โครงการการกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายแอดฮอกบนยานพาหนะ

ผู้ช่วยศาสตราจารย์ ดร.กุลธิดา โรจน์วิบูลย์ชัย

มิถุนายน 2555

สัญญาเลขที่ MRG5380164

รายงานวิจัยฉบับสมบูรณ์

โครงการการกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายแอดฮอกบนยานพาหนะ

ผู้ช่วยศาสตราจารย์ ดร.กุลธิดา โรจน์วิบูลย์ชัย

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

สนับสนุนโดยสำนักคณะกรรมการการอุดมศึกษาและสำนักงานกองทุนสนับสนุนการวิจัย

(ความเห็นในรายงานนี้เป็นของผู้วิจัย สกอ. และ สกว. ไม่จำเป็นต้องเห็นด้วยเสมอไป)

การกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายแอดฮอกบนยานพาหนะ

ผู้ช่วยศาสตราจารย์ ดร.กุลธิดา โรจน์วิบูลย์ชัย

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

บทคัดย่อ

การกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะเป็นส่วนสำคัญในการส่งข้อมูลของบริการต่างๆในระบบการจราจรอัจฉริยะ ซึ่งต้องการการส่งข้อมูลอย่างมีประสิทธิภาพ และสามารถเชื่อถือได้ ผลงานวิจัยทางด้านการกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะที่ผ่านมาส่วนใหญ่ไม่คำนึงถึงความสำคัญของความเร็วในการกระจายข้อมูล โดยความเร็วในการกระจายข้อมูลนั้นมีผลต่อความแม่นยำของบริการระบบจราจรอัจฉริยะ ในงานวิจัยนี้จึงออกแบบโพรโทคอลที่มีความยืดหยุ่นและความเร็วในการทำงานสูง คือ ใช้เพียงแค่ข้อมูลความหนาแน่นในการเลือกโหนดที่จะทำหน้าที่ในการกระจายข้อมูลต่อ ทำให้การทำงานของโพรโทคอลไม่ต้องพึ่งพาอุปกรณ์ระบุตำแหน่งในการทำงาน และขั้นตอนสำคัญในการทำงานคือการเลือกโหนดที่จะกระจายข้อมูลต่อโดยหลีกเลี่ยงการใช้เวลารอเพื่อให้มีการกระจายข้อมูลได้รวดเร็วที่สุด ในงานวิจัยนี้ยังเสนอวิธีการคำนวณความถี่ในการส่งข้อมูลแบบบีคอนให้มีประสิทธิภาพ การคำนวณเวลารอที่ช่วยลดการสูญเสียเวลา และวิธีการแคชข้อมูลที่ใช้พื้นที่ในการเก็บข้อมูลน้อยแต่ไม่ส่งผลต่อความเชื่อถือได้ของโพรโทคอล การวัดสมรรถนะในการทำงานแบ่งออกเป็นการทดสอบในโปรแกรมจำลอง NS-2 และ NS-3 ในสถานการณ์จราจรจำลองบนถนนทางหลวง และถนนในเมือง และการทดสอบแบบโปรแกรมเลียนแบบในโปรแกรมจำลอง NS-3 ซึ่งพบปัญหาของโพรโทคอลเมื่อมีการทำงานในสภาพแวดล้อมที่มีการเชื่อมต่อแบบอสมมาตร งานวิจัยนี้จึงเสนอวิธีการเลือกโหนดในการกระจายข้อมูลโดยพิจารณาระดับความแรงของสัญญาณเชื่อมต่อไร้สาย ซึ่งสามารถเพิ่มสมรรถนะในการทำงาน 17% สำหรับค่าความเชื่อถือได้ และ 19% สำหรับค่าใช้จ่ายในการทำงาน ส่วนสุดท้ายในงานวิจัยเป็นการพัฒนาโพรโทคอลลงอุปกรณ์ที่มีระบบปฏิบัติการลินุกซ์และแอนดรอยเพื่อทดสอบว่าโพรโทคอลสามารถทำงานได้ตามการออกแบบบนอุปกรณ์จริง และสิ่งแวดล้อมจริง

คำหลัก : เครือข่ายบนยาพาหนะ, การกระจายอย่างเชื่อถือได้, เครือข่ายไร้สายแบบแอดฮอก

ผลงานตีพิมพ์

1. ณวุฒ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA : Density-aware reliable broadcasting in vehicular ad-hoc networks.* IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology Conference **2553**; 7: 598-602.
2. ชญานิน ไทยนะ, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *A study of adaptive beacon transmission on vehicular ad-hoc networks.* International Conference on Communication Technology **2554**; 13: 597-602.
3. วิภาวี วิริยพงษ์สุกิจ, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *A novel packet dropping policy for vehicular ad-hoc networks.* International Conference on Communication Technology **2554**; 13: 603-608.

4. ณัฐวิทย์ กมลธรรม, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *Improving reliable broadcast over asymmetric VANETs based on RSSI voting algorithm*. International symposium on Intelligent Signal Processing and Communication Systems **2554**;1-6.
5. ณฐกร นพคุณวิจัย, สุขุมาลย์ อาชาสันติสุข, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA on android : reliable broadcasting in ad-hoc network on android platform*. ICT International Senior Project Conference **2555**;.
6. กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA-bewa: density-aware reliable broadcasting on vehicular ad-hoc networks*. IEICE Transactions on Communications **2555**;. (Under Review)

Tel.:02-218-6993. 02-218-6954
E-mail: kultida.r@chula.ac.th

Reliable Broadcast on Vehicular Ad-Hoc Networks

Rojviboonchai, K.

Department of Computer Engineering, Faculty of Engineering,
Chulalongkorn University, Bangkok, Thailand

Abstract

Reliable broadcasting in vehicular ad-hoc networks (VANET) is one of the keys to success for services and applications on Intelligent Transportation System (ITS). The reason is that these applications need a way to exchange their information. There are many previous reliable broadcasting protocols on VANET but none of them have concerned the speed of data dissemination. High-speed protocols can provide more accuracy service to ITS. This research proposes density-aware reliable broadcasting in vehicular ad-hoc networks (DECA). The next rebroadcast selection is made by a source or a precursor node to avoid waiting timeout which increases delay to rebroadcasting. DECA uses only density information in its selection algorithm so it does not require any position discovery equipment. DECA also has the new beacon mechanism, new waiting timeout calculation and new caching policy. These can improve performance of DECA to operate on both highway and urban areas with various traffic densities. The performance of DECA has been evaluated on the well-known network simulators NS-2 and NS-3. From the emulation results using NS-3, an asymmetric link problem has been found. The problem causes lots of redundant broadcasting messages. This research also proposes RSSI-Voting algorithm (RVA) to tackle with the asymmetric link problem. RVA selects the next rebroadcasting node by voting the highest RSSI level node. This can improve DECA performance up to 17% and reduce overhead up to 28% on asymmetric link scenarios. Finally, DECA has been implemented on Linux and Android devices to ensure that the protocol can operate on real devices and real environment as designed.

Keywords: vehicular networks, reliable broadcast, ad-hoc

Outputs

1. Na Nakorn N., Rojviboonchai K.* *DECA : Density-aware reliable broadcasting in vehicular ad-hoc networks.* IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology Conference **2010**; 7: 598-602.
2. Thaina C., Na Nakorn K., Rojviboonchai K.* *A study of adaptive beacon transmission on vehicular ad-hoc networks.* International Conference on Communication Technology **2011**; 13: 597-602.
3. Viriyapongkit W., Na Nakorn K., Rojviboonchai K.* *A novel packet dropping policy for vehicular ad-hoc networks.* International Conference on Communication Technology **2011**; 13: 603-608.
4. Kamoltham N., Na Nakorn K., Rojviboonchai K.* *Improving reliable broadcast over asymmetric VANETs based on RSSI voting algorithm.* International symposium on Intelligent Signal Processing and Communication Systems **2011**;1-6.
5. Nophakunvijai N., Archasantisuk S., Na Nakorn K., Rojviboonchai K.* *DECA on android : reliable broadcasting in ad-hoc network on android platform.* ICT International Senior Project Conference **2012**;.

6. Na Nakorn K., Rojviboonchai K.* *DECA-bewa: density-aware reliable broadcasting on vehicular ad-hoc networks.* IEICE Transactions on Communications **2012**;. (Under Review)

*Corresponding author.
Tel.:02-218-6993, 02-218-6954
E-mail: kultida.r@chula.ac.th

Executive Summary

Nowadays vehicles are going to be smarter than they are used to be in the past. Their applications and services need an efficient, fast and reliable communication especially for safety and emergency applications. Therefore, reliable broadcasting in vehicular ad-hoc networks is challenging due to its unique characteristics including intermittent connectivity and various vehicular scenarios. This research proposed density-aware reliable broadcasting protocol (DECA). The new reliable broadcasting protocol is designed for such characteristics. To address the issue of various vehicular environment, the protocol consists of three main modules; (1) *Node selection algorithm* is the way to select the next rebroadcasting node. DECA select the node with highest density to rapidly increase number of received nodes for each transmission. The efficient node selection algorithm can decrease redundant retransmissions and also increase reliability. (2) *Waiting timeout algorithm* is a solution for avoiding broadcasting collision. But waiting timeout affects to unnecessary delay so the efficient algorithm can avoid most of collision with least delay. (3) *Beacon mechanism* is used for discovering neighbor and detecting the missing data in intermittent connectivity scenarios. Unfortunately beacon mechanism can decrease overall performance if protocols have too frequent beaconing. So DECA uses a new adaptive beacon interval to suit various densities. To evaluate performance of DECA, simulation, emulation and real testbed are used to cover various scenarios.

In the first phase of this research, the performance of DECA has been evaluated by network simulator NS-2 on both highway and urban grid scenarios. DECA can outperform other previous works in term of reliability, overhead and speed of data dissemination. Although DECA can operate well on simple scenarios, in real scenarios such a map of Bangkok with up to 3700 nodes, DECA suffers from its own performance degradation. This is due to extremely high redundant broadcasting messages. After observation, the author improves the performance of the protocol with a new beacon mechanism and a new waiting timeout algorithm. So DECA can even efficiently work on such an extremely high node density. A caching policy also has been studied in this phase to support the real implementation.

In the next phase, DECA has been implemented on NS-3 for testing in emulation. The results have shown an asymmetric link problem due to various transmission ranges of devices. Therefore, an improved node selection algorithm has been designed based on RSSI level, so-called RSSI-voting algorithm. The algorithm selects a node that most of neighbors can receive its data so the new selecting algorithm can improve the old one up to 18% of reliability and decrease up to 27% of overhead in asymmetric scenarios.

In the final phase of this research, DECA has been evaluated on heterogeneous device scenarios. The protocol has been implemented in Linux laptops and Android phones. During the tests, the laptops and android phones have been moving around the building to ensure that intermittent connectivity occurs. The results show that the protocol can perform at least 95% of reliability for every testing device. The performance shows possibilities of DECA to be used in real traffic environment.

สารบัญ

หน้า

บทคัดย่อ ..... ก

Abstract ..... ค

Executive Summary ..... จ

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญของปัญหา 1

1.2 งานวิจัยที่เกี่ยวข้อง 4

1.3 วัตถุประสงค์ของงานวิจัย 10

1.4 ขั้นตอนและวิธีดำเนินการวิจัย 10

บทที่ 2 การออกแบบโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ 11

2.1 แนวคิดในการออกแบบ 11

2.2 หลักการทำงานของโพรโทคอล DECA 11

2.2.1 การเก็บข้อมูลของโพรโทคอล 12

2.2.2 การแลกเปลี่ยนข้อมูลจากโหนดเพื่อนบ้าน (Beaconing) 12

2.2.3 การเลือกโหนดส่งต่อข้อความ (Preferred Node Selection Algorithm) 14

2.2.4 การคำนวณเวลารอ (Waiting Timeout Calculation) 16

2.3 ตัวอย่างการทำงานของโพรโทคอล 17

2.4 การทดลองและวิเคราะห์ผลการทดลอง 18

2.4.1 ตัววัดสมรรถนะของโพรโทคอล (Performance Metrics) 18

2.4.2 เครื่องมือในการวัดสมรรถนะของโพรโทคอล 19

2.4.3 สภาพแวดล้อมที่ใช้ในการทดลอง 19

2.4.4 ผลการทดลองค่าความเชื่อถือได้ของโพรโทคอล 21

2.4.5 ผลการทดลองค่าใช้จ่ายของโพรโทคอล 23

2.4.6 ผลการทดลองความเร็วในการกระจายข้อมูล 27

2.4.7 ผลการทดลองค่าความสำเร็จในการเลือกโหนดแพร่ข้อความต่อ 30

บทที่ 3 การปรับปรุงสมรรถนะโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ 31

หน้า

3.1 การศึกษาการส่งบีคอนแบบปรับค่าช่วงเวลาได้ในรูปแบบต่างๆ 31
3.1.1 แนวคิดในการศึกษาการส่งบีคอนแบบปรับค่าช่วงเวลาได้ 31
3.1.2 หลักการส่งบีคอนแบบปรับค่าช่วงเวลาได้แบบต่างๆ 32
3.1.3 ตัวอย่างการทำงาน 36
3.1.4 การทดลองและวิเคราะห์ผล 40
3.2 การจัดการข้อมูลในบัฟเฟอร์ 46
3.2.1 แนวคิดในการออกแบบนโยบายการทิ้งกลุ่มข้อมูล (Dropping Policy) 46
3.2.2 หลักการทำงานของนโยบายการทิ้งกลุ่มข้อมูล 46
3.2.3 ตัวอย่างการทำงาน 49
3.2.4 การทดลองและวิเคราะห์ผล 52
3.3 การปรับปรุงการทำงานของโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง 56
3.3.1 ปัญหาที่ขึ้นเมื่อทดลองโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง 56
3.3.2 การปรับปรุงกระบวนการทำงานของโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง 57
3.3.3 การทดลองและวิเคราะห์ผลการทดลอง 60
บทที่ 4 การศึกษาผลกระทบ และการปรับปรุงโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะบนเครือข่ายจริง 63
4.1 ความแตกต่างระหว่างโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 63
4.1.1 การพัฒนาโพรโทคอลบนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 68
4.1.2 การสร้างรูปแบบการเคลื่อนที่ของยานพาหนะ 70
4.1.3 การเปรียบเทียบการทำงานของโพรโทคอลบนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 71
4.2 การทดสอบผลกระทบในการทำงานของโพรโทคอลบนเครือข่ายจริง 73
4.3 หลักการทำงานของขั้นตอนการเลือกโหนดส่งต่อข้อความแบบโหวตค่า RSSI 75
4.3.1 แนวคิดในการปรับปรุงขั้นตอนการเลือกโหนดส่งต่อข้อความ 75
4.3.2 หลักการทำงานการเลือกโหนดส่งต่อแบบ RVA 76
4.3.3 ปัญหาที่เกิดขึ้นจากขั้นตอนการโหวตอาร์เอสเอสไอ และวิธีแก้ไข 77
4.4 การทดลองและวิเคราะห์ผลการทดลอง 80
4.4.1 ตัววัดสมรรถนะของโพรโทคอล (Performance Metrics) 80
4.4.2 เครื่องมือในการวัดสมรรถนะของโพรโทคอล 81
4.4.3 สภาพแวดล้อมที่ใช้ในการทดลอง 81
4.4.4 ผลการทดลองค่าความเชื่อถือได้ 84

หน้า

4.4.5 ผลการทดลองค่าใช้จ่าย 86

บทที่ 5 การพัฒนาโพรโทคอลบนอุปกรณ์ระบบปฏิบัติการแอนดรอยด์ 90

5.1 ระบบปฏิบัติการแอนดรอยด์ (Android) 90

5.2 การพัฒนาโพรโทคอล DECA และแอพพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ (Android) 92

5.2.1 *Package Deca_agent* 92

5.2.2 *Package table* 96

5.2.3 *Package timer* 99

5.2.4 *Package packet* 99

5.2.5 *Package util* 101

5.3 การพัฒนาแอพพลิเคชั่นแอนดรอยด์สำหรับการกระจายข้อความอย่างง่าย 101

5.4 ปัญหาที่น่าสนใจในการพัฒนาโพรโทคอล DECA บนอุปกรณ์ที่มีระบบปฏิบัติการแอนดรอยด์ 103

5.4.1 ปัญหา Rom มาตรฐานของโทรศัพท์มือถือ Google Nexus One ไม่สามารถเปิดใช้งาน Wi-Fi ใน mode Ad-Hoc ได้ 103

5.4.2 ปัญหาโทรศัพท์มือถือ Google Nexus One ไม่รับ Broadcast UDP Packet ขณะเครื่องพักหน้าจอ 104

5.4.3 ปัญหา Packet เดียวกันในแต่ละโหนดหมดอายุในเวลาที่ต่างกัน 104

5.5 การทดลองและวิเคราะห์ผลการทดลอง 105

5.5.1 Functional Test 105

5.5.2 การทดลองการทำงานร่วมกับอุปกรณ์อื่น 109

บทที่ 6 สรุปผลการวิจัย 111

6.1 สรุปผลการวิจัย 111

6.2 ข้อจำกัด 112

6.3 ข้อเสนอแนะ 112

ผลงานที่ได้รับจากโครงการ 113

รายการอ้างอิง 114

ภาคผนวก

สารบัญตาราง

หน้า

ตารางที่ 1-1 ตารางเปรียบเทียบหลักการทำงานของโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ 8
ตารางที่ 2-1 การตั้งค่าต่างๆที่ใช้ในการทดลอง 20
ตารางที่ 2-2 ตารางแสดงช่วงเวลาการทำบีคอน ที่เหมาะสมสำหรับ DECA 21
ตารางที่ 2-3 ตารางแสดงค่าความสำเร็จในการเลือกโหนดแพร่ข้อความต่อของ DECA (%) 30
ตารางที่ 3-1 ช่วงเวลาในการส่งบีคอนที่เหมาะสมกับความหนาแน่น 32
ตารางที่ 3-2 ตัวอย่างค่าที่ใช้ในการคำนวณสมการการถดถอยเชิงเส้น 36
ตารางที่ 3-3 ชุดข้อมูลตัวอย่างของวิธีการ K-Nearest Neighbor 38
ตารางที่ 3-4 การตั้งค่าต่างๆที่ใช้ในการทดลอง 42
ตารางที่ 3-5 ปริมาณของรถยนต์ในการทดลอง 53
ตารางที่ 3-6 การตั้งค่าต่างๆที่ใช้ในการทดลอง 53
ตารางที่ 3-7 การตั้งค่าต่างๆที่ใช้ในการทดลอง 61
ตารางที่ 4-1 ค่าความผิดพลาดในการเคลื่อนที่เฉลี่ยระหว่าง NS-2 และ NS-3 (%) 71
ตารางที่ 4-2 การตั้งค่าเพื่อวัดสมรรถนะของโพรโทคอล DECA 72
ตารางที่ 4-3 การตั้งค่าพารามิเตอร์เพื่อทดลองประสิทธิภาพ 82
ตารางที่ 5-1 การตั้งค่าในการทดลอง 110

สารบัญรูป

หน้า

รูปที่ 1-1 ลักษณะการสื่อสารบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ[2] 1
รูปที่ 1-2 ตัวอย่างบริการในระบบจราจรอัจฉริยะ [2] 2
รูปที่ 1-3 ลักษณะเฉพาะของรถยนต์บนถนน 3
รูปที่ 2-1 กราฟแสดงจำนวนการส่งข้อความที่ช่วงเวลาการทำบีคอนต่างๆกัน 13
รูปที่ 2-2 กราฟแสดงการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น 14
รูปที่ 2-3 ปัญหาการเลือกโหนดส่งต่อแบบวนซ้ำ 15
รูปที่ 2-4 ผังงานแสดงการทำงานของโพรโทคอล DECA 16
รูปที่ 2-5 กราฟแสดงการคำนวณเวลารอสูงสุด 17
รูปที่ 2-6 ลักษณะของรถในการเชื่อมต่อแบบปกติ 18
รูปที่ 2-7 ลักษณะของรถในการเชื่อมต่อเป็นช่วงๆ 18
รูปที่ 2-8 ลักษณะถนนที่ใช้ในการทดลอง 20
รูปที่ 2-9 กราฟแสดงค่าความเชื่อถือได้จากการทดลองบนถนนทางหลวง 22
รูปที่ 2-10 กราฟแสดงค่าใช้จ่ายทั้งหมด 25
รูปที่ 2-11 กราฟแสดงค่าใช้จ่ายจากการส่งบีคอน 26
รูปที่ 2-12 กราฟแสดงความเร็วในการกระจายข้อมูลบนถนนทางหลวงที่ความหนาแน่นแตกต่างกัน 28
รูปที่ 2-13 กราฟแสดงความเร็วในการกระจายข้อมูลบนถนนในเมืองที่ความหนาแน่นแตกต่างกัน 29
รูปที่ 3-1 กราฟแสดงการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น 32
รูปที่ 3-2 รูปแบบถนนที่ใช้ในการทดลองในโปรแกรมจำลอง 41
รูปที่ 3-3 ผลการทดลองจำนวนครั้งในการส่งบีคอนโดยใช้วิธี Linear regression และ K-Nearest Neighbor 43
รูปที่ 3-4 ผลการทดลองจำนวนครั้งในการส่งบีคอนโดยใช้วิธี LIA+NCR 43
รูปที่ 3-5 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนทางหลวง 44
รูปที่ 3-6 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนในเมือง โดยใช้วิธี Linear regression และ K-Nearest Neighbor 44
รูปที่ 3-7 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนทางหลวงโดยใช้วิธี LIA+NCR 45
รูปที่ 3-8 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนในเมืองโดยใช้วิธี LIA+NCR 45
รูปที่ 3-9 โหนดบริเวณของพื้นที่ทับซ้อนของการสื่อสาร 47
รูปที่ 3-10 จำนวนโหนดเพื่อนบ้านของโหนดต้นทาง 47
รูปที่ 3-11 แผนภาพแสดงการทำงานของนโยบายการทิ้งข้อมูล 49
รูปที่ 3-12 การคำนวณค่าสำเนาในกรณีโหนดต้นทางแพร่กลุ่มข้อมูลให้กับเพื่อนบ้านตนเอง 50

หน้า

รูปที่ 3-13 การคำนวณค่าสำเนาในกรณีโหนดที่ถูกเลือกแพร่กลุ่มข้อมูลให้กับเพื่อนบ้านของตนเอง 50
รูปที่ 3-14 การคำนวณค่าสำเนาในกรณีโหนดที่ถูกเลือกไม่ทำงานและเพื่อนบ้านทำการกระจายข้อมูลแทน 51
รูปที่ 3-15 การคำนวณค่าสำเนาในกรณีโหนดเพื่อนบ้านบางโหนดยังไม่ได้รับข้อมูล 51
รูปที่ 3-16 ลักษณะของถนนที่ใช้ในการทดลอง 52
รูปที่ 3-17 กราฟแสดงจำนวนของกลุ่มที่เหลืออยู่ในระบบบัฟเฟอร์ขนาด 10 กลุ่มข้อมูลที่เวลาต่างๆ 54
รูปที่ 3-18 กราฟแสดงผลการทดลองค่าความเชื่อถือได้จากบัฟเฟอร์ขนาดต่างๆ 55
รูปที่ 3-19 แผนที่บริเวณถนนสุขุมวิท กรุงเทพมหานครที่ใช้ในการทดลอง 56
รูปที่ 3-20 กราฟแสดงความสัมพันธ์ระหว่างเวลารอและความความหนาแน่นของโหนด 58
รูปที่ 3-21 กราฟแสดงผลการทดลองค่าใช้จ่ายจากการกระจายข้อความซ้ำด้วยการคำนวณเวลารอที่แตกต่างกัน 58
รูปที่ 3-22 โครงสร้างบีคอนของโพรโทคอล DECA 59
รูปที่ 3-23 ผลการทดลองการจำกัดจำนวนครั้งในการร้องขอข้อความที่ไม่ได้รับ 59
รูปที่ 3-24 ผลการทดลองค่าความเชื่อถือได้ 62
รูปที่ 3-25 ผลการทดลองค่าใช้จ่าย 62
รูปที่ 3-26 ผลการทดลองความเร็วในการกระจายข้อมูล 62
รูปที่ 4-1 ตัวอย่างเอกสารประกอบ API ของโปรแกรมจำลองเครือข่ายเวอร์ชั่น 3 [23] 64
รูปที่ 4-2 การจำลอง Virtual Machine เพื่อทดลองในระบบจริง [23] 65
รูปที่ 4-3 การทำงานบนสถานการณ์และอุปกรณ์จริง [23] 66
รูปที่ 4-4 ระบบสถิติของโปรแกรมจำลองเครือข่าย NS-3 [23] 66
รูปที่ 4-5 ตัวอย่างกราฟฟิกการตั้งค่าในโปรแกรมจำลองเครือข่าย NS-3 [23] 67
รูปที่ 4-6 รูปแบบของส่วนจำเพาะของโปรแกรมจำลองเครือข่าย NS-3 [24] 67
รูปที่ 4-7 สถาปัตยกรรมของโปรแกรมจำลองเครือข่าย NS-2 [21] 68
รูปที่ 4-8 สถาปัตยกรรมของโปรแกรมจำลองเครือข่าย NS-3 [23] 68
รูปที่ 4-9 การเพิ่มโพรโทคอลใหม่บนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 69
รูปที่ 4-10 โครงสร้างการทำงานพื้นฐานของโปรแกรมของทั้งสองโปรแกรม 70
รูปที่ 4-11 ส่วนของโปรแกรมที่ทำให้เกิดการหยุดของโหนดในการเคลื่อนที่ใน NS-3 70
รูปที่ 4-12 ระยะการเคลื่อนที่ของโหนดในโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 71
รูปที่ 4-13 ผลการทดลองค่าความเชื่อถือได้ 72
รูปที่ 4-14 ผลการทดลองค่าใช้จ่าย 73
รูปที่ 4-15 การทดสอบบนเครือข่ายจริงในบริเวณจุฬาลงกรณ์มหาวิทยาลัย 73
รูปที่ 4-16 ความผิดพลาดที่เกิดขึ้นในรูปแบบการเชื่อมต่อแบบอสมมาตร 74
รูปที่ 4-17 กระบวนการในการเลือกโหนด (ลูกศรแทนทิศทางการส่งบีคอนสำเร็จ) 75

หน้า

รูปที่ 4-18 ขั้นตอนที่หนึ่งของขั้นตอนการโหวตแบบอาร์เอสเอสไอ 76
รูปที่ 4-19 ขั้นตอนที่สองของขั้นตอนการโหวตแบบอาร์เอสเอสไอ 77
รูปที่ 4-20 ขั้นตอนที่สามของอัลกอริทึมการโหวตแบบอาร์เอสเอสไอ 77
รูปที่ 4-21 สรุปผลการเลือกโหนดของขั้นตอนการโหวตแบบอาร์เอสเอสไอ 77
รูปที่ 4-22 ปัญหาโหนดที่ได้รับการโหวตอันดับหนึ่งไม่ได้เป็นเพื่อนบ้านกับผู้ส่งข้อมูล 78
รูปที่ 4-23 แบบโครงสร้างที่เปลี่ยนแปลงเร็ว 79
รูปที่ 4-24 แบบโครงสร้างที่เปลี่ยนแปลงช้า 80
รูปที่ 4-25 ปัญหาการเลือกตัวเองเป็นผู้กระจายข้อมูลลำดับถัดไป 80
รูปที่ 4-26 ลักษณะถนนที่ใช้ในการทดลอง 82
รูปที่ 4-27 เงื่อนไขในการจำลองสถานการณ์ในการทดลอง 83
รูปที่ 4-28 กราฟแสดงค่าความเชื่อถือได้ 86
รูปที่ 4-29 กราฟแสดงค่าใช้จ่ายในการส่งข้อความของระบบที่ได้จากการทดลองบนถนนทางหลวง 88
รูปที่ 4-30 กราฟแสดงค่าใช้จ่ายในการส่งบีคอนของระบบที่ได้จากการทดลองบนถนนทางหลวง 88
รูปที่ 4-31 กราฟแสดงค่าใช้จ่ายรวมของระบบที่ได้จากการทดลองบนถนนทางหลวง 88
รูปที่ 4-32 กราฟแสดงค่าใช้จ่ายในการส่งข้อความของระบบที่ได้จากการทดลองบนถนนในเมือง 89
รูปที่ 4-33 กราฟแสดงค่าใช้จ่ายในการส่งบีคอนของระบบที่ได้จากการทดลองบนถนนในเมือง 89
รูปที่ 4-34 กราฟแสดงค่าใช้จ่ายรวมของระบบที่ได้จากการทดลองบนถนนในเมือง 89
รูปที่ 5-1 โครงสร้างสถาปัตยกรรมแอนดรอยด์ [26] 90
รูปที่ 5-2 โครงสร้างการพัฒนาโพรโทคอล DECA 92
รูปที่ 5-3 รายละเอียด *Package Deca_agent* 92
รูปที่ 5-4 รายละเอียดของ *Class Deca_agent* 94
รูปที่ 5-5 รายละเอียดของ *Class Deca_agent_thread* 95
รูปที่ 5-6 รายละเอียดของ *Class PacketTimer* 95
รูปที่ 5-7 รายละเอียดของ *Class NeighborTimer* 95
รูปที่ 5-8 รายละเอียดของ *Class BeaconTimer* 95
รูปที่ 5-9 รายละเอียดของ *Class RebroadcastTimer* 95
รูปที่ 5-10 รายละเอียดของ *Package table* 96
รูปที่ 5-11 รายละเอียดของ *Class Deca_rp_entry* 96
รูปที่ 5-12 รายละเอียดของ *Class Deca_rptable* 97
รูปที่ 5-13 รายละเอียดของ *Class Deca_nb_entry* 97
รูปที่ 5-14 รายละเอียดของ *Class Deca_nbtable* 97

หน้า

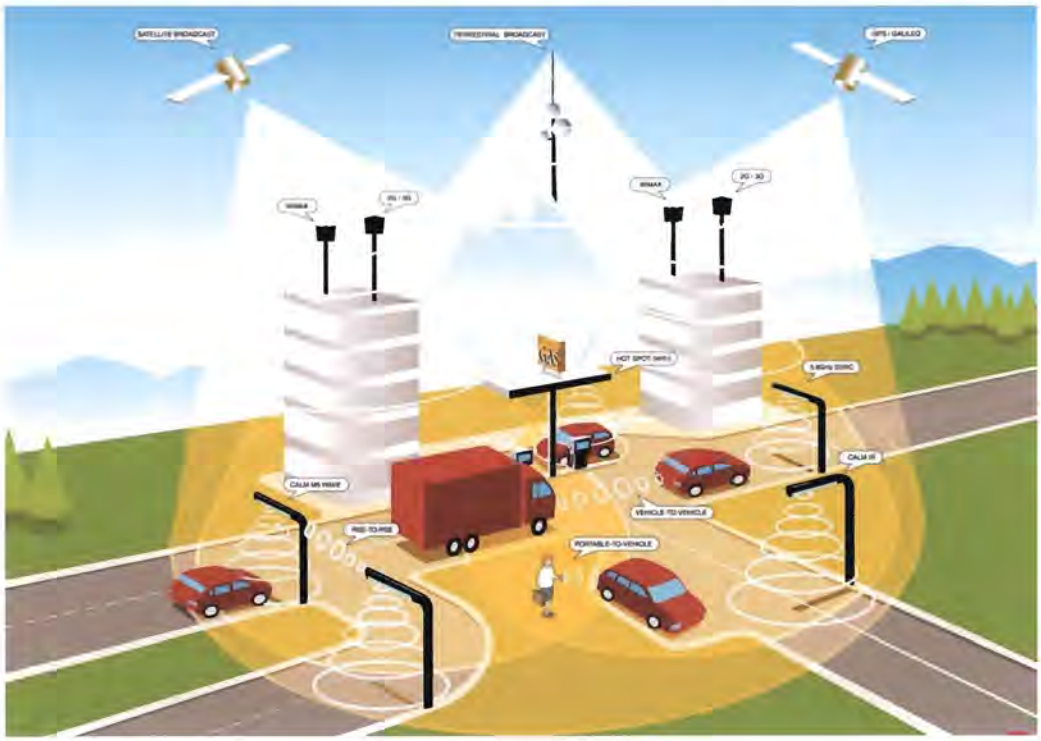
รูปที่ 5-15 รายละเอียดของ *Class Deca_temp_entry* 98
รูปที่ 5-16 รายละเอียดของ *Class Deca_ttable* 98
รูปที่ 5-17 รายละเอียด Class ต่างๆ ภายใน *Package timer* 99
รูปที่ 5-18 รายละเอียดของ *Package packet* 99
รูปที่ 5-19 รายละเอียดของ *Class Deca_packet* 100
รูปที่ 5-20 รายละเอียดของ *Class Deca_beacon_packet* 100
รูปที่ 5-21 รายละเอียดของ *Class Deca_broadcast_packet* 100
รูปที่ 5-22 รายละเอียดของ *Class Deca_util* 101
รูปที่ 5-23 การทำงานระหว่างแอพพลิเคชั่นแอนดรอยด์กับโพรโทคอล DECA 101
รูปที่ 5-24 รายละเอียด Class ต่างๆ ภายในแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย 101
รูปที่ 5-25 แบบร่าง User Interface สำหรับเปิด/ปิดเครือข่ายแอดฮอก 102
รูปที่ 5-26 แบบร่าง User Interface สำหรับรับ/ส่งข้อความ 102
รูปที่ 5-27 ปัญหาที่เกิดจากการใช้คำสั่ง insmod 103
รูปที่ 5-28 สถานะของโมดูล bcm4329 103
รูปที่ 5-29 การใช้คำสั่ง insmod และสถานะของโมดูลที่พร้อมใช้งาน 104
รูปที่ 5-30 การ insmod โมดูล bcm4329 โดยโดยการใส่พารามิเตอร์ dhd_pkt_filter_enable=0 104
รูปที่ 5-31ปัญหา Packet เดียวกันในแต่ละโหนดหมดอายุในเวลาที่ต่าง 105
รูปที่ 5-32 ผลการทดลอง *nbtable* หลังรับบีคอน 106
รูปที่ 5-33 ผลการทดลองลบเพื่อนบ้านที่หมดอายุออกจาก *nbtable* 106
รูปที่ 5-34 ผลการทดลองการกระจายข้อมูลซ้ำทันทีเมื่อโหนดถูกเลือกให้เป็นโหนดส่งต่อข้อความ 107
รูปที่ 5-35 การทดลองการลบข้อความออกจาก *ttable* เมื่อได้ยินผู้อื่นกระจายข้อความนั้นแล้ว 107
รูปที่ 5-36 ผลการทดลองการลบข้อความออกจาก *ttable* เมื่อแพร่ข้อความนั้นแล้ว 107
รูปที่ 5-37 ผลการทดลองการส่งบีคอนหาเพื่อนบ้านเพื่อร้องขอข้อความที่ตนเองไม่ได้รับ 108
รูปที่ 5-38 ผลการทดลองการได้รับข้อความที่ไม่ได้เลือกโหนดส่งต่อข้อความ 108
รูปที่ 5-39 ผลการทดลองการลบข้อความที่หมดอายุออกจาก *rptable* 108
รูปที่ 5-40 เส้นการการเคลื่อนที่ของโหนด และบริเวณที่ใช่ในการทดลอง 109
รูปที่ 5-41 ผลการทดลองค่าความเชื่อได้ของโหนดต้นทางแต่ละโหนด 110

บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของปัญหา

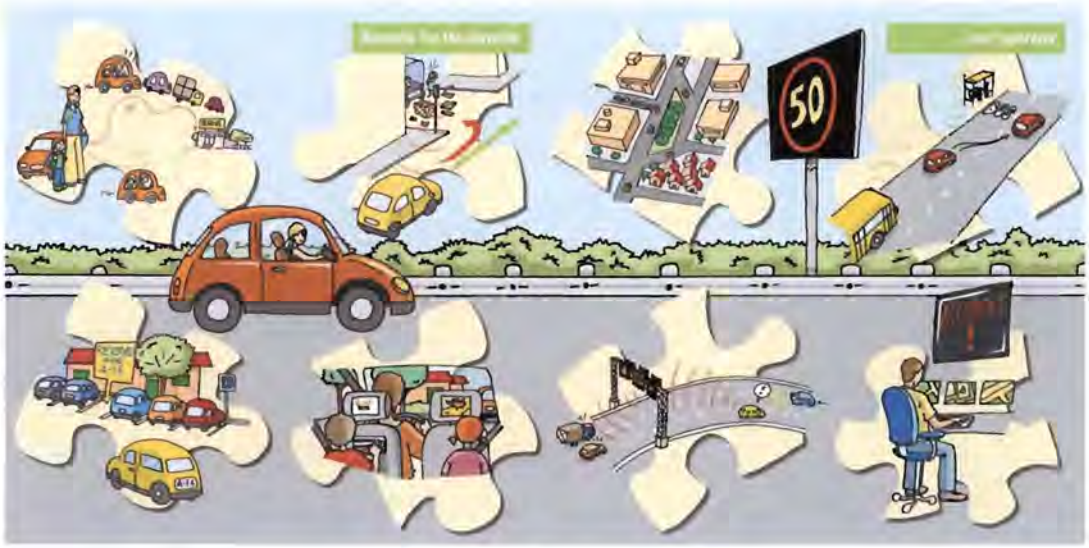
ในปัจจุบันการสื่อสารบนเครือข่ายไร้สายแบบแอดฮอก (MANET : Mobile Ad-hoc Networks) เป็นหัวข้อที่ได้รับความสนใจในการวิจัย เนื่องจากสามารถสื่อสารได้โดยไม่ต้องมีโครงสร้างพื้นฐาน ซึ่งสามารถเข้าไปทำงานในบริเวณที่ไม่สะดวกหรือไม่สามารถติดตั้งโครงสร้างพื้นฐานได้ ทำให้มีค่าใช้จ่ายในการติดตั้งระบบน้อยกว่ารูปแบบการสื่อสารที่พึ่งการทำงานของโครงสร้างพื้นฐาน อีกทั้งยังเป็นการทำงานแบบกระจาย (Distributed System) ซึ่งมีความทนทานต่อการถูกโจมตีหรือข้อผิดพลาดจากศูนย์กลาง การสื่อสารบนเครือข่ายไร้สายแบบแอดฮอกจึงถูกนำไปประยุกต์ใช้ในรูปแบบต่างๆกัน เช่น การสื่อสารบนเครือข่ายตัวรับรู้แบบไร้สาย (WSN : Wireless Sensor Network) หรือการสื่อสารบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ (VANET : Vehicular Ad-hoc Network) เป็นต้น โดยเฉพาะการสื่อสารบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะได้รับความสนใจเป็นอย่างมาก มีการเติบโตอย่างรวดเร็วทั้งด้านมาตรฐาน การพัฒนา และการวิจัย [1] มีโครงการที่เกี่ยวข้องกับการสื่อสารบนยานพาหนะจำนวนมากทั้งภายใน และภายนอกประเทศ เช่น CarTalk, CVIS [2] และ IntelliDrive เป็นต้น

เทคโนโลยีที่ถูกนำมาใช้ในการพัฒนารถยนต์มีมากขึ้น ซึ่งทำให้ระบบการทำงานบนรถยนต์มีความซับซ้อนมากขึ้น และมีความต้องการพื้นฐานในการทำงานมากขึ้น ระบบที่ไม่เคยถูกติดตั้งบนรถยนต์ ตัวอย่างเช่นระบบนำทางจีพีเอส (Global Positioning System) [3] ถูกนำมาใช้จนเกือบเป็นอุปกรณ์พื้นฐานของรถยนต์ในปัจจุบัน ซึ่งในการทำงานต้องการ



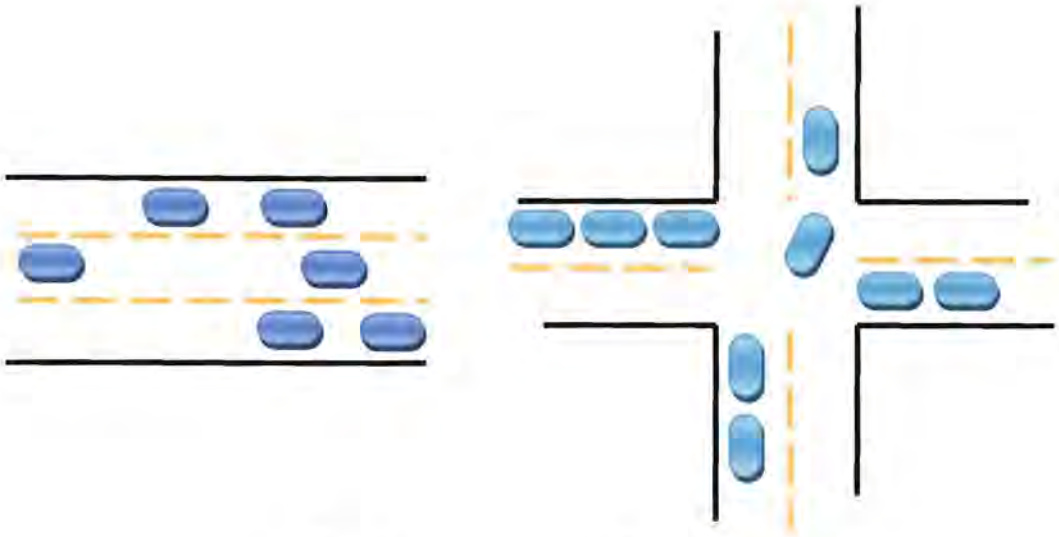
รูปที่ 1-1 ลักษณะการสื่อสารบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ[2]

อุปกรณ์รับสัญญาณดาวเทียม เช่นกันระบบการทำงานในอนาคตอย่างระบบจราจรอัจฉริยะ (Intelligent Transportation System)[4] ย่อมเข้ามาเป็นส่วนหนึ่งของรถยนต์ในอนาคตอันใกล้ แต่การทำงานของระบบจราจรอัจฉริยะนั้น นอกจากอุปกรณ์ที่จะถูกติดตั้งในรถยนต์แล้ว ยังจำเป็นที่จะต้องมีความสามารถพื้นฐานในการแลกเปลี่ยนข้อมูลอย่างเชื่อถือได้ระหว่างรถสู่รถ (Car-to-Car) หรือรถสู่โครงสร้างพื้นฐานบนถนน (Car-to-Infrastructure) เพื่อให้สามารถนำข้อมูลมาประมวลผลและให้บริการได้อย่างแม่นยำ บริการบางชนิด เช่น ระบบหลีกเลี่ยงอุบัติเหตุ หรือบริการแจ้งเตือนเหตุฉุกเฉิน บริการเหล่านี้นอกจากจะต้องการการกระจายข้อมูลอย่างเชื่อถือได้ (Reliable Broadcasting) เป็นพื้นฐานในการทำงานแล้ว โพรโทคอลที่สามารถทำงานภายในเวลาที่จำกัดเป็นเรื่องจำเป็น เพื่อให้ผู้ใช้บริการได้รับความปลอดภัยที่สูงขึ้นจากความแม่นยำของบริการ



รูปที่ 1-2 ตัวอย่างบริการในระบบจราจรอัจฉริยะ [2]

การทำงานของระบบจราจรอัจฉริยะผ่านโครงสร้างพื้นฐานที่ยังไม่พร้อม และการทำงานที่มีความล่าช้า ซึ่งไม่ทันท่วงทีขณะเกิดเหตุฉุกเฉิน จำเป็นต้องพึ่งพาเทคโนโลยีการสื่อสารไร้สายแบบแอดฮอกในการสื่อสารข้อมูล ขณะเดียวกันหากนำการกระจายข้อมูลแบบแอดฮอกแบบดั้งเดิม เช่น Simple Flooding ที่สามารถพัฒนาได้ง่ายนั้น แต่กลับไม่สามารถทำงานได้ในสภาพแวดล้อมที่มีความหนาแน่นของโหนดต่ำ ซึ่งทำให้การส่งต่อข้อมูลไม่มีความเชื่อถือได้ เนื่องจากเมื่อโหนดได้รับข้อความจะทำการส่งต่อข้อความนั้นทันทีโดยไม่ใช้ข้อมูลในการตัดสินใจเพิ่มเติม หรือในบริเวณที่มีความหนาแน่นสูงก็ทำให้เกิดปัญหาการชนของข้อมูล (Broadcast Storming Problem)[5] ตัวอย่างเช่น กระบวนการค้นหาเส้นทางของโพรโทคอล AODV (Ad-hoc On-demand Distance Vector) ที่ใช้ Simple Flooding ในการค้นหาเส้นทาง เมื่อนำมาทดสอบในสภาพแวดล้อมที่เป็นการจราจรของรถยนต์แล้วทำให้ประสิทธิภาพต่ำลง [6] หรือปัญหาที่เกิดจากการขาดการเชื่อมต่อเป็นเวลานาน (Disconnected Network) ที่มักเกิดขึ้นบนถนนที่มีรถน้อย ก็ต้องการโพรโทคอลที่มีความสามารถรองรับการทำงานในสภาพเช่นนี้ได้ [7]



รูปที่ 1-3 ลักษณะเฉพาะของรถยนต์บนถนน

เมื่อพิจารณาวิธีการกระจายข้อมูลอย่างเชื่อถือได้บนเครือข่ายแบบ MANET [8] [9] ซึ่งมีการนำเทคนิคต่างๆเข้ามาช่วยในการทำงาน ส่งผลให้โพรโทคอลมีประสิทธิภาพในการทำงานสูงขึ้น แต่เนื่องจากโพรโทคอลเหล่านี้ถูกออกแบบมาเพื่อการทำงานบนเครือข่ายไร้สายแอดฮอกแบบทั่วไป ซึ่งโหนดถูกทดสอบโดยมีการเคลื่อนที่แบบสุ่ม (Random Waypoint Model) ดังนั้นจึงไม่เหมาะสมที่จะนำมาใช้กับรถยนต์ที่มีความเร็วในการเคลื่อนที่สูง มีการเปลี่ยนแปลงสภาพแวดล้อมอย่างรวดเร็ว และมีพฤติกรรมที่ซับซ้อนกว่า เช่น การแซงของรถยนต์บนทางหลวง หรือในบริเวณหนึ่งอาจจะมีโหนดหนาแน่นมากกว่าอีกบริเวณหนึ่งมาก เช่น การจับกลุ่มของรถบริเวณทางหลวง หรือการหยุดรอสัญญาณไฟบริเวณสี่แยก ดังรูปที่ 1-3 จึงมีงานวิจัยที่ออกแบบมาทำงานโดยเฉพาะสำหรับการกระจายข้อมูลแบบเชื่อถือได้สำหรับเครือข่ายแอดฮอกบนรถยนต์ ซึ่งมีการแก้ไขปัญหาการเชื่อมต่อแบบเป็นช่วงๆ (Intermittent Connectivity) โดยใช้เทคนิค Store-and-Forward และหลีกเลี่ยงการเกิดปัญหาการชนของข้อมูล (Broadcast Storming Problem) งานวิจัยที่เกี่ยวข้อง ได้แก่ Edge-Aware Epidemic Protocol (EAEP) [10] และ AckPBSM (Acknowledge Parameterless Broadcast Protocol in Static to Highly Mobile Ad-hoc Networks) [11]

การทำงานของ EAEP ใช้ข้อมูลของจีพีเอสมาช่วยในการทำงาน โดยหลักการทำงานของโพรโทคอลจะให้โหนดมีเวลารอช่วงระยะหนึ่งหลังจากได้รับข้อความจากโหนดอื่นจากนั้นจึงนับจำนวนครั้งที่โหนดข้างเคียงส่งข้อความมาคิดเป็นค่าของความน่าจะเป็นที่โหนดนั้นๆจะส่งต่อข้อความนั้น ซึ่งโหนดที่อยู่บริเวณขอบของการส่งข้อมูลจะมีค่าความน่าจะเป็นสูงกว่าโหนดบริเวณอื่นๆ จากการทดลองใน [10] โพรโทคอลถูกทดสอบบนถนนทางหลวงเท่านั้น และใช้เวลาในการกระจายข้อมูลถึง 30 วินาทีเพื่อส่งให้รถยนต์ส่วนใหญ่ในพื้นที่

AckPBSM ใช้กระบวนการที่ซับซ้อนกว่า ซึ่ง AckPBSM ถูกพัฒนาขึ้นจาก PBSM [12] เป็นโพรโทคอลสำหรับการกระจายข้อมูลบน MANET การทำงานของ AckPBSM จะใช้ข้อมูลจีพีเอส เพื่อทราบตำแหน่งของโหนดเพื่อนบ้าน โดยอาศัยการทำงานของบีคอน (Beacon Message) ซึ่ง AckPBSM จะนำข้อมูลเหล่านี้มาใช้ในการคำนวณเพื่อสร้าง Connected

Dominating Sets (CDS) เมื่อมีการกระจายข้อความเกิดขึ้น โหนดที่ได้รับข้อความจะตั้งเวลาช่วงหนึ่ง โดยที่โหนดที่เป็น CDS จะมีเวลาในการคอยสั้นกว่าก่อนการส่งต่อข้อความอีกครั้ง ในกรณีเกิดปัญหาการเชื่อมต่อเป็นช่วงๆ AckPBSM จะแนบการตอบรับข้อความ (Acknowledgement) เข้าไปกับบีคอน (Beacon Message) เพื่อให้โหนดส่งข้อความให้กับโหนดเพื่อนบ้านกรณีที่มีข้อความไม่ครบ จากการทดลองใน [11] AckPBSM สามารถทำงานในสภาพแวดล้อมที่เป็นการจราจรทั้งบนทางหลวงและถนนในเมืองได้ดีกว่า PBSM และ DV-CAST [13] ที่ถูกนำมาเปรียบเทียบ

เมื่อพิจารณางานวิจัยที่กล่าวมาพบว่างานเหล่านี้สนใจในเรื่องของความเชื่อถือได้ (Reliability) และค่าใช้จ่ายในการทำงานของโพรโทคอล (Overhead) แต่ไม่มีการวัดประสิทธิภาพในเรื่องความเร็วของการกระจายข้อมูล (Speed of Data Dissemination) ซึ่งเป็นสิ่งที่จำเป็นสำหรับการทำงานของระบบจราจรอัจฉริยะ หรือการทำงานพื้นฐานอื่นๆที่ต้องการความแม่นยำในการทำงาน ซึ่งนับว่าเป็นหนึ่งในปัจจัยที่จะทำให้บริการบนระบบจราจรอัจฉริยะประสบความสำเร็จ ดังนั้นโพรโทคอลที่สามารถส่งข้อมูลโดยมีความเชื่อถือได้สูงที่สุด ภายในเวลาที่จำกัด และมีประสิทธิภาพในการทำงานสูงจึงเป็นเรื่องที่น่าสนใจ เนื่องจากการได้รับข้อมูลที่เร็วกว่าข้อมูลย่อมมีประโยชน์มากกว่า อีกทั้งหากมีค่าใช้จ่าย (Overhead)ในการทำงานต่ำทำให้ช่องสัญญาณในการสื่อสารมีที่ว่างมากพอสำหรับบริการอื่นๆที่เพิ่มขึ้น นอกจากนี้โพรโทคอลหากสามารถทำงานได้ในสภาพที่ไม่มีข้อมูลจีพีเอสโดยที่ประสิทธิภาพในการทำงานลดลงบ้าง แต่ยังมีความเชื่อถือได้คงเดิมจะสามารถเพิ่มความยืดหยุ่นในการทำงานบนความเป็นจริงที่ข้อมูลจีพีเอสไม่สามารถใช้งานได้ตลอดเวลาในการทำงานของโพรโทคอล

1.2 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่พัฒนาการกระจายข้อมูลแบบเชื่อถือได้บนเครือข่ายไร้สายแบบแอดฮอกสำหรับรถยนต์มักจะถูกพัฒนาอยู่บนพื้นฐานของการทำงานแบบ Store-and-Forward คือ โหนดที่ได้รับข้อความจะเก็บข้อความไว้จนกระทั่งข้อความนั้นหมดอายุ โดยโหนดอาจจะส่งต่อข้อความทันที หรือเก็บข้อความนั้นไว้ เพื่อแพร่ข้อความให้แก่โหนดเพื่อนบ้านที่ยังไม่ได้รับข้อความในเวลาต่อมา มีงานวิจัยที่ออกแบบวิธีการกระจายข้อมูลบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะที่น่าสนใจดังต่อไปนี้

PGB : Prefer Group Broadcast [6] เป็นขั้นตอนวิธีในการกระจายข้อมูลสำหรับยานพาหนะเพื่อลดปัญหาการชนกันของสัญญาณในบริเวณที่มีความหนาแน่นสูง เช่น บริเวณสี่แยกที่มีไฟสัญญาณจราจร เป็นต้น PGB ถูกออกแบบมาเพื่อใช้ในการแก้ปัญหาของการกระจายข้อความ Route Request (RREQ) ของ AODV ที่เกิดการชนสูง เนื่องจากใช้การกระจายแบบ Simple Flooding ที่ทุกโหนดจะส่งข้อความต่อทันที โดยไม่ใช้ข้อมูลใดๆในการตัดสินใจสำหรับการส่งต่อข้อความ และส่งผลให้ AODV มีประสิทธิภาพลดลงมาก การทำงานของ PGB โหนดจะใช้ระดับสัญญาณของโหนดเพื่อนบ้านในการคำนวณค่าเวลารอ โดยโหนดที่อยู่ห่างจากโหนดต้นทาง (Source) มากกว่าจะมีเวลารอสั้นกว่า ดังนั้นโหนดที่จะแพร่ต่อมาจึงมักจะเป็นโหนดที่บริเวณขอบของวงการกระจาย PGB จึงสามารถลดจำนวนข้อความ RREQ ได้เป็นจำนวนมาก แต่การทำงานของ PGB ยังไม่มีประสิทธิภาพในบริเวณที่มีความหนาแน่นน้อย และ PGB ถูกออกแบบมาเพื่อใช้ในการกระจาย RREQ ซึ่งไม่สนับสนุนการทำงานที่ต้องการการกระจายข้อมูลความเชื่อถือได้

EAEP : Edge-Aware Epidemic Protocol [10] เป็นโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ มีลักษณะการทำงานแบบ Store-and-Forward หลักการทำงาน คือ เมื่อโหนดได้รับข้อความใหม่ โหนดจะคำนวณเวลารอ โดยใช้ระยะทางจากตัวโหนดเองถึงผู้ส่ง ซึ่งต้องอาศัยข้อมูลตำแหน่งของโหนดจากจีพีเอส โหนดที่มีเวลารอสั้นที่สุดคือโหนดบริเวณขอบของความกว้างสัญญาณของตัวส่ง ในขณะที่อยู่ในช่วงเวลาที่คอยโหนดจะทำการนับจำนวนครั้งที่มีการส่งต่อข้อความจากเพื่อนบ้าน เมื่อหมดเวลารอ โหนดจะนับจำนวนครั้งที่เพื่อนบ้านส่งข้อความมาคำนวณหาค่าความน่าจะเป็นที่จะส่งข้อความนั้นต่อ จากหลักการทำงาน EAEP ไม่สามารถทำงานได้บนสภาพแวดล้อมที่มีปัญหาการเชื่อมต่อเป็นช่วงๆ (Intermittent Connectivity) ได้ดี กรณีที่มีโหนดเพื่อนบ้านใหม่เข้ามาบริเวณที่มีการส่งข้อความเสร็จสิ้นแล้ว จะทำให้โหนดเพื่อนบ้านใหม่นั้นไม่ได้รับการส่งข้อความใหม่

การใช้ความน่าจะเป็นมาคำนวณว่าควรมีการส่งต่อข้อความนั้นหรือไม่ย่อมจะทำให้เกิดการส่งข้อความซ้ำมากกว่าหนึ่งครั้งในบริเวณเดียวกันได้ จึงส่งผลให้ประสิทธิภาพในการทำงานลดลง นอกจากนี้จาก [10] พบว่า EAEP สามารถกำหนดทิศทางในการกระจายข้อมูล อีกทั้งยังรองรับการกระจายข้อมูลอิสระจากความสามารถในการส่งสัญญาณของอุปกรณ์ แต่เป็นการทดลองเพียงในสภาพแวดล้อมที่เป็นถนนทางหลวงเท่านั้น จึงอาจจะสรุปได้ว่า EAEP ถูกออกแบบเพื่อทำงานบนถนนทางหลวง หรืออาจจะทำงานได้ไม่ดีบนถนนในเมือง เมื่อพิจารณาถึงผลการทดลอง [10] EAEP สามารถทำงานได้ดีกว่า Simple Flooding มากทั้งทางด้านความเชื่อถือได้ และประสิทธิภาพ แต่ใช้เวลาในการกระจายข้อมูลให้กับรถยนต์ส่วนใหญ่บนถนนทางหลวงถึง 30 วินาที ซึ่งไม่สามารถนำไปใช้งานกับบริการที่ต้องการความเร็ว และแม่นยำของข้อมูลสูง

DV-Cast : Distributed Vehicular Broadcast Protocol for Vehicular Ad-Hoc Networks [13] เป็นวิธีการกระจายที่ออกแบบเพื่อให้แพร่ข้อมูลให้กับรถในบริเวณหนึ่งๆให้ได้รับมากที่สุด ตัวอย่างเช่นเมื่อเกิดอุบัติเหตุขึ้น โหนดต้องการส่งข้อความเตือนไปยังโหนดทั้งหมดที่ตามหลังเป็นระยะทาง 2 กิโลเมตร การทำงานจะอาศัยข้อมูลจีพีเอส และบีคอน เพื่อให้โหนดทราบถึงข้อมูลของตำแหน่งและทิศทางของโหนดเพื่อนบ้าน การทำงานจะแบ่งออกเป็นสามกรณี คือ

1) กรณีที่มีโหนดตามหลังที่อยู่ในระยะเชื่อมต่อ จะใช้การกระจายโดยตั้งเวลารอ โหนดที่อยู่บริเวณขอบของการกระจายจะมีเวลารอที่สั้นกว่า

2) กรณีที่ไม่พบโหนดตามหลังแต่มีโหนดที่วิ่งในทิศทางตรงข้าม โหนดจะแพร่ข้อความให้กับโหนดในทิศตรงข้ามทันที เพื่อให้โหนดนั้นเก็บข้อความและส่งต่อให้โหนดที่ตามหลังโหนดต้นทาง (Source) ที่มีระยะห่างมากกว่าระยะสัญญาณ

3) กรณีที่ไม่พบโหนดใด โหนดจะเก็บข้อความนั้นไว้จนกว่าจะพบโหนดเพื่อนบ้านใหม่ การส่งข้อมูลจะขึ้นอยู่กับความน่าจะเป็นที่โหนดนั้นคำนวณได้จากระยะห่างระหว่างโหนดกับโหนดแพร่ก่อนหน้า (Source/Precursor Node) และค่าความน่าจะเป็นจะลดลงเมื่อได้ยินการกระจายจากโหนดเพื่อนบ้าน

จากหลักการทำงานจะพบว่า DV-Cast มีการทำงานแบบ store-and-forward และมีการทำงานที่คล้ายคลึงกับการทำงานของ EAEP ซึ่งใช้ความน่าจะเป็นในการตัดสินใจการกระจายข้อมูลของโหนด แต่มีการใช้บีคอนเพื่อทราบตำแหน่ง

และทิศทางของโหนดเพื่อนบ้าน ซึ่งทำให้การกระจายมีโอกาสประสบความสำเร็จมากขึ้น แต่เนื่องจากการออกแบบที่เน้นให้มีการส่งข้อความแบบทิศเดียวมากกว่าทั้งสองทิศพร้อมกัน และโหนดไม่ทราบข้อมูลของข้อความที่เพื่อนบ้านมีจึงทำให้เกิดปัญหาในกรณีที่มีการขาดการเชื่อมต่อเป็นเวลานานได้เช่นเดียวกับ EAEP

AckPBSM : Acknowledge Parameterless Broadcast in Static to Highly Mobile Ad-Hoc Networks [11] เป็นอีกหนึ่งโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะที่ทำงานโดยใช้หลักการ Store-and-Forward ลักษณะการทำงานของ AckPBSM ในทุกๆช่วงเวลาหนึ่งโหนดจะมีการส่งบีคอนให้กับโหนดเพื่อนบ้าน ซึ่งภายในจะประกอบด้วยตำแหน่งของโหนด และการตอบรับ (Acknowledgement) ของข้อความที่โหนดเพื่อนบ้านนั้นได้รับทั้งหมด ตำแหน่งของโหนดเพื่อนบ้านจะถูกนำมาใช้เพื่อสร้าง Connected Dominating Sets (CDS) ซึ่งสามารถคำนวณโดยใช้ระยะห่างระหว่างแต่ละโหนด และความสามารถในการรับส่งสัญญาณ [12] ทำให้โหนดทราบได้ว่าตนเองอยู่ภายในกลุ่มของ CDS หรือไม่ ซึ่ง CDS จะเป็นกลุ่มของโหนดที่ต้องการให้มีการส่งต่อข้อมูลมากกว่า เมื่อมีการกระจายข้อมูล ทุกโหนดจะมีการตั้งเวลารอโดยที่คำนวณจากจำนวนของเพื่อนบ้านที่อยู่รอบโหนดนั้น ซึ่งโหนดที่อยู่ใน CDS จะมีสมการในการคำนวณเวลาให้ได้ค่าที่น้อยกว่าเสมอ ในกรณีที่เมื่อหมดเวลารอ โหนดจะตรวจจะตรวจสอบว่ายังมีเพื่อนบ้านที่ไม่ได้รับข้อความหรือไม่ ในกรณีที่โหนดได้รับการตอบรับ (ACK) จากโหนดเพื่อนบ้านทุกโหนด โหนดนั้นจะไม่ทำการส่งข้อมูลต่อ ซึ่งจะสามารถลดจำนวนการส่งข้อความในแต่ละครั้งที่มีการกระจายได้ แต่ในกรณีที่มีเพื่อนบ้านที่ยังไม่ได้รับข้อความก็จะมีการส่งต่อให้ตามปกติ จากงานวิจัย AckPBSM สามารถทำงานในสภาพแวดล้อมที่เป็นการจราจรทั้งบนทางหลวงและถนนในเมืองได้ดีกว่า PBSM และ DV-CAST [13] ที่ถูกนำมาเปรียบเทียบ

แต่เนื่องจากการทำงานของ AckPBSM ใช้การคำนวณเวลาจากจำนวนโหนดเพื่อนบ้านเพื่อส่งข้อมูลต่อ ดังนั้นกรณีโหนดที่โหนดใน CDS เหมือนกัน หรือแม้แต่ไม่เป็นโหนดใน CDS เหมือนกันแต่อยู่ในบริเวณเดียวกัน ย่อมมีโอกาสที่จะมีจำนวนโหนดเพื่อนบ้านเท่ากัน และทำให้เวลาที่คำนวณได้มีค่าเท่ากัน ส่งผลให้มีการกระจายข้อความนั้นพร้อมๆกัน เกิดการกระจายซ้ำซ้อนที่บริเวณเดียวกันจำนวนมาก ส่งผลให้ประสิทธิภาพในการทำงานของ AckPBSM จึงลดลงมาก

เมื่อพิจารณาถึงงานวิจัยที่กล่าวมา สามารถสรุปการหลักทำงานของโพรโทคอลได้ตามตารางที่ 1-1 ซึ่งโพรโทคอลทั้งหมดการกระจายข้อมูลแต่ละครั้งจำเป็นจะต้องมีการตั้งเวลารอเพื่อเก็บข้อมูลหรือคอยฟังโหนดเพื่อนบ้าน ดังนั้นการกระจายแต่ละครั้งจึงใช้ระยะเวลามากขึ้น เวลาที่โหนดในพื้นที่ทั้งหมดจะได้รับข้อความจะขึ้นอยู่กับเวลารอที่ใช้ในการกระจายแต่ละครั้งและจำนวน hop ดังนั้นการส่งผ่านข้อมูลโดยที่มีการกำหนดผู้ส่งไว้ล่วงหน้า สามารถทำให้การส่งต่อข้อความทำได้ทันที และลดจำนวนการกระจายซ้ำในบริเวณการส่งเดียวกันได้

นอกเหนือจากการจัดการที่ระดับ Network Layer แล้วยังมีงานวิจัยที่พัฒนาการกระจายข้อมูลที่มีขนาดเล็ก เช่น การกระจายบีคอนให้กับเพื่อนบ้าน ในระดับ MAC Layer [19] งานวิจัยนี้สนใจเวลาที่ถูกใช้ไป และสนใจค่าใช้จ่าย (Overhead) ที่เกิดขึ้นขณะที่รถยนต์มีการกระจายบีคอนจำนวนมากเข้าสู่เครือข่าย ซึ่งเมื่อเปรียบเทียบขนาดของ Packet ในการทำ CTS และ RTS ตามมาตรฐาน IEEE 802.11 [15] และบีคอนจะได้ความแตกต่างขนาดของข้อมูลไม่มากนัก งานวิจัยนี้จึงเสนอวิธีการนำข้อมูลในบีคอนแนบเข้าไปกับ Packet ของ CTS และ RTS เพื่อลดปริมาณการส่งข้อความที่เกิดขึ้นใน

ระบบ จึงเป็นวิธีการการส่งข้อมูลของบีคอนที่น่าสนใจ และนำมาใช้ร่วมกันการทำงานของโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ ที่มักจะมีการใช้งานบีคอนอยู่แล้ว

ตารางที่ 1-1 ตารางเปรียบเทียบหลักการทำงานของโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ

	PGB [6]	EAEP[10]	DV-CAST[13]	AckPBSM[11]	DECA
ตัวแปรในการพิจารณาโหนดส่งต่อ (forwarder node)	บริเวณของขอบการกระจายข้อมูล	บริเวณของขอบการกระจายข้อมูล	บริเวณของขอบการกระจายข้อมูล	บริเวณที่เป็นส่วนเชื่อมต่อของกลุ่ม (Connected Dominating Set :CDS)	บริเวณที่มีความหนาแน่นสูง
วิธีการเลือกโหนดส่งต่อ	ใช้การตั้งเวลารอ	ใช้การตั้งเวลารอ และคำนวณค่าความน่าจะเป็น	ใช้การตั้งเวลารอ และคำนวณค่าความน่าจะเป็น	ใช้การตั้งเวลารอ	โหนดต้นทาง หรือโหนดก่อนหน้าเป็นผู้เลือก
เวลาหน่วงที่เกิดขึ้นในการกระจายข้อมูลแต่ละครั้ง	เวลารอ และเวลาการส่ง	เวลารอ และเวลาการส่ง	เวลารอ และเวลาการส่ง	เวลารอ และเวลาการส่ง	เวลาการส่ง (propagation delay)
ความซับซ้อนในการคำนวณเพื่อเลือกโหนดส่งต่อ	O(1)	O(1)	O(1)	$O(n^3)$	O(n)
ค่าที่ใช้ในการคำนวณเวลารอ	ระดับของสัญญาณจาก RSSI	ระยะทางระหว่างโหนดและโหนดก่อนหน้า	ระยะทางระหว่างโหนดและโหนดก่อนหน้า	จำนวนของโหนดเพื่อนบ้าน	จำนวนของโหนดเพื่อนบ้าน
การตั้งเวลารอจะถูกใช้เมื่อ	ทุกๆครั้งเมื่อมีการส่งต่อข้อมูล	ทุกๆครั้งเมื่อมีการส่งต่อข้อมูล	ทุกๆครั้งเมื่อมีการส่งต่อข้อมูล	ทุกๆครั้งเมื่อมีการส่งต่อข้อมูล	เฉพาะเมื่อมีการเชื่อมต่อเป็นช่วงๆ

ตารางที่ 1-1 ตารางเปรียบเทียบหลักการทำงานของโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ(ต่อ)

	PGB [6]	EAEP[10]	DV-CAST[13]	AckPBSM[11]	DECA
ช่วงเวลาในการกระจายบีคอน (Beacon)	ปรับช่วงเวลาตามความหนาแน่นของโหนด	-	ทุก 1 วินาที	ทุก 0.5 วินาที	ปรับช่วงเวลาตามความหนาแน่นของเครือข่าย (ทุก 1.5-7 วินาที)
ข้อมูลภายในบีคอน (Beacon Message)	ข้อมูลจีพีเอส	-	ข้อมูลจีพีเอส	ข้อมูลจีพีเอส และข้อความตอบรับ (Acknowledge Message)	จำนวนเพื่อนบ้าน และข้อความตอบรับ (Acknowledge Message)
การรองรับเครือข่ายที่มีปัญหาขาดการเชื่อมต่อเป็นเวลานาน	ไม่ใช่	ไม่ใช่	ไม่ใช่	ใช่	ใช่

1.3 วัตถุประสงค์ของงานวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อนำเสนอโพรโทคอลสำหรับการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ (Reliable Broadcasting on Vehicular Ad-Hoc Networks) เพื่อสนับสนุนการทำงานของบริการที่มีอยู่บนระบบจราจรอัจฉริยะ (Intelligent Transportation System) และเป็นพื้นฐานของโพรโทคอลการแลกเปลี่ยนข้อมูลที่มีความซับซ้อนมากขึ้น

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1) ศึกษาวิธีการกระจายข้อมูลพื้นฐานบนเครือข่ายไร้สายแบบแอดฮอกสำหรับอุปกรณ์ทั่วไป และศึกษาวิธีการกระจายข้อมูลอย่างเชื่อถือได้ที่มีอยู่ในปัจจุบันสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ
2) ออกแบบวิธีการกระจายข้อมูลอย่างเชื่อถือได้สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ และพัฒนาโพรโทคอลเพื่อใช้ในการทดสอบการทำงานของการกระจายข้อมูลแบบเชื่อถือได้ตามที่ได้ออกแบบไว้ โดยประกอบด้วยขั้นตอนการพัฒนาดังต่อไปนี้
    a. พัฒนาโพรโทตามที่ออกแบบไว้เบื้องต้น พัฒนาชุดพฤติกรรมของรถยนต์แบบพื้นฐาน ได้แก่ เส้นทางตรงแทนการเคลื่อนที่บนทางหลวง และเส้นแบบตารางเพื่อแทนการเคลื่อนแบบในเมือง เพื่อนำมาใช้ในการทดสอบ เก็บข้อมูลการทำงานของโพรโทคอล และปรับปรุงส่วนที่มีการทำงานผิดพลาด
    b. ศึกษาผลการวิเคราะห์ในข้อมูลจากขั้นตอนข้างต้น นำมาพัฒนาส่วนประกอบอื่นๆของโพรโทคอล เพื่อความสมบูรณ์ในการทำในสภาพการจราจรจริง พัฒนาชุดพฤติกรรมของรถยนต์บนเส้นทางโดยอ้างอิงจากแผนที่จริงของกรุงเทพมหานคร ทดสอบ และเก็บข้อมูลการทำงานของโพรโทคอล
    c. นำโพรโทคอลที่ได้รับการพัฒนาทดสอบบนเครือข่ายจำลองที่มีลักษณะคล้ายคลึงกับการสื่อสารจริง รวมทั้งการพัฒนา และทดลองบนอุปกรณ์จริงที่มีบริการแบบพื้นฐานในการทดสอบการทำงาน
3) วิเคราะห์ผลการวิจัย และสรุปผลการวิจัย

บทที่ 2 การออกแบบโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ

การกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ (Density-Aware Reliable Broadcasting Protocol on Vehicular Ad-Hoc Networks : DECA) ถูกออกแบบโดยคำนึงถึงปัจจัยสำคัญ 3 ประการ ดังนี้

1) **ความเชื่อถือได้ (Reliability)** ซึ่งเป็นจุดประสงค์หลักในการทำงานของโพรโทคอล

2) **ค่าใช้จ่าย (Overhead)** ที่เกิดขึ้นจากการกระจายข้อมูล และการกระจายจากบีคอนซึ่งมีผลต่อระบบสื่อสารไร้สายแบบแอดฮอกที่มีทรัพยากรอย่างจำกัด

3) **ความเร็วในการกระจายข้อมูล (Speed of Data Dissemination)** ซึ่งยิ่งมีความเร็วสูงยิ่งทำให้ข้อมูลนั้นมีค่ามากขึ้น และส่งผลให้บริการในระดับผู้ใช้มีความแม่นยำมากขึ้น

ในบทนี้จะกล่าวถึงแนวคิดในการออกแบบโพรโทคอล หลักการทำงานของโพรโทคอล และการทดสอบสมรรถนะของโพรโทคอลในโปรแกรมจำลองเครือข่าย NS-2 ในสภาพการจราจรเสมือนจริงบนแผนที่อย่างง่าย

2.1 แนวคิดในการออกแบบ

การออกแบบสำหรับโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะมีแนวคิดในการออกแบบดังนี้

- *รถยนต์มักจะจับตัวกันเป็นกลุ่มบนถนน* ดังนั้นการเลือกโหนดที่มีเพื่อนบ้านรอบข้างสูงสุดในการกระจายข้อมูล ย่อมจะทำให้ข้อมูลนั้นครอบคลุมจำนวนโหนดได้มากกว่าโหนดอื่นๆ และหลีกเลี่ยงค่าใช้จ่ายที่จะเกิดขึ้นจากการกระจายซ้ำในบริเวณเดิมเพื่อครอบคลุมจำนวนโหนดที่เท่ากัน
- *ความเร็วในการกระจายข้อมูลขึ้นอยู่กับเวลารอ (Waiting Timeout)* ดังนั้นการหลีกเลี่ยงการใช้งานเวลารอให้น้อยที่สุดจะสามารถลดความล่าช้าที่จะเกิดขึ้นในการกระจายแต่ละครั้งได้ ส่งผลให้การทำงานของโพรโทคอลทำงานได้เร็วขึ้น
- *โพรโทคอลสามารถทำงานได้อย่างยืดหยุ่น* โดยใช้เพียงข้อมูลที่มีในการทำงานเท่านั้น ดังนั้นโพรโทคอลจะสามารถทำงานได้ดีแม้จะไม่มีข้อมูลทางด้านตำแหน่งและทิศทางหรือข้อมูลจากจีพีเอส

2.2 หลักการทำงานของโพรโทคอล DECA

หลักการทำงานที่สำคัญของโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะประกอบด้วยส่วนต่างๆดังต่อไปนี้

1) **Store-and-Forward** : การทำงานแบบ Store-and-Forward เพื่อรองรับการทำงานในสภาพที่มีการเชื่อมต่อเป็นช่วงๆที่เกิดขึ้นได้บ่อย โดยโหนดที่ได้รับข้อความจะเก็บข้อความนั้นไว้ในหน่วยความจำจนกว่าข้อความนั้นจะหมดอายุ โหนดที่เก็บข้อความสามารถส่งต่อข้อความนั้นให้แก่โหนดเพื่อนบ้านที่ยังไม่ได้รับข้อความนั้นได้

2) Beaconing with Adaptive Intervals : การใช้บีคอนในการค้นพบเพื่อนบ้าน แลกเปลี่ยนข้อมูล และใช้ในการตรวจสอบหาข้อความที่ยังไม่ได้รับ โดยการแลกเปลี่ยนบีคอนจะเกิดขึ้นภายในระยะสัญญาณการสื่อสารของโหนด (1-hop neighbor node) โดยระยะเวลาในการส่งสามารถปรับเปลี่ยนได้ตามความหนาแน่นของเครือข่ายขณะนั้น (Adaptive Beaconing Intervals)

3) Preferred Node Selection Algorithm : การให้โหนดต้นทางหรือโหนดก่อนหน้า (Source/Precursor Node) เป็นผู้กำหนดโหนดที่จะส่งต่อข้อความ โดยใส่หมายเลขเฉพาะของโหนด (Node ID) แนบไปกับข้อความที่ส่ง การเลือกโหนดส่งต่อจะเลือกจากโหนดที่มีความหนาแน่นบริเวณนั้นสูงที่สุด (จำนวนโหนดเพื่อนบ้านมีมากที่สุด) เพื่อลดจำนวนครั้งในการส่งต่อข้อความ หลีกเลี่ยงการส่งข้อความซ้ำในบริเวณเดิม และหลีกเลี่ยงการใช้เวลารอ

4) Waiting Timeout Calculation : โหนดจะใช้การตั้งเวลารอ (Waiting Time) แบบสุ่มสำหรับทุกโหนดที่ได้รับข้อความใหม่ที่ไม่ใช่โหนดที่ถูกเลือก กรณีที่โหนดที่ถูกเลือกไม่ทำงานหรือมีโหนดที่เห็นแก่ตัวทำงานอยู่ในระบบ โหนดอื่นสามารถส่งต่อข้อความ และเลือกโหนดส่งต่อใหม่จากรายชื่อเพื่อนบ้านของโหนดนั้นๆเอง

2.2.1 การเก็บข้อมูลของโพรโทคอล

รถยนต์แต่ละคัน หรือโหนดจะเก็บข้อมูลสำคัญ 3 ชุด คือ ข้อมูลของเพื่อนบ้าน คิวของข้อความที่จะถูกส่งต่อ และข้อความที่ยังไม่หมดอายุ

1) *ข้อมูลของเพื่อนบ้าน* จะประกอบด้วยหมายเลขประจำตัวของโหนด ความหนาแน่นของโหนด เพื่อใช้ในการเลือกโหนดที่จะส่งต่อข้อความ

2) *คิวของข้อความที่จะถูกส่งต่อ* เป็นรายละเอียดของข้อความพร้อมทั้งเวลาที่จะส่งข้อความออกไป คิวนี้ใช้เพื่อรอเวลาที่ข้อความนั้นจะถูกส่งต่อ แต่ในกรณีที่โหนดได้ยินโหนดเพื่อนบ้านส่งข้อความนั้นก่อน ข้อความก็จะถูกลบออกจากคิว เพื่อลดจำนวนการส่งของข้อความเดิมในบริเวณเดียวกัน ซึ่งคิวของข้อความที่จะถูกส่งต่อไม่ใช่หน่วยความจำที่ใช้ในการเก็บข้อมูล ซึ่งจะเก็บข้อความนั้นไว้จนกว่าข้อความนั้นหมดอายุ

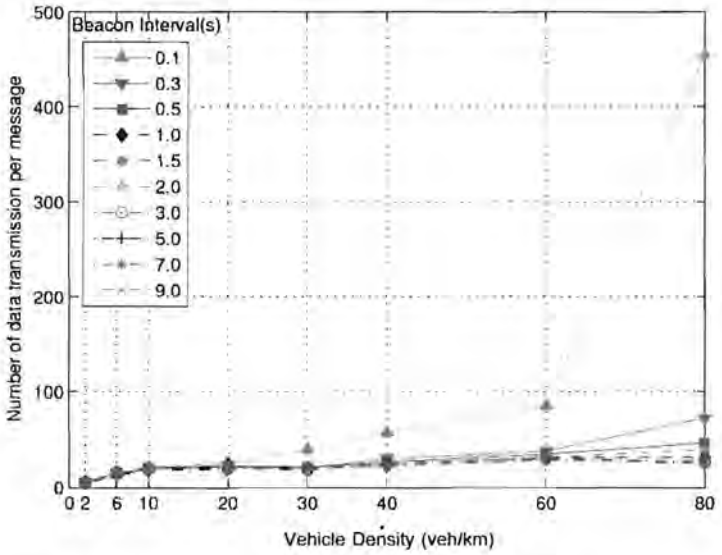
3) *ข้อความที่ยังไม่หมดอายุ* จะถูกเก็บไว้ตามหลักการทำงานแบบ Store-and-Forward เพื่อใช้ในการส่งให้กับโหนดเพื่อนบ้านที่ยังไม่รับข้อมูลกรณีที่มีการเชื่อมต่อเป็นช่วงๆไม่ต่อเนื่อง ข้อความจะถูกเก็บจนกว่าข้อความนั้นจะหมดอายุ

2.2.2 การแลกเปลี่ยนข้อมูลจากโหนดเพื่อนบ้าน (Beaconing)

ข้อมูลของโหนดเพื่อนบ้านจะได้จากการแลกเปลี่ยนข้อมูลผ่านบีคอน ซึ่งข้อมูลที่จะมีการส่งผ่านไปพร้อมกับบีคอน ประกอบด้วย

- หมายเลขเฉพาะตัวของโหนด
- ความหนาแน่นของพื้นที่ในบริเวณของโหนด โดยในที่นี้จะใช้จำนวนของเพื่อนบ้าน เป็นข้อมูลที่ใช้ในการเลือกโหนดที่จะส่งต่อข้อมูล โหนดที่มีความหนาแน่นสูงสุดจะถูกเลือก
- รายการของข้อความที่ได้รับ ประกอบด้วยโหนดที่แพร่ข้อความ และหมายเลขเฉพาะของข้อความ ใช้เพื่อตรวจสอบว่ามีข้อความที่ยังไม่ได้รับหรือไม่ รายการของความที่ได้รับจะไม่ถูกเก็บลงในข้อมูลเพื่อนบ้าน เมื่อทำการตรวจสอบเสร็จจะถูกลบออก

การแลกเปลี่ยนข้อมูลเพื่อนบ้านผ่านการทำบีคอนเป็นการเพิ่มค่าใช้จ่าย (Overhead) ให้กับการทำงานของโพรโทคอล ในปกติการส่งบีคอนจะเป็นเวลาคงที่ เช่น ทุกหนึ่งวินาทีหรือ 1 Hz. ซึ่งหากพิจารณาถึงความสำคัญในการทำงานของโพรโทคอลแล้ว การแลกเปลี่ยนบีคอนเกิดขึ้นเพื่อค้นหาโหนดเพื่อนบ้านในบริเวณการสื่อสาร ในกรณีที่มีความหนาแน่นของโหนดน้อย การทำบีคอนจำเป็นจะต้องมีความถี่สูงเพื่อให้สามารถพบเพื่อนบ้านได้เร็วที่สุด และในการกรณีที่มีความหนาแน่นสูงการทำบีคอนที่ความถี่สูงจะก่อให้เกิดปัญหาการชน และส่งผลต่อประสิทธิภาพของโพรโทคอลได้ ดังเช่นในกราฟรูปที่ 2-1 ซึ่งแสดงว่าการทำบีคอนที่ความถี่สูง แม้ว่าจะทำให้โพรโทคอลมีข้อมูลที่ทันสมัยตลอดเวลา แต่ส่งผลเสียต่อการใช้งานทรัพยากรที่มีอยู่อย่างจำกัด ปัญหาการชนการของข้อมูล ทำให้มีจำนวนการส่งต่อข้อความสูงขึ้นเมื่อเทียบกับความถี่ที่ต่ำกว่า



รูปที่ 2-1 กราฟแสดงจำนวนการส่งข้อความที่ช่วงเวลาการทำบีคอนต่างๆกัน

ดังนั้นการเลือกใช้ช่วงเวลาที่เหมาะสมในการทำบีคอนสำหรับความหนาแน่นของเครือข่ายในแต่ละพื้นที่สามารถลดค่าใช้จ่ายที่เกิดขึ้นระหว่างการทำงานของโพรโทคอลได้ โดยที่ความน่าเชื่อถือจากการกระจายของโพรโทคอลยังมีค่าเท่าเดิม สามารถแบ่งความหนาแน่นของเครือข่ายที่เกิดได้เป็น 2 ประเภท คือ

1) ความหนาแน่นของโหนด หากมีจำนวนโหนดหนาแน่น ในบริเวณนั้นจะมีจำนวนบีคอนจำนวนมาก ซึ่งเพิ่มโอกาสที่จะเกิดปัญหาการชนกัน

2) จำนวนของข้อความที่มีการกระจาย ในบริเวณที่มีการกระจายข้อความสูง ย่อมทำให้จำนวนการใช้ทรัพยากรในพื้นที่มีสูง ซึ่งเพิ่มโอกาสที่จะเกิดปัญหาการชนเช่นเดียวกัน

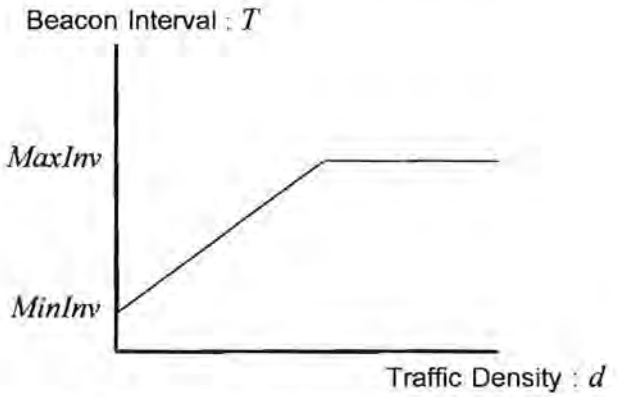
ความหนาแน่นที่จะเกิดขึ้นในการจราจรของเครือข่ายจึงขึ้นอยู่กับความหนาแน่นของโหนด และจำนวนของข้อความที่มีในระบบ สามารถเขียนอยู่ในรูปของสมการแสดงค่าความหนาแน่นของเครือข่ายได้ดังสมการที่ (1) โดยให้ d คือ ความหนาแน่นของเครือข่าย n คือ จำนวนโหนดเพื่อนบ้าน m คือ จำนวนของข้อความในระบบ, w_1 และ w_2 คือ ค่าถ่วงน้ำหนักของจำนวนโหนดเพื่อนบ้าน และจำนวนของข้อความในระบบตามลำดับ

$$d = (w_1 \times n) + (w_2 \times m) \quad (1)$$

ช่วงเวลาสำหรับการส่งบีคอนที่มีการเปลี่ยนแปลงตามความหนาแน่นของเครือข่าย (Adaptive Beacon Interval) สามารถใช้สมการเชิงเส้นอย่างง่ายมาใช้ในการคำนวณค่าที่เหมาะสมโดยใช้ค่าความหนาแน่นของเครือข่ายเป็นตัวกำหนด และมีการกำหนดช่วงเวลาสั้นสุดและยาวสุด เพื่อให้โพรโทคอลยังสามารถทำงานได้อย่างมีประสิทธิภาพ และสมรรถภาพคงเดิม เรียกการคำนวณช่วงเวลาแบบนี้ว่า การคำนวณช่างเวลาปรับตัวแบบเชิงเส้น (Linear Adaptive Algorithm: LIA)

การคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น (LIA) สามารถอธิบายได้ตามสมการที่ (2) โดย T คือ ช่วงเวลาสำหรับการทำบีคอนครั้งถัดไป *MinInv* คือ ช่วงเวลาสั้นสุด c เป็นค่าคงที่ในการเพิ่มช่วงเวลา d ความหนาแน่นของเครือข่าย และ *MaxInv* คือ ช่วงเวลายาวสุด สามารถนำมาเขียนกราฟได้ตามรูปที่ 2-2 การคำนวณช่วงเวลาจะทำหลังจากมีการกระจาย [u8vo]ไปแล้ว และจะใช้ช่างเวลาที่คำนวณได้เป็นเวลาที่จะแพร่บีคอนในครั้งถัดไป

$$T = min(MinInv + (c \times d), MaxInv) \quad (2)$$



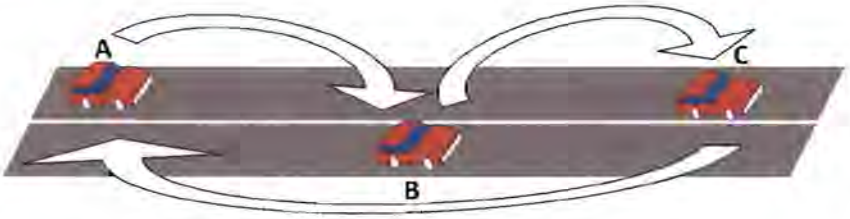
รูปที่ 2-2 กราฟแสดงการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น

2.2.3 การเลือกโหนดส่งต่อข้อความ (Preferred Node Selection Algorithm)

การทำงานเมื่อเริ่มการกระจายข้อความ โหนดที่เริ่มต้นการกระจายจะเลือกโหนดที่มีความหนาแน่นสูงที่สุดจากข้อมูลเพื่อนบ้านที่เก็บไว้ จากนั้นจึงแนบหมายเลขของโหนดนั้นพร้อมกับส่งข้อความออกไป โดยโหนดที่ได้รับข้อความนั้นแล้วพบว่าตัวเองเป็นโหนดที่ถูกเลือก ก็จะทำการเลือกโหนดที่ความหนาแน่นสูงสุดจากรายชื่อของตัวเอง แล้วแนบหมายเลขของโหนดนั้นลงไปกับข้อความก่อนส่งข้อความออกไป ซึ่งกระบวนการนี้จะเกิดขึ้นจนกว่ารถยนต์ในบริเวณได้รับข้อความนั้นทั้งหมด หรือข้อความนั้นหมดอายุ

การเลือกโหนดที่จะส่งข้อความต่อนั้น จะเลือกโหนดที่มีความหนาแน่นสูงสุด โดยที่โหนดนั้นจะไม่ใช่โหนดที่ส่งข้อความก่อนหน้า (Precursor Node) และโหนดที่ถูกเลือกจะทำงานก็ต่อเมื่อโหนดนั้นไม่เคยได้รับข้อความมาก่อน เพื่อป้องกันการเลือกโหนดวนซ้ำเดิม ดังเช่นในรูปที่ 2-3 ซึ่งสามารถเกิดขึ้นได้ในกรณีนี้ เนื่องจากโหนดจะไม่ทราบตำแหน่งของเพื่อนบ้านในการเลือก และไม่เก็บข้อมูลว่าโหนดเพื่อนบ้านมีข้อความใดที่ได้รับแล้ว

รูปที่ 2-3 ปัญหาการเลือกซ้ำ เกิดขึ้นโดย A เลือกโหนด B เป็นโหนดที่ส่งต่อข้อมูล จากนั้น B จึงเลือกโหนด C เป็นโหนดที่มีความหนาแน่นสูงสุดที่ไม่ใช่โหนดที่ส่งก่อนหน้า (Precursor Node) แต่เมื่อ C ทำการเลือกโหนดในข้อมูลเพื่อนบ้าน ซึ่งหากมี A อยู่ A จะเป็นโหนดที่ถูกเลือกโดยที่ A เป็นโหนดที่มีความหนาแน่นสูงสุดและ A ไม่ใช่โหนดที่มีการส่งข้อมูลก่อนหน้า ในกรณีนี้การทำงานของโพรโทคอลจะไม่ให้ A ส่งข้อมูลซ้ำ ซึ่งจะเกิดความสูญเปล่าในการส่งข้อความเดิมในบริเวณที่โหนดได้รับข้อความแล้ว แต่จะให้โหนดที่ได้รับข้อความนั้นเป็นครั้งแรกช่วยแชมการส่งข้อความแทน



รูปที่ 2-3 ปัญหาการเลือกโหนดส่งต่อแบบวนซ้ำ

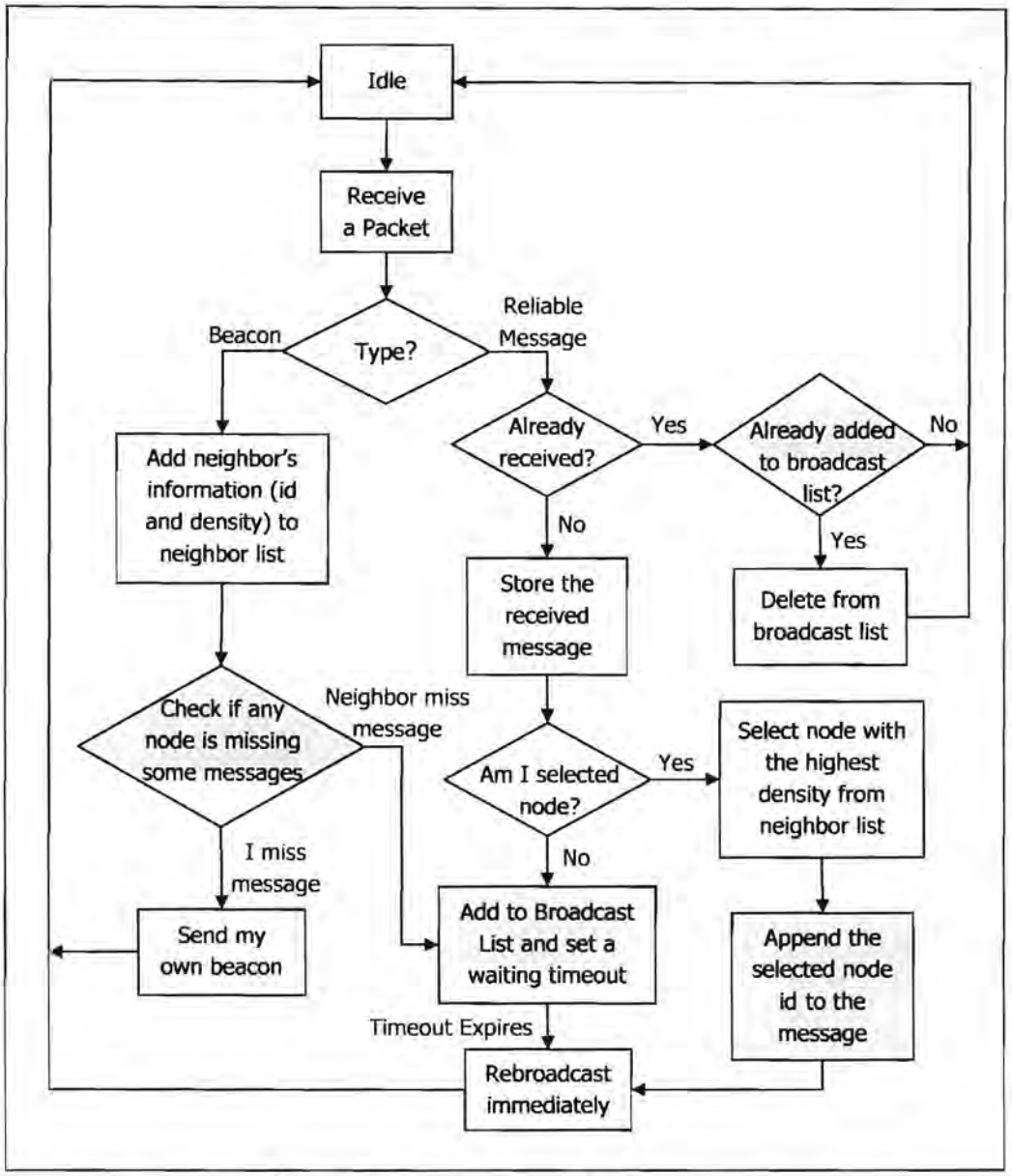
การช่วยแชมการส่งข้อความเป็นกระบวนการที่เกิดขึ้นเมื่อโหนดที่ถูกเลือกไม่ทำงานตามที่กำหนด เกิดขึ้นได้เมื่อโหนดที่ถูกเลือกได้รับข้อความนั้นแล้วเช่นกรณีข้างต้น หรือเมื่อเกิดการชนทำให้โหนดที่ถูกเลือกไม่ได้รับข้อความนั้น หรือเกิดจากโหนดที่เห็นแก่ตัว ซึ่งได้รับข้อความแล้วไม่ทำการส่งข้อความนั้นต่อ กระบวนการช่วยแชมนั้นจะเกิดขึ้นทันทีหลังจากที่โหนดที่ไม่ใช่โหนดที่ถูกเลือกได้รับข้อความใหม่ โหนดจะตั้งเวลาเพื่อรอการส่งข้อความนั้นซ้ำอีกครั้ง ซึ่งหากมีการส่งข้อความนั้นซ้ำ ข้อความที่ถูกตั้งเวลาไว้จะถูกลบออกจากคิว แต่ในกรณีที่โหนดนั้นไม่ได้ยินการส่งข้อความซ้ำอีกครั้งจนกระทั่งเวลารอที่ตั้งไว้หมด โหนดจะทำการส่งข้อความนั้นซ้ำอีกครั้ง พร้อมทั้งเลือกโหนดที่จะส่งต่อข้อความใหม่จากข้อมูลเพื่อนบ้านที่มีอยู่ ซึ่งเมื่อโหนดหนึ่งได้ทำการช่วยแชมโดยส่งต่อข้อความนั้นแล้ว โหนดอื่นๆที่อยู่ในบริเวณเดียวกันก็จะทำการลบข้อความนั้นออกจากคิว

เนื่องจากลักษณะเฉพาะของรถยนต์ที่ทำให้เกิดการเชื่อมต่อเป็นช่วงๆได้ ดังนั้นการใช้บีคอนจึงสามารถตรวจสอบได้เมื่อโหนดได้รับข้อความไม่ครบ โดยที่โหนดอื่นที่ได้รับบีคอนสามารถตรวจสอบได้ว่าข้อความแพร่ใดที่โหนดเพื่อนบ้านยังไม่ได้รับ รวมทั้งข้อความใดที่ตนเองยังไม่ได้รับเช่นกัน

- กรณีที่โหนดเพื่อนบ้านได้รับข้อความไม่ครบ โหนดจะนำข้อความที่เก็บไว้ใส่ในคิวเพื่อรอการส่งต่อ และตั้งเวลารอโดยการสุ่ม โหนดที่มีเวลารอน้อยที่สุดเท่านั้นจะทำการส่งข้อความให้โหนดเพื่อนบ้าน โหนดอื่นๆที่ได้ยินจะลบข้อความออกจากคิว ในการส่งข้อความในกรณีนี้ โหนดที่ส่งข้อความจะไม่เลือกโหนดที่จะส่งต่อข้อมูล เนื่องจากโหนดเพื่อนบ้านน่าจะมีข้อมูลของโหนดเพื่อนบ้านในบริเวณนั้นที่ดีกว่า

- กรณีที่โหนดพบว่าตนเองมีข้อความที่ขาดไป จะส่งบีคอนทันที เพื่อขอรับข้อความจากโหนดที่มีข้อความที่ตนเองไม่มีอยู่ ก่อนที่โหนดนั้นจะหายไปจากบริเวณนั้น

การทำงานของโพรโทคอลสามารถสรุปได้ตามผังงานรูปที่ 2-4

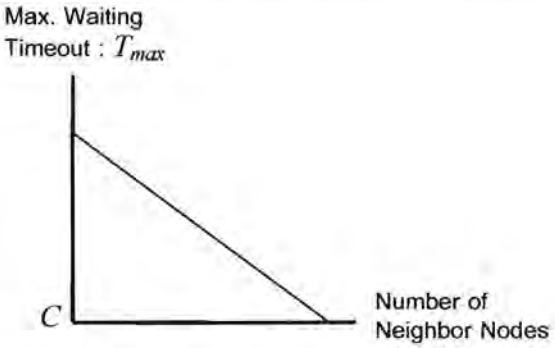


รูปที่ 2-4 ผังงานแสดงการทำงานของโพรโทคอล DECA

2.2.4 การคำนวณเวลารอ (Waiting Timeout Calculation)

เมื่อโหนดได้รับข้อความใหม่ แต่พบว่าหมายเลขโหนดที่ถูกเลือกไม่ตรงกับตัวเอง โหนดนั้นจะตั้งเวลารอเพื่อคอยฟังโหนดที่จะส่งข้อความต่อ แต่ในกรณีที่ไม่มีโหนดใดแพร่ข้อความนั้น ซึ่งอาจจะเกิดได้ในกรณีโหนดที่ถูกเลือกเปลี่ยนตำแหน่ง หรือเกิดความผิดพลาดในส่งข้อมูล หรืออาจจะเป็นเครือข่ายที่มีโหนดเห็นแก่ตัวอยู่ โหนดที่มีเวลารอสั้นสุดจะทำหน้าที่ส่งต่อข้อความนั้นแทน ดังนั้นระยะเวลาที่โหนดจะต้องรอนั้นจึงมีความสำคัญต่อประสิทธิภาพของโพรโทคอล คือ ในกรณีที่โหนดตั้งเวลารอนานเกินไปจะทำให้การกระจายครั้งถัดไปช้าลงทำให้ความเร็วในการกระจายลดลง หรือกรณีที่ตั้งเวลารอสั้นเกินไปก็อาจจะทำให้เกิดการกระจายซ้ำที่บริเวณเดียวกัน ก่อให้เกิดค่าใช้จ่ายที่เพิ่มขึ้นได้

กรณีที่มีการใช้งานเวลารอ แบ่งออกเป็น 2 กรณี คือ เมื่อโหนดที่ถูกเลือกไม่ทำงาน ดังที่กล่าวข้างต้น และกรณีที่โหนดเพื่อนบ้านมีข้อความที่ยังไม่ได้รับ ซึ่งทั้ง 2 กรณีมีวิธีการคำนวณเวลาเช่นเดียวกัน เนื่องจากโหนดใน DECA ทราบข้อมูลของความหนาแน่น หรือจำนวนโหนดเพื่อนบ้านเท่านั้น ดังนั้นหากโหนดที่มีจำนวนเพื่อนบ้านที่สุดทำการส่งต่อข้อมูล จะทำให้โอกาสในการกระจายครั้งนั้นครอบคลุมโหนดจำนวนมากขึ้นด้วย ดังนั้นการคำนวณเวลารอสูงสุดจะคำนวณโดยให้โหนดที่มีเพื่อนบ้านจำนวนมากที่เวลารอสั้นที่สุด เพื่อป้องกันการกระจายที่เวลาเดียวกันจึงใช้การสุ่มค่าในช่วง $(C, T_{max}]$ โดยที่ C มีค่าเป็นสองเท่าของค่าเวลาที่ใช้ในการส่ง (Propagation Delay) และ T_{max} เป็นเวลารอสูงสุดที่คำนวณได้จากจำนวนโหนดเพื่อนบ้าน สามารถอธิบายการคำนวณได้จากกราฟในรูปที่ 2-5

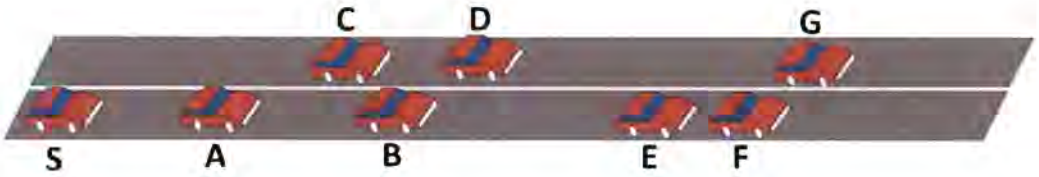


รูปที่ 2-5 กราฟแสดงการคำนวณเวลารอสูงสุด

2.3 ตัวอย่างการทำงานของโพรโทคอล

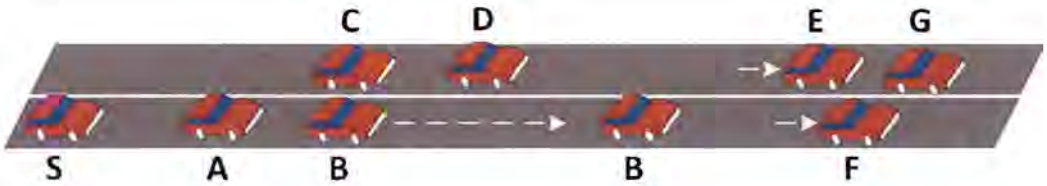
ในรูปที่ 2-6 เป็นการทำงานในลักษณะปกติ เมื่อโหนด S ต้องการเริ่มต้นแพร่ข้อมูลออกไป โหนด S มีโหนด A, B, C และ D เป็นโหนดเพื่อนบ้าน ในกรณีนี้ให้ D เป็นโหนดที่มีความหนาแน่นสูงสุด ดังนั้นในการกระจายข้อความ S จึงเลือก D แนบไปกับข้อความ หลังจากการกระจายข้อความของ S โหนด A, B, C และ D ได้รับข้อความพร้อมกัน เมื่อ D ได้รับข้อความ D ซึ่งทราบว่าตัวเองเป็นโหนดที่ถูกเลือกจะทำการส่งข้อมูลพร้อมทั้งเลือกโหนดส่งต่อข้อความต่อไป ซึ่งอาจจะเป็น E, F หรือ G

A, B และ C เมื่อได้รับข้อความแล้วพบว่าตนเองไม่ใช่โหนดที่ถูกเลือกจะนำข้อความใส่ลงในคิวและตั้งเวลา เพื่อรอการส่งต่อข้อความของ D ซึ่งในกรณีที่ D ไม่ได้ส่งต่อข้อความตามที่ S กำหนด A, B และ C จะทำการซ่อมแซม โดยโหนดที่มีเวลาในการคอยสั้นที่สุดจะเป็นผู้ส่งต่อข้อความ หาก B มีเวลารอสั้นสุด B จะทำการเลือกเพื่อนบ้านที่มีความหนาแน่นสูงสุด และส่งต่อข้อความ A และ C ที่ได้ยินการส่งต่อของ B จะลบข้อความนั้นออกจากคิว



รูปที่ 2-6 ลักษณะของรถในการเชื่อมต่อแบบปกติ

ในรูปที่ 2-7 แสดงการทำงานในกรณีที่มีการเชื่อมต่อเป็นช่วงๆ (Intermittent Connectivity) ในกรณีนี้สมมติให้ E, F และ G อยู่ห่างออกไปจากระยะสัญญาณสื่อสารของ D เมื่อ S เป็นโหนดเริ่มต้นการกระจายข้อความ และ D ถูกเลือก จากนั้น D จะเลือกโหนดส่งต่อถัดไปซึ่งอาจจะเป็น A, B และ C ซึ่งในกรณีนี้ A, B และ C จะไม่ส่งข้อความต่อ เนื่องจาก โหนดเหล่านี้ได้รับข้อความนั้นแล้ว แต่เมื่อเวลาผ่านไป B แซง C และ D ทำให้พบกับรถยนต์บริเวณด้านหน้าซึ่งคือ E, F และ G หลังจากได้รับบีคอน B จะทราบว่าทั้ง 3 โหนดได้รับข้อความไม่ครบถ้วน B จะส่งต่อข้อความแต่ไม่ระบุโหนดส่ง ต่อ เนื่องจากว่าในพื้นที่ใหม่นั้น E, F และ G น่าจะมีข้อมูลของโหนดเพื่อนบ้านที่ดีกว่า B ดังนั้นหลังจากที่ E, F และ G ได้รับข้อความก็จะตั้งเวลา โหนดที่มีเวลาสั้นสุดก็จะเลือกโหนดที่จะส่งต่อถัดไป แทนที่การเลือกของ B



รูปที่ 2-7 ลักษณะของรถในการเชื่อมต่อเป็นช่วงๆ

2.4 การทดลองและวิเคราะห์ผลการทดลอง

2.4.1 ตัววัดสมรรถนะของโพรโทคอล (Performance Metrics)

งานวิจัยนี้ได้ทำการวัดสมรรถนะของโพรโทคอลใน 4 ด้าน คือ

1) *ความเชื่อถือได้ (Reliability)* วัตถุประสงค์หลักในการออกแบบโพรโทคอลคือสามารถส่งผ่านข้อมูลให้กับ รถยนต์ในบริเวณพื้นที่หนึ่งหนึ่งให้ได้รับข้อมูลโดยอย่างเชื่อถือได้ ดังนั้นค่าความเชื่อถือได้เป็นค่าเปอร์เซ็นต์จากจำนวน รถยนต์ที่ได้รับข้อความต่อจำนวนรถยนต์ทั้งหมดในการทดลองแต่ละครั้ง ความเชื่อถือได้ที่สูงแสดงถึงสมรรถนะที่สูงของ โพรโทคอล

2) *ค่าใช้จ่าย (Overhead)* ในการกระจายข้อมูลโดยที่ยังมีความเชื่อถือสูงสุดย่อมมีค่าใช้จ่ายสูงกว่า ดังนั้นในการ ทดลองจึงมีการนับจำนวนครั้งการส่งข้อความในการกระจายข้อความซ้ำทั้งหมดเพื่อให้รถยนต์ทั้งหมดได้รับข้อความใน ช่วงเวลาที่กำหนด และจำนวนของบีคอนทั้งหมดที่เกิดขึ้น ค่าใช้จ่ายจะแสดงเป็นปริมาณของเครือข่ายที่ถูกใช้ในการทำงาน ต่อการกระจายหนึ่งข้อความ ซึ่งแสดงถึงประสิทธิภาพของโพรโทคอล

3) *ความเร็วของการกระจายข้อมูล (Speed of Data Dissemination)* เนื่องจากการบริการที่มีอยู่ในปัจจุบันสำหรับ รถยนต์ในระบบจราจรอัจฉริยะ เช่น การหลีกเลี่ยงอุบัติเหตุ หรือการวางแผนการเดินทาง ซึ่งข้อมูลที่บริการเหล่านี้ใช้จำเป็น

จะต้องเป็นข้อมูลที่ทันท่วงทีต่อเหตุการณ์ที่เกิดขึ้น ดังนั้นการกระจายข้อมูลจึงจะต้องทำได้รวดเร็วพอที่จะให้ข้อความที่แพร่ออกไป ยังมีประโยชน์ต่อระบบ ดังนั้นถ้าสามารถแพร่ข้อมูลได้รวดเร็วได้เท่าใดก็ยิ่งทำให้การกระจายข้อมูลครั้งนั้นมีโอกาสนำไปใช้ประโยชน์ได้เพิ่มมากขึ้น

4) *ความสำเร็จในการเลือกโหนดที่จะแพร่ข้อความต่อ (Preferred Selection Success Rate)* เนื่องจากประสิทธิภาพในการทำงานของโพรโทคอล DECA ขึ้นอยู่กับความสำเร็จในการเลือกโหนดส่งต่อ การเลือกโหนดถัดไปจะสำเร็จได้ก็ต่อเมื่อโหนดที่ถูกเลือกนั้นส่งต่อข้อความนั้นให้กับโหนดอื่นในระบบ ดังนั้นค่าความสำเร็จในการเลือกโหนดที่จะส่งข้อความแสดงถึงความสามารถในการเลือกโหนดของโพรโทคอล

2.4.2 เครื่องมือในการวัดสมรรถนะของโพรโทคอล

เนื่องจากโพรโทคอลถูกออกแบบเพื่อให้ทำงานในสภาพแวดล้อมที่เป็นการจราจรของรถยนต์ ซึ่งเป็นเรื่องยากที่จะมีการกำหนดการเคลื่อนไหวของรถยนต์บนถนน เพื่อให้ได้สภาพการทดลองในแต่ละโพรโทคอลที่เหมือนกัน นอกจากนี้การทดลองบนรถยนต์จริงยังต้องใช้งบประมาณที่สูง และไม่สามารถใช้โหนดจำนวนมากเพื่อทดลองได้ ดังนั้นในการวัดสมรรถนะของโพรโทคอลจึงใช้โปรแกรมจำลอง ซึ่งประกอบด้วยโปรแกรมจำลอง 2 ส่วน คือ

1) *โปรแกรมจำลอง SUMO v.0.10.3 (Simulation of Urban Mobility)* [20] ซึ่งจะจำลองพฤติกรรมเสมือนจริงของรถยนต์บนถนน สามารถกำหนดพฤติกรรมของรถยนต์ ความหนาแน่นของรถยนต์ และสามารถกำหนดลักษณะของถนนได้
2) *โปรแกรมจำลอง NS-2.34 (Network Simulation)* [21] เป็นโปรแกรมจำลองลักษณะการทำงานของเครือข่ายเสมือนจริง

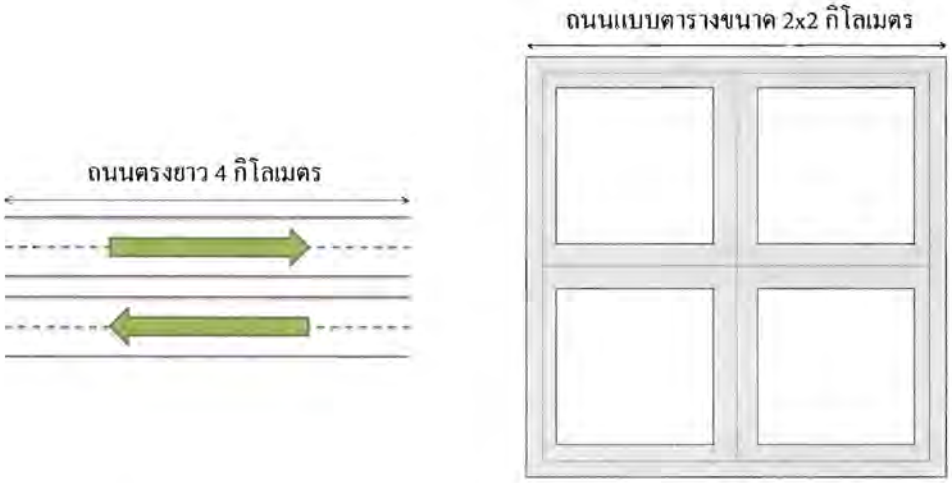
ในการทดลองเริ่มต้นด้วยการจำลองพฤติกรรมของรถยนต์โดยใช้ SUMO สร้างพฤติกรรมการวิ่งของรถยนต์บนถนน จากนั้นจึงนำบันทึกการวิ่งของรถยนต์มาแปลงจากรูปแบบ XML เป็นรูปแบบ tcl ที่สามารถใช้ได้บนโปรแกรมจำลอง NS-2.34 โดยใช้โปรแกรม TraNS 1.0 [22] ในการเปลี่ยนแปลงรูปแบบ

2.4.3 สภาพแวดล้อมที่ใช้ในการทดลอง

การทดลองจะแบ่งออกเป็นลักษณะของถนนทางหลวง และถนนในเมือง ถนนทางหลวงเป็นถนนทางตรงความยาว 4 กิโลเมตร มีช่องทางจราจรขนาด 4 ช่องทาง และถนนในเมืองเป็นถนนลักษณะตารางขนาด 2x2 ตารางกิโลเมตร ประกอบด้วยสี่แยก 1 แยก และสามแยกจำนวน 4 แยก ทั้งหมดไม่มีไฟจราจร แต่ละแยกห่างกัน 1 กิโลเมตร มีช่องทางจราจรขนาด 2 ช่องทาง ดังรูปที่ 2-8 ความหนาแน่นของรถยนต์โดยเฉลี่ยถูกแบ่งออกเป็นความหนาแน่นขนาดต่างๆ ดังตารางที่ 2-1 เพื่อทดสอบการทำงานของโพรโทคอล บนพฤติกรรมของรถยนต์ที่แตกต่างกัน และผลต่อประสิทธิภาพในการทำงานของโพรโทคอล สภาพการจราจรถูกจำลองโดยใช้โปรแกรมจำลอง SUMO รุ่น 0.10.3 และบันทึกการวิ่งถูกแปลงรูปแบบโดยใช้โปรแกรม TraNS รุ่น 1.0

ตารางที่ 2-1 การตั้งค่าต่างๆที่ใช้ในการทดลอง

การทดลอง	จำนวนครั้ง : 20 ครั้ง ผลการทดลองใช้ค่าเฉลี่ย ระยะเวลาในการทดลองต่อหนึ่งข้อความ : 20 วินาที
การสื่อสารไร้สาย	มาตรฐาน : IEEE802.11 ระยะเชื่อมต่อสูงสุด : 250 เมตร
สภาพการจราจร	ความหนาแน่นของรถยนต์ : 2, 6, 10, 20, 30, 40, 60, 80 คัน/กิโลเมตร ความเร็วสูงสุด : 50, 80 กิโลเมตร/ชั่วโมง
ข้อความ	อายุ : 10 วินาที ขนาด : 512 ไบต์
SFR	เวลารอสูงสุด : 2 วินาที
DECA	เวลารอสูงสุด : 0.2 วินาที ช่วงเวลาการทำบีคอน : LIA (ทุก 1.5-7 วินาที) $c = 0.2$, $MinInv = 1.5$, $MaxInv = 7$



ก) ถนนทางหลวง ข) ถนนในเมือง

รูปที่ 2-8 ลักษณะถนนที่ใช้ในการทดลอง

ในการทดลองใช้ NS-2.34 แต่ละครั้งจะมีข้อความที่ถูกแพร่จำนวน 1 ข้อความ ขนาด 512 ไบต์ ใช้ข้อความที่มีการกำหนดอายุ 10 วินาที ซึ่งหากข้อความหมดอายุจะสิ้นสุดการทดสอบครั้งนั้น และเก็บข้อมูลผลการทดลองทันที ทดลองระบบสื่อสารไร้สายของโหนดทำงานตามมาตรฐาน IEEE802.11 โดยที่มีการชนกันของสัญญาณตามปกติ มีระยะการเชื่อมต่อสูงสุดที่ 250 เมตร ใช้ Two-Ray Ground Propagation Loss Model โพรโทคอลที่ใช้ในการเปรียบเทียบมี ดังนี้

- *Simple Flooding (SF)* : วิธีการกระจายข้อมูลแบบดั้งเดิมที่ไม่ต้องการข้อมูลในการทำงาน และสามารถแพร่ข้อมูลได้เร็วที่สุด

- *Simple Flooding with Random Time of Rebroadcast (SFR)* : วิธีการกระจายข้อมูลเลียนแบบการทำงานของ SF แต่จะมีการหน่วงเวลาแบบสุ่มก่อนที่โหนดจะส่งต่อข้อมูลออกไป เพื่อจำลองลักษณะการทำงานของโพรโทคอลที่ใช้เทคนิค Store-and-Forward แบบง่ายๆ
- *AckPBSM* : โพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้ที่มีสมรรถนะดีที่สุดในงานวิจัยที่พบ ทำงานโดยใช้ข้อมูลจีพีเอส ใช้เทคนิค Store-and-Forward และใช้บีคอนในการทำงาน การตั้งค่าการทำงานเป็นไปตาม [11]
- *DECA* : การกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ ทำงานโดยไม่ใช้ข้อมูลจากจีพีเอส ใช้เทคนิค Store-and-Forward และใช้บีคอนในการทำงาน

การทำบีคอนของ DECA คำนวณได้จาก LIA ซึ่งการปรับแต่งค่าเป็นไปตามความเหมาะสมของช่วงเวลาที่ได้จากผลการทดลองในโปรแกรมจำลองเครือข่าย NS-2 สำหรับ DECA สรุปได้ตามตารางที่ 2-2 ส่วนการตั้งค่าอื่นๆที่กล่าวมาเป็นไปตามตารางที่ 2-1

ตารางที่ 2-2 ตารางแสดงช่วงเวลาการทำบีคอน ที่เหมาะสมสำหรับ DECA

ความหนาแน่นของรถยนต์ (คัน/กม.)	ช่วงเวลาการทำบีคอน (วินาที/บีคอน)
2 – 10	3
20 – 40	4
60 – 80	7

2.4.4 ผลการทดลองค่าความเชื่อถือได้ของโพรโทคอล

- **ค่าความเชื่อถือได้จากผลการทดลองบนถนนทางหลวง** (รูปที่ 2-9 ก))

เมื่อพิจารณาที่ความหนาแน่นต่ำ (2-10 คัน/กิโลเมตร) DECA สามารถแพร่ข้อมูลได้มากกว่าทุกโพรโทคอลที่นำมาเปรียบ โดยที่มีค่าความน่าเชื่อถือมากกว่าประมาณ 12.8% เนื่องจากโพรโทคอลพยายามที่จะแพร่ข้อมูลโดยไม่อาศัยการตั้งเวลารอซึ่งเพิ่มโอกาสในการกระจายถึงรถยนต์ที่มีโอกาสพบกันเพียงแค่ครั้งเดียวในระยะเวลาสั้นๆได้ เมื่อเปรียบเทียบกับ AckPBSM อย่างเชื่อถือได้ต่ำกว่า ส่วน SF และ SFR ที่ไม่ใช้ข้อมูลในการตัดสินใจ แต่ SFR จะให้สมรรถนะในการทำงานที่ดีกว่าเนื่องจากโหนดจะมีการหน่วงเวลาไว้ช่วงระยะเวลาหนึ่งทำให้เพิ่มโอกาสส่งต่อให้กับโหนดเพื่อนบ้านอื่นๆ มากกว่า SF ที่จะแพร่ข้อความทันทีที่ได้รับข้อความทันที ซึ่งแสดงถึงความแตกต่างของคุณสมบัติของโพรโทคอลแบบ Store-and-Forward

พิจารณาที่ความหนาแน่นปานกลาง (10-40 คัน/กิโลเมตร) DECA และ AckPBSM ที่ได้รับการออกแบบเพื่อทำงานบนรถยนต์สามารถทำการกระจายได้ 100% ส่วน SFR ที่มีคุณสมบัติ Store-and-Forward แบบง่ายๆ สามารถทำงานได้ดีกว่า SF ประมาณ 5% ซึ่งที่ความหนาแน่นในระดับนี้ยังมีปัญหาการเชื่อมต่อเป็นช่วงๆอยู่จึงทำให้ SF และ SFR ไม่สามารถให้ความเชื่อถือได้ที่ 100% แม้ว่า SFR จะมีการหน่วงเวลาในการส่งต่อข้อความแต่ก็ยังไม่เพียงในกรณีที่มีการขาดการเชื่อมเป็นเวลานาน

กรณีความหนาแน่นสูง (40-80 คัน/กิโลเมตร) ทุกโพรโทคอลสามารถทำงานได้ที่ ความเชื่อถือได้ 100% ยกเว้น SF และ SFR ที่ความหนาแน่น 40 คัน/กิโลเมตร

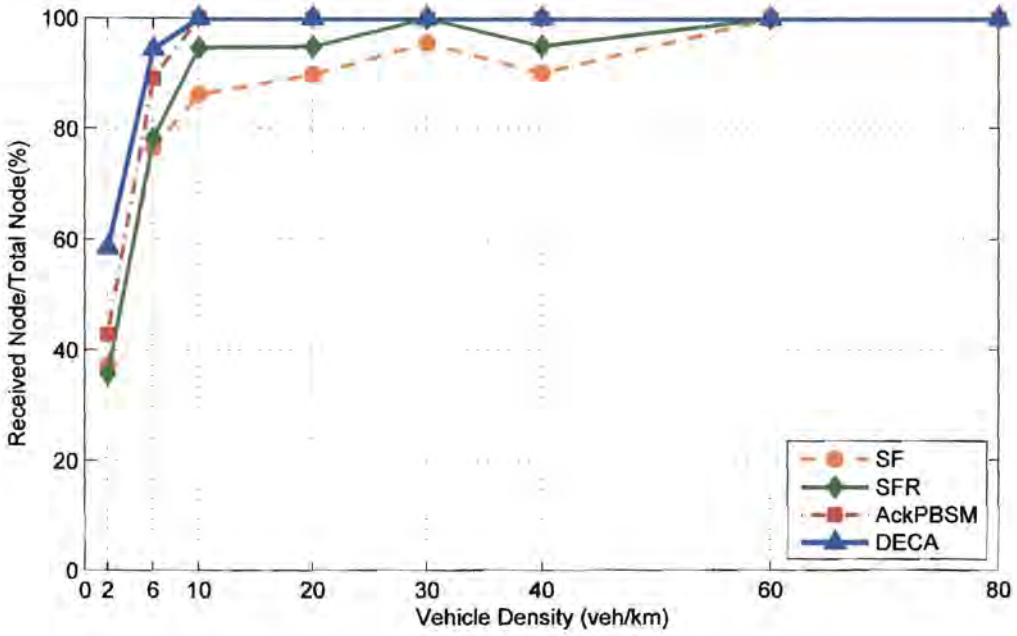
- **ค่าความเชื่อถือได้จากผลการทดลองบนถนนในเมือง** (รูปที่ 2-9 ข))

เมื่อพิจารณาที่ความหนาแน่นต่ำ (2-10 คัน/กิโลเมตร) เนื่องจากความซับซ้อนของถนนที่มากขึ้น ค่าความเชื่อถือได้ของทุกโพรโทคอลลดลง โดยที่ในระดับความหนาแน่นต่ำ DECA สามารถแพร่ข้อมูลได้มากกว่าทุกโพรโทคอลเช่นเดียวกับการทดลองบนถนนทางหลวง

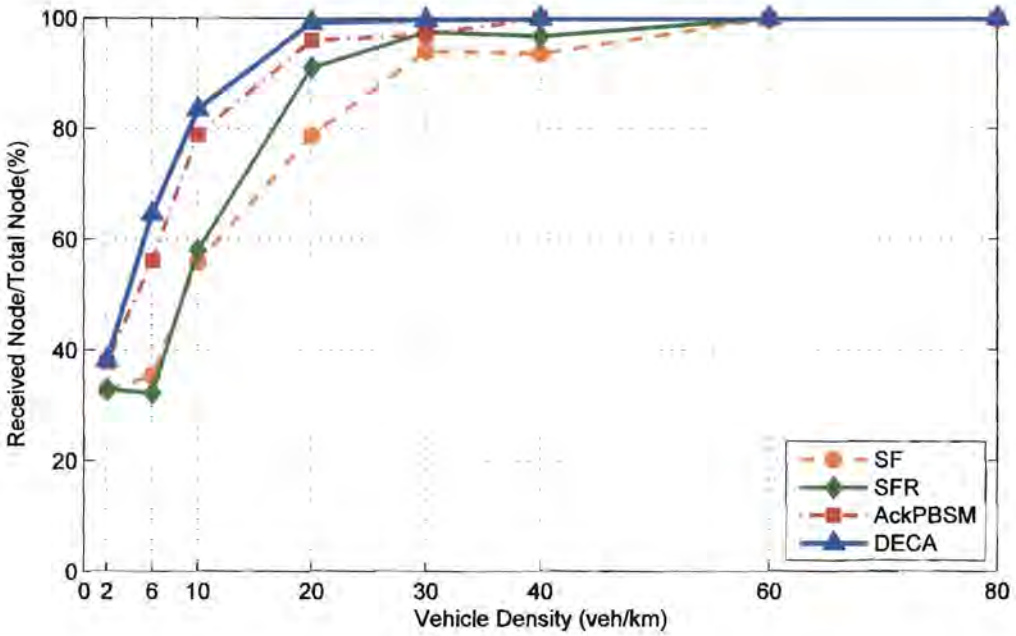
เมื่อพิจารณาที่ความหนาแน่นปานกลาง (10-40 คัน/กิโลเมตร) DECA มีค่าความเชื่อถือถึง 100% AckPBSM ที่ใช้เวลารอทั้งการกระจายแบบปกติ และการกระจายเมื่อเจอเพื่อนบ้านใหม่ ซึ่งในบางกรณีโหนดที่แพร่ข้อความไม่ได้อยู่ใน CDS ทำให้การกระจายช้าเกินไป และพลาดจากโหนดที่ต้องการส่งทำให้ค่าความเชื่อถือไม่ถึง 100% ส่วน SF และ SFR ก็มีปัญหากับการเชื่อมต่อเป็นช่วงๆเช่นเดียวกับที่ความหนาแน่นต่ำ ซึ่ง SF และ SFR จะไม่ส่งต่อให้กับเพื่อนบ้านที่ยังไม่ได้รับข้อความได้ ซึ่งเป็นปัญหาที่จะเกิดขึ้นกับโพรโทคอลอื่นๆที่ไม่มีสถานะภาพรับข้อความจากโหนดเพื่อนบ้าน

กรณีความหนาแน่นสูง (40-80 คัน/กิโลเมตร) ทุกโพรโทคอลสามารถทำงานได้ที่ ความเชื่อถือได้ 100% ยกเว้น SF และ SFR ที่ความหนาแน่น 40 คัน/กิโลเมตร

จากผลการทดลองสรุปได้ว่า DECA มีความเชื่อถือได้ในการกระจายข้อมูลทั้งบนถนนทางหลวง และถนนในเมืองมากกว่าโพรโทคอลอื่นๆ เนื่องจากแนวคิดในการเลือกโหนดที่จะแพร่ข้อมูลต่อโดยหลีกเลี่ยงการใช้เวลาคอย



รูปที่ 2-9 ก) กราฟแสดงค่าความเชื่อถือได้จากการทดลองบนถนนทางหลวง



รูปที่ 2-9 ข) กราฟแสดงค่าความเชื่อถือได้จากการทดลองบนถนนในเมือง

2.4.5 ผลการทดลองค่าใช้จ่ายของโพรโทคอล

ผลค่าใช้จ่ายในการทำงานของโพรโทคอลยิ่งมีค่าน้อยยิ่งแสดงถึงประสิทธิภาพของโพรโทคอล โดยผลการทดลองบนถนนทางหลวง และถนนในเมืองในรูปที่ 2-10 ก) และข) ตามลำดับ ซึ่งหากพิจารณาตามรูปกราฟจะสังเกตได้ว่าลักษณะค่าใช้จ่ายของทั้งการทดลองบนถนนทางหลวง และบนถนนในเมืองมีลักษณะสอดคล้อง สามารถนำมาพิจารณาแต่ละโพรโทคอลได้ดังนี้

โพรโทคอลที่มีค่าใช้จ่ายที่เกิดขึ้นสูงที่สุด คือ AckPBSM ซึ่งมีค่าใช้จ่ายสูงกว่า SF และ SFR ประมาณ 2 เท่าที่ทุกระดับความหนาแน่นของรถ ค่าใช้จ่ายที่เกิดขึ้นแบ่งออกเป็นส่วนที่เกิดจากการทำบีคอนที่มีความถี่สูงถึง 2 ครั้ง/วินาที โดยเป็นจากการออกแบบที่มีการตรวจสอบการรับข้อความของเพื่อนบ้านก่อนจะมีการส่งข้อความต่อ ทำให้ต้องการการทำ บีคอนที่มีความถี่สูงเพื่อลดความล่าช้าในการทำงาน อีกทั้งภายในบีคอนมีข้อมูลจีพีเอส และข้อความตอบรับจากโหนดเพื่อนบ้าน (Acknowledge Message) ที่มีขนาดใหญ่ จึงทำให้ขนาดของค่าใช้จ่ายที่เกิดจากการทำบีคอนมีขนาดประมาณ 55% ของค่าใช้จ่ายทั้งหมด อีกส่วนหนึ่งคือค่าใช้จ่ายที่เกิดจากการกระจายข้อมูล ซึ่ง AckPBSM ใช้จำนวนโหนดเพื่อนบ้านในการคำนวณค่าเวลารอ ดังนั้นในบริเวณของถนนส่วนเดียวกัน โหนดในบริเวณนั้นก็จะมีจำนวนเพื่อนบ้านเท่ากันเช่นกัน ทำให้เกิดการกระจายข้อมูลซ้ำในบริเวณเดียวกันเป็นจำนวนมาก แม้ว่าโหนดที่จะทำการกระจายข้อมูลนั้นมาจากโหนดที่อยู่ใน CDS ถึง 70-80% และจากกราฟเมื่อพิจารณาเพียงแค่ค่าใช้จ่ายที่เกิดจากการกระจาย AckPBSM จะมีค่าใช้จ่ายน้อยกว่า SF และ SFR เล็กน้อยในทุกๆความหนาแน่น

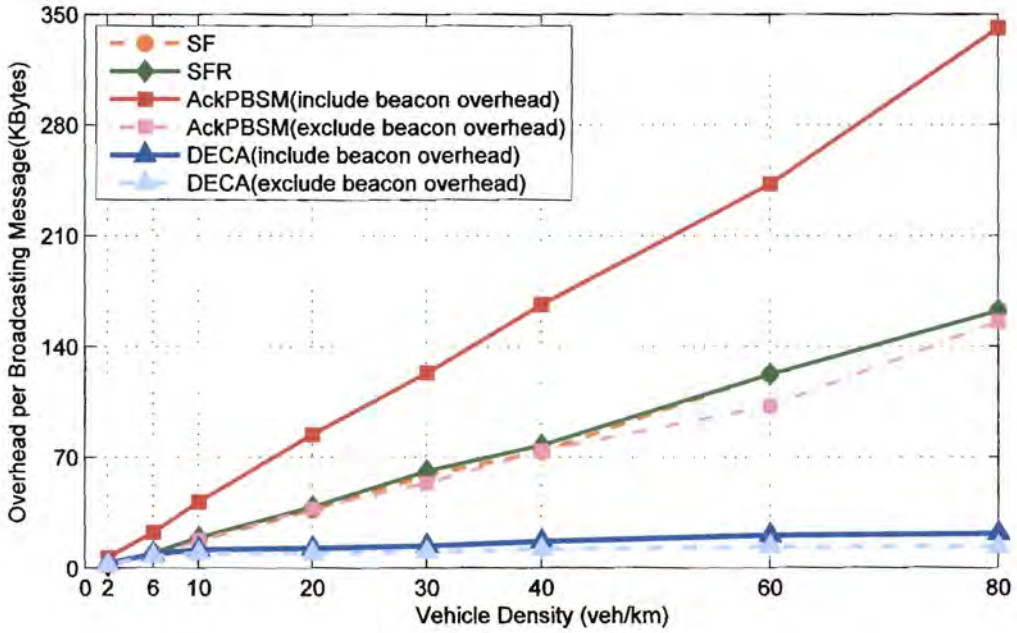
SF และ SFR ที่ไม่มีการทำบีคอนค่าใช้จ่ายจึงเกิดจากการกระจายข้อความซ้ำ ซึ่งโหนดทุกคนที่ได้รับข้อความใหม่จะแพร่ข้อความนั้นซ้ำอีกครั้ง ดังนั้นค่าใช้จ่ายที่เกิดขึ้นจึงสูง และจะมีค่าแปรผันตรงกับค่าความเชื่อถือได้ คือ ยิ่งมีรถยนต์ได้รับข้อความมากขึ้นก็จะยิ่งมีค่าใช้จ่ายได้การทำงานของโพรโทคอลมากขึ้น DECA ใช้วิธีการให้โหนดต้นทางหรือโหนดก่อนหน้าเลือกโหนดถัดไปในการกระจายข้อความต่อ ดังนั้นในบริเวณหนึ่งจะมีเพียงแค่โหนดเดียวเท่านั้นในการกระจายข้อความจึงมีจำนวนค่าใช้จ่ายที่เกิดขึ้นน้อยและเกือบคงที่เมื่อมีค่าความหนาแน่นของรถยนต์สูงขึ้น

เมื่อพิจารณา DECA ที่ใช้ความแน่นเท่านั้นในการเลือกโหนดถัดไป ซึ่งโหนดจะไม่สามารถรู้ตำแหน่งของโหนดเพื่อนบ้านได้เลย แต่ใช้ลักษณะการเคลื่อนที่ของรถยนต์ที่มักจะจับตัวกันเป็นกลุ่ม เช่น บนถนนทางหลวงที่รถยนต์วิ่งเกาะกลุ่มกัน หรือบนถนนในเมืองที่รถยนต์มันจะติดอยู่บริเวณสี่แยกที่มีไฟจราจร การเลือกโหนดที่มีความสูงสุดในการกระจายต่อจึงเป็นการรับประกันว่าในการกระจายครั้งนั้นจะมีจำนวนโหนดที่จะได้รับข้อความมากกว่า แต่การที่ทราบเฉพาะความหนาแน่นจะมีผลต่อการกระจายในสภาพแวดล้อมที่ซับซ้อนขึ้น เช่น ถนนในเมือง เนื่องจากการจับกลุ่มของรถจะหนาแน่นสูงในบริเวณหนึ่งๆ ทำให้การกระจายซ้ำเกิดขึ้นเป็นจำนวนมากขึ้น ข้อความจึงจะมีการกระจายตัวออกไป ดังนั้นค่าใช้จ่ายที่เกิดขึ้นเมื่อมีความหนาแน่นมากขึ้นจะมีจำนวนการกระจายข้อความซ้ำเพิ่มขึ้นด้วยเล็กน้อย เมื่อพิจารณาค่าใช้จ่ายที่เกิดขึ้นทั้งหมด DECA จึงเป็นโพรโทคอลที่มีค่าใช้จ่ายน้อยที่สุด อีกทั้งการที่ใช้เฉพาะข้อมูลความหนาแน่น ซึ่งได้จากการนับจำนวนของเพื่อนบ้านในบริเวณ 1 hop ทำให้ DECA มีความยืดหยุ่นในการทำงานสูงกว่า AckPBSM ที่ต้องใช้ข้อมูลจากจีพีเอสในการทำงานเท่านั้น

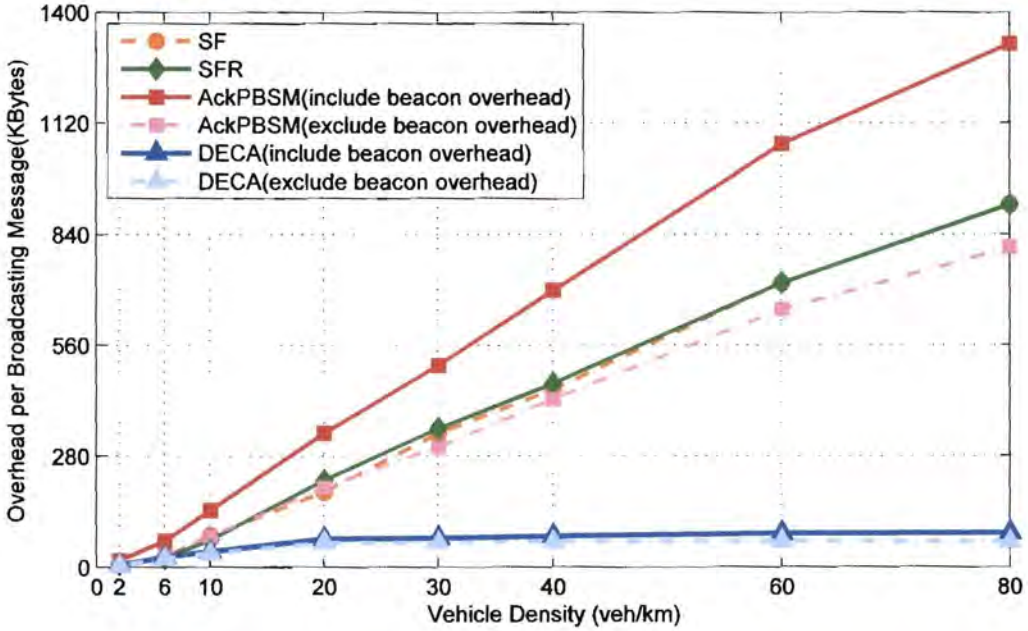
เนื่องจากค่าใช้จ่ายที่เกิดขึ้นจากการทำบีคอนเป็นค่าใช้จ่ายที่สูงเมื่อเปรียบเทียบจากค่าใช้จ่ายที่เกิดจากการกระจายข้อความซ้ำซึ่งเป็นกระบวนการสำคัญในการกระจายข้อมูล ดังนั้นงานวิจัยจึงสร้างการทดลองเพื่อเปรียบค่าใช้จ่ายที่เกิดขึ้นจากการทำบีคอนแบบใช้ช่วงเวลาคงที่ของ DECA โดยมีค่าความเชื่อถือ และค่าใช้จ่ายในการกระจายซ้ำใกล้เคียงกับการช่วงเวลาแบบปรับตัว โดยมีผลของค่าใช้จ่ายที่เกิดขึ้นตามกราฟในรูปที่ 2-11

ในการทดลองนี้แสดงในเห็นว่าการใช้ช่วงเวลาการทำบีคอนที่มีการปรับตัวได้ตามสภาพความหนาแน่นของเครือข่ายสามารถลดค่าใช้จ่ายได้เป็นจำนวนมาก DECA ใช้การคำนวณช่วงเวลาปรับแบบเชิงเส้น (LIA) ซึ่งมีช่วงการทำบีคอนระหว่าง 1.5-7 วินาที สามารถลดค่าใช้จ่ายที่เกิดขึ้นจากการทำบีคอนได้ประมาณ 57%

เมื่อเปรียบเทียบ DECA กับ AckPBSM จะเห็นได้ว่าค่าใช้จ่ายที่เกิดขึ้นจากบีคอนของ DECA มีค่าน้อยกว่ามาก เนื่องจากการทำงาน AckPBSM ที่ขึ้นอยู่กับข้อความตอบรับจากเพื่อนบ้าน ดังนั้นการเปลี่ยนช่วงเวลาตามสภาพความหนาแน่นของเครือข่ายไม่สามารถทำได้โดยตรง เพราะจะส่งผลกระทบต่อการเวลารอของโหนดที่จะแพร่ข้อความต่อไป แต่โดยดัดแปลงการทำงานของ AckPBSM เพื่อให้สามารถใช้งานบีคอนที่มีช่วงเวลาแบบปรับตัวได้ จากการทดลองโดยที่ให้ AckPBSM ยังมีค่าความน่าเชื่อถือ และค่าใช่จ่ายจากการกระจายคงเดิมเช่นเดียวกับการทำบีคอนแบบช่วงเวลาคงที่ การใช้บีคอนแบบช่วงเวลาปรับตัวได้นั้นสามารถลดค่าใช้จ่ายได้เพียง 24% เท่านั้น เนื่องจากความไม่ยืดหยุ่นในการทำงานของโพรโทคอล

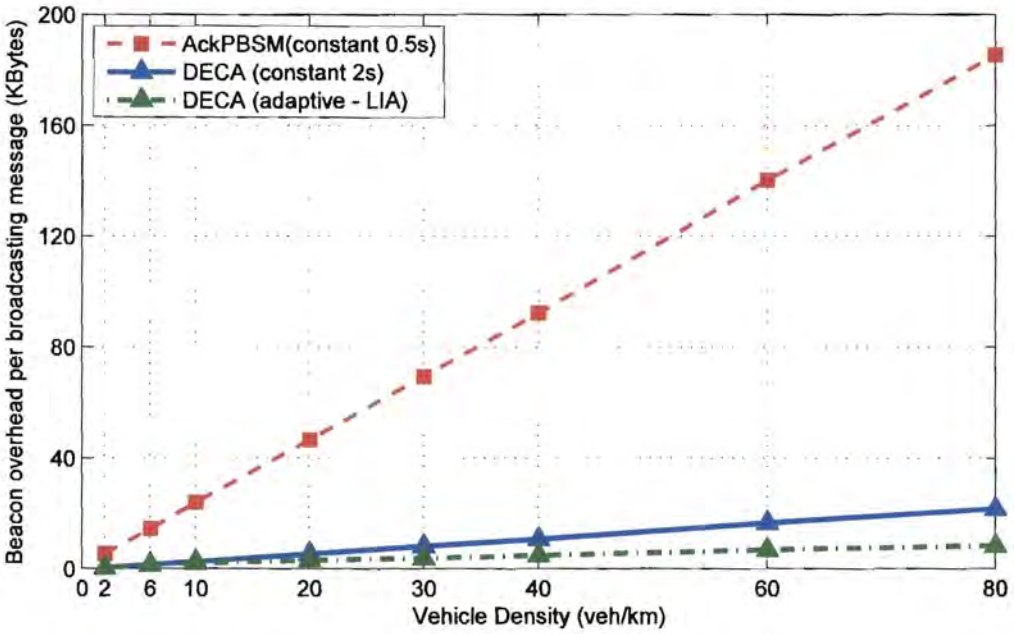


ก) ถนนทางหลวง

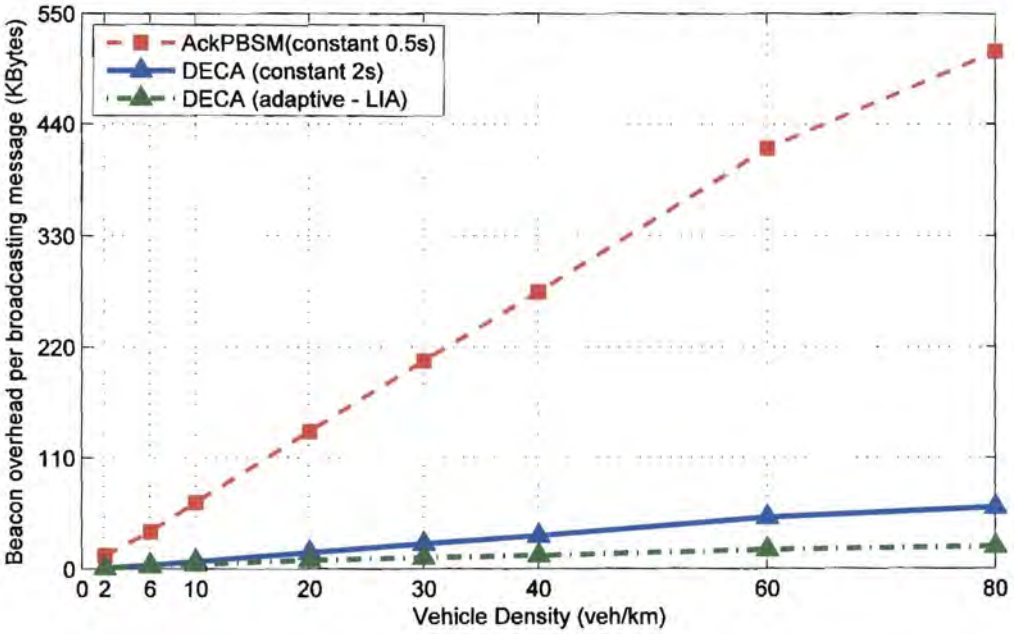


ข) ถนนในเมือง

รูปที่ 2-10 กราฟแสดงค่าใช้จ่ายทั้งหมด



ก) ถนนทางหลวง



ข) ถนนในเมือง

รูปที่ 2-11 กราฟแสดงค่าใช้จ่ายจากการส่งบีคอน

2.4.6 ผลการทดลองความเร็วในการกระจายข้อมูล

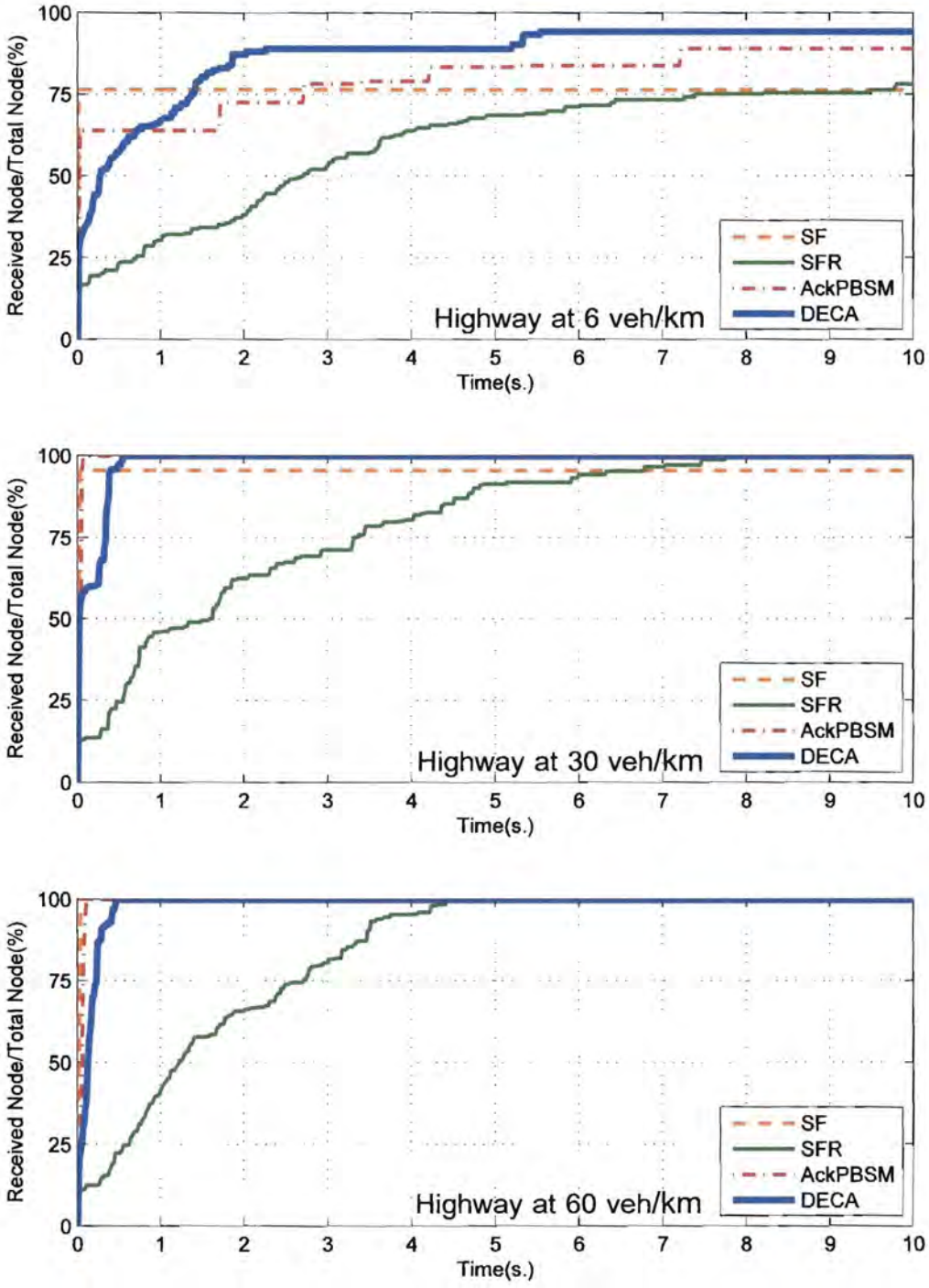
ผลการทดลองความเร็วในการกระจายข้อมูล เมื่อเปรียบเทียบที่เวลาเดียวกันโพรโทคอลที่มีค่าความเชื่อถือสูงกว่าแสดงถึงความเร็วในการกระจายที่เร็วกว่า ผลแบ่งออกเป็น 2 รูป คือ รูปที่ 2-12 และ 2-13 แสดงความเร็วในการกระจายข้อมูลจากการทดลองบนถนนทางหลวง และบนถนนในเมืองตามลำดับ โดยผลการทดลองจะแสดงตัวแทนของความหนาแน่นในแต่ละช่วงดังนี้ ความหนาแน่นที่ 6 คัน/กิโลเมตรเป็นความหนาแน่นต่ำ ความหนาแน่นที่ 30 คัน/กิโลเมตรเป็นความหนาแน่นปานกลาง และความหนาแน่นที่ 60 คัน/กิโลเมตรเป็นความหนาแน่นสูง

ในทุกการทดลองและความหนาแน่น SF จะเป็นโพรโทคอลที่มีความเร็วสูงที่สุดเสมอ เนื่องจากโหนดจะแพร่ข้อความทันทีที่ได้รับข้อความใหม่ แต่เนื่องจากไม่สามารถทำงานในสภาพที่มีการเชื่อมต่อเป็นช่วงๆได้จึงมีความหนาแน่นต่ำ และ SFR จะเป็นโพรโทคอลที่มีการทำงานช้าที่สุดเสมอเนื่องจากใช้การสุ่มเวลาหน่วงในก่อนจะมีการกระจายข้อความต่อไปให้โหนดอื่น จึงเป็นโพรโทคอลที่มีความเร็วในการกระจายข้อความช้าที่สุดในการทดลอง

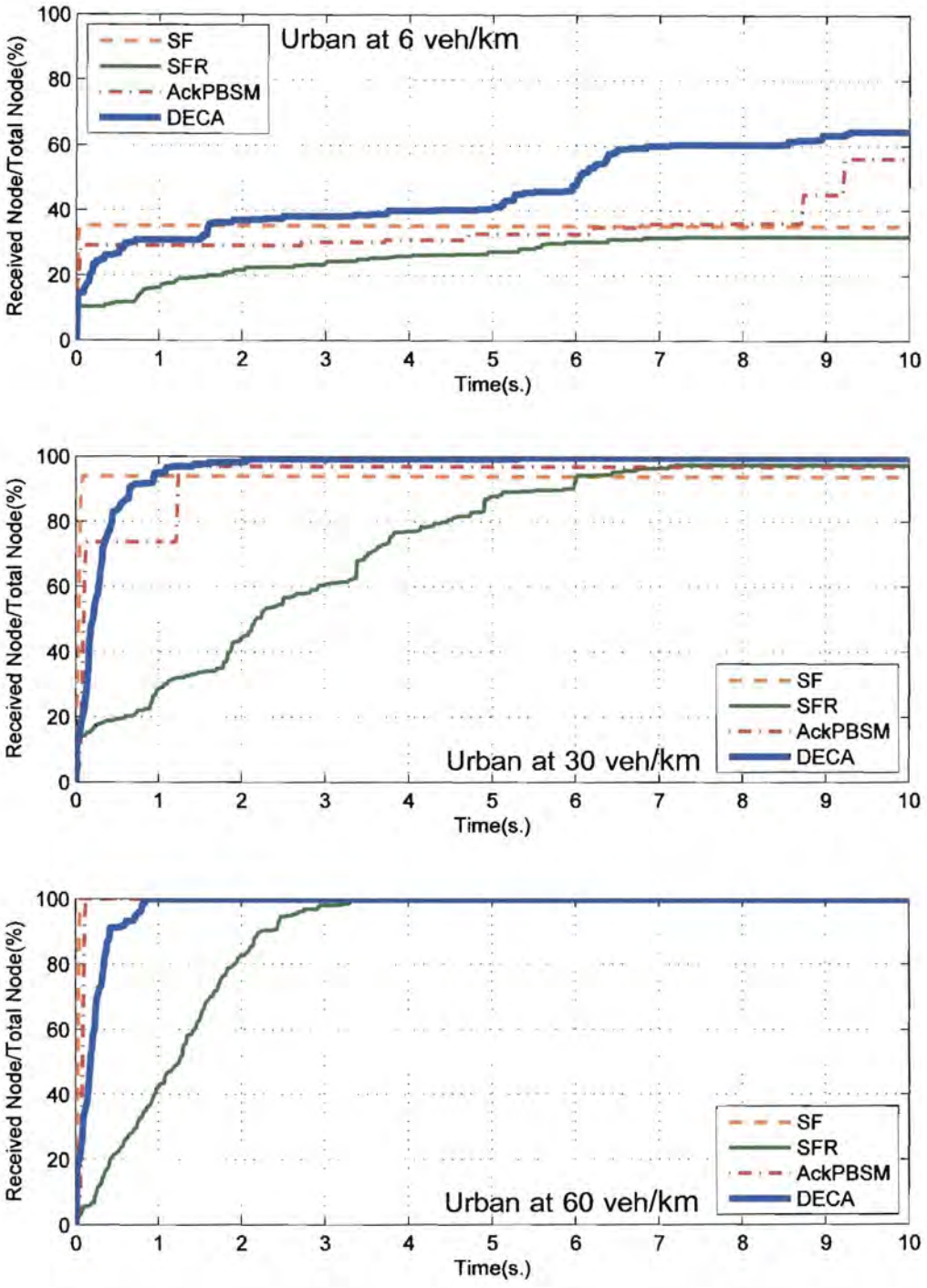
ความเร็วในการกระจายข้อความของ DECA จะมีความเร็วช้ากว่า AckPBSM เนื่องจากการเลือกโหนดจะมีโอกาสที่เลือกโหนดซ้ำเดิม และทำให้ต้องใช้การตั้งเวลารอเพื่อทำงานต่อ เพราะในการกระจายข้อมูลนั้น DECA จะไม่สามารถรู้ถึงตำแหน่งของโหนดถัดไปที่เลือกทำให้ต้องมีการกระจายหลายครั้งเพื่อให้กระจายตัวไปยังบริเวณอื่นๆที่โหนดยังไม่ได้รับข้อความนั้น แต่ด้วยวิธีการทำงานที่พยายามหลีกเลี่ยงการตั้งเวลาคอยจึงทำให้ความเร็วในการกระจายข้อความของ DECA ช้ากว่า AckPBSM เพียงเล็กน้อย และความแตกต่างจะน้อยลงเมื่อมีความหนาแน่นเพิ่มขึ้น

AckPBSM สามารถแพร่ข้อมูลได้อย่างมีความเร็วสูง แต่เนื่องด้วยการที่มีค่าใช้จากการกระจายซ้ำเป็นจำนวนมาก ดังนั้นความเร็วในการกระจายข้อมูลจึงมีพฤติกรรมที่ใกล้เคียงกับ SF ที่มีความเร็วในการกระจายช่วงเริ่มต้นสูง แต่เมื่อเวลาผ่านไปจะช้าลง เนื่องจากกระบวนการตั้งเวลารอของโพรโทคอล แต่อย่างไรก็ตามการที่มีการกระจายซ้ำเป็นจำนวนมากย่อมจะส่งผลในการทำงานที่มีความหนาแน่นของเครือข่ายสูงมากได้

การออกแบบให้ DECA ทำงานโดยหลีกเลี่ยงการใช้เวลารอนั้น สามารถทำให้การกระจายข้อมูลมีความเร็วสูงได้โดยที่มีค่าใช้จ่ายน้อยกว่ามาก เมื่อเปรียบเทียบกับโพรโทคอลอื่นๆ



รูปที่ 2-12 กราฟแสดงความเร็วในการกระจายข้อมูลบนถนนทางหลวงที่ความหนาแน่นแตกต่างกัน



รูปที่ 2-13 กราฟแสดงความเร็วในการกระจายข้อมูลบนถนนในเมืองที่ความหนาแน่นแตกต่างกัน

2.4.7 ผลการทดลองค่าความสำเร็จในการเลือกโหนดแพร่ข้อความต่อ

เนื่องจากประสิทธิภาพในการทำงานของ DECA ขึ้นอยู่กับความสำเร็จในการเลือกโหนดที่จะแพร่ข้อความถัดไป ซึ่งจะทำให้การทำงานของโพรโทคอลไม่ต้องพึ่งการตั้งเวลารอในกรณีที่การเลือกโหนดไม่สำเร็จ

ผลการทดลองค่าความสำเร็จในการเลือกโหนดของ DECA ในตารางที่ 2-3 สามารถประมาณค่าความสำเร็จในการเลือกโหนดบนถนนทางหลวงโดยเฉลี่ยได้ 74% และบนถนนในเมืองได้ 66% ซึ่งค่าความสำเร็จในการเลือกของ DECA มีค่าไม่สูง เนื่องจาก DECA ไม่ทราบตำแหน่งของเพื่อนบ้าน และเลือกโหนดที่ความหนาแน่นสูงสุด โดยที่การเลือกที่ไม่สำเร็จแบ่งออกได้จาก 2 สาเหตุ คือ การเลือกโหลดที่ได้รับข้อความนั้นแล้วซึ่งคิดเป็น 84.5% ของความผิดพลาดทั้งหมด และความผิดพลาด 16.5% เกิดจากการเลือกโหนดที่ไม่อยู่ในระยะของการเชื่อมต่อแล้ว ซึ่งหากใช้ข้อมูลตำแหน่งจากจีพีเอสจะทำให้โพโทคอลสามารถเลือกโหนดได้อย่างแม่นยำมากขึ้น แต่เป็นการลดความยืดหยุ่นในการทำงานของโพรโทคอล

ตารางที่ 2-3 ตารางแสดงค่าความสำเร็จในการเลือกโหนดแพร่ข้อความต่อของ DECA (%)

ความหนาแน่น (คัน/กม.) / ถนน	2	6	10	20	30	40	60	80
ทางหลวง	71.4	54.5	80.7	76	82.6	75	63.3	88.9
ในเมือง	75	72.2	64.3	64.4	54.8	57.1	64.7	75.9

บทที่ 3 การปรับปรุงสมรรถนะโพรโทคอลการกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ

การกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ (DECA) เป็นการออกแบบ และทดลองผลการทำงานในเบื้องต้นเพื่อให้ทราบถึงสมรรถนะของโพรโทคอลในด้านต่างๆ ตรงตามความต้องการในการออกแบบ ดังนั้นเพื่อให้โพรโทคอลสามารถทำงานได้ดีบนเครือข่ายจริงจึงต้องมีการพิจารณาการทำงานในด้านต่างๆเพิ่มเติมเพื่อให้ครอบคลุมในการทำงานทั้งหมด

ในบทนี้จะประกอบด้วยการศึกษากระบวนการคำนวณเวลาในการทำบีคอนที่เหมาะสมกับความหนาแน่นของเครือข่ายในรูปแบบต่างๆเพื่อลดค่าใช้จ่ายโดยรวม การออกแบบวิธีการจัดการบัฟเฟอร์ในการเก็บข้อมูลเมื่อมีพื้นที่ในการเก็บข้อมูลที่จำกัด และการปรับปรุงกระบวนการทำงานของโพรโทคอลในสภาพการจราจรที่มีความหนาแน่นสูง

3.1 การศึกษาการส่งบีคอนแบบปรับค่าช่วงเวลาได้ในรูปแบบต่างๆ

3.1.1 แนวคิดในการศึกษาการส่งบีคอนแบบปรับค่าช่วงเวลาได้

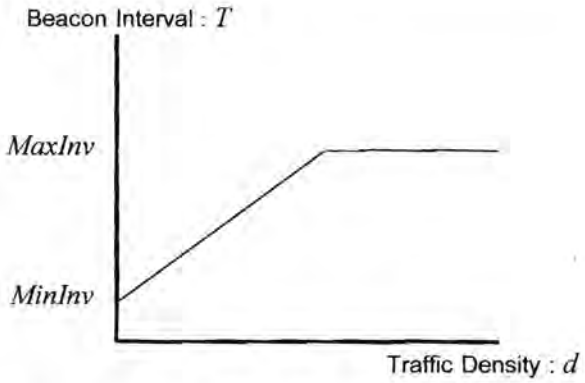
เนื่องจากการแลกเปลี่ยนข้อมูลจากการทำบีคอนมีความจำเป็นสำหรับเครือข่ายไร้สายแบบแอดฮอกเพื่อใช้ในการแลกเปลี่ยนข้อมูลพื้นฐานของแต่ละโหนด และใช้ในการค้นพบโหนดเพื่อนบ้าน ดังนั้นการเลือกช่วงเวลาในการทำบีคอนในแต่ละความหนาแน่นย่อมมีความสำคัญ และการที่ช่วงเวลาในการทำบีคอนในแต่ละช่วงความหนาแน่นย่อมจะส่งผลต่อสมรรถนะในการทำงาน และประสิทธิภาพของโพรโทคอลได้ มีงานวิจัยที่เกี่ยวข้องซึ่งเป็นการปรับค่าการส่งบีคอนอย่างง่าย [14] และใช้ข้อมูลจีพีเอส [16] ช่วยซึ่งไม่เหมาะกับการใช้งานในวิจัยนี้ ในเบื้องต้นของการศึกษาผู้วิจัยทดลองโพรโทคอล DECA โดยการใช้ช่วงเวลาการทำบีคอนแบบคงที่เพื่อหาช่วงเวลาที่เหมาะสมที่สุดในแต่ละความหนาแน่น โดยปัจจัยสำคัญคือการค้นหาวิธีการปรับช่วงเวลาที่มีค่าความเร็วในการกระจายสูงที่สุด และมีค่าใช้จ่ายในการทำงานต่ำที่สุด

ในการทดลองใช้การตั้งค่าและสภาพแวดล้อมเช่นเดียวกับการทดลองในบทที่ผ่านมา โดยใช้ช่วงเวลาในการทำงานแบบคงที่ 0.1, 0.3, 0.5, 1.0, 1.5, 2.0, 3.0, 5.0, 7.0 และ 9.0 วินาทีต่อบีคอน โดยค่าของช่วงเวลาที่เหมาะสมจะเป็นไปตามตารางที่ 3-1

ผลการทดลองนี้จะนำมาใช้เป็นชุดข้อมูลตัวอย่างเพื่อปรับช่วงเวลาในการส่งบีคอน โดยใช้วิธีการปรับช่วงเวลาในการส่งบีคอน โดยในขั้นแรกในการออกแบบโพรโทคอล DECA ใช้การส่งบีคอนแบบปรับค่าช่วงเวลาแบบเชิงเส้น หรือ Linear Adaptive Algorithm (LIA) ซึ่งมีรายละเอียดตามบทที่ 2.2.2 โดยมีลักษณะการลดความถี่ในการส่งบีคอนสัมพันธ์กับค่าความหนาแน่น ดังรูปที่ 3-1 จากนั้นจึงทดลองการปรับช่วงเวลาโดยการใช้วิธีการทางด้านสถิติ คือ การวิเคราะห์การถดถอยเชิงเส้น (Linear Regression Analysis) การใช้วิธีการด้านการเรียนรู้ของเครื่อง คือ K-Nearest Neighbor และการปรับปรุงวิธีการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้นโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน แล้วจึงนำมาเปรียบเทียบประสิทธิภาพในการทำงานกับ LIA

ตารางที่ 3-1 ช่วงเวลาในการส่งบีคอนที่เหมาะสมกับความหนาแน่น

ความหนาแน่นของรถยนต์ (คัน/กิโลเมตร)	ช่วงเวลาในการส่งบีคอน (วินาที/บีคอน)
2	1.5
6	3
10	7
20	7
30	9
40	9
60	9
80	9



รูปที่ 3-1 กราฟแสดงการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น

3.1.2 หลักการส่งบีคอนแบบปรับค่าช่วงเวลาได้แบบต่างๆ

1) การวิเคราะห์การถดถอยเชิงเส้น (Linear regression analysis) เป็นวิธีการทางด้านสถิติที่สามารถหาตัวแบบทางคณิตศาสตร์ (Math Model) โดยหาความสัมพันธ์ระหว่างตัวแปรตาม (Dependent Variable) ซึ่งเป็นตัวแปรที่ต้องการทราบค่า กับตัวแปรอิสระ (Independent Variable) หรือตัวแปรที่ทราบค่าแล้ว โดยที่ตัวแปรอิสระนั้นส่งผลกระทบต่อตัวแปรตาม ซึ่งสามารถนำมาประยุกต์ใช้กับการปรับช่วงเวลาในการส่งบีคอนดังนี้

- นำชุดข้อมูลตัวอย่างที่ได้จากการทดลอง มาหาความสัมพันธ์ระหว่างความหนาแน่นของเครือข่าย (ตัวแปรอิสระ) และช่วงเวลาในการส่งบีคอน (ตัวแปรตาม) ซึ่งจะได้สมการออกมา คือ

$$\hat{Y} = a + bX \tag{3}$$

เมื่อ $\hat{Y}$ คือ ช่วงเวลาในการส่งบีคอน

X คือ ความหนาแน่นของเครือข่าย

a, b คือ ค่าสัมประสิทธิ์การถดถอย ซึ่งคำนวณดังสมการ (4) และ (5)

$$a = \bar{y} - b\bar{x} \tag{4}$$

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{5}$$

เมื่อ $\bar{y}$ คือ ค่าเฉลี่ยของข้อมูลช่วงเวลาในการส่งบีคอน (y) ทั้งหมด

$\bar{x}$ คือ ค่าเฉลี่ยของข้อมูลความหนาแน่นของเครือข่าย (x) ทั้งหมด

สามารถนำสมการนี้ไปใช้ในการคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

- โหนดนับจำนวนโหนดเพื่อนบ้าน และจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด จากนั้นจึงทำการบวกค่าทั้งสอง ซึ่งค่าที่ได้นี้แทนความหนาแน่นของเครือข่าย X
- นำค่า X ที่คำนวณได้ ไปใส่ในสมการ (3) จะได้ค่า $\hat{Y}$ ซึ่งแทนช่วงเวลาในการส่งบีคอนครั้งถัดไป

2) วิธีการ K-Nearest Neighbor คือ เทคนิคการเรียนรู้ของเครื่อง (Machine Learning) และ เป็นการเรียนรู้โดยตัวอย่าง (Instance-Based Learning) สามารถนำมาประยุกต์ใช้กับการปรับช่วงเวลาในการส่งบีคอนดังนี้

- แต่ละโหนดจะมีตารางที่เก็บข้อมูลตัวอย่าง ซึ่งในแต่ละแถวของตาราง คือ คู่ของความหนาแน่นของเครือข่าย (x_i) และ ช่วงเวลาในการส่งบีคอน $(f(x_i))$
- กำหนดค่า k ที่ใช้ เพื่อระบุจำนวน Nearest Neighbor ของชุดข้อมูลตัวอย่างที่จะ นำมาใช้ในการหาค่าคำตอบที่ต้องการ

แต่ละโหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปได้ ดังนี้

- โหนดนับจำนวนโหนดเพื่อนบ้าน และจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด จากนั้นจึงทำการบวกค่าทั้งสอง ซึ่งค่าที่ได้นี้แทนความหนาแน่นของเครือข่าย (x_q)
- หา Nearest Neighbor ทั้งหมด โดยใช้สมการ (6) และค่าถ่วงน้ำหนักของแต่ละ neighbor โดยคำนวณจากสมการ (7)

$$d(x_q, x_i) = \sqrt{\sum_{r=1}^{n}(a_r(x_q) - a_r(x_i))^2} \tag{6}$$

$$w_i \equiv \frac{1}{d(x_q, x_i)^2} \tag{7}$$

- นำ Nearest Neighbor ทั้งหมดมาหาค่าเฉลี่ยตามค่าถ่วงน้ำหนัก โดยใช้สมการ (8) จะได้ $(\hat{f}(x_q))$ ซึ่งแทนช่วงเวลาในการส่งบีคอนครั้งถัดไป

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i(f(x_i))}{\sum_{i=1}^{k} w_i} \tag{8}$$

3) การปรับปรุงวิธีการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้นโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (Linear Adaptive Algorithm (LIA) with Neighbor Change Rate (NCR) (LIA+NCR)) เนื่องจากอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน สามารถบอกแนวโน้มของความหนาแน่นของเครือข่ายในอนาคตได้ ดังนี้

- ถ้ามีโหนดเพื่อนบ้านเพิ่มขึ้นเรื่อยๆ มีแนวโน้มที่โหนดกำลังเคลื่อนที่ไปสู่บริเวณที่มีความหนาแน่นของเครือข่ายมากขึ้น ดังนั้นโหนดควรจะปรับให้มีการส่งบีคอนที่ความถี่ต่ำลง ก่อนจะเข้าสู่บริเวณที่มีความหนาแน่นมากขึ้น เพื่อลดโอกาสในการชนกันของข้อมูล
- ถ้ามีโหนดเพื่อนบ้านลดลงเรื่อยๆ มีแนวโน้มที่โหนดกำลังเคลื่อนที่ไปสู่บริเวณที่มีความหนาแน่นของเครือข่ายน้อย ดังนั้นโหนดควรจะปรับให้มีการส่งบีคอนที่ความถี่สูงขึ้น ก่อนจะเข้าสู่บริเวณที่มีความหนาแน่นน้อยลง เพื่อเพิ่มโอกาสในการค้นพบโหนดเพื่อนบ้าน

การปรับปรุงวิธีการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้นโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านจะแบ่งออกเป็น 3 แบบ ดังนี้

3.1) Neighbor Change Rate (NCR) เป็นวิธีการปรับค่าบีคอนโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านเพียงอย่างเดียวเป็นตัวปรับค่าของช่วงเวลาการส่งบีคอนโดยใช้ฟังก์ชันเชิงเส้น เพื่อศึกษาแนวโน้มของกราฟเมื่อมีการพิจารณาอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านเพียงอย่างเดียวเท่านั้น ซึ่งมีวิธีการทำงานในการปรับช่วงเวลาในการส่งบีคอนดังนี้

- โหนดคำนวณอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) โดยที่เมื่อมีโหนดเพื่อนบ้านเพิ่มขึ้น โหนดจะทำการเพิ่มค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านทีละ 1 และเมื่อมีโหนดเพื่อนบ้านลดลง โหนดจะทำการการลดค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านลงทีละ 1
- โหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปได้ดังสมการ (9)

$$f(s) = \min(MinInv + (c \times NCR), MaxInv) \tag{9}$$

เมื่อ $f(s)$ คือ ช่วงเวลาในการส่งบีคอนครั้งถัดไป

$MinInv$ คือ ช่วงเวลาสั้นสุดในการส่งบีคอน

c คือ ค่าคงที่ในการเพิ่มเวลา

NCR คือ อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน

$MaxInv$ คือ ช่วงเวลายาวสุดในการส่งบีคอน

3.2) Linear Adaptive Algorithm with Neighbor Change Rate (limited) (LIA+NCR (limited)) เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าโดยใช้ฟังก์ชันเชิงเส้น ซึ่งจะมีการจำกัดช่วงเวลาในการส่งบีคอนตามแบบวิธีการของการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น (LIA) แบบเดิม คือ อยู่ในช่วง 1.5-7 วินาที ซึ่งมีวิธีการทำงานในการปรับช่วงเวลาในการส่งบีคอนดังนี้

- โหนดคำนวณอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) โดยที่เมื่อมีโหนดเพื่อนบ้านเพิ่มขึ้น โหนดจะทำการเพิ่มค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านทีละ 1 และเมื่อมีโหนดเพื่อนบ้านลดลง โหนดจะทำการการลดค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านลงทีละ 1
- โหนดคำนวณค่าความหนาแน่นของเครือข่าย จากจำนวนของโหนดเพื่อนบ้าน และจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนดดังสมการ (10)

$$s = (w_1 \times n) + (w_2 \times m) \tag{10}$$

เมื่อ s คือ ความหนาแน่นของเครือข่าย
n คือ จำนวนโหนดเพื่อนบ้าน
m คือ จำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด
w_1, w_2 คือ ค่าถ่วงน้ำหนักจำนวนโหนดเพื่อนบ้าน และจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด

เมื่อทราบค่าความหนาแน่นของเครือข่าย และอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน จะนำมาคำนวณช่วงเวลาในการส่งบีคอน ดังนี้

$$f(s) = \min(MinInv + (c \times (s + NCR)), MaxInv) \tag{11}$$

เมื่อ $f(s)$ คือ ช่วงเวลาในการส่งบีคอนครั้งถัดไป
$MinInv$ คือ ช่วงเวลาสั้นสุดในการส่งบีคอน
c คือ ค่าคงที่ในการเพิ่มเวลา
s คือ ความหนาแน่นของเครือข่าย
NCR คือ อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน
$MaxInv$ คือ ช่วงเวลายาวสุดในการส่งบีคอน

3.3) Linear Adaptive Algorithm with Neighbor Change Rate (unlimited) (LIA+NCR (unlimited)) เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าโดยใช้ฟังก์ชันเชิงเส้น ซึ่งจะมีช่วงเวลาในการส่งบีคอนอยู่ในช่วงมากกว่า หรือเท่ากับ 1.5 วินาที ซึ่งมีวิธีการทำงานในการปรับช่วงเวลาในการส่งบีคอนดังนี้

- โหนดคำนวณอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) และค่าความหนาแน่นของเครือข่าย เช่นเดียวกับวิธีการ 3.2)

เมื่อทราบค่าความหนาแน่นของเครือข่าย และอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน จะนำมาคำนวณช่วงเวลาในการส่งบีคอน ดังนี้

$$f(s) = MinInv + (c \times (s + NCR)) \quad (12)$$

เมื่อ $f(s)$ คือ ช่วงเวลาในการส่งบีคอนครั้งถัดไป

$MinInv$ คือ ช่วงเวลาสั้นสุดในการส่งบีคอน

c คือ ค่าคงที่ในการเพิ่มเวลา

s คือ ความหนาแน่นของเครือข่าย

NCR คือ อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน

3.1.3 ตัวอย่างการทำงาน

ตัวอย่างตาราง 3-1 จะได้ช่วงเวลาในการส่งบีคอนที่เหมาะสมกับความหนาแน่น และ พิจารณาความเร็วของการกระจายข้อมูลให้เร็วที่สุดภายใน 10 วินาที จากนั้นนำข้อมูลความหนาแน่นมาแปลงเป็นจำนวนโหนดเพื่อนบ้านที่อยู่ภายในระยะการเชื่อมต่อสูงสุด 250 เมตร ผลที่ได้จากตารางจะนำไปเป็นชุดข้อมูลตัวอย่างสอนของแต่ละวิธีการปรับช่วงเวลาการส่งบีคอน ซึ่งมีตัวอย่างการทำงานของแต่ละวิธีการดังต่อไปนี้

1) ตัวอย่างการปรับช่วงเวลาในการส่งบีคอนโดยใช้วิธีการวิเคราะห์การถดถอยเชิงเส้น

- นำชุดข้อมูลตัวอย่างที่ได้จากตาราง 3-1 มาตัดข้อมูลที่เลือกช่วงเวลาในการส่งบีคอนเท่ากันออกให้เหลือเพียงแค่ข้อมูลตัวแรกเพียงตัวเดียว จะได้ชุดข้อมูลใหม่ คือ {(1,1.5),(3,3),(5,7),(15,9)} แล้วจึงนำมาหาความสัมพันธ์ระหว่างความหนาแน่นของเครือข่าย (ตัวแปรอิสระ) และช่วงเวลาในการส่งบีคอน (ตัวแปรตาม) ดังนี้

<u>ตัวอย่างการคำนวณสมการ</u>

ตารางที่ 3-2 ตัวอย่างค่าที่ใช้ในการคำนวณสมการการถดถอยเชิงเส้น

x	y	$x_i - \bar{x}$	$y_i - \bar{y}$	$(x_i - \bar{x})(y_i - \bar{y})$	$(x_i - \bar{x})^2$
1	1.5	-5	-3.625	18.125	25
3	3	-3	-2.125	6.375	9
5	7	-1	1.875	-1.875	1
15	9	9	3.875	34.875	81
$\bar{x} = 6$	$\bar{y} = 5.125$			ผลรวม = 57.5	ผลรวม = 116

คำนวณค่าสัมประสิทธิ์การถดถอย b

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$b = \frac{57.5}{116} = 0.4957$$

คำนวณค่าสัมประสิทธิ์การถดถอย a

$$a = \bar{y} - b\bar{x}$$

$$a = 5.125 - 0.4957 \times 6 = 2.1509$$

จะได้สมการการถดถอยเชิงเส้น คือ

$$\hat{Y} = a + bX$$

$$\hat{Y} = 2.1509 + 0.4957X$$

จากนั้นโหนดสามารถนำสมการนี้ไปใช้ในการคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

- ยกตัวอย่างให้ ณ ตอนนี้ โหนดมีจำนวนโหนดเพื่อนบ้าน 3 โหนด และมีจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด 1 ข้อความ จะได้ค่าความหนาแน่นของเครือข่าย คือ *3+1 = 4*
- จากนั้นจึงนำค่าที่ได้นี้แทนความหนาแน่นของเครือข่าย X ในสมการการถดถอย จะได้ค่า $\hat{Y}$ ซึ่งแทนช่วงเวลาในการส่งบีคอนครั้งถัดไป ดังนี้

$$\hat{Y} = 2.1509 + 0.4957X$$

$$\hat{Y} = 2.1509 + (0.4957 \times 4) = 4.1337$$

ดังนั้นช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนด คือ 4.1337 วินาที

2) ตัวอย่างการปรับช่วงเวลาในการส่งบีคอนโดยใช้วิธีการ k-Nearest Neighbor

- แต่ละโหนดจะมีตารางที่เก็บชุดข้อมูลตัวอย่าง ซึ่งในแต่ละแถวของตาราง คือ คู่ของความหนาแน่นของเครือข่าย (x_i) และ ช่วงเวลาในการส่งบีคอน $(f(x_i))$ ดังในตารางที่ 3-3
- กำหนดค่า k ที่ใช้ เพื่อระบุจำนวน nearest neighbor ของชุดข้อมูลตัวอย่างที่จะนำมาใช้ในการหาค่าคำตอบที่ต้องการ ซึ่งในกรณีตัวอย่าง กำหนดให้ค่า k เท่ากับ 2

จากนั้นโหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

- ยกตัวอย่างให้ ณ ตอนนี้ โหนดมีจำนวนโหนดเพื่อนบ้าน 3 โหนด และมีจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด 1 ข้อความ จะได้ค่าความหนาแน่นของเครือข่าย คือ *3+1 = 4*
- จากนั้นจึงนำค่าที่ได้นี้แทนความหนาแน่นของเครือข่าย (x_q) แล้วจึงหา nearest neighbor โดยคำนวณระยะห่างระหว่าง x_q และ x_i ดังนี้

$$d(x_q, x_i) = \sqrt{\sum_{r=1}^{n} (a_r(x_q) - a_r(x_i))^2}$$

ตารางที่ 3-3 ชุดข้อมูลตัวอย่างของวิธีการ K-Nearest Neighbor

ความหนาแน่นของเครือข่าย(x_i)		ช่วงเวลาในการส่งบีคอน (วินาที) $f(x_i)$
(x_1)	1	1.5
(x_2)	3	3
(x_3)	5	7
(x_4)	10	7
(x_5)	15	9
(x_6)	20	9
(x_7)	30	9
(x_8)	40	9

$$d(x_q, x_1) = \sqrt{(4-1)^2} = 3 \qquad d(x_q, x_5) = \sqrt{(4-15)^2} = 11$$
$$d(x_q, x_2) = \sqrt{(4-3)^2} = 1 \qquad d(x_q, x_6) = \sqrt{(4-20)^2} = 16$$
$$d(x_q, x_3) = \sqrt{(4-5)^2} = 1 \qquad d(x_q, x_7) = \sqrt{(4-30)^2} = 26$$
$$d(x_q, x_4) = \sqrt{(4-10)^2} = 6 \qquad d(x_q, x_8) = \sqrt{(4-40)^2} = 36$$

- จากการคำนวณระยะห่างระหว่าง x_q และ x_i จะได้ Nearest Neighbor 2 ตัว ที่มีระยะห่างน้อยที่สุด คือ ชุดข้อมูลตัวอย่างคู่ที่ 2 และ 3
- จากนั้นจึงทำการคำนวณค่าถ่วงน้ำหนักของแต่ละ Nearest Neighbor ดังนี้

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

$$w_2 \equiv \frac{1}{d(x_q, x_2)^2} = \frac{1}{1^2} = 1, \quad w_3 \equiv \frac{1}{d(x_q, x_3)^2} = \frac{1}{1^2} = 1$$

- นำ Nearest Neighbor ทั้งหมดมาหาคำตอบโดยเฉลี่ยตามค่าถ่วงน้ำหนัก จะได้ $(\hat{f}(x_q))$ ซึ่งแทนช่วงเวลาในการส่งบีคอนครั้งถัดไป ดังสมการ

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i (f(x_i))}{\sum_{i=1}^{k} w_i}$$

$$\hat{f}(x_q) = \frac{(1 \times 3) + (1 \times 7)}{1 + 1} = 5$$

ดังนั้นช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนด คือ 5 วินาที

3) ตัวอย่างการปรับช่วงเวลาในการส่งบีคอนโดยใช้วิธีการปรับปรุงวิธีการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้นโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน แบ่งออกเป็น 3 แบบดังนี้

3.1) Neighbor Change Rate (NCR) เป็นวิธีการปรับค่าบีคอนโดยใช้ข้อมูลอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านเพียงอย่างเดียวเป็นตัวปรับค่าของช่วงเวลาการส่งบีคอนโดยใช้ฟังก์ชันเชิงเส้น

- ยกตัวอย่างให้ก่อนหน้านี้ โหนดมีจำนวนโหนดเพื่อนบ้าน 3 โหนด และมีค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) เท่ากับ 2 จากนั้นเมื่อมีการเปลี่ยนแปลงของโหนดเพื่อนบ้าน เช่น มีโหนดเพื่อนบ้านเพิ่มเข้ามา 2 โหนด โหนดจะทำการเพิ่มค่า (NCR) โดยการบวกหนึ่ง ดังนั้น ค่า NCR จะเท่ากับ 3

จากนั้นโหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

$$f(s) = \min(MinInv + (c \times NCR), MaxInv)$$

เมื่อ กำหนดให้ $MinInv$ เท่ากับ 1.5, c เท่ากับ 0.2 และ $MaxInv$ เท่ากับ 7 จะได้

$$f(s) = \min(1.5 + (0.2 \times 3), 7) = 2.1$$

ดังนั้นช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนด คือ 2.1 วินาที

3.2) Linear Adaptive Algorithm with Neighbor Change Rate (limited) (LIA+NCR (limited)) เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าโดยใช้ฟังก์ชันเชิงเส้น ซึ่งจะมีการจำกัดช่วงเวลาในการส่งบีคอน

- ยกตัวอย่างให้ก่อนหน้านี้ โหนดมีจำนวนโหนดเพื่อนบ้าน 25 โหนด และมีจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด 5 ข้อความรวมทั้งมีค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) เท่ากับ 5 จากนั้นเมื่อมีการเปลี่ยนแปลงของโหนดเพื่อนบ้าน เช่น มีโหนดเพื่อนบ้านลดลง 3 โหนด โหนดจะทำการลดค่า (NCR) โดยการลบหนึ่ง ดังนั้น ค่า NCR จะเท่ากับ 4
- จากนั้นโหนดจะคำนวณค่าความหนาแน่นของเครือข่าย คือ *22+5 = 27*

โหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

$$f(s) = \min(MinInv + (c \times (s + NCR)), MaxInv)$$

เมื่อ กำหนดให้ $MinInv$ เท่ากับ 1.5, c เท่ากับ 0.2 และ $MaxInv$ เท่ากับ 7 จะได้

$$f(s) = \min(1.5 + (0.2 \times (27 + 4)), 7) = 7$$

ดังนั้นช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนด คือ 7 วินาที

3.3) Linear Adaptive Algorithm with Neighbor Change Rate (unlimited) (LIA+NCR (unlimited)) เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าโดยใช้ฟังก์ชันเชิงเส้น ซึ่งจะไม่จำกัดช่วงเวลานานที่สุดในการส่งบีคอน

- ยกตัวอย่างให้ก่อนหน้านี้ โหนดมีจำนวนโหนดเพื่อนบ้าน 25 โหนด และมีจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด 5 ข้อความรวมทั้งมีค่าอัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน (NCR) เท่ากับ 5 จากนั้นเมื่อมีการเปลี่ยนแปลงของโหนดเพื่อนบ้าน เช่น มีโหนดเพื่อนบ้านลดลง 3 โหนด โหนดจะทำการลดค่า (NCR) โดยการลบหนึ่ง ดังนั้น ค่า NCR จะเท่ากับ 4
- จากนั้นโหนดจะคำนวณค่าความหนาแน่นของเครือข่าย คือ *22+5 = 27*

โหนดสามารถคำนวณช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนดได้ ดังนี้

$$f(s) = MinInv + \left(c \times (s + NCR)\right)$$

เมื่อ กำหนดให้ $MinInv$ เท่ากับ 1.5, c เท่ากับ 0.2

$$f(s) = 1.5 + \left(0.2 \times (27 + 4)\right) = 7.7$$

ดังนั้นช่วงเวลาในการส่งบีคอนครั้งถัดไปของโหนด คือ 7.7 วินาที

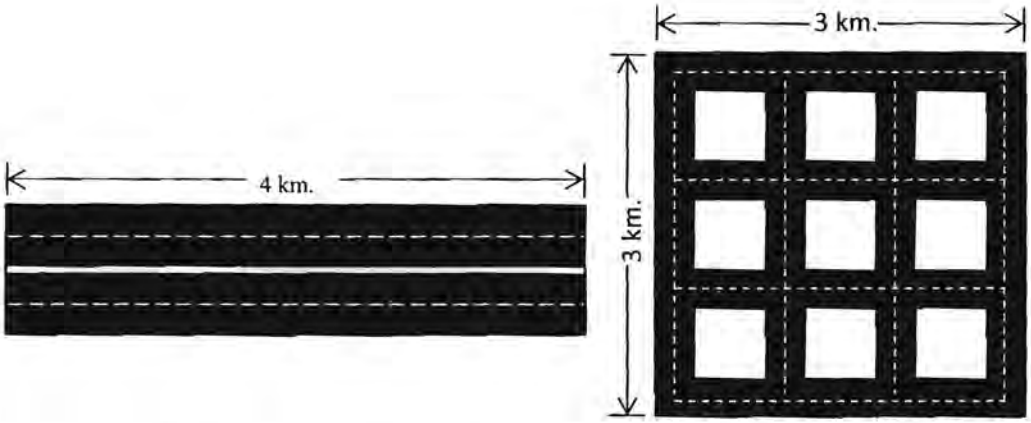
3.1.4 การทดลองและวิเคราะห์ผล

1) ตัววัดสมรรถนะ

- *จำนวนครั้งในการส่งบีคอน (Number of beacon)* เนื่องจากในการทำงานบนเครือข่ายไร้สายแบบแอดฮอกนั้น จำนวนครั้งในการส่งบีคอนเป็นสิ่งสำคัญที่จะแสดงให้เห็นถึงปริมาณการใช้ช่องสัญญาณ ซึ่งคำนวณจากจำนวนครั้งในการส่งบีคอนทั้งหมดที่เกิดขึ้นในระบบ และเป็นจุดประสงค์หลักในการออกแบบการส่งบีคอนแบบปรับค่าได้
- *ความเร็วของการกระจายข้อมูล (Speed of Data Dissemination)* เนื่องจากความเร็วในการกระจายข้อมูลเป็นจุดประสงค์หลักในการออกแบบโพรโทคอล DECA ซึ่งเป็นตัวแปรในการพิจารณาวิธีการปรับบีคอนที่นำมาศึกษามาใช้งาน โดยที่ความเร็วในการกระจายข้อมูลจะคำนวณจากจำนวนของโหนดที่ได้รับข้อมูลต่อจำนวนโหนดทั้งหมด ณ เวลานั้น

2) เครื่องมือในการทดลอง ใช้เครื่องมือเช่นเดียวกับการทดลองในบทที่ 2 คือ เริ่มจากการจำลองพฤติกรรมการเคลื่อนที่ของรถยนต์โดยใช้โปรแกรม SUMO ซึ่งจะได้พฤติกรรมการเคลื่อนที่ของรถยนต์ออกมาในรูปแบบของ xml จากนั้นจะใช้โปรแกรม TraNS เพื่อแปลง XML ให้อยู่ในรูปแบบของ tcl ที่สามารถนำไปใช้งานบนโปรแกรมจำลอง NS-2.34 ได้ และใช้โปรแกรมจำลองเครือข่าย NS-2.34 ในการทดลอง

3) สภาพแวดล้อมที่ใช้ในการทดลอง การทดลองสามารถแบ่งลักษณะของถนนที่ใช้ในการทดสอบคือ ถนนทางหลวง และถนนในเมือง โดยที่ถนนทางหลวงจะเป็นลักษณะของทางตรงความยาว 4 กิโลเมตร มี 4 ช่องทางการจราจร และถนนในเมือง จะเป็นลักษณะของตารางขนาด 3x3 ตารางกิโลเมตร ประกอบด้วยสี่แยกจำนวน 4 แยก และสามแยกจำนวน 8 แยก โดยที่ลักษณะของถนนที่ใช้เป็นไปดังภาพที่ 3-2



ก) รูปถนนทางหลวง ข) รูปถนนในเมือง

รูปที่ 3-2 รูปแบบถนนที่ใช้ในการทดลองในโปรแกรมจำลอง

ในการทดลองแต่ละครั้งข้อมูลที่ใช้ในการกระจายมีขนาด 512 ไบต์ โดยข้อมูลมีการกำหนดอายุดังนี้ บนถนนทางหลวงจะกำหนดให้ข้อมูลมีอายุ 10 วินาที ส่วนถนนในเมืองจะกำหนดให้ข้อมูลมีอายุ 50 วินาที ซึ่งถ้าข้อมูลหมดอายุการทดลองในครั้งนั้นก็จะสิ้นสุดลงและจะเก็บผลการทดลองทันที ในการทดลองจะใช้ระบบสื่อสารไร้สายบนมาตรฐาน IEEE802.11 ซึ่งมีระยะสื่อสารสูงสุดที่ 250 เมตร โดยการตั้งค่าโพรโทคอล และค่าต่างๆที่ใช้ในการทดลองตามตารางที่ 3-4

วิธีการที่นำมาใช้เพื่อทดสอบการปรับช่วงเวลาการส่งบีคอนมีดังนี้

- **Linear Adaptive Algorithm (LIA)** : การคำนวณช่วงเวลาปรับตัวแบบเชิงเส้นเป็นวิธีในการปรับค่าบีคอนให้เปลี่ยนแปลงไปตามความหนาแน่นของเครือข่าย โดยพิจารณาจากจำนวนโหนดเพื่อนบ้าน และจำนวนของข้อความที่อยู่ในบัฟเฟอร์ของโหนด

- **Linear Regression Analysis** : การวิเคราะห์การถดถอยเชิงเส้นเป็นวิธีการที่หาความสัมพันธ์ระหว่างตัวแปรตาม (Dependent Variable) ซึ่งเป็นตัวแปรที่ต้องการทราบค่า กับตัวแปรอิสระ (Independent Variable) ซึ่งเป็นตัวแปรที่ทราบค่าแล้วโดยที่ตัวแปรอิสระนั้นจะส่งผลต่อตัวแปรตาม ในการทดลองการปรับค่าบีคอนโดยใช้วิธีการวิเคราะห์การถดถอยเชิงเส้นนั้น ให้ตัวแปรตามคือ ช่วงเวลาที่ใช้ในการส่งบีคอน ส่วนตัวแปรอิสระ คือ ความหนาแน่นของเครือข่าย

- **K-Nearest Neighbor (K-NN)** : เป็นเทคนิคการเรียนรู้ของเครื่อง ซึ่งจะมีการเก็บชุดข้อมูลตัวอย่างสอน ในที่นี้คือข้อมูลการทดลองจากโปรแกรมจำลองของการส่งบีคอนในแต่ละช่วงเวลา แล้วนำข้อมูลเหล่านั้นมาใช้ในการหาค่าส่งออก โดยกำหนดจำนวน k แทนจำนวนตัวอย่างที่มีค่าใกล้เคียงกับค่านำเข้า แล้วคำนวณค่าส่งออกโดยถ่วงน้ำหนักของค่าที่ได้จากตัวอย่างการสอน k ตัวนั้น

- **Neighbor Change Rate (NCR)** : เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้านเป็นตัวปรับค่าช่วงเวลาของการส่งบีคอนโดยใช้ฟังก์ชันเชิงเส้น

- **Linear Adaptive Algorithm with Neighbor Change Rate (limited) (LIA+NCR (limited))** : เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าแบบเชิงเส้น ซึ่งจะมีการจำกัดช่วงเวลาในการส่งบีคอน

- **Linear Adaptive Algorithm with Neighbor Change Rate (unlimited) (LIA+NCR (unlimited))** : เป็นวิธีการปรับค่าบีคอนโดยใช้อัตราการเปลี่ยนแปลงของโหนดเพื่อนบ้าน และความหนาแน่นของเครือข่ายในการปรับค่าโดยใช้ฟังก์ชันเชิงเส้น ซึ่งจะไม่จำกัดช่วงเวลานานที่สุดในการส่งบีคอน

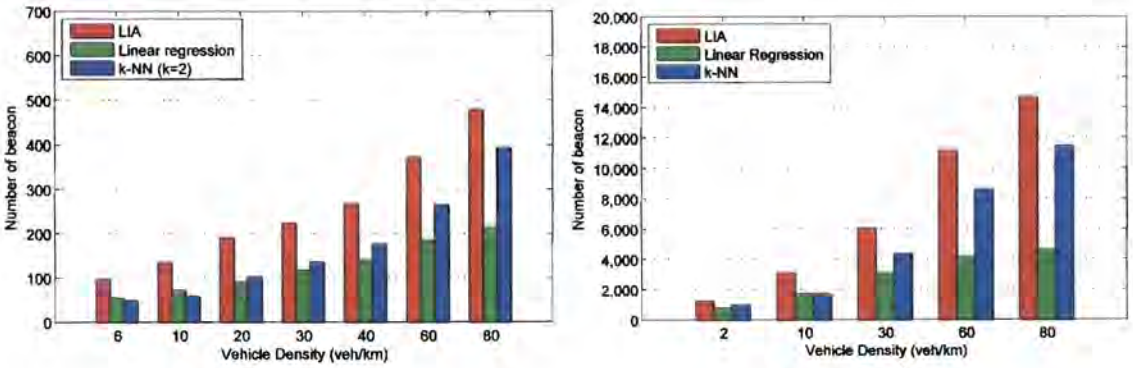
ตารางที่ 3-4 การตั้งค่าต่างๆที่ใช้ในการทดลอง

การทดลอง	จำนวนครั้งในการทดลอง : 10 ครั้ง ผลการทดลองใช้ค่าเฉลี่ย
สภาพจราจร	ถนนทางหลวง : 6, 10, 20, 30, 40, 60, 80 คัน/กิโลเมตร ความเร็วสูงสุด : 50, 80 กิโลเมตร/ชั่วโมง ถนนในเมือง : 2, 10, 30, 60, 80 คัน/กิโลเมตร ความเร็วสูงสุด : 120 กิโลเมตร/ชั่วโมง
ข้อความ	อายุข้อความ : ถนนทางหลวง 10 วินาที ถนนในเมือง 50 วินาที ขนาดของข้อความ : 512 ไบต์
Linear Adaptive Algorithm (LIA)	ช่วงเวลาในการส่งบีคอน : ทุก 1.5 – 7 วินาที $c = 0.2$, $MinInv = 1.5$, $MaxInv = 7$
Linear Regression Analysis	ค่าสัมประสิทธิ์การถดถอย : $a = 2.1509$, $b = 0.4957$
K-Nearest Neighbor (k-NN)	จำนวน Nearest Neighbor (k) = 2
Neighbor Change Rate (NCR)	ช่วงเวลาในการส่งบีคอน : ทุก 1.5 – 7 วินาที $c = 0.2$, $MinInv = 1.5$, $MaxInv = 7$
Linear Adaptive Algorithm with Neighbor Change Rate (limited) (LIA+NCR (limited))	ช่วงเวลาในการส่งบีคอน : ทุก 1.5 – 7 วินาที $c = 0.2$, $MinInv = 1.5$, $MaxInv = 7$
Linear Adaptive Algorithm with Neighbor Change Rate (unlimited) (LIA+NCR (unlimited))	ช่วงเวลาในการส่งบีคอน : >=1.5 วินาที $c = 0.2$, $MinInv = 1.5$
ความต้องการของแอพพลิเคชั่นทางด้านความเร็วของการกระจายข้อมูล	ถนนทางหลวง : ค่าความเชื่อถือได้สูงสุดภายใน 10 วินาที ถนนในเมือง : ค่าความเชื่อถือได้สูงสุดภายใน 15 วินาที

4) ผลการทดลอง

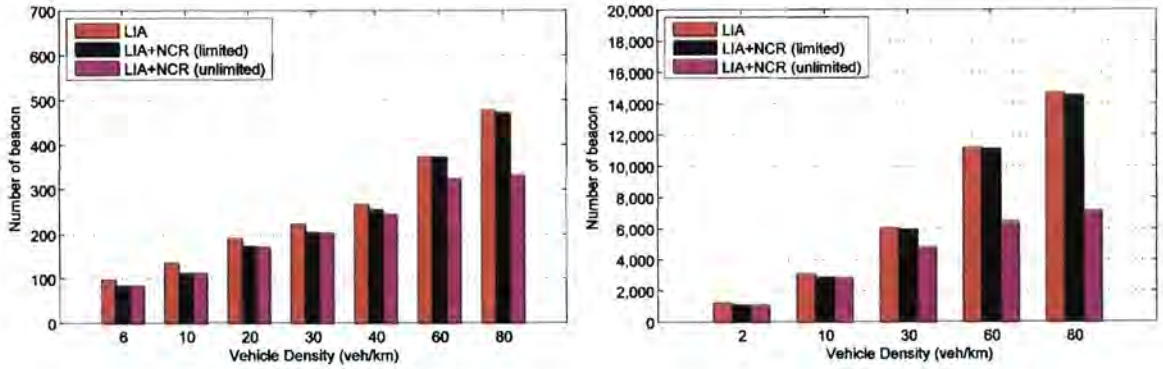
- **ผลการทดลองจำนวนครั้งในการส่งบีคอน** จากรูปที่ 3-3 และ 3-4 สามารถสังเกตได้ว่าจำนวนครั้งในการส่งบีคอนที่เกิดขึ้นสามารถเรียงตามลำดับจากจำนวนมากที่สุดไปน้อยที่สุดได้ดังนี้ NCR, LIA, LIA+NCR(limited), k-NN(2), LIA+NCR(unlimited) และ Linear Regression โดยผลการทดลองทั้งในถนนทางหลวงและถนนในเมืองให้ผลเช่นเดียวกัน ผลการทดลองของ NCR มีค่าใช่จ่ายที่สูงมากจึงไม่นำมาแสดงในผลการทดลอง ซึ่งผลการทดลองที่เกิดขึ้นมีผลมาจากค่าการคำนวณช่วงเวลาในการทำบีคอนต่อค่าความหนาแน่น ถ้าวิธีการใดมีความไวต่อข้อมูลความหนาแน่นสูงกว่าจะส่งผลให้เกิดการเพิ่มช่วงเวลาที่สูงกว่า และสามารถลดจำนวนบีคอนได้มากกว่า แต่ประสิทธิภาพในการทำงานขึ้นอยู่กับผลความเร็วในการกระจายข้อมูลต่อไป
- **ผลการทดลองความเร็วในการกระจายข้อมูล** พิจารณารูปที่ 3-5, 3-6, 3-7 และ 3-8 ผลการทดลองทั้งถนนทางหลวง และถนนในเมืองให้ผลที่เหมือนกัน คือ ที่ความหนาแน่นต่ำ NCR และ LIA ให้ความเร็ว แต่เนื่องจาก NCR มีค่าใช่จ่ายที่สูงมาก LIA จึงถือว่ามีประสิทธิภาพดีที่สุดในการทำงานที่ความหนาแน่นต่ำ ส่วนที่มีความหนาแน่นสูง Linear Regression มีความเร็วในการกระจายข้อมูลเร็วที่สุดและยังให้ค่าใช้จ่ายในการส่งบีคอนต่ำที่สุดอีกด้วย

จากผลการทดลองสามารถสรุปได้ว่าช่วงเวลาในการทำบีคอนมีผลต่อสมรรถนะและความในเร็วการกระจายข้อความของโพรโทคอล ดังนั้นการเลือกช่วงเวลาในการทำบีคอนจึงต้องมีการเลือกช่วงเวลาให้สอดคล้องกับแอพพลิเคชั่นที่ใช้งาน โดยในการทดลองนี้ Linear Regression สามารถให้ความเร็วในการกระจายต่อค่าใช้จ่ายในการทำบีคอนสูงที่สุด



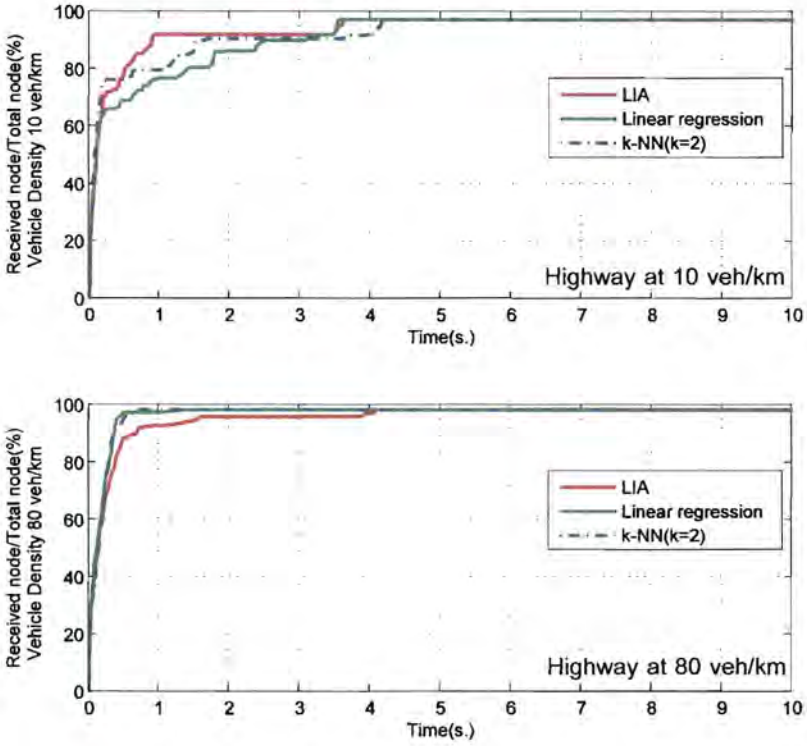
ก) ถนนทางหลวง ข) ถนนในเมือง

รูปที่ 3-3 ผลการทดลองจำนวนครั้งในการส่งบีคอนโดยใช้วิธี Linear regression และ K-Nearest Neighbor

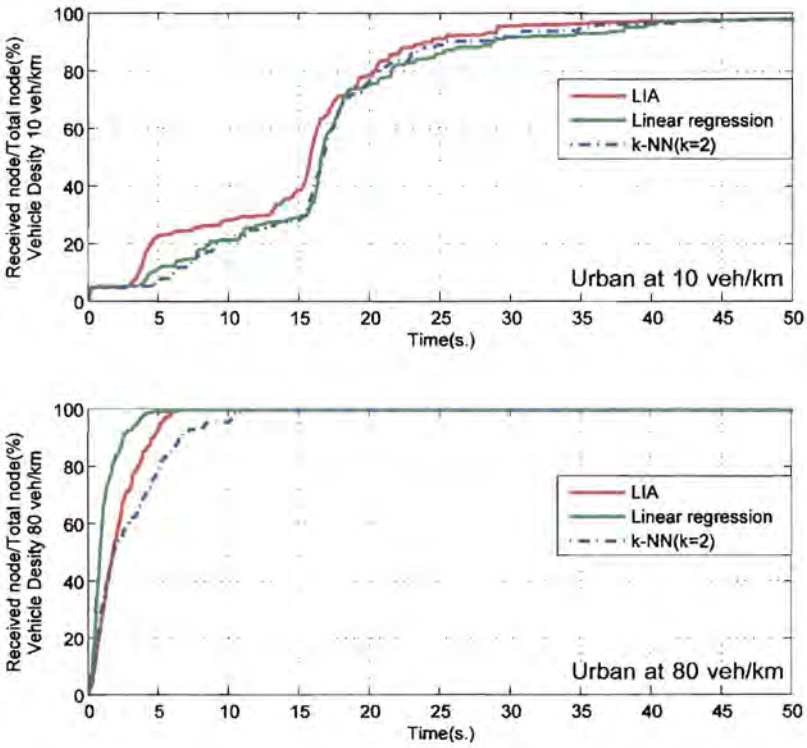


ก) ถนนทางหลวง ข) ถนนในเมือง

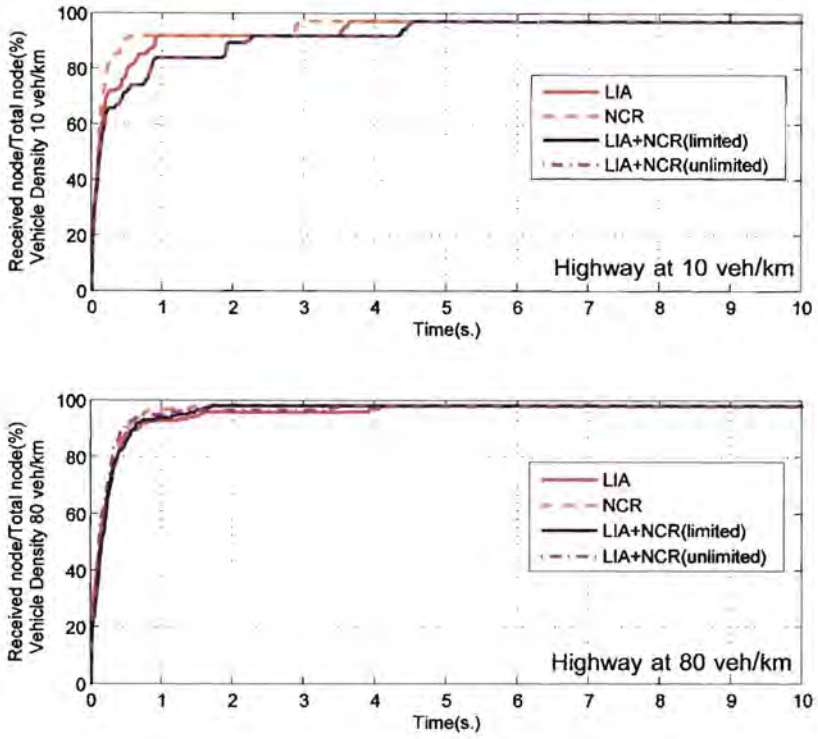
รูปที่ 3-4 ผลการทดลองจำนวนครั้งในการส่งบีคอนโดยใช้วิธี LIA+NCR



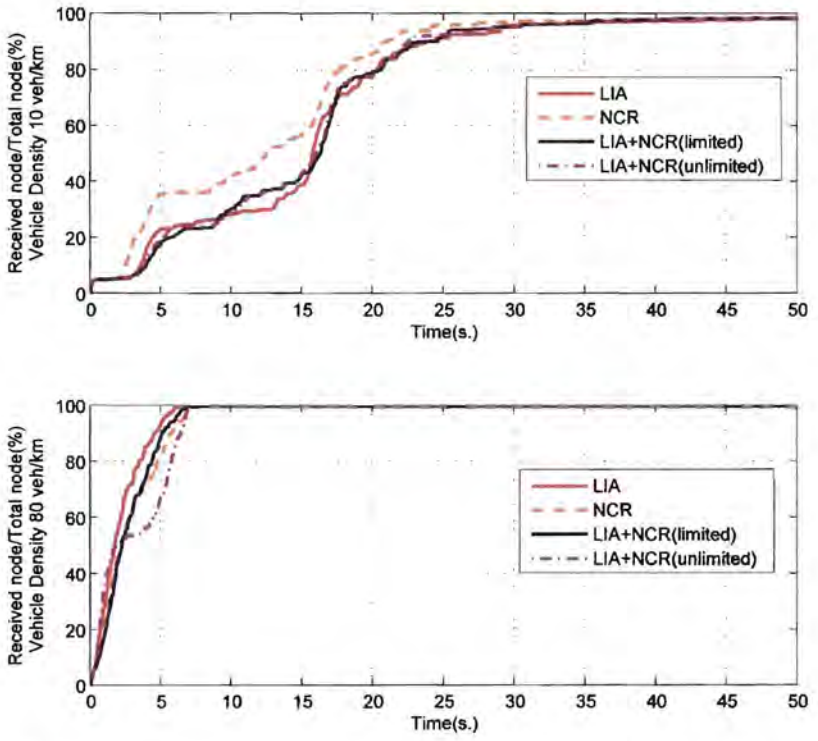
รูปที่ 3-5 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนทางหลวง
โดยใช้วิธี Linear Regression และ K-Nearest Neighbor



รูปที่ 3-6 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนในเมือง
โดยใช้วิธี Linear regression และ K-Nearest Neighbor



รูปที่ 3-7 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนทางหลวงโดยใช้วิธี LIA+NCR



รูปที่ 3-8 ผลการทดลองความเร็วของการกระจายข้อมูลบนถนนในเมืองโดยใช้วิธี LIA+NCR

3.2 การจัดการข้อมูลในบัฟเฟอร์

3.2.1 แนวคิดในการออกแบบนโยบายการทิ้งกลุ่มข้อมูล (Dropping Policy)

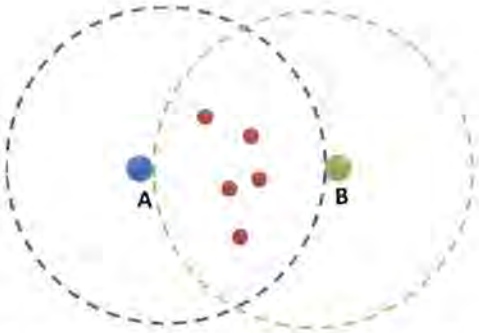
เนื่องจากโพรโทคอลการกระจายที่มีความเชื่อถือแบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะนั้นเป็นโพรโทคอลที่ทำงานในรูปแบบ Store-and-Forward เพื่อรองรับกับการเชื่อมต่อเป็นช่วงๆ (Intermittent Connectvity)ในสภาพการจราจรที่มีความหนาแน่นน้อย ซึ่งการทำงานแบบ Store-and-Forward โหนดจำเป็นจะต้องเก็บกลุ่มข้อมูล (Packet) จำนวนมากที่ยังไม่หมดอายุไว้ในบัฟเฟอร์ของตนเพื่อนที่จะเพิ่มความเชื่อถือได้ของข้อมูลนั้นสำหรับโหนดอื่นๆที่ยังไม่ได้รับข้อความ ดังนั้นหากมีนโยบายการทิ้งข้อมูลที่มีประวิทธิภาพสูงจะส่งให้โพรโทคอลสามารถทำงานได้โดยที่มีความต้องการบัฟเฟอร์ขนาดที่ต่ำกว่าการใช้การทิ้งข้อมูลแบบดั้งเดิม เช่น การทิ้งข้อมูลส่วนหน้าหรือหลัง (Drop Front/Tail) หรือการทิ้งข้อมูลแบบสุ่ม และจากการศึกษางานวิจัยนโยบายการทิ้งข้อมูล [17] [18] ที่พบเกี่ยวข้องกับการทำงานบนเครือข่ายที่มีความคงทนต่อเวลาหน่วง (Delay Tolerant Networks : DTN) ซึ่งไม่มีการออกแบบที่เหมาะสมกับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ

ดังนั้นโยบายการทิ้งข้อมูลสำหรับ DECA ควรใช้เพียงข้อมูลความหนาแน่นที่ได้รับจากการจากแลกเปลี่ยนบีคอนเพียงเท่านั้น โดยโหนดจะสามารถประมาณค่าจำนวนสำเนาของข้อความที่ถูกแพร่ออกไปจากโหนดได้จากจำนวนเพื่อนบ้านในบริเวณ 1 hop และสามารถที่จะคำนวณโหนดเพื่อนบ้านที่อยู่ในบริเวณที่มีการทับซ้อนของสัญญาณไร้สายซึ่งจะทำให้การประมาณค่าสำเนาของข้อความนั้นมีความเที่ยงตรงมากขึ้น จากนั้นการพิจารณาการทิ้งข้อมูลเมื่อบัฟเฟอร์เต็มก็พิจารณาจากข้อความที่มีค่าสำเนาสูงที่สุด เนื่องจากมีจำนวนเพื่อนบ้านที่ได้รับข้อความนี้สูงที่สุด และมีโอกาสที่จะส่งไปถึงโหนดส่วนใหญ่ในระบบแล้วมากกว่าข้อความที่มีค่าสำเนาน้อยกว่า เพื่อให้โพรโทคอลสามารถแพร่ข้อความนั้นแก่โหนดอื่นๆในระบบได้ต่อไป

3.2.2 หลักการทำงานของนโยบายการทิ้งกลุ่มข้อมูล

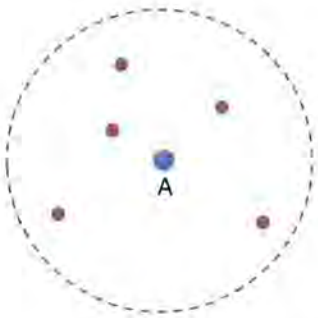
3.2.2.1 การประมาณจำนวนโหนดในพื้นที่ทับซ้อน

เนื่องจากเมื่อมีการกระจายข้อมูลระหว่างโหนดสองโหนด โพรโทคอลมักเลือกโหนดบริเวณขอบของระยะการสื่อสารเป็นโหนดถัดไปที่ทำหน้าที่ในการกระจายกลุ่มข้อมูลต่อเพื่อลดปริมาณค่าใช้จ่ายในการกระจายข้อมูล ดังนั้นในการกระจายข้อมูลจึงมีลักษณะ ดังรูปที่ 3-9 ซึ่งแสดงตัวอย่างเมื่อ A เป็นโหนดต้นทางและเลือก B เพื่อเป็นโหนดทำการกระจายต่อจะสังเกตได้ว่าโหนดสีแดงบริเวณพื้นที่ทับซ้อนของการสื่อสารนั้นได้รับการกระจายข้อมูลถึงสองครั้ง โหนดในบริเวณนั้นจึงไม่มีส่วนในการเพิ่มจำนวนสำเนาของกลุ่มข้อมูลในการกระจายข้อมูลของโหนด B ดังนั้นในการคำนวณหาจำนวนโหนดในพื้นที่ทับซ้อนในกรณีที่โหนดทราบแต่ความหนาแน่น สามารถทำได้โดยการประมาณความหนาแน่นในบริเวณพื้นที่ทับซ้อนของการสื่อสาร ซึ่งคำนวณจากความหนาแน่นของโหนดในพื้นที่ทับซ้อน และขนาดของพื้นที่ทับซ้อน



รูปที่ 3-9 โหนดบริเวณของพื้นที่ทับซ้อนของการสื่อสาร

1) การประมาณความหนาแน่นของโหนดเพื่อนบ้านสามารถทำได้โดยการประมาณจากจำนวนของโหนดเพื่อนบ้านทั้งหมดของโหนดใดๆ คือ ความหนาแน่นของโหนดเพื่อนบ้านต่อหนึ่งพื้นที่ของระยะการสื่อสารของโหนดนั้น ดังรูปที่ 3-10 จะสังเกตว่าโหนด A มีเพื่อนบ้านหนาแน่น 5 โหนดต่อหนึ่งพื้นที่ของระยะการสื่อสาร



รูปที่ 3-10 จำนวนโหนดเพื่อนบ้านของโหนดต้นทาง

2) การคำนวณหาอัตราส่วนพื้นที่ทับซ้อนกับพื้นที่ของระยะการสื่อสาร เมื่อกำหนดให้ระยะห่างระหว่างโหนดคือ R และให้รัศมีในการสื่อสารของทุกโหนดเท่ากันที่ R จะสามารถหาพื้นที่ทับซ้อนของการสื่อสารได้ดังสมการ (13) ดังนั้นอัตราส่วนของพื้นที่ทับซ้อนในกรณีทั่วไปที่ระยะห่างระหว่างโหนด (d) มีค่าเท่ากับระยะของการสื่อสาร (R) กับพื้นที่ของระยะการสื่อสารคือ 0.39 ดังสมการ (14)

พื้นที่ทับซ้อนของวงกลมกรณีที่ $d = R$

$$Overlap\ Area = R^2 \cos^{-1}\left(\frac{1}{2}\right) - \frac{R^2}{2}\sqrt{3} \tag{13}$$

หาอัตราส่วนของพื้นที่ทับซ้อนกันของวงกลมต่อพื้นที่วงกลมได้จาก

$$Overlap\ Ratio = \frac{R^2 \cos^{-1}\left(\frac{1}{2}\right) - \frac{R^2}{2}\sqrt{3}}{\pi R^2} = 0.39 \tag{14}$$

3.2.2.2 การคำนวณค่าสำเนา

หลักการคำนวณและปรับปรุงค่าสำเนาของกลุ่มข้อมูลสามารถแบ่งออกเป็นกรณีต่างๆ ได้ตามขั้นตอนการทำงานของโพรโทคอล DECA ที่มีการทำงานแบบ Store-and-Forward ได้ 4 กรณี ดังต่อไปนี้

1) กรณีที่โหนดต้นทาง (Source Node) แพร่ข้อมูลให้กับเพื่อนบ้านตนเอง ในกรณีนี้โหนดต้นทางเป็นโหนดเริ่มต้นในการกระจายกลุ่มข้อมูล ดังนั้นจำนวนสำเนาที่เกิดขึ้นจึงเกิดจากการกระจายของโหนดต้นทางเพียงโหนดเดียวเท่านั้น ดังนั้นค่าสำเนาในกรณีนี้จึงมีค่าเท่ากับจำนวนโหนดเพื่อนบ้านทั้งหมดของโหนดต้นทาง ซึ่งเป็นจำนวนโหนดที่อยู่ในระยะการส่งของโหนดต้นทาง ดังสมการ (17) ให้ $copy$ คือ ค่าสำเนาของกลุ่มข้อมูล และ n_s เป็นจำนวนโหนดเพื่อนบ้านของโหนดต้นทาง

$$copy = n_s \tag{15}$$

2) *กรณีโหนดที่ถูกเลือก (Next Rebroadcast Node) แพร่กลุ่มข้อมูลให้กับเพื่อนบ้านของตนเอง* ในกรณีนี้แสดงว่าก่อนหน้านี้มีการกระจายกลุ่มข้อมูลนี้มาแล้ว และโหนดกำลังจะแพร่กลุ่มข้อมูลนี้ต่อไป โดยมีค่าสำเนาเดิมแนบมากับกลุ่มข้อมูลนั้น ดังนั้นการกระจายในครั้งนี้จะเกิดพื้นที่ซ้อนทับของการสื่อสารขึ้นระหว่างโหนดก่อนหน้า (Precursor Node) และโหนดที่จะแพร่กลุ่มข้อมูลต่อ (Next Rebroadcast Node) จึงสามารถคำนวณค่าสำเนาได้จากการรวมค่าสำเนาเดิมและค่าสำเนาใหม่ และหักค่าสำเนาที่คาดว่าจะซ้ำซ้อนในพื้นที่ทับซ้อน ดังสมการ (16) โดยให้ $copy$ เป็นค่าสำเนาใหม่ $copy'$ เป็นค่าสำเนาเดิม n_p เป็นจำนวนเพื่อนบ้านของโหนดก่อนหน้า n_r เป็นจำนวนเพื่อนบ้านของโหนดที่ถูกเลือก และ α เป็นอัตราส่วนของพื้นที่ทับซ้อนกับพื้นที่ระยะสื่อสาร (มีค่าเท่ากับ 0.39)

$$copy = (copy' + n_r) - (n_p \times \alpha) \tag{16}$$

3) กรณีโหนดที่ถูกเลือกไม่ทำงาน และเพื่อนบ้านทำการกระจายข้อมูลแทน ซึ่งในกรณีนี้จะมีวิธีการคำนวณเช่นเดียวกับกรณีที่ 2

4) กรณีที่โหนดเพื่อนบ้านบางโหนดยังไม่ได้รับข้อมูล ในกรณีนี้จำนวนของสำเนาที่เพิ่มขึ้นเกิดขึ้นจากจำนวนของโหนดเพื่อนบ้านที่ยังไม่ได้รับข้อมูล นั่นคือจำนวนของโหนดที่แลกเปลี่ยนข้อมูลบีคอนแล้วพบว่ายังไม่ได้รับกลุ่มข้อมูลนั้น ดังนั้นโหนดจะทำการนับจำนวนของบีคอนที่ยังไม่ได้รับกลุ่มข้อมูลนั้นภายในเวลาก่อนจะมีการกระจายกลุ่มข้อมูล สามารถคำนวณค่าสำเนาได้ดังสมการ (17) ให้ $copy$ เป็นค่าสำเนาใหม่ $copy'$ เป็นค่าสำเนาเดิม และ n_b เป็นจำนวนบีคอนจากโหนดที่ไม่ได้รับกลุ่มข้อมูลนั้นภายในเวลาก่อนจะมีการกระจายกลุ่มข้อมูล

$$copy = copy' + n_b \tag{17}$$

3.2.2.3 ขั้นตอนการทำงาน

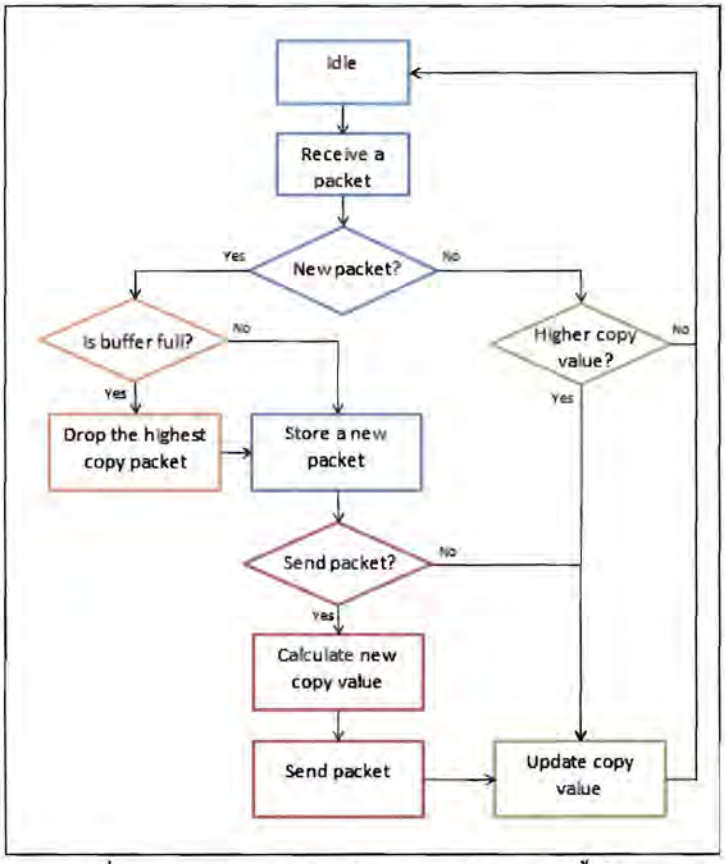
ขั้นตอนการทำงานสามารถแบ่งออกเป็นวิธีการทำงานที่สำคัญได้ดังต่อไปนี้

1) เมื่อโหนดได้รับกลุ่มข้อมูล โหนดจะทำการตรวจสอบว่าเป็นกลุ่มข้อมูลใหม่หรือไม่ หากไม่เป็นกลุ่มข้อมูลใหม่จะทำการตรวจสอบค่าสำเนา หากค่าสำเนาใหม่มีค่าสูงกว่าจะมีการปรับปรุงแทนค่าสำเนาเดิม

2) กรณีที่โหนดได้รับกลุ่มข้อมูลใหม่ โหนดจะตรวจสอบว่าบัฟเฟอร์มีขนาดพอสำหรับการเก็บข้อมูลหรือไม่ กรณีที่บัฟเฟอร์เต็มโหนดจะทิ้งกลุ่มข้อมูลเก่าในบัฟเฟอร์ที่มีค่าสำเนาสูงที่สุด จากนั้นจึงเก็บกลุ่มข้อมูลใหม่พร้อมเก็บค่าสำเนา

3) กรณีที่โหนดจะต้องแพร่กลุ่มข้อมูล ซึ่งเกิดขึ้นได้ 4 กรณีคือ การเริ่มต้นแพร่กลุ่มข้อมูล การถูกเลือกเพื่อแพร่กลุ่มข้อมูลต่อ การกระจายกลุ่มข้อมูลแทนโหนดที่ถูกเลือก และการกระจายให้โหนดที่ได้รับกลุ่มข้อมูลไม่ครบ โหนดจะทำการคำนวณค่าสำเนาใหม่ตามกรณีต่างๆที่ได้กล่าวมาแล้ว จากนั้นจึงแนบค่าสำเนาใหม่ไปพร้อมกับกลุ่มข้อมูล และแพร่กลุ่มข้อมูลนั้นออกไป จากนั้นจึงทำการปรับปรุงค่าสำเนาของตน

ขั้นตอนการทำงานสามารถสรุปได้ตามแผนภาพดังภาพที่ 3-11



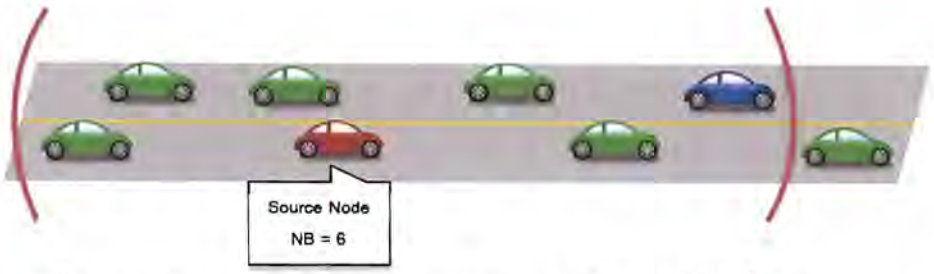
รูปที่ 3-11แผนภาพแสดงการทำงานของนโยบายการทิ้งข้อมูล

3.2.3 ตัวอย่างการทำงาน

ตัวอย่างการคำนวณค่าสำเนาของกลุ่มข้อมูลโดยแบ่งเป็นกรณีต่างๆ ดังนี้

1) กรณีที่โหนดต้นทาง (Source Node) แพร่กลุ่มข้อมูลให้กับเพื่อนบ้านตนเอง

โหนดต้นทางจะทราบว่ามีจำนวนเพื่อนบ้านทั้งหมดของตนเองเท่าไหร่ ดังภาพที่ 3.4 โหนดต้นทางจะแพร่กลุ่มข้อมูลให้กับโหนดเพื่อนบ้านทั้งหมด 6 โหนด ดังนั้นจำนวนสำเนาของกลุ่มข้อมูลก็จะมีค่าเท่ากับจำนวนเพื่อนบ้านของโหนดต้นทาง จากนั้นโหนดต้นทางจะแนบข้อมูลสำเนาไปกับกลุ่มของข้อมูลที่จะแพร่ให้กับโหนดเพื่อนบ้าน



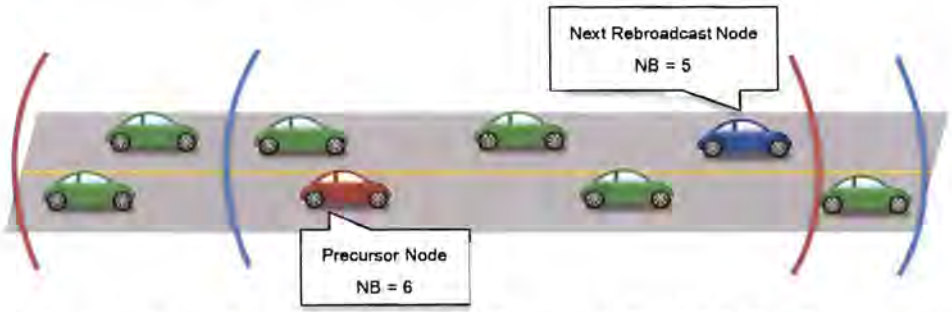
รูปที่ 3-12 การคำนวณค่าสำเนาในกรณีโหนดต้นทางแพร่กลุ่มข้อมูลให้กับเพื่อนบ้านตนเอง

ตัวอย่างการคำนวณค่าสำเนาของกลุ่มข้อมูล

$$copy = 6$$

2) กรณีโหนดส่งต่อข้อความแพร่กลุ่มข้อมูลให้กับเพื่อนบ้านของตนเอง

กรณีที่โหนดส่งต่อข้อความหรือโหนดที่ถูกเลือก (Next Rebroadcast Node) แพร่กลุ่มข้อมูลให้กับเพื่อนบ้านของตนเอง ดังภาพที่ 3.5 การกระจายในครั้งนี้จะเกิดพื้นที่ซ้อนทับของการสื่อสารขึ้น เนื่องจากก่อนหน้านี้มีการกระจายข้อมูลนี้มาแล้วโดยโหนดก่อนหน้า (Precursor Node) ซึ่งมีการแนบค่าสำเนาเดิมมากับกลุ่มข้อมูลนั้นด้วย ดังนั้นการคำนวณค่าสำเนาได้จากการรวมค่าสำเนาเดิมและค่าสำเนาใหม่ และหักค่าสำเนาที่คาดว่าจะซ้ำซ้อนในพื้นที่ทับซ้อนระหว่างโหนดก่อนหน้า (Precursor Node) และโหนดที่ถูกเลือก (Next Rebroadcast Node)



รูปที่ 3-13 การคำนวณค่าสำเนาในกรณีโหนดที่ถูกเลือกแพร่กลุ่มข้อมูลให้กับเพื่อนบ้านของตนเอง

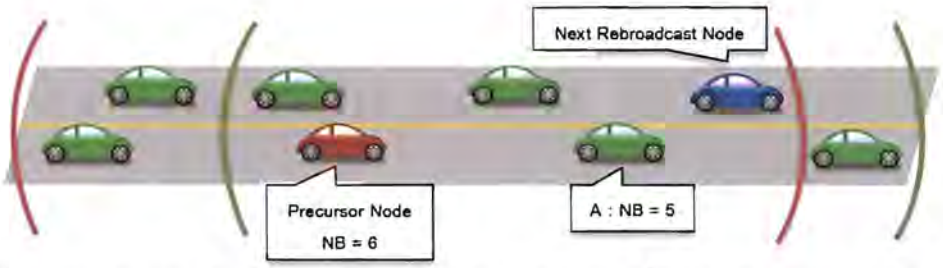
ตัวอย่างการคำนวณค่าสำเนาของกลุ่มข้อมูล

$$copy = (copy' + n_r) - (n_p \times \alpha)$$

$$copy = (6 + 5) - (6 \times 0.39) = 9$$

3) กรณีโหนดที่ถูกเลือกไม่ทำงาน และเพื่อนบ้านทำการกระจายข้อมูลแทน

การคำนวณค่าสำเนาของกลุ่มข้อมูลกรณีที่โหนดที่ถูกเลือก (Next Rebroadcast Node) ไม่ทำงาน ดังรูปที่ที่ 3-14 นั้น โหนดที่ได้รับกลุ่มข้อมูลจากโหนดก่อนหน้า (Precursor Node) ที่มีเวลาคอยสั้นที่สุดจะทำหน้าที่ในการกระจายข้อมูลแทน จากรูปสมมติว่าโหนด A มีเวลาคอยสั้นสุดและเป็นโหนดที่ทำหน้าที่แทนโหนดที่ถูกเลือก ซึ่งวิธีการคำนวณค่าสำเนากลุ่มข้อมูลจะเหมือนกันกับกรณีที่โหนดที่ถูกเลือกแพร่กลุ่มข้อมูล



รูปที่ 3-14 การคำนวณค่าสำเนาในกรณีโหนดที่ถูกเลือกไม่ทำงานและเพื่อนบ้านทำการกระจายข้อมูลแทน

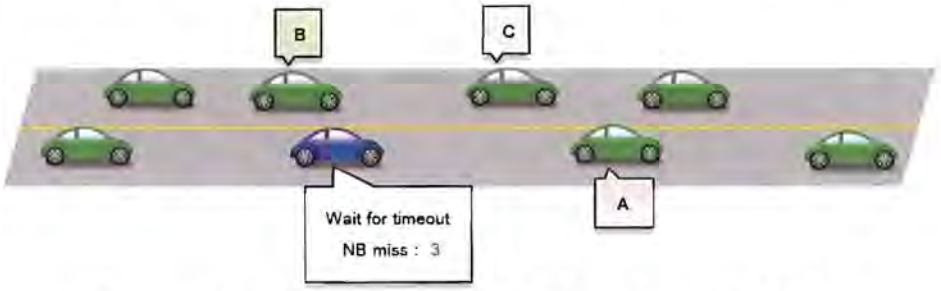
ตัวอย่างการคำนวณค่าสำเนาของกลุ่มข้อมูล

$$copy = (copy' + n_r) - (n_p \times \alpha)$$

$$copy = (6 + 5) - (6 \times 0.39) = 9$$

4) กรณีที่โหนดเพื่อนบ้านบางโหนดยังไม่ได้รับกลุ่มข้อมูล

กรณีที่โหนดเพื่อนบ้านบางโหนดยังไม่ได้รับกลุ่มข้อมูล ดังรูปที่ 3-15 จะสามารถทราบได้จากข้อมูลบีคอนที่โหนดใช้ในการแลกเปลี่ยนกัน จากภาพ สมมติว่าโหนดสีฟ้า ได้รับข้อมูลบีคอนแล้วพบว่า มีโหนด A ยังไม่ได้รับกลุ่มข้อมูล โหนดสีฟ้าก็จะทำการตั้งเวลาคอยเพื่อที่จะแพร่กลุ่มข้อมูลออกไป พร้อมทั้งนับจำนวนโหนดเพื่อนบ้านที่ยังไม่ได้รับข้อมูลที่ตรวจสอบพบภายในเวลาก่อนที่มีการกระจายข้อมูล จากรูปคือโหนด B,C เมื่อเวลาคอยสิ้นสุดลงโหนดสีฟ้าก็จะทำการคำนวณค่าสำเนาของกลุ่มข้อมูล โดยนำค่าสำเนาของข้อมูลเดิมบวกกับจำนวนโหนดที่ยังไม่ได้รับกลุ่มข้อมูลซึ่งหาได้จากจำนวนบีคอนของโหนดที่ยังไม่ได้รับกลุ่มข้อมูลนั้นภายในเวลาก่อนจะมีการกระจายกลุ่มข้อมูล จากนั้นก็แนบค่าที่คำนวณได้ไปกับกลุ่มข้อมูลที่ต้องส่งออกไป



รูปที่ 3-15 การคำนวณค่าสำเนาในกรณีโหนดเพื่อนบ้านบางโหนดยังไม่ได้รับข้อมูล

ตัวอย่างการคำนวณค่าสำเนาของกลุ่มข้อมูล

$$copy = copy' + n_b$$

$$copy = copy' + 3$$

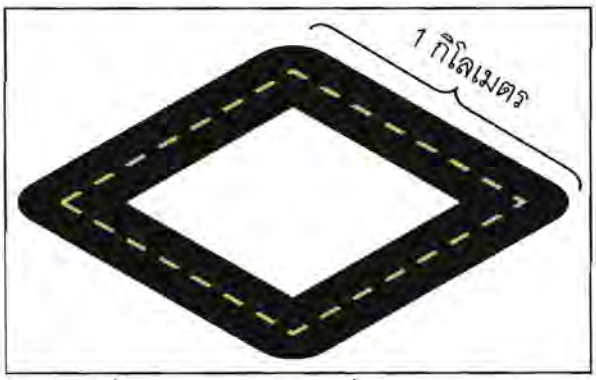
3.2.4 การทดลองและวิเคราะห์ผล

1) ตัววัดสมรรถนะ

- *จำนวนกลุ่มข้อมูลที่เหลืออยู่ในระบบ* เนื่องจากวัตถุประสงค์สำคัญของนโยบายในการทิ้งกลุ่มข้อมูลคือการทิ้งกลุ่มข้อมูลที่ถึงปลายทาง หรือกลุ่มข้อมูลที่หลงเหลืออยู่ในระบบมีการกระจายตัวที่ดี นั่นคือ หากกลุ่มข้อมูลใดยังไม่หมดอายุ ก็ควรที่จะมีโหนดที่เก็บกลุ่มข้อมูลนั้นเพื่อแพร่ต่อให้กับโหนดอื่นๆ และในกรณีที่กลุ่มข้อมูลมีความสำคัญเท่ากัน ปริมาณของกลุ่มข้อมูลที่เหลืออยู่ในระบบควรใกล้เคียงกัน เพื่อแสดงถึงสมรรถนะที่ดีของนโยบายการทิ้งกลุ่มข้อมูลนั้น จำนวนกลุ่มข้อมูลที่เหลืออยู่ในระบบคือจำนวนของกลุ่มข้อมูลนั้นๆที่โหนดในระบบเก็บเอาไว้ในบัฟเฟอร์
- *ความเชื่อถือได้ (Reliability)* เพื่อทดลองสมรรถนะของนโยบายการทิ้งกลุ่มข้อมูลในวิธีต่างๆ รวมทั้งนโยบายในการทิ้งกลุ่มข้อมูลที่เสนอ โดยค่าความเชื่อถือได้จะแสดงถึงความสามารถของนโยบายต่างๆในการจัดการบัฟเฟอร์ที่มีอย่างจำกัดให้ได้ประสิทธิภาพสูงสุด โดยค่าความเชื่อถือได้เป็นเปอร์เซ็นต์ของจำนวนรถที่ได้รับกลุ่มข้อมูลต่อจำนวนรถที่มีทั้งหมดในระบบ

2) เครื่องมือในการทดลอง ใช้เครื่องมือเช่นเดียวกับการทดลองในบทที่ 2 คือ เริ่มจากการจำลองพฤติกรรมการเคลื่อนที่ของรถยนต์โดยใช้โปรแกรม SUMO ซึ่งจะได้พฤติกรรมการเคลื่อนที่ของรถยนต์ออกมาในรูปแบบของ xml จากนั้นจะใช้โปรแกรม TraNS เพื่อแปลง XML ให้อยู่ในรูปแบบของ tcl ที่สามารถนำไปใช้งานบนโปรแกรมจำลอง NS-2.34 ได้ และใช้โปรแกรมจำลองเครือข่าย NS-2.34 ในการทดลอง

3) สภาพแวดล้อมที่ใช้ในการทดลอง การทดลองจะใช้ลักษณะของถนนทางหลวงขนาดความยาวรวม 4 กิโลเมตร ถนนจะเป็นลักษณะของสี่เหลี่ยมจัตุรัส มีความยาวด้านละ 1 กิโลเมตร รถยนต์สามารถวิ่งได้รอบ โดยเป็นถนน 2 ช่องทางจราจร คือ มี 1 ช่องทางสำหรับการเคลื่อนที่ในแต่ละทิศทาง โดยถนนที่ใช้ในการทดลองเป็นดังรูปที่ 3-16



รูปที่ 3-16 ลักษณะของถนนที่ใช้ในการทดลอง

ปริมาณของรถยนต์ในการทดลอง ใช้จำนวนรถยนต์ทั้งหมด 160 คัน เพื่อความสมจริงในการทดลอง รถยนต์จะมีการเข้าและออกจากระบบทุกๆ 70 วินาที เพื่อให้มีโหนดใหม่ที่ยังไม่ได้รับกลุ่มข้อมูลเข้ามาในระบบ ซึ่งจะสามารถทดลองสมรรถนะของนโยบายการทิ้งกลุ่มข้อมูลได้ดีขึ้น โดยลักษณะการเข้าออกของรถยนต์จะเป็นการสุ่มซึ่งไม่พิจารณาตำแหน่งของรถในขณะนั้น โดยปริมาณของรถยนต์ที่เข้า-ออกเป็นไปตามตารางที่ 3-5

ในการทดลองจะประกอบด้วยกลุ่มข้อมูลจำนวน 40 กลุ่มข้อมูลที่มีหมายเลขประจำตัวกลุ่มข้อมูลโดยเรียงจาก 1-40 ตามลำดับ ซึ่งจะถูกแพร่จากโหนดต้นทาง 40 โหนดแบบสุ่ม นั่นคือจะมี 1 โหนดในระบบขณะนั้นแพร่กลุ่มข้อมูลออกมา 1 กลุ่มข้อมูล โดยจะมีการสุ่มทุกๆ 0.5 วินาทีจนกว่าจะครบ 40 กลุ่มข้อมูลในระบบ โดยการทดลองจะเริ่มต้น

หลังจากปล่อยให้รถยนต์มีการวิ่งไปแล้ว 100 วินาที และจบหลังจากกลุ่มข้อมูลตัวสุดท้ายหมดอายุ รวมการทดลองในโปรแกรมจำลองเป็นเวลา 320 วินาที และค่าอื่นๆสามารถสรุปได้ตามตารางที่ 3-6

ตารางที่ 3-5 ปริมาณของรถยนต์ในการทดลอง

เวลาในการทดสอบ (วินาที)	จำนวนรถยนต์ในระบบขณะทำการทดลอง	ปริมาณโหนดที่เข้าออกในระบบ
0 - 170	80	-
170 - 240	80	เข้า 40 คัน ออก 40 คัน
240 - 320	80	เข้า 40 คัน ออก 40 คัน

ตารางที่ 3-6 การตั้งค่าต่างๆที่ใช้ในการทดลอง

การทดลอง	จำนวนครั้งในการทดลอง : 10 ครั้ง ผลจากการทดลองเป็นค่าเฉลี่ย ระยะเวลาที่ใช้ในการทดลองต่อครั้ง : 320 วินาที
การสื่อสารไร้สาย	มาตรฐาน : IEEE802.11 ระยะสื่อสารสูงสุด : 250 เมตร
สภาพการจราจร	ปริมาณรถยนต์ (จำนวนรถยนต์ขณะทดสอบ) : 160 (80) คัน ความเร็วสูงสุด : 50, 80 กิโลเมตร/ชั่วโมง
กลุ่มข้อมูล (Packet)	อายุ : 200 วินาที ขนาด : 512 ไบต์ จำนวนกลุ่มข้อมูลขณะทดสอบ : 40 กลุ่มข้อมูล
การตั้งค่าโพรโทคอล	ช่วงเวลาการทำบีคอน : ทุก 1.5-7 วินาที เวลาในการรอแพร่กลุ่มข้อมูลสูงสุด : 0.2 วินาที ขนาดของบัฟเฟอร์ : 5,10,15กลุ่มข้อมูล
นโยบายการทิ้งกลุ่มข้อมูลที่ใช้ในการทดลอง	- Drop Tail - Drop Front - Random Drop - Number of Copy Aware Policy

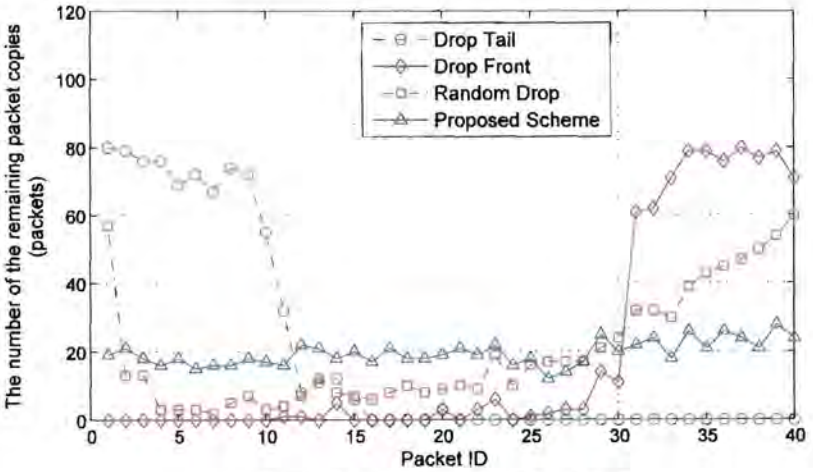
4) ผลการทดลอง

- **ผลการทดลองจำนวนกลุ่มข้อมูลที่เหลืออยู่ในระบบ** เมื่อพิจารณารูปที่ 3-17 พบว่านโยบายการทิ้งข้อมูลแบบพิจารณาค่าสำเนานั้นให้ผลการทดลองจำนวนกลุ่มข้อมูลที่เหลืออยู่ได้ดีที่สุด เนื่องจากโหนดในเครือข่ายมีการกระจายกลุ่มข้อมูลหมายเลขต่างๆได้ดี โดยแนวโน้มของผลการทดลองเป็นเช่นเดียวกันในทุกขนาดของบัฟเฟอร์ เมื่อเปรียบกับนโยบายการทิ้งกลุ่มข้อมูลแบบดั้งเดิมที่กลุ่มข้อมูลบางตัวจะถูกทิ้งจากบัฟเฟอร์จนสูญหายไปจากระบบ โดยการทิ้งกลุ่ม

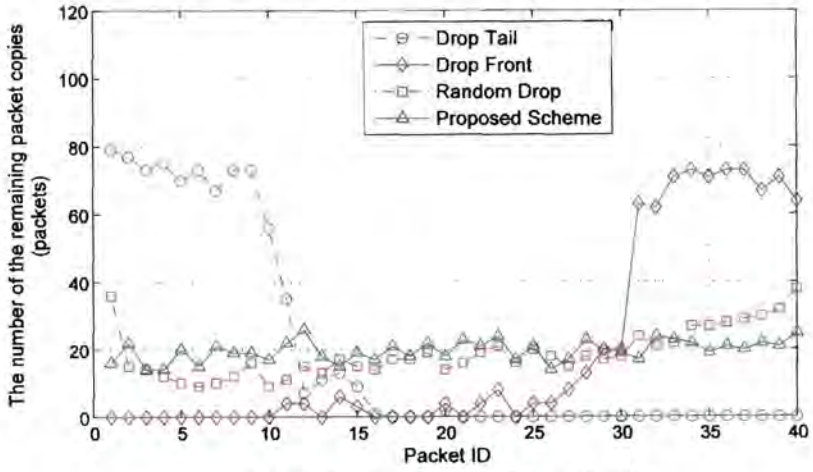
ข้อมูลแบบสุ่มให้ผลได้ดีที่สุดในการทิ้งกลุ่มข้อมูลแบบดั้งเดิม แต่ยังมีปัญหาในการกระจายกลุ่มข้อมูลที่เหลืออยู่ในได้แย่กว่านโยบายการทิ้งของมูลแบบพิจารณาค่าสำเนา

- **ผลการทดลองค่าความเชื่อถือได้** พิจารณารูปที่ 3-18 จากความสามารถในการกระจายกลุ่มข้อมูลที่มีอยู่ในระบบได้อย่างเท่าเทียมทำให้นโยบายการทิ้งกลุ่มข้อมูลแบบพิจารณาค่าสำเนาสามารถทำงานได้ดีมาก โดยสังเกตได้จากค่าความเชื่อถือได้ที่มีค่าสูงกว่านโยบายการทิ้งข้อมูลแบบดั้งเดิมที่บัฟเฟอร์มีขนาดน้อยๆ โดยการทิ้งข้อมูลแบบดั้งเดิมจะสามารถทำงานได้ดีขึ้นเมื่อมีขนาดบัฟเฟอร์ที่ใหญ่ขึ้น เนื่องจากกระบวนการทิ้งข้อมูลแบบดั้งเดิมไม่ได้พิจารณาการคงเหลือของกลุ่มข้อมูลซึ่งทำให้ทิ้งกลุ่มข้อมูลที่คงเหลืออยู่ในระบบน้อยจนทำให้กลุ่มข้อมูลเหล่านั้นไม่ถูกส่งไปยังโหนดอื่นๆ สังเกตได้จากกลุ่มข้อมูลบางเลขมีค่าความเชื่อถือที่ต่ำมากดังผลการทดลอง โดยแนวโน้มของผลการทดลองเป็นเช่นเดียวกันในทุกขนาดของบัฟเฟอร์

ดังนั้นการใช้งานนโยบายการทิ้งกลุ่มข้อมูลแบบพิจารณาค่าสำเนาจึงเหมาะกับการใช้งานบนโพรโทคอล DECA เนื่องจากให้ประสิทธิภาพในการทำงานที่สูง โดยใช้เพียงค่าความหนาแน่นของโหนดเพียงเท่านั้น

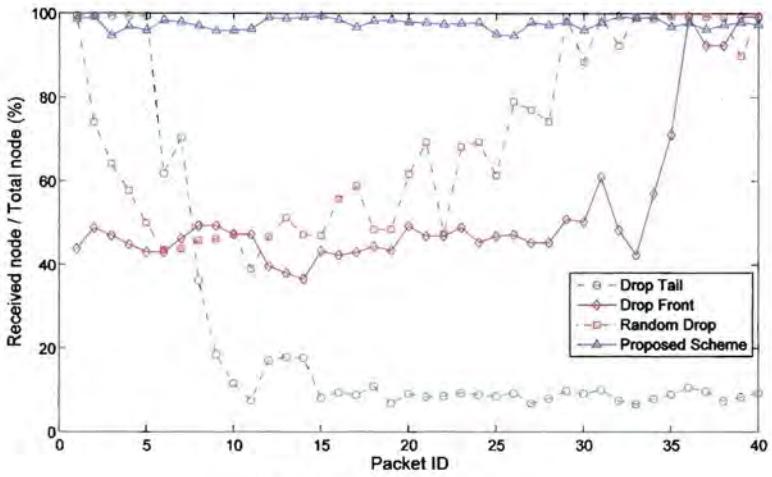


ก) เมื่อเวลาในการทดลองผ่านไป 50 วินาที

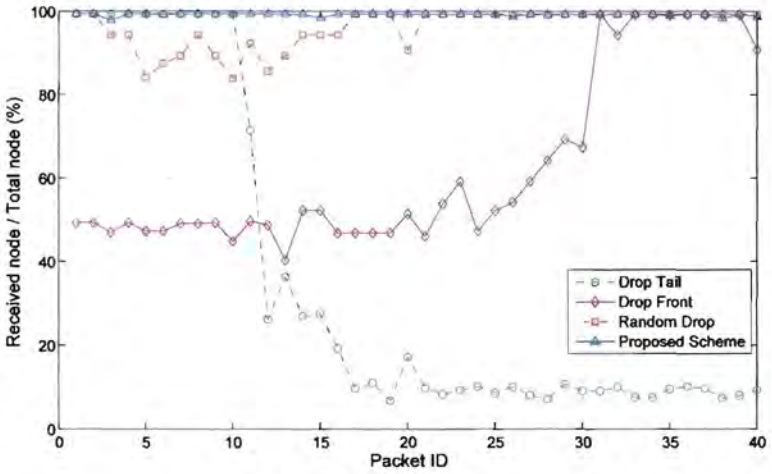


ข) เมื่อเวลาในการทดลองผ่านไป 150 วินาที

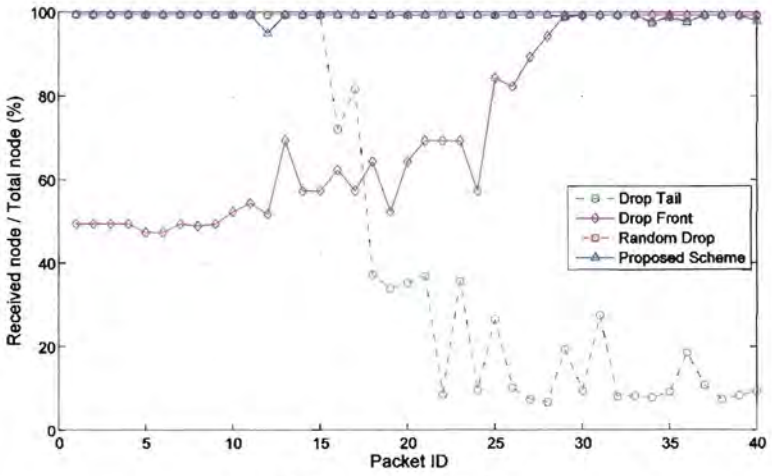
รูปที่ 3-17 กราฟแสดงจำนวนของกลุ่มที่เหลืออยู่ในระบบบัฟเฟอร์ขนาด 10 กลุ่มข้อมูลที่เวลาต่างๆ



ก) บัฟเฟอร์ขนาด 5 กลุ่มข้อมูล



ข) บัฟเฟอร์ขนาด 10 กลุ่มข้อมูล



ค) บัฟเฟอร์ขนาด 15 กลุ่มข้อมูล

รูปที่ 3-18 กราฟแสดงผลการทดลองค่าความเชื่อถือได้จากบัฟเฟอร์ขนาดต่างๆ

3.3 การปรับปรุงการทำงานของโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง

3.3.1 ปัญหาที่ขึ้นเมื่อทดลองโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง

ในสภาพการจราจรที่แท้จริงโพรโทคอลจะต้องสามารถรองรับการทำงานที่เกิดขึ้นในชั่วเร่งด่วนได้ ซึ่งในสภาพแวดล้อมนั้นย่อมประกอบด้วยรถยนต์จำนวนมาก รวมทั้งปริมาณของข้อมูลที่อยู่ในระบบย่อมเพิ่มขึ้นทวีคูณ ดังนั้นการทดสอบโพรโทคอลในสภาพการจราจรที่มีความแน่นสูงอย่างมากย่อมสะท้อนถึงประสิทธิภาพของโพรโทคอลในการรองรับการเจริญเติบโตของผู้ใช้งานในอนาคตได้

เนื่องจากการทดลองสมรรถนะในบทที่ผ่านมาเป็นการใช้สภาพจราจรอย่างง่ายและมีรถยนต์ในระบบไม่เกิน 1000 คัน การทดสอบในตอนนี้จะมีรถยนต์ในระบบสูงสุดถึง 2760 คัน โดยมีการทดลองลงบนแผนที่จริงบริเวณถนนสุขุมวิทของกรุงเทพมหานคร ดังรูปที่ 3-19โดยที่ถนนมีจำนวนช่องทางจราจร ทิศทางการวิ่ง และไฟจราจรเสมือนถนนจริง ดังนั้นผลการทดลองที่ได้ย่อมจะสะท้อนถึงความสามารถของโพรโทคอลในสถานการณ์การใช้งานจริงที่มีโหนดในระบบเป็นจำนวนมากได้

ก) แผนที่จริงที่ใช้ในการทดลอง ข) ลักษณะการเคลื่อนที่ของโหนดในโปรแกรมจำลอง

รูปที่ 3-19 แผนที่บริเวณถนนสุขุมวิท กรุงเทพมหานครที่ใช้ในการทดลอง

ผลจากการทดลองพบว่าประสิทธิภาพในการทำงานของโพรโทคอล DECA มีค่าความเชื่อถือได้ลดลง โดยที่มีค่าใช้จ่ายในการทำงานที่เพิ่ม จากการสังเกตสาเหตุของปัญหาเกิดขึ้นจากการการชนกันในการส่งข้อมูลในระบบ โดยเกิดขึ้นจาก 2 กระบวนการหลักคือ

1) ปัญหาจากการคำนวณเวลารอ เนื่องจากการคำนวณเวลารอนั้นใช้สำหรับการหลีกเลี่ยงการชนกันของการส่งข้อมูลในกรณีที่ไม่มีการเลือกโหนดส่งต่อข้อความ แต่เมื่อโหนดมีจำนวนมากขึ้นฟังก์ชันสำหรับการคำนวณเดิมนั้นไม่สามารถรองรับการทำงานได้ จึงเกิดปัญหาการชนเป็นจำนวนมาก และส่งผลต่อประสิทธิภาพของระบบ

2) ปัญหาจากกระบวนการร้องขอข้อความที่ไม่ได้รับ เนื่องจาก DECA ทำงานอยู่บนเครือข่ายที่มีการเชื่อมต่อเป็นช่วงๆ ดังนั้นจึงต้องอาศัยกระบวนการในการร้องขอข้อความจากโหนดอื่นๆเพื่อให้ตนเองมีข้อมูลในการทำงานครบ แต่

กระบวนการร้องขอเดิมจะมีการร้องขอทุกครั้งที่ได้รับบีคอนจากเพื่อนบ้านจึงทำให้เกิดการร้องขอจำนวนมหาศาลเมื่อมีโหนดเพื่อนบ้านมาก ส่งผลให้เกิดการชนการขณะที่มีการส่งข้อมูล และส่งผลต่อประสิทธิภาพของระบบ

3.3.2 การปรับปรุงกระบวนการทำงานของโพรโทคอลในสภาพแวดล้อมที่มีความหนาแน่นสูง

3.3.1.1 การปรับปรุงขั้นตอนการคำนวณเวลารอ

ขั้นตอนการคำนวณเวลาเดิมนั้นใช้ฟังก์ชั่นที่มีค่าแปรผกผันกันค่าความหนาแน่นของโหนดโดยรอบ โดยมีจุดประสงค์เพื่อให้โหนดบริเวณที่มีความหนาแน่นสูงได้โอกาสในการกระจายข้อความก่อนโหนดในบริเวณอื่นๆซึ่งสามารถทำงานได้ดีในสภาพแวดล้อมที่มีจำนวนโหนดไม่สูงมาก แต่เมื่อทำงานในสภาพแวดที่มีโหนดเป็นจำนวนมากทำให้ช่วงเวลาที่โหนดจะสุ่มค่าออกมาได้นั้นมีช่วงที่แคบลงทำให้เกิดการชนกันในการส่งข้อมูลเป็นจำนวนมาก ดังนั้นในการแก้ปัญหาจึงเปลี่ยนฟังก์ชั่นการคำนวณเวลารอเป็นเวลารอจะแปรผันตรงกับค่าความหนาแน่น โดยมีรายละเอียดในการออกแบบดังนี้

1) การคำนวณเวลารอแบบแปรผันตรงจะทำให้โหนดที่จะมีการกระจายข้อความบริเวณที่มีความหนาแน่นสูงสามารถสุ่มช่วงเวลาได้กว้างขึ้น ส่งผลให้ปัญหาการชนในบริเวณที่มีความหนาแน่นสูงลดลงได้

2) ในกรณีที่มีสภาพการจราจรแบบแยกส่วน (Partition Networks) โดยแบ่งออกเป็นบริเวณที่มีความหนาแน่นสูงและบริเวณความหนาแน่นต่ำ บริเวณที่มีความหนาแน่นสูงจะมีเวลารอนานกว่าแต่เนื่องจากมีความหนาแน่นสูงจึงมีปัญหาการเชื่อมต่อขาดน้อยกว่า ส่งผลให้การกระจายไม่สำเร็จมีโอกาสน้อยจากเวลารอที่นานขึ้น ส่วนบริเวณที่มีความหนาแน่นต่ำก็มีเวลารอที่สั้นกว่าทำให้เพิ่มโอกาสในการกระจายข้อความสำเร็จก่อนที่การเชื่อมต่อจะขาดได้

3) การคำนวณเวลารอใหม่มีการเพิ่มเวลารอที่จะเกิดจากความเสียงในการส่งบีคอน และ Propagation Delay ที่จะเกิดขึ้นเมื่อโหนดเพื่อนบ้านได้รับบีคอน แล้วทำการร้องขอข้อความจากโหนดอื่นๆ จึงลดปัญหาการชนกันในขณะที่มีการส่งข้อมูลได้

การคำนวณเวลาแบบใหม่ และแบบเก่าสามารถเปรียบเทียบกันได้ดังกราฟในรูปที่ 3-20โดยการคำนวณเวลารอแบบใหม่นั้นสามารถคำนวณได้ตามสมการที่ (20)

$$W_{min}(n) = n\tau \tag{18}$$

$$W_{max}(n) = (2 + n\beta)\tau \tag{19}$$

$$W = Random[W_{min}, W_{max}] \tag{20}$$

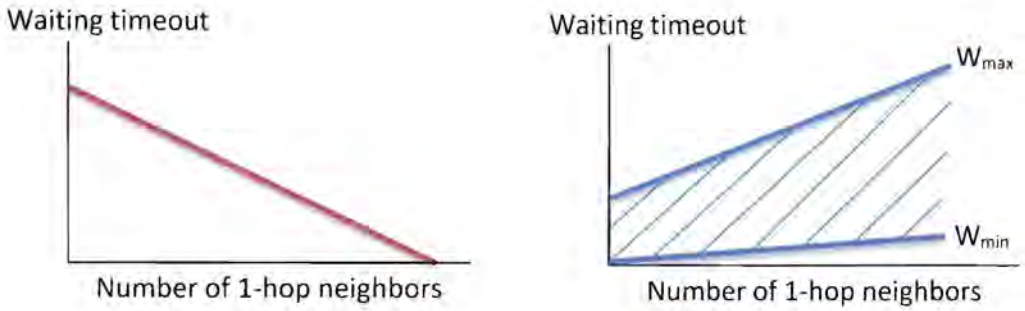
เมื่อ W คือ เวลารอที่คำนวณได้ (วินาที)

W_{min}, W_{max} คือ เวลารอต่ำสุด และสูงสุด

n คือ จำนวนโหนดเพื่อนบ้านใน 1 hop

τ คือ เวลาหน่วงที่ประมาณเวลาในการรับ-ส่งข้อมูล

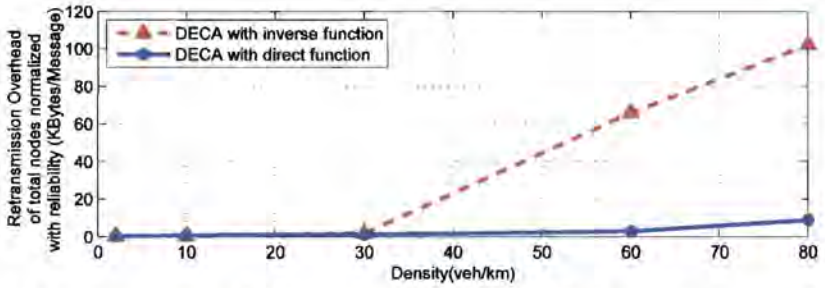
β คือ ค่าคงที่ที่ใช้ในการขยายช่วงของเวลารอเพื่อลดการชนในบริเวณที่มีความหนาแน่นสูง



ก) ความสัมพันธ์แบบแปรผกผัน ข) ความสัมพันธ์แบบแปรผันตรง

รูปที่ 3-20 กราฟแสดงความสัมพันธ์ระหว่างเวลารอและความความหนาแน่นของโหนด

ผลการทดลองการกระจายข้อความซ้ำเมื่อใช้งานวิธีการคำนวณเวลารอใหม่ ซึ่งเป็นค่าใช้จ่ายที่เกิดขึ้นในการทำงานของโพรโทคอลเป็นไปดังกราฟในรูปที่ 3-21 โดยการตั้งค่าเป็นไปตามการทดลองในหัวข้อที่ 3.3.3 จากผลการทดลองแสดงให้เห็นว่าขั้นตอนการคำนวณเวลารอใหม่นั้นสามารถช่วยลดค่าใช้จ่ายที่เกิดขึ้นได้มากถึง 90% ของค่าใช้จ่ายที่เกิดจากการกระจายข้อมูลซ้ำของวิธีการตั้งเวลารอแบบเดิม

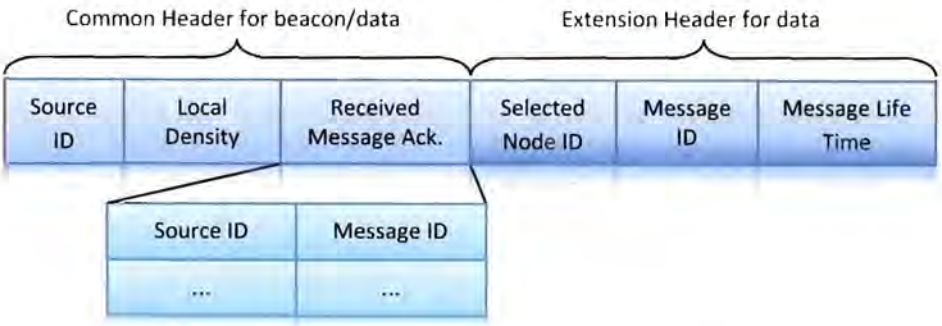


รูปที่ 3-21 กราฟแสดงผลการทดลองค่าใช้จ่ายจากการกระจายข้อความซ้ำด้วยการคำนวณเวลารอที่แตกต่างกัน

3.3.1.2 การปรับปรุงขั้นตอนการร้องขอข้อความที่ไม่ได้รับ

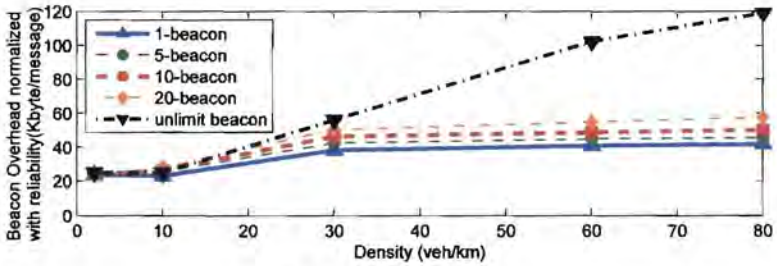
เนื่องจากโหนดสามารถตรวจสอบข้อความที่ตนเองมีกับโหนดเพื่อนบ้านโดยการแลกเปลี่ยนบีคอนกัน ซึ่งภายในบีคอนจะประกอบข้อมูลต่างๆดังที่กล่าวในบทที่ 2 หรือดังรูปที่ 3-22 โดยเมื่อโหนดตรวจสอบพบว่าตนเองมีข้อความที่ไม่ได้รับจะทำการส่งบีคอนของตนเองออกไปเพื่อให้โหนดเพื่อนบ้านสามารถตรวจพบและส่งข้อความนั้นให้ ซึ่งในกรณีที่มีความหนาแน่นน้อยปริมาณการส่งบีคอนออกไปก็มีน้อยจึงไม่ส่งผลต่อประสิทธิภาพของโพรโทคอล แต่ในกรณีที่มีความหนาแน่นสูงการที่โหนดส่งบีคอนออกไปเพื่อร้องขอข้อความที่ไม่ได้รับทุกครั้งที่ได้รับบีคอนจากเพื่อนย่อมส่งผลต่อความหนาแน่นของเครือข่าย ซึ่งมีความหนาแน่นสูงจากปริมาณโหนดอยู่แล้ว ดังนั้นจึงทำให้ประสิทธิภาพในการทำงานของโพรโทคอลลดลง รวมถึงค่าความเชื่อถือได้ก็มีค่าลดลงด้วย

ดังนั้นในการแก้ปัญหาคือการจำกัดปริมาณการร้องขอข้อความที่ไม่ได้รับจากเพื่อนบ้าน โดยจำกัดเป็นจำนวนครั้งหลังได้รับบีคอนที่พบว่าตนเองมีข้อความที่ไม่ได้รับ จากนั้นจะไม่สามารถส่งการร้องขอได้ ถ้ามีการส่งบีคอนเพื่อร้องขอครบตามจำนวนจนกว่าจะถึงช่วงเวลาในการทำบีคอนตามเวลาที่คำนวณได้ตามปกติ จึงจะสามารถส่งการร้องรอบใหม่ได้ แต่การจำกัดการร้องขอนั้นส่งผลถึงค่าความเชื่อถือได้ในการกระจายเนื่องจากโหนดมีโอกาสที่จะร้องขอข้อความที่ไม่ได้รับ

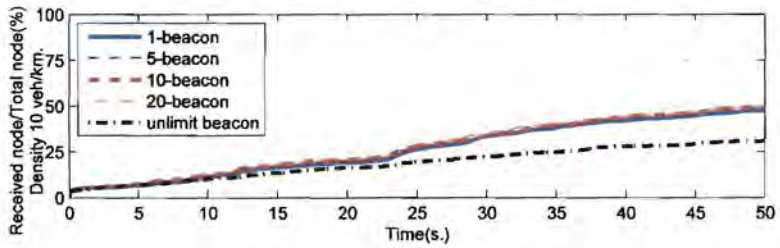


รูปที่ 3-22 โครงสร้างบีคอนของโพรโทคอล DECA

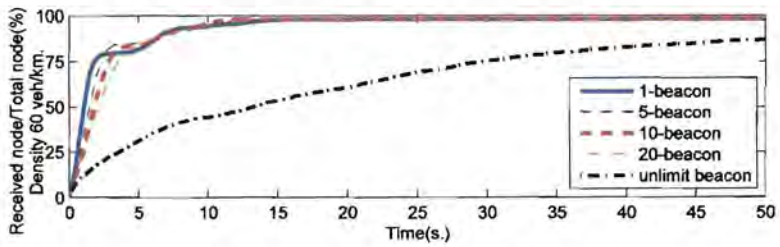
จากเพื่อนบ้านน้อยลง จำนวนครั้งการร้องขอที่ดีจึงต้องไม่ส่งผลต่อความเร็วในการกระจายข้อมูล และสามารถลดค่าใช้จ่ายที่เกิดขึ้นจากการส่งบีคอนได้มากที่สุด โดยจากการทดลองพบว่าการส่งการร้องขอ 1 ครั้งหลังจากพบข้อความที่ตนเองไม่ได้รับจนถึงการส่งบีคอนครั้งถัดไปให้ประสิทธิภาพในการทำงานได้ดีที่สุด สังเกตได้จากผลการทดลองค่าใช้จ่ายในการส่งบีคอน และผลการทดลองความเร็วในการกระจายข้อความที่มีความหนาแน่นแตกต่างกันดังรูปที่ 3-23 โดยการตั้งค่าต่างๆ เป็นไปตามการทดลองในหัวข้อที่ 3.3.3



ก) ค่าใช้จ่ายจากการส่งบีคอน



ข) ความเร็วในการกระจายที่ความหนาแน่น 10 คัน/กม.



ค) ความเร็วในการกระจายที่ความหนาแน่น 60 คัน/กม.

รูปที่ 3-23 ผลการทดลองการจำกัดจำนวนครั้งในการร้องขอข้อความที่ไม่ได้รับ

3.3.3 การทดลองและวิเคราะห์ผลการทดลอง

1) ตัววัดสมรรถนะ

- *ความเชื่อถือได้ (Reliability)* วัตถุประสงค์หลักในการออกแบบโพรโทคอลคือสามารถส่งผ่านข้อมูลให้กับรถยนต์ในบริเวณพื้นที่หนึ่งหนึ่งให้ได้รับข้อมูลโดยอย่างเชื่อถือได้ ดังนั้นค่าความเชื่อถือได้เป็นค่าเปอร์เซ็นต์จากจำนวนรถยนต์ที่ได้รับข้อความต่อจำนวนรถยนต์ทั้งหมดในการทดลองแต่ละครั้ง ความเชื่อถือได้ที่สูงแสดงถึงสมรรถนะที่สูงของโพรโทคอล

- *ค่าใช้จ่าย (Overhead)* ในการกระจายข้อมูลโดยที่ยังมีความเชื่อถือสูงสุดย่อมมีค่าใช้จ่ายสูงกว่า ดังนั้นในการทดลองจึงมีการนับจำนวนครั้งการส่งข้อความในการกระจายข้อความซ้ำทั้งหมดเพื่อให้รถยนต์ทั้งหมดได้รับข้อความในช่วงเวลาที่กำหนด และจำนวนของบีคอนทั้งหมดที่เกิดขึ้น ค่าใช้จ่ายจะแสดงเป็นปริมาณของเครือข่ายที่ถูกใช้ในการทำงานต่อการกระจายหนึ่งข้อความ ซึ่งแสดงถึงประสิทธิภาพของโพรโทคอล

- *ความเร็วของการกระจายข้อมูล (Speed of Data Dissemination)* เนื่องจากการบริการที่มีอยู่ในปัจจุบันสำหรับรถยนต์ในระบบจราจรอัจฉริยะ เช่น การหลีกเลี่ยงอุบัติเหตุ หรือการวางแผนการเดินทาง ซึ่งข้อมูลที่บริการเหล่านี้ใช้จำเป็นจะต้องเป็นข้อมูลที่ทันท่วงทีต่อเหตุการณ์ที่เกิดขึ้น ดังนั้นการกระจายข้อมูลจึงจะต้องทำได้รวดเร็วพอที่จะให้ข้อความที่แพร่ออกไป ยังมีประโยชน์ต่อระบบ ดังนั้นถ้าสามารถแพร่ข้อมูลได้รวดเร็วได้เท่าใดก็ยิ่งทำให้การกระจายข้อมูลครั้งนั้นมีโอกาสนำไปใช้ประโยชน์ได้เพิ่มมากขึ้น

2) เครื่องที่ใช้ในการทดลอง ใช้เครื่องมือเช่นเดียวกับการทดลองในบทที่ 2 คือ เริ่มจากการจำลองพฤติกรรมการเคลื่อนที่ของรถยนต์โดยใช้โปรแกรม SUMO ซึ่งจะได้พฤติกรรมการเคลื่อนที่ของรถยนต์ออกมาในรูปแบบของ xml จากนั้นจะใช้โปรแกรม TraNS เพื่อแปลง XML ให้อยู่ในรูปแบบของ tcl ที่สามารถนำไปใช้งานบนโปรแกรมจำลอง NS-2.34 ได้ และใช้โปรแกรมจำลองเครือข่าย NS-2.34 ในการทดลอง

3) สภาพแวดล้อมที่ใช้ในการทดลอง การทดลองใช้ถนนเสมือนจากแผนที่จริง ดังรูปที่ 3-19 มีระยะทางรวม 34.5 กิโลเมตร โดยในการทดลองมีการจำลองจำนวนช่องจราจร การกำหนดทิศทางบังคับ และไฟจราจรเสมือนจริง

ในการทดลองมีโหนดต้นทางทำหน้าที่ในการกระจายข้อมูลจำนวน 10 โหนดพร้อมกัน โดยจะมีการกระจายข้อความทุกๆ 50 วินาที ในการทดลองจะใช้ระบบสื่อสารไร้สายบนมาตรฐาน IEEE802.11 ซึ่งมีระยะสื่อสารสูงสุดที่ 250 เมตร โดยการตั้งค่าโพรโทคอล และค่าต่างๆที่ใช้ในการทดลองตามตารางที่ 3-7 โพรโทคอลที่ใช้ในการทดลองมีดังต่อไปนี้

- *Simple Flooding (SF)* : วิธีการกระจายข้อมูลแบบดั้งเดิมที่ไม่ต้องการข้อมูลในการทำงาน และสามารถแพร่ข้อมูลได้เร็วที่สุด
- *EAEP* : วิธีการกระจายข้อมูลใช้เทคนิค Store-and-Forward โดยใช้ความน่าจะเป็นในการคำนวณโหนดที่จะแพร่ข้อความต่อ โดยให้โหนดบริเวณขอบของระยะการเชื่อมต่อมีความน่าจะเป็นสูงกว่า [10]
- *AckPBSM* : โพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้ที่มีสมรรถนะดีที่สุดในงานวิจัยที่พบ ทำงานโดยใช้ข้อมูลจีพีเอส ใช้เทคนิค Store-and-Forward และใช้บีคอนในการทำงาน การตั้งค่าการทำงานเป็นไปตาม [11]
- *DECA* : การกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ ทำงานโดยไม่ใช้ข้อมูลจากจีพีเอส ใช้เทคนิค Store-and-Forward และใช้บีคอนในการทำงาน
- *DECA-bewa* : การกระจายอย่างเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นบนเครือข่ายไร้สายแบบแอดฮอกสำหรับยานพาหนะ โดยปรับปรุงวิธีการคำนวณเวลารอ และจำกัดการร้องขอข้อความที่ไม่ได้รับ โดยทดลองการส่งบีคอนทุก 1 วินาที และการส่งบีคอนแบบการปรับตามความหนาแน่นโดยใช้ LIA

ตารางที่ 3-7 การตั้งค่าต่างๆที่ใช้ในการทดลอง

การทดลอง	จำนวนครั้งในการทดลอง : 20 ครั้ง ผลจากการทดลองเป็นค่าเฉลี่ย ระยะเวลาที่ใช้ในการทดลองต่อครั้ง : 200 วินาที
การสื่อสารไร้สาย	มาตรฐาน : IEEE802.11 Two-Ray Ground Propagation Loss Model ระยะสื่อสารสูงสุด : 250 เมตร
สภาพการจราจร	ความหนาแน่น : 2, 10, 20, 30, 60 และ 80 คัน ความเร็วสูงสุด : 50, 80 กิโลเมตร/ชั่วโมง
กลุ่มข้อมูล (Packet)	อายุ : 50 วินาที ขนาด : 512 ไบต์ จำนวนข้อความขณะทดสอบ : 10 ข้อความ
EAEP	ช่วงเวลาในการกระจายข้อมูลซ้ำ : 0 - 10 วินาที
AckPBSM	ช่วงเวลาการทำบีคอน : ทุก 0.5 วินาที
DECA/DECA-bewa	เวลารอสูงสุด : 0.2 วินาที ช่วงเวลาการทำบีคอน : LIA (ทุก 1.5 - 7 วินาที) $c = 0.2$, $MinInv = 1.5$, $MaxInv = 7$

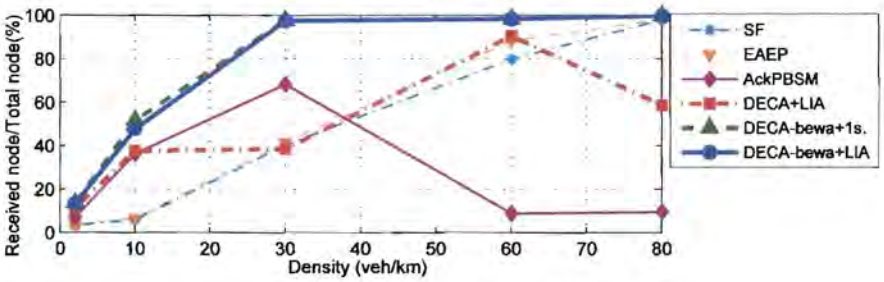
4) ผลการทดลอง

- **ผลการทดลองค่าความเชื่อถือได้** พิจารณารูปที่ 3-24 ค่าความเชื่อถือได้ SF และ EAEP เป็นไปตามการทดสอบในบทที่ 2 คือที่ความหนาแน่นต่ำไม่สามารถทำงานได้ดี เนื่องจากไม่มีกระบวนการตรวจสอบข้อความที่ไม่ได้รับ ดังนั้นจึงเกิดปัญหาโหนดไม่ได้รับข้อความเมื่อมีการเชื่อมต่อเป็นช่วงๆเป็นเวลานาน และค่าความเชื่อถือได้จะมีค่าเพิ่มขึ้นเมื่อมีความหนาแน่นเพิ่มขึ้น กรณี DECA เดิมและ AckPBSM มีค่าความเชื่อถือได้ที่ลดลงเมื่อความหนาแน่นเพิ่มขึ้น เนื่องจาก DECA เจอปัญหาที่กล่าวมาข้างต้น คือ การชนกันของข้อมูลเนื่องจากเวลารอที่สั้นเกินไป และการร้องขอข้อความที่ไม่ได้รับจำนวนมากเกินไป ส่วน AckPBSM เกิดการการตั้งเวลารอที่สั้นเกินไปเช่นเดียวกันซึ่งทำให้โหนดที่มีการกระจายข้อมูลเกิดการชนกันเป็นจำนวนมาก อีกทั้งช่วงเวลาในการทำบีคอนที่สั้นมากจึงทำให้เกิดปัญหาการชนขึ้นที่เกิดขึ้นร้ายแรงกว่าของ DECA และค่าความเชื่อถือได้ลดลงที่ความหนาแน่นของโหนดต่ำกว่า

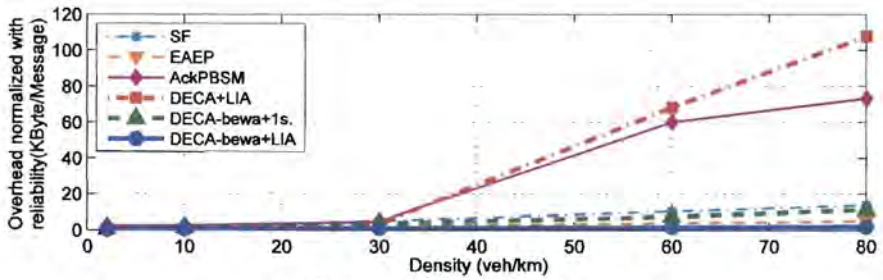
DECA-bewa ให้ความน่าเชื่อถือที่สูงที่สุด เนื่องจากหลีกเลี่ยงปัญหาการชนที่เกิดขึ้นระหว่างการทำงานที่บริเวณที่มีความหนาแน่นมาก โดยช่วงเวลาการส่งบีคอนแบบคงที่ และแบบปรับค่าตามความหนาแน่นมีค่าความหนาแน่นเท่าดังนั้นการส่งบีคอนแบบค่าได้ LIA ไม่ส่งผลต่อสมรรถนะในการทำงานของโพรโทคอล

- **ผลการทดลองค่าใช้** พิจารณารูปที่ 3-25 จากกราฟสังเกตได้ว่า DECA และ AckPBSM มีค่าใช้จ่ายสูงใกล้เคียงกันซึ่งทำให้เกิดการชนและส่งผลต่อค่าความเชื่อถือได้ของโพรโทคอล ซึ่งเมื่อพิจารณาโพรโทคอลที่มีการทดลองอื่นๆ DECA-bewa มีค่าใช้จ่ายในการทำงานต่ำที่สุด ซึ่งเมื่อเทียบกับค่าความเชื่อถือที่ได้ DECA-bewa จึงมีประสิทธิภาพในการทำงานสูงที่สุด ดังนั้นการปรับปรุงการทำงานของ DECA สามารถช่วยให้ DECA ทำงานในสภาพแวดล้อมที่มีความหนาแน่นสูงได้อย่างมีประสิทธิภาพสูงที่สุด และในการทดสอบ DECA-bewa ที่ใช้ LIA สามารถลดค่าใช้จ่ายลงได้ถึง 79%

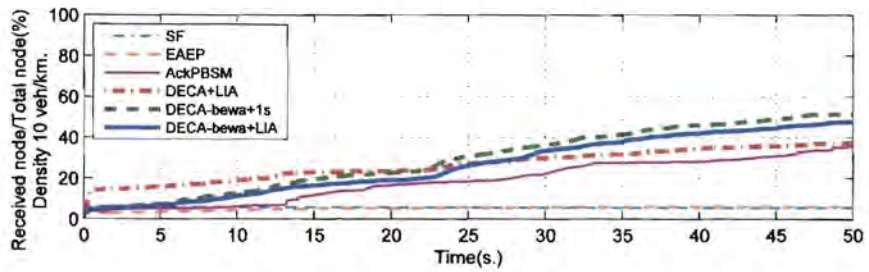
- **ผลการทดลองความเร็วในการกระจายข้อความ** พิจารณารูปที่ 3-26 จากกราฟที่ความหนาแน่นทั้งสองสถานการณ์ DECA-bewa มีความเร็วในแพร่สูงที่สุด โดยมี DECA ก่อนการปรับปรุงสามารถทำงานที่ความเร็วรองลงมา ส่วน SF แม้ว่าจะมีความเร็วในช่วงแรกสูง แต่เนื่องจากไม่สามารถรองรับการทำงานในสถานการณ์ที่มีการเชื่อมต่อเป็นช่วงๆได้จึงไม่เพิ่มค่าความเชื่อถือได้เมื่อเวลาผ่านไป EAEP สามารถทำงานได้ดีที่ความหนาแน่นสูงขึ้น แต่ยังพบในการทำงานที่ช้าเมื่อเทียบกับโพรโทคอลอื่นๆ และ AckPBSM ซึ่งมีปัญหาเรื่องการชนของสัญญาณมีความเร็วค่อนข้างต่ำที่ความหนาแน่นต่ำ และไม่สามารถทำงานได้เลยที่ความหนาแน่นสูง ดังนั้น DECA-bewa จึงมีความเร็วในการกระจายสูงและมีค่าใช้จ่ายในการทำงานน้อย ซึ่งเหมาะสมกับการทำงานในทุกสภาพแวดล้อมในทุกความหนาแน่นดังผลการทดลองที่ผ่านมา



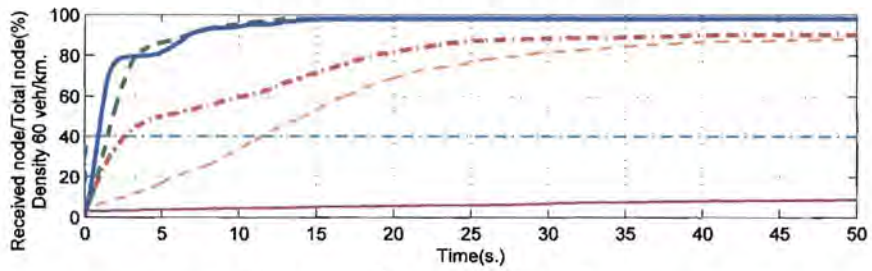
รูปที่ 3-24 ผลการทดลองค่าความเชื่อถือได้



รูปที่ 3-25 ผลการทดลองค่าใช้จ่าย



ก) การทดลองที่ความหนาแน่น 10 คัน/กม.



ข) การทดลองที่ความหนาแน่น 60 คัน/กม.

รูปที่ 3-26 ผลการทดลองความเร็วในการกระจายข้อมูล

บทที่ 4 การศึกษาผลกระทบ และการปรับปรุงโพรโทคอลการกระจายข้อมูลอย่างเชื่อถือได้ สำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะบนเครือข่ายจริง

เนื่องจากการทดสอบการทำงานของโพรโทคอลในบทที่ผ่านมาเป็นการทดสอบสมรรถนะโดยการใช้โปรแกรมจำลองในสภาพการจราจรเหมือนจริงบนแผนที่อย่างง่าย และแผนที่จริงของกรุงเทพมหานคร แต่เพื่อทราบถึงผลกระทบในการทำงานบนอุปกรณ์จริง และเครือข่าย เพื่อนำปัญหาที่เกิดขึ้นมาปรับปรุงแก้ไขให้โพรโทคอลทำงานได้มีประสิทธิภาพสูงที่สุด ในการทดสอบจะใช้โปรแกรมจำลองเครือข่าย NS-3 แทนการใช้งานโปรแกรมจำลองเครือข่าย NS-2 ซึ่งรองรับในการพัฒนาโพรโทคอลเพื่อใช้ทดสอบในอุปกรณ์จริง และเครือข่ายจริงได้

ในบทที่จะกล่าวถึงความแตกต่างระหว่างโปรแกรมจำลองทั้งสองแบบ ขั้นตอนการพัฒนาโพรโทคอลในโปรแกรมจำลองที่แตกต่างกัน ผลกระทบในการทดสอบบนสภาพแวดล้อมจริง และการทดสอบสมรรถนะของโพรโทคอลเมื่อได้รับการปรับปรุงขั้นตอนการทำงาน

4.1 ความแตกต่างระหว่างโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

โปรแกรมจำลองเครือข่าย NS-2 เป็นโปรแกรมจำลองเครือข่ายที่มีประสิทธิภาพ เริ่มมีการพัฒนาโครงการโปรแกรมจำลองเครือข่ายมาตั้งแต่ปี พ.ศ. 2532 ซึ่งพัฒนาต่อยอดมาจากระบบจำลองเครือข่าย REAL โดย ISI (Information Sciences Institute) และใช้กันอย่างแพร่หลายในงานวิจัยเกี่ยวกับเครือข่าย สังเกตได้จากการที่มีผู้ใช้มากกว่าครึ่งหนึ่งที่เลือกใช้โปรแกรมจำลองเครือข่าย NS-2 ในการทำวิจัยมีการตีพิมพ์ใน ACM และ IEEE ในระหว่างปี พ.ศ. 2543 และ พ.ศ. 2547

โปรแกรมจำลองเครือข่าย NS-2 เป็นโปรแกรมจำลองเครือข่ายการทำงานของระบบเครือข่ายแบบจำลองเหตุการณ์ไม่ต่อเนื่อง (Discrete Event Simulator) ซึ่งสนับสนุนการจำลองเครือข่ายมากมาย เช่น การเลือกเส้นทางในการขนส่งสินค้า จำลองการทำงานของโพรโทคอล เช่น UDP, TCP, RTP ทั้งที่อยู่บนเครือข่ายแบบมีสายและแบบไร้สาย อีกทั้งยังสนับสนุนโพรโทคอลแบบหลากหลาย (Multiple Protocol) สามารถแสดงรายละเอียดการจราจรของระบบเครือข่ายออกมาในรูปแบบของกราฟฟิก สนับสนุนกระบวนการในการหาเส้นทาง (Routing) และการลำดับข้อมูลต่างๆ ด้วย โปรแกรมจำลองเครือข่าย NS-2 เป็นโปรแกรมแบบโอเพ่นซอร์ส (Open Source) ซึ่งสามารถทำงานได้บนทั้ง Linux, FreeBSD, SunOS, Solaris และ Windows โดยจะใช้ภาษา C++ ในการเขียนส่วนของโปรแกรมที่เป็นการกำหนดกระบวนการต่างๆรวมถึงการสร้างโพรโทคอล และใช้ OTcl ในการจำลองสถานการณ์ขึ้น โดยกำหนดคุณลักษณะต่างๆ ซึ่งจะเขียนออกมาในรูปแบบของ TCL Script

แม้ว่าโปรแกรมจำลองเครือข่าย NS-2 เป็นโปรแกรมที่มีประสิทธิภาพในการทำงานสูง และได้รับความนิยมอย่างแพร่หลายก็ตาม ก็ยังคงมีปัญหาในการใช้งานอยู่หลายประการ เช่น

- การพัฒนาโปแกรมหรือโค๊ดเป็นแบบเก่า ไม่สามารถใช้งานได้กับการพัฒนาโปรแกรมหรือโค๊ดแบบใหม่ได้

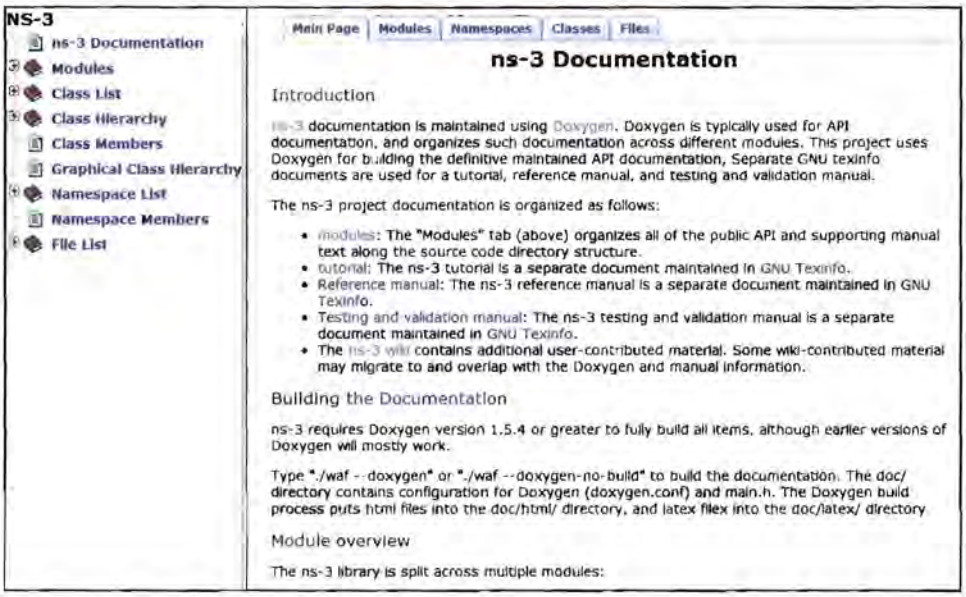
- ความสามารถในการรองรับจำนวนเครื่องในเครือข่ายไม่เพียงพอต่อความต้องการ
- การใช้งานระบบสืบข้อมูล (Tracing) มีความซับซ้อนใช้งานได้ค่อนข้างยาก

สาเหตุหลักที่โปรแกรมจำลองเครือข่าย NS-2 มีปัญหาด้านการใช้งานไม่สามารถตอบสนองการทดสอบที่ทันสมัยได้ คือ การที่โปรแกรมไม่ได้รับการดูแลจากผู้พัฒนาเป็นเวลานาน ซึ่งสังเกตได้จากการที่โปรแกรมจำลองเครือข่าย NS-2 มีการแก้ไขครั้งล่าสุด คือ เวอร์ชั่น 2.34 ในปี พ.ศ. 2552 ซึ่งห่างจากรุ่นก่อนหน้านั้นคือ เวอร์ชั่น 2.33 เป็นเวลาถึงสองปี และหยุดการพัฒนาและดูแลในปัจจุบัน ดังนั้นความนิยมในการใช้โปรแกรมจำลองเครือข่าย NS-2 ก็มีน้อยลงไปตามลำดับ

ในปี พ.ศ. 2549 ได้มีการเปิดตัวโปรแกรมจำลองเครือข่าย NS-3 ขึ้น โดยการพัฒนานั้นไม่ได้ต่อยอดจากโปรแกรมจำลองเครือข่าย NS-2 แต่ทำออกแบบและพัฒนาใหม่ทั้งหมด โดยที่มีมหาวิทยาลัย เช่น University of Washington, Georgia Tech University รวมถึงภาคเอกชนร่วมทุนในการวิจัยโปรแกรมจำลองเครือข่าย NS-3 ขึ้น เป้าหมายหลักของการพัฒนาคือการพัฒนาสภาพแวดล้อมการจำลองที่เป็นที่นิยมและเปิดกว้างสำหรับผู้วิจัยทางด้านเครือข่าย รูปแบบการทำงานที่น่าใจและแตกต่างจากโปรแกรมจำลองเครือข่าย NS-2 มีดังต่อไปนี้

1) แกนของซอฟต์แวร์สามารถเพิ่มเติมได้

โปรแกรมจำลองเครือข่าย NS-3 พัฒนาโดยใช้ภาษา C++ และมีภาษา Python เป็นส่วนต่อประสาน นอกจากนั้นยังมีเอกสารประกอบ API โดยใช้ Doxygen



รูปที่ 4-1 ตัวอย่างเอกสารประกอบ API ของโปรแกรมจำลองเครือข่ายเวอร์ชั่น 3 [23]

ผู้พัฒนาจะทำการดูแลในส่วนของแกนหลักซึ่งผู้ใช้ไม่สามารถเปลี่ยนแปลงได้ และส่วนของกลุ่มผู้ใช้งานจะทำหน้าที่ในการพัฒนาส่วนเสริมนอกเหนือจากที่ผู้พัฒนาดูแล โดยการออกแบบพยายามหลีกเลี่ยงปัญหาที่เกิดขึ้นในโปรแกรมจำลองเครือข่าย NS-2 เช่น ปัญหาการเข้ากันไม่ได้ระหว่างแบบจำลอง และการจัดการหน่วยความจำที่ทำได้ไม่ดีพอ

2) การพัฒนาที่มีความเสมือนจริงและสามารถใช้งานจริงได้

ปัญหาที่มีความสำคัญมากในการวิจัยหลักๆ คือ การที่งานวิจัยจำเป็นต้องผสมผสานการทำงานระหว่างการจำลองสถานการณ์ และการใช้งานจริงของเครือข่าย ซึ่ง NS-2 ไม่สามารถแสดงความเสมือนจริงได้เพียงพอ ส่งผลให้เป็นยากต่อการสังเกตพฤติกรรมของโปรแกรมเมื่อนำไปใช้จริง และยากในการเปรียบเทียบผลลัพธ์ที่ได้กับเหตุการณ์จริงในระบบเครือข่าย แต่โปรแกรมจำลอง NS-3 ได้รับการออกแบบการจำลองใหม่ให้คล้ายกับการทำงานบนระบบเครือข่ายจริงให้มากที่สุด เช่น การออกแบบโหนดมาเพื่อสอดคล้องกับคอมพิวเตอร์จริงๆ การใช้ระบบซอกเก็ต (Socket) และส่วนประสานผู้ใช้ด้วย IP (IP/Device Driver Interface) การนำกลับมาใช้ใหม่ของ Kernel และรหัสต้นฉบับต่างๆ ได้

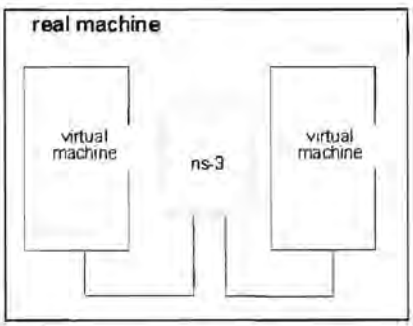
3) การบูรณาการของซอฟต์แวร์

ในส่วนนี้ปัญหาเดิมของการจำลองเครือข่าย คือ การสร้างโค๊ดและเครื่องมือขึ้นใหม่ ซึ่งมีเครื่องมือแบบโอเพ่น-ซอร์ส (Open source) ต่างๆที่สามารถนำมาใช้ได้อยู่แล้ว ดังนั้นโปรแกรมจำลองเครือข่าย NS-3 จึงรักษามาตรฐานของข้อมูลขาเข้า (Input) และข้อมูลขาออก (Output) เพื่อที่เครื่องมืออื่นๆสามารถใช้ข้อมูลร่วมกันได้ เช่น Pcap Trace Output, NS-2 Mobility Script เป็นต้น นอกจากนี้โปรแกรมจำลองเครือข่าย NS-3 ยังมีการเพิ่มส่วนช่วยเหลือในการทำงานของโค๊ด เช่น Network Simulator Cradle ที่สามารถทำงานได้กับ Linux TCP Code

4) รองรับการทำงานของการสร้างมโนภาพและการทดสอบโปรแกรม

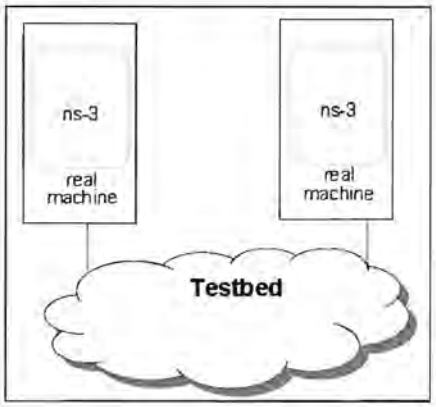
เนื่องจากนักวิจัยต้องการย้ายการพัฒนาระหว่างการจำลองไปเป็นการทดลองบนระบบจริง และยังไม่มีโปรแกรมจำลองเครือข่ายที่สนับสนุนนัก โปรแกรมจำลองเครือข่าย NS-3 ได้แก้ไขปัญหานี้โดยการสร้างรูปแบบการทำงานอีก 2 แบบ คือ

- การสร้าง Virtual Machine ให้ทำงานบนอุปกรณ์ และช่องสัญญาณของโปรแกรมจำลองเครือข่าย NS-3



รูปที่ 4-2 การจำลอง Virtual Machine เพื่อทดลองในระบบจริง [23]

- การใช้กองซ้อน (Stack) ในการทำงานแบบสถานการณ์จริง (Emulation) ซึ่งสามารถส่งข้อมูลได้บนระบบและอุปกรณ์จริง

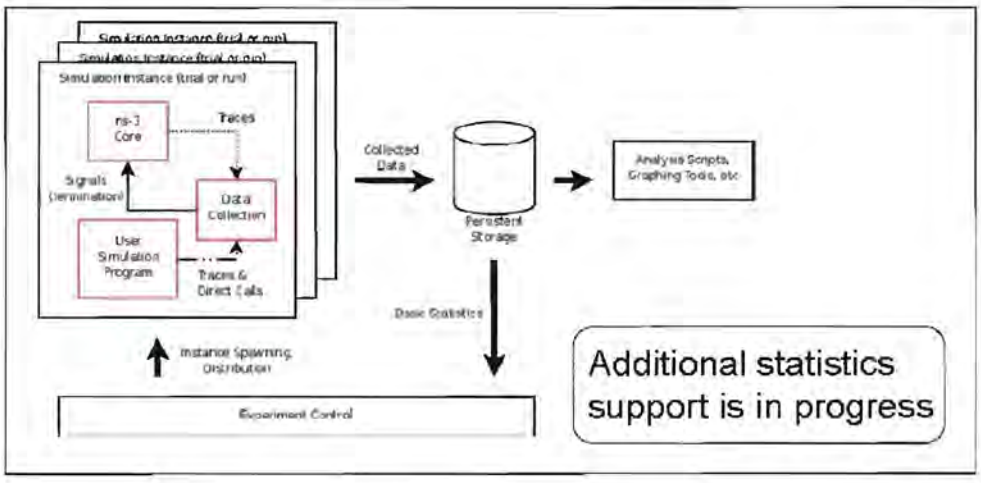


รูปที่ 4-3 การทำงานบนสถานการณ์และอุปกรณ์จริง [23]

5) มีระบบสืบข้อมูล (Tracing) และสถิติที่มีความยืดหยุ่นสูง

โปรแกรมจำลองเครือข่าย NS-3 ได้ทำการแยก Trace Source ออกจาก Trace Sink เพื่อให้ผู้ใช้งานสามารถปรับแต่ง Trace Sink ได้ โดยโปรแกรมจำลองเครือข่าย NS-3 ได้มีการสร้าง Trace Source ให้เลือกใช้มากมาย เช่น Packet Reception, State Machine Transition ระบบสืบข้อมูลยังรองรับการทำงานสถิติและการจัดข้อมูลโดยการทำงานเป็นดังนี้

- จัดการการทำงานหลายๆ สถานการณ์ที่ไม่เกี่ยวข้องกันได้
- จัดรูปแบบข้อมูลให้ออกมาในหลายรูปแบบของข้อมูลขาออก
- ทำงานเกี่ยวเนื่องกับ Trace Source
- ข้อมูลทางสถิติสามารถเชื่อมต่อกับการจำลองเครือข่ายได้ในขณะกำลังทำงานอยู่ เช่น หลักการทำงานของการจำลองเครือข่ายถ้าหากตัวนับ (Counter) ถึงค่าหนึ่งๆ



รูปที่ 4-4 ระบบสถิติของโปรแกรมจำลองเครือข่าย NS-3 [23]

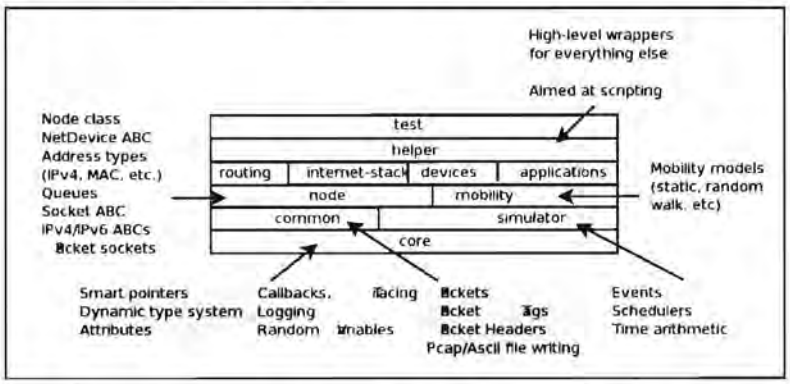
6) ระบบคุณลักษณะ (Attribute)

เนื่องจากนักวิจัยต้องการที่จะรู้ค่าที่เปลี่ยนแปลงในระหว่างการจำลองเครือข่ายและสามารถจัดการได้อย่างง่าย ในโปรแกรมจำลองเครือข่าย NS-3 จึงได้มีการสร้างชุดคุณลักษณะขึ้น โดยที่จะประกอบด้วย ชื่อและข้อความช่วยเหลือ ชนิด และค่าเริ่มต้น ซึ่งจะทำหน้าที่ดังนี้

- จัดการตัวแปรของการจำลองเครือข่ายสำหรับวัตถุนิ่ง (Static Object) ทุกตัว
- ทำการเก็บข้อมูล (Dump) และอ่านข้อมูลที่เก็บทั้งหมดในไฟล์การปรับค่า (Configuration)
- สร้างมโนภาพโดยใช้ส่วนประสานกราฟฟิกกับผู้ใช้ (Graphic User Interface) ทำให้ง่ายต่อการตรวจสอบตัวแปรของการจำลองเครือข่าย
- มีการสร้างเอกสารประกอบโดยใช้ Doxygen
- มีเครื่องมือสำหรับการตั้งค่าที่เป็นกราฟฟิก ดังรูปที่ 4-5

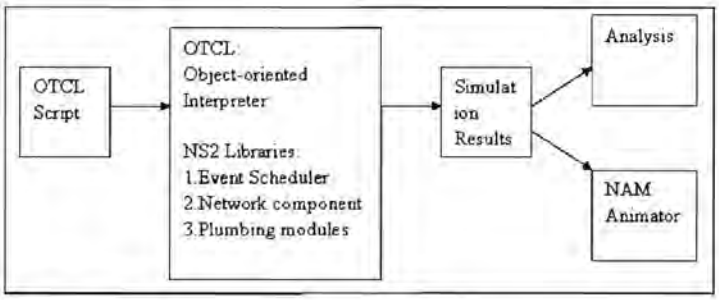
รูปที่ 4-5 ตัวอย่างกราฟฟิกการตั้งค่าในโปรแกรมจำลองเครือข่าย NS-3 [23]

นอกจากนั้นการทำงานของโปรแกรมจำลองเครือข่าย NS-3 ยังได้แบ่งการทำงานออกเป็นส่วนๆ (Module) แยกออกจากกันเพื่อให้ง่ายต่อการพัฒนา และสอดคล้องกับการทำงานในระบบเครือข่ายจริง ดังภาพที่ 4-6

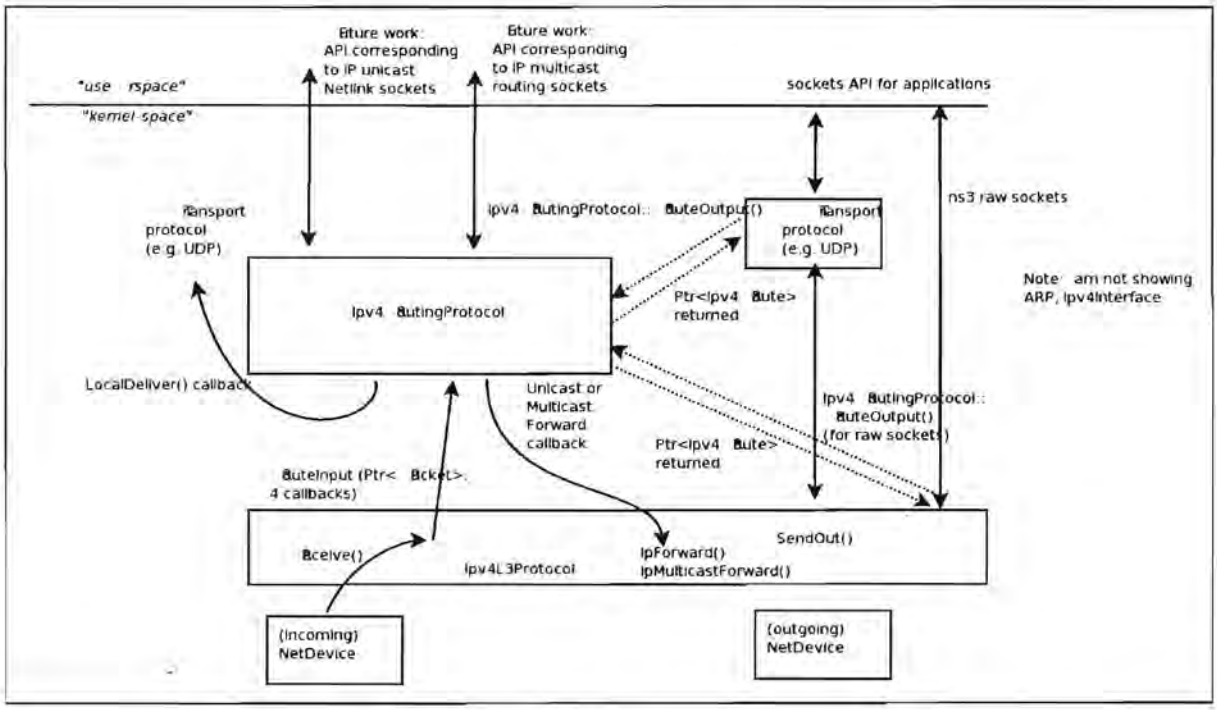


รูปที่ 4-6 รูปแบบของส่วนจำเพาะของโปรแกรมจำลองเครือข่าย NS-3 [24]

ดังที่กล่าวมาข้างต้นทำให้เห็นภาพได้ว่าการพัฒนาของโปรแกรมจำลองเครือข่ายทั้งสองเวอร์ชั่นนั้นแตกต่างกันออกไปโดยสิ้นเชิง เพราะการต้องการปรับเปลี่ยนระบบให้สอดคล้องกับการทำงานในระบบจริงมากขึ้น ดังนั้นผู้ใช้งานจึงต้องเข้าใจในการทำงานของโปรแกรมทั้งสองเพื่อความง่ายในการใช้งานโปรแกรมจำลองเครือข่ายในการพัฒนางานวิจัยของตน โดยสถาปัตยกรรมของแต่ละโปรแกรมเป็นดังภาพที่ 4-7 สำหรับโปรแกรมจำลองเครือข่าย NS-2 และภาพที่ 4-8 สำหรับโปรแกรมจำลองเครือข่าย NS-3



รูปที่ 4-7 สถาปัตยกรรมของโปรแกรมจำลองเครือข่าย NS-2 [21]



รูปที่ 4-8 สถาปัตยกรรมของโปรแกรมจำลองเครือข่าย NS-3 [23]

4.1.1 การพัฒนาโพรโทคอลบนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

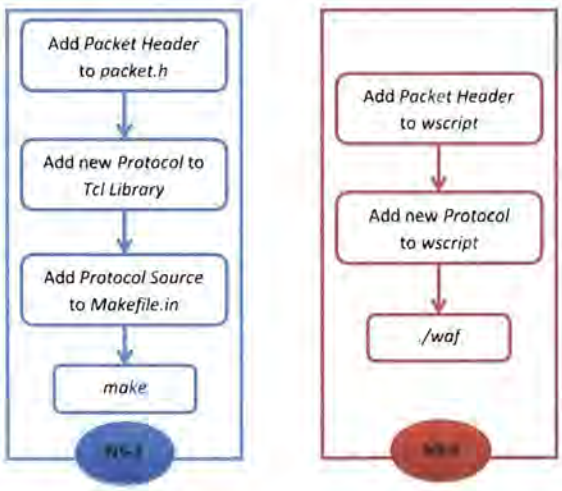
เมื่อศึกษาการทำงานของแต่ละโปรแกรมทั้งโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 และศึกษาความแตกต่างของโครงสร้างโปรแกรมแล้ว พบว่าโปรแกรมจำลองเครือข่าย NS-3 สามารถทำงานได้ทั้งการจำลองเครือข่ายและการทำงานบนระบบจริง ซึ่งแตกต่างจาก NS-2 ดังนั้นการเลือกใช้งาน NS-3 ในการทำงานวิจัยจึงง่ายต่อการศึกษาการทำงานใน

ระบบจริงของโพรโทคอล นอกจากนี้ยังมีงานวิจัย [24][25] ที่ได้ศึกษาการทำงานของโปรแกรมจำลองเครือข่ายแต่ละประเภท ซึ่งพบว่าโปรแกรมจำลองเครือข่าย NS-3 นั้นทำงานได้อย่างมีประสิทธิภาพมากที่สุด ทั้งในเชิงความเร็วของการทำงาน ความง่ายในการใช้งานของโพรโทคอล และความสามารถในการจัดการหน่วยความจำที่เป็นระบบมากขึ้น ดังนั้นการนำโพรโทคอลที่พัฒนาบน NS-2 ใช้งานบน NS-3 นั้นจำเป็นจะต้องมีการพัฒนาใหม่ มีหัวข้อที่น่าสนใจดังต่อไปนี้

1) การเพิ่มโพรโทคอลใหม่บนโปรแกรมจำลองเครือข่าย

การเพิ่มโพรโทคอลใหม่ในโปรแกรมจำลองเครือข่าย NS-2 เริ่มต้นจากการเพิ่ม Packet Header ของโพรโทคอลใหม่ที่ packet.h จากนั้นทำการเพิ่มโพรโทคอลใหม่เข้าสู่ Tcl Library และเพิ่ม Packet Source ที่ Makefile.in แล้วทำการสั่งคำสั่ง make ก็เป็นการสิ้นสุดการเพิ่มโพรโทคอลใหม่บนโปรแกรมจำลองเครือข่าย NS-2 ดังรูปที่ที่ 4-9 ด้านซ้าย

ส่วนโปรแกรมจำลองเครือข่าย NS-3 ที่มีขั้นตอนที่ต่างกันออกไป เริ่มจากการเพิ่ม Packet Header และโพรโทคอลใหม่ที่พัฒนาเข้าสู่ wscript ในแฟ้มข้อมูลของแต่ละส่วนของโปรแกรมนั้นๆ เพื่อทำให้วัตถุเหล่านั้นเชื่อมต่อเป็นส่วนหนึ่งกับ NS-3 Library หลังจากนั้นก็สั่ง ./waf ก็เสร็จการเพิ่มโพรโทคอล ขั้นตอนในความแตกต่างแสดงในรูปที่ 4-9 ด้านขวา

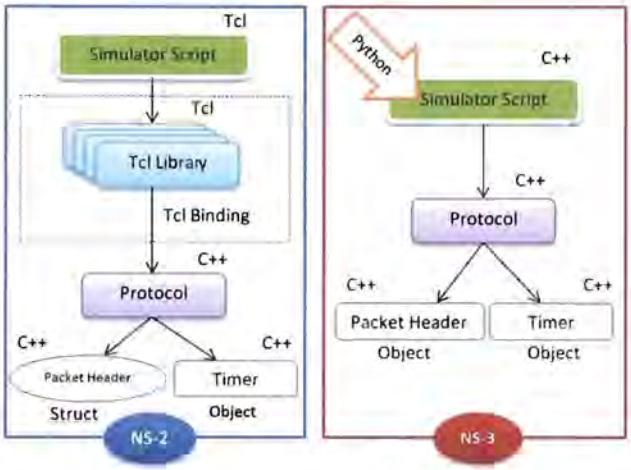


รูปที่ 4-9 การเพิ่มโพรโทคอลใหม่บนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

2) โครงสร้างการทำงานพื้นฐานของโปรแกรมของโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

ในโปรแกรมจำลองเครือข่าย NS-2 เมื่อทำการสั่งให้โปรแกรมทำงานทาง Tcl Script ก็จะมีการสร้าง Tcl Binding ไปสู่โพรโทคอลที่พัฒนาด้วยภาษา C++ ซึ่งงานวิจัยนี้เน้นที่โพรโทคอลในชั้นเครือข่าย (Network Layer) ซึ่งโพรโทคอลจะทำการเรียก Packet Header ที่เป็น Struct รวมทั้งการเรียกใช้ Timer ที่เป็นวัตถุได้ เป็นต้น

ส่วนโปรแกรมจำลองเครือข่าย NS-3 หลังจากสั่งคำสั่ง ./waf ในส่วนของ Simulator Script ที่ถูกพัฒนาด้วย C++ ทั้งหมดก็จะเรียกใช้โพรโทคอลได้โดยตรง ไม่ต้องมี Tcl Binding จากนั้นโพรโทคอลก็เรียกใช้ Packet Header และ Timer ที่เป็นวัตถุทั้งคู่ได้ตามปกติ ความแตกต่างของทั้งสองโปรแกรมนั้นแสดงในรูปที่ 4-10



รูปที่ 4-10 โครงสร้างการทำงานพื้นฐานของโปรแกรมของทั้งสองโปรแกรม

4.1.2 การสร้างรูปแบบการเคลื่อนที่ของยานพาหนะ

โปรแกรมจำลองเครือข่าย NS-3 นั้นสามารถเรียกใช้งานไฟล์รูปแบบการเคลื่อนที่ (Mobility trace file) ซึ่งเป็นไฟล์ที่สร้างรูปแบบการเคลื่อนที่ของยานพาหนะที่ใช้สำหรับโปรแกรมจำลองเครือข่าย NS-2 ได้ โดยเรียกใช้งานฟังก์ชั่น NS2MobilityHelper ที่จะทำการเปลี่ยนรูปแบบของการเคลื่อนที่สำหรับโปรแกรม NS-2 ให้กลายเป็นรูปแบบของโปรแกรม NS-3 ซึ่งใน NS-2 นั้นโหนดจะเคลื่อนที่ตามค่าเวกเตอร์ความเร็วที่ถูกกำหนดไว้ แต่ NS-3 โหนดจะใช้ความแตกต่างของอัตราเร็วในแต่ละแกน x, y และ z แทนเวกเตอร์ความเร็ว เมื่อทำการทดสอบการทำงานของโพรโทคอลด้วยสภาพแวดล้อมรูปแบบถนนทางหลวง (Highway) บน NS-3 โดยการเรียกใช้ไฟล์รูปแบบการเคลื่อนที่ดังกล่าว พบว่าประสิทธิภาพที่ออกมานั้นแตกต่างจากผลทดลองบนโปรแกรม NS-2 โดยพบว่ามีความคลาดเคลื่อนในการเคลื่อนที่ของโหนด โดยมีความแตกต่างประมาณ 6-7 % ซึ่งตำแหน่งการเคลื่อนที่ที่แตกต่างกันนั้นมีระยะความคลาดเคลื่อนสูงสุดถึง 200 เมตร ซึ่งผลให้เกิดความแตกต่างของผลการทดลอง จากการตรวจสอบพบว่าปัญหาเกิดจากการแปลง Trace File ของ Ns2MobilityHelper ในโปรแกรมจำลองเครือข่าย NS-3 ซึ่งมีการกำหนดให้โหนดมีการหยุดการเคลื่อนในช่วงเวลาสั้นๆเมื่อโหนดมีการเปลี่ยนทิศทางหรือความเร็วและไม่สามารถสังเกตได้จากส่วนแสดงผลแบบกราฟฟิก ทำให้พฤติกรรมการเคลื่อนที่โดยรวมนั้นแตกต่างจากโปรแกรมจำลองเครือข่าย NS-2 การแก้ปัญหานี้จึงทำได้ด้วยการเปลี่ยนให้โหนดเคลื่อนที่อย่างต่อเนื่องไม่มีการหยุดในช่วงเวลาสั้นๆ ก่อนการเปลี่ยนทิศทางและความเร็ว โดยการยกเลิกการหยุดใน Ns2MobilityHelper.cc อยู่ในบรรทัดที่ 614 (NS-3.10) ดังแสดงในรูปที่ 4-11

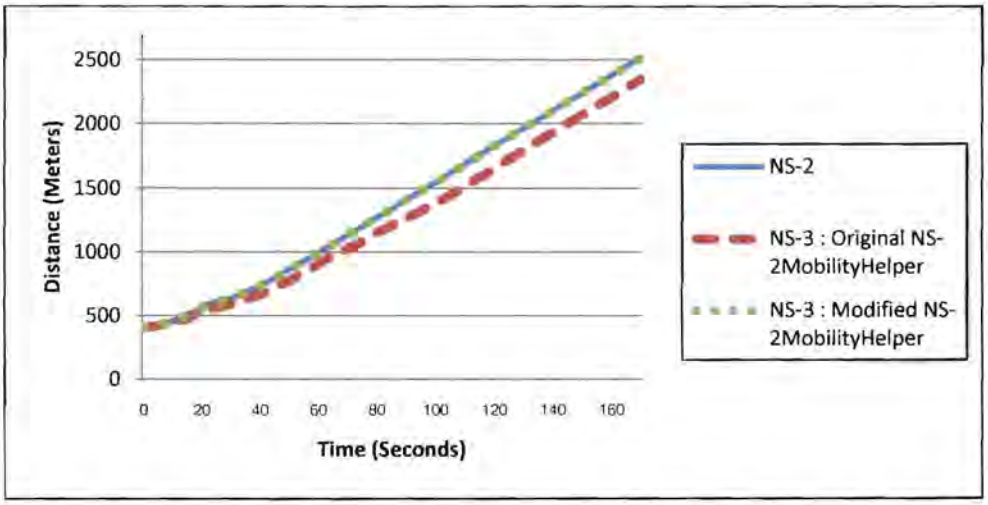
```
if (time >= 0)
  {
    Simulator::Schedule (Seconds (at + time),
    &ConstantVelocityMobilityModel::SetVelocity,model, Vector (0, 0, 0));
  }
```

รูปที่ 4-11 ส่วนของโปรแกรมที่ทำให้เกิดการหยุดของโหนดในการเคลื่อนที่ใน NS-3

การทดลองการทำงานของโพรโทคอลหลังจากแก้ไขเรื่องการเคลื่อนที่ใช้โปรแกรมจำลองการจราจร Simulation of Urban and Mobility (SUMO) ในการสร้างสถานการณ์ในการเคลื่อนที่ของโหนดและเปลี่ยนเป็นรูปแบบที่โปรแกรมจำลองเครือข่าย NS-2 ใช้งานด้วยโปรแกรม Traffic and Network Simulation Environment (TraNS) โดยใช้สภาพแวดล้อมรูปแบบถนนทางหลวง ซึ่งเป็นเส้นทาง 2 เลน ความยาว 4 กิโลเมตร และแสดงการเคลื่อนที่ของทุกโหนดบนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 ด้วยความหนาแน่น 30 ยานพาหนะต่อกิโลเมตร ดังรูปที่ 4-12 ส่วนตารางที่ 4-1 แสดงค่าเฉลี่ยในการเคลื่อนที่ของทุกโหนดเปรียบเทียบระหว่างโปรแกรมจำลองเครือข่าย NS-2 กับ NS-3 ซึ่งแสดงค่าว่าเมื่อทำการปรับเปลี่ยนการเคลื่อนที่ให้ไม่มีการหยุดในการเคลื่อนที่แล้ว ค่าผิดพลาดที่เกิดขึ้นมีน้อยมากทำให้การเคลื่อนที่ใกล้เคียงกับการเคลื่อนที่บนโปรแกรมจำลองเครือข่าย NS-2

ตารางที่ 4-1 ค่าความผิดพลาดในการเคลื่อนที่เฉลี่ยระหว่าง NS-2 และ NS-3 (%)

	ความหนาแน่นของยานพาหนะ (คัน/กิโลเมตร)				
	2	10	30	60	120
Ns2MobilityHelper NS-3 เดิม	6.6853	6.1845	6.1958	7.8375	7.4393
Ns2MobilityHelper NS-3 ปรับปรุง	0.0013	0.0009	0.0019	0.0036	0.3843



รูปที่ 4-12 ระยะการเคลื่อนที่ของโหนดในโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

4.1.3 การเปรียบเทียบการทำงานของโพรโทคอลบนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3

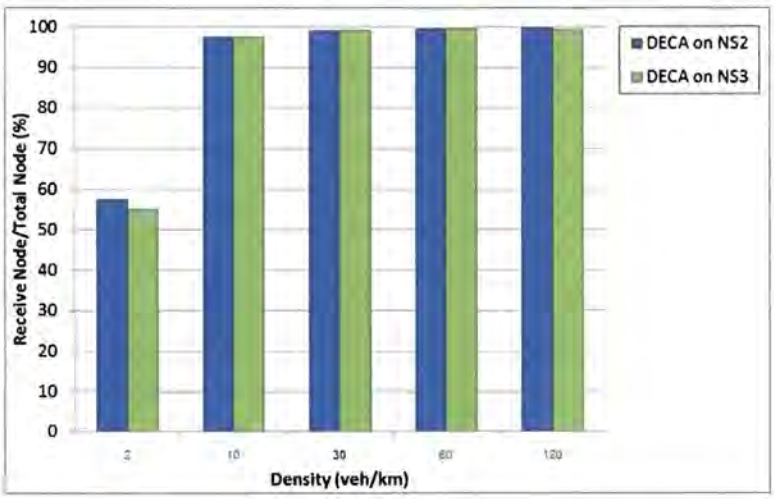
เมื่อโปรแกรมจำลองเครือข่าย NS-3 สามารถเรียกใช้ไฟล์รูปแบบการเคลื่อนที่ได้อย่างสมบูรณ์ ขั้นต่อไปเป็นการเปรียบเทียบประสิทธิภาพการทำงานของโพรโทคอล DECA บนโปรแกรมจำลองเครือข่าย NS-2 และ NS-3 เพื่อเป็นการตรวจสอบสมรรถนะของการพัฒนาโพรโทคอลให้สามารถทำงานบนโปรแกรมจำลองทั้งสองโดยไม่มีความแตกต่างกัน เพื่อนำไปสู่การทดลองในงานวิจัยในขั้นถัดไป การวัดสมรรถนะทำการทดลองบนสภาพแวดล้อมรูปแบบถนนทางหลวงเส้นตรงยาว 4 กิโลเมตร ซึ่งสร้างโดย SUMO และเปลี่ยนให้เป็นรูปแบบที่ใช้งานกับโปรแกรมจำลองเครือข่าย NS-2 ด้วย TraNS เป็นตัวกำหนดการเคลื่อนที่ของโหนดในการทดลอง ดังเช่นการทดสอบในบทที่ 2 และบนโปรแกรมจำลองเครือข่าย

NS-3 ได้มีการปรับเปลี่ยนฟังก์ชั่นการเคลื่อนที่ของโหนดดังที่กล่าวข้างต้นเพื่อให้มีการเคลื่อนที่เช่นเดียวกับโปรแกรมจำลองเครือข่าย NS-2 ในการทดลองแต่ละโหนดมีระยะในการส่งข้อมูล 250 เมตร และมีการตั้งค่าต่างๆในการทดลอง ดังตารางที่ 4-2

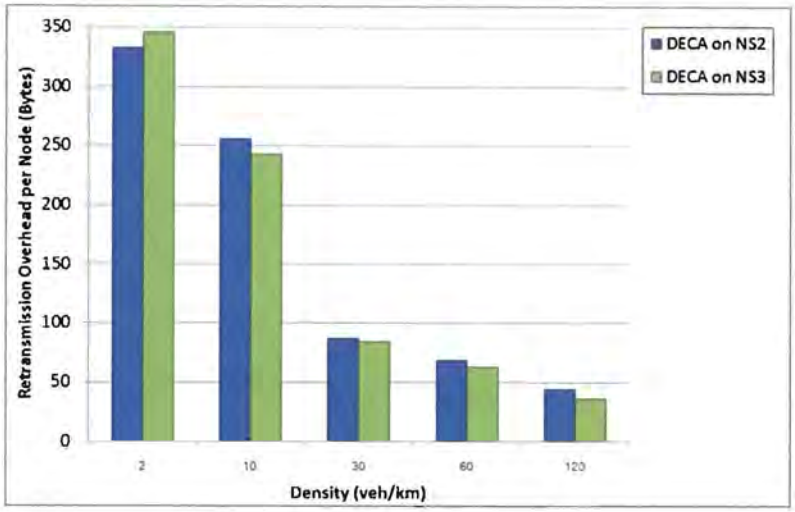
ตารางที่ 4-2 การตั้งค่าเพื่อวัดสมรรถนะของโพรโทคอล DECA

อายุของข้อความ	10 วินาที
จำนวนข้อความ	5
ขนาดของข้อความ	512 ไบต์
เวลาในการทดลอง	50 วินาที
ความเร็วสูงสุดของยานพาหนะ (กิโลเมตร/ชั่วโมง)	50, 80
ความหนาแน่นของยานพาหนะ(คัน/กิโลเมตร)	ถนนทางหลวง (Highway) 2, 10, 30, 60, 120
โปรแกรมจำลองเครือข่าย	NS-2.34, NS-3.10

การทำงานนั้นโหนดจะเริ่มต้นส่งข้อความทุกๆ 10 วินาที จำนวน 5 ข้อความ ผลลัพธ์ที่ได้เป็นค่าเฉลี่ยที่เกิดขึ้นจากทั้ง 5 ข้อความ ดังรูปที่ 4-13 เป็นค่าความเชื่อถือได้ (Reliability) ระหว่างการทดสอบโพรโทคอล DECA บนโปรแกรมจำลองเครือข่าย NS-2 กับ NS-3 โดยในแกน x แทนความหนาแน่นของยานพาหนะต่อกิโลเมตร และแกน y แทนค่าความเชื่อถือได้เป็นเปอร์เซ็นต์ ซึ่งพบว่าผลที่ออกมามีค่าใกล้เคียงกัน ส่วนรูปที่ 4-14 แสดงผลค่าใช้จ่ายของโพรโทคอลทั้งสองโปรแกรม แกน x นั้นยังแทนความหนาแน่นของยานพาหนะ ส่วนแกน y นั้นแทนค่าใช้จ่ายของโพรโทคอล ผลลัพธ์ที่ได้ออกมานั้นมีค่าใกล้เคียงกันทั้งสองโปรแกรม ดังนั้นสรุปว่าประสิทธิภาพการทำงานของโพรโทคอลบนโปรแกรมจำลองเครือข่าย NS-2 กับ NS-3 นั้นมีค่าเท่ากัน และมีพฤติกรรมในการทำงานเหมือนกัน



รูปที่ 4-13 ผลการทดลองค่าความเชื่อถือได้

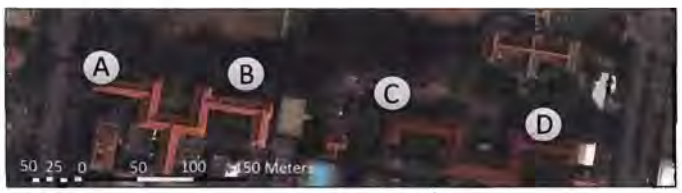


รูปที่ 4-14 ผลการทดลองค่าใช้จ่าย

4.2 การทดสอบผลกระทบในการทำงานของโพรโทคอลบนเครือข่ายจริง

การทดลองการทำงานบนระบบจริงของโพรโทคอล DECA ด้วยโปรแกรมจำลองเครือข่าย NS-3 ดังรูปที่ 4-15 ซึ่งจัดวางคอมพิวเตอร์พกพาจำนวนสี่เครื่องบนตำแหน่งที่แตกต่างกันภายในจุฬาลงกรณ์มหาวิทยาลัย กำหนดให้มีสถานการณ์สองแบบ คือ

- สถานการณ์ที่มีการเชื่อมต่อคงที่ : ให้ทุกเครื่องอยู่นิ่งในตำแหน่งของแต่ละเครื่อง แล้วให้ทุกเครื่องทำงานโพรโทคอล DECA พร้อมกันและส่งข้อมูลหากันและกัน
- สถานการณ์ที่มีการเชื่อมต่อเป็นช่วงๆ : ให้เครื่อง A, C และ D เคลื่อนที่ไปในทิศทางที่กำหนด แต่ B ยังอยู่นิ่งแล้วทำการส่งข้อมูลหากัน



ก) รูปแบบโหนดอยู่นิ่ง

ข) รูปแบบโหนดเคลื่อนที่

รูปที่ 4-15 การทดสอบบนเครือข่ายจริงในบริเวณจุฬาลงกรณ์มหาวิทยาลัย

การทดสอบทั้งสองสถานการณ์มีโหนด A เป็นโหนดต้นทาง (Source) ผลการทดลองค่าความเชื่อถือได้ของโพรโทคอลมีค่า 100 % เนื่องจากทุกเครื่องได้รับข้อมูลที่ส่งออกมาจากโหนดต้นทางทั้งหมด แต่ผลค่าใช้จ่ายในการทำงานของโพรโทคอล

นั้นมีค่าสูงมาก ซึ่งแตกต่างจากการทดลองบนครือข่ายจำลองถึง 5 เท่า ทำให้เกิดปัญหาขึ้นในการทำงานของโพรโทคอลบนระบบจริง หลังจากนั้นได้ทำการทดลองและตรวจสอบถึงสาเหตุที่เกิดขึ้นทำให้พบว่าเกิดปัญหาการเชื่อมต่อแบบอสมมาตร เนื่องจากในการทดลองนั้นคอมพิวเตอร์พกพาแต่ละเครื่องมีความสามารถในการส่งข้อมูลของอุปกรณ์รับส่งสัญญาณไร้สายที่แตกต่างกัน ทำให้บางเครื่องมีความแรงของสัญญาณสูงซึ่งสามารถส่งข้อมูลได้ไกล แต่บางเครื่องมีความแรงของสัญญาณต่ำทำให้ไม่สามารถส่งข้อมูลไปยังเครื่องอื่นได้ ทำได้เพียงรับข้อมูลจากเครื่องอื่น ซึ่งสาเหตุนี้ส่งผลกับการทำงานของโพร-โทคอล DECA อย่างมาก เพราะเครื่องที่มีความแรงของสัญญาณต่ำ สามารถส่งข้อมูลไปยังผู้อื่นได้ยาก ปัญหาจะเกิดขึ้นเมื่อโหนดนี้ได้รับข้อมูลมาแล้วไม่สามารถทำการส่งข้อมูลต่อได้ ทำให้โหนดที่ไม่ได้รับข้อมูลจะส่งบีคอนออกมาเพื่อร้องขอข้อมูลนั้น เมื่อโหนดที่มีความแรงของสัญญาณต่ำได้รับบีคอนก็จะพยายามส่งข้อมูลนั้นออกไปให้ แต่โหนดที่ร้องขอก็จะไม่ได้รับข้อมูลอยู่ดี เนื่องจากสัญญาณที่ส่งไปนั้นไม่ถึงโหนดที่มีการร้องขอ ทำให้ค่าใช้จ่ายในการทำงานของระบบมีมากขึ้น และเหตุการณ์จะเกิดวนซ้ำจนกว่าจะมีโหนดอื่นที่สามารถส่งข้อมูลแทนโหนดที่มีระยะส่งข้อมูลสั้นได้ ซึ่งเกิดการชนกันของข้อมูลมากขึ้น ส่งผลให้ประสิทธิภาพการทำงานของโพรโทคอลต่ำลง

จากปัญหาที่เกิดขึ้นดังกล่าวทำให้ทราบว่าการทำงานของโพรโทคอล DECA นั้นไม่สามารถทำงานได้ด้วยสาเหตุคือการส่งข้อมูลนั้นส่งไม่สำเร็จ เพราะในสถานการณ์การเชื่อมต่อแบบอสมมาตรนั้นนอกจากปัญหาจะเกิดจากปัจจัยภายในคือความสามารถของอุปกรณ์รับส่งสัญญาณไร้สายที่มีความแรงของสัญญาณต่ำแล้ว ยังเกิดจากปัจจัยภายนอกด้วย อย่างเช่น สัญญาณที่ส่งออกไปนั้นอาจจะถูกเบี่ยงเบน แทรกสอด หรือดูดซับ ทำให้การส่งข้อมูลนั้นทำได้ไม่สำเร็จ ส่วนสาเหตุที่สองคือการเลือกผู้กระจายข้อมูลลำดับถัดไปที่ไม่มีประสิทธิภาพเพียงพอ เพราะผู้ถูกเลือกอาจจะเป็นโหนดที่มีความแรงของสัญญาณต่ำทำให้การส่งข้อมูลไปให้กับเพื่อนบ้านโดยรอบไม่สำเร็จ ดังรูปที่ 4-16 ส่งผลให้เกิดปัญหาการไม่ได้รับข้อมูลและค่าใช้จ่ายในระบบที่สูง

ก) ผู้ส่งข้อมูลผิดพลาด

ข) ผู้ถูกเลือกในการกระจายข้อมูลผิดพลาด

รูปที่ 4-16 ความผิดพลาดที่เกิดขึ้นในรูปแบบการเชื่อมต่อแบบอสมมาตร

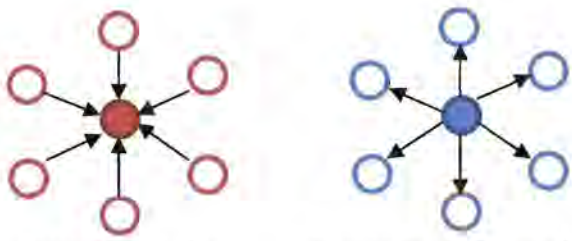
ดังนั้นในสถานการณ์การเชื่อมต่อแบบอสมมาตรการที่ผู้ส่งจะส่งข้อมูลให้กับโหนดเพื่อนบ้านส่วนใหญ่นั้นเป็นไปได้ยาก เพราะความแรงของสัญญาณในการส่งนั้นมีความแตกต่างกันไปในแต่ละโหนด หรืออาจเกิดขึ้นเพราะปัจจัยภายนอก ในบทที่จึงเน้นในการปรับปรุงกระบวนการในการเลือกโหนดส่งต่อข้อความ โดยมีแนวคิดว่าการเลือกโหนดที่มีความแรงของสัญญาณสูงน่าจะทำให้มีระยะในการส่งข้อมูลไกลและครอบคลุมการเชื่อมต่อกับเพื่อนบ้านมากที่สุด และจะทำให้การกระจายข้อมูลไปถึงโหนดเพื่อนบ้านได้มากกว่า

4.3 หลักการทำงานของขั้นตอนการเลือกโหนดส่งต่อข้อความแบบโหวตค่า RSSI

4.3.1 แนวคิดในการปรับปรุงขั้นตอนการเลือกโหนดส่งต่อข้อความ

ขั้นตอนในการเลือกโหนดส่งต่อข้อความไปนั้นมีแนวคิดในการพัฒนามาจากหลักการทำงานเดิมของโพรโทคอล DECA ในการเลือกโหนดดังภาพที่ 4-17 ก) โดยการเลือกโหนดเดิมนั้นจะเลือกโหนดที่ได้รับบีคอนจากเพื่อนบ้านจำนวนมากที่สุด ซึ่งจะถือว่าโหนดนั้นมีความหนาแน่นของเพื่อนบ้านมากที่สุดทำให้โหนดนั้นถูกเลือกให้เป็นโหนดส่งต่อข้อความ แต่ในสถานการณ์ที่มีการเชื่อมต่อแบบอสมมาตร ความหนาแน่นจากการรับบีคอนไม่สามารถยืนยันได้ว่าโหนดที่ถูกเลือกจะสามารถส่งข้อมูลไปให้กับเพื่อนบ้านที่เชื่อมต่อได้สำเร็จ เพราะว่าโหนดที่ถูกเลือกนั้นทำได้แค่เพียงรับบีคอนมาจากเพื่อนบ้าน ซึ่งโหนดที่ถูกเลือกอาจจะมีความแรงของสัญญาณต่ำ ทำให้การส่งข้อมูลไปไม่ถึงเพื่อนบ้านส่วนใหญ่ ด้วยเหตุนี้ทำให้ DECA ไม่สามารถทำงานได้อย่างมีประสิทธิภาพในสถานการณ์การเชื่อมต่อแบบอสมมาตร

ในกรณีนี้หากมองในมุมตรงข้าม การเลือกโหนดที่สามารถส่งข้อมูลให้กับเพื่อนบ้านได้จริง จึงเป็นทางเลือกที่ดีกว่าในการทำงานของโพรโทคอล ดังภาพที่ 4-17 ข) โดยมีหลักการว่าโหนดที่สามารถส่งบีคอนให้กับเพื่อนบ้านได้มากที่สุดก็ถือว่าโหนดนั้นจะสามารถส่งข้อมูลไปให้กับเพื่อนบ้านได้มากที่สุดเช่นกัน ซึ่งการทำงานคือโหนดนั้นจะส่งบีคอนไปให้กับเพื่อนบ้านโดยรอบ และหากเพื่อนบ้านส่วนใหญ่ในละแวกนั้นทำการโหวตออกมาว่าโหนดที่ส่งบีคอนมานั้นมีความแรงของสัญญาณมาก โหนดที่ได้รับการโหวตจากเพื่อนบ้ายมากที่สุดจะได้รับการเลือกเป็นโหนดที่จะทำการกระจายข้อความลำดับถัดไป



ก) DECA แบบเดิม ข) กระบวนการเลือกแบบใหม่

รูปที่ 4-17 กระบวนการในการเลือกโหนด (ลูกศรแทนทิศทางการส่งบีคอนสำเร็จ)

นอกจากกระบวนการโหวตของเพื่อนบ้านแล้วการแก้ไขปัญหาในการเลือกของโพรโทคอลนั้นยังใช้งานควบคู่กับการค่าวัดความแรงสัญญาณที่ได้รับหรือค่า RSSI โดยค่านี้จะเป็นตัวแปรหลักที่ใช้ในการเลือกผู้ส่งสัญญาณที่แรงที่สุด

เพื่อที่จะเป็นโหนดส่งต่อข้อความที่ใช้ควบคู่กับการโหวตอีกด้วย จึงได้กระบวนการเลือกโหนดส่งต่อข้อความแบบใหม่ที่เรียกว่า ขั้นตอนการโหวตแบบอาร์เอสเอสไอ (RVA : RSSI-Voting Algorithm)

4.3.2 หลักการทำงานการเลือกโหนดส่งต่อแบบ RVA

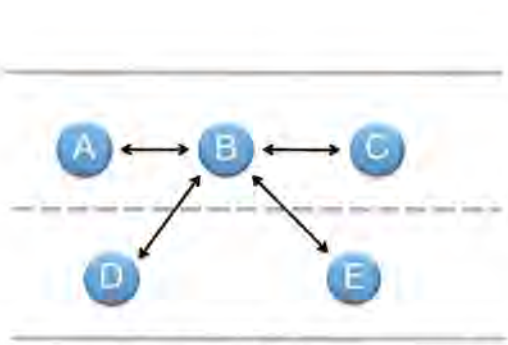
การปรับปรุงการทำงานของ DECA ในสถานการณ์การเชื่อมต่อแบบอสมมาตรเน้นไปที่การแก้ไขกระบวนการเลือกโหนดส่งต่อข้อความดังที่กล่าวข้างต้น โดยกระบวนการอื่นๆ ในการทำงานของโพรโทคอลนั้นยังคงการทำงานในส่วนหลักๆ ไว้ อย่างเช่น การแลกเปลี่ยนบีคอนเพื่อปรับข้อมูลของเพื่อนบ้านให้ทันสมัย การตั้งเวลารอของโหนด เป็นต้น

หลักการทำงานของกระบวนการที่แก้ไขจากการเลือกโหนดส่งต่อข้อความเดิม

1) แต่ละโหนดทำการโหวตเพื่อนบ้านของตนว่าโหนดไหนมีค่า RSSI มากที่สุด (RSSI ได้มาจากการแลกเปลี่ยนบีคอนครั้งก่อนหน้า) แล้วแนบผลโหวตไปกับบีคอน โดยโหนดจะสามารถโหวตได้เพียงแค่โหนดเพื่อนบ้านที่มีค่า RSSI สูงที่สุดเพียงโหนดเดียว
2) เมื่อโหนดได้รับบีคอนมาจากเพื่อนบ้านก็จะเก็บข้อมูลของเพื่อนบ้านพร้อมผลโหวตไว้ในตารางเพื่อนบ้านของตน
3) หากโหนดใดต้องการส่งข้อมูลจะทำค้นหาจากตารางโหนดเพื่อนบ้านเพื่อหาโหนดที่ถูกโหวตจากโหนดเพื่อนบ้านอื่นๆมากที่สุด และโหนดนั้นต้องเป็นโหนดเพื่อนบ้านของตน โหนดที่ได้รับการโหวตมากที่สุดจะถูกเลือกเป็นโหนดส่งต่อข้อความ

ตัวอย่างในการทำงาน

1) โหนดจะแลกเปลี่ยนบีคอนของตนเองกับเพื่อนบ้าน ตัวอย่างด้านล่างในรูปที่ 4-18 เป็นตัวอย่างตารางเพื่อนบ้านของโหนด A ซึ่งพบว่าเพื่อนบ้านของ A ประกอบด้วย B กับ D ซึ่ง A พบว่า D มีค่า RSSI มากที่สุดจากการแลกเปลี่ยนบีคอนครั้งก่อนหน้านี้ ทำให้ A เลือกที่จะโหวต D แนบไปกับการบีคอนครั้งต่อไป

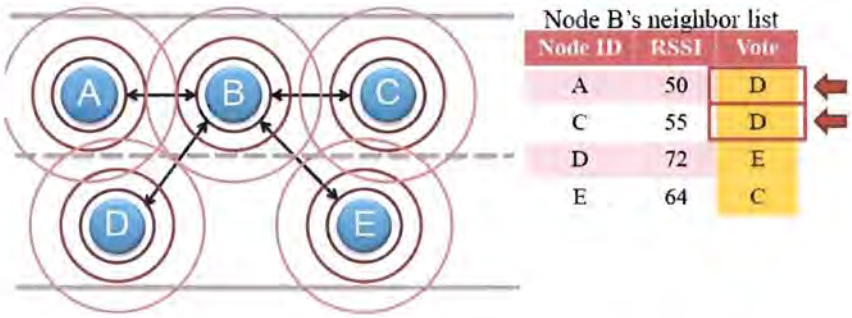


Node A's neighbor list

Node ID	RSSI
B	60
D	71

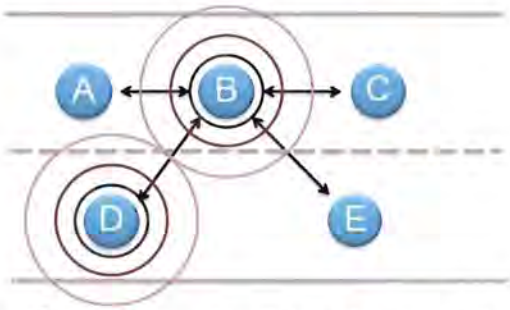
รูปที่ 4-18 ขั้นตอนที่หนึ่งของขั้นตอนการโหวตแบบอาร์เอสเอสไอ

2) จากนั้นทุกโหนดทำการแลกเปลี่ยนบีคอนกัน และแก้ไขข้อมูลของเพื่อนบ้านของตนในตารางเพื่อนบ้านให้ทันสมัย จากตัวอย่างด้านล่างในรูปที่ 4-19 เป็นตัวอย่างของตารางเพื่อนบ้านของโหนด B ซึ่งได้รับข้อมูลเพื่อนบ้านของคนมาและแก้ไขให้ทันสมัยแล้ว ซึ่งในตารางจะมีการระบุค่า RSSI ของเพื่อนบ้านแต่ละโหนด นอกจากนั้นยังมีข้อมูลว่าเพื่อนบ้านของตนนั้นทำการโหวตโหนดใดว่ามีความแรงของสัญญาณมากที่สุด ซึ่ง B พบว่าโหนด D นั้นมีผลโหวตจากเพื่อนบ้านมากที่สุดถึง 50 % ของการโหวตจากเพื่อนบ้านทั้งหมด



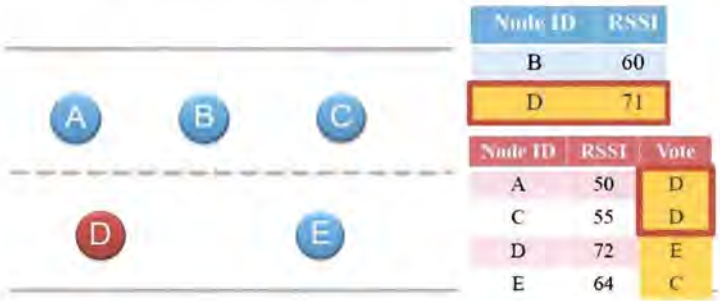
รูปที่ 4-19 ขั้นตอนที่สองของขั้นตอนการโหวตแบบอาร์เอสเอสไอ

3) ดังนั้นหากโหนด B ต้องการกระจายข้อมูลไปให้กับเพื่อนบ้านทั้งหมด ก็จะทำการเลือกให้โหนด D เป็นผู้กระจายข้อมูลลำดับถัดไป ดังตัวอย่างด้านล่างต่อไปนี้ในรูปที่ 4-20



รูปที่ 4-20 ขั้นตอนที่สามของอัลกอริทึมการโหวตแบบอาร์เอสเอสไอ

ดังนั้นสรุปได้ว่าโหนดที่มีความแรงของสัญญาณของผู้ส่งมากที่สุด (ค่า RSSI มากที่สุด) และได้รับการโหวตจากเพื่อนบ้านมากที่สุดจะถูกเลือกให้เป็นโหนดส่งต่อข้อความ ดังตัวอย่างด้านล่างต่อไปนี้ดังรูปที่ 4-21



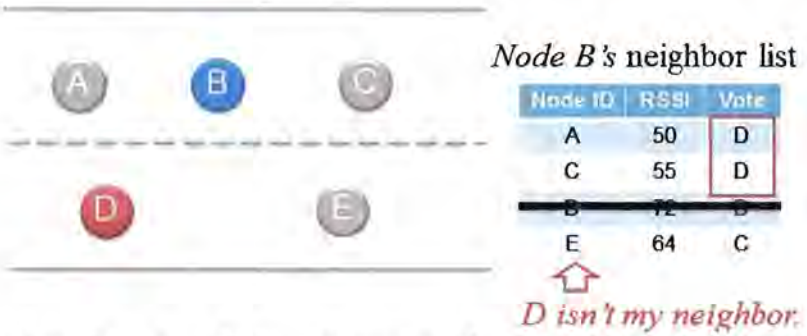
รูปที่ 4-21 สรุปผลการเลือกโหนดของขั้นตอนการโหวตแบบอาร์เอสเอสไอ

4.3.3 ปัญหาที่เกิดขึ้นจากขั้นตอนการโหวตอาร์เอสเอสไอ และวิธีแก้ไข

1) โหนดที่ได้รับผลโหวตอันดับหนึ่งไม่ใช่เพื่อนบ้านของผู้ส่งข้อมูล

จากการทดลองบนสถานการณ์ที่มีจำนวนยานพาหนะมากพบว่าโหนดที่ถูกเลือกนั้นจะได้รับการโหวตเป็นอัตราส่วนประมาณ 50-80 % จากการโหวตของเพื่อนบ้านทั้งหมด ดังนั้นถ้าโหนดที่ได้รับการโหวตนี้ไม่ได้เป็นเพื่อนบ้าน

กับผู้ส่งข้อมูล ดังรูปที่ 4-22 โหนด D เป็นโหนดที่ได้รับการโหวตมากที่สุด แต่โหนด D นั้นไม่ได้เป็นเพื่อนบ้านกับโหนด B ซึ่งเป็นผู้ส่งข้อมูล ทำให้โหนดอื่นที่ได้รับการโหวตในอันดับต่อๆมาจากเพื่อนบ้านจะคิดเป็นอัตราส่วนเพียง 20 % ดังนั้นโหนดในอันดับถัดมาจะไม่สามารถทำงานในส่วนของการเป็นผู้กระจายข้อมูลลำดับถัดไปแทนได้ เพราะว่าโหนดนั้นอาจจะมีความแรงของสัญญาณต่ำและครอบคลุมเพื่อนบ้านน้อย ทำให้การส่งข้อมูลไปให้กับเพื่อนบ้านนั้นย่อมจะสำเร็จน้อย และเพื่อนบ้านอาจจะได้รับข้อมูลไม่ทั่วถึง นอกจากนั้นจากการทดลองพบว่าการเลือกให้สามารถใช้โหนดที่ได้รับการโหวตลำดับรองจากลำดับหนึ่งในการเป็นผู้ถูกเลือกได้แทนนั้นเกิดปัญหาคำตอบที่ดีที่สุดเฉพาะที่ (Local Optimum) ทำให้เกิดการกระจุกตัวของข้อมูลเฉพาะบางที่ ส่งผลให้ข้อมูลนั้นไม่กระจายไปทั่วระบบซึ่งโหนดในบริเวณอื่นจะไม่ได้รับข้อมูล และทำให้ประสิทธิภาพของโพรโทคอลต่ำลง



รูปที่ 4-22 ปัญหาโหนดที่ได้รับการโหวตอันดับหนึ่งไม่ได้เป็นเพื่อนบ้านกับผู้ส่งข้อมูล

การแก้ไขปัญหาดังกล่าวทำโดยให้ผู้ส่งข้อมูลนั้นกระจายข้อมูลออกไปโดยไม่มีการเลือกผู้กระจายข้อมูลลำดับถัดไปเพื่อให้เพื่อนบ้านที่ได้รับข้อมูลนี้ทำการนับเวลาถอยหลังแบบสุ่ม แล้วเมื่อเพื่อนบ้านใดที่ทำการนับเวลาหมดก่อนก็จะทำการกระจายข้อมูลลำดับถัดไปโดยจะทำการเลือกว่าจะให้เพื่อนบ้านโหนดใดเป็นโหนดส่งต่อข้อความไปด้วย โดยกระบวนการนี้จะทำให้การทำงานกลับไปสู่กระบวนการของโพรโทคอลตามปกติ ทำให้ข้อมูลนั้นสามารถกระจายไปทั้งระบบอย่างทั่วถึงมากขึ้น

2) การเลือกตัวเองเป็นผู้กระจายข้อมูลลำดับถัดไป

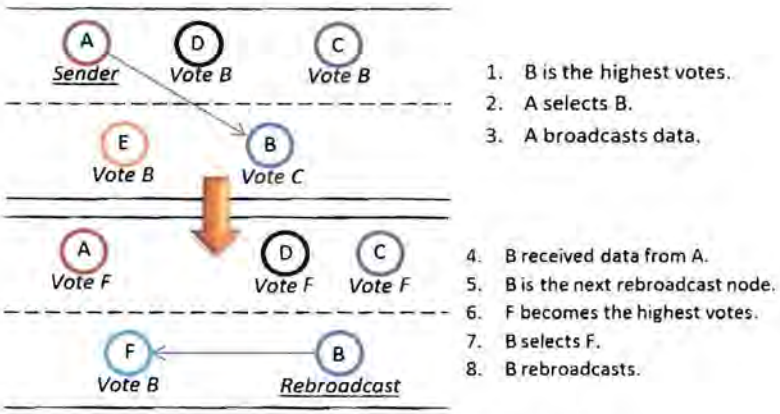
แบบโครงสร้าง (Topology) ของสถานการณ์การเคลื่อนที่ของยานพาหนะในการทดลองในงานวิจัยนี้สามารถแบบออกเป็นสองโครงสร้าง คือ แบบโครงสร้างที่เปลี่ยนแปลงเร็ว กับแบบโครงสร้างที่เปลี่ยนแปลงช้า ซึ่งโครงสร้างแบบหลังนี้จะทำให้เกิดปัญหาการเลือกตัวเองเป็นผู้กระจายข้อมูลลำดับถัดไปส่งผลให้การทำงานของโพรโทคอลนั้นมีประสิทธิภาพที่ต่ำลง โดยทั้งสองโครงสร้างมีรายละเอียดดังนี้

- **แบบโครงสร้างที่เปลี่ยนแปลงเร็ว (Fast-changing topology)**

แบบโครงสร้างนี้เกิดจากการที่ยานพาหนะนั้นเคลื่อนที่ด้วยความเร็วสูง หรือมีความหนาแน่นของยานพาหนะที่ต่ำทำให้ตำแหน่งของยานพาหนะนั้นจะมีการเปลี่ยนแปลงที่เร็วเคลื่อนที่ไปยังตำแหน่งอื่นๆ ต่างจากจุดเดิม ดังรูปที่ 4-23 โหนด A เป็นผู้ส่งข้อมูลแล้ว A พบว่าโหนด B ได้รับการโหวตจากเพื่อนบ้านมากที่สุด ต่อมา A จึงกระจายข้อมูลไปให้กับ

เพื่อนบ้านและเลือก B ให้เป็นผู้กระจายข้อมูลลำดับถัดไป ซึ่งในขณะนั้นทุกโหนดก็มีการเคลื่อนที่ทำให้เกิดการเปลี่ยนตำแหน่งของโหนด หลังจากที่ B ได้รับข้อมูลจาก A แล้ว B ก็จะทำการกระจายข้อมูลต่อ โดย B พบว่าผลของการโหวตจากเพื่อนบ้านนั้นได้เปลี่ยนแปลงแล้ว ซึ่ง F ได้รับการโหวตจากเพื่อนบ้านในละแวกให้เป็นผู้กระจายข้อมูลลำดับถัดไป ดังนั้น B จึงกระจายข้อมูลโดยมี F กระจายข้อมูลต่อ

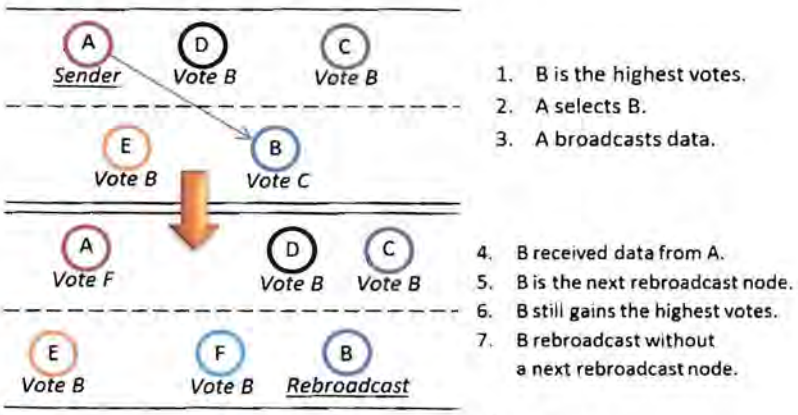
แบบโครงสร้างนี้นั้นพบว่าการทำงานของโพรโทคอลจะเป็นปกติ ลำดับในการกระจายข้อมูลจะเปลี่ยนไปเรื่อยในแต่ละโหนด ทำให้ประสิทธิภาพในการทำงานของโพรโทคอลนั้นดี



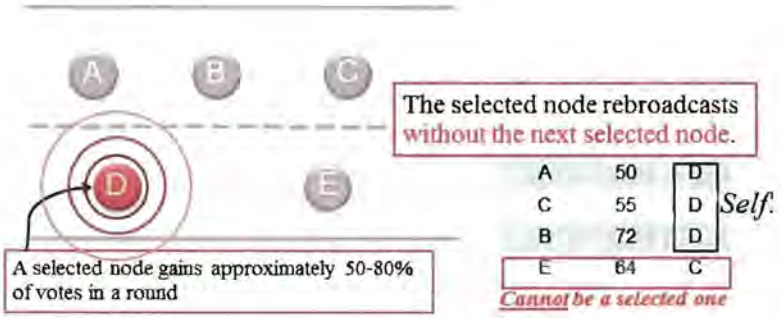
รูปที่ 4-23 แบบโครงสร้างที่เปลี่ยนแปลงเร็ว

- **แบบโครงสร้างที่เปลี่ยนแปลงช้า (Slow-changing topology)**

แบบโครงสร้างนี้เกิดจากการที่ยานพาหนะนั้นมีการเคลื่อนที่ช้า หรือมีความหนาแน่นของยานพาหนะสูง ทำให้ตำแหน่งของยานพาหนะนั้นแทบจะไม่เปลี่ยนแปลงไปจากเดิมแตกต่างจากแบบโครงสร้างที่กล่าวไปก่อนหน้า ซึ่งส่งผลให้เกิดปัญหาในการทำงานของโพรโทคอล ปัญหาดังกล่าวคือปัญหาการเลือกตัวเองเป็นผู้กระจายข้อมูลลำดับถัดไป ดังรูปที่ 4-24 โหนด A พบว่าโหนด B ได้รับการโหวตมากที่สุด และ A ทำการเลือก B ให้เป็นผู้กระจายข้อมูลลำดับถัดไป ซึ่งการทำงานในส่วนนี้ยังเป็นปกติ แต่การเคลื่อนที่ของโหนดช้า หรือมีความหนาแน่นของโหนดสูง ทำให้ตำแหน่งของโหนดไม่เกิดการเปลี่ยนแปลงมากนัก เมื่อ B ได้รับข้อมูลจาก A แล้ว B ก็ยังพบว่าเพื่อนบ้านในระแวกของตนเองนั้นยังคงเลือก B เองให้เป็นผู้กระจายข้อมูลลำดับถัดไป ทำให้ B พบปัญหาว่าตัวเองนั้นไม่อยู่ในตารางเพื่อนบ้านของตน ซึ่งปัญหานี้ก็คล้ายคลึงกับปัญหาก่อนหน้า เพราะการเลือกผู้ถูกเลือกลำดับรองลงมานั้นส่งผลให้การกระจายข้อมูลไม่มีประสิทธิภาพและเกิดปัญหาคำตอบที่ดีที่สุดเฉพาะที่ (Local Optimum) ทำให้เกิดการกระจุกตัวของข้อมูลเฉพาะที่ ส่งผลให้ข้อมูลนั้นไม่กระจายไปทั่วระบบซึ่งโหนดอื่นๆ รูปที่ 4-25 ก็เป็นอีกตัวอย่างของการเกิดปัญหานี้ขึ้นหากโหนด D ต้องการกระจายข้อมูลต่อแต่พบว่าตนเองเป็นผู้ถูกเลือกลำดับถัดไป



รูปที่ 4-24 แบบโครงสร้างที่เปลี่ยนแปลงช้า



รูปที่ 4-25 ปัญหาการเลือกตัวเองเป็นผู้กระจายข้อมูลลำดับถัดไป

ดังนั้นการแก้ไขปัญหาดังกล่าวทำโดยให้ผู้กระจายข้อมูลที่พบว่าตนเองเป็นผู้ส่งข้อมูลลำดับถัดไปนั้นกระจายข้อมูลออกไป โดยไม่เลือกผู้กระจายข้อมูลลำดับถัดไปเพื่อให้เพื่อนบ้านที่ได้รับข้อมูลนี้ทำการนับเวลาถอยหลังแบบสุ่มแล้วเมื่อเพื่อนบ้านที่ทำการนับเวลาหมดก่อนก็จะทำการกระจายข้อมูลลำดับถัดไป โดยจะทำการเลือกว่าจะให้เพื่อนบ้านโหนดใดเป็นผู้กระจายข้อมูลลำดับถัดไปด้วย ซึ่งจะทำให้การทำงานของโพรโทคอลนั้นกลับไปเป็นปกติ และกระจายข้อมูลให้ทั่วทั้งเครือข่ายไม่เกิดการกระจุกตัวของข้อมูลเฉพาะที่

4.4 การทดลองและวิเคราะห์ผลการทดลอง

4.4.1 ตัววัดสมรรถนะของโพรโทคอล (Performance Metrics)

การปรับปรุงขั้นตอนการเลือกของโพรโทคอล DECA มีตัววัดสมรรถนะที่สนใจ ดังนี้

1) *ความเชื่อถือได้ (Reliability)* เพื่อทดสอบว่าการปรับปรุงขั้นตอนการเลือกของโพรโทคอลสามารถปรับปรุงสมรรถนะด้านความเชื่อได้ในสภาพการเชื่อมต่อแบบอสมมาตรได้ ดังนั้นค่าความเชื่อถือได้จะคำนวณจากจำนวนข้อมูลที่ได้รับหารด้วยจำนวนข้อมูลทั้งหมดแล้ววัดผลออกมาเป็นเปอร์เซ็นต์

2) *ค่าใช้จ่าย (Overhead)* เพื่อทดสอบว่าการปรับปรุงขั้นตอนการเลือกของโพรโทคอลสามารถปรับปรุงค่าใช้จ่ายการทำงานในสภาพการเชื่อมต่อแบบอสมมาตรได้ โดยค่าใช้จ่ายที่เกิดขึ้นประกอบด้วยค่ายใช้จ่ายจากการกระจายข้อความซ้ำและค่าใช้จ่ายที่เกิดจากการส่งบีคอนในการบวนการทำงานทั้งหมด

4.4.2 เครื่องมือในการวัดสมรรถนะของโพรโทคอล

เครื่องมือในการวัดสมรรถนะจะใช้เครื่องมือเดียวกับการวัดสมรรถนะของโพรโทคอล DECA โดยที่การทดสอบเพิ่มเติมในโปรแกรมจำลองเครือข่าย NS-3 ดังนี้

1) *โปรแกรมจำลอง Simulator of Urban Mobility (SUMO)* ซึ่งจำลองพฤติกรรมการเคลื่อนที่ของยานพาหนะบนถนน โดยสามารถกำหนดรูปแบบการเคลื่อนที่ต่างๆ ได้ ทั้ง สภาพแวดล้อมแบบถนนในเมือง หรือถนนทางหลวง อีกทั้งยังสามารถกำหนดความหนาแน่นของยานพาหนะได้ด้วย
2) *โปรแกรม Traffic and Network Simulation Environment (TraNS)* ซึ่งเป็นโปรแกรมที่ใช้ในการเปลี่ยนรูปแบบการเคลื่อนที่ของยานพาหนะที่ได้จากโปรแกรม SUMO ให้อยู่ในรูปแบบที่โปรแกรมจำลองเครือข่ายเวอร์ชั่น 2 สามารถนำไปใช้งานได้
3) *โปรแกรมจำลองเครือข่าย NS-2.34* เป็นโปรแกรมจำลองเครือข่ายในลักษณะของการทำงานเครือข่ายเสมือนจริง
4) *โปรแกรมจำลองเครือข่าย NS-3.10* เป็นโปรแกรมจำลองเครือข่ายในลักษณะของการทำงานเครือข่ายเสมือนจริง ซึ่งสามารถใช้งานได้ทั้งการทดลองระบบจำลองเครือข่าย และการทดลองระบบจริง

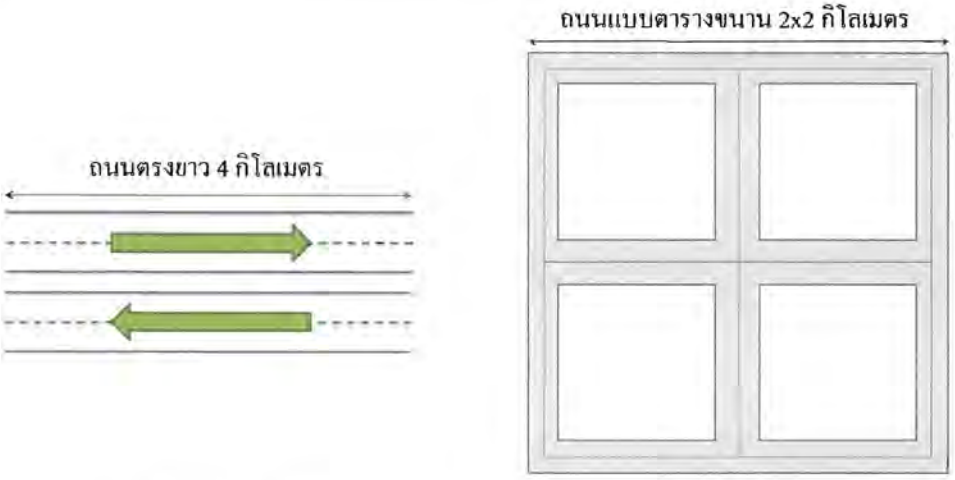
ในการทดลองนั้นเริ่มต้นด้วยการจำลองพฤติกรรมการเคลื่อนที่ของยานพาหนะ โดยใช้โปรแกรมจำลอง SUMO ในทั้งสภาพแวดล้อมบนถนนทางหลวงและสภาพแวดล้อมถนนในเมือง จากนั้นใช้โปรแกรม TraNS เพื่อที่จะเปลี่ยนรูปแบบของการเคลื่อนที่ดังกล่าวไปเป็นรูปแบบที่โปรแกรมจำลองเครือข่าย NS-2 นั้นสามารถเรียกใช้งานได้ ซึ่งโปรแกรมจำลองเครือข่าย NS-3 นั้นก็สามารถที่จะเรียกใช้รูปแบบดังกล่าวสำหรับ NS-2 ด้วยการเรียกใช้ฟังก์ชั่น Ns2MobilityHelper

4.4.3 สภาพแวดล้อมที่ใช้ในการทดลอง

สภาพแวดล้อมในการทดลองแบบออกเป็นสองรูปแบบคือ สภาพแวดล้อมของถนนทางหลวง และสภาพแวดล้อมของถนนในเมือง ซึ่งถนนทางหลวงนั้นมีลักษณะเป็นถนนทางตรงยาว 4 กิโลเมตร มีช่องทางจราจรขนาด 4 ช่องทาง ส่วนถนนในเมืองมีลักษณะเป็นตารางขนาด 2x2 ตารางกิโลเมตร ประกอบด้วยสี่แยก 1 แยก และสามแยกจำนวน 4 แยก ทั้งหมดนั้นไม่มีไฟจราจร โดยแต่ละแยกมีระยะห่างกัน 1 กิโลเมตร มีช่องทางจราจรขนาด 2 ช่องทาง เช่นเดียวกับการทดสอบในบทที่ 2 ดังรูปที่ 4-26 ความหนาแน่นของยานพาหนะโดยเฉลี่ยจะถูกแบ่งออกเป็นขนาดต่างๆ ดังตารางที่ 4-3 เพื่อทดลองการทำงานของโพรโทคอลและพฤติกรรมของยานพาหนะในสถานการณ์ต่างๆ ที่แตกต่างกัน เพื่อพิจารณาถึงผลของประสิทธิภาพในการทำงานของโพรโทคอล ซึ่งสภาพการจราจรนั้นถูกจำลองโดยใช้โปรแกรมจำลอง SUMO รุ่น 0.10.3 และเปลี่ยนแปลงรูปแบบให้เหมาะสมกับโปรแกรมจำลองเครือข่าย NS-2 ด้วยโปรแกรม TraNS รุ่น 1.0

ตารางที่ 4-3 การตั้งค่าพารามิเตอร์เพื่อทดลองประสิทธิภาพ

อายุของข้อความ	10 วินาที
จำนวนของข้อความ	5
ขนาดของข้อความ	512 ไบต์
ระยะเวลาในการทดลอง	50 วินาที
ความเร็วสูงสุดของยานพาหนะ (กิโลเมตร/ชั่วโมง)	50, 80
ความหนาแน่นของยานพาหนะ (คัน/กิโลเมตร)	2, 10, 30, 60, 120
โปรแกรมจำลองเครือข่าย	NS-3.10
สถานการณ์การสื่อสารไร้สาย (ระยะในการส่งข้อมูลสูงสุด (เมตร)/ Propagation Loss Model)	- 250 เมตร/ Two-Ray Ground - 250 เมตร/ Nakagami - 150, 250 เมตร/ Two-Ray Ground - 150, 250 เมตร/ Nakagami



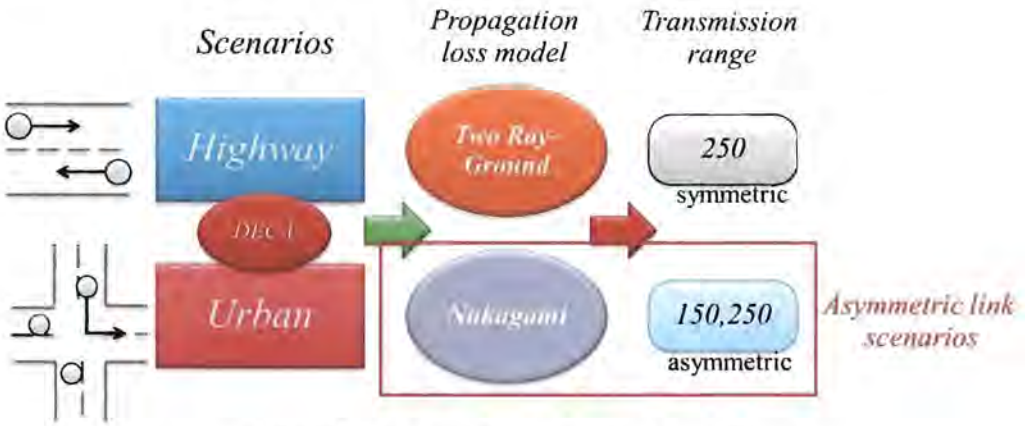
ก) ถนนทางหลวง ข) ถนนในเมือง

รูปที่ 4-26 ลักษณะถนนที่ใช้ในการทดลอง

ในสถานการณ์การเชื่อมต่อแบบอสมมาตรนั้นมีผลกระทบอย่างมากในสภาพการจราจรที่มีความหนาแน่นต่ำ เนื่องจากโหนดนั้นจะมีระยะห่างระหว่างกันมากซึ่งเมื่อความแรงสัญญาณต่ำทำให้ระยะการส่งข้อมูลนั้นสั้นกว่าระยะห่างระหว่างโหนดทำให้เกิดผลกระทบจากการเชื่อมต่อแบบอสมมาตรอย่างชัดเจน แต่ในรูปแบบของโหนดที่มีความหนาแน่นมากตำแหน่งของแต่ละโหนดนั้นใกล้กันจนเมื่อความแรงของสัญญาณต่ำ ระยะส่งข้อมูลสั้นก็ไม่มีผลกระทบต่อการทำงาน เพราะว่าระยะห่างระหว่างโหนดอยู่ในระยะที่ยังส่งข้อมูลได้ ดังนั้นงานวิจัยนี้จึงเน้นรูปแบบของยานพาหนะที่มีความหนาแน่นต่ำโดยใช้สภาพแวดล้อมรูปแบบถนนทางหลวงที่มีความหนาแน่น 2, 6, 10, 30, 60 และ 120 คันต่อกิโลเมตร ซึ่งมากกว่าสภาพแวดล้อมรูปแบบถนนในเมืองที่ใช้ความหนาแน่นในการทดลองเพียงแค่ 2, 6 และ 10 คันต่อกิโลเมตร เพราะ

รูปแบบถนนทางหลวงนั้นมีความหนาแน่นของโหนดที่น้อยและลักษณะการเคลื่อนที่ของโหนดที่เร็วกว่าทำให้ไม่เกิดการกระจุกเหมาะกับการทดลองสถานการณ์ที่มีการเชื่อมต่อแบบอสมมาตร

นอกจากนั้นการทดลองนี้ยังเปลี่ยน Propagation Loss Model ซึ่งเดิมโปรแกรมจำลองเครือข่าย NS-3 นั้นใช้ Two-Ray Ground Propagation Loss Model ที่มีค่าแน่นอนในการใช้งานและมีความเป็นไปได้ในการส่งข้อมูลที่ไม่เป็นความจริง เพราะว่าสามารถกำหนดระยะในการส่งข้อมูลได้แน่นอนซึ่งหากเลยระยะที่กำหนดจะไม่สามารถส่งข้อมูลได้เลย ดังนั้นหากต้องการให้การส่งข้อมูลของเครือข่ายมีความเสมือนจริงจึงเปลี่ยนการทดลองเป็น Nakagami Propagation Loss Model แทน เพราะมีรูปแบบที่คล้ายกับการทำงานบนระบบจริง เนื่องจากสัญญาณในระบบนั้นมีการถูกรบกวนและบิดเบือนทำให้ในแต่ละช่วงของระยะที่ส่งข้อมูลได้นั้นจะมีความเป็นไปได้แบบสุ่ม และถ้าผู้รับห่างจากผู้ส่งมากขึ้นความน่าจะเป็นในการส่งข้อมูลสำเร็จก็จะน้อยลง หากกำหนดระยะการส่งข้อมูลไว้ที่ระยะหนึ่งในการทดลองนี้จะกำหนดให้ระยะนั้นมีความน่าจะเป็นในการส่งข้อมูลสำเร็จเพียง 50 % และหากเลยระยะการส่งข้อมูลไปแล้วความน่าจะเป็นในการส่งข้อมูลสำเร็จจะมีค่าน้อยลงตามระยะทางที่เพิ่มขึ้น



รูปที่ 4-27 เงื่อนไขในการจำลองสถานการณ์ในการทดลอง

การทดลองยังประกอบไปด้วยการจำลองสถานการณ์ระยะการส่งสัญญาณสองแบบคือการเชื่อมต่อสมมาตรและการเชื่อมต่อแบบอสมมาตร ซึ่งในสถานการณ์แรกแบบสมมาตรนั้นจะให้ทุกโหนดในเครือข่ายนั้นมีความแรงของสัญญาณเท่ากันให้มีระยะการส่งข้อมูลอยู่ที่ 250 เมตร ส่วนอีกสถานการณ์จะให้โหนดแบ่งออกเป็นสองกลุ่ม คือ กลุ่มที่มีระยะการส่งข้อมูลอยู่ที่ 150 เมตร กับ 250 เมตร เพื่อแทนลักษณะของยานพาหนะที่มีความแตกต่างของระยะการส่งสัญญาณ ซึ่งจากรูปที่ที่ 4-27 จะสังเกตได้ว่าเมื่อใช้ Nakagami Propagation Loss Model กับสถานการณ์แบบอสมมาตร ซึ่งทำให้สอดคล้องกับสถานการณ์การเชื่อมต่อแบบอสมมาตรเสมือนการทำงานบนระบบจริง

ในการทดลองนั้นจะแบ่งสถานการณ์ในการทดลองเป็น 5 สถานการณ์ ดังนี้

1) DECA Symmetric with Two-Ray Ground สถานการณ์จำลองที่แทนลักษณะของสถานการณ์แบบสมมาตรซึ่งทำงานด้วยโพรโทคอล DECA โดยให้แต่ละโหนดมีระยะการส่งข้อมูลที่ 250 เมตร และใช้ Two-Ray Ground Propagation Loss Model ซึ่งเป็นลักษณะการทำงานของอุปกรณ์ไร้สายแบบอุดมคติ ในสถานการณ์นี้มักถูกใช้ในการทดลองจำลองเครือข่ายของงานวิจัยส่วนใหญ่

2) DECA Symmetric with Nakagami สถานการณ์จำลองที่แทนลักษณะของสถานการณ์แบบสมมาตรซึ่งทำงานด้วยโพรโทคอล DECA โดยให้แต่ละโหนดมีระยะการส่งข้อมูล 250 เมตร และใช้ Nakagami Propagation Loss Model ซึ่งให้ค่าที่ใกล้เคียงกับสัญญาณในระบบจริง ทำให้การทดลองนั้นใกล้เคียงระบบจริงมากขึ้น

3) DECA Asymmetric with Two-Ray Ground สถานการณ์จำลองที่แทนลักษณะของสถานการณ์แบบอสมมาตรซึ่งทำงานด้วยโพรโทคอล DECA โดยให้โหนดแบ่งออกเป็นสองกลุ่มคือ ระยะส่งสัญญาณ 150 เมตร กับ 250 เมตร และใช้ Two-Ray Ground Propagation Loss Model ทำให้การจำลองนั้นใกล้เคียงกับระบบจริงที่แต่ละโหนดมีความแรงของสัญญาณแตกต่างกันมากขึ้น

4) DECA Asymmetric with Nakagami สถานการณ์จำลองที่แทนลักษณะแบบสถานการณ์แบบอสมมาตรซึ่งทำงานด้วยโพรโทคอล DECA โดยแบ่งโหนดออกเป็นสองกลุ่มคือระยะการส่งข้อมูล 150 เมตร กับ 250 เมตร และใช้ Nakagami Propagation Loss Model ซึ่งให้ค่าเสมือนการทำงานบนระบบจริง

5) RVA Asymmetric with Nakagami สถานการณ์จำลองที่แทนลักษณะแบบสถานการณ์อสมมาตรที่ทำงานด้วยอัลกอริทึมการโหวตแบบอาร์เอสเอสไอ (RSSI-Voting algorithm : RVA) ในการเลือกโหนด โดยแบ่งโหนดออกเป็นสองกลุ่มคือ ระยะส่งข้อมูล 150 เมตร กับ 250 เมตร และใช้ Nakagami Propagation Loss Model ซึ่งให้ค่าเสมือนกับการทำงานบนระบบจริง

ในการทดลองใช้โปรแกรมจำลองเครือข่ายเวอร์ชั่น 3.10 แต่ละครั้งข้อมูลจะถูกแพร่กระจายจำนวน 1 ข้อความเป็นจำนวน 5 ข้อความ ซึ่งมีขนาดข้อความละ 512 ไบต์ โดยอายุของข้อความนั้นคือ 10 วินาที ซึ่งเมื่อข้อความหมดอายุก็จะถูกลบออกจากระบบ และข้อความใหม่ก็จะถูกแพร่กระจายออกมาจากผู้ส่งข้อมูลทันที การทดลองระบบสื่อสารไร้สายของโหนดทำงานตามมาตรฐาน IEEE 802.11 โดยมีการชนกันของข้อมูลตามปกติ ดังตารางที่ 4-3

4.4.4 ผลการทดลองค่าความเชื่อถือได้

ค่าความเชื่อถือได้จากผลการทดลองบนถนนทางหลวง ดังรูปที่ 4-28 ก) เมื่อพิจารณาจากสถานการณ์จำลองต่างๆ มีรายละเอียดดังนี้

1) **DECA Symmetric with Two-Ray Ground** สถานการณ์นี้มักถูกใช้ทั่วไปในการจำลองเครือข่ายของงานวิจัยทั่วไป ซึ่งค่าความเชื่อถือได้ที่ออกมานั้นมีค่าสูงเกินจริงเพราะว่า Two-Ray Ground นั้นให้ค่าของสัญญาณที่ไม่สมจริงอีกทั้งทุกโหนดในเครือข่ายยังมีความแรงของสัญญาณเท่ากัน ทำให้ค่าความเชื่อถือได้นั้นออกมาดี แต่ก็ไม่สามารถที่จะนำผลจากการทดลองนี้ไปใช้ได้บนระบบจริงเพราะว่าสถานการณ์ในการทดลองนั้นยังขาดอีกหลายปัจจัยที่มีบนระบบจริง

2) **DECA Symmetric with Nakagami** สถานการณ์รูปแบบนี้นั้นผลที่ออกมาจะแย่กว่าสถานการณ์แบบที่หนึ่งเพราะว่า Nakagami นั้นให้ค่าของสัญญาณในการส่งข้อมูลที่เสมือนจริงมากขึ้น แต่ทุกโหนดในเครือข่ายยังคงมีความแรงของสัญญาณที่เท่ากันทำให้ค่าที่ออกมานั้นยังไม่เสมือนจริง

3) **DECA Asymmetric with Two-Ray Ground** สถานการณ์รูปแบบนี้ค่าความเชื่อถือได้ที่ออกมานั้นจะดีกว่าสถานการณ์แบบที่สอง แต่ยังแย่กว่าสถานการณ์แบบที่หนึ่งถึงแม้ว่าโหนดในเครือข่ายจะถูกแบ่งออกเป็นสองกลุ่มทำให้มีความแรงของสัญญาณที่แตกต่างกัน แต่ว่า Two-Ray Ground ยังให้ค่าที่ไม่เสมือนจริงทำให้ผลที่ออกมานั้นดีกว่า Nakagami

4) **DECA Asymmetric with Nakagami** สถานการณ์รูปแบบนี้นั้นมีค่าความเชื่อถือได้ที่ต่ำที่สุดเมื่อเทียบกับสถานการณ์ที่ทำการทดลองทั้งหมด เพราะว่าโหนดทั้งเครือข่ายนั้นมีความแรงของสัญญาณที่แตกต่างกันอีกทั้ง Nakagami ยังให้ค่าของสัญญาณในการส่งข้อมูลที่เสมือนจริงทำให้การทำงานของระบบนั้นคล้ายคลึงกับการทำงานบนระบบจริงมาก

ที่สุด ซึ่งยานพาหนะแต่ละคันอาจมีความแรงของสัญญาณที่แตกต่างกันและสัญญาณนั้นอาจถูกบิดเบือนหรือดูดซับจากบรรยากาศภายนอกต่างๆ

5) **RVA Asymmetric with Nakagami** สถานการณ์ในการทดลองสุดท้ายนี้ทำงานด้วยขั้นตอนการโหวตแบบอาร์เอสเอสไอซึ่งแก้ไขปัญหาในการเลือกโหนดของโพรโทคอลแล้วทำให้ค่าความเชื่อถือได้ที่วัดออกมานั้นมีค่าที่มากขึ้นเมื่อเทียบกับสถานการณ์แบบที่สี่ที่เสมือนจริงมากที่สุด โดยค่าที่ออกมานั้นดีขึ้นมากสุดถึง 17 %

เมื่อพิจารณาในมุมมองของความหนาแน่นของยานพาหนะในเครือข่าย สามารถแยกได้เป็นสองความหนาแน่นหลักๆ ดังนี้

- **ความหนาแน่นต่ำ (2-10 คัน/กิโลเมตร)**

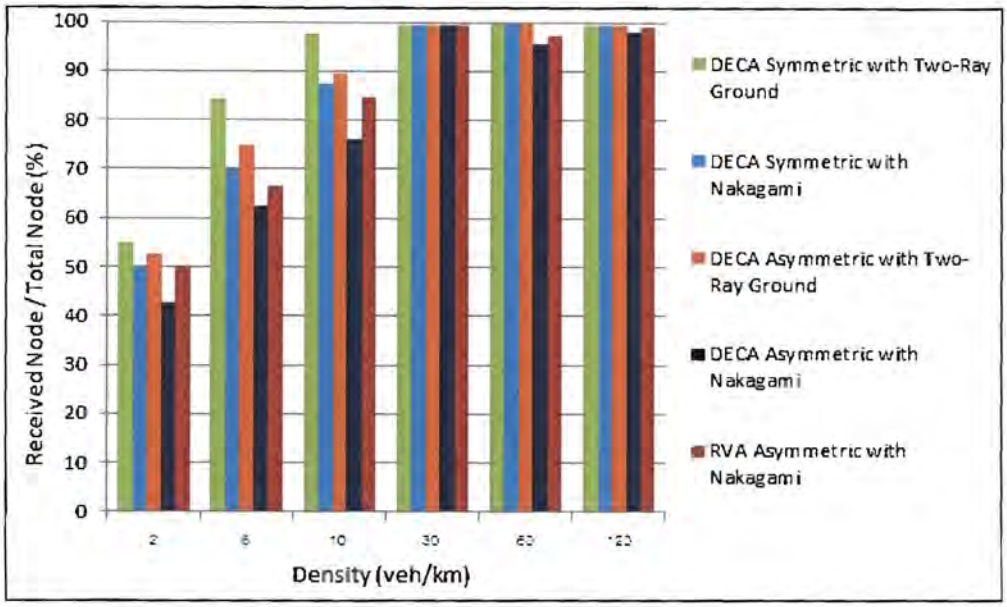
ความเชื่อถือได้ที่วัดออกมานั้นจะมีค่าออกมาต่ำเมื่อเทียบกับความหนาแน่นที่สูงขึ้นเรื่อยๆ เพราะว่าในสถานการณ์ที่มีความหนาแน่นต่ำนั้นจะได้รับผลกระทบจากการเชื่อมต่อแบบอสมมาตรมากที่สุด โดยระยะในการส่งข้อมูลของแต่ละโหนดนั้นจะสั้นกว่าระยะห่างระหว่างโหนดแต่ละโหนด ทำให้เมื่อแต่ละโหนดมีความแรงของสัญญาณที่แตกต่างกันและค่าของสัญญาณที่เสมือนจริง ระยะในการส่งข้อมูลของแต่ละโหนดยิ่งน้อยลงกว่าเดิม ทำให้การส่งข้อมูลในเครือข่ายนั้นไม่ทั่วถึงแล้วยิ่งความหนาแน่นต่ำมากก็ทำให้ค่าความเชื่อถือที่ออกมานั้นจะต่ำมากเช่นกัน แต่เมื่อแก้ไขปัญหาด้วยขั้นตอนการโหวตแบบอาร์เอสเอสไอก็ทำให้ผลออกมาดีขึ้นโดยในความหนาแน่นที่ต่ำสุดนั้นสามารถเพิ่มค่าความเชื่อถือได้ให้เท่ากับสถานการณ์การเชื่อมต่อแบบสมมาตรในรูปแบบสัญญาณที่เป็น Nakagami Propagation Loss Model ได้

- **ความหนาแน่นสูง (20-120 คัน/กิโลเมตร)**

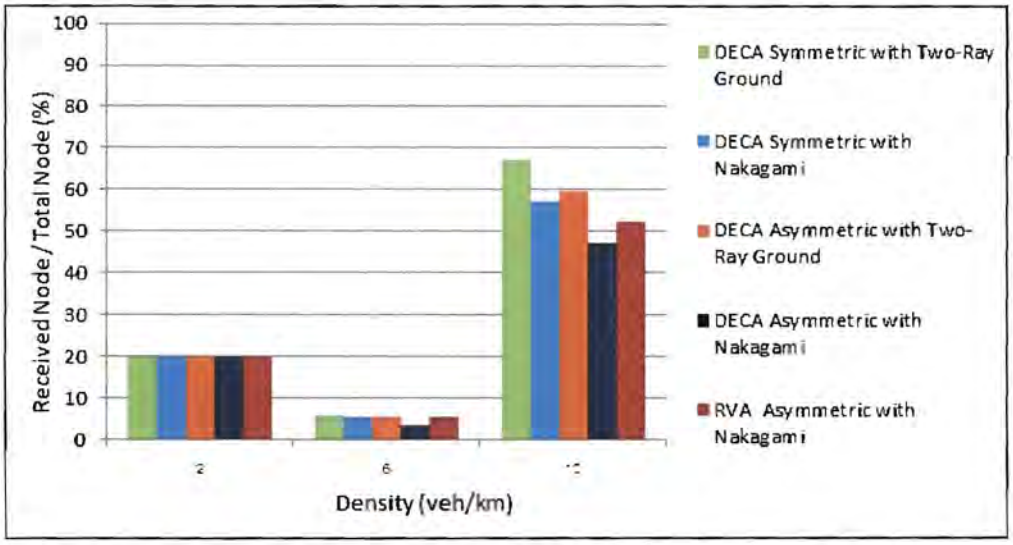
การทดลองนี้ค่าความเชื่อถือได้จะมีค่ามากกว่าความหนาแน่นต่ำ เพราะว่าในสถานการณ์นี้จะไม่ได้รับผลกระทบจากการเชื่อมต่อแบบอสมมาตรมาก เนื่องจากระยะในการส่งข้อมูลของแต่ละโหนดนั้นมีระยะที่ยาวกว่าระยะห่างของแต่ละโหนดแม้ว่าความแรงของสัญญาณที่ส่งจะน้อยก็ตามทำให้ค่าความเชื่อถือได้นั้นออกมาสูงกว่าสถานการณ์ที่มีความหนาแน่นต่ำ แต่อย่างไรก็ตามการเชื่อมต่อแบบอสมมาตรก็ส่งผลกระทบแต่ไม่มากนักซึ่งการแก้ไขด้วยขั้นตอนการโหวตอาร์เอสเอสไอก็ส่งผลให้ค่าความเชื่อถือได้ของระบบดีขึ้นตามมา

ค่าความเชื่อถือได้จากผลการทดลองบนถนนในเมือง ดังรูปที่ 4-28 ข) การทดลองบนสภาพแวดล้อมแบบถนนในเมืองนั้นมีลักษณะเหมือนกันการทดลองสภาพแวดล้อมแบบถนนทางหลวงที่มีความหนาแน่นสูง เพราะว่าถนนในเมืองนั้นมีลักษณะที่เป็นตารางทำให้การเคลื่อนที่ของยานพาหนะนั้นจะกระจายไปตามเส้นทางที่เลี้ยวไปมา และมีการหยุดเพื่อรอให้รถที่อยู่สี่แยกหรือสามแยกเดินทางไปก่อนซึ่งการหยุดก็ทำให้สามารถส่งข้อมูลไปให้กับโหนดอื่นๆ ได้ ดังนั้นค่าความเชื่อถือได้ที่ออกมานั้นจึงมีผลที่แตกต่างกับถนนทางหลวง

ในส่วนของความหนาแน่น 2 คัน/กิโลเมตรนั้นเนื่องจากการกำหนดการเคลื่อนที่ของโหนดเป็นไปในทางที่ทำให้ยานพาหนะสามารถเจอกันได้หมดทำให้ค่าความเชื่อถือได้ในแต่ละสถานการณ์จำลองนั้นออกมาเท่าๆ กัน แต่ในส่วนของความหนาแน่นอื่นๆ อย่างเช่น 6 กับ 10 คัน/กิโลเมตร ค่าความเชื่อถือได้ของสถานการณ์เชื่อมต่อแบบอสมมาตรนั้นมีผลออกมาต่ำ ซึ่งแก้ไขด้วยอัลกอริทึมการโหวตแบบอาร์เอสเอสไอแล้วก็ทำให้การทำงานของโพรโทคอลนั้นมีประสิทธิภาพที่ดีขึ้นมากสุดถึง 5 %



ก) ถนนทางหลวง



ข) ถนนในเมือง

รูปที่ 4-28 กราฟแสดงค่าความเชื่อถือได้

จากผลการทดลองสรุปได้ว่าค่าความเชื่อถือได้ของโพรโทคอลนั้นได้รับผลกระทบมากที่สุดจากสถานการณ์ที่โหนดมีความหนาแน่นต่ำ และโหนดแต่ละโหนดมีความแรงของสัญญาณที่แตกต่างกันตามการเชื่อมต่อแบบอสมมาตร อีกทั้งสัญญาณมีการตั้งค่าแบบ Nakagami Propagation Loss Model ซึ่งเสมือนสัญญาณของระบบจริง

4.4.5 ผลการทดลองค่าใช้จ่าย

ค่าใช้จ่ายได้จากผลการทดลองบนถนนทางหลวง (ดังรูปที่ 4-29, 4-30 และ 4-31) และถนนในเมือง (ดังรูปที่ 4-32, 4-33 และ 4-34) ซึ่งพิจารณาจากกราฟของค่าใช้จ่ายทั้งการทดลองบนถนนทางหลวงและถนนในเมืองจะพบความสอดคล้องกัน สามารถนำมาพิจารณาคล้ายกับค่าความเชื่อถือได้ดังนี้

เมื่อพิจารณาในมุมมองของสถานการณ์จำลอง สถานการณ์จำลองที่เชื่อมต่อกันแบบสมมาตรและมีสัญญาณแบบ Two-Ray Ground นั้นจะสามารถส่งข้อมูลหากันได้ดีทำให้ค่าใช้จ่ายในการส่งข้อมูลทั้งหมดของระบบนั้นน้อยกว่าสถานการณ์แบบอื่นๆ แต่ว่าด้วยลักษณะที่ไม่เสมือนจริงทำให้ค่านั้นไม่เหมาะกับการใช้อ้างถึงประสิทธิภาพของโพรโทคอลได้ ส่วนสถานการณ์จำลองที่ใกล้เคียงกับระบบจริงมากขึ้นก็ทำให้ค่าใช้จ่ายของระบบนั้นมีมากขึ้นเนื่องจากเมื่อการส่งข้อมูลระหว่างโหนดเป็นไปได้ยากโหนดที่ไม่ได้รับข้อมูลก็ต้องส่งบีคอนออกมาหาเพื่อนบ้านให้ส่งข้อมูลนั้นๆ ให้กับตนเอง ส่งผลให้ค่าใช้จ่ายก็เพิ่มขึ้นเรื่อยๆ และขึ้นมากที่สุดในสถานการณ์จำลองการเชื่อมต่อแบบอสมมาตรที่มีสัญญาณแบบ Nakagami เมื่อแก้ไขปัญหาด้วยอัลกอริทึมการโหวตก็ทำให้ค่าใช้จ่ายของการส่งข้อมูลในระบบนั้นน้อยลงมากสุดถึง 28 %

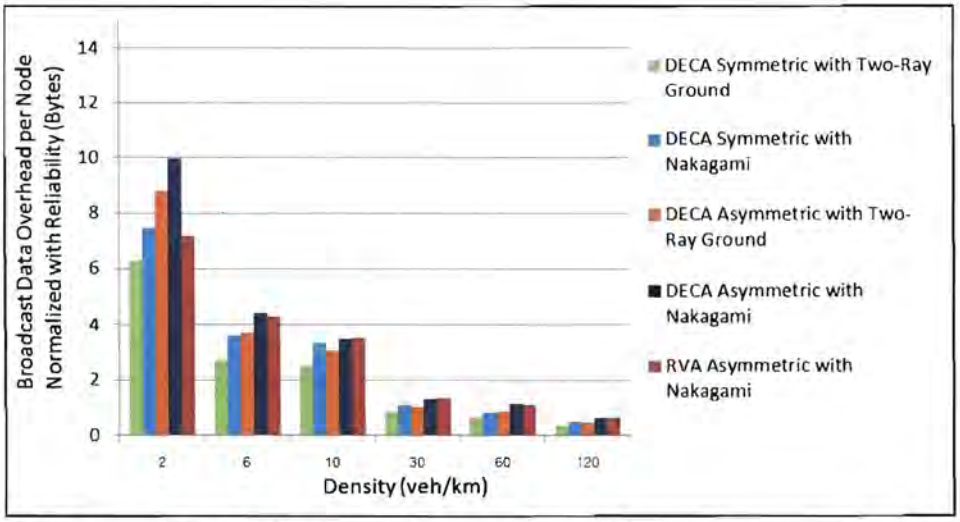
เมื่อพิจารณาในมุมมองของความหนาแน่นของยานพาหนะในเครือข่าย ในสภาพแวดล้อมของการทดลองที่มีความหนาแน่นต่ำนั้น การส่งข้อมูลระหว่างโหนดเป็นเรื่องที่ยาก เพราะระยะในการส่งข้อมูลนั้นจะสั้นกว่าระยะห่างระหว่างโหนดทำให้มีการไม่ได้รับข้อมูลเกิดขึ้นในบางโหนด ซึ่งโหนดนั้นก็ต้องส่งบีคอนออกมาให้เพื่อนบ้านเพื่อให้ส่งข้อมูลนั้นกลับมายังตนเองอีก ส่งผลให้ค่าใช้จ่ายของระบบนั้นจะมาก ซึ่งเมื่อความหนาแน่นเพิ่มขึ้นทำให้ระยะระหว่างโหนดนั้นน้อยลง การส่งข้อมูลก็สำเร็จมากขึ้นก็ทำให้ค่าใช้จ่ายของระบบนั้นน้อยลงตามมาด้วยเช่นกัน แต่เมื่อสังเกตการทดลองบนถนนในเมืองที่ความหนาแน่น 2 คัน/กิโลเมตร นั้นค่าใช้จ่ายจะน้อยกว่าความหนาแน่น 6 คัน/กิโลเมตร เพราะว่าการกำหนดรูปแบบการเคลื่อนที่ของโหนดนั้นส่งผลให้การเคลื่อนที่ของโหนดสามารถที่จะส่งข้อมูลให้กับโหนดในระบบได้ทั้งหมดทำให้ค่าใช้จ่ายจึงออกมาน้อยกว่าความหนาแน่นลำดับถัดไป

โดยจากกราฟแสดงค่าใช้จ่ายนั้นสามารถแยกออกได้เป็น 3 แบบ ดังนี้

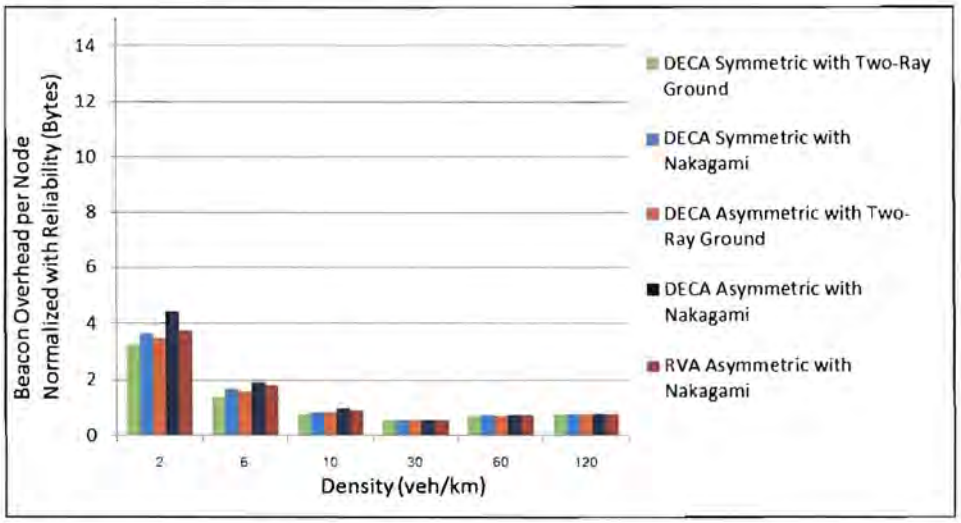
1) *กราฟแสดงค่าใช้จ่ายในการส่งข้อความ* เมื่อผู้ส่งเริ่มกระจายข้อความให้กับโหนดเพื่อนบ้าน ระบบก็จะทำการกระจายข้อความไปให้กับโหนดอื่น ๆ ที่อยู่ในละแวกต่อไป โดยข้อความนั้นจะมีขนาด 512 ไบท์ ซึ่งมากกว่าค่าใช้จ่ายในการแลกเปลี่ยนข้อมูลระหว่างเพื่อนบ้านในข้อความประเภทบีคอน

2) *กราฟแสดงค่าใช้จ่ายในการส่งบีคอน* การส่งบีคอนนั้นแต่ละโหนดจะทำการส่งให้เพื่อนบ้านของตนเองเป็นระยะเพื่อแลกเปลี่ยนข้อมูลของของกันและกัน โดยข้อความนั้นจะประกอบด้วยข้อมูลที่บอกว่าได้รับข้อความใดแล้ว และผลของการโหวตเพื่อนบ้านที่มีค่าอาร์เอสเอสไอมากที่สุด

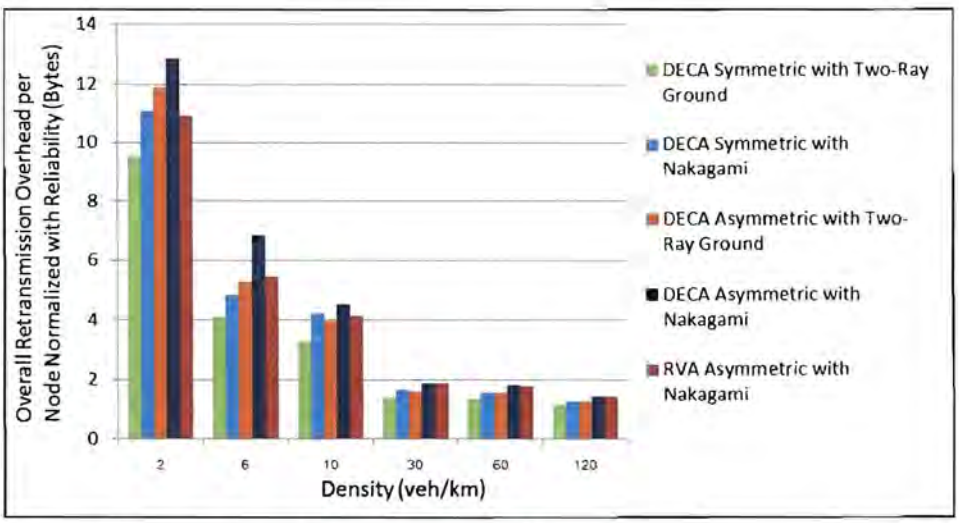
3) *กราฟแสดงค่าใช้จ่ายรวม* กราฟนี้เป็นกราฟแสดงผลรวมของค่าใช้จ่ายในการส่งข้อความกับค่าใช้จ่ายในการส่งบีคอนซึ่งจะแสดงให้ทราบว่าค่าใช้จ่ายที่ใช้ไปจริงต่อหนึ่งโหนดต่อหนึ่งข้อความนั้นมีค่าเท่าใด



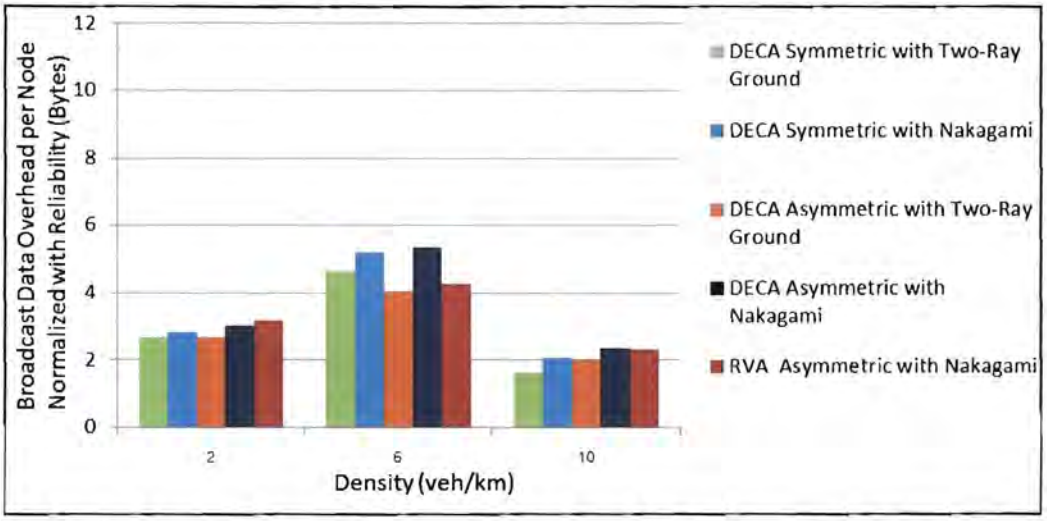
รูปที่ 4-29 กราฟแสดงค่าใช้จ่ายในการส่งข้อความของระบบที่ได้จากการทดลองบนถนนทางหลวง



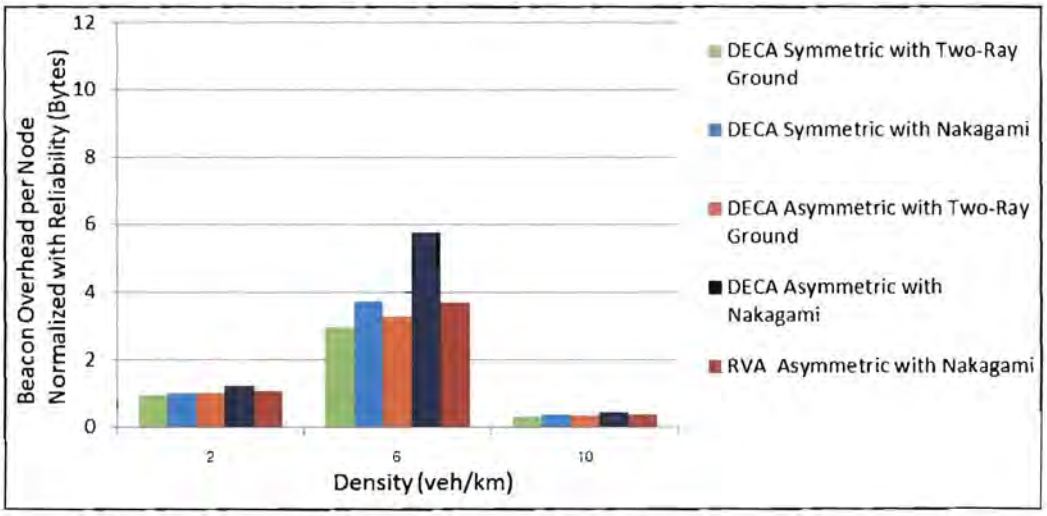
รูปที่ 4-30 กราฟแสดงค่าใช้จ่ายในการส่งบีคอนของระบบที่ได้จากการทดลองบนถนนทางหลวง



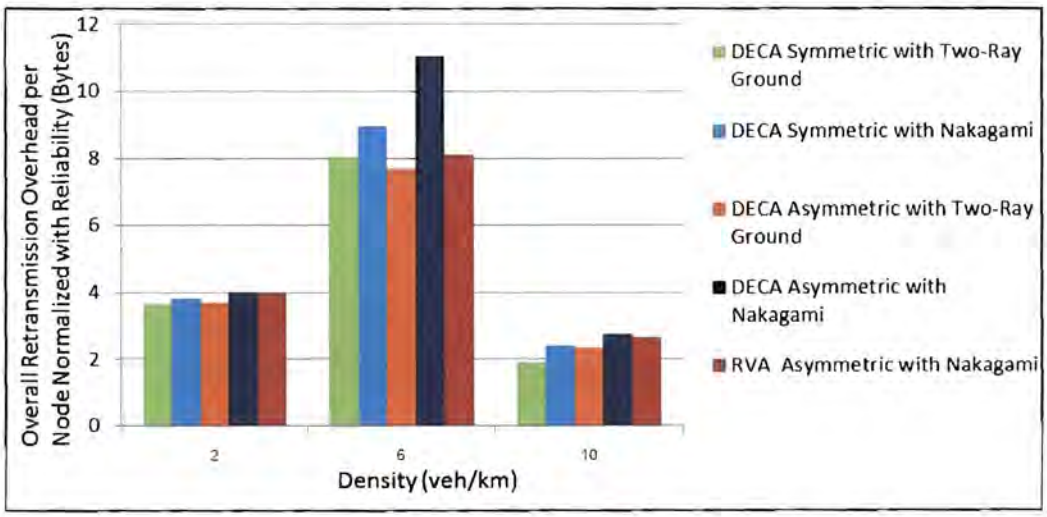
รูปที่ 4-31 กราฟแสดงค่าใช้จ่ายรวมของระบบที่ได้จากการทดลองบนถนนทางหลวง



รูปที่ 4-32 กราฟแสดงค่าใช้จ่ายในการส่งข้อความของระบบที่ได้จากการทดลองบนถนนในเมือง



รูปที่ 4-33 กราฟแสดงค่าใช้จ่ายในการส่งบีคอนของระบบที่ได้จากการทดลองบนถนนในเมือง



รูปที่ 4-34 กราฟแสดงค่าใช้จ่ายรวมของระบบที่ได้จากการทดลองบนถนนในเมือง

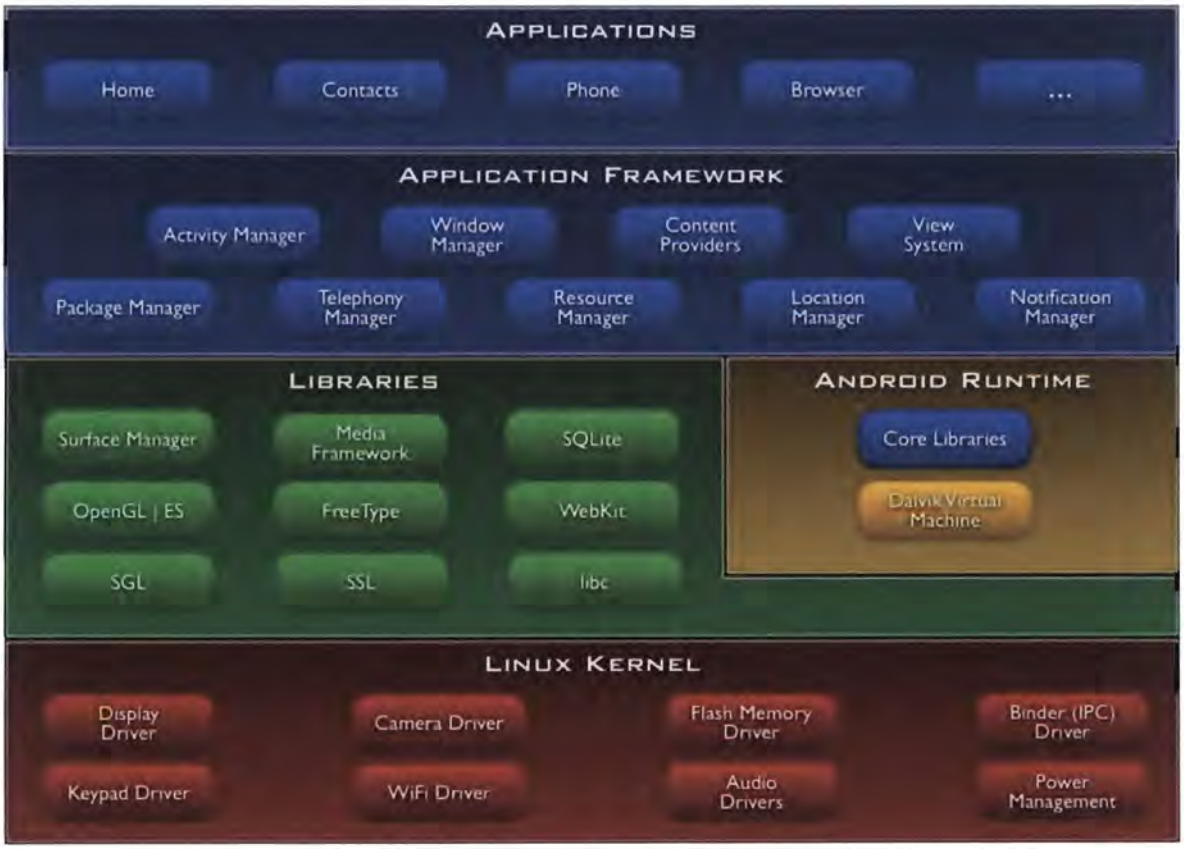
บทที่ 5 การพัฒนาโพรโทคอลบนอุปกรณ์ระบบปฏิบัติการแอนดรอยด์

การพัฒนาโพรโทคอลในบทที่ผ่านมาเพื่อปรับปรุงสมรรถนะการทำงานของโพรโทคอลเพื่อให้มีสมรรถนะสูงที่สุดในการใช้งานบนอุปกรณ์ และเครือข่ายจริง ในบทนี้จึงเป็นการนำโพรโทคอลมาพัฒนาบนอุปกรณ์มีที่ระบบปฏิบัติการแอนดรอยด์ และทดสอบการใช้งานผ่านแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย โดยในการทดสอบนั้นจะประกอบด้วยอุปกรณ์ที่แตกต่าง คือ คอมพิวเตอร์แบบพกพาที่มีระบบปฏิบัติการแตกต่างกัน และโทรศัพท์มือถือที่ใช้ระบบปฏิบัติการแอนดรอยด์

ในบทนี้จะกล่าวถึงส่วนประกอบที่สำคัญของระบบปฏิบัติการแอนดรอยด์ การพัฒนาโพรโทคอล DECA บนระบบปฏิบัติการแอนดรอยด์ และการทดสอบการทำงานของโพรโทคอลกับอุปกรณ์ที่มีความหลากหลาย

5.1 ระบบปฏิบัติการแอนดรอยด์ (Android)

แอนดรอยด์เป็นระบบปฏิบัติการสำหรับอุปกรณ์พกพา เช่น โทรศัพท์มือถือและแท็บเล็ตคอมพิวเตอร์ ซึ่งมีพื้นฐานมาจากระบบปฏิบัติการลินุกซ์ (Linux) ข้อดีของระบบปฏิบัติการแอนดรอยด์คือเป็นซอฟต์แวร์โอเพ่นซอร์ส (Open Source) ซึ่งเปิดกว้างให้ผู้พัฒนาสามารถนำไปพัฒนาปรับแต่งได้อย่างอิสระ



รูปที่ 5-1 โครงสร้างสถาปัตยกรรมแอนดรอยด์ [26]

สถาปัตยกรรมของแอนดรอยด์ [26] ประกอบด้วยส่วนต่างๆที่น่าสนใจดังต่อไปนี้

1) **Application** : แอพพลิเคชั่นของแอนดรอยด์จะถูกเขียนด้วยภาษาจาวา(Java)

2) **Application Framework** : เนื่องจากเป็น Open Development Platform ทำให้ผู้พัฒนาสามารถใช้ประโยชน์จากฮาร์ดแวร์ของเครื่องได้เต็มที่ ผู้พัฒนามีสิทธิ์ใช้งาน Framework APIs เดียวกับที่ใช้พัฒนาแอพพลิเคชั่นหลักของแอนดรอยด์

3) **Libraries** : แอนดรอยด์มีไลบรารี่ C/C++ ซึ่งถูกใช้โดยส่วนประกอบต่างๆ ของระบบ ผู้พัฒนาสามารถใช้ความสามารถของไลบรารี่ต่างๆ นี้ได้ผ่าน Android Application Framework

ตัวอย่างไลบรารี่หลักที่ผู้พัฒนาสามารถใช้งานได้

- *System C Library* เป็น BSD-Derived Implementation ของไลบรารี่ C System มาตรฐานที่ถูกปรับแต่งให้เหมาะสมกับระบบฝังตัว (Embedded System)
- *Media Libraries* เป็นไปตาม PacketVideo's OpenCORE ไลบรารี่นี้สนับสนุนการเล่นและบันทึกไฟล์เสียง ไฟล์วีดีโอ และไฟล์รูปภาพที่เป็นที่นิยมต่างๆ เช่น MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- *Surface Manager* ใช้จัดการแอพพลิเคชั่นตั้งแต่หนึ่งแอพพลิเคชั่นขึ้นไปในการเข้าใช้งาน Display Subsystem
- *LibWebCore* เป็น Web Browser Engine
- *SGL* เป็น 2D Graphics Engine
- *3D Libraries* เป็นการพัฒนาตาม OpenGL ES 1.0 APIs ไลบรารี่นี้จะใช้ Hardware 3D Acceleration หรือ 3D Software Rasterizer
- *FreeType* ใช้ทำ Font Rendering
- *SQLite* เป็น Lightweight Relational Database Engine

4) **Android Runtime** : เนื่องจากแอพพลิเคชั่นของแอนดรอยด์จะถูกพัฒนาโดยใช้ภาษาจาวา (Java) แอนดรอยด์จึงมีไลบรารี่เพื่อให้บริการฟังก์ชั่นต่างๆ ที่มีในไลบรารี่หลักของจาวา (Java) แอพพลิเคชั่นของแอนดรอยด์จะทำงานอยู่บน Dalvik Virtual Machine ของตัวเองในโปรเซสของตน Dalvik ถูกพัฒนาขึ้นมาให้เครื่องสามารถใช้งาน Virtual Machine หลายๆ ตัวพร้อมกันได้อย่างมีประสิทธิภาพ

โปรแกรมภาษาจาวา (Java) ที่เขียนขึ้น จะถูกคอมไพล์ (Compile) ด้วยคอมไพเลอร์ภาษาจาวา (Java Language Compiler) และถูกแปลงเป็นไฟล์ Dalvik Executable (นามสกุล dex) ด้วยเครื่องมือชื่อ dx ไฟล์ Dalvik Executable มีข้อดีคือถูกดัดแปลงให้ใช้หน่วยความจำอย่างประหยัด ซึ่งสุดท้ายไฟล์ Dalvik Executable จะถูกนำไปทำงานบน Dalvik Virtual Machine และเป็นแอพพลิเคชั่นที่ผู้ใช้ได้ใช้งาน

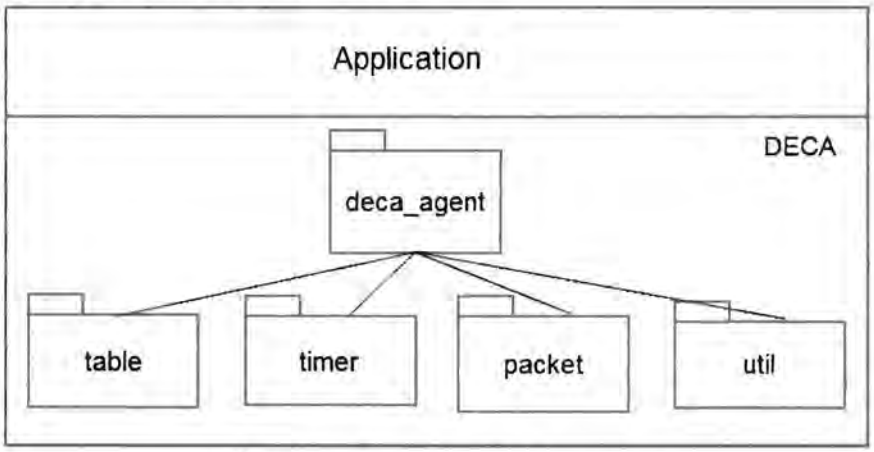
Dalvik Virtual Machine จะมีการเรียกใช้งานเคอร์เนลลินุกซ์สำหรับฟังก์ชันบางอย่าง เช่น Threading และการจัดการหน่วยความจำในระดับ Low-Level

5) **Linux Kernel** : ระบบปฏิบัติการแอนดรอยด์ใช้เคอร์เนลลินุกซ์เวอร์ชั่น 2.6 ในการจัดการ System Services ต่างๆ เช่น การจัดการหน่วยความจำ การจัดการโปรเซส การจัดการ Network Stack และการจัดการ Driver ต่างๆ นอกจากนี้เคอร์เนลยังเป็น Abstraction Layer ระหว่าง Hardware และ Software Stack อีกด้วย

5.2 การพัฒนาโพรโทคอล DECA และแอพพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ (Android)

แนวคิดในการพัฒนามาจากการพัฒนาในวิทยานิพนธ์ของ Rabie Khodr Jradi และ Lasse Seligmann Reedtz [27] ซึ่งนำเสนอกระบวนการและขั้นตอนการพัฒนาไลบรารี่เพื่อให้บริการโพรโทคอล Ad-hoc On-demand Distance Vector (AODV) บนระบบปฏิบัติการแอนดรอยด์ นอกจากนี้ในวิทยานิพนธ์ยังได้ทำการทดสอบไลบรารี่ด้วยการสร้างแอพพลิเคชั่นส่งข้อความอย่างง่าย โดยใช้ไลบรารี่ดังกล่าวเป็นพื้นฐาน โทรศัพท์มือถือที่ถูกทดสอบในวิทยานิพนธ์นี้ คือ HTC Hero และ Google Nexus One

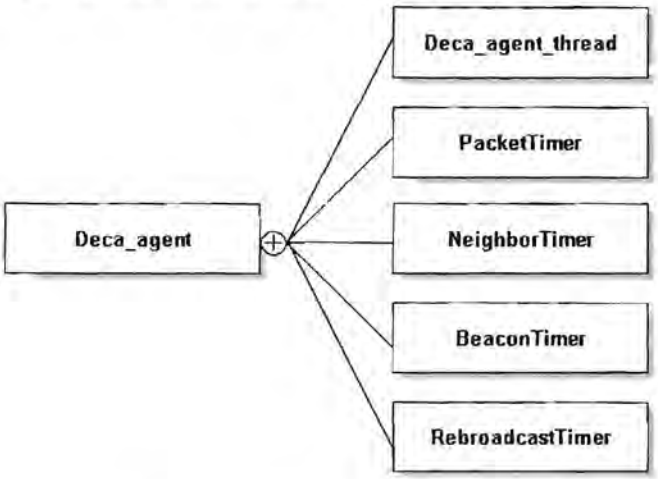
การพัฒนาโปรโตคอล DECA แบ่งออกเป็น 5 Packages ได้แก่ *Deca_agent, table, timer, packet* และ *util* ดังรูปที่ 5-2 ซึ่งในแต่ละ Package มีรายละเอียดการพัฒนาดังต่อไปนี้



รูปที่ 5-2 โครงสร้างการพัฒนาโพรโทคอล DECA

5.2.1 *Package Deca_agent*

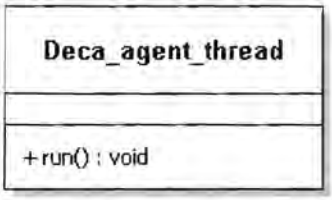
ใน *Package Deca_agent* ประกอบด้วยโครงสร้างคลาสต่างๆ ดังรูปที่ 5-3 ซึ่งแต่ละคลาสมีรายละเอียดดังนี้



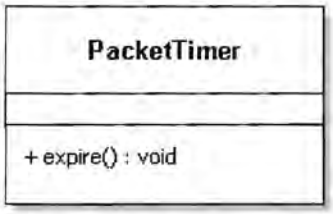
รูปที่ 5-3 รายละเอียด *Package Deca_agent*

- *Class Deca_agent* : เป็นคลาสสำหรับจัดการการทำงานของโพรโทคอล DECA ซึ่งทำหน้าที่ต่างๆ ได้แก่
  - แพร่กระจายข้อมูลจากโหนดของตัวเอง
  - รับข้อความที่ผู้อื่นเป็นผู้กระจาย
  - ตรวจสอบข้อความที่ตัวเองได้รับกับข้อความที่เพื่อนบ้านได้รับ ถ้าตัวเองขาดข้อความให้ส่งบีคอนหาเพื่อนบ้านเพื่อขอข้อความที่ตัวเองขาด หากเพื่อนบ้านได้รับข้อความไม่ครบให้ส่งข้อความนั้นให้เพื่อนบ้าน
  - แพร่กระจายข้อความซ้ำทันทีถ้าถูกเลือกให้เป็นโหนดส่งต่อข้อความ (Preferred Node)
  - แพร่กระจายข้อความซ้ำหากโหนดที่ถูกเลือกเป็นโหนดส่งต่อข้อความ (Preferred Node) ไม่ทำหน้าที่แพร่กระจายข้อความซ้ำ
- *Class Deca_agent_thread* : เป็น Thread แยกออกมาเพื่อทำหน้าที่รับข้อความจากเพื่อนบ้าน
- *Class PacketTimer* : เป็นคลาสสำหรับจับเวลาว่าข้อความที่ได้รับมาจะหมดอายุเมื่อใด เมื่อข้อความหมดอายุให้ลบข้อความนั้นออกจาก *rptable*
- *Class NeighborTimer* : เป็นคลาสสำหรับจับเวลาว่าเพื่อนบ้านหมดอายุเมื่อใด เมื่อเพื่อนบ้านหมดอายุก็ให้ลบเพื่อนบ้านออกจาก *nbtable*
- *Class BeaconTimer* : เป็นคลาสสำหรับจับเวลาในการส่งบีคอนครั้งต่อไป
- *Class RebroadcastTimer* : เป็นคลาสสำหรับจับเวลาว่าเวลารอของข้อความที่อยู่ใน *ttable* หมดเมื่อใด เมื่อหมดเวลารอก็ให้โหนดแพร่ข้อความนั้น

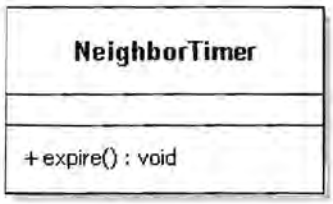
รูปที่ 5-4 รายละเอียดของ *Class Deca_agent*



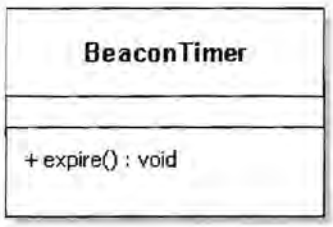
รูปที่ 5-5 รายละเอียดของ *Class Deca_agent_thread*



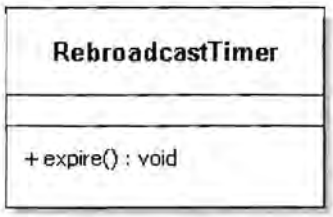
รูปที่ 5-6 รายละเอียดของ *Class PacketTimer*



รูปที่ 5-7 รายละเอียดของ *Class NeighborTimer*



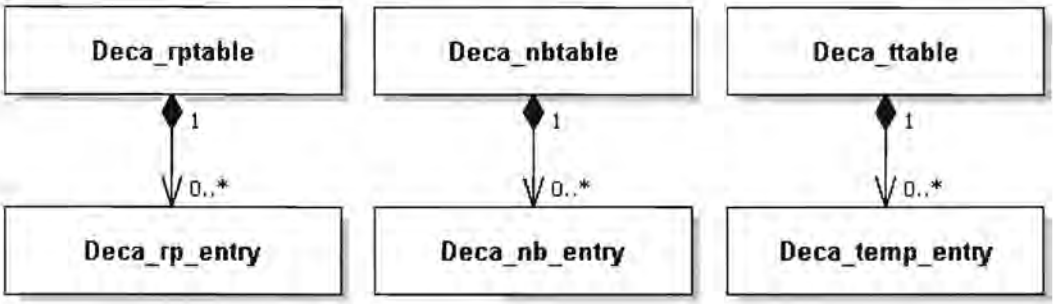
รูปที่ 5-8 รายละเอียดของ *Class BeaconTimer*



รูปที่ 5-9 รายละเอียดของ *Class RebroadcastTimer*

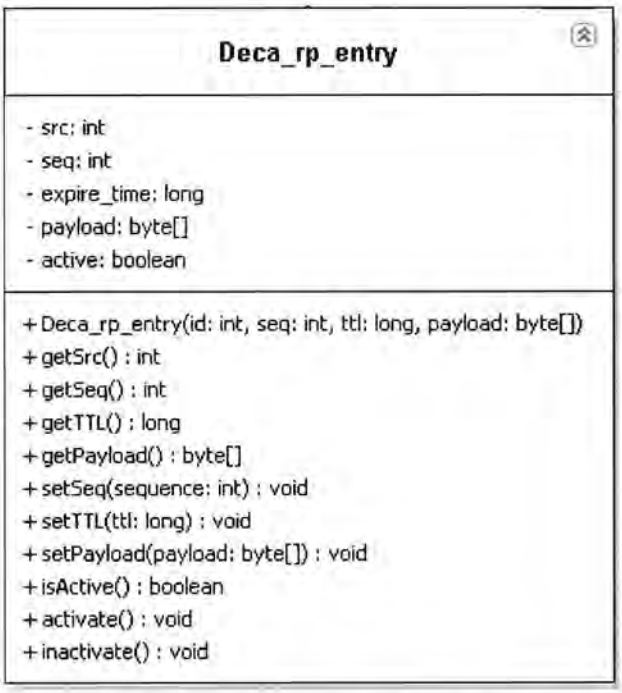
5.2.2 *Package table*

ใน *Package table* ประกอบด้วยโครงสร้างคลาสต่างๆ ดังรูปที่ 5-10 ซึ่งในแต่ละคลาสมีรายละเอียดดังนี้

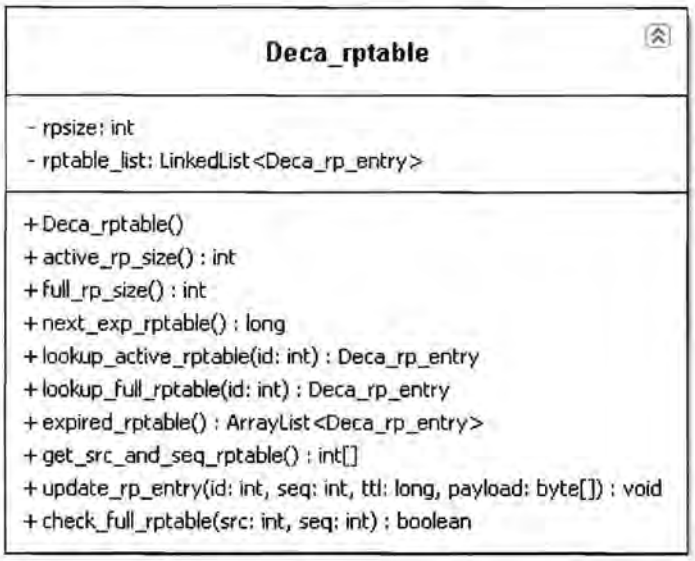


รูปที่ 5-10 รายละเอียดของ *Package table*

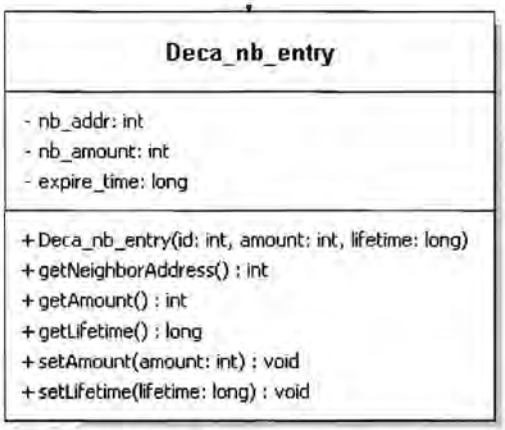
- *Class Deca_rp_entry* : ใช้สำหรับเก็บข้อมูลข้อความที่ได้รับจากเพื่อนบ้านและข้อความที่ตัวเองส่งออกไป
- *Class Deca_rptable* : ตารางสำหรับเก็บข้อมูล *Deca_rp_entry*
- *Class Deca_nb_entry* : ใช้สำหรับเก็บข้อมูลของเพื่อนบ้าน ซึ่งได้แก่ Address และจำนวนเพื่อนบ้าน
- *Class Deca_nbtable* : ตารางสำหรับเก็บข้อมูล *Deca_nb_entry*
- *Class Deca_temp_entry* : ใช้สำหรับเก็บข้อมูลของข้อความที่นับเวลาถอยหลังเพื่อแพร่กระจายซ้ำ ถ้าหากได้ยินเพื่อนบ้านแพร่กระจายข้อมูลซ้ำแล้ว ข้อมูลนี้จะถูกลบออก
- *Class Deca_ttable* : ตารางสำหรับเก็บข้อมูล *Deca_temp_entry*



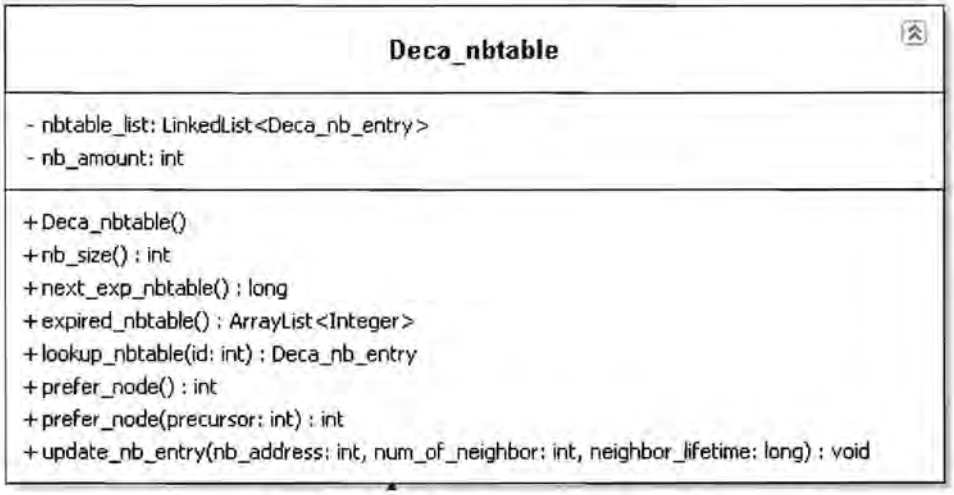
รูปที่ 5-11 รายละเอียดของ *Class Deca_rp_entry*



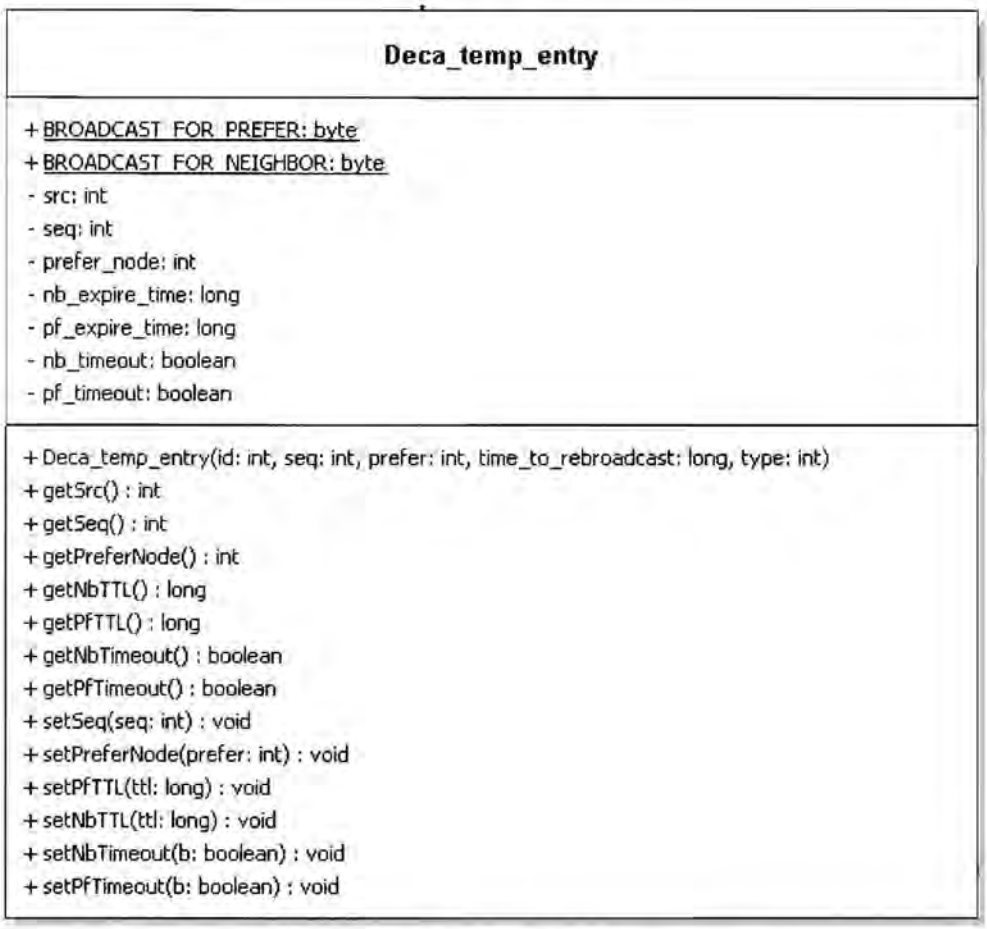
รูปที่ 5-12 รายละเอียดของ *Class Deca_rptable*



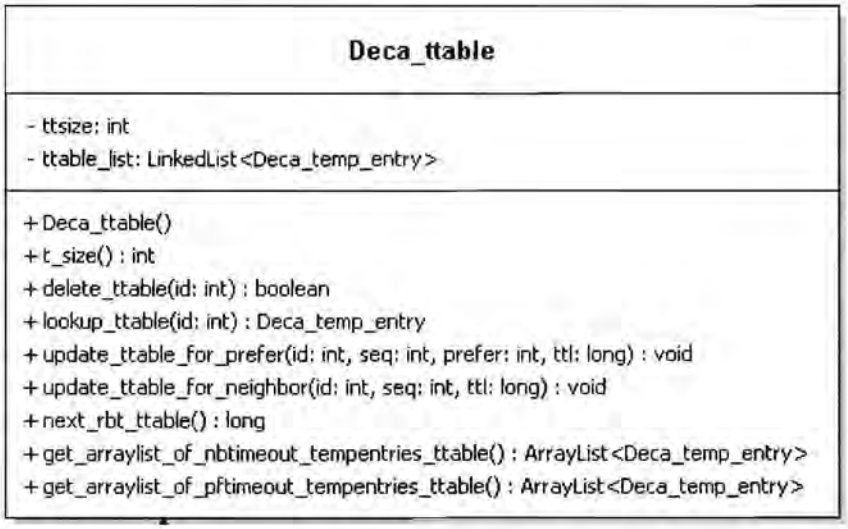
รูปที่ 5-13 รายละเอียดของ *Class Deca_nb_entry*



รูปที่ 5-14 รายละเอียดของ *Class Deca_nbtable*



รูปที่ 5-15 รายละเอียดของ *Class Deca_temp_entry*

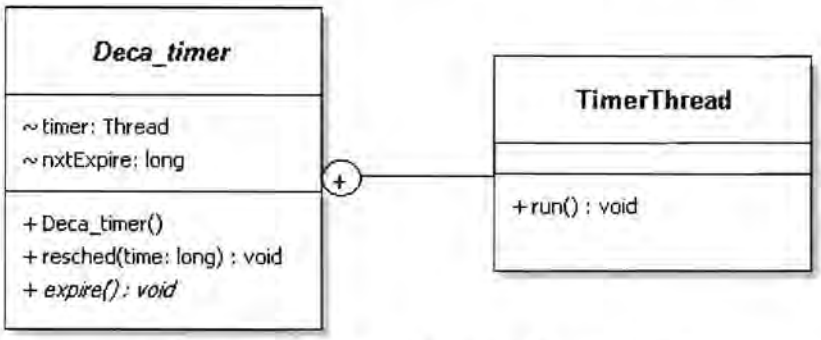


รูปที่ 5-16 รายละเอียดของ *Class Deca_ttable*

5.2.3 *Package timer*

ใน *Package timer* จะประกอบด้วยคลาสต่างๆ ดังรูปที่ 5-17

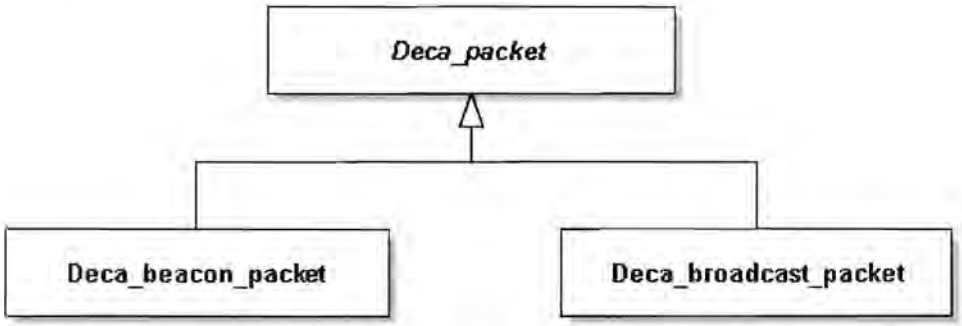
- *Class Deca_timer* : ภายในมี Inner class ใช้สำหรับนับเวลา และสามารถตั้งเวลาที่ต้องการให้หมดอายุได้
- *Class TimerThread* : เป็น Thread สำหรับนับเวลาตามค่าที่ตั้ง เมื่อหมดเวลาแล้วจะทำงานในคำสั่ง expire()



รูปที่ 5-17 รายละเอียด Class ต่างๆ ภายใน *Package timer*

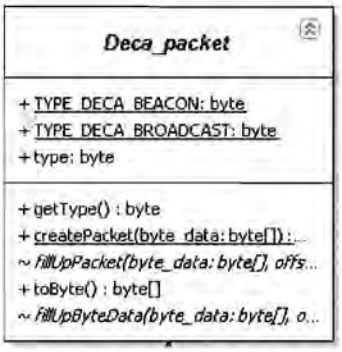
5.2.4 *Package packet*

ใน *Package packet* จะประกอบด้วยคลาสต่างๆ ดังรูปที่ 5-18 แต่ละคลาสมีรายละเอียดดังนี้

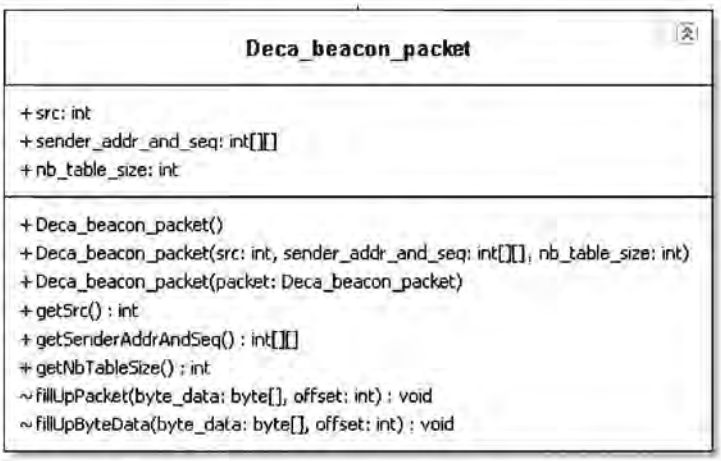


รูปที่ 5-18 รายละเอียดของ *Package packet*

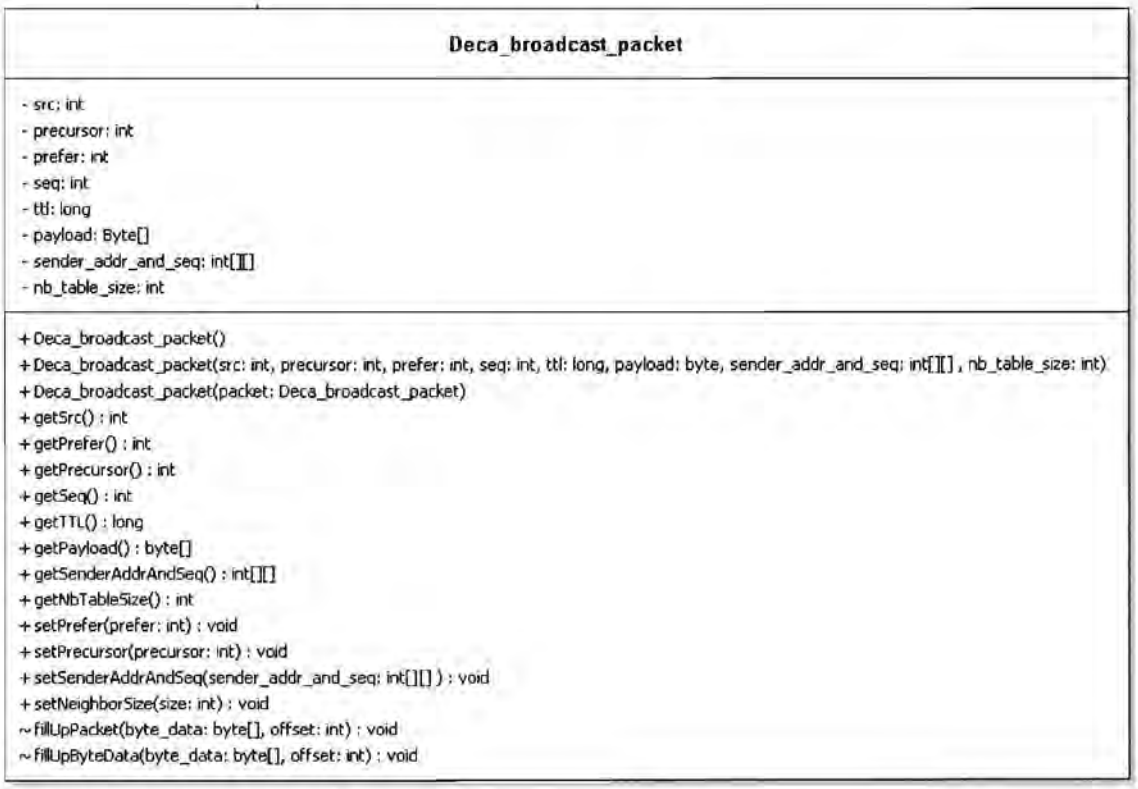
- *Class Deca_packet* : ใช้สร้าง Packet แล้วแปลงให้อยู่ในรูปแบบที่สามารถแพร่กระจายให้โหนดอื่นได้
- *Class Deca_beacon_packet* : ใช้สร้าง Beacon Packet ที่ใช้แลกเปลี่ยนข้อมูลระหว่างโหนดที่อยู่ติดกัน
- *Class Deca_broadcast_packet* : ใช้สร้าง Broadcast Packet ซึ่งเป็นข้อความที่ใช้แพร่กระจายโดยโพรโทคอล DECA



รูปที่ 5-19 รายละเอียดของ *Class Deca_packet*



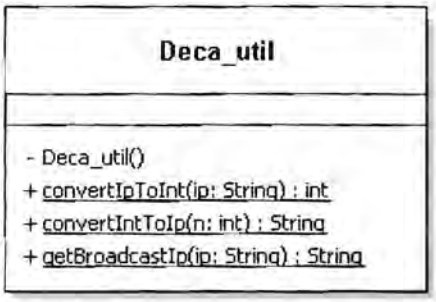
รูปที่ 5-20 รายละเอียดของ *Class Deca_beacon_packet*



รูปที่ 5-21 รายละเอียดของ *Class Deca_broadcast_packet*

5.2.5 *Package util*

ใน *Package util* จะประกอบด้วย *Class Deca_util* เพียงคลาสเดียวซึ่งเป็นคลาสที่อำนวยความสะดวกให้กับการทำงานของโปรโตคอล DECA มีฟังก์ชั่นสำหรับเปลี่ยนข้อมูล Address จาก Integer เป็น String หรือจาก String เป็น Integer และฟังก์ชั่นคำนวณหา Broadcast Address

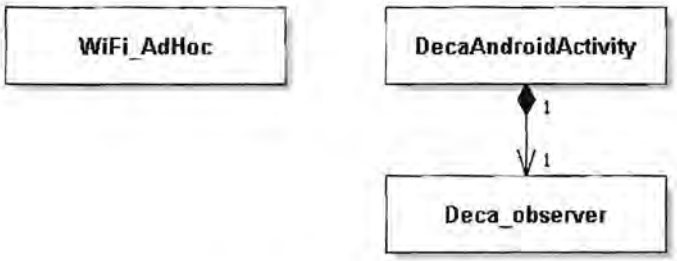


รูปที่ 5-22 รายละเอียดของ *Class Deca_util*

5.3 การพัฒนาแอพพลิเคชั่นแอนดรอยด์สำหรับการกระจายข้อความอย่างง่าย

แอพพลิเคชั่นสำหรับประชาสัมพันธ์ข้อมูลถูกพัฒนาอยู่ภายใน *Package decaandroid* ซึ่งภายใน *Package decaandroid* ประกอบด้วยคลาสต่อไปนี้ *WiFi_AdHoc*, *DecaAndroidActivity* และ *DecaObserver* ซึ่งมีโครงสร้างคลาสต่างๆ ดังรูปที่ 5-24 แต่ละคลาสมีรายละเอียดดังนี้

รูปที่ 5-23 การทำงานระหว่างแอพพลิเคชั่นแอนดรอยด์กับโพรโทคอล DECA



รูปที่ 5-24 รายละเอียด Class ต่างๆ ภายในแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย

- *Class WiFi_AdHoc :* คลาสนี้ทำหน้าที่ 2 อย่างคือ
  1) กำหนดชื่อที่ผู้ใช้ต้องการใช้ในระบบ หากผู้ใช้ไม่ใส่ชื่อระบบจะสร้างชื่อ Anonymous ให้
  2) เปิด/ปิดเครือข่ายแอดฮอก โดยเมื่อผู้ใช้กดเปิดเครือข่ายแอดฮอก แอพพลิเคชั่นจะกำหนดค่าที่จำเป็น จากนั้นจึงสร้างเครือข่ายแอดฮอกหรือเข้าร่วมเครือข่ายแอดฮอกในกรณีที่มีผู้อื่นสร้างเครือข่ายแอดฮอกไว้แล้ว และเมื่อเครือข่ายแอดฮอกพร้อมใช้งาน แอพพลิเคชั่นจะอนุญาตให้ผู้ใช้กดเข้าสู่หน้าแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย
- *Class DecaAndroidActivity :* คลาสนี้จะเรียกใช้งาน *Deca_agent* ซึ่งทำหน้าที่จัดการการรับ-ส่งข้อความที่ นอกจากนี้ทำหน้าที่ในการแสดงผลข้อความที่ได้รับหรือแจ้งเตือนผู้ใช้ผ่านทาง Notification Bar และมีช่องสำหรับให้พิมพ์ข้อความที่ผู้ใช้ต้องการกระจายข้อความให้กับผู้อื่น
- *Class Deca_observer :* เมื่ออุปกรณ์ได้รับข้อความใหม่จากอุปกรณ์อื่นๆภายในเครือข่าย คลาส *Deca_observer* จะแจ้งเตือนไปที่คลาส *DecaAndroidActivity* ว่ามีข้อความใหม่เข้ามา เพื่อให้ Application นำข้อความไปอัพเดต User Interface หรือแจ้งเตือนผู้ใช้ผ่าน Notification Bar

รูปที่ 5-25 แบบร่าง User Interface สำหรับเปิด/ปิดเครือข่ายแอดฮอก

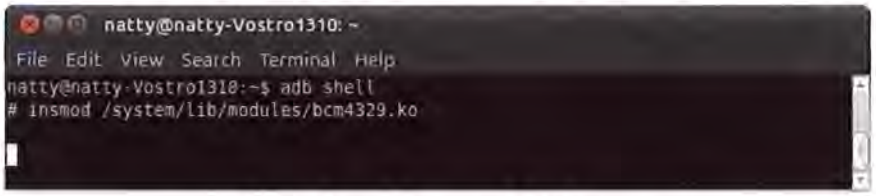
รูปที่ 5-26 แบบร่าง User Interface สำหรับรับ/ส่งข้อความ

5.4 ปัญหาที่น่าสนใจในการพัฒนาโพรโทคอล DECA บนอุปกรณ์ที่มีระบบปฏิบัติการแอนดรอยด์

5.4.1 ปัญหา Rom มาตรฐานของโทรศัพท์มือถือ Google Nexus One ไม่สามารถเปิดใช้งาน Wi-Fi ใน mode Ad-Hoc ได้

เมื่อใช้ ROM มาตรฐานที่ติดตั้งมาให้บนโทรศัพท์มือถือ Google Nexus One พบว่าไม่สามารถเปิดใช้งาน Wi-Fi ในโหมดแอดฮอกได้ จึงทำการติดตั้ง ROM ใหม่ โดยเลือกใช้ CyanogenMod 7 เนื่องจาก CyanogenMod เป็น Custom ROM ที่สามารเปิดใช้งาน Wi-Fi ในโหมดแอดฮอกได้

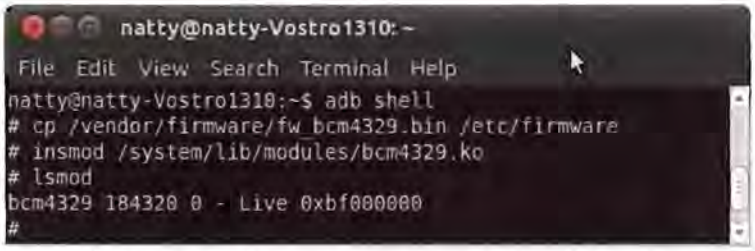
เมื่อทำการติดตั้ง CyanogenMod 7 พบว่าเครื่องสามารถมองเห็นและเชื่อมต่อกับเครือข่ายแอดฮอกที่สร้างขึ้นโดยคอมพิวเตอร์ที่ใช้ระบบปฏิบัติการ Ubuntu 10.04 ได้ แต่พบปัญหาเครื่องไม่สามารถสร้างเครือข่ายแอดฮอกขึ้นมาเองได้ มีผลทำให้ไม่สามารถเชื่อมต่อระหว่างเครื่องมือถือ 2 เครื่องขึ้นไปโดยไม่พึ่งเครื่องคอมพิวเตอร์ได้ โดยในการสร้างเครือข่ายแอดฮอกขึ้นมาเองบนเครื่องมือถือ สามารถแก้ปัญหาโดยการใช้วิธีต่อเครื่องมือถือ Google Nexus One เข้ากับเครื่องคอมพิวเตอร์ที่ลง Android SDK ไว้ แล้วใช้คำสั่ง ./adb shell เพื่อเรียก shell ขึ้นมาสั่งการแอนดรอยด์ จากนั้นจะใช้คำสั่ง insmod เพื่อใส่โมดูล Wi-Fi ตามด้วยคำสั่ง ifconfig และ iwconfig เพื่อสร้างเครือข่ายแอดฮอกขึ้นมา แต่เกิดปัญหาในขณะที่ insmod คือเมื่อเรียกแล้วคำสั่งจะค้างและทำงานไม่เสร็จ ดังรูปที่ 5-27 สำหรับวิธีการแก้ปัญหาในขณะที่ insmod ทำงานค้างอยู่ ผู้พัฒนาใช้คำสั่ง ./adb shell เพื่อเรียกอีก shell ขึ้นมา จากนั้นได้ลองใช้คำสั่ง lsmod พบว่าโมดูล bcm4329 กำลัง Loading อยู่ ดังรูปที่ 5-28 (ปกติหาก insmod สำเร็จจะต้องเป็น Live)



รูปที่ 5-27 ปัญหาที่เกิดจากการใช้คำสั่ง insmod

รูปที่ 5-28 สถานะของโมดูล bcm4329

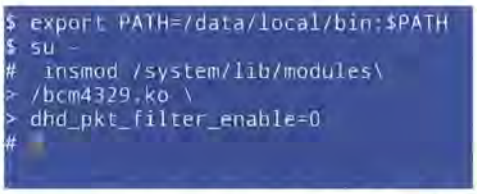
ในการหาสาเหตุผู้พัฒนาใช้คำสั่ง dmesg เพื่อแสดง message buffer ของ kernel และพบว่า kernel มีการมองหาไฟล์ /system/etc/firmware/fw_bcm4329.bin แต่เมื่อใช้คำสั่ง ls พบว่าไม่มีไฟล์ดังกล่าวอยู่ ในการค้นหาพบค้นหาพบไฟล์ตามไดเร็กทอรี่ /system/vendor/firmware/fw_bcm4329.bin จึงคัดลอกไฟล์ fw_bcm4329.bin จากไดเร็กทอรี่ /system/vendor/firmware มาไว้ที่ไดเร็กทอรี่ /system/etc/firmware แล้ว insmod พบว่าสามารถ insmod ได้สำเร็จ และสามารถใช้คำสั่ง ifconfig และ iwconfig ในการสร้างเครือข่ายแอดฮอกขึ้นมาได้เอง



รูปที่ 5-29 การใช้คำสั่ง insmod และสถานะของโมดูลที่พร้อมใช้งาน

5.4.2 ปัญหาโทรศัพท์มือถือ Google Nexus One ไม่รับ Broadcast UDP Packet ขณะเครื่องพักหน้าจอ

ปัญหานี้เกิดกับโทรศัพท์ในระบบปฏิบัติการมือถือแอนดรอยด์ทุกเครื่องที่ใช้ Wireless Chipset ของ Broadcom เพราะว่า Driver ของ Wireless Chipset ตัวนี้จะกรอง Broadcast UDP Packet ที่ได้รับมาออกขณะที่มือถือกำลังพักหน้าจออยู่ [28] สำหรับวิธีการแก้ปัญหาสามารถทำได้โดยการใส่พารามิเตอร์ dhd_pkt_filter_enable=0 เข้าไปในขณะที่ใช้คำสั่ง insmod โมดูล bcm4329



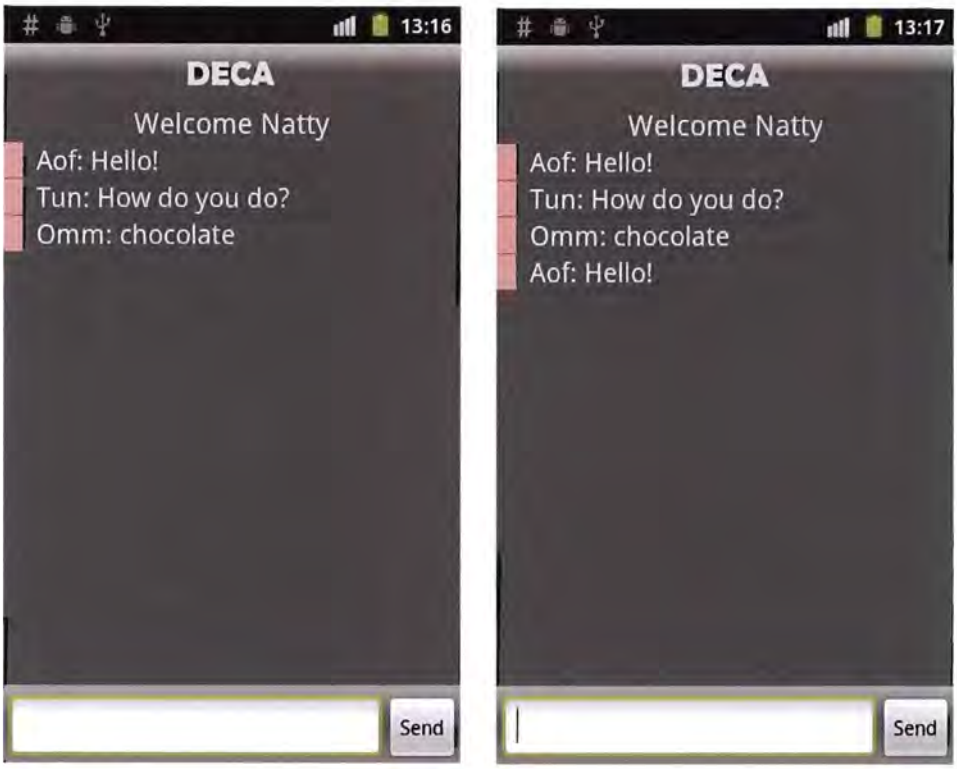
รูปที่ 5-30 การ insmod โมดูล bcm4329 โดยโดยการใส่พารามิเตอร์ dhd_pkt_filter_enable=0

5.4.3 ปัญหา Packet เดียวกันในแต่ละโหนดหมดอายุในเวลาที่ต่างกัน

โปรโตคอล DECA จะมี Broadcast Packet ซึ่งถูกตั้งเวลาหมดอายุไว้ ซึ่งใน Simulation เวลาของแต่ละโหนดจะเท่ากันหมดเพราะเป็นการใช้ Global Clock แต่เมื่อนำ DECA มาทำงานในสภาพแวดล้อมจริงซึ่งแต่ละโหนดจะใช้ Local Clock ของตัวเองทำให้เกิดปัญหา Packet เดียวกันในแต่ละโหนดหมดอายุในเวลาที่ต่างกัน

โดยปกติส่วนต่างของเวลาหมดอายุจะน้อยมากทำให้โปรโตคอล DECA ทำงานได้เป็นปกติ แต่ในกรณีที่มีการรับ-ส่งข้อมูลจำนวนมากในบางกรณีจะเกิดเหตุการณ์ผิดปกติ คือ โหนดที่ Packet ยังไม่หมดอายุส่ง Packet ดังกล่าวไปให้โหนดที่เคยได้รับ Packet นั้นแต่สำหรับโหนดนั้น Packet หมดอายุแล้ว ซึ่งส่งผลต่อค่าใช้จ่ายในการทำงานของโพรโทคอล

แก้ไขปัญหาโดยทำการดัดแปลงตารางเก็บข้อมูล Broadcast Packet ให้เมื่อสั่งลบข้อมูล Broadcast Packet แล้วข้อมูลดังกล่าวจะถูกเก็บอยู่ในตารางในสภาพ Inactive อีกระยะหนึ่งก่อนที่จะถูกลบออกจากตารางจริง ๆ การทำเช่นนี้จะช่วยป้องกันเหตุการณ์ผิดปกติดังกล่าวได้เพราะเมื่อโหนดที่ Packet ยังไม่หมดอายุตรวจพบโหนดที่ไม่มี Packet ดังกล่าวเพราะ Packet นั้นหมดอายุไปแล้ว ก่อนที่โหนดจะส่ง Packet นั้นออกไปจะทำการตรวจในตารางก่อนว่า Packet ดังกล่าวอยู่ในสภาพ Inactive หรือไม่ ถ้าอยู่ในสภาพ Inactive ก็จะไม่ส่ง Packet ดังกล่าวออกไป



รูปที่ 5-31ปัญหา Packet เดียวกันในแต่ละโหนดหมดอายุในเวลาที่ต่าง

5.5 การทดลองและวิเคราะห์ผลการทดลอง

การทดลองการทำงานของโพรโทคอล DECA ในบทนี้แบ่งออกเป็นสองส่วน คือ ส่วนแรกเป็นการทดสอบการทำงานของโพรโทคอล DECA บนระบบปฏิบัติการแอนดรอยด์ว่าสามารถทำงานได้อย่างที่มีการออกแบบไว้อย่างสมบูรณ์หรือไม่ และถัดมาจะเป็นการทดลองการวัดสมรรถนะของโพรโทคอล DECA ที่พัฒนาลงอุปกรณ์ที่มีระบบปฏิบัติการแอนดรอยด์ทำงานร่วมกับคอมพิวเตอร์พกพาที่มีระบบปฏิบัติการที่แตกต่างกันผ่านแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย

5.5.1 Functional Test

การทดลองในขั้นนี้เป็นการทดสอบพฤติกรรมของโปรโตคอล DECA ว่ามีพฤติกรรมตามที่ผู้พัฒนาได้ออกแบบไว้หรือไม่ โดยมีการทดลองการทำงานตามขั้นตอนต่างๆ ดังต่อไปนี้

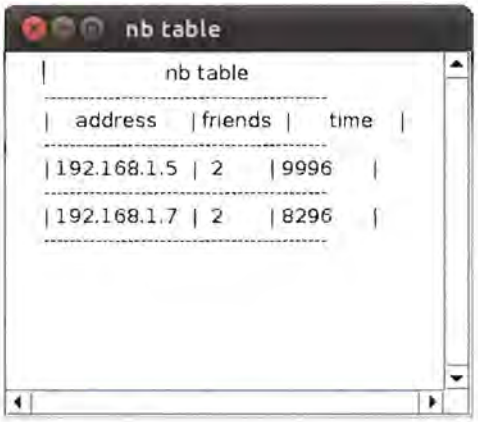
1) เมื่อโหนดรับบีคอนมาแล้ว เครื่องทำการ Update *nbtable* ของตนเอง
2) เมื่อข้อมูลเพื่อนบ้านใน *nbtable* ของตนเองหมดอายุ จะลบข้อมูลเพื่อนบ้านดังกล่าวออกไปจากตาราง *nbtable*
3) เมื่อโหนดได้รับข้อความใหม่แล้วและตนเองเป็นโหนดที่ถูกเลือกจะทำการกระจายข้อความดังกล่าวออกไปทันที
4) เมื่อโหนดได้รับข้อความใหม่แล้วแต่ตนเองไม่ได้เป็นโหนดที่ถูกเลือกจะทำการเก็บข้อความใส่ *ttable*
   i. เมื่อได้ยินโหนดอื่นแพร่ข้อความดังกล่าวแล้ว ทำการลบข้อความดังกล่าวใน *ttable* ของตนเอง
   ii. เมื่อข้อความดังกล่าวใน *ttable* หมดอายุ ทำการกระจายข้อความดังกล่าว

5) เมื่อโหนดรับบีคอนหรือ Broadcast Message มาแล้วตรวจพบว่าขาดข้อความบางข้อความ จะทำการส่งบีคอนออกไป เพื่อให้เครื่องอื่นทราบว่าขาดข้อความ

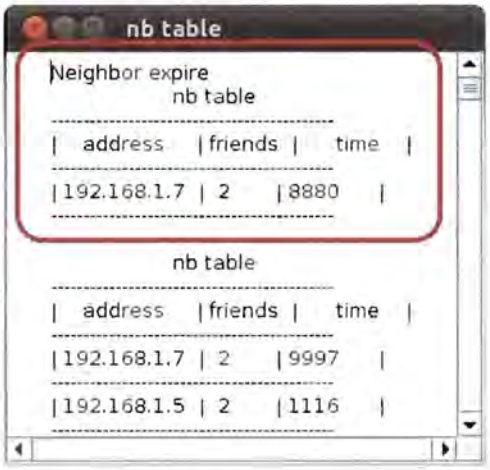
6) เมื่อพบว่าโหนดเพื่อนบ้านขาดข้อความจะทำการเพิ่มข้อความเข้าไปใน *ttable* และเมื่อเวลารอของข้อความดังกล่าวหมดจะทำการกระจายข้อความดังกล่าวออกไปโดยไม่เลือกโหนดส่งต่อ

7) เมื่อข้อความใน *rptable* ของตนเองหมดอายุ จะทำการลบข้อความดังกล่าวออกจาก *rptable*

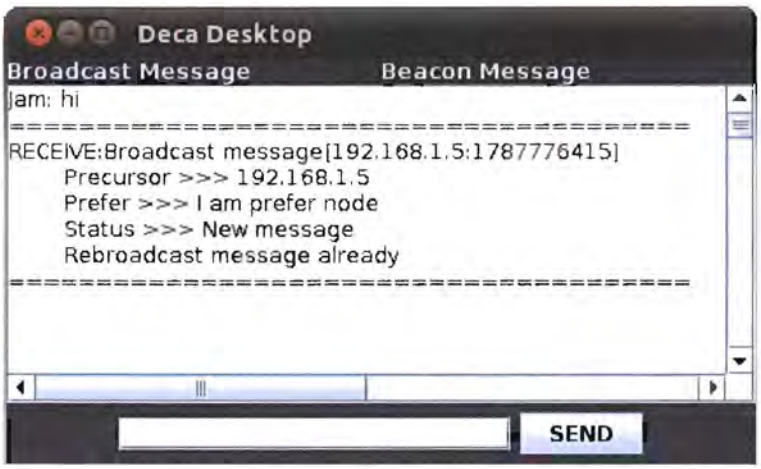
ผลการทดลองการพัฒนาโพรโทคอล DECA บนระบบปฏิบัติการณ์แอนดรอยด์สามารถทำงานได้อย่างสมบูรณ์ทุกกรณี โดยพิจารณาจากผลการทดลอง ดังรูปที่ 5-32 ถึง 5-39



รูปที่ 5-32 ผลการทดลอง *nbtable* หลังรับบีคอน



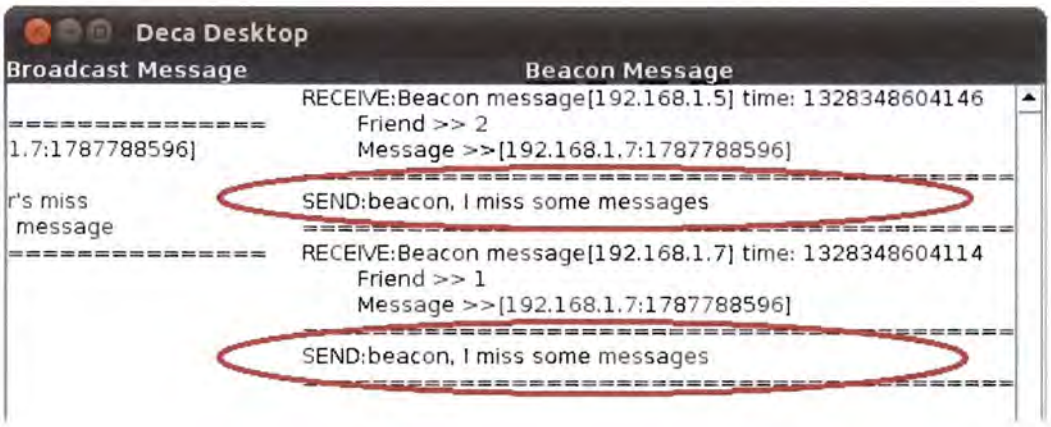
รูปที่ 5-33 ผลการทดลองลบเพื่อนบ้านที่หมดอายุออกจาก *nbtable*



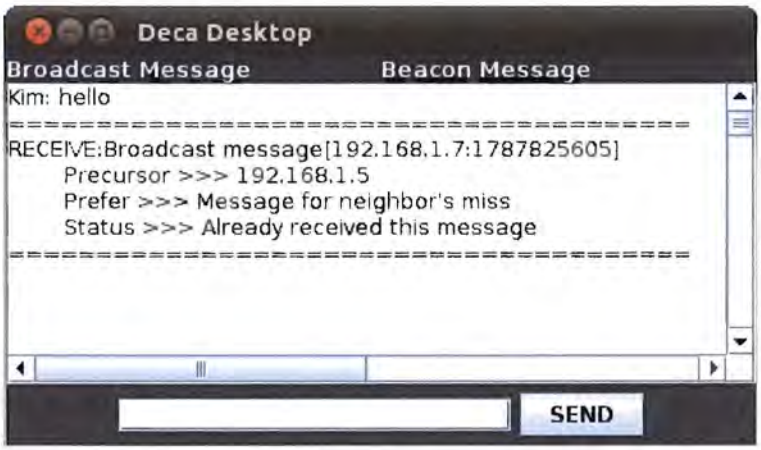
รูปที่ 5-34 ผลการทดลองการกระจายข้อมูลซ้ำทันทีเมื่อโหนดถูกเลือกให้เป็นโหนดส่งต่อข้อความ

รูปที่ 5-35 การทดลองการลบข้อความออกจาก *ttable* เมื่อได้ยินผู้อื่นกระจายข้อความนั้นแล้ว

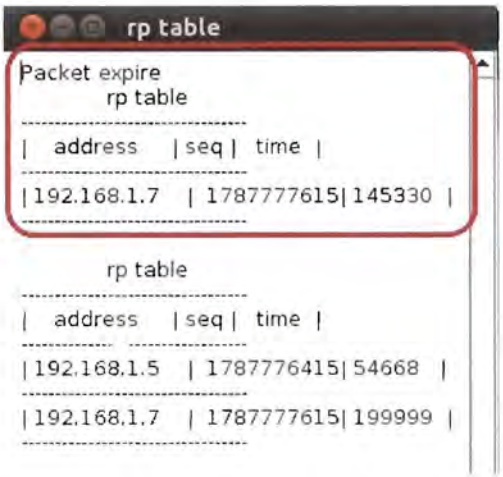
รูปที่ 5-36 ผลการทดลองการลบข้อความออกจาก *ttable* เมื่อแพร่ข้อความนั้นแล้ว



รูปที่ 5-37 ผลการทดลองการส่งบีคอนหาเพื่อนบ้านเพื่อร้องขอข้อความที่ตนเองไม่ได้รับ



รูปที่ 5-38 ผลการทดลองการได้รับข้อความที่ไม่ได้เลือกโหนดส่งต่อข้อความ



รูปที่ 5-39 ผลการทดลองการลบข้อความที่หมดอายุออกจาก *rptable*

5.5.2 การทดลองการทำงานร่วมกับอุปกรณ์อื่น

การทดลองนี้เป็นการวัดสมรรถนะของโพรโทคอล DECA ที่พัฒนาลงอุปกรณ์ที่มีระบบปฏิบัติการแอนดรอยด์ทำงานร่วมกับคอมพิวเตอร์พกพาที่มีระบบปฏิบัติการที่แตกต่างกันผ่านแอพพลิเคชั่นสำหรับการกระจายข้อความอย่างง่าย มีรายละเอียดดังต่อไปนี้

1) **ตัววัดสมรรถนะ** ใช้ค่าความเชื่อได้ในการวัดสมรถนะ เนื่องจากเป็นจุดประสงค์หลักในการออกแบบโพรโท-คอล โดยการวัดค่าความเชื่อได้นั้น จะเป็นอัตราส่วนของจำนวนโหนดที่ได้รับข้อความกับจำนวนข้อความที่โหนดต้นทางได้ส่งออกไป โดยเฉลี่ยจากข้อความทั้งหมดที่ถูกส่งออกไปจากโหนดนั้น เพื่อแสดงถึงสมรรถนะในการทำงานโพรโทคอล

2) **อุปกรณ์ที่ใช้ในการทดลอง** ประกอบด้วยอุปกรณ์ต่างๆ ดังต่อไปนี้

- *คอมพิวเตอร์แบบพกพา* จำนวน 4 เครื่อง ประกอบด้วยระบบปฏิบัติการ Linux Ubuntu 10.10 จำนวน 3 เครื่อง และระบบปฏิบัติการ OSX 10.7 จำนวน 1 เครื่อง
- *โทรศัพท์มือถือ* Google Nexus One จำนวน 2 เครื่อง ทั้งหมดทำงานบนระบบปฏิบัติการแอนดรอดย์ CyanogenMod 7

3) **สภาพแวดล้อมในการทดลอง** ใช้พื้นที่รอบตึกเจริญวิศวกรรม และตึกข้างเคียง ในพื้นที่ของคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย โดยที่แต่ละโหนดที่การเคลื่อนที่แบบสุ่มตามเส้นทาง ดังรูปที่ 5-40 เพื่อให้สัญญาณในการติดต่อมีการเชื่อมต่อเป็นช่วงๆ ไม่สามารถเชื่อมต่อกันได้เป็นเวลานาน โดยจะใช้คอมพิวเตอร์แบบพกพาเป็นโหนดหยุดนิ่งบริเวณใต้ตึกเจริญวิศวกรรมจำนวน 2 เครื่อง (Node 1 และ Node 2) เพื่อทำหน้าที่ในการรับข้อความและส่งต่อให้โหนดอื่น และเป็นมาตรฐานในการรับข้อความได้สูงที่สุด

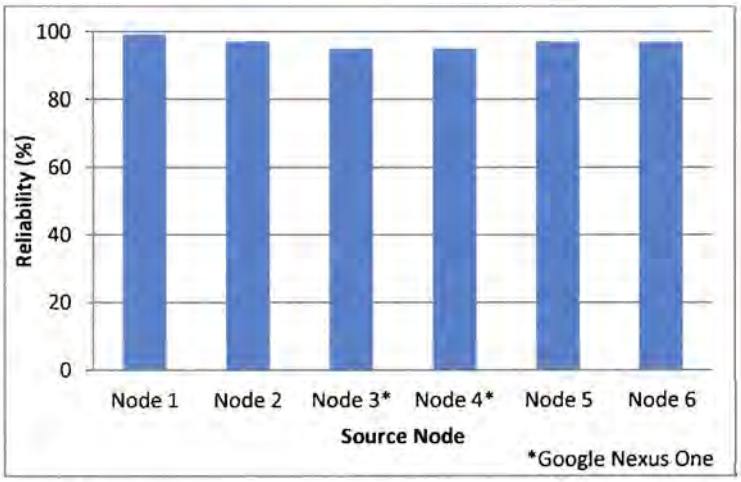
รูปที่ 5-40 เส้นการการเคลื่อนที่ของโหนด และบริเวณที่ใช่ในการทดลอง

4) การตั้งค่าในการทดลอง ทุกโหนดจะมีการส่งข้อความทุกๆ 20 วินาที ช่วงเวลาที่ใช้ในการทดลองประมาณ 500 วินาที หรือมีการกระจายประมาณ 25 ข้อความต่อโหนด การตั้งค่าอื่นๆเป็นไปตามตารางที่ 5-1

ตารางที่ 5-1 การตั้งค่าในการทดลอง

อายุของข้อความ	200 วินาที
ช่วงเวลาการส่งบีคอน	4.0-4.5 วินาที
ขนาดของข้อความ	2048 ไบต์
เวลารอสูงสุด	0.5 วินาที
ช่วงเวลาการส่งข้อความ	20 วินาที

ผลการทดลองพบว่าโหนดที่เป็นคอมพิวเตอร์แบบพกพามีค่าความเชื่อถือได้ในการส่งข้อความเฉลี่ย 97.65% และโหนดต้นทางที่เป็นโทรศัพท์มือถือมีค่าความเชื่อถือได้เฉลี่ย 95% โดยโหนดหยุดนิ่งมีค่าความเชื่อถือได้สูงสุดที่ 99.26% ผลการทดลองของโหนดอื่นๆเป็นไปตามรูปที่ 5-41 จากการทดลองพบว่าค่าความเชื่อได้ที่แตกต่างกันนั้นเกิดจากระยะการส่งสูงสุดของโทรศัพท์มือถือมีระยะที่สั้นกว่าคอมพิวเตอร์แบบพกพาจึงทำให้ในกรณีที่ต้องมีการส่งข้อความให้แก่โหนดที่ไม่ได้รับข้อความทำได้ไม่สำเร็จ คือ เมื่อมีการตรวจพบโหนดเพื่อนบ้านที่ไม่ได้รับข้อความ โหนดที่เป็นโทรศัพท์มือถือมีการส่งข้อความนั้นออกไปยังโหนดที่มีการร้องขอแต่กลับไม่ได้รับข้อความนั้น



รูปที่ 5-41 ผลการทดลองค่าความเชื่อได้ของโหนดต้นทางแต่ละโหนด

บทที่ 6 สรุปผลการวิจัย

6.1 สรุปผลการวิจัย

เนื่องจากในปัจจุบันความก้าวหน้าทางเทคโนโลยีทำให้ผู้ทำวิจัยเกิดความสนใจในการพัฒนาความสามารถในด้านแอพพลิเคชั่นของรถยนต์ หรือระบบจราจรอัจฉริยะ (Intelligent Transport System : ITS) ซึ่งต้องการช่องทางในการสื่อสารและแลกเปลี่ยนข้อมูลระหว่างกัน โพรโทคอลการกระจายอย่างเชื่อถือได้สำหรับเครือข่ายแอดฮอกบนยานพาหนะก็ถือเป็นหนึ่งในกุญแจสำคัญสำหรับการแลกเปลี่ยนข้อมูลในวงกว้าง โดยโพรโทคอลที่ดีนั้นต้องพิจารณาถึงค่าความเชื่อถือได้ ค่าใช้จ่ายที่เกิดขึ้น และความเร็วในการกระจายข้อมูล

ในงานวิจัยนี้ได้เสนอโพรโทคอลการกระจายที่ความเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายแอดฮอกบนยานพาหนะ (DECA) ซึ่งถูกออกแบบให้ใช้เพียงข้อมูลความหนาแน่นจากการแลกเปลี่ยนบีคอนเท่านั้น เพื่อความยืดหยุ่นในการทำงานสูงที่สุดเมื่อเปรียบกับโพรโทคอลอื่นที่ต้องการข้อมูลตำแหน่งหรือจีพีเอสในการทำงาน นอกจากนี้ยังรองรับการทำงานที่มีการเชื่อมต่อเป็นช่วงๆซึ่งมักจะพบบนเครือข่ายแอดฮอกบนยานพาหนะ โดยการออกแบบให้มีการทำงานแบบ Store-and-Forward และโหนดสามารถตรวจสอบและร้องขอข้อมูลที่ตนเองไม่ได้รับจากการแลกเปลี่ยนบีคอนซึ่งมีข้อความตอบรับ (Acknowledgement) เนื่องจากการแลกเปลี่ยนบีคอนถูกใช้การแลกเปลี่ยนข้อมูลพื้นฐานที่สำคัญระหว่างโหนด ดังนั้นจึงมีการออกแบบการส่งบีคอนแบบปรับตัวแบบเชิงเส้นหรือ LIA เพื่อให้มีค่าใช้จ่ายในการทำงานต่ำที่สุด

ในการออกแบบและพัฒนา DECA ให้มีความสมบูรณ์มากที่สุดจึงมีการพัฒนาพร้อมกับการวัดสมรรถนะในระดับต่างๆ ได้แก่ การทดลองโดยโปรแกรมจำลองเครือข่ายที่ใช้สถานการณ์จำลองสภาพการจราจรในถนนเสมือนจริงอย่างง่าย การทดลองโดยโปรแกรมจำลองเครือข่ายที่ใช้สถานการณ์จำลองสภาพการจราจรในถนนเสมือนจริงจากแผนที่กรุงเทพมหานคร และการทดลองบนอุปกรณ์จริง

ในการทดลองโดยโปรแกรมจำลองเครือข่ายที่ใช้สถานการณ์จำลองสภาพการจราจรในถนนเสมือนจริงอย่างง่าย DECA ให้ผลการทดลองที่มีสมรรถนะสูงที่สุด ซึ่งมีความเร็วในการกระจายข้อมูลเทียบเท่า Simple Flooding และมีค่าใช้จ่ายในทำงานที่ต่ำมากเมื่อเปรียบเทียบกับโพรโทคอลอื่นที่ใช้ในการทดลอง เพื่อให้ครอบคลุมในการทำงานนอกจากการทดลองบนถนนเสมือนจริงอย่างง่าย จึงมีการศึกษาวิธีการปรับช่วงเวลาการทำบีคอนที่เหมาะสม โดยนำวิธีการเรียนรู้แบบ Machine Learning และวิธีการเรียนรู้ทางด้านสถิติเข้ามาทดลองเพื่อให้โหนดมีค่าใช้จ่ายต่ำที่สุด โดยที่โพรโทคอลยังมีค่าใช้จ่ายที่ต่ำที่สุด และมีการศึกษาวิธีจัดการบัฟเฟอร์ที่เหมาะสมสำหรับโพรโทคอล

ขั้นตอนถัดมาการทดลองโดยโปรแกรมจำลองเครือข่ายที่ใช้สถานการณ์จำลองสภาพการจราจรในถนนเสมือนจริงจากแผนที่กรุงเทพมหานคร ซึ่งมีการทดลองในสถานการณ์ที่มีความหนาแน่นของโหนดสูง ผู้วิจัยมีการปรับปรุงการทำงานเพื่อของโพรโทคอล เพื่อให้โพรโทคอลทำงานได้อย่างมีประสิทธิภาพในสถานการณ์ที่มีความหนาแน่นสูง โดยการแก้ไขวิธีการคำนวณเวลารอซึ่งใช้ในการหลีกเลี่ยงการชนกันของข้อมูล และเสนอวิธีการจำกัดปริมาณการร้องข้อมูลที่โหนดไม่ได้รับ ซึ่งส่งผลให้ DECA ที่ได้รับปรับปรุงแล้วมีสมรรถนะในการทำงานสูงที่สุด โดยที่มีค่าใช้จ่ายในการทำงานต่ำที่สุด และเหมาะสมที่จะนำไปใช้งานในสถานการณ์ทั้งที่มีความหนาแน่นต่ำ และความหนาแน่นสูงมาก

ในขั้นตอนสุดท้ายมีการพัฒนา DECA ลงบนอุปกรณ์คอมพิวเตอร์แบบพกพา โดยมีการเสนอการขั้นตอนการเลือกโหนดส่งต่อข้อมูลด้วยวิธีการโหวตจากระดับสัญญาณ RSSI เพื่อแก้ปัญหาในการทำงานบนเครือข่ายที่มีการเชื่อมต่อแบบอสมมาตร ซึ่งมีประสิทธิภาพในการทำงานบนเครือข่ายที่มีการเชื่อมต่อแบบอสมมาตรมากกว่าการเลือกด้วยข้อมูลความหนาแน่นเพียงอย่างเดียว การออกแบบวิธีการโหวตจากระดับสัญญาณ RSSI นั้นมีพื้นฐานในการทำงานเช่นเดียวกับการทำงานของ DECA นอกจากนั้นยังมีการพัฒนา DECA ลงบนอุปกรณ์ที่มีระบบปฏิบัติการแอนดรอย์และทำการทดลองร่วมกับคอมพิวเตอร์แบบพกพา ผลการทดลองแสดงให้เห็นว่า DECA สามารถทำงานได้ดีบนอุปกรณ์จริง และสามารถทำงานร่วมกับอุปกรณ์หลากหลายชนิดได้

ดังนั้น DECA จึงได้รับการพัฒนา และปรับปรุงจนมีความสามารถในการทำงานได้ดีทั้งบนโปรแกรมจำลองเครือข่าย และบนอุปกรณ์จริงดังที่ออกแบบไว้ โดยผลการทดลองโพรโทคอลสามารถให้ค่าความเชื่อถือได้ที่สูง มีความเร็วในการทำงานสูงที่สุด และมีค่าใช้จ่ายน้อยที่สุดเมื่อเปรียบเทียบกับโพรโทคอลที่มีการศึกษา

6.2 ข้อจำกัด

เนื่องจากโพรโทคอล DECA ต้องใช้ข้อมูลข้อความตอบรับจากบีคอนในการตรวจสอบข้อมูลที่ได้รับ ดังนั้นบีคอนจึงมีขนาดเพิ่มขึ้นเมื่อข้อความในระบบมีมากขึ้น ซึ่งเมื่อนำมาใช้งานบนระบบที่มีความหนาแน่นสูงย่อมจะเกิดปัญหาเรื่องค่าใช้จ่ายของบีคอนในการทำงาน หากสามารถทำให้ขนาดของบีคอนมีขนาดคงที่ย่อมจะทำให้สามารถคาดเดาพฤติกรรมของโพรโทคอลได้ดีขึ้น

นอกจากนี้แม้ว่า DECA จะได้รับการพัฒนาลงบนอุปกรณ์จริง และมีการทดลองเพื่อปรับปรุงสมรรถนะในการทำงานให้สูงที่สุดบนอุปกรณ์จริง แต่ยังขาดการทดสอบลงบนรถยนต์จริง ซึ่งมีความเร็ว และมีการเปลี่ยนแปลงการเคลื่อนที่ที่รวดเร็ว ซึ่งส่งผลต่อประสิทธิภาพในการสื่อสารมากกว่าการทดสอบในงานวิจัย ผลการทดลองอาจจะแสดงถึงปัญหาที่สามารถค้นพบได้งานวิจัยนี้

6.3 ข้อเสนอแนะ

DECA แสดงผลการทำงานได้ดีทั้งบนการทดลองในโปรแกรมจำลอง และในการทดลองบนอุปกรณ์ในสถานการณ์อย่างง่าย แต่หากต้องการนำใช้งานจริงจำเป็นจะต้องมีการทดสอบร่วมกับแอพพลิเคชั่น และบนเครือข่ายจริงที่มีอุปกรณ์ตามมาตรฐานการทำงานของเครือข่ายแอดฮอกบนยานพาหนะ เช่น IEEE802.11P และมีการทดสอบบนรถยนต์หรือพาหนะจริงที่มีโครงข่ายจำนวนมาก เพื่อแสดงให้เห็นปัญหาที่จะเกิดขึ้น และการปรับปรุงการทำงานของโพรโทคอลให้มีสมรรถนะที่เหมาะสมกับแอพพลิเคชั่นที่นำไปใช้ได้

นอกจากนี้การพัฒนาโพรโทคอลโดยให้มีการทำงานร่วมกับเครือข่ายระดับอื่น เช่นการนำบีคอนทำงานร่วมกับเครือข่ายในระดับ MAC โดยแนบบีคอนไปกับการทำ CTS และ RTS จะช่วยลดค่าใช้จ่ายที่เกิดขึ้นจากการส่งบีคอนได้ หรือวิธีการเข้ารหัสส่วนที่เก็บข้อความตอบรับ (Acknowledgement) ของข้อมูลที่ได้รับซึ่งจะเพิ่มขึ้นเมื่อมีข้อมูลเพิ่มขึ้นให้มีค่าคงที่ได้ จะส่งผลให้ค่าใช้จ่ายจากบีคอนลดลง

ผลงานที่ได้รับจากโครงการ

1. ณวุฒ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA : Density-aware reliable broadcasting in vehicular ad-hoc networks.* IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology Conference **2553**; 7: 598-602.
2. ชญานิน ไทยนะ, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *A study of adaptive beacon transmission on vehicular ad-hoc networks.* International Conference on Communication Technology **2554**; 13: 597-602.
3. วิภาวี วิริยพงษ์สุกิจ, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *A novel packet dropping policy for vehicular ad-hoc networks.* International Conference on Communication Technology **2554**; 13: 603-608.
4. ณัฐวิทย์ กมลธรรม, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *Improving reliable broadcast over asymmetric VANETs based on RSSI voting algorithm.* International symposium on Intelligent Signal Processing and Communication Systems **2554**;1-6.
5. ณฐกร นพคุณวิจัย, สุขุมาลย์ อาชาสันติสุข, กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA on android : reliable broadcasting in ad-hoc network on android platform.* ICT International Senior Project Conference **2555**;.
6. กุลิสร์ ณ นคร, กุลธิดา โรจน์วิบูลย์ชัย. *DECA-bewa: density-aware reliable broadcasting on vehicular ad-hoc networks.* IEICE Transactions on Communications **2555**:. (Under Reviewing)

รายการอ้างอิง

[1] Conti, M., and Giordano, S. Multihop Ad Hoc Networking: The Reality. IEEE Communications Magazine. 45, 4 (April 2007): 88-95.

[2] Kompfner, P. Cooperative Vehicle-Infrastructure System (CVIS). [online]. 2009. Available from: http://cvisproject.org [2010, October 28]

[3] Hofmann-Wellenhof, B., Lichtenegger, H., and Collins, J. GPS theory and practice. New York, USA: 2001.

[4] PIARC. The intelligent transport system handbook. New York, USA : Thomson press, 2004.

[5] Ni, S., Tseng, Y., Chen, Y., and Sheu, J. The broadcast storm problem in a mobile ad hoc network. Proc. ACM international conference on Mobile Computing and Networking (MobiCom'99), Seattle, USA: IEEE, 1999.

[6] Naumov, V., Baumann, R., and Gross, T. An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces. Proc. ACM the 7th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc'06), Florence, Italy: ACM, 2006.

[7] Wisitpongphan, N., Bai, F., Mudalige, P., and Tonguz, O. K. On the routing problem in disconnected vehicular ad hoc networks. IEEE the 26th International Conference on Computer Communications (INFOCOM'07), Anchorage, Alaska, USA: IEEE, 2007.

[8] Williams, B., and Camp, T. Comparison of broadcasting techniques for mobile adhoc networks, Proc. ACM the 3rd ACM international symposium on Mobile ad hoc networking and computing (MobiHoc'02), Lausanne, Switzerland: ACM, 2002.

[9] Pongthawornkamol, T., Nahrstedt, K. and Wang, G. HybridCast: A hybrid probabilistic/deterministic approach for adjustable broadcast reliability in mobile wireless ad hoc networks, IEEE International Conference on Communications (ICC'09), Dresden, Germany: IEEE, 2009

[10] Nekovee, M., and Bjarni, B. B. Reliable and efficient information dissemination in intermittenly connected vehicular ad hoc networks, IEEE the 65th Vehicular Technology Conference (VTC'07-Spring), Dublin, Ireland: IEEE, 2007.

[11] Ros, J. F., Ruiz, P. M., and Stojmenovic, I. Reliable and efficient broadcasting in vehicular ad hoc networks, IEEE the 69th Vehicular Technology Conference (VTC'09-Spring), Barcelona, Spain: IEEE, 2009.

[12] Khan, A. A., Stojmenovic, I., and Zeguia, N. Parameterless broadcasting in static to highly mobile wireless ad hoc, sensor and actautor networks, IEEE the 22nd International Conference on Advanced Information Networking and Applications (AINA'08), Okinawa, Japan: IEEE, 2008.

[13] Tonguz, O. K., Wisitpongphan, N., and Bai, F. DV-CAST: A distributed vehicular broadcast protocol for vehicular ad hoc networks. Proc. IEEE Wireless Communications. 17 (April, 2010).

[14] Naumov, V., and Gross, T. R. Connectivity-Aware Routing (CAR) in Vehicular Ad-hoc Networks, IEEE the 26th International Conference on Computer Communications (INFOCOM'07), pp.1919-1927. Anchorage, Alaska, USA: IEEE, 2007.

[15] IEEE. Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications. IEEE standard for information technology telecommunications and information exchange between systems local and metropolitan area networks specific requirements. (June 2007).

[16] Boukerche, A., Rezende, C., and Pazzi, R. W. Improving Neighbor Localization in Vehicular Ad Hoc Networks to Avoid Overhead from Periodic Messages, IEEE Global Telecommunications Conference (GLOBECOM'09), Honolulu, Hawaii, USA: IEEE, 2009.

[17] Ke, M., Nenghai, Y., and Bin, L. A new packet dropping policy in delay tolerant network, IEEE the 12th International Conference on Communication Technology (ICCT'10), pp.378-381, Nanjing, China, November 11-14, 2010.

[18] Krifa, A., Barakat, C., and Spyropoulos, T. Optimal Buffer Management Policies for Delay Tolerant Networks, IEEE the 5th Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'08), San Francisco, June 16-20, 2008.

[19] Farnoud (Hassanzadeh), F. and Valaee, S. Reliable broadcast on safety messages in vehicular adhoc network, IEEE the 28th International Conference on Computer Communications (INFOCOM'09), Rio de Janeiro, Brazil: IEEE, 2009.

[20] German Aerospace Center (DLR). Simulation of Urban MObility (SUMO) [online]. (2010). Available from: http://sumo.sourceforge.net [2010, October 28]

[21] Varadhan, K. The Network Simulator (NS-2). [online]. (2010). Availabble from: http://www.isi.edu/nsnam/ns [2010, October 28]

[22] Piorkowski, M., Raya, M., and Hubaux, J. P. Traffic and Network Simulation Environment (TraNS). [online]. (2008). Available from: http://trans.epfl.ch [2010, October 28]

[23] Abraham, John. The network simulator NS-3 [online]. 2011. Available from : http:/www.nsnam.org [2011, March 13]

[24] T. R. Henderson, S. Roy, S. Floyd and G. F. Riley. ns-3 Project Goals [online]. 2006. Available from : http://www.nsnam.org/docs/meetings/wns2/wns2-ns3.pdf [2006, October 10]

[25] E. Weingartner, H.vom Lehn and K. Wehrle. A Performance Comparison of Recent Network Simulators. In IEEE International Conference on Communications (ICC'09), pp.1-5. 2009.

[26] Google Inc. Android Developers [online]. 2011. Available from: http://developer.android.com/index.html [2011, August 31]

[27] Rabie Khodr Jradi and Lasse Seligmann Reedtz. Ad-hoc network on Android [online]. 2011. Available from: http://adhoc-on-android.googlecode.com /files/bachelorthesis.pdf [2011, August 31]

[28] Jordi Cucurull and Paul Gardner-Stephen. *Reception of UDP packets in sleep mode* [online]. 2012. Available from: https://groups.google.com/forum/#!msg/android-platform/OpbSdp9FTmA/Y0zK8AfbAY1J [2012, February 22]

ภาคผนวก

ผลงานวิจัยที่ได้รับการตีพิมพ์

ชื่องานวิจัย : "DECA: Density-Aware Reliable Broadcasting in Vehicular Ad-Hoc Networks"

คณะผู้วิจัย : ณวุฒ ณ นคร และ กุลธิดา โรจน์วิบูลย์ชัย

การตีพิมพ์ : บันทึกการประชุม "The seventh IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology Conference (ECTI-CON 2010)" ซึ่งจัดขึ้น ณ จังหวัดเชียงใหม่ ประเทศไทย ระหว่างวันที่ 19-21 พฤษภาคม 2553

DECA: Density-Aware Reliable Broadcasting in Vehicular Ad Hoc Networks

Nawut Na Nakorn
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
g52nnn@cp.eng.chula.ac.th

Kultida Rojviboonchai*
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
kultida.r@chula.ac.th

Abstract- **Reliable broadcasting in vehicular ad hoc networks is challenging due to its unique characteristics including intermittent connectivity and various vehicular scenarios (car traffic is possibly very dense in urban areas or very sparse on highways). In this paper, we propose a new reliable broadcast protocol which is suitable for such characteristics. To address the issue of various vehicular scenarios, our protocol performs periodic beaconing to gather local density information of 1-hop neighbors and uses such information to adapt its broadcast decision dynamically. Specifically, before broadcasting each message, a node selects a neighbor with the highest density. Upon the reception of the broadcast message, each node checks if it is the selected neighbor. If so, it is responsible for rebroadcasting the message immediately. Otherwise, it stores the message for possible rebroadcasting. To address the issue of intermittent connectivity, message information in beacons is used to discover new neighbors, which have not yet received the message. Simulation results show that, in both highway and urban scenarios, our proposed protocol outperforms other competing solutions in terms of reliability, overhead and speed of data dissemination.**

I. INTRODUCTION

Mobile Ad Hoc Networks (MANET) is a hot research topic and gains a lot of interests because nodes can communicate with no fixed infrastructure. The concept of multi-hop ad hoc network has been applied to the real world. One of the most successful areas that are rapidly penetrating the market is Vehicular Ad Hoc Networks (VANET) [1]. Applications for Intelligent Transportation System (ITS) such as road accident calls and emergency notification require an efficient reliable broadcasting scheme that can deliver broadcast messages to all vehicles in an area of interest.

To design an efficient reliable broadcast protocol, the following metrics should be taken into considerations. The first metric is reliability; that is, a broadcast message should be delivered to as many cars as possible in a certain area. The second metric is overhead; that is, delivery of a broadcast message to all cars should generate as few redundant messages as possible. The last metric is speed of data dissemination; that is, a broadcast message should be delivered to all cars as fast as possible. Although a broadcast message can reach all cars, it can be meaningless if it arrives too late. This metric is very critical for emergency-related services.

A traditional approach for broadcasting is simple flooding. This approach can provide very high speed of data dissemination. It is also simple as it does not require neighbor information. However, it does not perform well in dense area due to collisions leading to low reliability with a lot of redundant broadcast messages. This problem has been reported as broadcast storm problem [2] [3]. In sparse area, network can be disconnected intermittently. The simple flooding is useless in such situation as there is no neighbor being able to receive and convey the message [4].

In the context of MANET, more efficient broadcasting techniques have been proposed [5]. However, those techniques have considered random waypoint mobility model. Such model is unrealistic for VANET where cars usually move in the same or opposite directions, overtake other slower cars, stop at the intersection, or move fast causing intermittent connectivity. In the context of VANET, several reliable broadcast protocols [6] [8] have been proposed. These protocols employ store-and-forward technique to address the intermittent-connectivity issue. Their performance evaluation has been done on highway and/or urban scenarios.

Edge-aware epidemic protocol (FAEP) has been proposed in [6]. Each car has its geographical knowledge by means of GPS. Upon the receipt of a new broadcast message, a car randomly sets a waiting timeout. When the timeout expires, the car decides whether or not to rebroadcast with a probabilistic function. Higher probability will be given to cars in the edge of circulated broadcast. This protocol outperforms the simple flooding in terms of reliability and overhead. However, it provides slow speed of data dissemination due to its waiting time (According to simulation results shown in [6], it takes more than 30 seconds to deliver a broadcast message to the majority of cars).

AckPBSM [6] has been proposed. It is a modified version of PBSM [7], which is a parameterless broadcast protocol in static to highly mobile ad hoc networks. AckPBSM uses GPS to retrieve position knowledge. It employs 1-hop position information obtained by periodic beacons to construct Connected Dominating Sets (CDS). Upon the receipt of a broadcast message, nodes set waiting timeout before possible rebroadcasting. Nodes in CDS set shorter waiting time. To address intermittent connectivity issue, acknowledgements of broadcast messages are piggybacked in periodic beacons so that nodes can rebroadcast only if their neighbors have not received the broadcast messages. It is reported in [8] that AckPBSM outperforms PBSM [7] and DV-Cast [9] in terms of

*Corresponding author

978-1-4244-5607-9/10/$25.00 ©2010 ECTI

identifiers of all 1-hop neighbors and their local density. The list is updated every time a beacon is received. *Broadcast List* maintains identifiers of broadcast messages and their waiting timeouts. A broadcast message will be removed from the list by two timings. The first one is when the timeout expires and the node rebroadcasts the message. The second one is when the node hears other nodes retransmit the message. This is to minimize number of redundant retransmissions.

DECA uses periodic beacons. A beacon contains local density and identifiers of broadcast messages received so far. The local density is used for selecting the next rebroadcasting node. The identifiers of the received messages allow nodes to check whether they miss some messages that other neighbors have or other neighbors miss some messages that they have.

When a source node would like to broadcast a message, it will select the next rebroadcast node from its Neighbor List. The node that has the highest density will be selected. But if the node with the highest density is the precursor node, another node with the second highest density will be selected instead. However, only such selection mechanism cannot prevent the loop problem as shown in Fig. 2.

In Fig. 2, A receives a message and retransmits it with the selected node B. Then B receives the message and retransmits it with the selected node C. C has no other neighbors so C may retransmits the message with the selected node A. This causes loop problem and useless redundant retransmissions. To prevent such loop problem, the selected node also checks if it has already received the message before. If yes, it will not rebroadcast the message. As a result, unnecessary retransmissions will not happen.

By beaconing, a node can know whether its neighbors have missed some messages or the node itself has missed some messages. When the node receives a beacon from its neighbor, the node updates *Neighbor List* and check for any missing broadcast message. In the case that the node found its neighbor missing the message, it will put identifier of the message into *Broadcast List* and set a short waiting timeout. When the timeout expires, the node will retransmit the message and other nodes that hear this transmission will remove the message from their *Broadcast List*. In the case that the node found itself missing the message, it will send a beacon immediately allowing its neighbors which have the missing message to retransmit the message for itself.

III. Adaptive Beaconing Scheme

Beaconing is the most important solution for VANET protocols to discover and exchange local information. This information can help protocols to operate more accurate and more reliable. However, high beacon rate can degrade protocol performance. Fig. 3 shows number of data transmissions of

Figure 2. Loop Problem Scenario

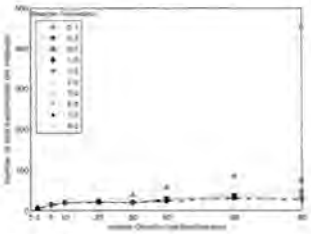


Figure 3. Number of broadcasting message transmissions

DECA at various beaconing intervals. As can be seen, high beacon frequency at 10Hz (0.1 second/beacon) can cause broadcast storm problem. A large number of data messages and beacon messages are dropped so the protocol needs to retransmit data and beacon messages to keep reliable status. However, this even causes more collision, leading to low speed of data dissemination and low reliability. Due to space limitation, the results are not shown.

By keep this in mind, we have tested DECA in various beacon intervals to obtain the optimum beacon interval for each density scenario. The optimum beacon interval means the highest beacon interval that provides the fastest data dissemination. With this experimental result, we propose for DECA to dynamically calculate Adaptive Beacon Interval (ABI) according to (1).

$$f(n)=min(MinInv + (C \times n), MaxInv) \quad (1)$$

where $f(n)$ is an optimum beacon interval (second/beacon) used by DECA, *MinInv* and *MaxInv* are constant values analyzed from experimental results (in this paper we use 1.5 and 7.0, respectively), n is number of neighbors, and C is a constant for adjusting function slope.

Other solutions to reduce beacon overhead can be found in [10] [11]. However, they are not designed for reliable broadcasting, therefore not effectively performed with DECA.

IV. Performance Evaluation

In order to evaluate performances of our protocol, we use a traffic simulator called Simulation of Urban Mobility (SUMO) [12] to generate realistic traffic situations. The vehicle traces obtained from SUMO are in XML format. It is converted to TCL format by a tool called Traffic and Network Simulation Environment (TraNS) [13]. Two traffic scenarios (highway and urban) are generated. In highway scenario, we generate a straight road with the length of 4 kilometers and two lanes per direction. In urban scenario, we generate a square grid of 2 kilometers for each side and one lane per direction. The vehicles are consisted of two groups; a group with the maximum velocity of 50 km/h and the other group with the maximum velocity of 80 km/h. Cars' density are varied from 2 veh/km to 80 veh/km. We implemented all of the following protocols in the well known network simulation NS-2.34 [14].

- DECA: our proposed reliable broadcast protocol with a fixed beacon interval at 2 seconds/beacon.
- DECA+ABI: our proposed reliable broadcast protocol with Adaptive Beacon Interval (ABI) as mentioned in Section III.
- Simple Flooding (SF): SF represents the protocol with the faster speed of data dissemination.
- Simple Flooding with Random (SFR): SFR is a modification of SF. Instead of rebroadcasting the message immediately after it receives the message, SFR waits for a random time before rebroadcasting. SFR is the simplest protocol that uses store-and-forward technique.
- AckPBSM: AckPBSM [8] represents the protocol which uses store-and-forward technique and uses waiting time to alleviate the broadcast storm problem. We choose AckPBSM as it is shown to provide the highest reliability among existing protocols we found in the literature. Its parameters have set as [8].

Unless stated otherwise, parameter settings for simulations are configured as indicated in TABLE I.

All protocols use IEEE 802.11 with collision for MAC and their bandwidth are 2 Mbps. The following three metrics are considered for performance evaluation.

- *Reliability* is measured as a percentage of number of nodes that received the message at the end of simulation. 100% reliability means all nodes in the scenario received the message. In some scenarios, reliability may not reach 100% due to some vehicles are far apart from others.
- *Overhead* is measured in two types as number of data transmissions for one message broadcasting and as number of beacon messages that all nodes generated in each runs.
- *Speed of Data Dissemination* is measured as a percentage of number of nodes that received the message at time t. Equation (2) is used for plotting graphs.

$$y(t) = \frac{\sum_{i=0}^{t} r_i}{n} \times 100 \qquad (2)$$

where r_i represents number of received node at the time i and n is total number of vehicles in the scenario.

TABLE I
PARAMETER SETTING

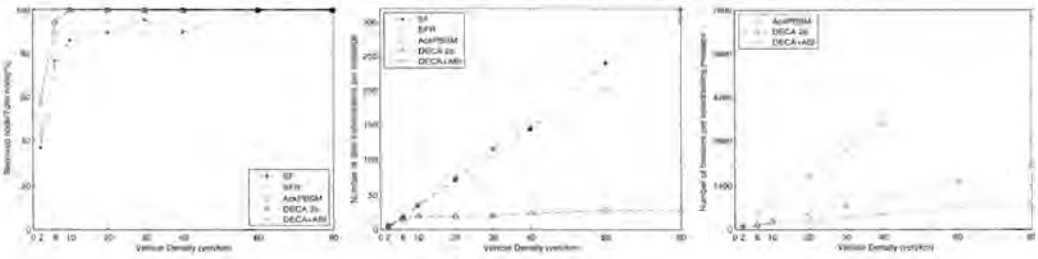
Packet life time[a]	10 s.
Packet size of a broadcast message	512 bytes
Maximum waiting timeout for SFR	2 s.
Maximum waiting timeout for DECA	0.2 s.
Simulation Time	20 s.
Number of simulation runs	10
Vehicles density (veh/km)	2, 6, 10, 20, 30, 40, 60, 80

a. An interval from packet was released until packet is expired

Fig. 4 and Fig. 5 show simulation results in highway and urban scenario respectively. Fig.6 show speed of data dissemination result on both highway scenario and urban scenario. All plots shown in this paper are average values of 10 simulation runs.

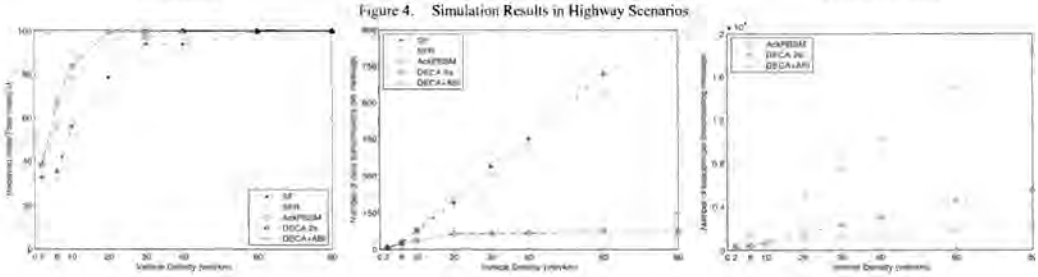
A. *Highway Scenario*

Reliability on low density scenarios in Fig. 4(a) shows different performance between the simple-flooding protocols (SF and SFR) and the store-and-forward protocols (AckPBSM and DECA). DECA can outperform AckPBSM at most 10%.



a) reliability b) data transmission overhead c) beacon overhead

Figure 4. Simulation Results in Highway Scenarios



a) reliability b) data transmission overhead c) beacon overhead

Figure 5. Simulation Results in Urban Scenarios

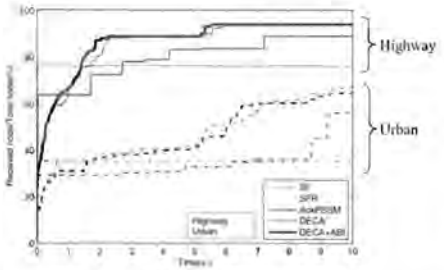


Figure 6. Speed of Data Dissemination at 6 veh/km

On higher density scenarios, SF and SFR still do not reach 100% reliability due to lack of forwarder when intermittent connectivity happens. But DECA and AckPBSM can handle and perform at 100% reliability.

Overhead from data transmission is showed in Fig. 4(b). DECA can reduce a lot of redundant data transmissions due to its forwarder selection algorithm. For each data transmission, the data can reach more number of receivers when comparing to other protocols. AckPBSM generates high number of transmissions closely to SF and SFR. This is because AckPBSM calculates a waiting timeout based on number of neighbors. As a result, nodes running in the same road section mostly have the same number of neighbors. This eventually causes more than one data transmission in a particular area.

See *overhead from beaconing* in Fig.4(c). DECA require only density information and do not need acknowledgement from neighbors. So DECA can work with smaller size of beacon message and operate with longer beacon interval than AckPBSM does. Fig.4(c) also shows that DECA with our proposed adaptive beaconing scheme can save 50% bandwidth comparing to DECA with a fixed beaconing scheme.

Consider *speed of data dissemination* in Fig. 6 (Due to space limitation, only the result at 6 veh/km density is shown). DECA has higher speed of data dissemination than other protocols. This is because of its efficient forwarder selection algorithm. When a node carrying the message meets a group of new neighbors, it will transmit the message to the highest density area and consequently gain lots of received nodes. This causes the graph of DECA in Fig. 6 looks like stair steps.

B. *Urban Scenario*

As shown in Fig. 5(a), Fig. 5(b), Fig. 5(c) and Fig. 6, DECA achieves higher reliability, lower overhead and higher speed of data dissemination than other protocols. The results of urban scenarios show the same trend as happened in highway scenarios. Comparing Fig. 5(a) to Fig. 4(a), it can be seen that all protocols provide lower reliability in urban scenarios. This is because roads are more complicated in urban scenarios and hence some nodes never have chances to meet others.

V. CONCLUSION

Applications and services for vehicular networks need a reliable and efficient broadcast protocol for data dissemination. We propose a new protocol called DECA for this purpose. The protocol does not require GPS but uses store-and-forward technique and employs local density information (number of neighbors) to make decision on forwarding. A source node or a precursor selects a neighbor with the highest density to be the next rebroadcasting node. This neighbor is responsible for rebroadcasting the message immediately without waiting time. By this mechanism, number of nodes that receive the message in one transmission can be maximized. This is because cars on the real traffic always form groups. We also propose an adaptive beaconing scheme, which helps DECA to reduce bandwidth used for beaconing. The simulation results show that DECA can outperform other protocols. Even in the extreme cases such as very low density or very high density scenarios, DECA still operates with the highest reliability, the lowest overhead and the highest speed of data dissemination among all protocols evaluated in this paper.

REFERENCES

[1] M. Conti, and S. Giordano, "Multihop Ad Hoc Networking: The Reality," *IEEE Communications Magazine*, Vol. 45, Issue 4, pp. 88-95, April, 2007.

[2] V. Naumov, R. Baumann, and T. Gross, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," *Proc. ACM the 7th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc'06)*, Florence, Italy, May 22-25, 2006.

[3] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Proc. ACM international conference on Mobile Computing and Networking (MobiCom'99)*, Seattle, USA, August, 1999.

[4] N. Wisitpongphan, F. Bai, P. Mudalige, and O. K. Tonguz, "On the routing problem in disconnected vehicular ad hoc networks," *IEEE the 26th International Conference on Computer Communications (INFOCOM'07)*, Anchorage, Alaska, USA, May 6-12, 2007.

[5] B. Williams, and T. Camp, "Comparison of broadcasting techniques for mobile adhoc networks," *Proc. ACM the 3rd ACM international symposium on Mobile ad hoc networking and computing (MobiHoc'02)*, Lausanne, Switzerland, June, 2002.

[6] M. Nekovee, and B. Bjarni Bogason, "Reliable and efficient information dissemination in intermittently connected vehicular ad hoc networks," *IEEE the 65th Vehicular Technology Conference (VTC'07-Spring)*, Dublin, Ireland, April 22-25, 2007.

[7] A. Afsar Khan, I. Stojmenovic, and N. Zeguia, "Parameterless broadcasting in static to highly mobile wireless ad hoc, sensor and actuator networks," *IEEE the 22nd International Conference on Advanced Information Networking and Applications (AINA'08)*, Okinawa, Japan, March 25-28, 2008.

[8] F. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Reliable and efficient broadcasting in vehicular ad hoc networks," *IEEE the 69th Vehicular Technology Conference (VTC'09-Spring)*, Barcelona, Spain, April 26-29, 2009.

[9] O. K. Tonguz, N. Wisitpongphan, F. Bai, P. Mudalige and V. Sadekar, "Broadcasting in VANET," *Proc. IEEE INFOCOM MOVE Workshop 2007*, Anchorage, USA, May, 2007.

[10] V. Naumov and T. R. Gross, "Connectivity-Aware Routing (CAR) in Vehicular Ad Hoc Networks," *IEEE the 26th International Conference on Computer Communications (INFOCOM'07)*, Anchorage, Alaska, USA, May 6-12, 2007.

[11] F. Farnoud (Hassanzadeh), and S. Valaee, "Reliable broadcast of safety messages in vehicular ad hoc network," *IEEE the 28th International Conference on Computer Communications (INFOCOM'09)*, Rio de Janeiro, Brazil, April 19-25, 2009.

[12] Simulation of Urban MObility (SUMO); http://sumo.sourceforge.net/.

[13] Traffic and Network Simulation Environment (TraNS), http://trans.epfl.ch/.

[14] The Network Simulator (NS-2); http://www.isi.edu/nsnam/ns/.

ชื่องานวิจัย : "A Study of Adaptive Beacon Transmission on Vehicular Ad-Hoc Networks"

คณะผู้วิจัย : ชญานิน ไทยนะ, กุลิสร์ ณ นคร และ กุลธิดา โรจน์วิบูลย์ชัย

การตีพิมพ์ : บันทึกการประชุม "The 13th International Conference on Communication Technology (ICCT 2011)" ซึ่งจัดขึ้น ณ เมืองจี่หนาน (Jinan) ประเทศจีน ระหว่างวันที่ 25-28 กันยายน 2554

A Study of Adaptive Beacon Transmission on Vehicular Ad-Hoc Networks

Chayanin Thaina
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
chayanin.th@student.chula.ac.th

Kulit Na Nakorn
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
kulit.n@student.chula.ac.th

Kultida Rojviboonchai [1]
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
kultida.r@chula.ac.th

***Abstract*—Vehicular Ad-Hoc Networks (VANETs) are special forms of Mobile Ad-Hoc Networks (MANETs). Due to vehicles moving with high speed, network topology frequently changes. Communicating and exchanging information between vehicles could be done by beaconing. Nevertheless, most of previously-proposed protocols for VANETs are designed to use constant beaconing rate. This results in an increase of beacon overhead and a degradation of protocols' performances in dense area. Therefore, in this paper, we study parameters that affect adaptive beaconing, and then propose two methods to apply to adaptive beaconing. One is based on a statistical technique and the other is based on the machine learning technique. Simulation results indicate that our proposed methods can reduce beacon overhead while maintaining the reliability and speed of data dissemination of the protocol.**

I. Introduction

Today, Mobile Ad-Hoc Networks (MANETs) receive considerable attention as a hot research topic. Due to the network's infrastructure-less characteristic, mobile nodes can communicate to each other directly, easily to be set up in lower costs than the network with fixed infrastructure. In addition, MANETs have been applied to several types of network, such as Wireless Sensor Networks (WSNs) and Vehicular Ad-Hoc Networks (VANETs) [1].

VANETs are networks with basic mechanisms as follows: a vehicle has to discover other vehicles nearby and exchange information to each other by sending beacon messages. However, most of the protocols in VANETs have been designed to use constant beaconing rate such as the routing protocol VADD proposed in [2] or the reliable broadcast protocols AckPBSM and DV-CAST proposed in [3] and [4], respectively.

Basically, sending constant beaconing rate can increase beacon overhead and affect the protocols' performance. On one hand, sending high beacon rate in dense area will cause collision and reduce the reliability of the protocols. On the other hand, sending low beacon rate in sparse area will delay vehicles in discovering their neighbors and also reduce the reliability of the protocols [5].

Adaptive beaconing is necessary for VANETs because it can reduce beacon overhead but still maintain the protocols' efficiency. Accordingly, several researches have been conducted for developing adaptive beaconing schemes [6] [7] [8] [9].

Connectivity-Aware Routing (CAR) [6] adjusts beaconing rate according to the number of neighbor nodes. The beaconing rate is less frequent, when the number of neighbors increases.

Boukerche et al. [7] introduce a technique that predicts the current position of a node by using its last received beacon message. If the difference between the predicted position and the actual position is greater than a threshold value, the next beacon will be sent out.

Nakorn et al. [8] suggest adjusting beacon rate according to two methods. The first one is Linear Adaptive Algorithm (LIA) with limited minimum and maximum beacon intervals. The second one is Step Adaptive Algorithm (STA) with divided steps of beacon intervals.

Schmidt et al. [9] propose an approach of adaptive beaconing based on the current traffic situation. Specifically, the movement (e.g. velocity and acceleration) of the vehicle itself and the movement of the surrounding vehicles are considered.

We conclude the drawbacks of the previous works here. Some works have to use so many tests to find the constant value for adjusting beacon interval. It could not get the best performance if the constant value was not the optimal value [6][8]. In some other works, vehicles need GPS data for adjusting beacon interval; that is, it would not work if it could not retrieve any GPS data [7][9].

To the best of our knowledge, none of the previously-proposed adaptive beaconing schemes has applied a statistical technique and the machine learning technique to help solve the problems. If these are applied, the adaptive beaconing scheme will be much more intelligent and optimized. Beacon overhead can be reduced while protocols' performance remain effective. More importantly, because each application has different requirements, the adaptive beaconing scheme based on a statistical technique and the machine learning technique can help find the optimal solution that satisfies the application requirements with the lowest beacon overhead. In this paper, we propose two adaptive beaconing schemes. The first one utilizes a statistical technique. The second one applies the machine learning technique.

In our schemes, we gather offline data as data samples

[1]Corresponding author

978-1-61284-307-0/11/$26.00 ©2011 IEEE

for adjusting beacon interval. Note that these offline samples can be used for future real-time run. The offline samples are general information that a node can know by itself without depending on any GPS data which makes it flexible.

The rest of this paper is organized as follows. Section II describes a general beaconing scheme in VANETs and its problem. In section III, our proposed adaptive beaconing schemes are explained. In section IV, simulation study and performance evaluation are shown. Finally, this paper is concluded in section V.

II. Beaconing in VANET

A. Beaconing

The purpose of beaconing in VANETs is to assist nodes in discovering their neighbors and exchange their information. Most of the protocols in VANETs use constant beaconing rate such as 2 Hz (sending beacon every 0.5 s.). The exchange information in a beacon message includes node's ID, position, velocity, direction, acknowledgment, etc.

B. Problems of using constant beaconing rate

The distribution of vehicles in different areas is not equivalent. Some areas such as highways, the density is low, while some areas such as urban intersections, the density is high. Sending constant beaconing rate, therefore, causes problems as shown in the following simulation results.

Fig.1(a) and Fig.1(b) illustrate the reliability and beacon overhead of the protocol proposed in [5] with different vehicle density and different beaconing rate.

In Fig.1(a), it can be observed that sending low beacon rate (every 5, 7 and 9 s.) in low density area (2 veh/km) delays nodes in discovering their neighbors and reduces reliability. On the other hand, sending high beacon rate (every 0.1, 0.3 and 0.5 s.) in high density area (30-80 veh/km) causes collision and reduces reliability.

In Fig.1(b), it can be seen that sending high beaconing rate (every 0.1, 0.3 and 0.5 s.) leads to increasing beacon overhead in overall.

III. Adaptive beaconing schemes

A. Design of our adaptive beaconing schemes

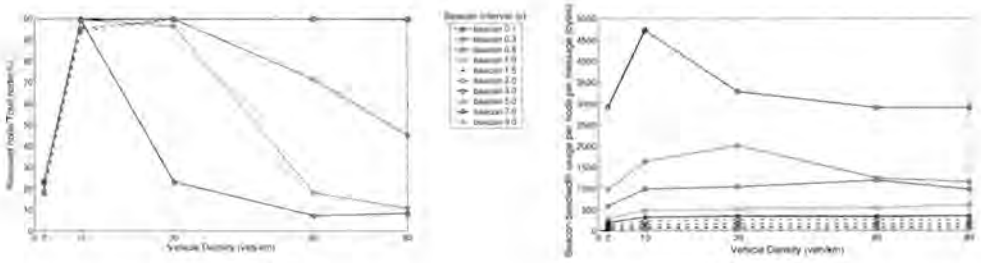
Our study on adaptive beaconing is divided into two parts.

1. Considering the following parameters that affect adaptive beacon interval.
   - 1.1) *Node's environment;* the number of neighbors and the number of buffered messages are considered. These parameters reflect the node density and network traffic. It can be expected that when the number of neighbors and the number of buffered messages are high, beacon should be sent with low frequency. This is to reduce collision in high network traffic situation. Similarly, when the number of neighbors and the number of buffered messages are low, beacon should be sent with high frequency. This is to discover neighbors as soon as possible.
   - 1.2) *Application requirements;* each application requires different speed of data dissemination. Those which do not require very high speed of data dissemination can be adapted to low beaconing rate in order to reduce overhead.
2. Study the following methods that can be applied to adapt beacon rate.
   - 2.1) Method that determines a statistical model for demonstrating the relationship between the node's environment parameters and beacon interval. For our preliminary study in this paper, the linear regression analysis is used.
   - 2.2) Method that can intelligently adapt beacon interval using the machine learning technique. For our preliminary study, the k-nearest neighbor (k-NN) is used.

B. Basic concept of our adaptive beaconing schemes

Our studies have been conducted based on the following principles.

- Test sending beacon with different beacon intervals and different node's environment. The type of scenario that



(a) reliability (b) beacon overhead

Fig. 1: Simulation results in Urban scenarios (using constant beaconing rate)

is suitable for testing is the highway scenario. This is because it has fine distribution that can obviously show the relationship between beacon intervals and node's environment parameters.
- Gather all the results and conclude the appropriate beacon intervals for different levels of node's environment parameters. These results are used as data samples.
- Use the data samples to adapt beacon interval. Primarily, as stated above, two methods which are the linear regression analysis and the k-nearest neighbor are studied.

Linear regression analysis

Linear regression is a scheme that models the relationship between a dependent variable Y and independent variable X_i. The model can be written as the following equation.

$$\hat{Y} = a + bX \tag{1}$$

where a and b are regression coefficients which can be calculated by

$$a = \bar{y} - b\bar{x}, \quad b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

where $\bar{x}$ is the average of all recorded x_i and $\bar{y}$ is the average of all recorded y.

In our method, the node's environment parameter is given as the independent variable while the beacon interval is the dependent variable.

A node can use Equation (1) to calculate the next beacon interval as follows:
- Count the number of neighbor nodes and the number of buffered messages. The summation of the two values is denoted as the node's environment parameter (X).
- Put the X value into Equation (1) in order to obtain the Y value which will be used as the next beacon interval.

k-Nearest Neighbor ($k-NN$)

k-Nearest Neighbor algorithm is an instance-based learning that is used to approximate real-valued or discrete-valued target function.

In the k-nearest neighbor learning, the training examples will be collected in the form of $(x_i, f(x_i))$ by assuming that each pair of training examples correspond to point in the n-dimensional space.

The nearest neighbors of a training example are defined in terms of the standard Euclidean distance as follows:

Let instance x composes of attribute $\langle a_1(x), a_2(x), \cdots, a_n(x)\rangle$, thus $a_r(x)$ is the value of the r th attribute of instance x.

If query instance is x_q, then the distance between x_q and x_i can be denoted by (x_q, x_i) and determined by the following equation.

$$d(x_q, x_i) \equiv \sqrt{\sum_{r=1}^{n}(a_r(x_q) - a_r(x_i))^2} \tag{2}$$

From Equation (2), the k instances from training examples that are nearest to query point (x_q) are calculated.

After that the weight value of each k instance (w_i) is determined according to their distance to the query point (x_q) using the following equation.

$$w_i \equiv \frac{1}{d(x_q, x_i)^2} \tag{3}$$

Finally, the real-valued target function is calculated by

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i} \tag{4}$$

In our study, the k-nearest neighbor algorithm is used to apply in the adaptation of beacon rate as follows:
- Each node will contain a table that collects the training examples and the data in each row is a pair of the node's environment and beacon interval.
- Define k value which denotes the number of the nearest neighbors.

Furthermore, each node will calculate the next beacon interval by the followings.
- Counts the number of neighbor nodes and the number of buffered messages. The summation of the two values is denoted as the node's environment parameter (x_q).
- Determines all of the nearest neighbors and calculate the weight value of each nearest neighbor by using Equations (2) and (3).
- Finally, uses Equation (4) to calculate the next beacon interval.

IV. Performance Evaluation

A. Case study

The protocol used for case study is Density-aware reliable broadcasting protocol (DECA) [5]. DECA is a reliable broadcast protocol, which uses store and forward techniques and performs very well [10]. Beacon is sent in order to discover neighbor nodes and exchange information between nodes using LIA to determine beacon interval.

Overview of DECA's mechanism describes as follows.
- When a sender starts to broadcast a message, it will select a forwarder who has the highest node density. Then, attaches the forwarder's ID to the sending message.
- Receiving node who is the forwarder will immediately rebroadcast the message.
- Receiving node who is not the forwarder will store the message and set waiting timeout.
- In case that the forwarder node does not rebroadcast the message, another node with the shortest waiting timeout will rebroadcast that message instead.

Furthermore, DECA uses Linear Adaptive Algorithm (LIA) to calculate the beacon interval according to Equation (5).

$$f(n) = \min(MinInv + (C * n), MaxInv) \quad (5)$$

where $f(n)$ is beacon interval, $MinInv$ and $MaxInv$ are constant values which are 1.5 and 7, respectively, n is the number of neighbors, and C is a constant value used for adjusting function slope.

B. Simulation Setup

In our experiment, we use Network Simulation (ns-2.34) [11] to implement our methods. The vehicle mobility trace file is generated by a tool called Simulation of Urban Mobility (SUMO) [12]. Then we use Traffic and Network Simulation Environment (TraNS) [13] to convert XML format file from SUMO into TCL file. We use two scenarios (highway and urban) in simulation. In the highway scenario, we generate a straight road with the length of 4 kilometers and 2 lanes per direction. In the urban scenario, we generate a square grid of 3 kilometers with 1 lane per direction. The details of simulation parameters are predicted in Table I.

TABLE I: SIMULATION PARAMETERS

	Highway	Urban
Packet size of a broadcast message	512 bytes	512 bytes
Simulation Time	10 s	50 s
Vehicles density (veh/km)	Low (2,6,10) Medium (20,30,40) High (60,80)	Low (2,10) Medium (30) High (60,80)
Simulation runs	10	10

C. Metrics

- *Reliability* is measured by the percentage of the number of nodes that received the message at the end of simulation to the number of total nodes.
- *Data transmission overhead* is measured by bandwidth that has been used for data transmission, normalized with the number of nodes and the number of messages in each scenario.
- *Beacon overhead* is measured by bandwidth that has been used for every beacon, normalized with the number of nodes and the number of messages in each scenario.
- *Speed of data dissemination* is measured by the percentage of the number of nodes that have received the message at time (t).

D. Simulation results

We use DECA [5] to evaluate three following beaconing schemes:

- LIA : *Linear Adaptive Algorithm*
- Linear regression : *Linear regression analysis*
- k-NN : *k-Nearest Neighbor*

In this simulation, we specify the speed of data dissemination in highway and urban scenarios to be within 10 and 15 seconds, respectively.

Fig. 2 and Fig. 3 show simulation results in highway and urban scenarios, respectively.

Highway Scenario

Reliability shown in Fig. 2(a) is nearly the same for all methods. In low density scenarios, the reliability is low because nodes have little chance to meet other nodes and exchange message to each other. In high density scenarios, the reliability is high because there are enough nodes to broadcast the message.

Data transmission overhead shown in Fig. 2(b) is similar for all methods. The transmission overhead per node in the low density scenarios is higher than that in the high density scenarios. This is because more works will be loaded to each node to broadcast messages.

Beacon overhead is the most significant result which is the main purpose for this research. See Fig. 2(c). It can be seen that both of our adaptive beacon schemes (linear regression analysis and k-nearest neighbor) can save more bandwidth than LIA up to 20% in low density scenarios, 13% in medium density scenarios and 8% in high density scenarios.

Speed of data dissemination is an important result to check whether our schemes can help the protocol to satisfy the application requirements or not. In highway scenario, we specify the speed of data dissemination to be within 10 s. As shown in Fig. 4, all schemes can reach the highest reliability before 10 seconds, meaning that all schemes can satisfy the application requirements.

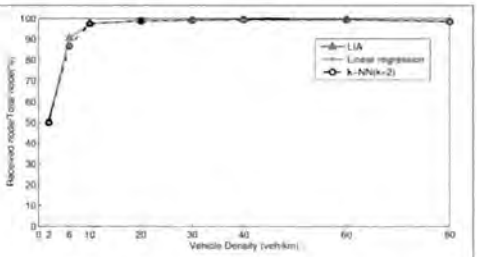
Urban Scenario

As shown in Fig. 3(a), 3(b), 3(c) and Fig. 5, both of our adaptive beacon schemes (linear regression analysis and k-nearest neighbor) can maintain the performance of protocol in terms of reliability and data transmission overhead. Most importantly, from the result, they could reduce beacon overhead than LIA up to 30%. Furthermore, the speed of data dissemination as shown in Fig. 5 is according to the specification (the highest speed of data dissemination within 15 s.). However, in the scenario of 10 (veh/km), all schemes cannot reach the highest reliability before 15 s. We observe that it is because network partition occurred. This will be included in our future work.

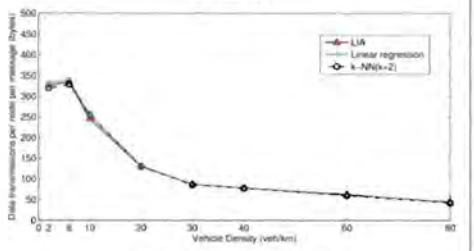
From all results, we can conclude that, in most scenarios, both of our proposed methods (linear regression analysis and k-nearest neighbor) can reduce beacon overhead and maintain other aspects of protocol's performance as the application requires.

V. Conclusion

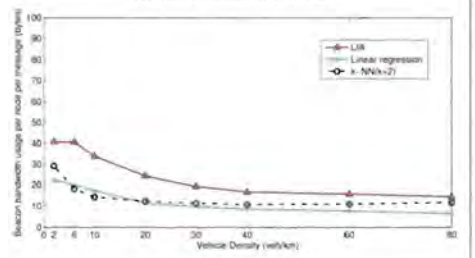
In VANETs, sending beacon is a major mechanism, which will make a node discover its neighbors and exchange information to each other. However, sending constant beaconing rate will increase beacon overhead and affect the protocols' performance. Therefore, adaptive beaconing is necessary. In this paper, we propose two adaptive beaconing methods. First is linear regression analysis, which is the method that can determine the statistical model. Second is k-nearest neighbor, which is the technique of machine learning. Both of the methods can be applied to adjust beacon interval according



(a) reliability

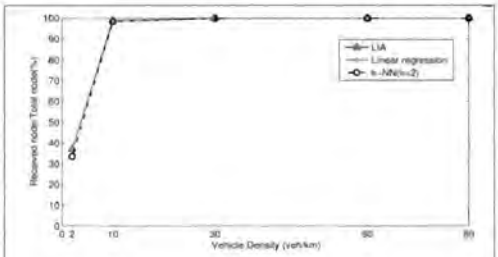


(b) data transmission overhead

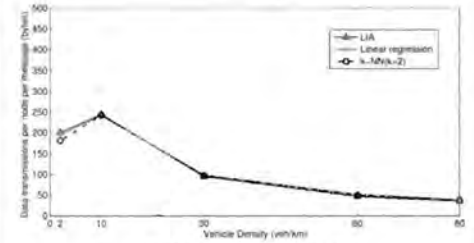


(c) beacon overhead

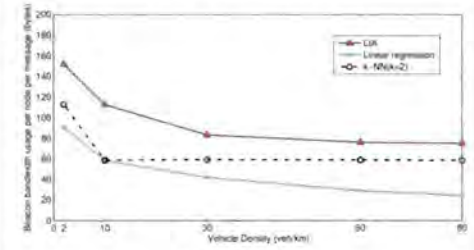
Fig. 2: Simulation results in Highway scenarios



(a) reliability



(b) data transmission overhead



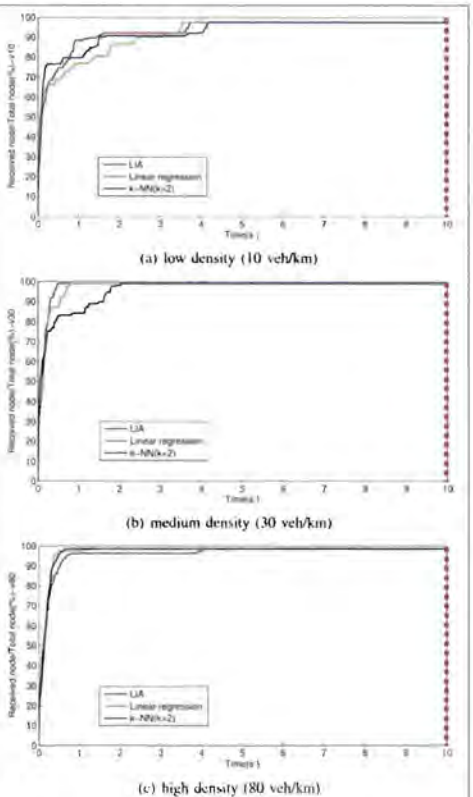
(c) beacon overhead

Fig. 3: Simulation results in Urban scenarios

to node's environment (number of neighbors and number of buffered messages) and application requirements. The simulation results show that our proposed methods can save more bandwidth than LIA up to 20 % in highway scenarios and 30% in urban scenarios while maintaining the reliability and speed of data dissemination of the protocols. Furthermore, these proposed methods can be applied to other protocols as well. Evaluation of the proposed methods in real-map scenarios and study on applying other techniques are included in our future work.

ACKNOWLEDGMENT

This research was supported in part by the TRF (Thailand Research Fund) [MRG5380164], Thailand, and the Ratchadaphiseksomphot Endowment Fund, Chulalongkorn University, Bangkok, Thailand. The authors would like to thank Prof. Dr. Prabhas Chongstitvatana for his useful discussion on this work.

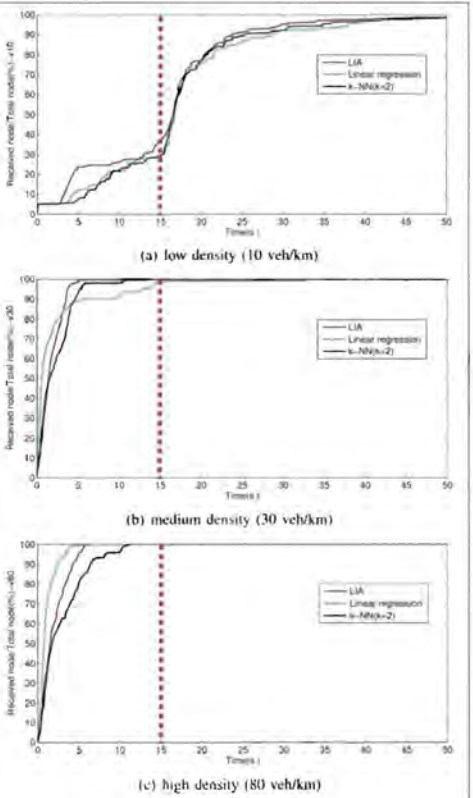


(a) low density (10 veh/km)

(b) medium density (30 veh/km)

(c) high density (80 veh/km)

Fig. 4: Speed of data dissemination in Highway scenarios



(a) low density (10 veh/km)

(b) medium density (30 veh/km)

(c) high density (80 veh/km)

Fig. 5: Speed of data dissemination in Urban scenarios

REFERENCES

[1] H. Hartenstein, and K. P. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Communications Magazine*, Vol. 46, Issue 6, pp. 164-171, June, 2008.

[2] J. Zhao, and G. Cao, "VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks," *IEEE Transactions on Vehicular Technology*, Vol. 57, Issue 3, pp. 1910-1922, May, 2008.

[3] E. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Reliable and efficient broadcasting in vehicular ad hoc networks," *IEEE the 69th Vehicular Technology Conference (VTC'09-Spring)*, Barcelona, Spain, 2009.

[4] O. K. Tonguz, N. Wisitpongphan, and F. Bai, "DV-CAST: A distributed vehicular broadcast protocol for vehicular ad hoc networks," *IEEE Wireless Communications*, Vol. 17, Issue 2, pp. 47-57, April, 2010.

[5] N. N. Nakorn, and K. Rojviboonchai, "DECA: Density-aware reliable broadcasting in vehicular ad hoc networks," *IEEE International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON'10)*, pp. 598-602, May 19-21, 2010.

[6] V. Naumov, and T. R. Gross, "Connectivity-Aware Routing (CAR) in Vehicular Ad-hoc Networks," *IEEE the 26th International Conference on Computer Communications (INFOCOM'07)*, Anchorage, Alaska, USA, 2007.

[7] A. Boukerche, C. Rezende, and R. W. Pazzi, "Improving Neighbor Localization in Vehicular Ad Hoc Networks to Avoid Overhead from Periodic Messages," *IEEE Global Telecommunications Conference (GLOBECOM'09)*, 2009.

[8] N. N. Nakorn, and K. Rojviboonchai, "Efficient Beacon Solution for Wireless Ad-hoc Networks," *The 7th International Joint Conference on Computer Science and Software Engineering (JCSSE'10)*, Bangkok, Thailand, 2009.

[9] R. Schmidt, T. Leinmuller, E. Schoch, F. Kargl, and G. Schafer, "Exploration of adaptive beaconing for efficient intervehicle safety communication," *IEEE Network Magazine*, Vol. 24, Issue 1, pp. 14-19, Jan.-Feb., 2010.

[10] N. N. Nakorn, and K. Rojviboonchai, "Comparison of reliable broadcasting protocols for vehicular ad-hoc networks," *IEEE the 12th International Conference on Communication Technology (ICCT'10)*, pp.1168-1171, Nov 11-14, 2010.

[11] The Network Simulator (NS-2), http://www.isi.edu/nsnam/ns/.

[12] Simulation of Urban MObility (SUMO), http://sumo.sourceforge.net/.

[13] Traffic and Network Simulation Environment (TraNS), http://trans.epfl.ch/.

ชื่องานวิจัย : "A Novel Packet Dropping Policy for Vehicular Ad-Hoc Networks"

คณะผู้วิจัย : วิภาวี วิริยพงษ์สุกิจ, กุลิสร์ ณ นคร และ กุลธิดา โรจน์วิบูลย์ชัย

การตีพิมพ์ : บันทึกการประชุม "The 13th International Conference on Communication Technology (ICCT 2011)" ซึ่งจัดขึ้น ณ เมืองจี่หนาน (Jinan) ประเทศจีน ระหว่างวันที่ 25-28 กันยายน 2554

A Novel Packet Dropping Policy for Vehicular Ad-Hoc Networks

Wipawee Viriyapongsukit
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
wipawee.v@student.chula.ac.th

Kulit Na Nakorn
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
kulit.n@student.chula.ac.th

Kultida Rojviboonchai [1]
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
kultida.r@chula.ac.th

***Abstract*—The characteristics of Vehicular Ad-Hoc Networks (VANETs) are challenging. Because nodes in VANETs have high mobility, intermittent connectivity usually happens. As a result, most of protocols in VANETs use "store and forward" techniques to increase the chances that a message can distribute into the network and finally reach the destination. In this paper, we proposed a novel packet dropping policy that drops the packet with the largest number of message copies when the buffer is full. We take advantages of periodic beacons to gather local information and use the information to calculate the number of copies. To evaluate our dropping policy, we compare with the traditional dropping policies including drop tail, drop front and random drop. Simulation results show that our dropping policy outperforms the traditional dropping policies in terms of reliability.**

I. Introduction

Delay Tolerant Networks (DTNs) [1][2] are topics that have received attention in the past few years. The DTNs architecture is designed to provide communication in intermittently-connected networks such as interplanetary communication, communication in urban and rural areas. Characteristics of DTNs include no end-to-end path between source and destination and long propagation delay between nodes. In DTNs, messages are transmitted by "store and forward" techniques. Each node stores and carries messages, and then forwards the messages when a new communication opportunity arises.

Vehicular Ad-Hoc Networks (VANETs) [3][4] are types of Delay Tolerant Networks (DTNs) [5]. Many nodes in VANETs are moving at high velocity and unpredictability of node positions. Due to high node mobility, the network topology in VANETs tends to change frequently and can cause the network partition.

To address the problem of Intermittent Connectivity, Most of protocols in VANETs such as VADD [6], SADV [7], CAR [8], AckPBSM [9], DECA [10] use "store and forward" techniques that each node stores messages in the buffer until the messages expire or forwards the messages to new nodes encountered, in order to improve the performance of the network.

Another issue that should be considered is buffer management. This is because when a lot of messages are being transferred over the network, network congestion usually happens due to buffer overflow. Hence, how to manage messages in the buffer when the buffer is full becomes important.

However, most of researches on VANETs and DTNs are focused on routing. Thereby, designing a buffer management scheme to improve the performance of the network and compatibility with various protocols in VANETs and DTNs very interesting topics.

A new packet dropping policy for DTNs has been proposed in [11], based on "store and forward" techniques. Specifically, when the buffer is full, the node will drop a packet with a weighting value. The weight of a packet consists of two parts. The first part is the possibility that the packet can be transferred to the destination directly. To obtain the value of this part, the calculation and the definition of inter-contact time between source and destination are needed to be done. The second part is the possibility that the packet can be relayed to a next-hop node and eventually conveyed to the destination. To obtain the value of this part, the information about the number of packet copies in the network is needed. Due to the simulation results, this dropping policy outperforms the traditional policy such as drop tail, drop front and random drop. However, to calculate the weight of a packet, the number of total nodes in the network is required. In fact, it is difficult to know the exact number of total nodes in the network as it is global information.

Algorithms for optimal buffer management policy on DTNs including Global Knowledge Based Drop (GBD) and History Based Drop (HBD) have been proposed in [12]. GBD is fully based on global information. HBD is based on a network history to estimate the number of message copies in the network and the number of nodes that have seen the message. The network history collected from each node is maintained in a form of data structure. Therefore, in a large network, each node must store and maintain more data resulting in high overhead. In calculation, the number of message copies and the number of nodes that have seen the message can be gathered by exchanging local information, whereas the number of total nodes in the network must be relied on a global knowledge. Again, the exact value of the total nodes is difficult to know in practice. The simulation results show the performance comparison with the traditional policy such as drop tail, drop front, drop oldest and drop youngest. As can

[1] Corresponding author

978-1-61284-307-0/11/$26.00 ©2011 IEEE

be predicted. GBD provides the best performance and HBD outperforms the traditional policy.

According to our survey mentioned above, our conclusion is divided into threefold. Firstly, a research on packet dropping policy for DTNs is necessary as the traditional ones are shown to be inefficient. Secondly, there is only a few research on that has been conducted so far. Thirdly, all of the previously-proposed policies rely on global knowledge that is difficult to use in the real world.

In this paper, therefore, we propose a novel packet dropping policy for DTNs and VANETs. When the buffer is full, the node drops the packet that has the largest number of copies first. Our policy relies only on local information that can be gathered easily in practice.

The rest of this paper is organized as follows. In Section II, design concept and details of our novel packet dropping policy are described. In Section III, simulation and performance evaluation are shown. Finally, we conclude this paper in Section IV.

II. Packet Dropping Policy

A. Design Concept

Various protocols in VANETs and DTNs communicate based on "store and forward" techniques. Basically, each node stores and carries the messages and forwards them when new communication opportunity arises. The protocols usually use periodic beacons to discover neighbors and exchange information to 1-hop neighbors.

Knowing the number of packet copies that exist over the network can help decide which packet should be dropped. The packet that has more number of copies over the network has higher possibility to reach the destination, comparing to the packet that has less number of copies. Therefore, when the buffer is full, the node should drop the packet that has the largest number of copies over the network.

To calculate the number of copies of a particular packet, we use the concept of overlapping circle area.

In Fig. 1, illustrates the overlapping circle area, node *A* broadcasts the message to all its neighbors within the transmission range. When node *C* receives the message from node *A*, node *C* then broadcasts the message to all its neighbors within the transmission range. Some nodes (node *B* and node *D*) which are located within the overlapping area receive the same message from node *A* and node *C*.

We can estimate the number of nodes within the overlapping area from the ratio of the area of the overlapping area to the circle area. In this scenario, we assume that the transmission range is set to 250 meters and the node located in the edge of transmission range. This scenario is considered as the worst case that the number of packet copies is the maximum value.

The overlapping area and the ratio to calculate the number of nodes within the overlapping area can be calculated using Equations (1) and (2), respectively.

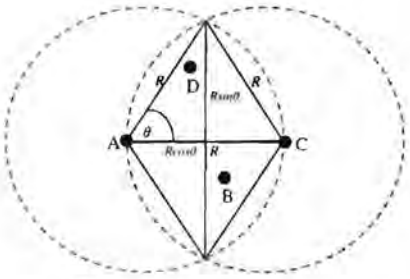


Fig. 1: Overlapping Circle Area

$$Overlapping\ area = 2(\theta - \cos\theta \sin\theta)R^2 \quad (1)$$

$$\alpha = \frac{Overlapping\ area}{Circle\ area} = \frac{2(\theta - \cos\theta \sin\theta)R^2}{\pi R^2} \quad (2)$$

where α is the ratio of the overlapping area to the circle area, R is the transmission range, and θ is an angle of equilateral triangle (60 degrees).

B. Dropping policy details

Most of protocols in VANETs and DTNs consist of the following basic mechanism. The packet is broadcasted from the source node. Then, when neighbors within transmission range receive the packet, the preferred node (the next rebroadcast node) from each protocol algorithm will broadcast the packet. Then, when a node found that its neighbors miss some packets. The node has a responsibility to broadcast missing packets to its neighbors. According to the mechanism mentioned above, we can estimate the number of copies more preciously as follows.

- When a source node broadcasts a packet

See Fig. 2, when a source node would like to broadcast a packet, all nodes in transmission range of the source node will receive the packet. The number of the packet copies would be close to the number of neighbors of the source node. So the source node (node *S*) will piggyback the number of neighbor nodes as a copy value with the packet and broadcast the packet to all its neighbors.

Fig. 2: A source node broadcasts a packet.

- When the preferred node broadcasts a packet

See Fig. 3, when a source node (node *S*) broadcasts a packet to all its neighbors, all neighbor nodes within transmission range will receive the packet. The preferred node from each protocol algorithm (assume that it is node *P*) will broadcast this packet again. This causes the overlapping area between the source node and the preferred node. To calculate the copy value, the preferred node will remove number of copies in the overlapping area to reduce redundant copy value as in the design concept. Then the preferred node will piggyback the copy value with the packet and broadcast the packet to all its neighbors. The copy value is calculated according to Equation (3).

$$New\ Copy = (Copy + NB) - (NB\_Precursor \times \alpha) \quad (3)$$

where $New\ Copy$ represents the copy value that will be piggyback to the packet, $Copy$ represents the latest copy value that node has collected, NB is the number of neighbors of the preferred node, $NB\_Precursor$ is the number of neighbors of the precursor node, and α is the ratio that can be obtained using Equation (2).

Fig. 3: The preferred node broadcasts a receiving packet.

The calculation in Equation (3) removes the number of nodes in the overlapping area. As a result, it will make the copy value more accurately. For more clarification, Equation (4) shows an example of the calculation according to the situation shown in Fig. 3.

$$New\ Copy = (6 + 5) - (6 \times 0.39) = 9 \quad (4)$$

- When a neighbor node misses a packet

We have found that many protocols in VANETs [9][10] try to deliver the missed packet by using waiting timeout mechanism to avoid redundant transmission. A node has to wait for an amount of time before it can rebroadcast the missed packet. So this node can listen to its other neighbors' beacon and count for number of the neighbors that miss the packet. To calculate the new copy value in this case more accurately, this number is added to the copy value and piggybacked with the packet.

See Fig. 4, node *M* detects a neighbor node (node *A*) that misses a packet, then node *M* sets waiting timeout and counts the number of neighbors which miss the same packet. When node *M* is waiting for timeout, it receives beacons from node *B* and node *C*. Node *M* will know that node *B* and node *C* also miss the same packet as node *A*. Therefore, node *M* will calculate the new copy value, piggyback the new copy value with the packet and broadcast the packet to all its neighbors. The copy value is calculated according to Equation (5).

$$New\ Copy = (Copy + NBM) \quad (5)$$

where $New\ Copy$ represents the copy value that will be piggy backed to the packet, $Copy$ represents the latest copy value that node has collected, and NBM is the number of neighbor nodes that misses a packet.

Fig. 4: A neighbor node misses a packet.

Equation (6) shows an example of the calculation according to the situation shown in Fig. 4.

$$New\ Copy = 5 + 3 = 8 \quad (6)$$

To update a copy value for all scenarios, each node compares its latest copy value of a packet with the new copy value which nodes have heard from other nodes' retransmission. If the new copy value is greater than their latest copy value, nodes will replace it with the new copy value.

III. Performance Evaluation

A. Simulation Setup

To evaluate our policy, we have implemented our novel packet dropping policy into Network Simulator (NS-2.34) [14] and used traffic simulator called Simulation of Urban Mobility (SUMO) [15] to generate the movements of vehicles. Then we use Traffic and Network simulation Environment (TraNS) [16] to convert the vehicle behaviors in the format of XML to the format of TCL. The output of TraNS is used in NS-2.34. We generate traffic in highway scenario with the road length of 4 kilometers. As a preliminary study, DECA [10] is the protocol that we use throughout the performance evaluation, because it's perform very well [13].

Density-aware reliable Broadcasting protocol (DECA) [10] is a store and forward technique with periodic beacons to discover neighbors and exchange information in 1-hop neighbors. When a source node would like to broadcast a packet, it will select the preferred node with the highest number of neighbors and broadcast the packet. All nodes within transmission range have received the packet, store the packet into the buffer and set a waiting timeout. The preferred node selects the next rebroadcast node and rebroadcasts the message. If the preferred node does not rebroadcast the message, other nodes with the shortest waiting timeout will do it instead. In case of nodes detect some neighbor nodes that miss the packet, they

will set waiting timeout and a node with a shortest waiting time will rebroadcast the packet.

We compare our novel packet dropping policy to the following traditional policy.

- *Drop Tail*: drop incoming packet in the buffer.
- *Drop Front*: drop the first packet in the buffer.
- *Random Drop*: drop a packet in the buffer randomly.

The detail settings of simulation parameters are shown in Table I.

TABLE I: PARAMETER SETTING

Number of vehicles	100,160 vehicles
TTL of packet	200 s.
Packet size of a broadcast message	512 bytes.
Number of packets	40 packets.
Buffer size	5, 10, 15, 20 packets.
Simulation Time	900 s.
Number of simulation runs	10

B. Evaluation Metrics

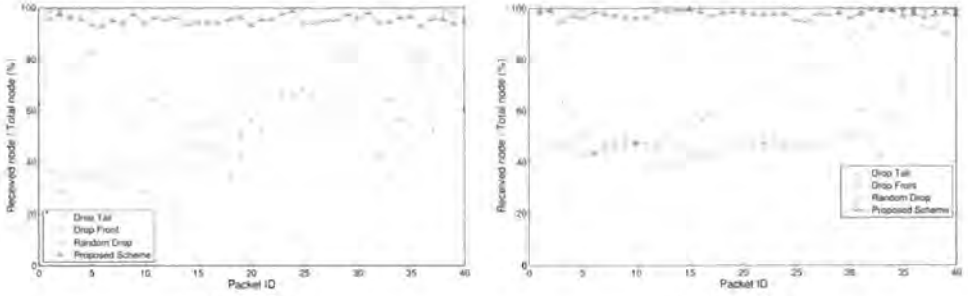
We evaluate the performance in term of reliability that is measured as a percentage of the number of nodes that received message to the total node in simulation.

C. Simulation Result

In Fig. 5, 6, 7 and 8, show simulation results when the buffer size is set to 5, 10, 15 and 20 packets, respectively. In simulation, multiple sources broadcast multiple packets. Packet ID is put into each packet in increasing order. Forty packets are broadcasted so the packet ID is run from 1 to 40. All simulation results shown in this paper are average values of ten simulation runs.

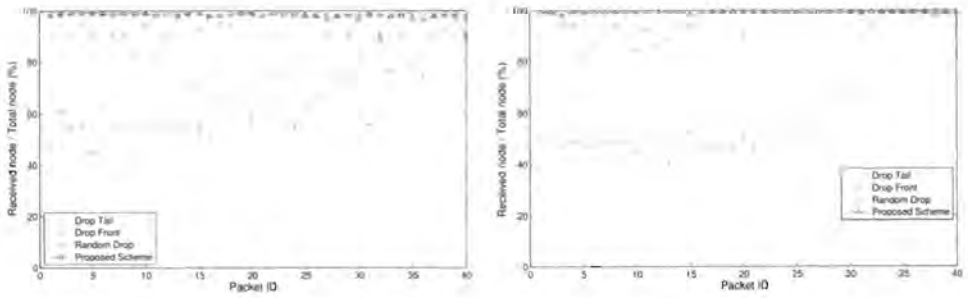
It can be seen that our packet dropping policy outperforms others in terms of reliability of overall packets. This is because our policy drops the packet with the largest number of copies. Such a packet is likely to be replicated and stored in more number of nodes in the network. Dropping such a packet does not affect to reliability of the packet as there are still a lot of other nodes storing the packet copy. *Random Drop* performs the best among the traditional dropping policies. *Drop Front* will drop the first packet in the buffer. That's why reliability of the packets with smaller packet ID is lower than the packets with larger packet ID.

The policy that performs worst is *Drop Tail*. It is because



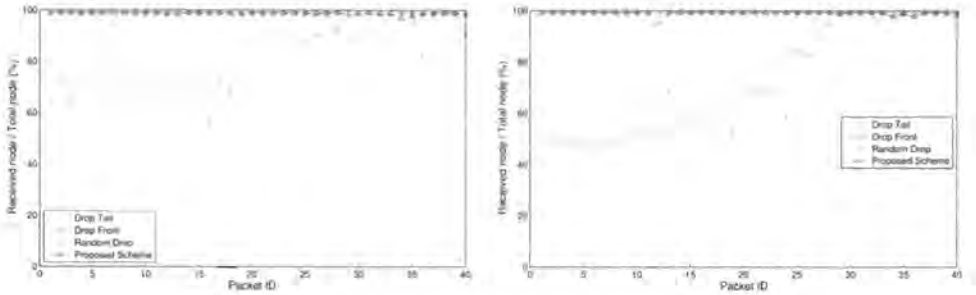
(a) 100 vehicles (b) 160 vehicles

Fig. 5: Reliability with buffer size 5 packets



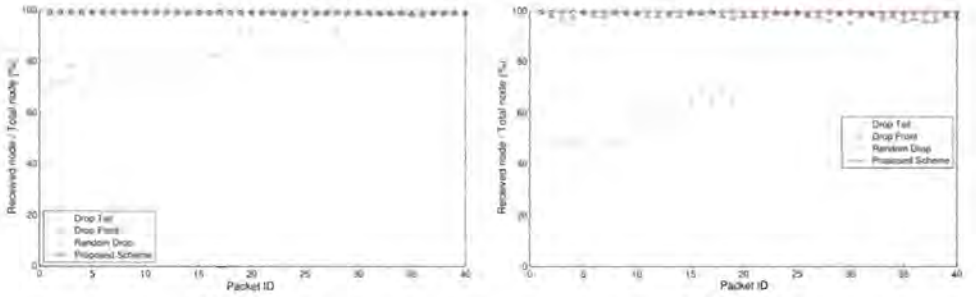
(a) 100 vehicles (b) 160 vehicles

Fig. 6: Reliability with buffer size 10 packets



(a) 100 vehicles (b) 160 vehicles

Fig. 7: Reliability with buffer size 15 packets



(a) 100 vehicles (b) 160 vehicles

Fig. 8: Reliability with buffer size 20 packets

Drop Tail will drop the incoming packet in the buffer. This decreases the chance of the incoming packet to be replicated and stored at other nodes. The simulation results with different buffer sizes show the same trend. It can be also seen that as the buffer size increases, the reliability of all policies increases. This is because the large buffer size provides more space for packet copies in the network.

IV. Conclusion

Vehicular Ad-Hoc Networks (VANETs) are kinds of Delay Tolerant Networks (DTNs). Due to high node mobility, the network topology in VANETs and DTNs tends to change frequently and can cause the intermittent connectivity (networks partition). To address this problem, most of the protocols in VANETs and DTNs use "store and forward" techniques to improve the performance of the network. More importantly, due to the fact that nodes in the network have finite buffer, the buffer management should be taken into consideration. We propose a novel packet dropping policy which drops a packet that has the largest number of copies, when the node's buffer is full. The simulation results show that our policy outperforms the tradition dropping policy and improves the performance of the network in terms of reliability. Although our proposed policy is evaluated in VANETs, it is general and can be applied to DTNs as well. In the future, we would like to determine a solution that can count the number of copies more accurately and consider other information to improve efficiency of dropping policy.

Acknowledgment

This research was supported in part by the TRF (Thailand Research Fund) [MRG5380164], Thailand, and the Ratchadaphiseksomphot Endowment Fund, Chulalongkorn University, Bangkok, Thailand. The authors would like to thank Prof. Dr. Prabhas Chongstitvatana for his useful discussion on this work.

References

[1] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-Tolerant Networking: An Approach to Interplanetary Internet," *IEEE Communications Magazine*, Vol. 41, No. 6, pp.128-136, 2003.

[2] K. Fall, "A delay-tolerant network architecture for challenged internets," *Proc. ACM 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM'03)*, New York, USA, pp.27-34, 2003.

[3] M. Conti, and S. Giordano, "Multihop Ad Hoc Networking: The Reality," *IEEE Communications Magazine*, Vol. 45, Issue 4, pp. 88-95, April, 2007.
[4] A. Dahiya, and Dr.R. K. Chauhan, "A Comparative study of MANET and VANET Environment," *Journal of computing*, Vol. 2, Issue 7, July, 2010.
[5] S. Olariu, and M. C. Weigle, "Vehicular Networks: From Theory to Practice," Chapter 10, 2009.
[6] J. Zhao, and G. Cao, "VADD: Vehicle-assisted data delivery in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1910-1922, May, 2008.
[7] Y. Ding, C. Wang, and L. Xiao, "A static-node assisted adaptive routing protocol in vehicular networks," *Proc ACM the 4th ACM international workshop on Vehicular ad hoc networks (VANET'07)*, New York, USA, 2007.
[8] V. Naumov, and T. R. Gross, "Connectivity-aware routing (CAR) in vehicular ad hoc networks," *IEEE the 26th International Conference on Computer Communications (INFOCOM'07)*, Anchorage, AK, May, 2007.
[9] F. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Reliable and efficient broadcasting in vehicular ad hoc networks," *IEEE the 69th Vehicular Technology Conference (VTC'09-Spring)*, Barcelona, Spain, April 26-29, 2009.
[10] N. N. Nakorn, and K. Rojviboonchai, "DECA: Density-Aware Reliable Broadcasting in Vehicular Ad-Hoc Networks," *IEEE the 7th International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON'10)*, pp.598-602, Chiang Mai, Thailand, May 19-21, 2010.
[11] M. Ke, Y. Nenghai, and L. Bin, "A new packet dropping policy in delay tolerant network," *IEEE the 12th International Conference on Communication Technology (ICCT'10)*, pp.378-381, Nanjing, China, November 11-14, 2010.
[12] A. Krifa, C. Barakat, and T. Spyropoulos, "Optimal Buffer Management Policies for Delay Tolerant Networks," *IEEE the 5th Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'08)*, San Francisco, June 16-20, 2008.
[13] N. N. Nakorn, and K. Rojviboonchai, "Comparison of Reliable Broadcasting Protocols for Vehicular Ad-Hoc Networks," *IEEE the 12th International Conference on Communication Technology (ICCT'10)*, Nanjing, China, November 11-14, 2010.
[14] The Network Simulator (NS-2); http://www.isi.edu/nsnam/ns/.
[15] Simulation of Urban Mobility (SUMO); http://sumo.sourceforge.net/.
[16] Traffic and Network Simulation Environment (TraNS); http://trans.epfl.ch/.

ชื่องานวิจัย : "DECA on android : reliable broadcasting in ad-hoc network on android platform"

คณะผู้วิจัย : ณฐกร นพคุณวิจัย, สุขุมาลย์ อาชาสันติสุข, กุลิสร์ ณ นคร และ กุลธิดา โรจน์วิบูลย์ชัย

การตีพิมพ์ : บันทึกการประชุม "ICT International Senior Project Conference 2012" ซึ่งจัดขึ้น ณ จังหวัดนครปฐม ประเทศไทย ระหว่างวันที่ 20 เมษายน 2555

The 2012 First ICT International Senior Project Conference (ISPC2012)
and IEEE Thailand Senior Project Contest

DECA on Android: Reliable Broadcasting in Ad Hoc Network on Android Platform

Nathakorn Nophakunvijai Sukhumarn Archasantisuk Kulit Na Nakorn Kultida Rojviboonchai*
Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
{nathakorn.n, sukhumarn.a, kulit.n}@student.chula.ac.th kultida.r@chula.ac.th

***Abstract*—In this paper, we have implemented DECA on Android platform. To the best of our knowledge, this is the first work that implements Ad Hoc reliable broadcasting protocol on Android devices. In this protocol, before broadcasting each message, a node selects a neighbor with the highest number of friends to be a next rebroadcasting node. The selected node rebroadcasts the message immediately while others store the message for possible rebroadcasting. Several problems about characteristics of Android operating system and time synchronization have been addressed. Testing has also been conducted to ensure correctness and to evaluate reliability.**

Keywords-wireless Ad Hoc; broadcasting protocol; Android

I. INTRODUCTION

Nowadays, mobile devices such as smartphones or tablets have an important role in daily activities of people. Using these devices with internet connection, people can access information they want anytime and anywhere. However, there are limitations of this system since it has to rely on pre-installed infrastructure; so a new concept of using Ad Hoc network is developed.

Ad Hoc network is a temporary network created using only the nodes without relying on any infrastructure. Seeing its benefits, we have an idea to increase usability of smartphones or tablets by developing Ad Hoc network on these devices. Ad Hoc network can form a small social network. These small social networks are useful in many ways. For instance, they can be used in academies or schools to share knowledge and in exhibitions or trade fairs to publicize information. Also, to communicate in Ad Hoc network, users do not need to register with the service providers and pay the service cost. Moreover, Ad Hoc network is more flexible than Infrastructure system because it can be set up easily anywhere.

To the best of our knowledge, although there are some implementations of Ad Hoc network on Android platform such as AODV routing protocol [1], there is no Ad Hoc reliable broadcasting protocol implementation on Android platform. Ad Hoc reliable broadcasting protocol can be used in many applications such as publicizing application or advertisement application. Moreover, in the emergency case, it is very useful for rescue team members to communicate with each other in the place where there is no infrastructure. Some routing protocols can also use broadcasting protocol for establishing or maintaining routes. The easiest way for broadcasting is Simple Flooding. In Simple Flooding, each node will rebroadcast a message immediately after receiving that message. However, there are many disadvantages to Simple Flooding: it is not reliable and it generates many redundant messages. Because mobile devices have resources limitation such as energy or network bandwidth, it is essential to use protocol that can distribute data reliably and rapidly while generating redundant messages as less as possible.

In this paper, we have implemented DECA (Density-Aware Reliable Broadcasting in Vehicular Ad-Hoc Networks) [2] on Android platform [3]. DECA has been developed and tested in our laboratory on Network Simulator 2 [4]. A testing result on Network Simulator 2 shows that DECA is highly reliable, uses fewer resources, and disseminates data rapidly. Moreover, we have also developed an Android publicizing application using DECA as an underlying protocol; the application has been designed for the use in an exhibition or event. By using this application, participants will be able to receive distributed information conveniently and immediately without any charge.

The rest of this paper is organized as follows. In Section II, relating works are briefly described. In Section III, our implementation details are explained along with limitations and solutions to emerging problems. In Section IV, we describe how we test our work. Section V concludes this paper. Finally, we propose the future works that we consider useful in Section VI.

II. RELATING WORKS

A. *DECA(Density-Aware Reliable Broadcasting Protocol on Vehicle Ad-Hoc Networks) [2]*

This protocol uses 3 major concepts to reliably broadcast messages.

1. Store-and-Forward: This protocol uses Store-and-Forward to mitigate intermittent connection problem that occurs frequently. A node will store received messages until those messages expire. When it finds a neighbor that has missed a message, it will rebroadcast that message to the neighbor.

*Corresponding author

Figure 1. Flowchart showing how DECA works

2. Preferred Node Selection Algorithm: A source node selects its neighbor node with the highest number of neighbors as its preferred node. Then the source node attaches its preferred node's ID to a broadcast message. After receiving the message, the preferred node will be responsible for rebroadcasting that message. This algorithm can maximize the number of nodes that can receive a broadcast message per one transmission.

3. Waiting Timeout Calculation: All nodes that are not selected to be a preferred node set their Waiting Timeout randomly. In the case that the preferred node does not rebroadcast a message, another neighbor will select its preferred node and rebroadcast the message instead when its Waiting Timeout is up.

B. *Ad-hoc network on Android [1]*

In this work, Ad Hoc network has been created and Ad hoc On-demand Distance Vector (AODV) routing protocol has been implemented on Android devices. This protocol has been developed as a library to provide many functions for an application such as starting/joining Ad Hoc network, terminating/leaving Ad Hoc network, sending data to specific destination, broadcasting data to all neighbors, and notifying an application when receiving a new message.

The functional test of this work has been conducted using laptop computers running Windows XP and/or Ubuntu. The testing scenarios have been divided into 2 following scenarios.

1. Two directly connected computers.

2. Three computers, when there is an intermediate computer between two computers.

After setting scenario, they have tested all possible actions of this protocol and compared tested results with expected results.

Moreover, a text messenger application has also been developed as a proof of concept. The work has been tested with rooted Android 2.2 Platform on Google Nexus One and rooted Android 2.1 HTC Hero.

III. IMPLEMENTATION

A. *Creating Ad Hoc Network on Android Devices*

Because Android devices' hardware is various, each device requires a unique method to set up a wireless Ad Hoc network. In this work, we have chosen Google Nexus One as our testing device because it is an Android device for developers [5]. To set up a wireless Ad Hoc network on Google Nexus One, we have to install a customized ROM; in our case, we use CyanogenMod 7 ROM [6].

Another limitation we encounter is about configuring network information. Because there is no server in wireless Ad Hoc network, Dynamic Host Configuration Protocol (DHCP) cannot be used. Hence, each device's IP Address has to be set manually.

We use Linux command line as shown in Fig. 5 to set up a wireless Ad Hoc network.

B. *Implementing DECA*

We use Java programming language to implement the protocol. Because the lowest network protocols that Java provides are UDP and TCP [7], we decide to use a UDP packet as an underlying packet instead of an IP packet used in the original work. Hence, our DECA implementation works in the Application Layer while the original DECA works in the Network Layer.

As shown in Fig. 2, our DECA design comprises five packages: table, timer, packet, util, and deca_agent. The table package provides storage for DECA. The timer package provides a timer for DECA. The packet package defines two network packet types used in DECA (beacon type and data type). The util package provides methods for doing trivial calculations. The deca_agent package combines all other packages' functions to provide DECA service.

C. *Developing a Plubicizing Application*

As a proof of concept, we have developed a text-based publicizing application as shown in Fig. 3 using our DECA implementation as an underlying protocol. A user can type a

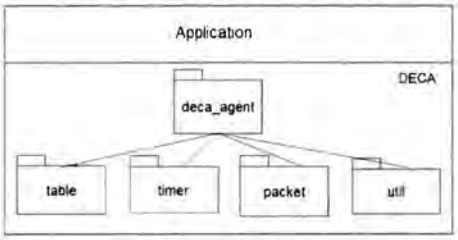


Figure 2. Our DECA design

Figure 3. Publicizing application

message and press 'SEND' button to broadcast the message to all devices in the same Ad Hoc network. When a device receives a message, the protocol will notify the application to show the message to the user.

D. *Implementation Problems*

After DECA has been implemented and tested, we encounter three problems as follows.

Firstly, when an Android device enters sleep mode, some nonessential radio signals will have been shut down including wireless signal. Therefore, devices in sleep mode cannot receive any wireless UDP packet from neighbors. This problem needs to be solved because reliability is a major characteristic of DECA. We do that by setting a Wi-Fi sleep policy to 'Never' as shown in Fig. 4.

Secondly, Google Nexus One's wireless driver default setting will filter out a broadcasted UDP packet when the phone is in sleep mode [8]. This problem also affects the protocol's reliability. We solve it by adding an option 'dhd_pkt_filter_enable=0' when using 'insmod' Linux command line as shown in Fig. 5.

Lastly, unlike a simulation, there is no global clock. In order to create reliable protocol, nodes have to store received messages for a while, so they can send stored messages to other nodes that have missed those messages. When a message expires, it will be deleted from every node's storage. Because each node uses its local clock, the same message in different nodes may be deleted at different time. This can cause a

Figure 4. Setting Wi-Fi sleep policy

Figure 5. Setting up wireless Ad Hoc network on Android

problem. For example, if message M in node X is deleted before message M in node Y is deleted, node Y will think that node X has missed message M and rebroadcast message M. To mitigate this problem, when a message in a node expires, we set that message's status to be inactive and extend the time to keep that message in the node's storage. After the second expiration, the message is truly deleted from the node's storage. If a node finds it has a message which other node does not have, it will firstly check whether that message is inactive or not. If the message is inactive, it will not rebroadcast the message. Otherwise, it will rebroadcast the message normally. By doing so, when node Y mistakenly thinks that node X has missed the message, it will not rebroadcast that message because that message in node Y probably is already inactive.

IV. TESTING

In order to ensure that our implementation functions properly, we have conducted a unit test, a functional test and a reliability evaluation. For the unit test, we have used JUnit test suite to test all possible inputs for each method.

After that, we have set up 2 following scenarios.

1. Two Google Nexus One devices both running ROM CyanogenMod7 and one laptop running Ubuntu 10.10.

2. Six devices: one moving Google Nexus One device, one fixed Google Nexus One device, both running ROM CyanogenMod 7, two moving laptops running MacOS 10.7 and Ubuntu 10.10, and two fixed laptops both running Ubuntu 10.10

In the first scenario, we have performed the functional test by creating all possible actions of DECA's behavior. Then, we compare these results with our expected results. The result

TABLE I. PARAMETER SETTING

Parameter	Value
Packet life time	200 s
Beacon interval	4.0-4.5 s
Maximum packet size of a broadcast message	2048 bytes
Maximum waiting timeout for DECA	0.5 s
Packet sending interval (only in the 2nd scenario)	20 s

from the functional test can be summarized as follows.

- A preferred node rebroadcasts a message immediately after receiving the message.
- Every node sends beacons periodically, so it has information about its neighbors to select a next rebroadcasting node.
- Other node which is not selected to be preferred node rebroadcasts a message instead when a preferred node has not rebroadcasted the message.
- A node rebroadcasts a message when it detects a neighbor that has missed the message.
- Information about messages and neighbors has a limited time in the storage and it is deleted properly when it expires.

The second scenario has been conducted to test reliability of this protocol on real devices in dynamic topology. Fig. 6 shows our walking path. One Google Nexus One device and two laptops have been kept static in the same position in the path. One Google Nexus One device and one laptop have been moving along the path in the counterclockwise direction and one laptop has been moving along the path in the clockwise direction. The reliability of each device is evaluated using (1).

$$r_i = [\ (\ \Sigma_{i \neq j}\ m_{i,j}\ ) / (\ m_i \times (n-1)\ )\ ] \times 100 \qquad (1)$$

where r_i is the reliability of device i, $m_{i,j}$ is number of device i's messages that device j has received, m_i is total number of device i's messages, and n is total number of devices in the scenario. Fig. 7 shows the reliability of each device. The average reliability of two Google Nexus One devices is 95% and the average reliability of four laptops is 97.65%. The reliