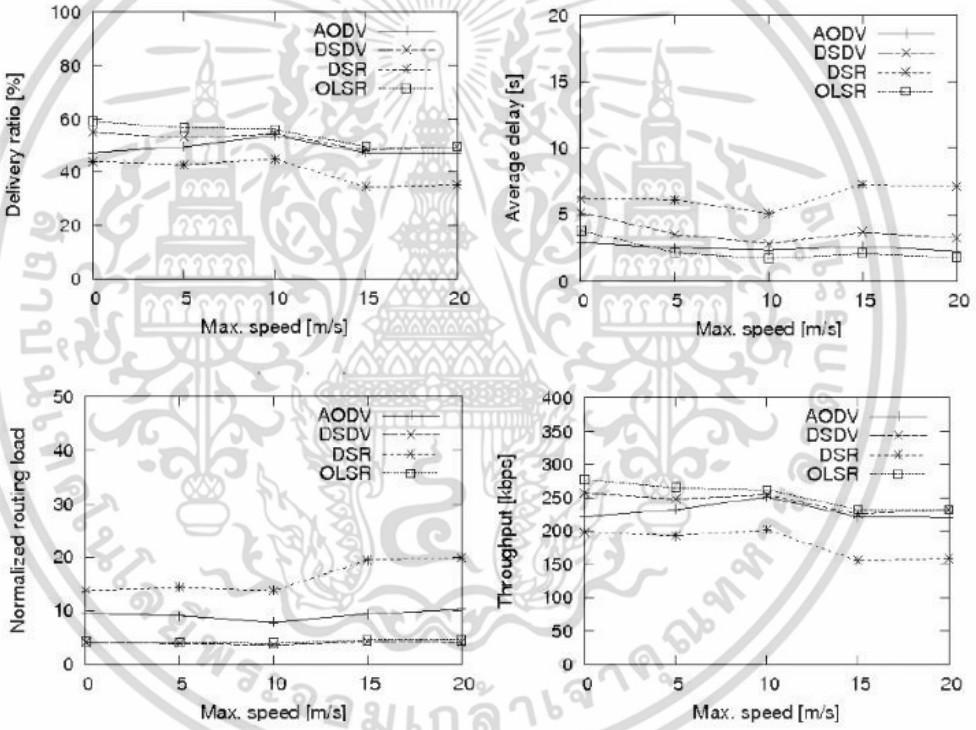
- อัตราการส่งข้อมูลสำเร็จ (PDR) ของโพรโทคอล AODV และ DSDV มีค่ามากที่สุด
- AODV มีค่าเฉลี่ยของ End to End delay ที่น้อยที่สุด ถึงแม้จะมีการเพิ่มขนาดของเครือข่าย
- Routing load ของ DSR มีค่าสูงมากๆ เมื่อมีจำนวนโหนดที่ทดสอบในเครือข่ายเพิ่มจนมี 200 โหนด แต่สำหรับ AODV กราฟจะเป็นเส้นตรง แม้จะมีขนาดของเครือข่ายที่ใหญ่ขึ้น การที่ DSR มีค่า Routing load ที่เพิ่มขึ้นตั้งแต่มีจำนวนโหนดตั้งแต่ 50 โหนดขึ้นไป แสดงว่า DSR ไม่เหมาะสมกับเครือข่ายเมชไร้สายที่มีการขยายตัว เหมาะสำหรับเครือข่ายขนาดเล็กที่มี traffic load ที่ต่ำ
-



รูปที่ 2.18 แสดงประสิทธิภาพเมื่อมีขนาดของเครือข่ายที่แตกต่างกัน

ผลการทดสอบแบบที่สอง

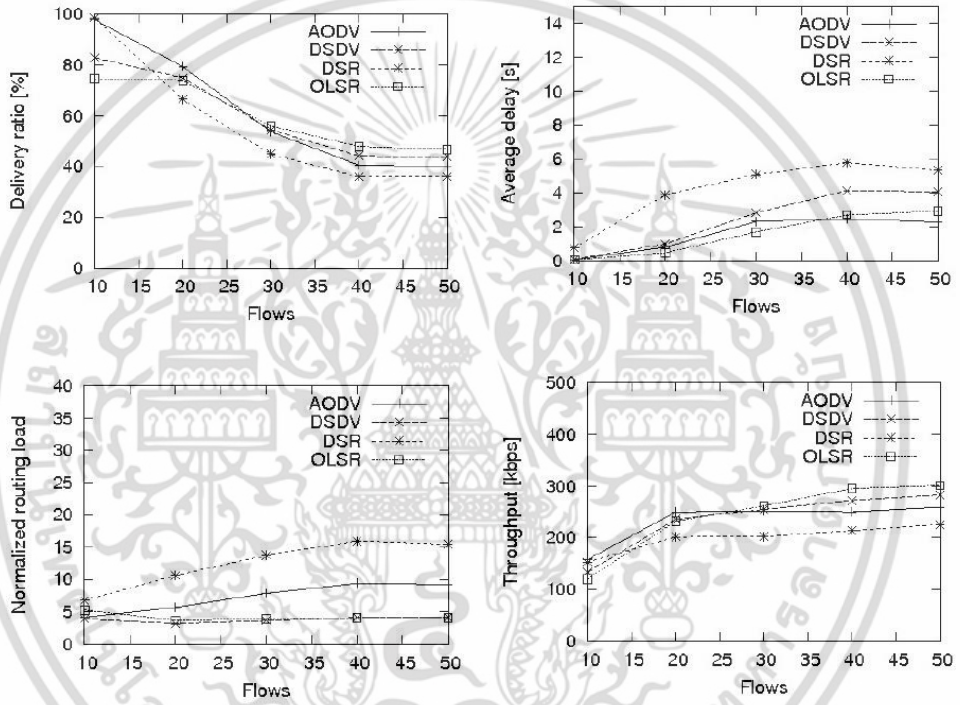
- อัตราการส่งข้อมูลสำเร็จ (PDR) เมื่อมีความเร็วน้อยกว่า 10 m/s ไม่แตกต่างกันมาก เมื่อมีความเร็วมากกว่า 10 m/s AODV ก็ยังมีประสิทธิภาพดี แต่ OLSR มีค่า PDR มากที่สุด และ DSR ก็ยังคงมีผลลัพธ์ที่ไม่ดีอยู่
- OLSR มีค่า End to End delay น้อยที่สุด
- DSDV และ OLSR มี routing overhead น้อยทั้งคู่ เพราะเป็นโพโตคอลแบบ proactive และมีการปรับปรุงเส้นทางเป็นระยะๆ
- เนื่องจาก DSR มีอัตราการส่งข้อมูลสำเร็จ (PDR)ที่ต่ำ ทำให้มีค่าเฉลี่ยของ throughput น้อยกว่าโพรโทคอลตัวอื่นๆ



รูปที่ 2.19 แสดงประสิทธิภาพเมื่อมีการเคลื่อนที่ของโหนดที่แตกต่างกัน

ผลการทดสอบแบบที่สาม

- AODV และ DSR มีอัตราการส่งข้อมูลสำเร็จถึง 80% เมื่อมีตั้งแต่ data flow 0-20 flow
- AODV และ OLSR มีค่าดีเลย์น้อยที่สุดในการทดลองแบบนี้
- DSDV และ OLSR มี routing overhead น้อยเหมือนการทดลองแบบที่สอง
- ในช่วงแรกค่า throughtput มีค่าเพิ่มขึ้นและกลายเป็นเส้นตรงตั้งแต่ data flow เป็น 20 และมีค่า throughtput 250 kbps ซึ่งเป็นจุดที่ DSR มีค่า throughtput ที่คงที่ แต่โพรโทคอลตัวอื่นๆยังคงมีค่าเพิ่มขึ้น ซึ่ง OLSR มีค่าสูงถึง 300 kbps



รูปที่ 2.20 แสดงประสิทธิภาพเมื่อมีปริมาณการสื่อสารที่แตกต่างกัน

2.7.2 งานวิจัยของ Feng และคณะ

งานวิจัยของ Feng และคณะ ในปี 2009 [11] ได้นำเสนอการเปรียบเทียบโดยวัดประสิทธิภาพของโพรโทคอลในการค้นหาเส้นทางในเครือข่าย ad hoc โดยใช้โพรโทคอลที่นำมาเปรียบเทียบ คือ AODV, DSR, DSDV และ OLSR และนำโพรโทคอลเหล่านี้มาทดสอบกับโปรแกรมจำลองการทำงานของเครือข่าย NS2 มีค่าพารามิเตอร์สำหรับการจำลองเครือข่ายดังรูปที่ 2.18

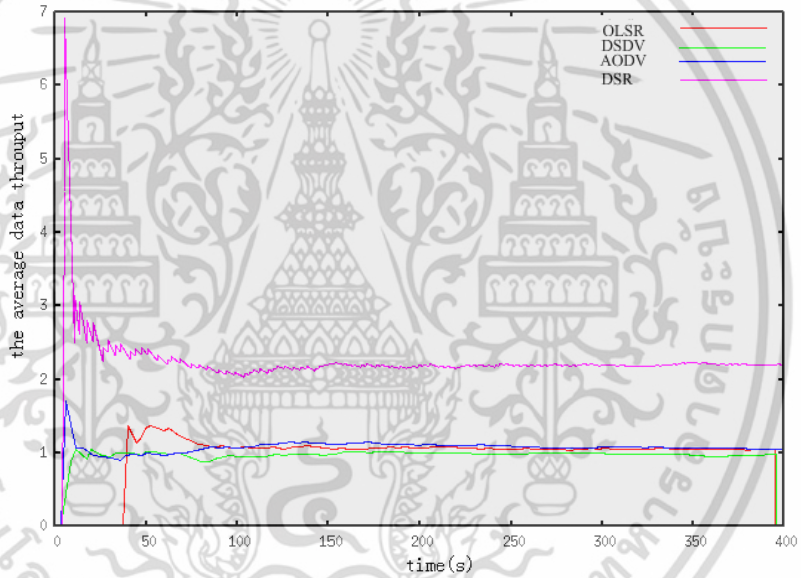
SIMULATION PARAMETERS

Parameter name	Value
Terrain Range	700m×700m
Number of nodes	10,21 and 50
Simulation time	400 seconds
Hello period	2 seconds
TC period	5 seconds
Traffic type	CBR
Data payload	512 bytes/packet
Average node degree	1-2

รูปที่ 2.21 แสดงค่าพารามิเตอร์ในโปรแกรมจำลองเครือข่าย

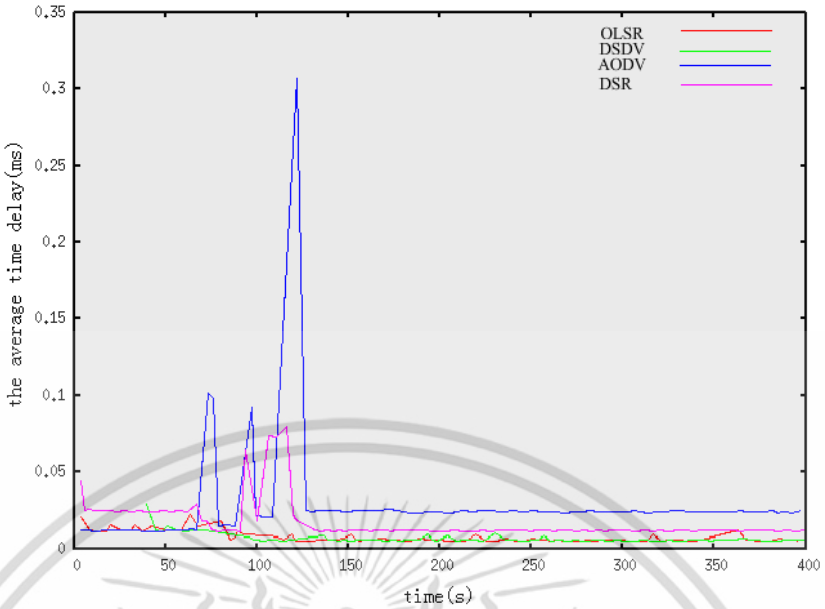
ผลการทดลองที่ได้คือ

- จากรูปที่ 2.21 จะเห็นว่าโพรโทคอล DSR จะมีค่า Throughput ที่ดีที่สุด



รูปที่ 2.22 เปรียบเทียบค่าเฉลี่ยของ Throughput เมื่อมี 50 โหนดในเครือข่าย

แต่ DSDV และ OLSR ก็มีระยะเวลาดีเลย์ที่น้อยว่าในเครือข่ายที่มีขนาดเล็ก



รูปที่ 2.23 เปรียบเทียบค่าเฉลี่ยของ time delay เมื่อมี 50 โหนดในเครือข่าย

Protocol name	10 node	21 node	50 node
OLSR	0.00247	0.00286	0.00772
DSDV	0.00256	0.00312	0.00850
AODV	0.00354	0.00478	0.02887
DSR	0.00273	0.00354	0.01281

รูปที่ 2.24 แสดงค่าเฉลี่ยของ time delay ของโพรโทคอลทั้ง 4 ตัว

- ถ้าสรุปจากปัจจัยที่มีผลทั้งหมดแล้ว ถ้าเครือข่ายต้องการเน้นเรื่องของเวลา DSDV และ OLSR จะมีเหมาะสมมากกว่า แต่ถ้าเป็นเครือข่ายมีการเปลี่ยนแปลงบ่อยและต้องการค่า throughput ที่สูง AODV และ DSR จะมีประสิทธิภาพมากกว่าเช่นกัน

2.7.3 งานวิจัยของ Taksande และ Kulat

งานวิจัยของ Taksande และ Kulat ในปี 2011 [12] ได้นำเสนอการวัดประสิทธิภาพของโพรโทคอล คือ DSDV, DSR และ AODV แต่จะใช้ MAC Layer Protocol คือ IEEE 802.11 ใช้กับ Chain topology และใช้ทดสอบกับเครือข่าย Mobile Ad hoc จะทดสอบกับโปรแกรมจำลองการทำงานของเครือข่าย NS2 มีการเพิ่มจำนวนโหนดเครือข่าย มีค่าพารามิเตอร์สำหรับการจำลองเครือข่ายดังรูปที่ 2.25

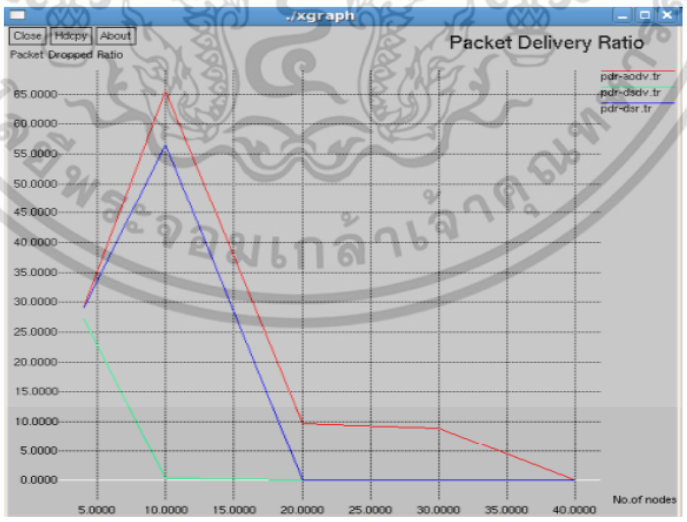
Simulation Parameters	
Simulator	ns-2.34
Protocols	AODV,DSDV,DSR
Simulation duration	200 seconds
Simulation area	2200 m x 500 m
Number of nodes	5,9,25,35,40
Transmission range	250 m
Movement model	Chain topology
MAC Layer Protocol	IEEE 802.11
Maximum speed	50 m/s
Packet rate	4 packets/sec
Traffic type	CBR
Data payload	512 bytes/packet

รูปที่ 2.25 แสดงค่าพารามิเตอร์ในโปรแกรมจำลองเครือข่าย

ผลที่ได้จากการทดสอบดังตารางนี้

ตารางที่ 2.6 แสดงโพรโทคอลที่เหมาะสมที่สุดในแต่ละค่าวัดประสิทธิภาพ

Parameters	Best Protocol
Generated Packets	DSR
Received Packets Vs. No. of nodes	DSR up to 10 nodes AODV for more than 10 nodes
Total Dropped Packets Vs. No. of nodes	DSR
Packet Delivery Ratio Vs. No. of nodes	AODV
Average End To End Delay Vs. No. of nodes	AODV up to 10 nodes DSR for more than 10 nodes

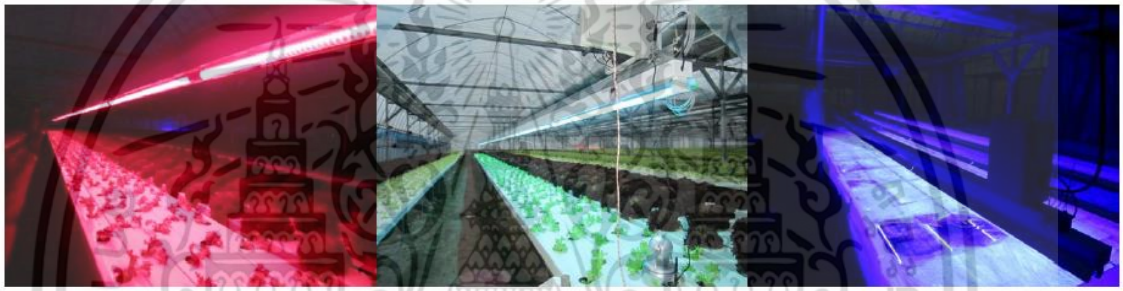


Packet Delivery Ratio Vs. numbers of nodes

รูปที่ 2.26 เปรียบเทียบค่าเฉลี่ยของ Packet Delivery Ratio เมื่อมีโหนดตั้งแต่ 5-40 ในเครือข่าย

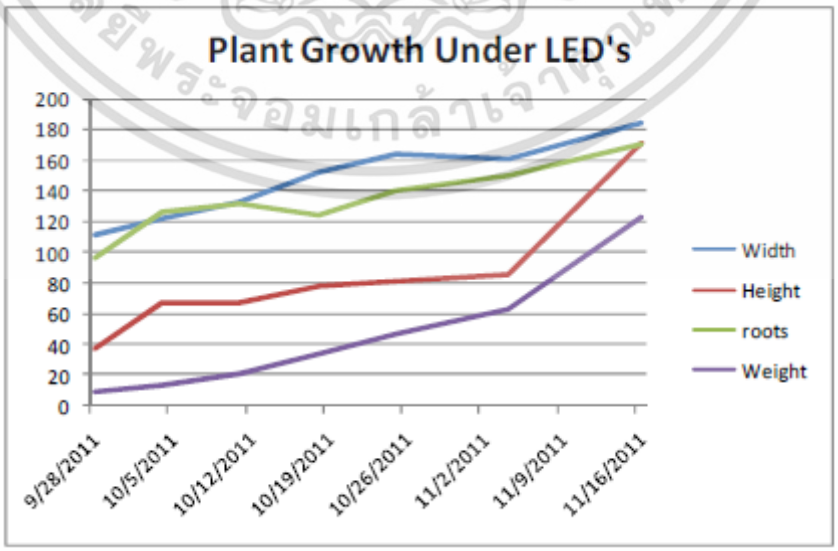
2.7.4 งานวิจัยของ Ijaz และ คณะ

งานวิจัย Ijaz และ คณะ ในปี 2012 [13] ได้เสนอระบบควบคุมไฟ LED ในโรงงานเพาะปลูกซึ่งมีการติดตั้งเซ็นเซอร์เพื่อติดตามสภาพแวดล้อม เช่น อุณหภูมิ ความชื้นในอากาศ ตรวจค่าของสี LED ความเข้มแสงและตรวจวัด CO_2 เป็นต้น งานวิจัยนี้ได้ใช้ ZigBee และเครือข่ายเมชไร้สาย สำหรับใช้เก็บข้อมูลที่ได้จากเซ็นเซอร์และใช้โพรโทคอลค้นหาเส้นทาง AODV เครือข่ายนี้จะประกอบไปด้วย gateway node 1 ตัว ทำหน้าที่ในการจัดการด้านเครือข่ายเชื่อมต่อเครือข่ายภายนอกด้วย TCP/IP ซึ่งจะไปแสดงผลการตรวจวัดที่โรงงาน, sensor node ตรวจวัดค่าต่างๆ, dimming Node ใช้ตรวจสอบข้อบกพร่องของไฟ LED และเซ็นเซอร์วัดอุณหภูมิ และ meter node ใช้วัดค่าการใช้พลังงาน



รูปที่ 2.27 ระบบต้นแบบที่ติดตั้งในโรงงานเพาะปลูกที่ใช้ RGB LED

งานวิจัยนี้ได้สร้างระบบต้นแบบและติดตั้งในพื้นที่ทดสอบเล็กๆ และติดตามผลการเจริญเติบโตของพืช หลังจากที่นำระบบต้นแบบนี้มาใช้ ได้ดังภาพประกอบด้านล่างนี้



รูปที่ 2.28 แสดงค่าเฉลี่ยการเจริญเติบโตของพืชในแต่ละส่วน หลังจากใช้ระบบต้นแบบ RGB LED

ในงานวิจัยที่กล่าวมาข้างต้นพบว่างานวิจัยเหล่านี้ได้นำเสนอวิธีการเปรียบเทียบวัดประสิทธิภาพและความน่าเชื่อถือของโพรโทคอลที่ใช้ค้นหาเส้นทางในเครือข่ายหลายๆแบบทั้งเครือข่าย ad hoc, เครือข่าย Mobile Ad hoc, เครือข่ายเซ็นเซอร์ไร้สายและเครือข่ายเมชไร้สาย ในโปรแกรมจำลองการทำงานของเครือข่าย โดยโพรโทคอลที่นำมาทดสอบมี AODV, DSDV, DSR และ OLSR และมีงานวิจัยที่ได้นำเครือข่ายเมชไร้สายไปใช้ในด้านเกษตรกรรม ใช้ในการควบคุมการทำงานของไฟ LED ในโรงงานเพาะปลูกมีการใช้ ZigBee ช่วยในการจัดการด้านพลังงานทำให้เซ็นเซอร์โหนดมีอายุการใช้งานมากขึ้น

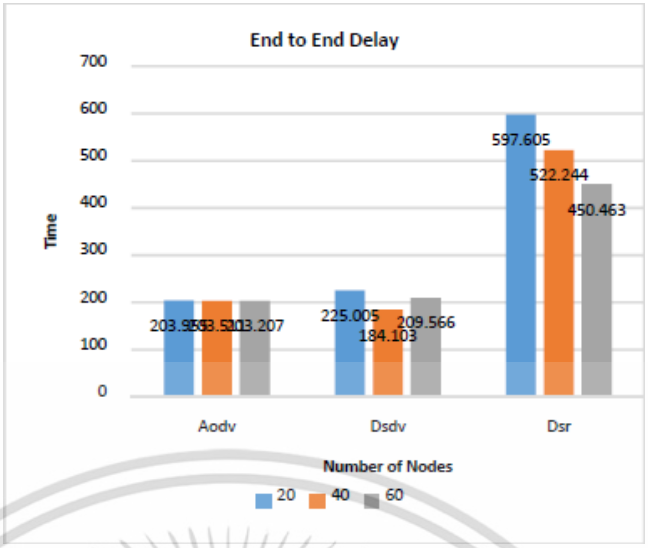
2.7.5 งานวิจัยของ Arya และ Singh

งานวิจัยของ Arya และ Singh ในปี 2014 [14] ได้เปรียบเทียบวัดประสิทธิภาพและความน่าเชื่อถือของโพรโทคอลค้นหาเส้นทาง AODV, DSDV และ DSR โดยใช้โปรแกรมจำลองการทำงานของเครือข่าย NS2 และใช้ทดสอบกับเครือข่ายเซนเซอร์ไร้สาย มีค่าพารามิเตอร์สำหรับการจำลองเครือข่ายดังรูปที่ 2.29

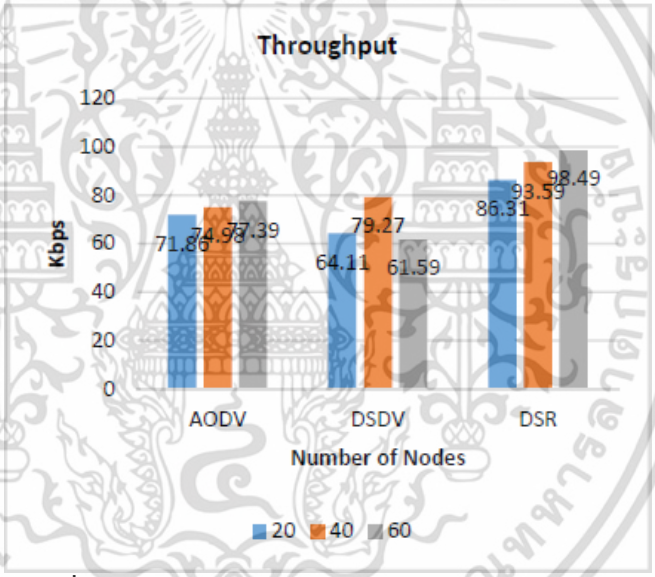
Parameter Type	Parameter Value
Protocols	AODV,DSDV,DSR
Simulation Time	150ms
Number of Nodes	20,40,60
Packet Type	TCP Packet
Queue Type	Priority Queue
Environment size	1000m*500m
Traffic Type	Constant Bit Rate
Platform	Ubuntu
Simulator	NS2

รูปที่ 2.29 แสดงค่าพารามิเตอร์ในโปรแกรมจำลองเครือข่าย

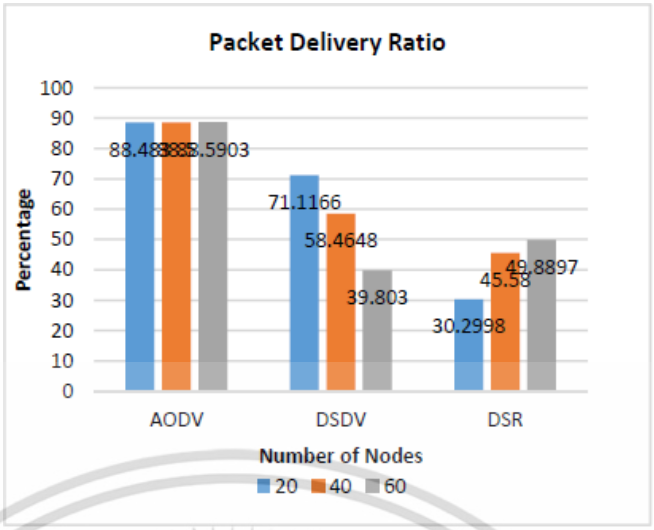
ผลการทดลองที่ได้คือ ค่า throughput โพรโทคอล DSR ทำได้ดีที่สุด แต่ AODV มีประสิทธิภาพมากกว่าสองตัวที่เหลือในเรื่องของระยะเวลาดีเลย์และความน่าเชื่อถือในอัตราการส่งแพ็กเก็ตที่ส่งจากต้นทางไปยังปลายทางได้สำเร็จมากที่สุด (ดังรูปที่ 2.30)



รูปที่ **2.30** ผลการทดสอบเปรียบเทียบค่า End to End Delay



รูปที่ 2.31 ผลการทดสอบเปรียบเทียบค่า Throughput



รูปที่ 2.32 ผลการทดสอบเปรียบเทียบค่า Packet Delivery Ratio

ในงานวิจัยนี้เลือกใช้โพรโทคอลการหาเส้นทางแบบ AODV เนื่องจาก AODV มีหลักการทำงานแบบ Reactive หรือ On-Demand คือ จะค้นหาเส้นทางก็ต่อเมื่อมีโหนดที่ต้องการจะส่งข้อมูลในเส้นทางนั้น โดยไม่จำเป็นต้องมีปรับปรุงเส้นทางที่ปลายทางที่ยังไม่ได้มีการใช้งานในขณะนั้น และในขณะที่การสื่อสารยังคงมีการทำงานอยู่ ช่วยให้มีการใช้ Bandwidth ได้อย่างมีประสิทธิภาพ สามรถเปลี่ยนเส้นทางได้อย่างรวดเร็ว และใช้การประมวลผลและหน่วยความจำน้อย และเหมาะสำหรับสถานการณ์ที่มีการเปลี่ยนแปลงหรือการเคลื่อนที่เสมอๆ และจากงานวิจัยข้างต้นจะเห็นว่าโพรโทคอล AODV มีประสิทธิภาพต่างๆโดยรวมที่อยู่ในเกณฑ์ดี ผู้ทำงานวิจัยจึงได้เลือกโพรโทคอล AODV มาใช้พัฒนาเพื่อให้สามารถนำมาใช้กับเครือข่ายเมชไร้สายได้อย่างเหมาะสมและมีประสิทธิภาพมากขึ้นกับการนำไปใช้ในด้านเกษตรกรรม

บทที่ 3
วิธีการดำเนินงานวิจัย

บทนี้เป็นการนำเสนอรายละเอียด แนวทางการพัฒนา ปรับปรุงและกระบวนการทำงานของโพรโทคอลดังต่อไปนี้

3.1 แนวทางการปรับปรุงโพรโทคอล

โดยทั่วไปโหนดในเครือข่ายไร้สายจะมีความสามารถทั้งรับส่งข้อมูล และแลกเปลี่ยนข้อมูลกับโหนดข้างเคียงได้ แต่โหนดที่จะนำมาใช้สร้างเครือข่ายไร้สายสำหรับการเกษตรนี้ โหนดไม่จำเป็นต้องมีการติดต่อสื่อสารและแลกเปลี่ยนข้อมูลกันมากมาย แค่ให้ค่าตรวจวัดที่ได้จากเซ็นเซอร์ของโหนดที่สามารถส่งข้อมูลต่อไปยังโหนดข้างเคียงและส่งข้อมูลต่อกันไปจนไปถึง gateway เพื่อส่งข้อมูลออกไปยังเครือข่ายอื่นๆ บนอินเทอร์เน็ตหรือเก็บข้อมูลลงฐานข้อมูลเท่านั้น

ดังนั้นจึงต้องมีการปรับปรุงพัฒนาโพรโทคอล เพื่อให้สามารถนำมาใช้งานได้เหมาะสมกับเครือข่ายที่ต้องการสร้างดังนี้

1. หลักการทำงานของโพรโทคอล AODV จะเห็นว่าเมื่อโหนดมีความต้องการที่จะส่งข้อมูลไปยังโหนดปลายทาง แต่โหนดต้นทางไม่มีเส้นทาง ทำให้โหนดจะทำการส่งข้อความค้นหาเส้นทางแบบแพร่กระจาย (broadcast) ออกไปยังโหนดข้างเคียงโดยโหนดที่ได้รับข้อความก็จะทำแบบเดิมซ้ำๆ จนกว่าโหนดที่ได้รับข้อความนั้นจะเป็นโหนดปลายทางหรือมีเส้นทางของโหนดปลายทาง จากที่กล่าวมาข้างต้น ทำให้มีการส่งข้อความในเครือข่ายที่มาจากการส่งข้อความแบบแพร่กระจายซ้ำๆมากเกินไป และการค้นหายังกระจายไปทั่วทั้งเครือข่าย โดยที่เครือข่ายที่เราต้องการนำไปใช้มีโหนดปลายทางแค่โหนดเดียว คือ gateway ดังนั้นผู้ทำงานวิจัยจึงคิดวิธีปรับปรุงการค้นหาให้มีขอบเขตในการค้นหาที่น้อยลงและลดข้อความที่ส่งในเครือข่าย แต่ยังคงสามารถค้นหาเส้นทางและส่งข้อมูลได้อย่างมีประสิทธิภาพมากขึ้น
2. จากที่กล่าวมาในข้อแรก ทำให้โพรโทคอล AODV มีเส้นทางในตารางจากการส่งข้อความค้นหาแบบแพร่กระจายอยู่มาก แต่มีการตอบกลับจากโหนดปลายทางแค่เส้นทางเดียวและใช้แค่เส้นทางนั้น ทำให้โหนดมีการเก็บเส้นทางที่ไม่จำเป็นหรือไม่ได้ใช้งานนั้นตลอดเวลา ดังนั้นโพรโทคอลใหม่ โหนดจะเก็บเส้นทางของโหนดข้างเคียงทั้งหมดและโหนดปลายทาง หรือ gateway เท่านั้น
3. โพรโทคอล AODV มีการเก็บเส้นทางของโหนดปลายทางแค่เส้นทางเดียวที่สั้นที่สุดเท่านั้น เพราะมีการเก็บเส้นทางจากโหนดอื่นๆ ที่ส่งข้อความค้นหาผ่านโหนดนั้น ๆ แต่เครือข่ายที่ต้องการนำไปใช้ มีโหนดปลายทางเพียงโหนดเดียว ดังนั้นจึงให้มีการเก็บเส้นทางที่สั้นที่สุดได้มากกว่า 1 เส้นทางของ gateway เพื่อเป็นการลดระยะเวลาในการค้นหาเส้นทาง ในกรณี

เส้นทางที่เคยใช้งานเกิดหมดอายุการใช้งานหรือเกิดความเสียหาย แต่จะไม่เก็บมากเกินกว่าจำนวนโหนดข้างเคียงที่สามารถส่งข้อมูลได้ เช่น ถ้ามีโหนดข้างเคียงอยู่ 4 โหนด ตารางเส้นทางของ gateway ของโหนดนั้นจะมีได้มากที่สุดได้ 4 เส้นทาง เพื่อเป็นทางเลือกในการส่งข้อมูลกรณีเส้นทางเกิดเสียหาย แต่ก็ไม่ให้มีการเก็บเส้นทางที่มากเกินไปด้วยเช่นกัน

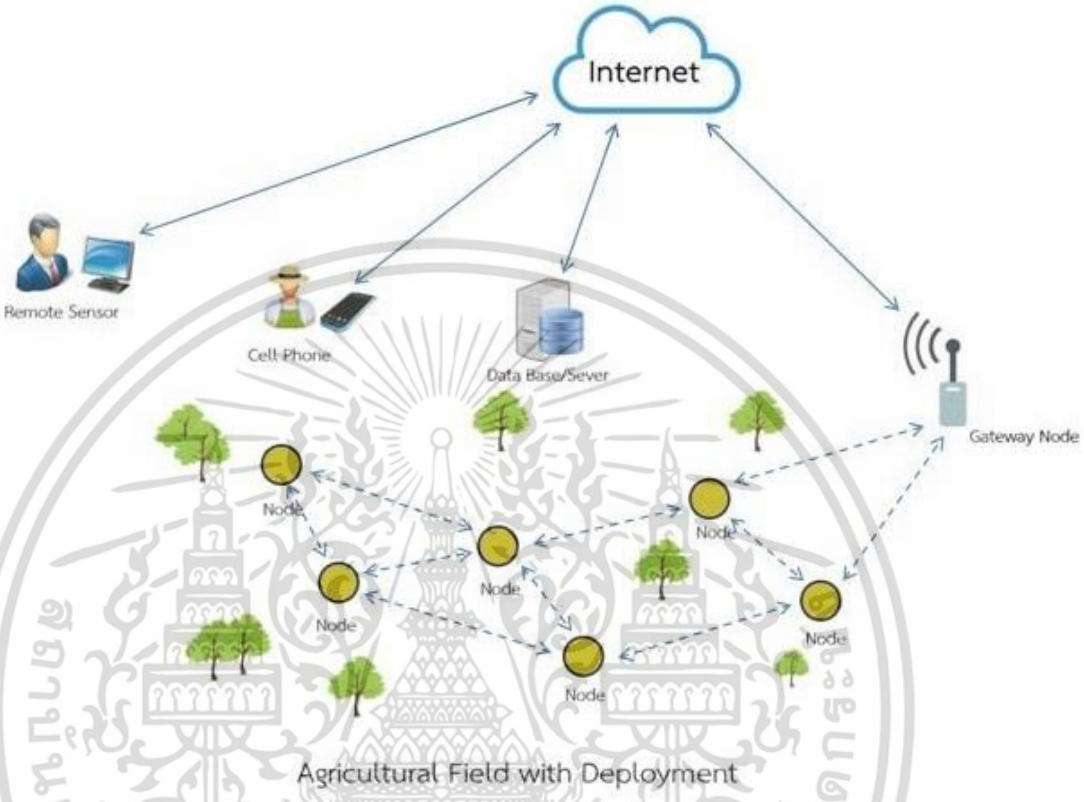
4. การลดขอบเขตการค้นหาและตารางเส้นทางของโพรโทคอลใหม่ ทำให้เมื่อมีโหนดหรือเส้นทางที่เสียหาย ทำให้โหนดไม่ต้องส่งข้อความ RERR แบบแพร่กระจายเพื่อแจ้งให้ทุกโหนดทราบว่ามีโหนดเสียหายหรือเส้นทางเสียหายแบบโพโทรคอล AODV เพียงแค่ส่งข้อความ RERR ไปยัง gateway จากเส้นทางที่เหลืออยู่ของโหนดทั่วไป ทำให้สามารถส่งข้อความ RERR ไปยัง gateway ได้ เพื่อซ่อมแซมเส้นทางให้กับตารางของ gateway ในเส้นทางที่ได้รับผลกระทบ

3.2 ขอบเขตและข้อจำกัดของการใช้งานโพรโทคอล HAC-SF ในเครือข่ายไร้สาย

ระบบเครือข่ายไร้สาย ประกอบไปด้วยโหนด (Node) ซึ่งเป็นอุปกรณ์ที่สามารถรับส่งสัญญาณและทำหน้าที่หาเส้นทางและส่งต่อข้อมูลในเครือข่ายได้ และ sensor ทำหน้าที่รับข้อมูลสภาพแวดล้อมในพื้นที่ที่สนใจ โดยโหนดหลายตัวจะมีตำแหน่งการวางที่กระจายเต็มพื้นที่และให้มีระยะห่างมากพอที่จะทำการติดต่อสื่อสารและส่งข้อมูลกันได้ โหนดจะมีการติดต่อสื่อสารในเครือข่ายผ่านทางคลื่นวิทยุ (Radio Frequency) ผ่าน ISM bands เป็นคลื่นความถี่สาธารณะที่ใช้สำหรับอุตสาหกรรม วิทยาศาสตร์และการแพทย์ (2.4–2.5 GHz) ถ้าโหนดตัวไหนอยู่ในระยะที่สามารถส่งข้อมูลกันได้ก็จะทำการสร้างการเชื่อมต่อเพื่อให้สามารถติดต่อสื่อสารกันได้จนเกิดเป็นเครือข่าย และถ้ามีโหนดใหม่เข้ามาในเครือข่าย โหนดนั้นต้องสามารถทำการเชื่อมต่อกับเครือข่ายได้อย่างอัตโนมัติ โดยขอบเขตและข้อจำกัดของการใช้งานโพรโทคอล HAC-SF ดังนี้

1. โพรโทคอลนี้จะสามารถใช้งานได้มีประสิทธิภาพในเครือข่ายที่มี gateway เพียงตัวเดียวเป็นโหนดศูนย์กลาง ทำหน้าที่เชื่อมต่อระหว่างโหนดภายในเครือข่ายไปยังเครือข่ายภายนอกบนอินเตอร์เน็ต เพื่อให้สามารถส่งข้อมูลต่าง ๆ ไปจัดเก็บไว้ที่ระบบฐานข้อมูล (Database) และการประมวลผลบน server หรืออัพเดตข้อมูลส่งมายังโทรศัพท์ให้กับผู้ใช้รู้ความเคลื่อนไว้ของระบบ
2. โพรโทคอลนี้จะมีข้อจำกัดคือ โหนดที่เป็นศูนย์กลางจะต้องมีประสิทธิภาพในการทำงานที่มากกว่าโหนดทั่วไปเป็นพิเศษ เพราะต้องทำหน้าที่ทั้งรับข้อความค้นหาเส้นทาง รับค่าวัดสภาพแวดล้อม และส่งข้อมูลที่ใช้ในการควบคุมการทำงานของโหนดทั้งหมดในเครือข่าย ทำให้ค่าใช้จ่ายของอุปกรณ์ที่เป็นโหนดศูนย์กลางซึ่งมีหน่วยประมวลผลและหน่วยความจำที่มากกว่าโหนดทั่วไป

3.3 การพัฒนาโพรโทคอล



รูปที่ 3.1 สถาปัตยกรรม โพรโทคอล HAC-SF

3.3.1 ที่มาและลักษณะของ โพรโทคอล HAC-SF

เป็นโพรโทคอลการค้นหาเส้นทางพัฒนามาจากโพรโทคอล Ad hoc On-demand Distance Vector Routing (AODV) นำมาใช้กับเครือข่ายไร้สายแบบ ad hoc กึ่ง centralized ที่มีรูปแบบการสื่อสารที่มีโหนดศูนย์กลาง หรือ gateway ทำหน้าที่รับข้อมูลจากโหนดทั่วไปในเครือข่ายและสามารถควบคุมการทำงานของโหนดทั่วไปได้ โดยที่โหนดแต่ละตัวสามารถทำหน้าที่เป็นได้ทั้งโหนดรับ โหนดส่ง และโหนดเชื่อมต่อได้

3.3.2 ลักษณะของโพรโทคอล HAC-SF

โพรโทคอล HAC-SF เป็นโพรโทคอลการจัดการเส้นทางในเครือข่ายไร้สาย โดยที่เส้นทางอาจมีการติดต่อสื่อการหลายช่วง โพรโทคอล HAC-SF เส้นทางจะถูกสร้างขึ้นเฉพาะเมื่อมีการร้องขอเส้นทางเท่านั้นและไม่จำเป็นต้องทำการปรับปรุงข้อมูลเส้นทางไปยังโหนดปลายทางที่ยังไม่ใช้งานในขณะนั้น และในขณะการสื่อสารดำเนินอยู่ โดยเส้นทางยังคงสามารถทำงานได้ เพื่อให้การส่งข้อมูลนั้นเป็นไปอย่างถูกต้อง และโพรโทคอลนี้มีการส่ง

ข้อความควบคุม (Control Messages) เป็นช่วง ๆเพื่อใช้ในการกำหนดเส้นทางหรือปรับปรุงข้อมูลเส้นทางเหมือนโพโทคอล AODV แต่จะมีข้อความควบคุม โครงสร้างของข้อความควบคุมและกระบวนการทำงานที่เปลี่ยนไปจากเดิม

3.3.3 ตารางเส้นทาง (Routing table)

ตารางเส้นทาง มีหน้าที่ ในการเก็บรายละเอียดเกี่ยวกับเส้นทางที่ใช้ในการสื่อสาร โดยมีโครงสร้างสำหรับเก็บข้อมูลเพื่อตรวจสอบข้อความค้นหาเส้นทางต่าง ๆ โดยตารางที่ 3.1 เป็นตารางเส้นทางของโหนดทั่วไป ซึ่งจะเหมือนโพรโทคอล AODV ดังเดิม

ตารางที่ 3.1 ความหมายของส่วนของข้อมูลในตารางเก็บค่าเส้นทางของโหนดทั่วไป (Node)

ส่วนของข้อมูล	ความหมาย
Destination IP Address	หมายเลขโหนดของโหนดปลายทาง
Destination Sequence Number	หมายเลขลำดับของโหนดปลายทาง
Valid Destination Sequence Number flag	ตัวแปรที่ใช้บอกว่าข้อมูลเส้นทางนั้น ๆ มีหมายเลขลำดับของโหนดปลายทางถูกต้องหรือไม่
Hop Count	จำนวนโหนดที่จะต้องการส่งข้อมูลไปจนถึงโหนดปลายทาง
Next Hop	หมายเลขโหนดถัดไปในเส้นทางการสื่อสาร
Life time	เวลาที่โหนดใช้ในการพิจารณาว่าเส้นทางนั้นยังสามารถใช้งานได้

ตารางที่ 3.2 ความหมายของส่วนของข้อมูลในตารางเก็บค่าเส้นทางของโหนดศูนย์กลาง (gateway)

ส่วนของข้อมูล	ความหมาย
Destination IP Address	หมายเลขโหนดของโหนดปลายทาง
Destination Sequence Number	หมายเลขลำดับของโหนดปลายทาง
Valid Destination Sequence Number flag	ตัวแปรที่ใช้บอกว่าข้อมูลเส้นทางนั้น ๆมีหมายเลขลำดับของโหนดปลายทางถูกต้องหรือไม่
Hop Count	จำนวนโหนดที่จะต้องการส่งข้อมูลไปจนถึงโหนดปลายทาง
Next Hop	หมายเลขโหนดถัดไปในเส้นทางการสื่อสาร
Life time	เวลาที่โหนดใช้ในการพิจารณาว่าเส้นทางนั้นยังสามารถยังใช้งานได้
Path IP Address	หมายเลขโหนดของโหนดที่ข้อความค้นหาเส้นทางส่งผ่านมาทั้งหมด

จากตารางที่ 3.1 ตารางเก็บเส้นทางของโหนดทั่วไปนั้นยังคงรูปแบบเหมือนโพรโทคอล AODV ดังเดิม แต่ในส่วนของตารางเก็บเส้นทางของโหนดปลายทางหรือ gateway มีการเปลี่ยนแปลงเพื่อให้สามารถเก็บเส้นทางจากข้อความ RREQ แบบใหม่ได้ และสามารถส่งข้อความตอบรับกลับไปยังโหนดต้นทางในเครือข่ายได้ ดังตารางที่ 3.2 และรูปที่ 3.2 นี้

Des IP	Des Se	flag	Hop Count	Next hop	Life time	Paths			

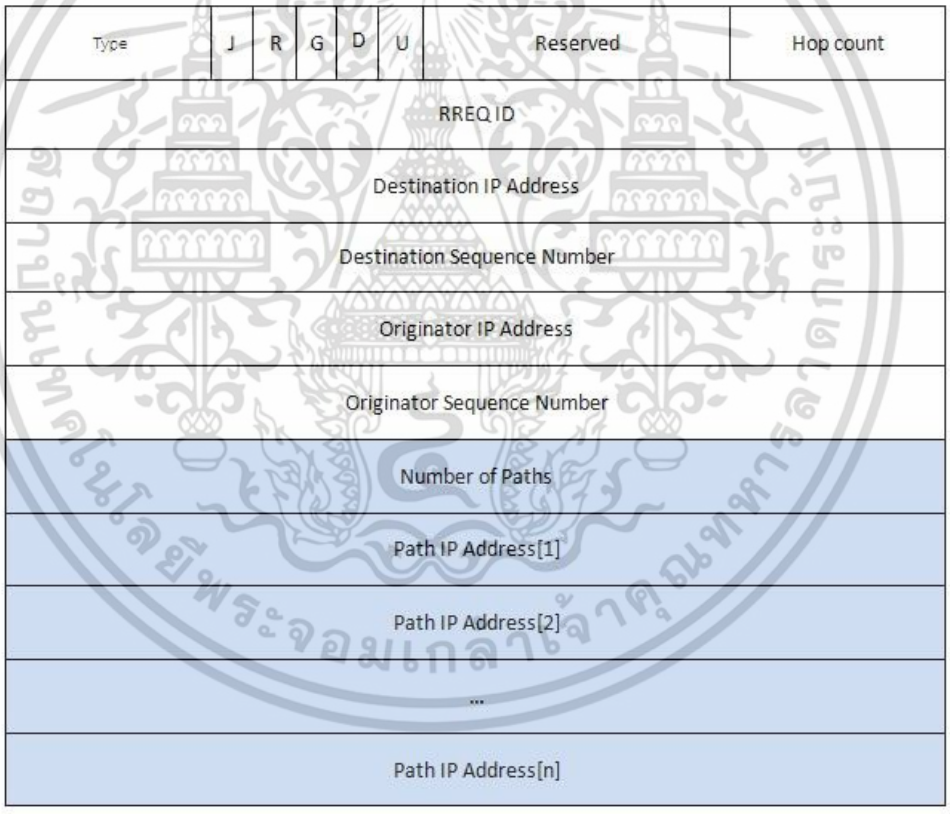
รูปที่ 3.2 ตารางค้นหาเส้นทางของ gateway

จากรูปที่ 3.2 แสดงลักษณะของตารางเก็บเส้นทางของ gateway โดยส่วนที่เพิ่มเติมเข้ามา คือ ส่วนที่เก็บหมายเลขโหนดของเส้นทางทั้งหมด (Path IP Address) ที่ได้รับมาจากข้อความ RREQ แบบใหม่ ซึ่งเป็นส่วนกรอบสีเข้ม

3.3.4 รูปแบบข้อความ (Message format)

คือ ข้อความที่ใช้ในการค้นหาและการจัดการเส้นทางที่โหนดต้องการ แบ่งข้อความได้ 5 ประเภทได้แก่

1) RREQ (Route Request message) คือ ข้อความควบคุมที่ใช้สำหรับการค้นหาเส้นทางไปยังโหนดปลายทาง ในกรณีที่ไม่มีเส้นทางข้อมูลนั้นอยู่ในตารางเส้นทาง โดยโหนดจะเริ่มส่งข้อความ RREQ แบบแพร่กระจายไปยังโหนดข้างเคียง ซึ่งข้อความ RREQ มีขนาดน้อยที่สุด 25 ไบต์ และมีโครงสร้างของข้อความดังรูปที่ 3.3 นี้



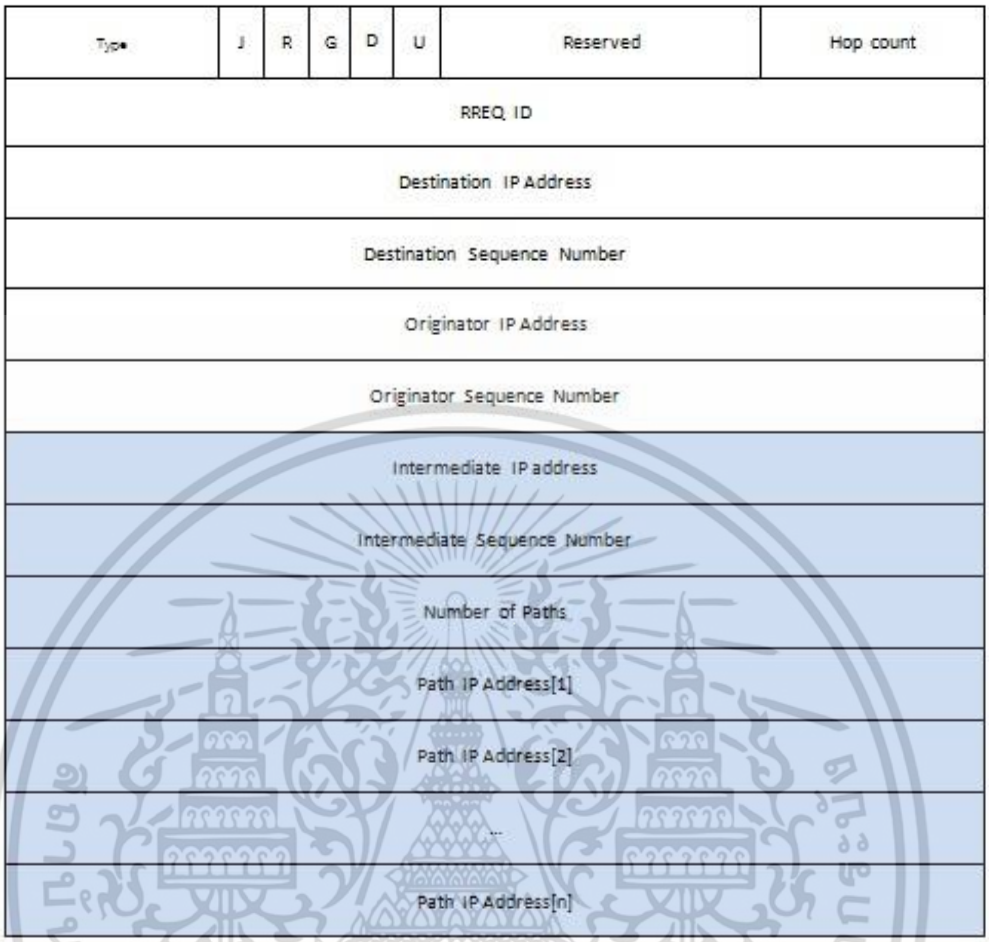
รูปที่ 3.3 ตัวอย่างโครงสร้างของข้อความ RREQ

ตารางที่ 3.3 ความหมายของส่วนของข้อมูลในข้อความ RREQ

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
J	Join Flag ไว้ใช้สำหรับการส่งแบบแพร่กระจาย
R	Repair Flag ไว้ใช้สำหรับการส่งแบบแพร่กระจาย
G	Gratuitous RREP Flags เป็นตัวที่บอกว่าข้อความ RREP ที่ต้องการควรส่งแบบปลายทางเดียวไปยังโหนดปลายทาง
D	Destination only flag เป็นตัวบ่งบอกว่าให้โหนดปลายทางเท่านั้นที่สามารถตอบกลับข้อความ RREQ ได้
U	Unknown Sequence number เป็นตัวที่บอกว่าโหนดปลายทางที่ต้องการค้นหาเส้นทางนั้นไม่ทราบค่าหมายเลขลำดับของโหนดปลายทาง
Reserved	ถูกสงวนไว้ โดยตั้งค่าให้มีค่าเป็น 0
Hop Count	จำนวนโหนดที่ข้อความ RREQ ผ่านมาจากโหนดต้นทางจนถึงโหนดที่ RREQ มาถึง
RREQ ID	หมายเลขลำดับของข้อความ แต่ละชุดที่ทำการ Broadcast ซึ่งจะไม่ซ้ำกันเพื่อระบุความใหม่ของข้อความ RREQ
Destination IP Address	หมายเลขโหนดของโหนดปลายทางที่โหนดต้นทางต้องการส่งข้อมูล
Destination Sequence Number	หมายเลขลำดับล่าสุดของโหนดปลายทางที่เคยได้รับมาก่อนหน้านี้
Originator IP Address	หมายเลขโหนดของโหนดต้นทางที่สร้างและส่งข้อความ RREQ เพื่อหาเส้นทางไปยังโหนดปลายทาง
Originator Sequence Number	หมายเลขลำดับปัจจุบันของโหนดต้นทาง
Number of Paths	จำนวนโหนดของเส้นทางที่ข้อความ RREQ ส่งผ่านมาทั้งหมด (Path IP Address)
Path IP Address	หมายเลขโหนดของเส้นทางที่ข้อความ RREQ ส่งผ่านมาทั้งหมด

จากรูปที่ 3.3 นี้ แสดงลักษณะของข้อความ RREQ ในส่วนที่ยังคงรูปแบบเหมือนเดิม อยู่ในส่วนที่เป็นกรอบสีขาวและส่วนที่เพิ่มข้อมูลเข้าไปใหม่จากรูปแบบข้อความเดิมคือ ส่วนที่เป็นกรอบสีเข้มซึ่งส่วนที่เพิ่มเติมเข้าไปในข้อความมี 2 ส่วน คือ จำนวนโหนดของเส้นทางที่ข้อความ RREQ นี้ได้เก็บบันทึกลงในข้อความ (Number of Paths) อีกส่วนคือ หมายเลขโหนดของเส้นทางที่ข้อความ RREQ เก็บบันทึกทั้งหมด (Path IP Address) โดยส่วนนี้จะมีขนาดเพิ่มขึ้นตามจำนวนโหนดที่ข้อความ RREQ ส่งผ่านมาทั้งหมด ทำให้ข้อความมีขนาดใหญ่ขึ้นตามจำนวนโหนดของการหาเส้นทางนั้นด้วย

2) RREQInter (Route Request Intermediate) คือ ข้อความควบคุมที่ใช้สำหรับการสอบถามเส้นทางไปยังปลายทางในกรณีที่โหนดได้รับข้อความ RREQ นั้น มีเส้นทางข้อมูลของโหนดปลายทางอยู่ในตารางเส้นทาง โหนดจะทำการสร้างข้อความ RREQInter แทน RREQ และส่งข้อความไปยังโหนดปลายทางตามเส้นทางในตาราง โดยเปลี่ยนการส่งข้อความเป็นแบบปลายทางเดียว (Unicast) แทนการส่งแบบแพร่กระจาย(Broadcast) ของข้อความ RREQ โดยโหนดส่งข้อความ RREQInter จะเรียกว่า โหนดตัวแทน หรือ Intermediate Node ซึ่งเป็นโหนดที่มีเส้นทางของปลายทางที่ยังสามารถใช้งานได้ แล้วได้รับข้อความ RREQ จากโหนดอื่นที่ต้องการส่งข้อความไปยังโหนดปลายทาง โดยข้อความ RREQInter ซึ่งข้อความ RREQInter มีขนาดน้อยที่สุด 33 ไบต์ และมีโครงสร้างของข้อความดังรูปที่ 3.4 นี้ โดยส่วนที่ไฮไลท์ด้วยสีฟ้าคือส่วนของ HAC-SF ที่เพิ่มจากของเดิม



รูปที่ 3.4 ตัวอย่างโครงสร้างของข้อความ RREQInter

ตารางที่ 3.4 ความหมายของส่วนของข้อมูลในข้อความ RREQInter

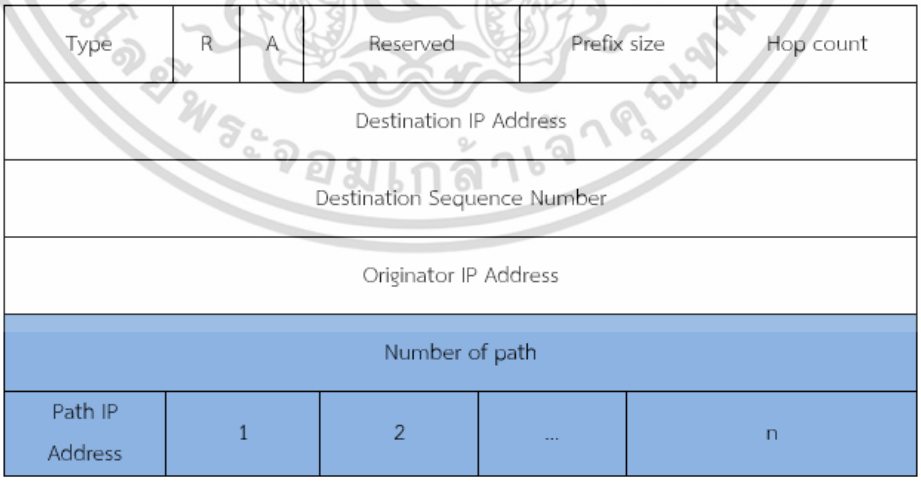
ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
J	Join Flag ไว้ใช้สำหรับการส่งแบบแพร่กระจาย
R	Repair Flag ไว้ใช้สำหรับการส่งแบบแพร่กระจาย
G	Gratuitous RREP Flags เป็นตัวที่บอกว่าข้อความ RREP ที่ต้องการควรส่งแบบปลายทางเดียวไปยังโหนดปลายทาง
D	Destination only flag เป็นตัวบ่งบอกว่าให้โหนดปลายทางเท่านั้นที่สามารถตอบกลับข้อความ RREQ ได้
U	Unknown Sequence number เป็นตัวที่บอกว่าโหนดปลายทางที่ต้องการค้นหาเส้นทางนั้นไม่ทราบค่าหมายเลขลำดับของโหนดปลายทาง
Originator IP Address	หมายเลขโหนดของโหนดต้นทางที่สร้างและส่งข้อความ RREQ เพื่อหาเส้นทางไปยังโหนดปลายทาง
Originator Sequence Number	หมายเลขลำดับปัจจุบันของโหนดต้นทาง
Intermediate IP Address	หมายเลขโหนดของโหนดตัวแทนที่สร้างและส่งข้อความ RREQ เพื่อหาเส้นทางไปยังโหนดปลายทาง
Intermediate Sequence Number	หมายเลขลำดับปัจจุบันของโหนดตัวแทน
Number of Paths	จำนวนโหนดของเส้นทางโหนดที่ข้อความ RREQ ผ่านมาทั้งหมด
Path IP Address	หมายเลขโหนดของเส้นทางที่ข้อความ RREQ ผ่านมาทั้งหมดก่อนถึงโหนดตัวแทน (คัดลอกมาจากข้อความ RREQ ทั้งหมด)

จากรูปที่ 3.4 นี้ ลักษณะของข้อความ RREQInter จะมีลักษณะคล้าย RREQ เป็นส่วนใหญ่มี 2 ส่วนที่เพิ่มเติมคือ หมายเลขโหนดของโหนดตัวแทน (Intermediate IP Address) และหมายเลขลำดับของโหนดตัวแทน (Intermediate Sequence Number) เพื่อใช้ในการอัพเดตเส้นทางของโหนดต้นทางในตารางของโหนดปลายทาง ถ้าโหนดตัวแทนมีเส้นทางที่สามารถใช้งานได้อยู่ แสดงว่าโหนดปลายทางยังมีเส้นทางที่ใช้งานได้อยู่ของโหนดตัวแทนและสามารถนำเส้นทางนั้น คัดลอกมาให้กับโหนดต้นทางได้ส่วนหนึ่ง เส้นทางอีกส่วนหนึ่งจะมาจากหมายเลขโหนดของเส้นทาง(Path IP Address) ในข้อความ RREQInter การเรียกใช้กระบวนการนี้ ทำให้ขนาดของข้อความค้นหาเส้นทาง

มีขนาดเล็กลง และลดจำนวนของข้อความที่อยู่ในเครือข่ายจากการเปลี่ยนเป็นการส่งข้อความแบบปลายทางเดียวของ RREQInter แทนที่การส่งข้อความแบบแพร่กระจายของข้อความ RREQ

เมื่อโหนดตัวแทนส่งข้อความ RREQInter โหนดตัวแทนสร้างข้อความ RREP ตอบกลับไปยังโหนดต้นทางแทนโหนดปลายทาง โดยการบันทึกเส้นทางของโหนดปลายทางในตารางของโหนดตัวแทนลงในข้อความ RREP ต่อมาเมื่อโหนดถัดไปได้รับข้อความ RREQInter จะส่งข้อความ RREQInter ยังโหนดต่อไปและส่งข้อความ RREP ตอบกลับโหนดตัวแทนที่ส่ง RREQInter มาเพื่อเป็นการตรวจสอบว่าข้อความนั้นส่งถึง เพื่อป้องกันการสูญหายของข้อความ RREQInter ที่เป็นการส่งข้อความแบบปลายทางเดียว โดยจะมีการตอบกลับแบบนี้จนกระทั่งข้อความ RREQInter ถึงโหนดปลายทาง

3) RREP (Route Reply message) คือ ข้อความควบคุมที่ใช้สำหรับตอบกลับเส้นทางที่ถูกร้อง
4) ขอกลับไปยังโหนดต้นทางที่ร้องขอ โดยจะใช้เส้นทางที่ถูกบันทึกในตารางของโหนดปลายทาง หลังจากโหนดปลายทางได้รับข้อความร้องขอ จะมีกระบวนการสร้างเส้นทางสำหรับการใช้ส่งข้อมูลกลับไป ซึ่งข้อความ RREP มีขนาดน้อยที่สุด 21 ไบต์ โหนดปลายทางจะสามารถส่งข้อความ RREP กลับมายังโหนดต้นทางได้ ข้อความ RREP ต้องมีการเก็บเส้นทางเหมือนข้อความ RREQ เช่นกัน โดยมีโครงสร้างของข้อความดังรูปที่ 3.5 นี้ โดยส่วนที่ไฮไลท์ด้วยสีฟ้าคือส่วนของ HAC-SF ที่เพิ่มจากของเดิม



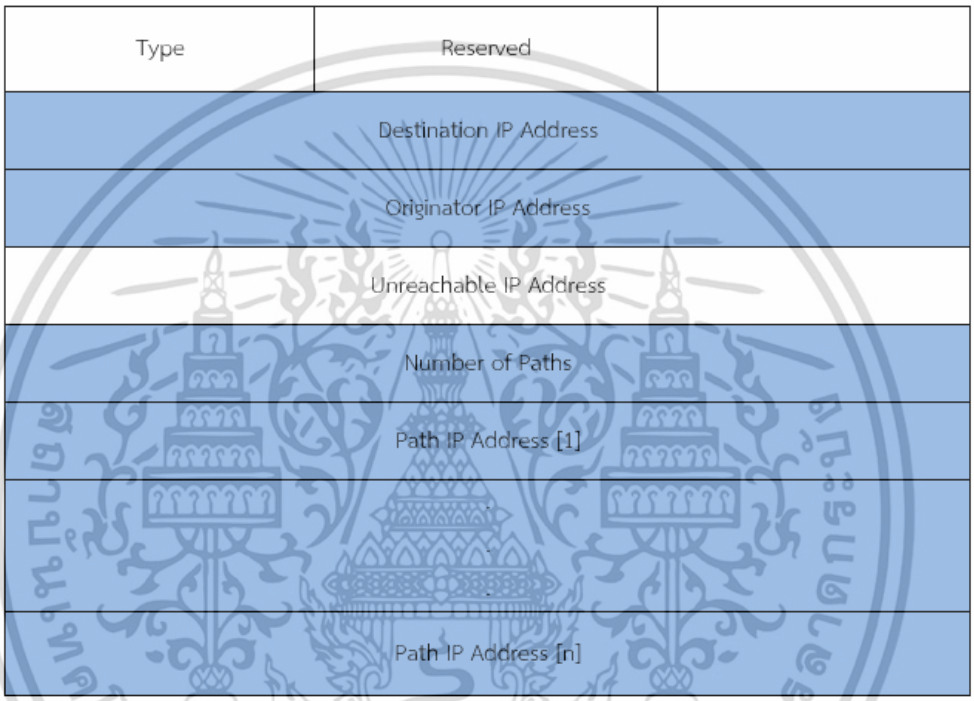
รูปที่ 3.5 ตัวอย่างโครงสร้างของข้อความ RREP

จากรูปที่ 3.5 นี้ แสดงลักษณะของข้อความ RREP โดยส่วนที่เพิ่มเติมเข้ามาในข้อความ คือ จำนวนโหนดของเส้นทาง (Number of Paths) และ หมายเลขโหนดของเส้นทางทั้งหมด (Path IP Address) ซึ่งเหมือนข้อความ RREQ แต่ส่วนที่ได้มาจากตารางเก็บเส้นทางของ gateway

ตารางที่ 3.5 ความหมายของส่วนของข้อมูลในข้อความ RREP

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
Hop Count	จำนวนของ Hop จากโหนดปลายทางถึงโหนดต้นทางที่ได้รับจากข้อความ RREQ หรือ RREQInter
R	Repair Flag ไว้ใช้สำหรับการส่งแบบแพร่กระจาย
A	Acknowledgement required เป็นตัวบอกว่าข้อความ RREP นี้มีต้องการในการตอบกลับข้อมูล (Acknowledge)
Reserved	เมื่อได้รับข้อความ โหนดจะไม่สนใจข้อความดังกล่าว
Prefix Size	ถ้า Prefix size ไม่เป็น 0 แล้ว 5-bit ในข้อมูลของ Prefix size จะเป็นตัวกำหนดหมายเลขของโหนดถัดไปที่ถูกใช้สำหรับทุกโหนดด้วยเส้นทางในการส่งข้อมูลเส้นทางเดียวกัน
Destination IP Address	หมายเลขโหนดของโหนดปลายทางในเส้นทางการส่งข้อมูล
Destination Sequence Number	หมายเลขลำดับล่าสุดของโหนดปลายทาง
Originator IP Address	หมายเลขโหนดของโหนดต้นทางที่สร้างข้อความ RREQ สำหรับการค้นหาเส้นทางดังกล่าว
Lifetime	เวลาในหน่วยมิลลิวินาที ที่โหนดจะต้องได้รับข้อความ RREP เพื่อยืนยันเส้นทางที่ใช้ในการสื่อสาร
Number of Paths	จำนวนโหนดของเส้นทางที่ข้อความ RREQ ผ่านมาทั้งหมด
Path IP Address	หมายเลขโหนดของเส้นทางที่ข้อความ RREQ ผ่านมาทั้งหมด ซึ่งได้มาจากตารางเส้นทางของ gateway

1) RERR (Route Error message) คือ ข้อความควบคุมที่ใช้สำหรับแจ้งว่าโหนดปลายทางนั้นหรือโหนดนั้นไม่สามารถไปถึงได้ เมื่อโหนดปลายทางได้รับ RERR แล้วก็จะทำการลบโหนดปลายทางนั้นออกจากตารางเส้นทางหรือหากมีโหนดถัดไป (Next hop) เป็นโหนดนั้น ก็จะทำการลบเส้นทางด้วยเช่นกัน ข้อความ RREP มีขนาดน้อยที่สุด 15 ไบต์ โดยมีโครงสร้างของข้อความดังรูปที่ 3.6 นี้ โดยส่วนที่ไฮไลท์ด้วยสีฟ้าคือส่วนของ HAC-SF ที่เพิ่มจากของเดิม



รูปที่ 3.6 โครงสร้างของข้อความ RERR

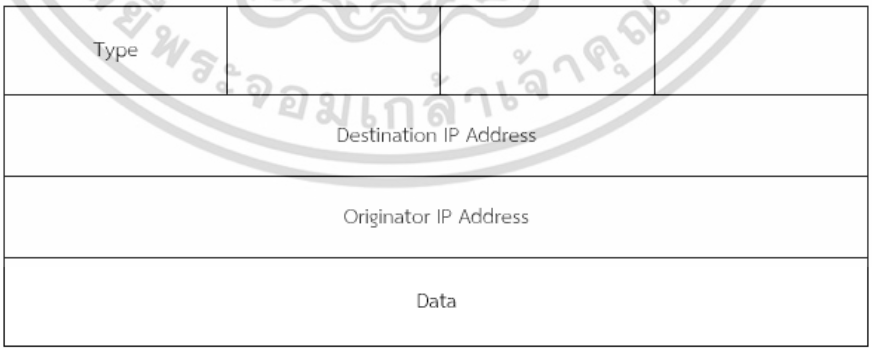
ตารางที่ 3.6 ความหมายของส่วนของข้อมูลในข้อความ RERR

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
Reserved	เมื่อได้รับข้อความ โหนดจะไม่สนใจข้อความดังกล่าว
Destination IP Address	หมายเลขที่อยู่ของโหนดปลายทางที่ต้องการการส่งข้อความ RERR
Originator IP Address	หมายเลขโหนดที่อยู่ของโหนดต้นทางที่สร้างข้อความ RERR
Unreachable Destination IP Address	หมายเลขโหนดของโหนดปลายทางที่ไม่สามารถติดต่อได้
Number of Paths	จำนวนโหนดของเส้นทางที่ข้อความ RERR ผ่านมาทั้งหมด
Path IP Address	หมายเลขโหนดของโหนดที่ข้อความ RERR ผ่านมาทั้งหมด เพื่อนำไปใช้ซ่อมแซมเส้นทางให้กับโหนดปลายทาง

จากรูปที่ 3.6 นี้ แสดงโครงสร้างของข้อความ RERR โดยหมายเลขโหนดของโหนดต้นทางและโหนดที่เสียหายทั้ง 2 ส่วนนี้จะใช้ในการค้นหาเส้นทางในตารางของโหนดปลายทางว่าเส้นทางใดบ้างที่ต้องมีการซ่อมแซม หมายเลขโหนดของเส้นทางทั้งหมด (Path IP Address) จะใช้ในการซ่อมแซมหลังจากค้นหาเส้นทางในตารางที่ได้รับผลกระทบได้แล้ว

3.3.5 รูปแบบแพ็กเก็ตข้อมูล (Data Packet Format)

1) Data Packet คือ แพ็กเก็ตที่ทำหน้าที่เก็บข้อมูลต่าง ๆที่ต้องการจะส่งให้โหนดปลายทางหรือ gateway เพื่อส่งไปยังเครือข่ายอินเทอร์เน็ต เก็บบันทึกลงฐานข้อมูลหรือใช้ในการประมวลผลต่อไป โดยแพ็กเก็ตนี้มีโครงสร้าง ดังรูปที่ 3.7



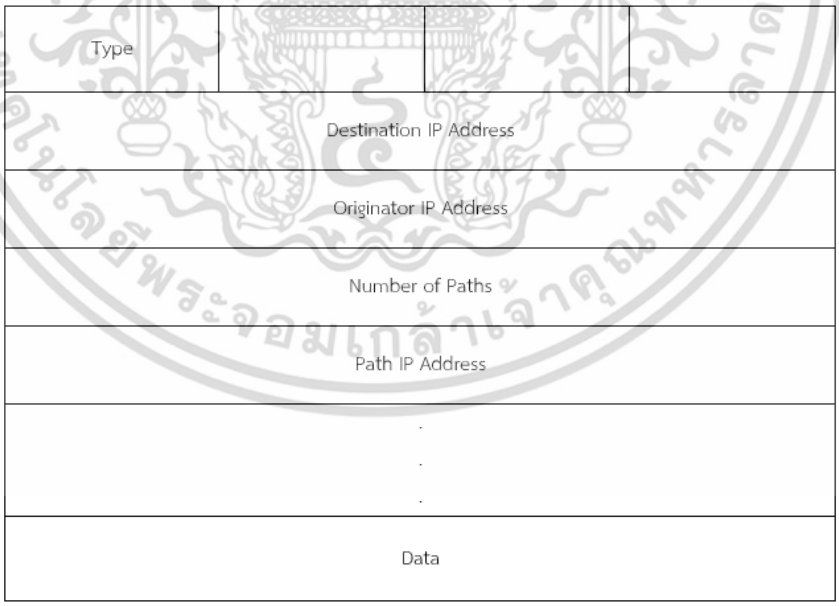
รูปที่ 3.7 ตัวอย่างโครงสร้างของ Data Packet

ตารางที่ 3.7 ความหมายของส่วนของข้อมูลในแพ็กเก็ตข้อมูล

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
Destination IP Address	หมายเลขโหนดของโหนดปลายทางที่โหนดต้นทางต้องการส่งข้อมูล
Originator IP Address	หมายเลขโหนดของโหนดต้นทางที่สร้างและส่งข้อความ RREQ เพื่อหาเส้นทางไปยังโหนดปลายทาง
Data	ข้อมูลที่ต้องการส่งให้โหนดปลายทาง

จากรูปที่ 3.7 นี้ แสดงโครงสร้างของแพ็กเก็ตข้อมูล จะประกอบไปด้วยหมายเลขโหนดของโหนดต้นทางและโหนดปลายทางและข้อมูลที่ต้องการส่ง ส่วนเส้นทางที่จะส่งสามารถค้นหาได้ในตารางเส้นทางของโหนดนั้น ๆที่แพ็กเก็ตนี้ส่งถึง

2) Data Control Packet คือ แพ็กเก็ตที่ทำหน้าที่เก็บข้อมูลหรือข้อความควบคุมที่โหนด gateway ต้องการส่งไปยังโหนดทั่วไปในเครือข่าย เพื่อใช้ในควบคุมการทำงานของโหนดทั่วไปนั้นให้ทำตามคำสั่งนั้น โดยแพ็กเก็ตนี้มีโครงสร้าง ดังรูปที่ 3.8 นี้



รูปที่ 3.8 ตัวอย่างโครงสร้างของ Data Control Packet

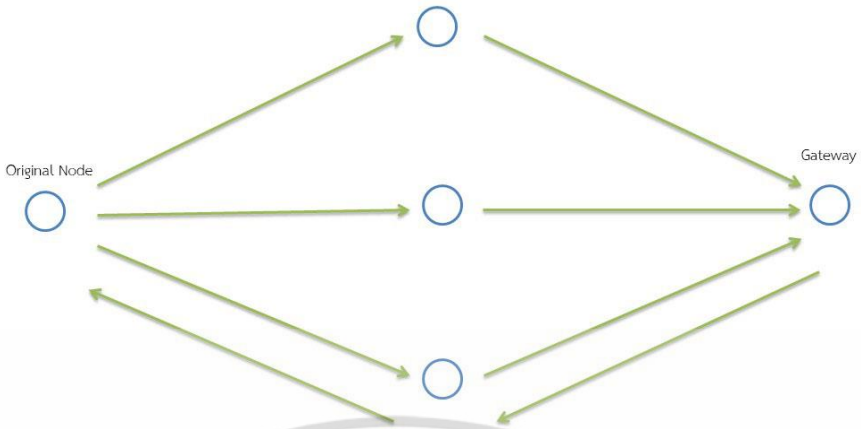
ตารางที่ 3.8 ความหมายของส่วนของข้อมูลในแพ็กเก็ตข้อมูลควบคุม

ส่วนของข้อมูลของข้อความ	ความหมาย
Type	ชนิดของข้อความ
Hop Count	จำนวนของ Hop จากต้นทางถึงปลายทาง
Destination IP Address	หมายเลขโหนดของโหนดปลายทางในเส้นทางการส่งข้อมูล
Originator IP Address	หมายเลขโหนดของโหนดที่สร้างข้อความ RREQ
Number of Paths	จำนวนโหนดของเส้นทางที่ข้อความ RREQ ผ่านมาทั้งหมด
Path IP Address	หมายเลขโหนดของโหนดที่ข้อความ RREQ ผ่านมาทั้งหมด
Data Control	ข้อมูลหรือข้อความที่ต้องการควบคุมการทำงานของโหนดปลายทาง

จากรูปที่ 3.8 แสดงโครงสร้างของแพ็กเก็ตข้อมูลควบคุม โดยจะมีส่วนที่เพิ่มเติมจากแพ็กเก็ตข้อมูล คือ Number of Paths และ Path IP Address ซึ่ง 2 ส่วนนี้จะเป็นเส้นทางให้กับแพ็กเก็ตนี้ให้สามารถส่งแพ็กเก็ตไปยังโหนดทั่วไปที่ต้องการควบคุมนั้นได้ ซึ่งจะมีกระบวนการส่งแพ็กเก็ตเหมือนกับข้อความ RREP

3.3.6 ขั้นตอนการทำงานของโพรโทคอล HAC-SF

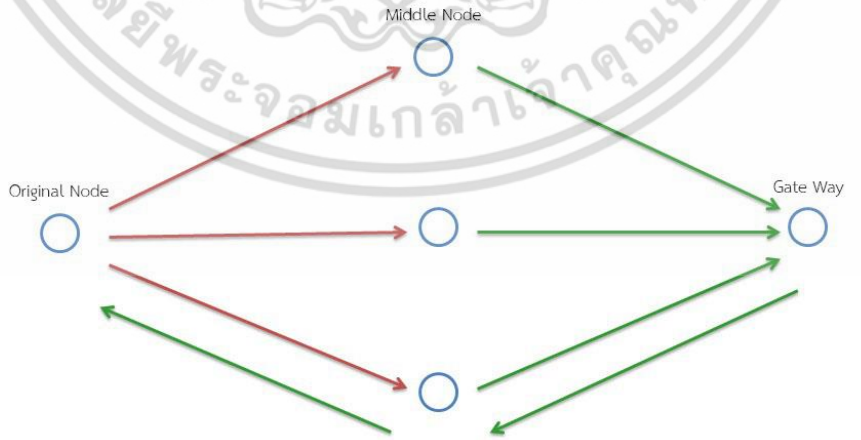
โพรโทคอล AODV มีการส่งข้อความแบบแพร่กระจาย (Broadcast) ไปเรื่อยจนกว่าจะพบโหนดปลายทาง ทำให้โหนดมีการบันทึกเส้นทางของโหนดต้นทางทุกตัวที่ผ่านโหนดนั้น ถึงแม้เส้นทางนั้นโหนดต้นทางและ gateway จะไม่ได้มีการบันทึกลงในตารางค้นหาเส้นทาง หรือ ได้ใช้เส้นทางนั้นในการส่งข้อมูล ทำให้เส้นทางในตารางค้นหามีมากเกินความจำเป็น ดังรูปที่ 3.9 นี้



รูปที่ 3.9 การบันทึกเส้นทางในตารางค้นหาเส้นทางของโพรโทคอล AODV

จากรูปที่ 3.9 เส้นสีเขียวแสดงการบันทึกเส้นทางของโหนดต้นทาง จะเห็นได้ว่ามีการบันทึกเส้นทางของโหนดต้นทาง ในโหนดทุกโหนดที่ข้อความ RREQ ของโหนดต้นทางผ่าน แต่มีการเลือกเส้นทางในการส่งข้อมูลเฉพาะเส้นทางที่สามารถไปยังโหนดปลายทางได้เร็วที่สุดเส้นทางเดียวเท่านั้น และส่งข้อมูลเส้นทางกลับไปให้โหนดต้นทาง ทำให้มีข้อมูลของเส้นทางของโหนดในตารางที่ไม่ได้มีการใช้งาน ปล่อยทิ้งไว้จนเวลาของเส้นทางนั้นหมดลงและยกเลิกการใช้งาน

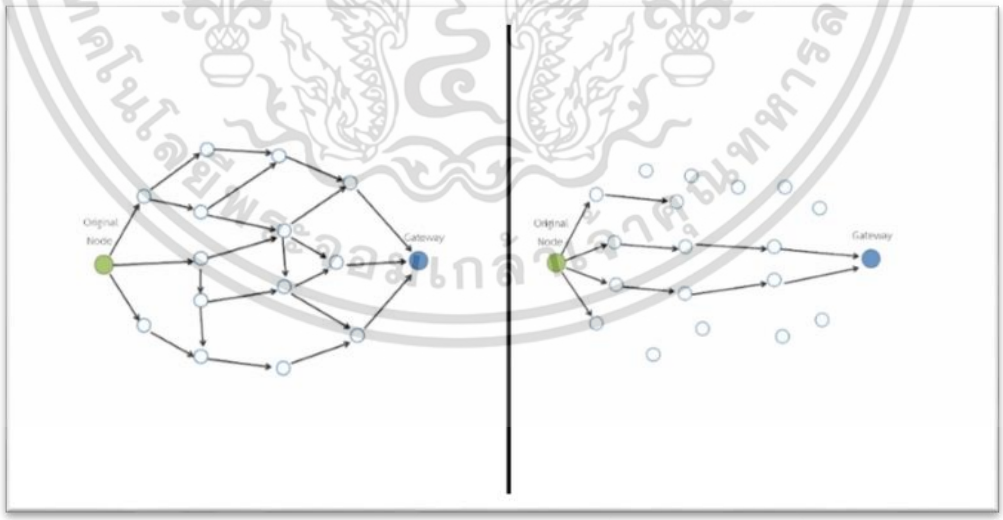
ดังนั้นผู้ทำงานวิจัยได้คิดการปรับปรุง โดยไม่บันทึกเส้นทางของโหนดต้นทาง เมื่อโหนดทั่วไปได้รับข้อความ RREQ มีการบันทึกเส้นทางเฉพาะเมื่อ gateway ได้รับข้อความ RREQ เท่านั้น แต่จะมีปัญหาในการส่งข้อความตอบกลับ RREP กลับไปยังโหนดต้นทางจากโหนดปลายทาง เพราะจะไม่มีเส้นทางกลับมาที่โหนดต้นทางอีกครั้ง



รูปที่ 3.10 การปรับปรุงการบันทึกเส้นทางในตารางค้นหาเส้นทาง

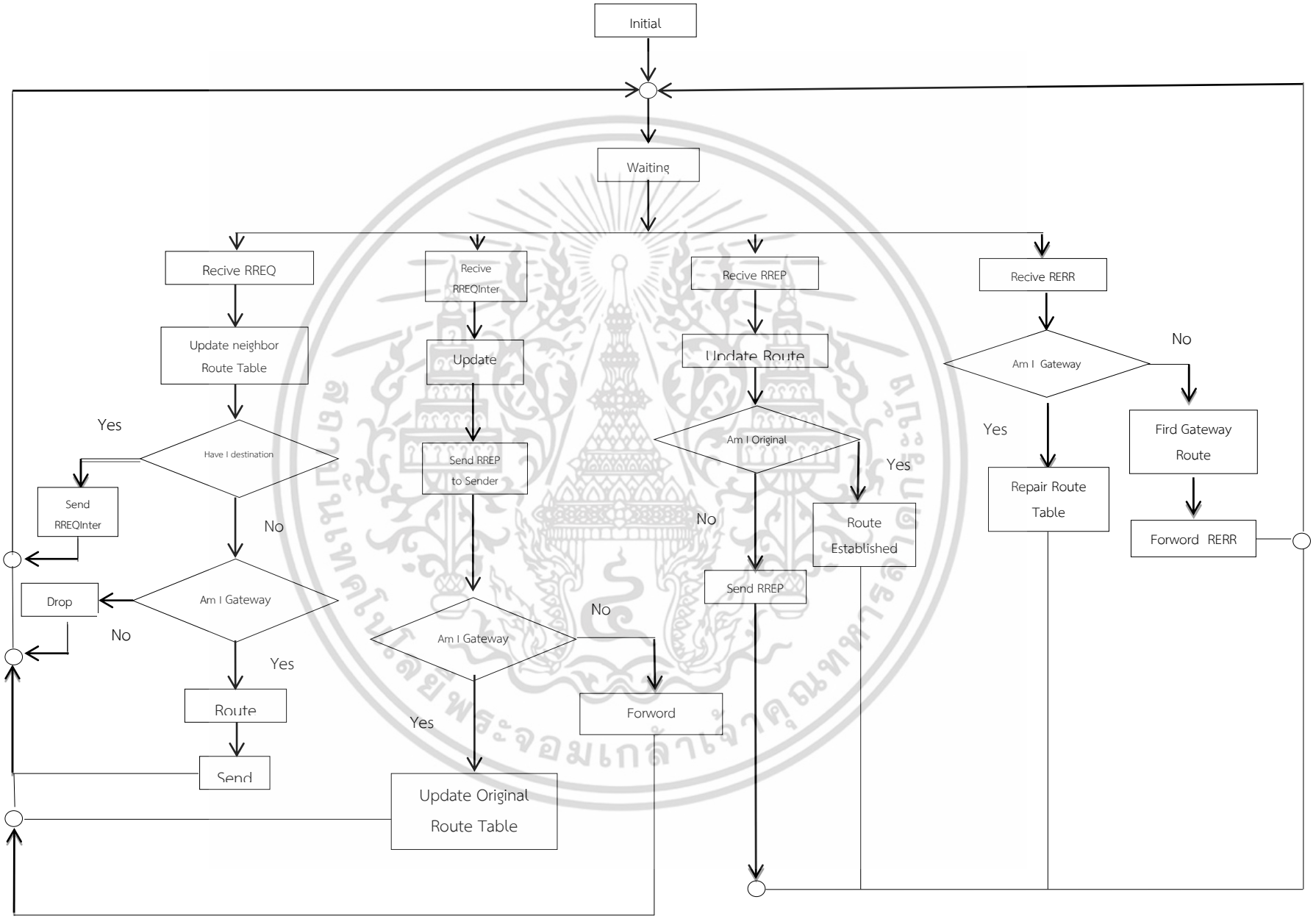
จากรูปที่ 3.10 แสดงการปรับปรุงวิธีการบันทึกเส้นทางลงในตารางของโหนดในเครือข่าย โดยเส้นสีเขียวแสดงการบันทึกเส้นทางทางของโหนดต้นทางหรือโหนดปลายทางลงในตารางของโหนดนั้น และเส้นสีแดงแสดงว่าไม่มีการบันทึกเส้นทางลงในตารางของโหนดนั้น

การทำงานของโพรโทคอล HAC-SF คือ เมื่อมีโหนดต้นทางที่ต้องการส่งข้อมูลไปยังโหนดปลายทาง โหนดต้นทางจะทำการส่งข้อความ RREQ ไปยังโหนดเพื่อนบ้านแบบแพร่กระจายเพียง 1 ครั้ง เพื่อลดขอบเขตการค้นหาเส้นทาง ดังรูปที่ 3.11 นี้ และเมื่อโหนดที่ได้รับข้อความก็จะทำการตรวจสอบก่อนว่าโหนดที่ได้รับเป็นโหนดปลายทางหรือไม่ ถ้าเป็นโหนดปลายทางจะเลือกบันทึกเส้นทางจากข้อความที่มาถึงตัวแรก เพราะถือว่าใช้เวลาน้อยที่สุดในการส่ง RREQ มาจากต้นทาง และทำการส่งข้อความ RREP ส่งกลับไปยังโหนดต้นทาง ถ้าไม่เป็นโหนดปลายทางจะทำการตรวจสอบว่ามีเส้นทางของโหนดปลายทางในตารางเส้นทางหรือไม่และต้องเป็นเส้นทางที่ยังสามารถใช้งานได้ด้วย ถ้าไม่มีจะทำการทิ้งข้อความ RREQ นั้นไป แต่ถ้าโหนดนั้นมีเส้นทางของโหนดปลายทางในตารางเส้นทาง โหนดจะสร้างข้อความ RREQInter และส่งข้อความต่อไปยังโหนดที่ใกล้เคียงตามเส้นทางในตารางต่อกันไปจนถึงโหนดที่ต้นทางต้องการจะติดต่อด้วยหรือโหนดปลายทางนั้นเอง โดยส่งข้อความแบบปลายทางเดียว และเมื่อโหนดปลายทางได้รับ RREQInter โหนดปลายทางก็จะทำการส่ง RREP กลับไปยังโหนดตัวแทนที่ทำการส่ง RREQInter มาให้โดยจะส่งกลับไปในเส้นทางตามที่ gateway ได้บันทึกเส้นทางไว้ในตาราง เพราะโหนดตัวแทนมีเส้นทางของ gateway แสดงว่า โหนดตัวแทนต้องเคยร้องขอเส้นทางของ gateway และ gateway ก็ต้องมีเส้นทางของโหนดตัวแทนด้วย RREQ มาจากต้นทาง ดังรูปที่ 3.12 นี้



รูปที่ 3.11 เปรียบเทียบขอบเขตการค้นหาเส้นทางของ AODV และ HAC-SF

จากรูปที่ 3.11 ภาพซ้ายแสดงขอบเขตการค้นหาของโพรโทคอล AODV ที่มีการค้นหาแบบแพร่กระจายไปเรื่อยๆจนกว่าจะพบโหนดปลายทาง และภาพขวาแสดงขอบเขตการค้นหาของโพรโทคอล HAC-SF ที่จะค้นหาโดยแบบแพร่กระจายเพียงครั้งเดียว ถ้าพบโหนดที่มีเส้นทางจะเปลี่ยนเป็นการส่งแบบปลายทางเดียว ซึ่งลดขอบเขตการค้นหาและยังลดจำนวนข้อความในเครือข่ายอีกด้วย



รูปที่ 3.12 ตาราง Flowchart การทำงานพื้นฐานของโพรโทคอล HAC-SF

จากรูปที่ 3.12 นี้ หลักการทำงานของโพรโทคอล HAC-SF โหนดจะทำการรอรับข้อความการจัดการเส้นทาง ถ้าหากเป็นข้อความ RREQ โหนดจะเข้าสู่กระบวนการปรับปรุงตารางเส้นทางของโหนดข้างเคียงที่ส่งข้อความนี้และตรวจสอบข้อมูลเส้นทางของตัวเองว่ามีข้อมูลที่ต้องการหรือไม่ หากมีจะสร้างและส่งข้อความ RREQInter โหนดจะเข้าสู่กระบวนการปรับปรุงตารางเส้นทางของโหนดข้างเคียงที่ส่งข้อความนี้และหากไม่มีจะตรวจสอบว่าเป็นโหนดปลายทางหรือไม่ ถ้าไม่จะทำการทิ้งข้อความนั้น แต่ถ้าเป็นโหนดปลายทางจะทำการสร้างหรือปรับปรุงตารางเส้นทางแล้วจึงส่งข้อความ RREP กลับไปยังโหนดต้นทาง ในกรณีโหนดได้รับข้อความ RREQInter ในกรณีโหนดได้รับข้อความ RREP โหนดจะทำการพิจารณาว่าข้อความ RREP ดังกล่าวมีความต้องการส่งมายังตนเองหรือไม่ หากข้อความดังกล่าวไม่ได้ต้องการส่งมายังตนเอง โหนดจะไม่พิจารณาข้อความนั้น แต่หากข้อความดังกล่าวส่งมายังตนเองโหนดจะเข้าสู่กระบวนการปรับปรุงตารางเส้นทางและส่งต่อข้อความ RREP กลับไปยังโหนดที่มีความต้องการเส้นทางนั้น ๆต่อไป โดยหลักการทำงานแบบละเอียดของโพรโทคอล HAC-SF สามารถ แบ่งได้เป็น 12 ขั้นตอนดังนี้

1. หมายเลขลำดับ (Sequence Numbers)

หมายเลขลำดับของโหนดปลายทาง (Destination Sequence Number) ซึ่งจะถูกทำการปรับปรุงก็ต่อเมื่อโหนดได้รับหมายเลขลำดับใหม่จากข้อความ RREQ ข้อความ RREQInter ข้อความ RREP หรือ ข้อความ RRER ที่มีความเกี่ยวข้องกับโหนดปลายทาง โดยในโพรโทคอล HAC-SF ยังใช้หมายเลขลำดับปลายทางเพื่อทำให้ไม่เกิดลูปของการค้นหาเส้นทางทั้งหมดด้วย โหนดแต่ละโหนดจะมีค่าหมายเลขลำดับเป็นของตัวเองและจะเพิ่มค่าดังกล่าวดังกรณีต่อไปนี้

1) ก่อนที่โหนดต้นทางจะค้นหาเส้นทาง โหนดจะเพิ่มค่าหมายเลขลำดับของตัวเองเพื่อหลีกเลี่ยงการซ้ำซ้อนของข้อมูลก่อนหน้าที่เคยถูกส่งและใช้งานในเส้นทางย้อนกลับ

2) ก่อนที่ปลายทางที่ได้รับ RREQ จะทำการส่ง RREP กลับ โหนดนั้นจะต้องเพิ่มหมายเลขลำดับของตัวเองเป็นค่าที่สูงที่สุด ระหว่างค่าหมายเลขลำดับของตัวเองและค่าหมายเลขลำดับปลายทางในข้อความ RREQ และเพิ่มค่าไปอีกหนึ่ง เมื่อค่าหมายเลขลำดับเพิ่มจนถึงขีดจำกัดของตัวแปรที่เก็บแล้วก็ให้วนมาที่ค่า 0 ใหม่ ในกรณีที่ได้รับ RREP มาแล้วการที่จะเปลี่ยนค่าหมายเลขลำดับปลายทางของปลายทางนั้นๆได้ ก็ต่อเมื่อมีค่าหมายเลขลำดับปลายทางสูงกว่าเท่านั้น

สำหรับค่าของหมายเลขลำดับในตารางเส้นทางของโหนดจะมีการเปลี่ยนแปลงได้ 3 กรณีดังต่อไปนี้

1) เมื่อตนเองเป็นโหนดปลายทางและมีเส้นทางใหม่ไปยังตนเอง

2) โหนดดังกล่าวได้รับข้อมูลการจัดการเส้นทางที่มีข้อมูลที่ใหม่กว่าสำหรับปลายทางนั้นๆ

3) เส้นทางที่ใช้ในการส่งข้อมูลไปยังปลายทางหมดอายุหรือเกิดความเสียหาย

2. ข้อมูลเส้นทางหลายเส้นทางต่อหนึ่งโหนดปลายทาง

เมื่อโหนดได้รับข้อความควบคุม (control packet) จากโหนดข้างเคียง หรือมีการสร้างหรือปรับปรุงเส้นทางเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทาง โหนดดังกล่าวจะทำการตรวจสอบว่าตารางเส้นทางของตนเองมีข้อมูลเส้นทางเพื่อใช้ในการส่งไปยังโหนดปลายทางหรือไม่ หากไม่มีข้อมูลเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทาง โหนดดังกล่าวจะทำการสร้างข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายขึ้นก่อน โดยที่หมายเลขลำดับจะถูกกำหนดจากข้อมูลที่มีอยู่ในข้อความที่ได้รับมา และข้อมูลเส้นทางจะปรับปรุงให้มีหมายเลขลำดับใหม่ด้วยสาเหตุต่อไปนี้

1) เมื่อได้รับข้อความควบคุมที่มีค่าหมายเลขลำดับสูงกว่าในหมายเลขลำดับของโหนดปลายทางในตารางเส้นทางนั้นๆ

2) เมื่อได้รับข้อความควบคุมที่มีค่าหมายเลขที่เท่ากับหมายเลขลำดับของโหนดปลายทางในตารางเส้นทางแต่มีค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลน้อยกว่า

3) เมื่อโหนดไม่ทราบหมายเลขลำดับที่ใช้ในการส่งข้อมูลไปยังโหนดปลายทาง

โดยโหนดจะสามารถบันทึกเส้นทางเพิ่มได้ เมื่อได้รับข้อความควบคุมจากโหนดข้างเคียงที่ไม่ซ้ำกันเท่านั้น หรือถ้าซ้ำหรือเคยรับข้อความจากโหนดข้างเคียงนี้มาก่อนก็จะพิจารณาจากข้อกำหนดดังข้างบนที่กล่าวมา สำหรับเวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้ (Life time) ที่จัดเก็บอยู่ในตารางเส้นทางจะมีการกำหนดค่าจากข้อความควบคุมหรือมีการตั้งค่าเริ่มต้นจากตัวแปร ACTIVE_ROUTE_TOMEOUT

สำหรับทุกเส้นทางที่โหนดพิจารณาให้เป็นเส้นทางที่ยังคงทำงานได้ (Valid route) โหนดดังกล่าวจะต้องเก็บรักษารายการของหมายเลขโหนดที่ถูกใช้งานในการส่งข้อความ RREP โดยค่ารายการดังกล่าวจะมีการจัดเก็บโหนดข้างเคียง (Neighboring nodes) ที่ทำการสร้างหรือส่งต่อข้อความ RREP

3. การสร้างข้อความ RREQ

เมื่อโหนดต้นทางต้องการส่งข้อความไปยังโหนดปลายทาง และโหนดต้นทางดังกล่าวไม่มีข้อมูลเส้นทางเพื่อใช้ในการส่ง โหนดดังกล่าวจะสร้างข้อความ RREQ และส่งต่อไปยังโหนดข้างเคียง ในส่วนของหมายเลขลำดับของโหนดปลายทางในข้อความ RREQ จะเป็นค่าสุดท้ายที่โหนดดังกล่าวเคยมีข้อมูลอยู่ แต่ถ้าไม่มีค่าตัวแปร Unknown sequence number ในข้อความ RREQ ก็จะถูกใช้งาน ส่วนหมายเลขลำดับของโหนดต้นทาง (Originator sequence number) ในข้อความ RREQ จะเป็นค่าเริ่มต้นหมายเลขลำดับโหนดนั้นๆซึ่งจะถูกเพิ่มครั้งละ 1 เหมือนกับค่าหมายเลขลำดับของ RREQ (RREQ ID) ซึ่งทั้งระบบจะมีการเก็บรักษาหมายเลขลำดับ RREQ ร่วมกัน สำหรับค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลจะถูกตั้งค่าเริ่มต้นให้มีค่าเป็น 0 ก่อนที่จะแพร่กระจายข้อความ RREQ

โหนดต้นทางจะเก็บค่าชั่วคราวของข้อมูลหมายเลขลำดับ RREQ และหมายเลขของโหนดต้นทางข้อความ RREQ ในช่วงระยะเวลา PATH_DISCOVERY_TIME และเมื่อโหนดดังกล่าวได้รับข้อความ RREQ จากโหนดใกล้เคียงซึ่งมีค่าหมายเลขลำดับ RREQ และหมายเลขของโหนดต้นทางที่ตนเองเป็นผู้ส่งออกไป โหนดจะไม่พิจารณาข้อมูลดังกล่าว นอกจากนี้โหนดต้นทางจะต้องไม่สร้างข้อความ RREQ มากกว่าค่าที่กำหนดไว้ใน RREQ_RATELIMIT ต่อวินาที ซึ่งหลังจากส่งข้อความ RREQ แบบแพร่กระจายออกไป โหนดจะรอข้อความ RREP ถ้าไม่ได้รับข้อมูลในช่วงเวลาที่กำหนดไว้ในตัวแปร NET_TRAVERSEL_TIME มีหน่วยเป็นมิลลิวินาที โหนดดังกล่าวอาจพยายามค้นหาเส้นทางใหม่อีกครั้ง โดยการเพิ่มค่าหมายเลขลำดับ RREQ และจำนวนครั้งในการพยายามในการค้นหาเส้นทางใหม่นั้นจะไม่ได้มีจำนวนที่จำกัด แต่จะมีการใส่ระยะเวลาในการค้นหาเส้นทางใหม่ให้นานขึ้นเพื่อ เป็นการลดความหนาแน่นในระบบ และเวลาในการส่งข้อความใหม่นั้นจะไม่เพิ่มขึ้นแบบ Binary exponential back off เหมือนกับโพโทคอล AODV ดังเดิมที่ระยะเวลาในการรอข้อความ RREP จะมีค่าเพิ่มขึ้นเป็น 2 เท่าของระยะเวลาในการรอล่าสุดและมีการจำกัดจำนวนในการค้นหาเส้นทางใหม่ เพราะโพรโทคอล HAC-SF มีการส่งข้อความ RREQ แบบแพร่กระจายแค่ครั้งเดียว หรือ ค่า Time to Live (TTL) ในข้อความเป็น 1 เสมอ ถ้าหากยังคงไม่ได้รับข้อความ RREP ในช่วงเวลาดังกล่าว ข้อความ RREQ จะถูกส่งแบบแพร่กระจายใหม่อีกครั้ง และทุกๆครั้งที่มีความพยายามค้นหาเส้นทางระยะเวลาในการรอข้อความ RREP จะมีค่าเท่าเดิมตลอด และก่อนการส่งข้อความ RREQ จะทำการบันทึกหมายเลขไอพีของโหนดต้นทางลงในส่วนของ Path IP Address เพื่อเป็นเส้นทางให้กับโหนดปลายทางส่งข้อความ RREP กลับมาได้

4. การควบคุมการส่ง Route Request message

เพื่อหลักเลี่ยงการส่งข้อความ RREQ แบบกระจายไปทั่วทั้งระบบโดยไม่จำเป็น โดยโหนดต้นทางจะเริ่มทำการกำหนดค่าเริ่มต้นของ Time to life (TTL) ให้มีค่าเท่ากับ 1 ค่า TTL หมายถึงค่าจำนวนแพ็กเกจที่สามารถถูกส่งต่อได้ และตั้งค่าระยะเวลาที่กำหนด (Timeout) สำหรับการรอรับข้อความ RREP ให้มีค่าเป็นระยะเวลาหนึ่ง และถ้าโหนดไม่ได้รับข้อความ RREP ภายในเวลาที่กำหนด โหนดต้นทางจะทำการส่งข้อความ RREQ แบบแพร่กระจายอีกครั้ง และค่า TTL จะเป็น 1 เหมือนเดิม

5. การพิจารณาข้อมูลที่ได้รับจากข้อความ RREQ

เมื่อโหนดได้รับข้อความ RREQ ขั้นแรกโหนดดังกล่าวจะทำการตรวจสอบว่าเคยได้รับข้อความ RREQ ที่มีค่าหมายเลขของโหนดต้นทางและหมายเลขลำดับ RREQ โหนดดังกล่าวจะไม่กระทำกระบวนการใดๆ แต่หากยังไม่เคยได้รับข้อความ RREQ โหนดจะเริ่มกระบวนการต่อไปนี้ ขั้นแรกโหนดจะทำการเพิ่มค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลที่อยู่ภายในข้อความ RREQ ขึ้นหนึ่งค่า เพื่อใช้ในการส่งต่อไปยังโหนดข้างเคียง หลังจากนั้นโหนดจะสร้างหรือปรับปรุงค่าของโหนดข้างเคียงที่ส่งข้อความนี้มาก่อนหน้า โดยไม่สนใจหมายเลขลำดับ หลังจากนั้นโหนดจะทำการตรวจสอบว่าโหนดตัวเองเป็น โหนดปลายทาง(gateway) ที่ระบุในข้อความ RREQ หรือไม่ ถ้าเป็นโหนดปลายทางจะโหนดจะสร้างหรือปรับปรุงค่าของโหนดต้นทางที่ส่งข้อความนี้ในตารางเส้นทางโดยไม่สนใจหมายเลขลำดับ โหนดปลายทางจะบันทึกเส้นทางของโหนดต้นทางโดยดึงค่า Path IP Address ในข้อความบันทึกลงในส่วนของ Path IP Address ในตารางของ gateway เช่นกัน แต่ถ้าโหนดตัวเองไม่ใช่โหนดปลายทางจะตรวจสอบว่าโหนดตังเองมีเส้นทางของโหนดปลายทางที่ยังสามารถใช้งานได้ในตารางเส้นทางหรือไม่ ถ้ามีเส้นทางของโหนดปลายทางต้องตรวจสอบด้วยว่าเส้นทางนั้น โหนดถัดไปไม่ใช่โหนดเดียวกับโหนดต้นทางเพื่อป้องกันไม่ให้เกิดลูปเวลาส่งข้อมูลและเลือกเส้นทางที่สั้นที่สุดของโหนดปลายทางในตารางเส้นทาง เพื่อส่งข้อความRREP กลับไปยังโหนดต้นทาง ดังนั้นโหนดต้นทางจะได้เส้นทางที่สั้นที่สุดแล้วไม่เกิดลูปแน่นอน โหนดจะสร้างข้อความ RREQInter ส่งไปยังโหนดปลายทางแบบปลายทางเดียว เพราะโหนดมีเส้นทางของปลายทางดังนั้นไม่จำเป็นต้องส่งข้อความแบบแพร่กระจายอีกต่อไป โดยข้อความ RREQInter จะคัดลอกค่าต่างๆในข้อความ RREQ และเพิ่มข้อมูลบางส่วนลงในข้อความด้วย ถ้าโหนดตรวจพบว่าไม่มีข้อมูลของโหนดปลายทาง ไม่สามารถส่งข้อความต่อไปได้ จะทำการทิ้งข้อความนั้น เพื่อให้โหนดต้นทางร้องขอเส้นทางใหม่อีกครั้ง

จากที่กล่าวมาข้างต้นจะเห็นว่าโพรโทคอล HAC-SF จะแตกต่างจากโพรโทคอล AODV ตรงที่ AODV มีการส่งข้อความแบบแพร่กระจาย (Broadcast) และเมื่อโหนดได้รับข้อความ RREQ โหนดจะบันทึกเส้นทางของโหนดต้นทางในระหว่างทางที่ข้อความ RREQ ผ่านถึงแม้เส้นทางนั้นจะไม่ได้เป็นเส้นทางที่ใช้ในการตอบกลับข้อความ RREP หรือเป็นเส้นทางที่โหนดต้นทางและ gateway นั้นใช้ในการส่งข้อมูลซึ่งกันและกัน ทำให้มีเส้นทางในตารางค้นหามีมากเกินความจำเป็นในโหนดต้นทางทั่วไปและเปลืองพื้นที่หน่วยความจำ เมื่อนำมาใช้ในเครือข่ายแบบที่ต้องการ ดังนั้นโพรโทคอล HAC-SF จึงไม่มีการบันทึกเส้นทางของโหนดต้นทางระหว่างการส่งข้อความ RREQ หรือ RREQInter แต่ปัญหาคือเมื่อไม่มีการบันทึกเส้นทางของโหนดต้นทางทำให้ไม่มีเส้นทางที่จะส่งข้อความตอบกลับไปยังโหนดต้นทางได้ ดังนั้นผู้ทำงานวิจัยจึงได้เปลี่ยนมาเก็บข้อมูลเส้นทางในข้อความค้นหาเส้นทางแทนการบันทึกลงในตารางเส้นทางของโหนดระหว่างทางแทน และปรับเปลี่ยนโครงสร้างของข้อความ RREQ และ RREP ด้วย

6. การสร้างข้อความ RREQInter

เมื่อโหนดได้รับข้อความ RREQ โหนดจะทำการตรวจสอบว่าโหนดตัวเองเป็น โหนดปลายทาง (gateway)ที่ระบุในข้อความ RREQ หรือไม่ ถ้าโหนดตัวเองไม่ใช่โหนดปลายทางจะตรวจสอบว่าโหนดตัวเองมีเส้นทางของโหนดปลายทางที่ยังสามารถใช้งานได้ในตารางเส้นทางหรือไม่ ถ้ามีเส้นทางของโหนดปลายทางโหนดจะสร้างข้อความRREQInter โดยจะคัดลอกค่าจากข้อความ RREQ ทั้งหมดมาบันทึกลงในข้อความได้แก่ จำนวน Hop ,หมายเลขไอพีโหนดต้นทางและโหนดปลายทาง ,หมายเลขลำดับของโหนดต้นทางและโหนดปลายทาง ,หมายเลขลำดับของ RREQ (RREQ ID) , Number of Path และหมายเลขไอพีของ Path ทั้งหมด แต่จะมี 2 ส่วนที่เพิ่มเข้าไปในข้อความ RREQInter คือ หมายเลขไอพีของโหนดที่ส่งข้อความ RREQInter หรือโหนดตัวแทนและหมายเลขลำดับของโหนดตัวแทน โดยข้อความ RREQInter จะมีการตั้งค่า Time to live เท่ากับค่าจำนวน hop ในตารางเส้นทางของโหนดปลายทาง โดยโหนดตัวแทนจะมีการรอรับข้อความ RREP เพื่อเป็นการตรวจสอบว่าข้อความ RREQInter ไปถึงโหนดปลายทางอย่างแน่นอน ถ้าโหนดไม่ได้รับข้อความ RREP ในช่วงเวลาที่กำหนดไว้ โหนดดังกล่าวอาจพยายามส่งข้อความ RREQInter ใหม่อีกครั้ง เพราะก่อนที่โหนดตัวแทนจะส่งข้อความ RREQInter โหนดจะทำการบันทึกข้อความไว้ในกรณีที่ไม่ได้รับข้อความตอบกลับ สามารถนำข้อความนั้นมาส่งใหม่ได้

7. การพิจารณาข้อมูลที่ได้รับจากข้อความ RREQInter

เมื่อโหนดได้รับข้อความ RREQInter ขั้นแรกโหนดจะทำการเพิ่มค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลที่อยู่ภายในข้อความ RREQ ขึ้นหนึ่งค่า เพื่อใช้ในการส่งต่อไปยังโหนดข้างเคียง หลังจากนั้นโหนดจะสร้างหรือปรับปรุงค่าของโหนดข้างเคียงที่ส่งข้อความนี้มาก่อนหน้า โดยไม่สนใจหมายเลขลำดับ หลังจากนั้นโหนดจะทำการตรวจสอบว่าโหนดตัวเองเป็น โหนดปลายทาง (gateway)ที่ระบุในข้อความ RREQ หรือไม่ ถ้าเป็นโหนดปลายทางจะโหนดจะสร้างหรือปรับปรุงค่าของโหนดต้นทางที่ส่งข้อความนี้ในตารางเส้นทาง โดยไม่สนใจหมายเลขลำดับ โหนดปลายทางจะบันทึกเส้นทางของโหนดต้นทางโดย gateway จะตรวจสอบเส้นทางในตารางของโหนดตัวแทนและทำการคัดลอกค่าหมายเลขไอพี Path ทั้งหมดของโหนดตัวแทน มาให้กับหมายเลขไอพี Path ของโหนดต้นทาง แล้วทำการคัดลอกค่าหมายเลขไอพีของ Path ในข้อความ RREQInter นำมาใส่ในต่อจากหมายเลขไอพี Path ของโหนดต้นทางก่อนหน้านี้ ที่นี้ gateway ก็มีเส้นทางของโหนดต้นทาง แต่ถ้าโหนดที่ได้รับข้อความไม่เป็น gateway โหนดทำการหาเส้นทางในตารางเส้นทางเพื่อส่งข้อความ RREQInter ไปต่อเรื่อยๆจนกว่าจะถึงโหนดปลายทางที่ต้องการนั้น

8. การสร้างข้อความ RREP

กรณีที่โหนดจะทำการส่งข้อความ RREP กลับมีขั้นตอนดังต่อไปนี้

1) โหนดที่ได้รับข้อความ RREQ ดังกล่าวเป็นโหมดปลายทาง ถ้าหากว่าหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ นั้นมีค่ามากกว่าค่าหมายเลขลำดับของตนเอง โหนดจะทำการเพิ่มค่าของหมายเลขลำดับของตนเองขึ้นหนึ่งแต่ถ้าหากว่าค่าหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ ไม่มากกว่าค่าหมายเลขลำดับตนเอง โหนดดังกล่าวจะไม่กระทำกระบวนการเพิ่มค่าหมายเลขลำดับของตนเอง หลังจากนั้นจะทำการตั้งค่าเริ่มต้นให้ค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลมีค่าเป็น 0 และตั้งค่าให้หมายเลขลำดับของโหนดปลายทางภายในข้อความ RREP มีค่าเท่ากับค่าหมายเลขลำดับของตนเอง และทำการส่งข้อความ RREP กลับผ่านทางเส้นทางย้อนกลับ

2) โหนดดังกล่าวได้รับข้อความ RREQ และตรวจสอบแล้วว่ามีข้อมูลเส้นทางที่ใช้งานได้ (Active route) ของโหนดปลายทาง เพื่อส่งข้อมูลไปยังโหนดปลายทาง โดยที่หมายเลขลำดับของโหนดปลายทางในตารางเส้นทางนั้นมีค่ามากกว่าหรือเท่ากับหมายเลขลำดับของโหนดปลายทางภายในข้อความ RREQ ที่ได้รับมาและจะทำการคัดลอกค่าหมายเลขลำดับดังกล่าวไปยังหมายเลขลำดับของโหนดปลายทางลงในข้อความ RREP และคัดลอกค่าหมายเลขไอพีของ Path ในข้อความ RREQ และหมายเลขไอพีของโหนดตัวเองใส่ในหมายเลขไอพี path ในข้อความ RREP ด้วย เพื่อเป็นเส้นทางให้กับข้อความ RREP ก่อนทำการส่งกลับไปยังโหนดต้นทาง

3) โหนดดังกล่าวได้รับข้อความ RREQInter โหนดจะสร้างข้อความ RREP ส่งกลับไปให้โหนดก่อนหน้านี้ เพื่อเป็นการยืนยันว่าข้อความส่งถึงแล้วและเป็นการอัพเดตข้อมูลของเส้นทางให้กับโหนดที่ส่งไปด้วย เพราะเนื่องจากการส่งข้อความ RREQInter เป็นการส่งแบบปลายทางเดียวจึงต้องมีการตรวจสอบว่าข้อความส่งถึงปลายทางได้ถูกต้องสมบูรณ์

ในการสร้างข้อความ RREP โหนดจะทำการคัดลอกค่าของหมายเลขไอพีของโหนดปลายทาง หมายเลขลำดับของโหนดต้นทางและหมายเลขของ Path ที่ได้จากข้อความ RREQ ไปยังข้อความ RREP และการส่งข้อความ RREP จะเป็นการส่งแบบปลายทางเดียวไปยังโหนดตัวถัดไป โดยมีปลายทางเป็นโหนดต้นทางที่ทำการสร้างข้อความ RREQ ส่วนค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลจะถูกเพิ่มขึ้นครั้งละหนึ่ง เมื่อมีการส่งข้อมูลผ่านแต่ละโหนดเช่นเดียวกับการส่งข้อความ RREQ ดังนั้นเมื่อข้อความ RREP ถูกส่งไปถึงโหนดต้นทางที่มีความต้องการเส้นทาง ค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลภายในข้อความ จะหมายถึงระยะทางหรือจำนวนของโหนดที่จำเป็นต้องทำการส่งต่อข้อมูลระหว่างโหนดต้นทางกับโหนดปลายทาง โหนดต้นทางจะสามารถรับข้อความ RREP จากโหนดปลายทางได้หลายครั้ง เพื่อที่โหนดนั้นจะได้มีเส้นทางของโหนดปลายทางได้หลายเส้นทาง เพื่อเป็นทางเลือกหากเส้นทางที่สั้นที่สุดใช้งานไม่ได้ ก็ยังสามารถใช้เส้นทางอื่นแทนได้ โดยเส้นทางที่สามารถทำการบันทึกลงในตารางได้นั้น โหนดถัดไป (Next hop) จะต้องไม่ซ้ำกันในตารางของเส้นทางต้องไม่ซ้ำกัน กล่าวคือ โหนดๆหนึ่งจะมีเส้นทางของโหนดปลายทางได้มากที่สุด

เท่ากับจำนวนโหนดข้างเคียงที่เชื่อมต่ออยู่กับโหนดตัวเองเท่านั้น และเส้นทางจะไม่เกิดลูปจากการเลือกเส้นทางก่อนจะส่งข้อความ RREP กลับของโหนดตัวแทน

9. การรับและส่งต่อข้อความ RREP

ขั้นตอนแรกหลังจากเมื่อโหนดได้รับข้อความ RREP จะทำการค้นหาเส้นทางเพื่อทำการส่งข้อมูลไปยังโหนดตัวก่อนหน้า (Previous hop) จากหมายเลขโหนดของ Path โดยจะเรียงหมายเลขตามลำดับการส่งแล้ว ถ้าต้องการหาโหนดต่อไปที่ต้องส่งข้อความ ให้หาตำแหน่งหมายเลขโหนดของโหนดตัวเองแล้วโหนดถัดไปจะอยู่ตำแหน่งถัดไปจากโหนดตัวเองอยู่ 1 ตำแหน่ง หลังจากนั้นจะทำการเพิ่มค่าจำนวนโหนดที่ใช้ในการส่งข้อมูลไปยัง RREP ขึ้นหนึ่ง และจะทำการสร้างเส้นทางไปข้างหน้า (Forward route) ไปยังโหนดปลายทางถ้าโหนดไม่มีการสร้างเส้นทางไปข้างหน้า แต่ถ้าเคยมีเส้นทางไปข้างหน้าอยู่โหนดจะทำการเปรียบเทียบค่าหมายเลขลำดับของโหนดปลายทางภายในข้อความกับหมายเลขลำดับของโหนดปลายทางที่มันเคยเก็บไว้ซึ่งจะถูกปรับปรุงก็ต่อเมื่อมีเหตุการณ์ดังต่อไปนี้

1) หมายเลขลำดับภายในตารางเส้นทางที่โหนดเก็บค่าอยู่นั้นถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้ (Invalid Routing table) ภายในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมาย

2) หากหมายเลขลำดับของโหนดปลายทางในข้อความ RREP มีค่ามากกว่าหมายเลขลำดับภายในตารางเส้นทาง ค่าของหมายเลขลำดับของโหนดปลายทางจะถูกคัดลอกเก็บไว้

3) หมายเลขลำดับมีค่าเท่ากัน แต่เส้นทางดังกล่าวถูกตั้งค่าให้เป็นเส้นทางที่ไม่สามารถใช้งานได้

4) หมายเลขลำดับมีค่าเท่ากัน แต่ค่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลใหม่ซึ่งได้รับมาจากข้อความ RREP มีค่าน้อยกว่าของจำนวนโหนดที่ใช้ในการส่งข้อมูลใหม่ที่เคยถูกจัดเก็บไว้ในข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายก่อนหน้า

ถ้าหากข้อมูลเส้นทางแต่ละเส้นทางต่อหนึ่งเป้าหมายเพื่อใช้ในการส่งข้อมูลไปยังโหนดปลายทางถูกสร้างหรือถูกปรับปรุงก็จะดำเนินไปตามขั้นตอนต่อไปนี้

1) เส้นทางจะถูกตั้งค่าให้เป็นเส้นทางที่ใช้ได้

2) หมายเลขลำดับของโหนดปลายทางจะถูกตั้งค่าให้เป็นค่าที่ใช้งานได้

3) โหนดตัวถัดไปในตารางเส้นทางจะถูกตั้งค่าให้เป็นค่าของหมายเลขโหนดที่ได้ทำการส่งข้อความ RREP มาสู่โหนดที่ได้รับข้อมูลดังกล่าว

4) จำนวนโหนดที่ใช้ในการส่งข้อมูลจะได้จากการเพิ่มค่าขึ้นหนึ่งค่าจากค่าที่ได้จากข้อความ RREP

5) ส่วนของค่าเวลาหมดอายุ (Expiry time) จะถูกตั้งค่าให้มีค่าเท่ากับค่าเวลาของเวลาปัจจุบันบวกกับค่าของเวลาที่ใช้พิจารณาว่าเส้นทางดังกล่าวยังใช้งานได้ (Lifetime) ที่ได้จากข้อความ RREP

6) หมายเลขลำดับของโหนดปลายทางจะเป็นค่าเดียวกับหมายเลขลำดับของโหนดปลายทางที่อยู่ภายในข้อความ RREP

เมื่อใดก็ตามที่โหนดมีการส่งข้อความ RREP ส่วนของรายการของหมายเลขโหนดที่ถูกใช้ในการส่งข้อความ RREP สำหรับโหนดปลายทางจะถูกปรับปรุงโดยการเพิ่มหมายเลขโหนดถัดไปที่จะทำการส่งข้อความ RREP ต่อและเส้นทางที่ใช้ในการส่งหมายเลข RREP กลับคือหมายเลขโหนดของเส้นทางในข้อความ RREP นั้น

10. ข้อความตรวจสอบโหนดข้างเคียง (Hello message)

โหนดอาจใช้วิธีการในการตรวจสอบการเชื่อมต่อกับระหว่างสองโหนดโดยวิธีการส่งข้อความตรวจสอบโหนดข้างเคียงแบบแพร่กระจาย ซึ่งโหนดควรจะมีการใช้งานข้อความตรวจสอบโหนดข้างเคียงเฉพาะในเส้นทางที่ใช้งานเท่านั้น โดยจะทำการส่งแบบแพร่กระจายในทุกๆช่วงเวลา HELLO_INTERVAL มิลลิวินาที ซึ่งข้อความตรวจสอบโหนดข้างเคียงที่ทำการส่งแบบแพร่กระจายนั้นจัดอยู่ในประเภทของข้อความ RREP ที่มีค่า TTL = 1 โดยที่มีการแก้ไขข้อมูลบางส่วนดังนี้

1) หมายเลขของโหนดปลายทางคือหมายเลขของโหนดที่ทำการส่งข้อความตรวจสอบโหนดข้างเคียงแพร่กระจาย
2) หมายเลขลำดับของโหนดปลายทางคือหมายเลขลำดับล่าสุดของโหนดที่ส่งแบบแพร่กระจาย
3) เวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้มีค่าจากการคำนวณของ ALLOWED_HELLO_LOSS*HELLO_INTERVAL

โหนดอาจจะตรวจสอบการเชื่อมต่อโดยคอยรับฟังข้อความที่ได้จากโหนดข้างเคียง ถ้าโหนดดังกล่าวไว้รับข้อความตรวจสอบโหนดข้างเคียงจากโหนดข้างเคียง และภายในช่วงเวลา ALLOWED_HELLO_LOSS *HELLO_INTERVAL มิลลิวินาที โหนดที่ไม่ได้รับข้อความตรวจสอบโหนดข้างเคียงจากโหนดข้างเคียงจากโหนดเดิม โหนดดังกล่าวจะสมมุติว่าเกิดการเชื่อมต่อเสียหาย (Link failure) ระหว่างตนเองกับโหนดข้างเคียงขึ้น

เมื่อโหนดได้รับข้อความตรวจสอบโหนดข้างเคียงจากโหนดข้างเคียง และโหนดดังกล่าวมีเส้นทางที่ใช้งานได้และต้องใช้งานโหนดข้างเคียงเป็นโหนดถัดไป โหนดจะเพิ่มค่าเวลาที่พิจารณาว่าเส้นทางดังกล่าวยังคงใช้งานได้ให้แก้เส้นทางนั้นๆ โดยมีค่าในการเพิ่มอย่างน้อยที่สุดคือผลบวกระหว่างค่าของ ALLOWED_HELLO_LOSS *HELLO_INTERVAL

11. การบำรุงรักษาการเชื่อมต่อระหว่างโหนด (Maintaining Local Connectivity)

โหนดแต่ละตัวที่ทำหน้าที่ในการส่งข้อมูลต่อในระบบ (โหนดที่อยู่ระหว่างเส้นทางที่ใช้งานได้) ควรจะทำการตรวจสอบการเชื่อมต่อสื่อสารไปยังโหนดถัดไปตลอดจนโหนดข้างเคียงที่เคยมีการส่งข้อความตรวจสอบโหนดข้างเคียง ภายในช่วงระยะเวลาของ ALLOWED_HELLO_LOSS*HELLO_INTERVAL

ถ้าหากว่าโหนดไม่ได้รับข้อมูลใด ๆจากโหนดถัดไปภายในช่วงเวลา ALLOWED_HELLO_LOSS *HELLO_INTERVAL สมมุติว่าเกิดการเชื่อมต่อเสียหาย และเริ่มทำการส่งข้อความ RERR ในขั้นตอนต่อไป

12. การจัดการข้อความ RERR และการลบข้อมูลเส้นทาง

โดยทั่วไปแล้วข้อความ RERR จะใช้เมื่อการเชื่อมต่อมีการเสียหายโดยจะมีกระบวนการต่อไปนี้

1) ตรวจสอบเส้นทางที่มีอยู่และเกิดผลกระทบจากเส้นทางที่เสียหาย
2) ทำการส่งข้อความ RERR ไปยัง gateway

เนื่องจากตารางเส้นทางในการค้นหาของโพรโทคอล HAC-SF นั้นเก็บเฉพาะเส้นทางของโหนดข้างเคียงและโหนด gateway ดังนั้นเมื่อมีเส้นทางที่เสียหาย โหนดที่ต้องซ่อมแซมจะมีโหนดข้างเคียงของโหนดที่เสียหายและโหนด gateway ซึ่งข้อความ RERR ส่งในรูปแบบปลายทางเดียวไป gateway และโหนดจะไม่ทำการสร้างข้อความ RERR เกินค่าของ RERR_RATELIMIT ข้อความต่อ 1 วินาที โดยโหนดจะเริ่มทำการสร้างข้อความ RERR และมีการบวนการดังต่อไปนี้

1) ถ้าโหนดตรวจพบได้ว่าการเชื่อมต่อเสียหายระหว่างตนเองกับโหนดถัดไป (Next Hop) โหนดทำการลบเส้นทางในตารางโดยจะลบข้อมูลเส้นทางของโหนดที่เสียหายนั้นและเส้นทางที่มีโหนดที่เสียหายเป็นโหนดถัดไป โหนดจะทำการสร้างข้อความ RERR และหมายเลขของโหนดที่เสียหายที่ไม่สามารถส่งข้อมูลไปถึงได้ลงในข้อความ RERR และส่งข้อความไปยัง gateway เนื่องจากโหนดทุก ๆโหนดจะมีเส้นทางไป gateway มากกกว่า 1 เส้นทาง ดังนั้นโหนดจะยังคงสามารถส่งข้อความ RERR แบบปลายทางเดียวไปยัง gateway ได้ และก่อนที่จะส่งข้อความ RERR ให้บันทึกหมายเลขของโหนดตัวเองลงใน Path IP Address ในข้อความ RERR เพื่อนำไปซ่อมแซมเส้นทางของโหนด gateway
2) ถ้าโหนดทั่วไปได้รับข้อความ RERR จะทำการเพิ่มหมายเลขโหนดของโหนดตัวเองลงใน Path IP Address ในข้อความ RERR และค้นหาเส้นทางของ gateway เพื่อส่งข้อความ RERR ต่อไป

3) ถ้า gateway ได้รับข้อความ RERR โหนดจะทำการตรวจสอบตารางเส้นทางโดยจะลบข้อมูลเส้นทางของโหนดที่เสียหายนั้นและเส้นทางที่มีโหนดที่เสียหายเป็นโหนดถัดไป (Next Hop) และเนื่องจากโพรโทคอลนี้ ตารางของโหนด gateway เก็บเส้นทางเป็นรายการหมายเลขของโหนดทั้งหมด ดังนั้นถ้ามีโหนดที่เสียหายอยู่ในรายการต้องมีการซ่อมแซม โดยจะค้นหาเส้นทางที่ต้องซ่อมแซม ถ้าพบหมายเลขของโหนดที่ส่งข้อความ RERR และโหนดที่เสียหายตามลำดับในรายการของหมายเลขโหนด แสดงว่าเส้นทางนั้นมีการส่งข้อมูลผ่านโหนดที่เสียหายจึงนำรายการของหมายเลขโหนดในข้อความ RERR มาเปลี่ยนแปลงเส้นทางส่วนที่เสียหายในตาราง คือ ตั้งแต่หมายเลขของโหนดที่ส่งข้อความ RERR จนถึงโหนด gateway

ดังนั้นโหนด gateway จะมีเส้นทางใหม่ที่สามารถส่งข้อมูลไปยังโหนดอื่น ๆ โดยไม่ผ่านโหนดที่เสียหายอีกต่อไป การซ่อมแซมของโหนดทั่วไปสามารถทำได้โดยการค้นหาเส้นทางของ gateway ใหม่ได้ เนื่องจากการนำไปใช้ทางเกษตรกรรม สมมุติมีการส่งข้อมูลการตรวจวัดทุก ๆ 10 นาที อาจจะมีการค้นหาเส้นทางของ gateway ทุก ๆ 5 นาที เพื่อให้มีเส้นทางในการส่งข้อมูลให้เป็นปัจจุบันมากที่สุด ถ้าเกิดมีความเปลี่ยนแปลงภายในเครือข่าย

บทที่ 4
ผลการวิจัยและการอภิปรายผล

บทที่ 4 เป็นการวิเคราะห์และแสดงผลการทดสอบโพรโทคอล AODV ที่ได้พัฒนาการวิจัย โดยการเปรียบเทียบโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF

4.1 การทดสอบประสิทธิภาพโพรโทคอล HAC-SF

ค่าพารามิเตอร์ต่างๆที่ใช้ในการทดสอบจะถูกพิจารณาบนสมมุติฐานที่ต้องการนำเครือข่ายเซนเซอร์ไร้สายแบบเมชไปใช้ฟาร์มเพื่อใช้ในการเก็บข้อมูลของผลผลิตทางการเกษตร โดยกำหนดให้โหนดอยู่กับที่

ในการทดสอบประสิทธิภาพของโพรโทคอล AODV ที่พัฒนาขึ้นใช้ในงานเครือข่ายเซนเซอร์ไร้สายแบบเมชจะพิจารณาจากค่า 4 ค่าดังต่อไปนี้

- ค่าเปอร์เซ็นต์การส่งสำเร็จ (Packet Delivery Ratio)

คือค่าที่บอกความสำเร็จในการส่งข้อมูล มีค่าไม่เกิน 100 เปอร์เซ็นต์ ซึ่งในการจำลองการทำงานของระบบเครือข่ายที่มีโหนดจำนวนมาก และมีการเชื่อมต่อพร้อมๆกันหลายการเชื่อมต่อ หากเกิดการชนกันของข้อมูล ค่าเปอร์เซ็นต์การส่งสำเร็จจะมีค่าลดลง

- ค่าเปอร์เซ็นการส่งไม่สำเร็จ (Packet Loss Ratio)

เป็นค่าความน่าจะเป็นที่ข้อมูล Packet หนึ่ง ๆ จะไม่ได้รับอย่างถูกต้องภายในช่วงเวลาที่แน่นอน เวลาหนึ่ง นั่นหมายความว่าหากเส้นทางใดก็ตามมี Packet Loss Ratio สูง แสดงว่าข้อมูลที่ไหลไปตามเส้นทางนั้นมีโอกาสที่จะเสียหายได้มากกว่าไหลไปตามเส้นทางที่มี Packet Loss Ratio ต่ำ

- ค่าอัตราปริมาณงาน (Throughput)

คือ ค่าของจำนวนข้อมูลที่โหนดปลายทางสามารถรับได้ทั้งหมด ต่อหนึ่งหน่วยเวลาที่สนใจ ซึ่งในช่วงเวลาที่สนใจหากข้อมูลจากโหนดต้นทาง ส่งไปไม่ถึงโหนดปลายทางย่อมส่งผลให้ค่าอัตราปริมาณงานมีค่าลดลง

- ค่าหน่วงเวลา (Delay)

คือค่าความล่าช้าในการส่งแพ็กเก็ตในเครือข่าย โดยจะเป็นค่าเฉลี่ยของค่าหน่วงเวลาที่ได้จากการทำงานในทุก ๆโหนดในเครือข่าย

วิธีการจัดวางโหนดโดยจะวางโหนดเป็นรูปคล้ายๆสี่เหลี่ยมคางหมู โดย gateway จะอยู่ปลายสามเหลี่ยมตามรูปและโหนดจะวางเรียงกระจายตัวกันออกไปไกลเท่าไรก็ยิ่งกว้างมากเท่านั้น โดยเริ่มการค้นหาเส้นทางจากโหนดที่แถวอยู่ใกล้ gateway ก่อนแล้วจึงค้นหาโหนดที่อยู่ไกลออกไป

จำลองเหมือนกับการใช้งานจริงคือ เครือข่ายมี gateway เป็นศูนย์กลาง นำโหนดไปวางในพื้นที่ที่ต้องการตรวจวัดและส่งข้อมูล และใกล้ๆ gateway เปิดสวิตซ์ให้โหนดเริ่มทำงาน โหนดจะเริ่มหาเส้นทางก่อน ซึ่งหลังจากนั้นถ้าต้องการใช้งานโหนดที่ไหน ก็นำไปวางและเปิดการใช้งาน โดยการวางโหนดจะเริ่มไกลออกจากศูนย์กลาง เพื่อให้ได้ขอบเขตของการสื่อสารที่มากขึ้น ก็เหมือนกับการจำลองที่เริ่มค้นหาเส้นทางจากโหนดใกล้ เหมือนกับการเปิกการใช้งานของโหนดนั้น โดย 1 วินาที จะมีโหนดที่ค้นหาเส้นทางทั้งหมด 3 โหนดโดยเริ่มจากโหนดใกล้ศูนย์กลางก่อน และหลังจากที่โหนดมีการค้นหาเส้นทางแล้วโหนดจะมีการส่ง Data packet ทุก ๆ 5-6 วินาทีจนจบการจำลองเครือข่าย

รูปที่ 4.1 ภาพหลักการค้นหาเส้นทาง

ในส่วนของการเปรียบเทียบโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF นั้นจะมีการกำหนดพารามิเตอร์พื้นฐานสำหรับจำลองการทำงานของเครือข่ายดังแสดงในตารางที่ 4.1

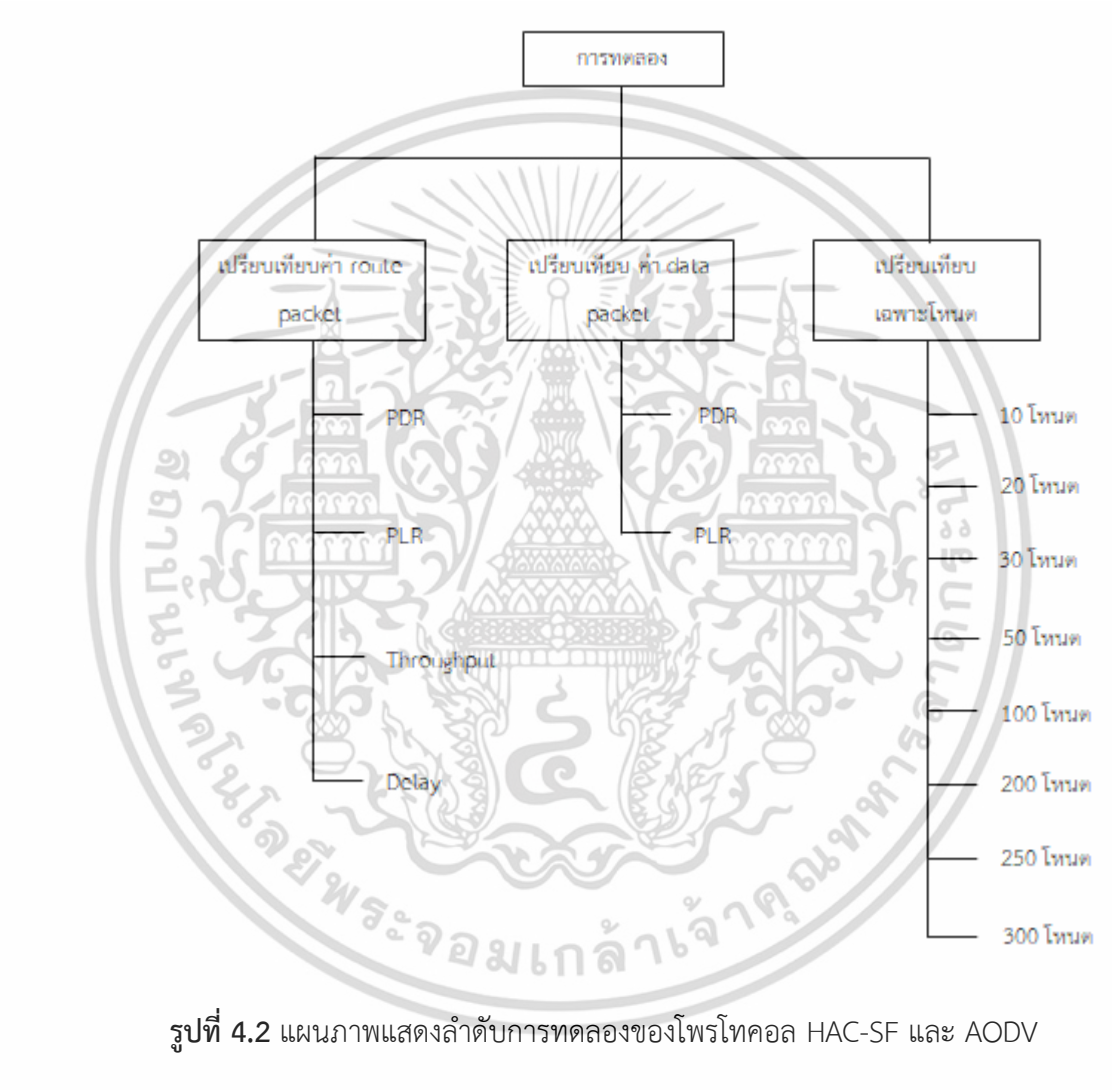
ตารางที่ 4.1 พารามิเตอร์พื้นฐานในการทดสอบการทำงานของเครือข่าย

พารามิเตอร์	ค่าที่ใช้งาน
Mac Layer	Mac/802.11b
Network Interface	Phy/WirlessPhy/802.11b
Routing protocol	HAC-SF
No. of nodes	10, 20, 30, 50, 100, 200, 250, 300
HELLO_INTERFACE	1 seconds(Defaulted AODV)
Number of simulation runs	10 times
Simulation time	300
X dimension	200 m
Y dimension	200 m
Packet type	CBR packet
Connection	4
Rx range	300 m
Tx range	300 m
Rx power	25 dBm
Tx power	25 dBm

จากตารางที่ 4.1 เป็นพารามิเตอร์พื้นฐานเพื่อใช้ในการจำลองการทำงานเปรียบเทียบประสิทธิภาพของโพรโทคอลเมื่อมีขนาดมากขึ้น ในพื้นที่ทดสอบมากขึ้น แต่ระยะห่างเท่าเดิม (200 เมตร * 200 เมตร) โดยการกำหนดให้มีการส่งข้อความตรวจสอบโหนดข้างเคียงในทุก ๆ 1 วินาที ซึ่งเป็นค่าดั้งเดิมของโพรโทคอล AODV และมีจำนวนโหนดสูงสุดที่ใช้ในการทดสอบมีค่า 200 โหนด เนื่องจากในที่นี้ทำการพิจารณาเฉพาะเครือข่ายที่มีการใช้งานหมายเลขโหนดแบบ 8 บิต (หมายเลข ID ที่สามารถทำได้บนโหนดจริง) เพื่อให้ได้พื้นที่ของเครือข่ายมากที่สุด ซึ่งจะสามารถทำการจัดเก็บหมายเลขที่แตกต่างกันได้ที่สุด 256 หมายเลข และการทดสอบจะใช้โพรโทคอล AODV แบบดั้งเดิม โดยใช้ทดสอบบนเครื่อง sever ที่ใช้ CPU Intel® Xeon® Processor E5-2620 [15]

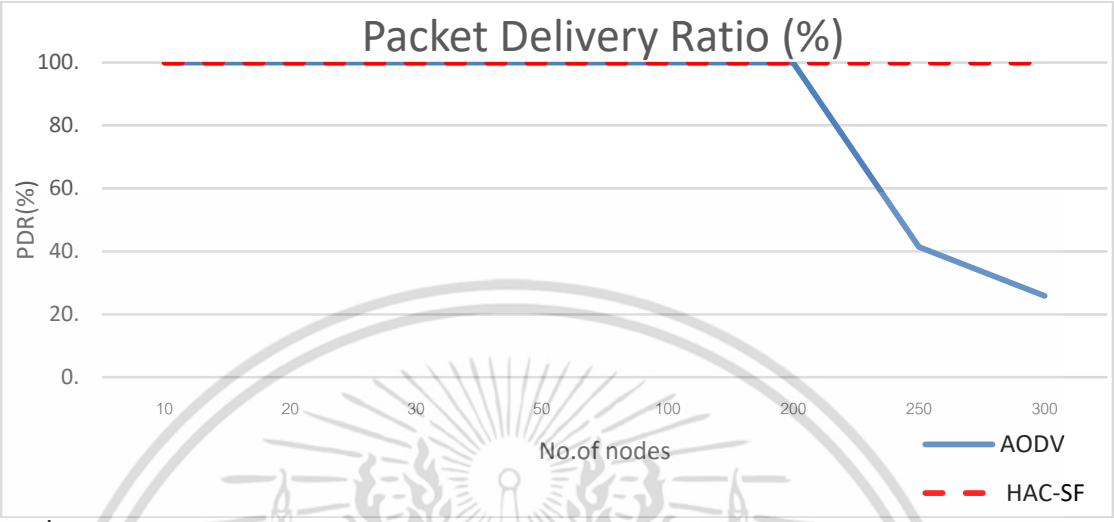
4.2 ผลการทดลอง

จากการใช้พารามิเตอร์ในตาราง 4.1 ในการทดสอบประสิทธิภาพ โดยผลการวิจัยได้แบ่งออกเป็นการหาค่า PDR PLR Throughput และ Delay ของ route packet และค่า PDR และ PLR ของ data packet ซึ่งแบ่งการทดสอบตามจำนวนโหนดในเครือข่ายออกเป็น 10, 20, 30, 50, 100, 200, 250 และ 300 โหนด ตามแผนภาพ ดังรูปที่ 4.2



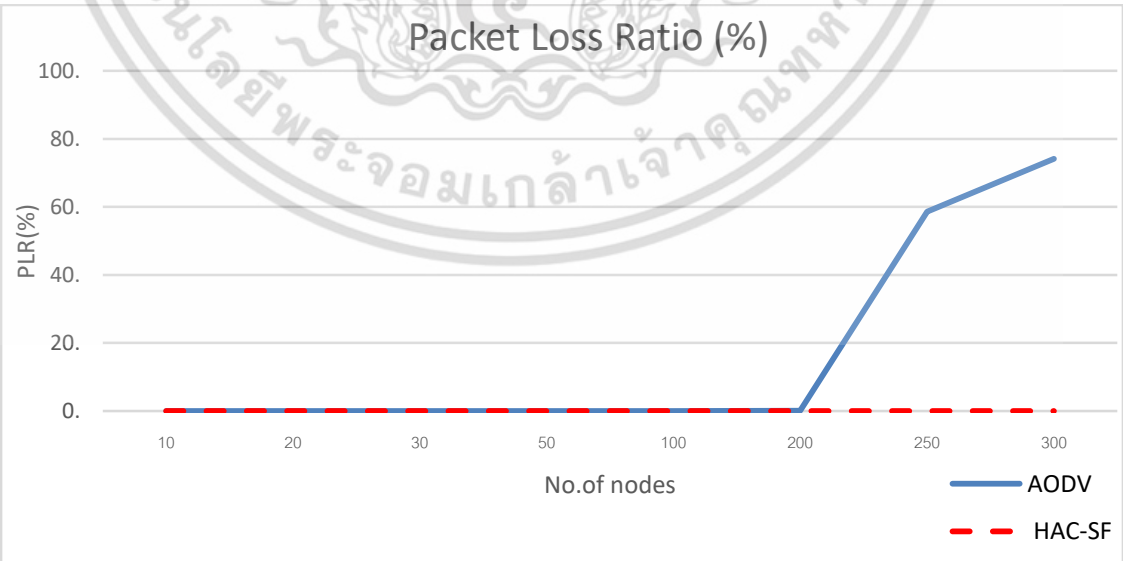
รูปที่ 4.2 แผนภาพแสดงลำดับการทดลองของโพรโทคอล HAC-SF และ AODV

ซึ่งได้นำเสนอออกมาเป็นรูปแบบกราฟเพื่อให้เห็นความแตกต่างของโพรโทคอล AODV และ HAC-SF อย่างชัดเจนดังนี้



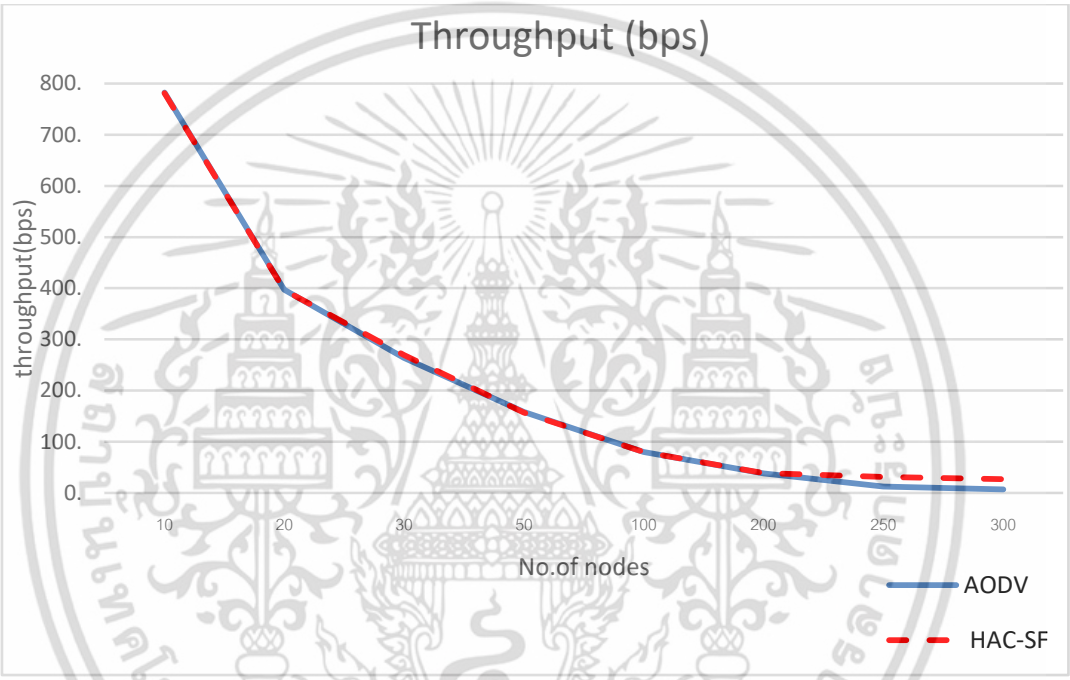
รูปที่ 4.3 กราฟความสัมพันธ์ระหว่างอัตราความสำเร็จในการส่งข้อมูลของ route packet กับจำนวนโหนดในเครือข่าย

จากรูปที่ 4.3 เป็นกราฟแสดงความสัมพันธ์ระหว่างอัตราความสำเร็จในการส่งข้อมูลของ route packet กับจำนวนโหนดในเครือข่ายโดยจะเห็นได้ว่า โพรโทคอล HAC-SF นั้นมีค่าประมาณ 100% ตั้งแต่จำนวนโหนด 10 ถึง 300 โหนด แต่โพรโทคอล AODV จะมีค่าลดลงหลังจากมีจำนวนโหนดตั้งแต่ 200 โหนดเป็นต้นไป



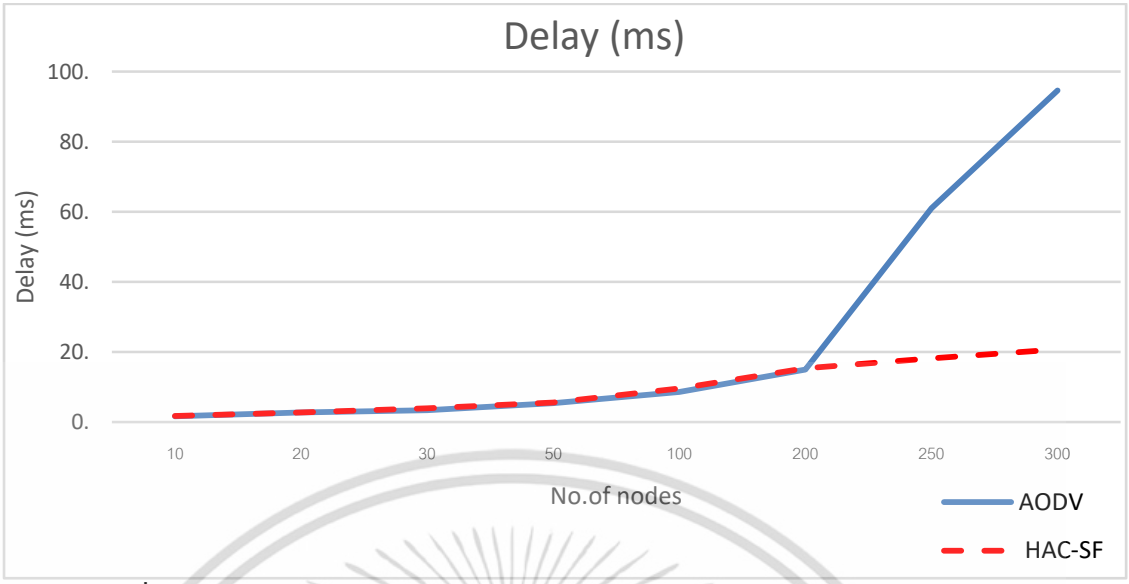
รูปที่ 4.4 กราฟความสัมพันธ์ระหว่างอัตราความล้มเหลวในการส่งข้อมูลของ route packet กับจำนวนโหนดในเครือข่าย

จากรูปที่ 4.4 เป็นกราฟแสดงความสัมพันธ์ระหว่างอัตราความล้มเหลวในการส่งข้อมูลของ route packet กับจำนวนโหนดในเครือข่าย โดยค่าการสูญหายของของโพรโทคอล HAC-SF จำนวนโหนดตั้งแต่ 10 ถึง 300 โหนด แทบไม่มีข้อความสูญหายเลย ส่วนโพรโทคอล AODV มีข้อความสูญหายอย่างเห็นได้ชัดเจนเมื่อมีจำนวนโหนด 200 โหนดขึ้นไป โดยจำนวนโหนดที่ 250 โหนด มีอัตราข้อความสูญหายอยู่ที่ 58.56% และที่จำนวนโหนด 300 โหนด มีอัตราข้อความสูญหายอยู่ที่ 74.14%



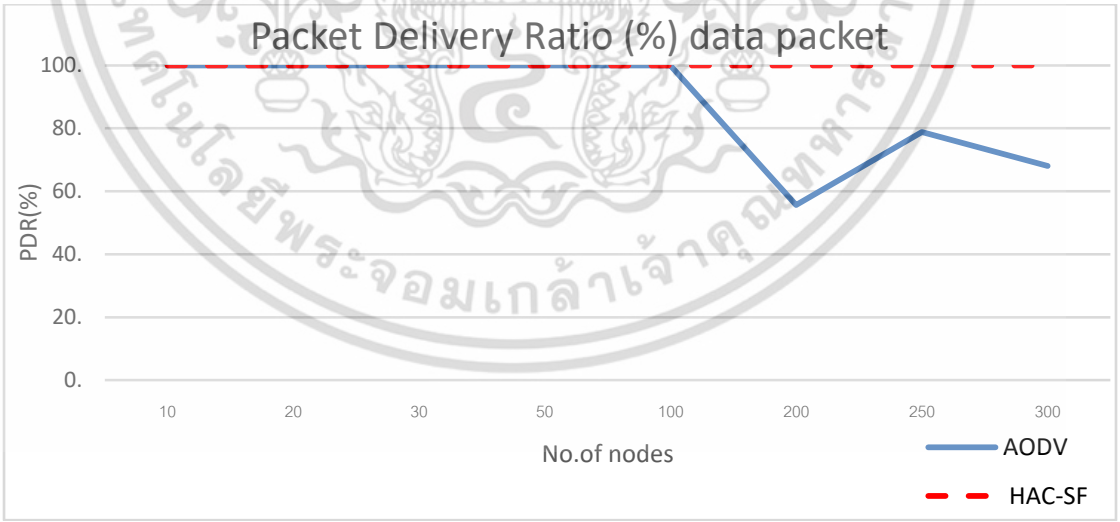
รูปที่ 4.5 กราฟแสดงค่าของจำนวนข้อมูลที่โหนดปลายทางสามารถรับได้ทั้งหมดต่อหนึ่งหน่วยเวลา

จากรูปที่ 4.5 เป็นกราฟแสดงค่าของจำนวนข้อมูลที่โหนดปลายทางสามารถรับได้ทั้งหมดต่อหนึ่งหน่วยเวลาซึ่งอยู่ในรูปกราฟชันแนวดิ่ง โดยโพรโทคอลทั้งสองแบบนั้นมีค่าเฉลี่ยที่ได้จากการทดลองใกล้เคียงกันมาก โดยโพรโทคอล HAC-SF นั้นมีค่ามากกว่าโพรโทคอล AODV อยู่เพียงเล็กน้อย



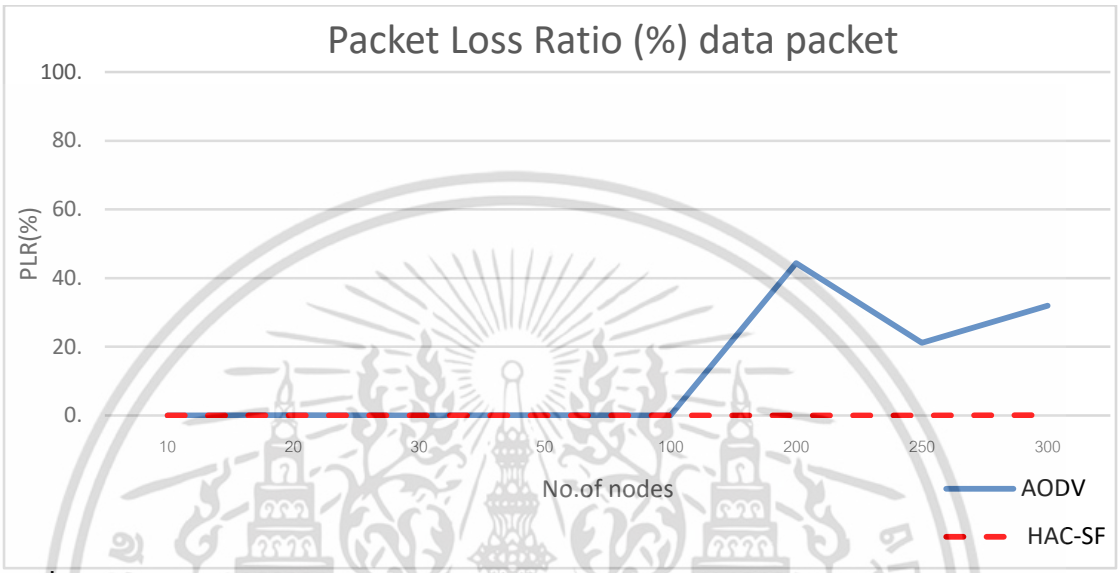
รูปที่ 4.6 กราฟแสดงความสัมพันธ์ระหว่างค่าหน่วงเวลากับจำนวนโหนดในเครือข่าย

จากรูปที่ 4.6 เป็นกราฟแสดงความสัมพันธ์ระหว่างค่าหน่วงเวลากับจำนวนโหนดในเครือข่ายที่อยู่ในรูปกราฟชันขึ้น โดยจากการทดลองนั้นโพรโทคอลทั้งสองเมื่อทดสอบได้ค่าเฉลี่ยออกมาแล้ว โพรโทคอล AODV มีค่าหน่วงเวลาโดยเฉลี่ยน้อยกว่าเล็กน้อยเมื่อมีจำนวนโหนดตั้งแต่ 10 ถึง 200 โหนด แต่เมื่อมีจำนวนโหนดมากกว่า 200 โหนดขึ้นไป ค่าหน่วงเวลาของโพรโทคอล AODV มีค่าเฉลี่ยเพิ่มขึ้นมากกว่าเดิมหลายเท่า ส่วนโพรโทคอล HAC-SF มีค่าหน่วงเวลาเพิ่มขึ้นตามปกติ



รูปที่ 4.7 กราฟความสัมพันธ์ระหว่างอัตราความสำเร็จในการส่งข้อมูลของ data packet กับจำนวนโหนดในเครือข่าย

จากรูปที่ 4.7 เป็นกราฟแสดงความสัมพันธ์ระหว่างอัตราความสำเร็จในการส่งข้อมูลของ data packet กับจำนวนโหนดในเครือข่ายโดยจะเห็นได้ว่า โพรโทคอล HAC-SF นั้นมีค่า PDR ประมาณ 100% ตั้งแต่ทีจำนวนโหนดตั้งแต่ 10 ถึง 300 โหนด แต่โพรโทคอล AODV จะมีค่า PDR ลดลง เมื่อมีจำนวนโหนดตั้งแต่ 200 โหนดขึ้นไป



รูปที่ 4.8 กราฟความสัมพันธ์ระหว่างอัตราความล้มเหลวในการส่งข้อมูลของ data packet กับจำนวนโหนดในเครือข่าย

จากรูปที่ 4.8 เป็นกราฟแสดงความสัมพันธ์ระหว่างอัตราความล้มเหลวในการส่งข้อมูลของ data packet กับจำนวนโหนดในเครือข่าย โดยค่าความล้มเหลวของโพรโทคอล HAC-SF ตั้งแต่มีจำนวนโหนด 10 ถึง 300 โหนด ข้อความแทบจะไม่สูญหายเลย ส่วนโพรโทคอล AODV มีข้อความสูญหายอย่างเห็นได้ชัดเมื่อมีจำนวนโหนดตั้งแต่ 100 โหนดขึ้นไป

ตารางที่ 4.2 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของโพรโทคอล AODV แบบดั้งเดิม กับโพรโทคอล HAC-SF ที่มีจำนวนโหนด 10 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	100	100	เท่ากัน
PLR (%)	0	0	เท่ากัน
Throughput (bps)	782.63	781.36	น้อยกว่า 1.27
Delay (ms)	1.65	1.78	ช้ากว่า 0.13
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	66	75	มากกว่า 9
PDR (%) ของ data packet	100	100	เท่ากัน
PLR (%) ของ data packet	0	0	เท่ากัน

จากตารางที่ 4.2 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนดเป็น 10 โหนด แสดงให้เห็นว่าค่า PDR และ PLR ของโพรโทคอล HAC-SF มีค่าเท่ากัน ส่วนค่า Throughput และ Delay นั้นโพรโทคอล AODV มีค่าดีกว่า ในด้านจำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดในโพรโทคอล HAC-SF จะมีผลการทดลองที่แย่กว่าไม่มาก เนื่องจากจำนวนโหนดยังไม่มากพอให้เห็นผลการทดลองที่ชัดเจน ส่วนค่า PDR และ PRL ของ data packet ของทั้งสองโพรโทคอลได้ผลลัพธ์ที่ดีเท่ากัน

ตารางที่ 4.3 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของโพรโทคอล AODV แบบดั้งเดิม กับโพรโทคอล HAC-SF ที่มีจำนวนโหนด 20 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	100	100	เท่ากัน
PLR (%)	0	0	เท่ากัน
Throughput (bps)	397.03	396.62	น้อยกว่า 0.41
Delay (ms)	2.71	2.72	ช้ากว่า 0.01
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	160	166	มากกว่า 6
PDR (%) ของ data packet	99.94	100	ดีกว่า 0.06
PLR (%) ของ data packet	0.07	0	ดีกว่า 0.06

จากตารางที่ 4.3 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนดเป็น 20 โหนด แสดงให้เห็นว่าค่า PDR และ PLR มีค่าเท่ากัน ส่วนค่า Throughput และ Delay นั้นโพรโทคอล AODV มีค่าดีกว่า ในด้านจำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดของโพรโทคอล HAC-SF จะมีผลการทดลองที่ดีขึ้นกว่า 10 โหนด ส่วนค่า PDR และ PLR ของ data packet นั้นโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อยู่เล็กน้อย

ตารางที่ 4.4 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF ที่มีจำนวนโหนด 30 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	100	100	เท่ากัน
PLR (%)	0	0	เท่ากัน
Throughput (bps)	263.66	269.86	มากกว่า 6.20
Delay (ms)	3.42	3.90	ช้ากว่า 0.48
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	262	258	น้อยกว่า 4
PDR (%) ของ data packet	100	100	เท่ากัน
PLR (%) ของ data packet	0	0	เท่ากัน

จากตารางที่ 4.4 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนด เป็น 30 โหนด แสดงให้เห็นว่าค่า PDR และ PLR ของ route packet และ data packet มีค่าเท่ากัน ส่วนค่า Throughput นั้น โพรโทคอล HAC-SF มีค่าดีกว่า แต่ค่า Delay ยังช้ากว่าโพรโทคอล AODV แบบดั้งเดิม ส่วนค่าจำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดในโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV

ตารางที่ 4.5 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF ที่มีจำนวนโหนด 50 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	100	100	เท่ากัน
PLR (%)	0	0	เท่ากัน
Throughput (bps)	158.35	156.97	น้อยกว่า 1.38
Delay (ms)	5.41	5.55	ช้ากว่า 0.14
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	491	445	น้อยกว่า 46
PDR (%) ของ data packet	99.97	100	ดีกว่า 0.03
PLR (%) ของ data packet	0.03	0	ดีกว่า 0.03

จากตารางที่ 4.5 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนด เป็น 50 โหนด แสดงให้เห็นว่าแม้ว่าค่า Throughput และ Delay นั้นโพรโทคอล AODV มีค่าดีกว่า แต่จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดในโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV ส่วนค่า PDR และ PLR ของ data packet นั้นโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อยู่เล็กน้อย

ตารางที่ 4.6 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF ที่มีจำนวนโหนด 100 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	100	100	เท่ากัน
PLR (%)	0	0	เท่ากัน
Throughput (bps)	80.76	80.14	น้อยกว่า 0.6162
Delay (ms)	8.63	9.60	ช้ากว่า 0.9716
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	1,320	920	น้อยกว่า 400
PDR (%) ของ data packet	99.99	100	ดีกว่า 0.01
PLR (%) ของ data packet	0.01	0	ดีกว่า 0.01

จากตารางที่ 4.6 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนด เป็น 100 โหนด แสดงให้เห็นว่าค่า PDR และ PLR มีค่าเท่ากัน ส่วนค่า Throughput และ Delay นั้นโพรโทคอล AODV มีค่าดีกว่า แต่จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดในโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV โหนด ส่วนค่า data packet นั้นโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อยู่เล็กน้อย

ตารางที่ 4.7 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF ที่มีจำนวนโหนด 200 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	99.94	100	ดีกว่า 0.06
PLR (%)	0.06	0	ดีกว่า 0.06
Throughput (bps)	38.12	38.64	มากกว่า 0.52
Delay (ms)	15.03	15.32	ช้ากว่า 0.29
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	2,278	1,889	น้อยกว่า 389
PDR (%) ของ data packet	55.69	99.99	ดีกว่า 44.30
PLR (%) ของ data packet	44.31	0.01	ดีกว่า 44.30

จากตารางที่ 4.7 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนด เป็น 200 โหนด แสดงให้เห็นว่าค่า PDR และ PLR ของโพรโทคอล HAC-SF มีค่าดีกว่าเล็กน้อย ส่วนค่า Throughput นั้นค่าโพรโทคอล HAC-SF มีผลการทดลองที่ดีกว่า แต่ค่า Delay ยังช้ากว่าโพรโทคอล AODV แบบดั้งเดิม ส่วนค่าจำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนดในโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV โหนด ส่วนค่า PDR และ PLR ของ data packet นั้นโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อย่างเห็นได้ชัด

ตารางที่ 4.8 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF ที่มีจำนวนโหนด 250 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	41.44	99.98	ดีกว่า 58.54
PLR (%)	58.56	0.02	ดีกว่า 58.54
Throughput (bps)	12.41	30.99	มากกว่า 18.58
Delay (ms)	61.06	18.16	เร็วกว่า 42.90
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	23,626	2,381	น้อยกว่า 21,245
PDR (%) ของ data packet	78.83	99.98	ดีกว่า 21.15
PLR (%) ของ data packet	21.17	0.02	ดีกว่า 21.15

จากตารางที่ 4.8 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนดเป็น 250 โหนด แสดงให้เห็นว่าค่า PDR PLR Throughput และ Delay จำนวนเส้นทางเฉลี่ยใน routing table และค่า PDR และ PLR ของ data packet ของโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อย่างชัดเจน

ตารางที่ 4.9 แสดงตารางการเปรียบเทียบผลที่ได้จากการทำงานของ AODV แบบดั้งเดิมกับ HAC-SF โดยใช้โหนดจำนวน 300 โหนด

Value	AODV แบบดั้งเดิม	HAC-SF	ผลการเปรียบเทียบ
PDR (%)	25.86	99.96	ดีกว่า 74.10
PLR (%)	74.14	0.04	ดีกว่า 74.10
Throughput (bps)	6.80	26.99	มากกว่า 20.18
Delay (ms)	94.67	20.72	เร็วกว่า 73.95
จำนวนเส้นทางเฉลี่ยใน routing table ของแต่ละโหนด(route)	34,153	2,862	น้อยกว่า 31,291
PDR (%) ของ data packet	68.03	99.97	ดีกว่า 31.94
PLR (%) ของ data packet	31.97	0.03	ดีกว่า 31.94

จากตารางที่ 4.9 แสดงผลเปรียบเทียบผลที่ได้จากการทดสอบระหว่างโพรโทคอล AODV แบบดั้งเดิมกับโพรโทคอล HAC-SF ที่มีจำนวนโหนดเป็น 300 โหนด แสดงให้เห็นว่าค่า PDR PLR Throughput และ Delay จำนวนเส้นทางเฉลี่ยใน routing table และค่า PDR และ PLR ของ data packet ของโพรโทคอล HAC-SF มีผลลัพธ์ที่ดีกว่าโพรโทคอล AODV อย่างชัดเจน

4.3 อภิปรายผล

ตารางที่ 4.10 สรุปการอภิปรายผล

การทดลอง	ผลโดยรวม
PDR (%)	โพรโทคอล HAC-SF มีค่าที่ดีกว่า
PLR (%)	โพรโทคอล HAC-SF มีค่าที่ดีกว่า
Throughput (bps)	เมื่อจำนวนโหนดที่มากขึ้น โพรโทคอล HAC-SF จะมีค่าที่ดีกว่าโพรโทคอล AODV
Delay (ms)	เมื่อจำนวนโหนดที่มากขึ้น โพรโทคอล HAC-SF จะมีค่าที่ดีกว่าโพรโทคอล AODV
เส้นทาง routing table	เมื่อจำนวนโหนดที่มากขึ้น โพรโทคอล HAC-SF จะมีค่าที่ดีกว่าโพรโทคอล AODV
data packet	โพรโทคอล HAC-SF มีค่าที่ดีกว่า

จากผลการทดลองจะเห็นได้ว่าโพรโทคอล HAC-SF มีประสิทธิภาพกว่าโพรโทคอล AODV อย่างเห็นได้ชัดเจนเมื่อโหนดมากขึ้น โดยในโหนดที่ 300 โพรโทคอล HAC-SF มี อัตราการรับส่งแพ็กเก็ตข้อมูลมากกว่า 31.93% อัตราการรับส่งแพ็กเก็ตค้นหาเส้นทางมากกว่าถึง 74.1% ค่าความล่าช้าน้อยกว่า 73.95 ms ค่าอัตราปริมาณงานมากกว่า 20.18 bps เมื่อเทียบกับโพรโทคอล AODV ซึ่งสาเหตุเกิดจากโพรโทคอล AODV จะเก็บเส้นทางเดียวไปยัง gateway แต่โพรโทคอล HAC-SF เก็บหลายเส้นทางแต่ไม่เกินจำนวนโหนดข้างเคียงของโหนดตัวเองทำให้ความล้มเหลวในการรับส่งข้อมูลน้อยกว่า โดยมีข้อสังเกตเพิ่มเติมคือ gateway ของโพรโทคอล AODV มีเส้นทางของโหนดในเครือข่ายไม่ครบทั้งหมดและมีระยะเวลาในการจำลองการทำงานที่นานกว่า เนื่องจากมีการทำงานการจัดการขอข้อความในเครือข่ายที่มากกว่าโพรโทคอล AODV จากการได้ลองทดลองเพิ่มเติมที่ 400 โหนดเวลาการจำลองการทำงานของโพรโทคอล HAC-SF มีระยะเวลา 5 ชั่วโมง ส่วนโพรโทคอล AODV มีระยะเวลา 14 ชั่วโมง โดยจะเห็นได้ว่าโพรโทคอล AODV มีระยะเวลาในการจำลองมากกว่าเกือบ 3 เท่า

บทที่ 5
บทสรุป ปัญหาและข้อเสนอแนะ

5.1 บทนำ

ในบทนี้จะกล่าวถึงบทสรุปและข้อเสนอแนะของการดำเนินงานทำวิจัย ปัญหาและอุปสรรคต่างๆ ที่เกิดขึ้นขณะที่ทำงานวิจัย และท้ายที่สุดจะกล่าวถึงรายละเอียดและข้อเสนอแนะแก่ผู้ที่สนใจที่จะนำงานวิจัยนี้ไปพัฒนาต่อไป

5.2 บทสรุปของการทำงานวิจัย

งานวิจัยนี้เป็นการนำเสนอวิธีการในการพัฒนาโพรโทคอล เพื่อให้สามารถนำมาใช้ในเครือข่ายแอดฮอกแบบเข้าสู่ศูนย์กลางทางด้านการเกษตร โดยมีปรับปรุงการค้นหาเส้นทางให้มีขอบเขตในการค้นหาที่น้อยลงและลดข้อความที่รับส่งในเครือข่าย แต่ยังคงสามารถค้นหาเส้นทางและส่งข้อมูลได้อย่างมีประสิทธิภาพมากขึ้น โดยโพรโทคอล HAC-SF จะเก็บเส้นทางของโหนดข้างเคียงทั้งหมดและโหนด gateway มีการเก็บเส้นทางของปลายทางได้หลายเส้นทาง แต่ไม่เกินกว่าจำนวนของโหนดข้างเคียง เพื่อลดระยะเวลาในการค้นหาเส้นทางในกรณีเส้นทางที่ใช้งานหมดอายุการใช้งานหรือเกิดความเสียหาย ซึ่งโพรโทคอล HAC-SF จะสามารถใช้งานได้อย่างมีประสิทธิภาพในเครือข่ายที่มี gateway เพียงตัวเดียวเป็นโหนดศูนย์กลาง และโหนดศูนย์กลางจะต้องมีประสิทธิภาพมากกว่าโหนดทั่วไป เนื่องจากต้องรับส่งข้อมูลที่ใช้ในการควบคุมการทำงานของโหนดทั้งหมดในเครือข่าย

จากการทดลองทั้งจำนวนโหนดในเครือข่ายตั้งแต่ 10, 20, 30, 50, 100, 200, 250 และ 300 โหนด สรุปได้ว่า ค่า PDR PLR ของ route packet และ data packet นั้น โพรโทคอล HAC-SF มีผลการทดสอบที่ดีกว่า แต่โพรโทคอล HAC-SF มีค่า Delay ที่มากกว่าโพรโทคอล AODV เมื่อมีจำนวนโหนดที่น้อย เพราะมีการหน่วงเวลาเพื่อรอส่งข้อความซ้ำกรณีที่ไม่ได้รับข้อความยืนยันว่าข้อความนั้นส่งถึงโหนดปลายทาง โดยโพรโทคอล AODV ไม่มีการรอหรือตรวจสอบว่าข้อความนั้นส่งถึงหรือไม่ในบางกรณี แต่เมื่อมีจำนวนโหนดเพิ่มมากขึ้น ค่าความล้มเหลวในการส่งข้อมูลของโพรโทคอล AODV มากขึ้นทำให้มีค่าความล่าช้าเพิ่มขึ้นหลายเท่าตัว ส่วนค่า Throughput และจำนวนเส้นทางในตารางเส้นทางนั้นโพรโทคอล AODV จะมีผลการทดลองที่ดีกว่าถ้ามีจำนวนโหนดในเครือข่ายที่น้อย แต่ถ้ามีจำนวนโหนดที่มากขึ้น โพรโทคอล HAC-SF มีผลการทดสอบที่ดีกว่า ซึ่งเกิดจากโพรโทคอล AODV จะเก็บเส้นทางของโหนดทุกโหนดที่ส่งข้อความค้นหาเส้นทางผ่านโหนดตัวเอง แม้ว่าเส้นทางนั้นจะไม่ได้มีการใช้งานและเส้นทางของไปยังโหนด gateway แต่โพรโทคอล HAC-SF เก็บหลายเส้นทางแต่ไม่เกินจำนวนโหนดข้างเคียงของโหนดตัวเองและเส้นทางของโหนดข้างเคียงเท่านั้น ทำให้เส้นทางของโพรโทคอล AODV มีมากกว่าโพรโทคอล HAC-SF และในการทดสอบจำลองเครือข่ายโพรโทคอลมีระยะเวลาที่ใช้จำลองมากกว่าโพรโทคอล HAC-SF เพราะโพรโทคอล AODV มีจำนวนข้อความค้นหาเส้นทางและข้อมูลที่มากกว่า

5.3 ปัญหาและอุปสรรคของการทำงานวิจัย

เนื่องจากทรัพยากรในงานวิจัยนี้มีจำกัด ทำให้ไม่สามารถทำการจำลองเครือข่ายได้เมื่อมีจำนวนโหนดตั้งแต่ 300 โหนดเป็นต้นไป ถ้ามีทรัพยากรที่เพิ่มมากขึ้นจะสามารถทำให้การทดสอบในงานวิจัยนี้มีประสิทธิภาพมากขึ้น

5.4 ข้อเสนอแนะ

เนื่องจากเครือข่าย ad hoc มีการใช้พลังงานที่มากและโพรโทคอลในงานวิจัยนี้นี้ยังไม่มีการจัดการเรื่องพลังงาน ดังนั้นควรมีการปรับปรุงพัฒนาโพรโทคอลให้ประหยัดพลังงานมากขึ้นและความปลอดภัยของการส่งข้อมูลที่เพิ่มมากขึ้นอีก

เอกสารอ้างอิง

[1] สำนักงานที่ปรึกษาการเกษตรต่างประเทศ. 2548. **เกษตรกรรมความแม่นยำสูง.** [Online]. Available: http://www.thaieurope.net/เกษตรกรรมความแม่นยำสูง/. เข้าถึงเมื่อวันที่ 7 ม.ค. 60.

[2] สุกันยา ซาอุรัมย์, **Wireless Mesh Networks**. วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม มหาวิทยาลัยเทคโนโลยีสุรนารี, 2553.

[3] วาริส จันอิ, **การศึกษาถึงผลกระทบของการแทรกสอดช่องสัญญาณร่วมที่มีต่อสมรรถนะของเครือข่ายเมชไร้สายที่ใช้สายอากาศเก่ง.** วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม มหาวิทยาลัยเทคโนโลยีสุรนารี, 2554.

[4] ภาคภูมิ มโนยุทธ, **การปรับปรุงโพโทคอลเอโอดีวีสำหรับเครือข่ายเซ็นเซอร์ไร้สาย.** วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์, 2555.

[5] Wireless Services for Education มหาวิทยาลัยศรีนครินทรวิโรฒ. 2547. **มาตรฐาน IEEE 802.11**. [Online]. Available: http:// wise.swu.ac.th/Default.aspx? tabid=3440. เขาถึงเมื่อวันที่ 10 ก.พ. 60.

[6] ฉัตรชัย และคณะ. 2555. **Network simulator**. [Online]. Available: https://sites.google.com/site/ns2porkaermcalxngrabkheruxkhay/khwam-hmay-laea-kha-saphth-thi-keiywkhxng. เขาถึงเมื่อวันที่ 19 ก.พ. 60.

[7] วาริส จันอิ, **โพรโทคอลการค้นหาเส้นทางเอโอดีวีที่คำนึงถึงการใช้พลังงานด้วยวิธีการข้ามระดับชั้น Cross Layer**. วิทยานิพนธ์ วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์, 2556.

[8] สุกันยา ซาอุรัมย์, **การออกแบบตำแหน่งโหนดสำหรับเครือข่ายเมชไร้สายภายในตัว.** วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม มหาวิทยาลัยเทคโนโลยีสุรนารี, 2553.

[9] I.F. Akyildiz, X. Wang and W. Wang, **Wireless Mesh Networks: a Survey**. Computer Networks, Volume 47, Issue 4, 15 March 2005, pp 445–487.

[10] A. Zakrzewska, L. Koszałka, I. Poźniak-Koszałka, **Performance Study of Routing Protocols for Wireless Mesh Networks**. 19th International Conference on Systems Engineering (2008), pp 331 – 336.

[11] Q. Feng, Z. Cai, J. Yang, X. Hu, **A Performance Comparison of the Ad hoc Network Protocols**. Second International Workshop on Computer Science and Engineering (2009), pp 293-297.

เอกสารอ้างอิง (ต่อ)

[12] V.K. Taksande, Dr.K.D. Kulat, **Performance Comparison of DSDV, DSR, AODV Protocol with IEEE 802.11 MAC for Chain Topology for Mobile Ad hoc Network using NS-2**. IJCA Special Issue on 2nd National Conference-Computing, Communication and Sensor Network (CCSN) (3):26-31, 2011.

[13] F. Ijaz, A. A. Siddiqui, B. Kwan Im, C. Lee, **Remote management and control system for LED based plant factory using ZigBee and Interne.** Advanced Communication Technology (ICACT) 14th International Conference (2012), pp 942-946.

[14] A. Arya, J. Singh, **Comparative Study of AODV, DSDV and DSR Routing Protocols in Wireless Sensor Network Using NS-2 Simulator**. (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4), 2014, pp 5053-5056.

[15] Intel Thailand . **โปรเซสเซอร์ Intel.** [Online]. Available:: https://ark.intel.com/th/products/64594/Intel-Xeon-Processor-E5-2620-15M-Cache-2_00-GHz-7_20-GTs-Intel-QPI. /. เขาถึงเมื่อวันที่ 17 พ.ค. 60.

ภาคผนวก

ภาคผนวก ก
ตารางผลการวิจัย

ตารางที่ ก.1 อัตราความสำเร็จในการส่งข้อมูลของโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
AODV(%)	100	100	100	100	100	99.94	41.44	25.86
HAC-SF (%)	100	100	100	100	100	100	99.98	99.96

ตารางที่ ก.2 อัตราความล้มเหลวในการส่งข้อมูลของโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
AODV(%)	0	0	0	0	0	0.06	58.56	74.14
HAC-SF (%)	0	0	0	0	0	0	0.02	0.04

ตารางที่ ก.3 ค่าของจำนวนข้อมูลที่โหนดปลายทางสามารถรับได้ทั้งหมดต่อหนึ่งหน่วยเวลาของโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
AODV (bps)	782.63	397.03	263.66	158.35	80.76	38.12	12.41	6.80
HAC-SF (bps)	781.36	396.62	269.86	156.97	80.14	38.64	30.99	26.99

ตารางที่ ก.4 ความสัมพันธ์ระหว่างค่าหน่วงเวลากับจำนวนโหนดในเครือข่ายของโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
AODV(%)	1.65	2.71	3.42	5.41	8.63	15.03	61.06	94.67
HAC-SF (%)	1.78	2.72	3.90	5.55	9.60	15.32	18.16	20.72

ตารางที่ ก.5 Data packet ของโพรโทคอล AODV ในโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
จำนวน data packet ที่รับทั้งหมด	534	1,491	2,340	3,359	8,345	9,310	12,545	11,706
จำนวน data packet ที่ส่งทั้งหมด	534	1,492	2,340	3,360	8,346	16,718	15,913	17,206
PDR (%)	100	99.94	100	99.97	99.99	55.69	78.83	68.03
PLR (%)	0	0.06	0	0.03	0.01	44.31	21.17	31.97

ตารางที่ ก.6 Data packet ของโพรโทคอล HAC-SF ในโหนดที่ 10-300

จำนวนโหนด	10	20	30	50	100	200	250	300
จำนวน data packet ที่รับทั้งหมด	854	1,924	3,086	5,042	9,992	19,019	23,064	26,841
จำนวน data packet ที่ส่งทั้งหมด	854	1,924	3,086	5,042	9,992	19,020	23,068	26,850
PDR (%)	100	100	100	100	100	99.99	99.98	99.97
PLR (%)	0	0	0	0	0	0.01	0.02	0.03

ตารางที่ ก.7 routing table ของโหนดที่ 10-300

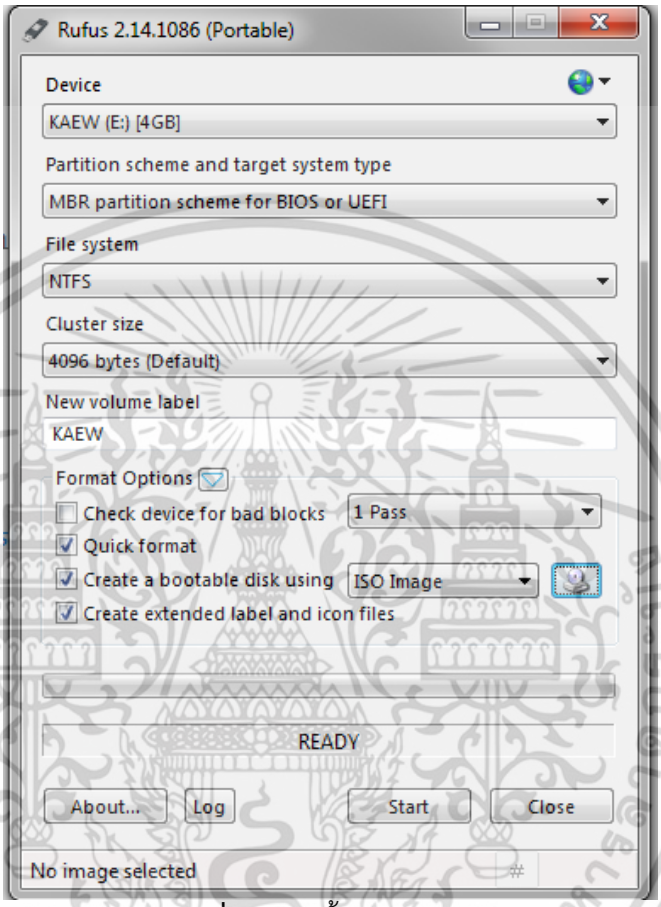
จำนวนโหนด	10	20	30	50	100	200	250	300
AODV (route)	66	160	262	491	1,320	2,278	23,626	34,153
HAC-SF (route)	75	166	258	445	920	1,889	2,381	2,862

ภาคผนวก ข

วิธีติดตั้ง Ubuntu

วิธีติดตั้ง Ubuntu 16.04

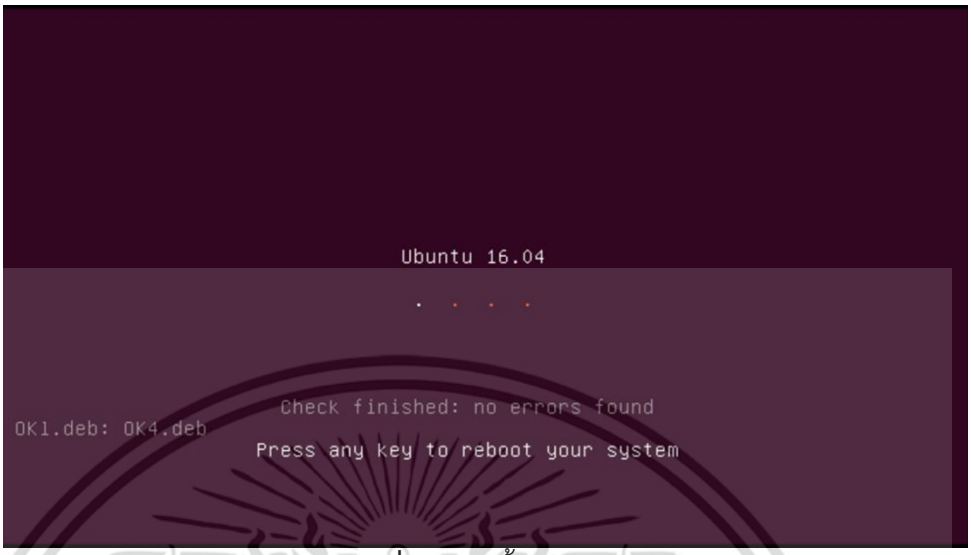
1. ดาวน์โหลดโปรแกรม Ubuntu จากเว็บ https://www.ubuntu.com/download/desktop
2. เตรียม USB และ ดาวน์โหลดโปรแกรม Rufus ได้ที่ https://rufus.akeo.ie/?locale=th_TH เพื่อสามารถบูทและติดตั้ง Ubuntu ผ่าน USB ได้โดยตั้งค่าตามนี้



รูปที่ ข.1 ติดตั้งโปรแกรม Rufus

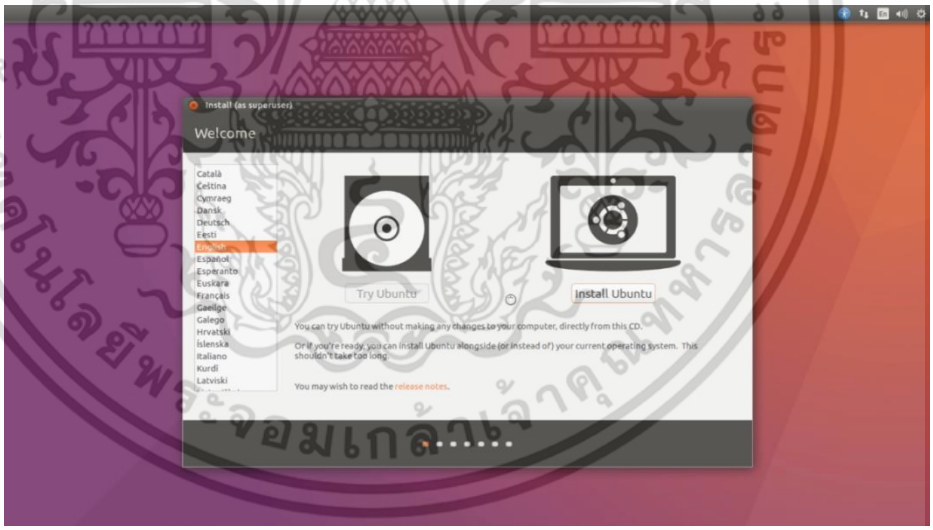
โดยคลิกที่รูป CD ข้างๆ ISO Image เพื่อเลือกไฟล์ ISO มาใส่จากนั้นกดปุ่มเริ่มเพื่อให้โปรแกรมทำงานและเริ่ม format USB และคัดลอกไฟล์ระบบปฏิบัติการ

3. เปิดคอมขึ้นมาใหม่แล้วบูทผ่าน USB ที่ได้ทำการเตรียมไว้



รูปที่ ข.2 ติดตั้ง Ubuntu

4. เลือกภาษาที่ต้องการติดตั้ง และ กด install Ubuntu

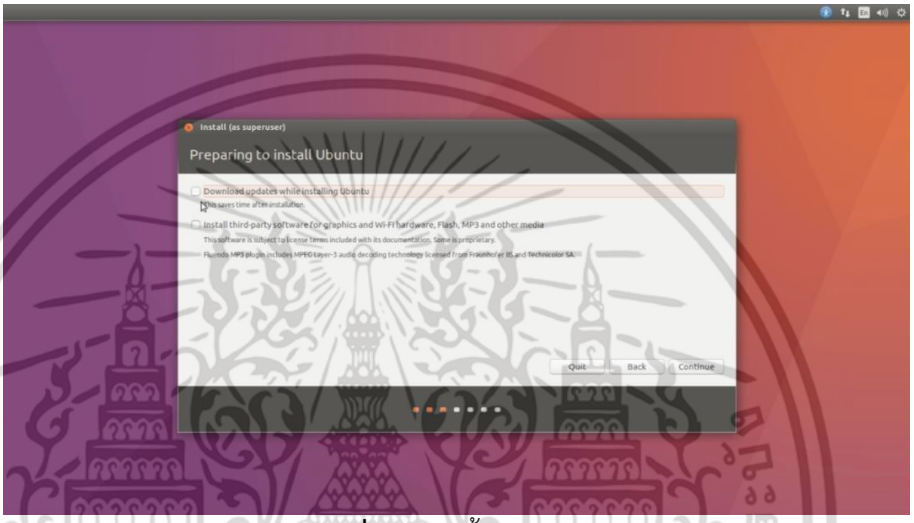


รูปที่ ข.3 ติดตั้ง Ubuntu

5. ตัวเลือกสำหรับติดตั้ง

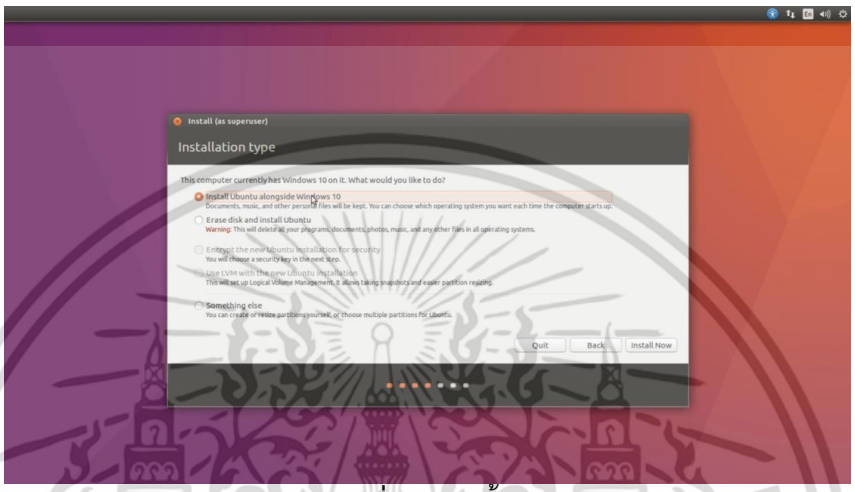
ตัวเลือกแรก Download updates จะเลือกเพื่อให้ระบบอัพเกรดไปพร้อมการติดตั้ง แต่สามารถอัพเกรดภายหลังจากติดตั้งได้ เพื่อความรวดเร็วในการติดตั้ง

ตัวเลือกที่สอง Install third-party software ต้องการติดต้อง software เพิ่มเติมอื่นเข้าไปด้วยไหม เช่น โปรแกรมเล่นเพลง เล่นหนัง เป็นต้น ซึ่งโปรแกรมดังกล่าว ไม่ใช้โปรแกรมของ Ubuntu โดยตรง



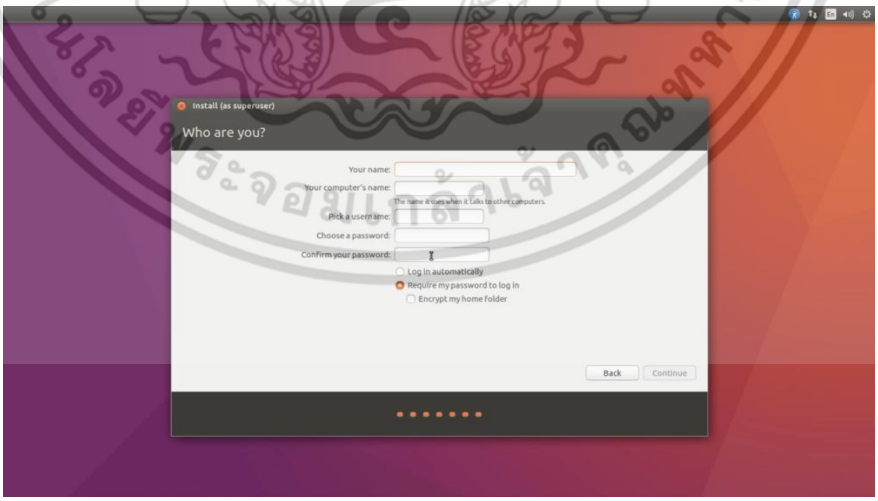
รูปที่ ข.4 ติดตั้ง Ubuntu

6. เมื่อใช้ฮาร์ดดิสก์ใหม่และไม่ได้มีการติดตั้งระบบปฏิบัติการอื่น ใช้ควบคู่กันบนฮาร์ดดิสก์นี้ จะใช้ตัวเลือกแรกคือ ลบดิสก์และติดตั้ง Ubuntu ลงไป โดยการแบ่งพาร์ทิชั่นบนฮาร์ดดิสก์นั้น ระบบจะเป็นตัวจัดการให้เองทั้งหมด แต่ถ้าหากต้องการแบ่งพาร์ทิชั่นเอง ให้เลือกที่ตัวเลือกอื่นๆ โดยเมื่อกดต่อไปแล้วก็เลือกเวลาตามประเทศที่อยู่อาศัย และแป้นพิมพ์ที่ต้องการ



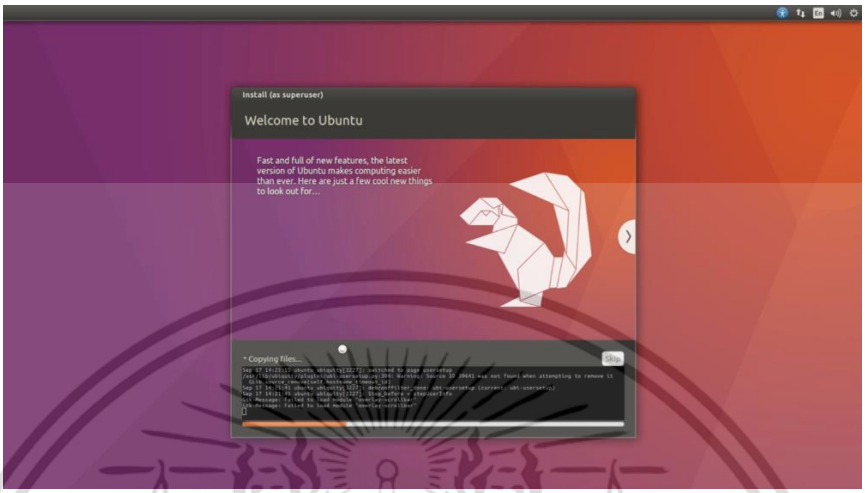
รูปที่ ข.5 ติดตั้ง Ubuntu

7. จากนั้นทำการใส่ชื่อผู้ใช้ ชื่อเครื่องคอมพิวเตอร์ รหัสผ่าน และรูปแบบการเข้าใช้เครื่อง เลือกที่เข้าระบบอัตโนมัติหากไม่ต้องการ ใส่รหัสผ่านตอนเข้าเครื่องหรือต้องการ ให้เลือกต้องการรหัสผ่านตอนเข้าเครื่อง



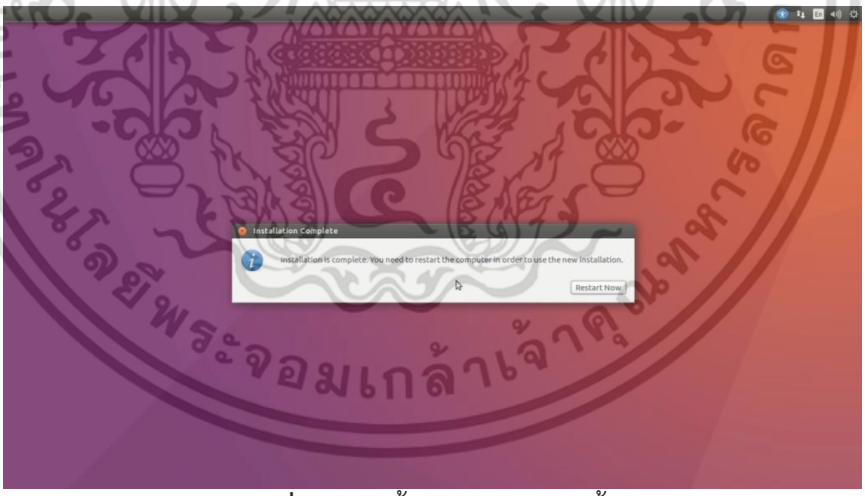
รูปที่ ข.6 ติดตั้ง Ubuntu

8.จากนั้นระบบจะเริ่มทำการคัดลองไฟล์และติดตั้งระบบปฏิบัติการ Ubuntu ลงบนฮาร์ดดิสก์



รูปที่ ข.7 ติดตั้ง Ubuntu

9.ระบบจะแจ้งเตือนให้รีสตาร์ทเมื่อเสร็จสิ้น



รูปที่ ข.8 ติดตั้ง Ubuntu เสร็จสิ้น

ภาคผนวก ค
วิธีติดตั้ง NS-3

วิธีการติดตั้ง โปรแกรม NS-3 (NS-3 Simulator Version 3.26)

1. Download ไฟล์ติดตั้งโปรแกรม ns-3 จากเว็บไซต์ https://www.nsnam.org/ns-3-26/download/

   เข้าสู่เว็บไซต์ https://www.nsnam.org

   คลิกที่ “Download ns-3.26 code”

รูปที่ ค.1 ดาวน์โหลดไฟล์ติดตั้ง โปรแกรม NS-3

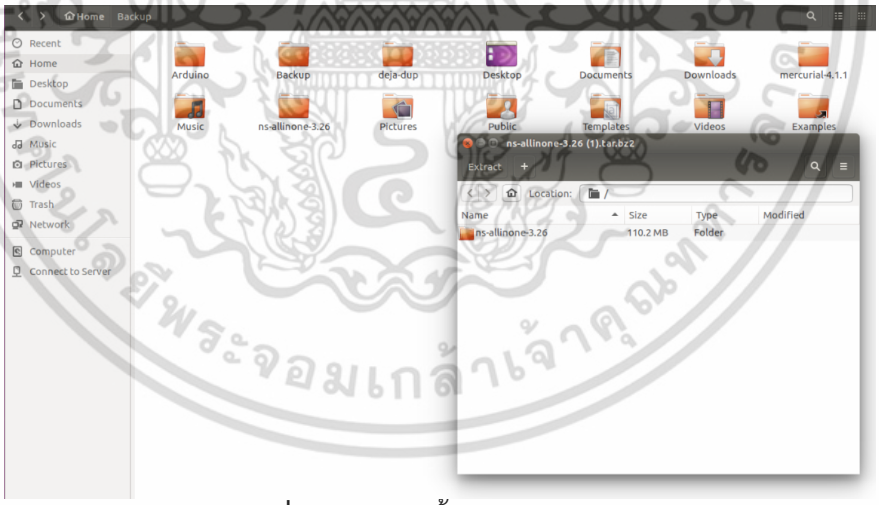
จากนั้นคลิกให้คลิกที่ “Download”

รูปที่ ค.2 ดาวน์โหลดไฟล์ติดตั้ง โปรแกรม NS-3

จะได้หน้าต่าง Download ขึ้นมา ให้คลิกที่ “ns-allinone-3.26” เพื่อทำการ Download ลงเครื่อง

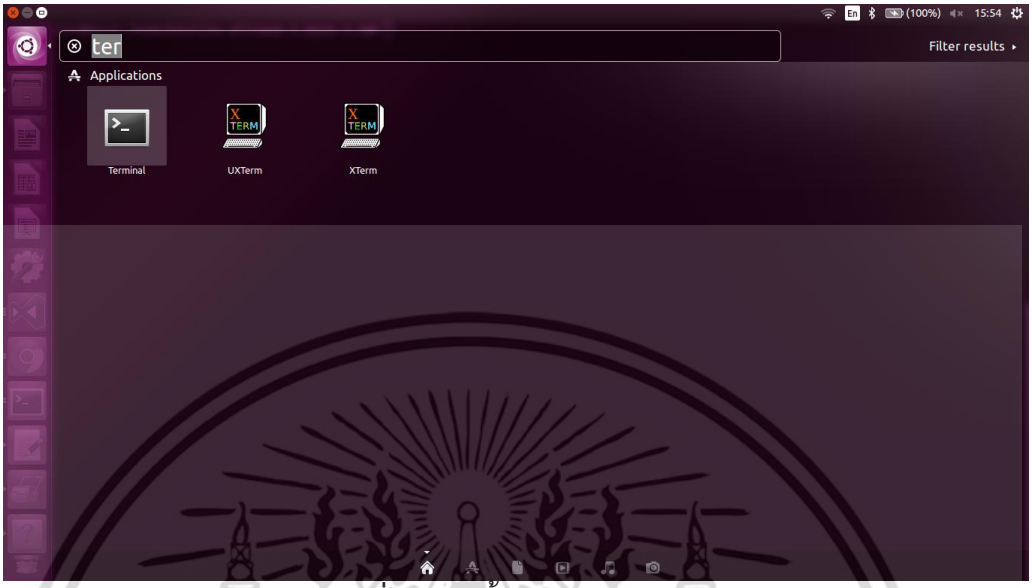
รูปที่ ค.3 ดาวน์โหลดไฟล์ติดตั้ง โปรแกรม NS-3

เมื่อ download เสร็จสิ้นแล้วจะได้ File ns-3 ดังนี้



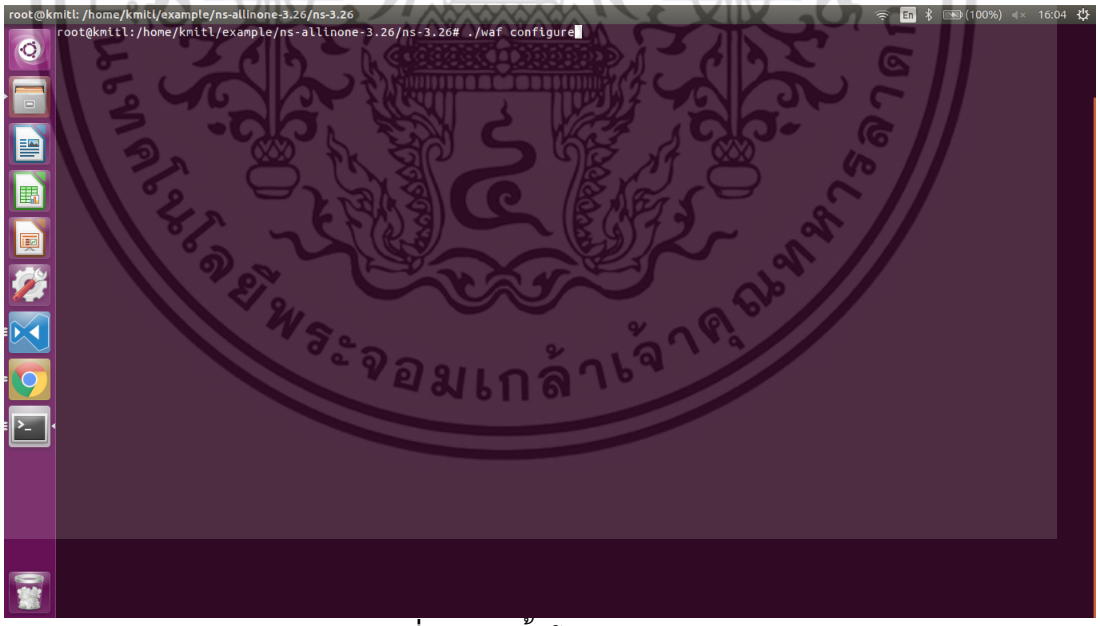
รูปที่ ค.4 ไฟล์ติดตั้ง โปรแกรม NS-3

2. เปิด terminal ขึ้นมา



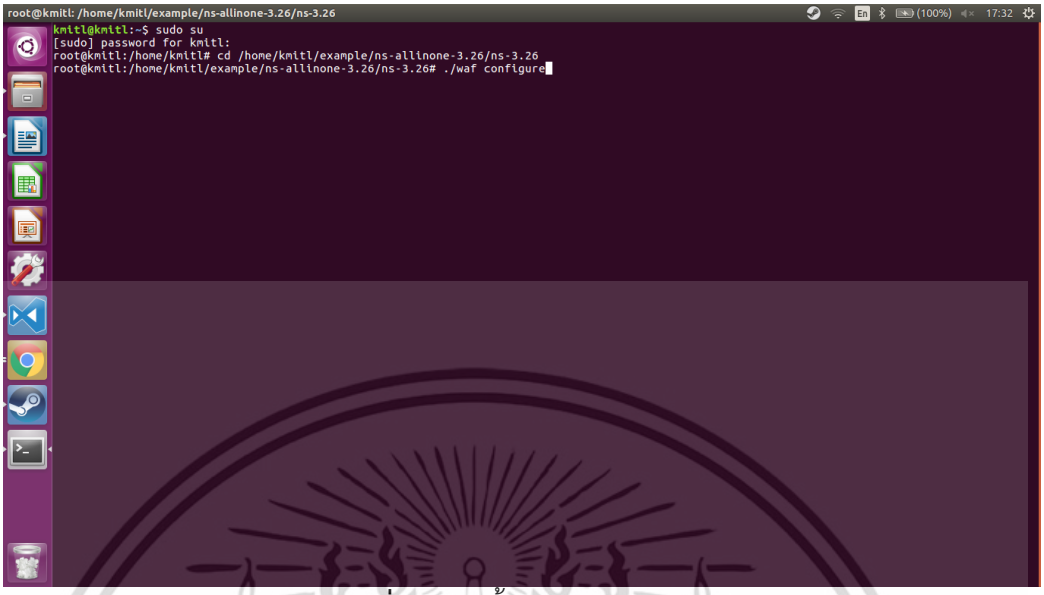
รูปที่ ค.5 ติดตั้ง โปรแกรม NS-3

จากนั้น ใช้ terminal เปิดโฟลเดอร์ตัวติดตั้ง ns-3 ที่ได้ทำการแตกไฟล์ไว้



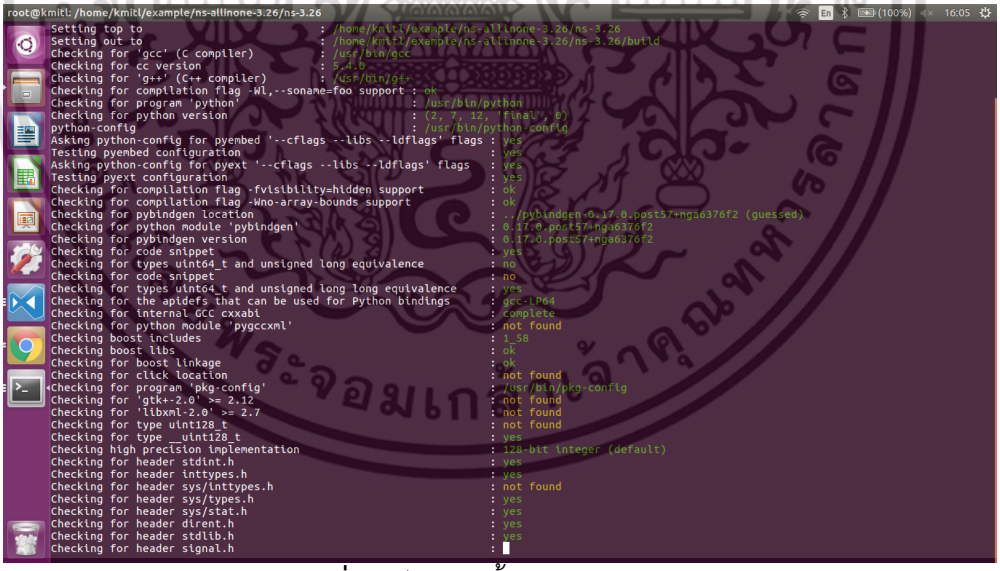
รูปที่ ค.6 ติดตั้ง โปรแกรม NS-3

3. ใช้คำสั่ง ./waf เพื่อทำการตั้งค่าและติดตั้งไฟล์ต่างๆจากไฟล์ ns-3.26



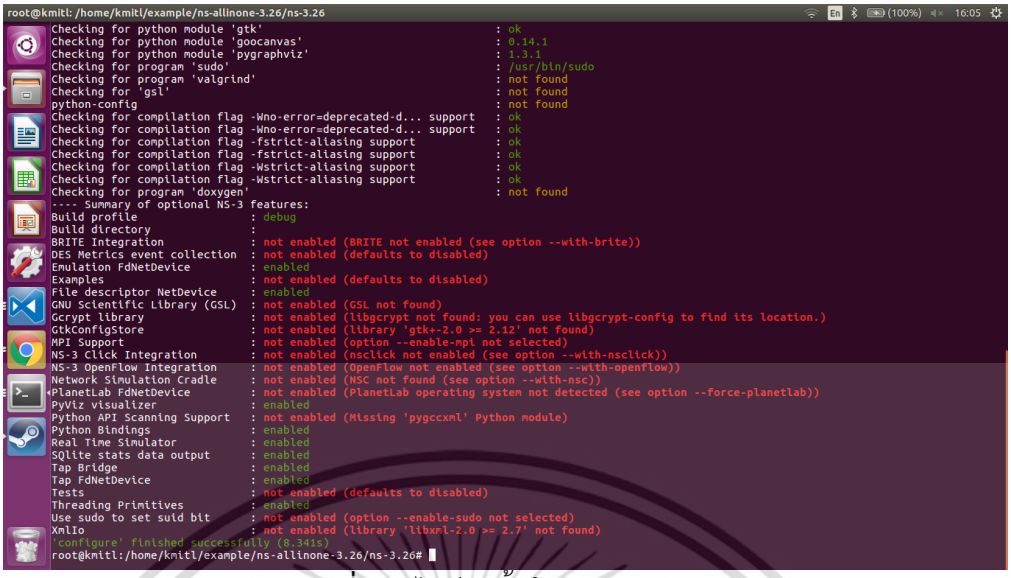
รูปที่ ค.7 ติดตั้ง โปรแกรม NS-3

4. ใช้คำสั่ง ./waf configure เพื่อ Config กำหนดค่าต่างๆ



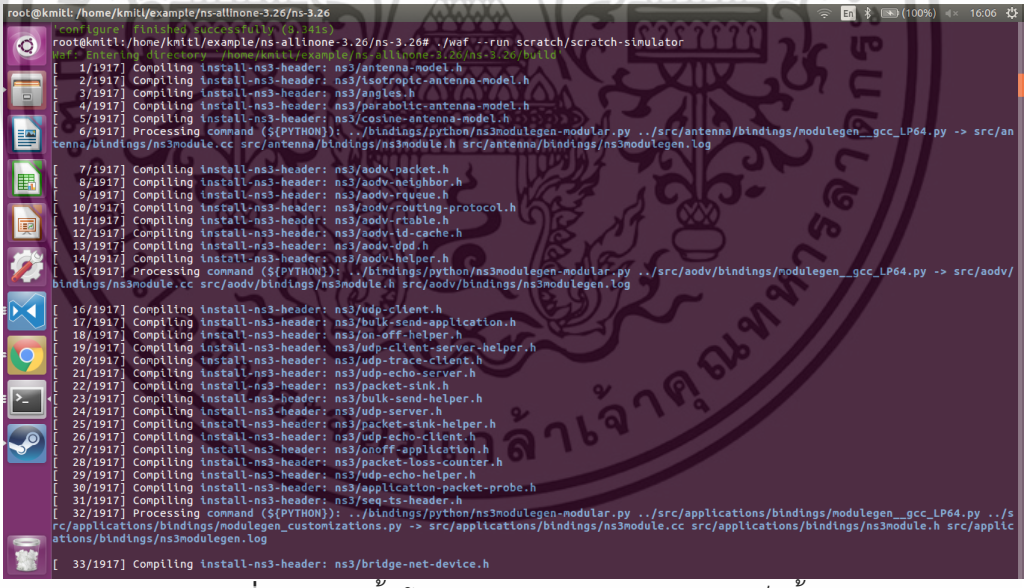
รูปที่ ค.8 ไฟล์ติดตั้ง โปรแกรม NS-3

เมื่อทำการ config เสร็จสิ้น



รูปที่ ค.9 ไฟล์ติดตั้ง โปรแกรม NS-3

5. ใช้คำสั่ง “./waf –run (ชื่อไฟล์ที่ต้องการ run)”สำหรับทำการ Run ในการใช้งาน ns-3 ในกานรันครั้งแรกจะทำการคอมไพล์ไฟล์ทั้งหมดใน ns-3



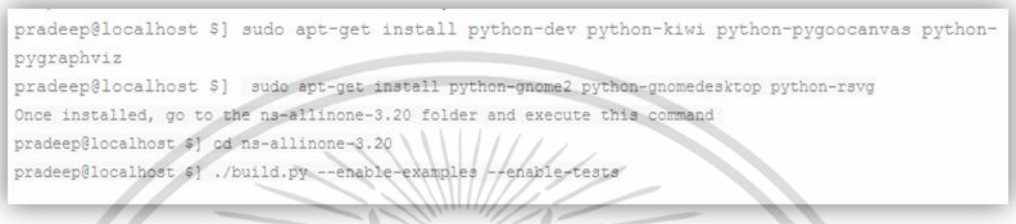
รูปที่ ค.10 ติดตั้ง โปรแกรม NS-3 ลง Ubuntu เสร็จสิ้น

ภาคผนวก ง
วิธีติดตั้ง PyViz

วิธีติดตั้ง PyViz ใน NS-3

วิธีการติดตั้ง โปรแกรม PyViz (หรือโมดูล visualizer) ใน ns3 ทำให้ภาพที่จำลองแสดงออกมาเป็นกราฟฟิก เพื่อง่ายต่อจำลองและทำให้มองเห็นภาพมากยิ่งขึ้น เป็นโมดูลที่แสดงให้เห็นว่าการส่งออกในรูปแบบของภาพเคลื่อนไหวและมีแพ็กเก็ตบางส่วนที่จำเป็นจะต้องมีการติดตั้ง

1. เปิด terminal แล้วเรียกใช้คำสั่งดังนี้ เพื่อทำการติดตั้ง visualize



รูปที่ ง.1 ติดตั้ง โปรแกรม PyViz

เมื่อ ทำการติดตั้ง visualize เรียบร้อยแล้ว

2. ให้ใช้คำสั่ง “./waf --run myprogram –vis” หรือ “./waf --run myprogram –visualize” เพื่อใช้ในการ run visualize

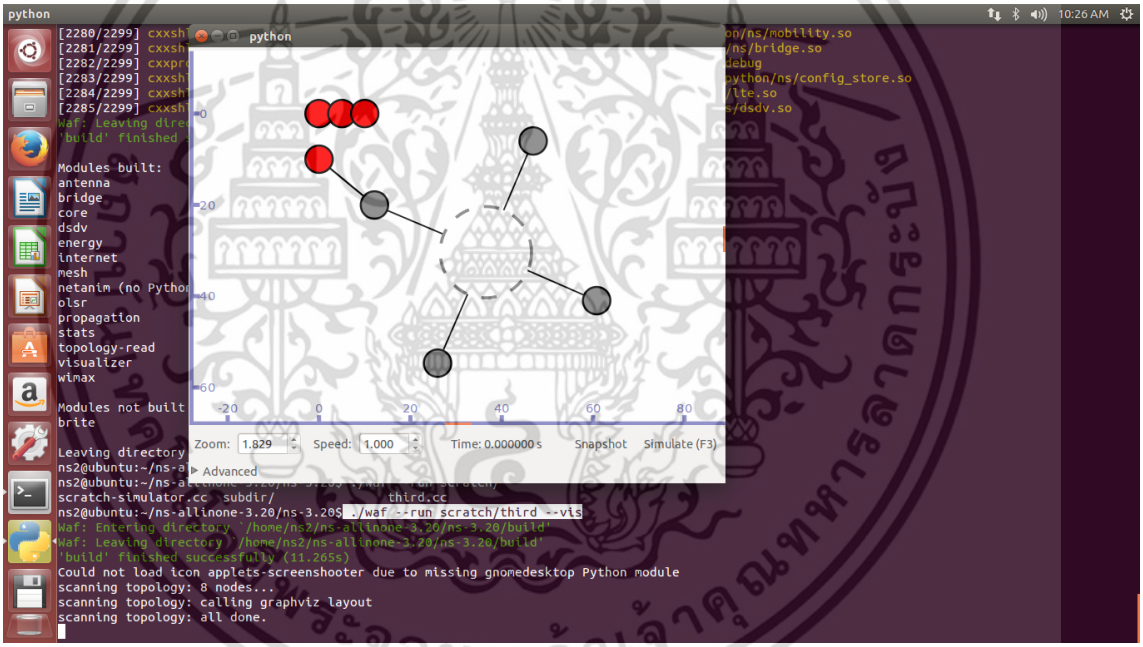
รูปที่ ง.2 ติดตั้ง โปรแกรม PyViz

หากต้องการดูผลลัพธ์ใน Visualizer ขั้นตอนให้ใช้คำสั่ง ตามรูป...ด้านล่างนี้

```
pradeep@localhost $] cd ns-allinone-3.20/ns-3.20
pradeep@localhost $] ./waf --run examples/tutorial/third --vis
or
pradeep@localhost $] ./waf --run examples/tutorial/third --visualize
Here is the screenshot of the visualize.
```

รูปที่ ง.3 ติดตั้ง โปรแกรม PyViz

ตัวอย่างการแสดงผลของ NS-3 ในโปรแกรม PyViz



รูปที่ ง.4 ติดตั้ง โปรแกรม PyViz เสร็จสิ้นพร้อมสำหรับการใช้งาน

ภาคผนวก จ

วิธีใช้งาน Flow Monitor

Model Description

source code สำหรับโมดูลใหม่จะใช้ directory src/flow-monitor โดย Flow Monitor module มีเป้าหมายคือการจัดหาระบบที่มีความยืดหยุ่นเพื่อวัดประสิทธิภาพของโพรโทคอลเครือข่าย โมดูลจะใช้โพรบที่ติดตั้งไว้ในโหนดเครือข่ายเพื่อติดตามการส่งแพ็กเก็ตโดยโหนดและจะวัดค่าของพารามิเตอร์ แพ็กเก็ตจะถูกแบ่งตามการไหลของแพ็กเก็ตซึ่งการไหลนั้นจะถูกกำหนดตามลักษณะโพรบสถิติที่เก็บรวบรวมได้จากการไหลในแต่ละครั้งสามารถนำออกมาในรูปแบบ xml ได้ นอกจากนี้ผู้ใช้ยังสามารถเข้าถึงโพรบได้โดยตรงเพื่อขอสถิติแบบเจาะจงเกี่ยวกับการไหลแต่ละครั้ง

Design

Flow Monitor module ได้รับการออกแบบในรูปแบบโมดูลาร์ สามารถขยายได้โดยการ subclassing ns3 :: FlowProbe และ ns3 :: FlowClassifier

Scope and Limitations

ปัจจุบันโพรบและclassifiersสามารถใช้ได้ในipv4และipv6 โดยโพรบแต่ละตัวจะแบ่งแพ็กเก็ตเป็น4จุด

- เมื่อแพ็กเก็ตส่ง(SendOutgoing IPv[4,6] traces)
- เมื่อแพ็กเก็ตส่งต่อ(UnicastForward IPv[4,6] traces)
- เมื่อแพ็กเก็ตรับ(LocalDeliver IPv[4,6] traces)
- เมื่อแพ็กเก็ตดรอป (Drop IPv[4,6] traces)

เนื่องจากมีการติดตั้งแพ็กเก็ตในระดับ IP การส่งข้อมูลที่เกิดจากชั้น transport layer(เช่น TCP)จะถูกมองว่าโพรบเป็นแพ็กเก็ตใหม่ โดยจะ Tag เพิ่มลงในแพ็กเก็ต (ns3::Ipv[4,6]FlowProbeTag) ซึ่งแท็กจะมีข้อมูลของพื้นฐานแพ็กเก็ตที่เป็นประโยชน์สำหรับการจัดหมวดหมู่ของแพ็กเก็ต ข้อมูลที่เก็บรวบรวมสำหรับการไหลแต่ละครั้งได้แก่

- timeFirstTxPacke เมื่อแพ็กเก็ตแรกถูกส่ง
- timeLastTxPacke เมื่อแพ็กเก็ตสุดท้ายถูกส่ง
- timeFirstRxPacket เมื่อแพ็กเก็ตแรกได้รับโดยโหนดสุดท้าย
- timeLastRxPacket เมื่อแพ็กเก็ตสุดท้ายได้รับ
- delaySum ผลรวมความล่าช้า
- jitterSum ผลรวมความล่าช้าที่ถูกกำหนดใน RFC 3393
- txBytes, txPackets จำนวนไบต์ที่ส่ง/แพ็กเก็ตที่ส่งทั้งหมด
- rxBytes, rxPackets จำนวนไบต์ที่ได้รับ/แพ็กเก็ตที่ส่งทั้งหมด
- lostPackets จำนวนแพ็กเก็ตที่คาดว่าสูญหาย (ห้ามรายงานเกิน 10 วินาที)
- timesForwarded จำนวนครั้งที่มีการส่งต่อแพ็กเก็ต

- delayHistogram, jitterHistogram, packetSizeHistogram ฮิตโตแกรมสำหรับค่าความล่าช้า ไซต์แพ็กเก็ต
- packetsDropped, bytesDropped จำนวนแพ็กเก็ตและไบต์ที่สูญหาย

ซึ่งสถิติเหล่าแสดงออกมาในรูปแบบ xml

Reference

G. Carneiro, P. Fortuna, and M. Ricardo. 2009. FlowMonitor: a network monitoring framework for the network simulator 3 (NS-3). In Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS '09). http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7493

Usage

การใช้งานโมดูลทำได้ง่ายมากเพราะมี Helper คอยช่วยอยู่

ตัวอย่างคือ

```
// Flow monitor
Ptr<FlowMonitor> flowMonitor;
FlowMonitorHelper flowHelper;
flowMonitor = flowHelper.InstallAll();

Simulator::Stop (Seconds(stop_time));
Simulator::Run ();

flowMonitor->SerializeToXmlFile("NameOfFile.xml", true, true);
```

รูปที่ จ.1 ตัวอย่างการใช้โมดูล

ฟังก์ชั่น SerializeToXmlFile () ในพารามิเตอร์ 2 และ 3 จะถูกใช้งานตามลำดับเพื่อเปิด/ปิดการใช้งานฮิตโตแกรมและรายละเอียดของแต่ละโพรบ

Attributes

ในโมดูลมีแอตทริบิวต์ดังต่อไปนี้

- MaxPerHopDelay (Time, default 10s)
- StartTime (Time, default 0s)
- DelayBinWidth (double, default 0.001)
- JitterBinWidth (double, default 0.001)
- PacketSizeBinWidth (double, default 20.0)
- FlowInterruptionsBinWidth (double, default 0.25)
- FlowInterruptionsMinTime (double, default 0.5)

Output

ผลลัพธ์ที่ได้จะอยู่ในรูปแบบ xml ตัวอย่างเช่น

```
<?xml version="1.0" ?>
<FlowMonitor>
  <FlowStats>
  <Flow flowId="1" timeFirstTxPacket="+0.0ns" timeFirstRxPacket="+20067198.0ns" timeLastTxPacket="+2235764408.0ns" timeLastRxPacket="+2255831606.0ns"
  </Flow>
  </FlowStats>
  <Ipv4FlowClassifier>
  <Flow flowId="1" sourceAddress="10.1.3.1" destinationAddress="10.1.2.2" protocol="6" sourcePort="49153" destinationPort="50000" />
  </Ipv4FlowClassifier>
  <Ipv6FlowClassifier>
  </Ipv6FlowClassifier>
  <FlowProbes>
  <FlowProbe index="0">
    <FlowStats  flowId="1" packets="3735" bytes="2149400" delayFromFirstProbeSum="+0.0ns" >
    </FlowStats>
  </FlowProbe>
  <FlowProbe index="2">
    <FlowStats  flowId="1" packets="7466" bytes="2224020" delayFromFirstProbeSum="+199415389258.0ns" >
    </FlowStats>
  </FlowProbe>
  <FlowProbe index="4">
    <FlowStats  flowId="1" packets="3735" bytes="2149400" delayFromFirstProbeSum="+138731526300.0ns" >
    </FlowStats>
  </FlowProbe>
  </FlowProbes>
</FlowMonitor>
```

รูปที่ จ.2 ผลลัพธ์ในรูปแบบ xml

ผลลัพธ์ที่ได้จาก TCP flow คือ 10.1.3.1 ถึง 10.1.2.2

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง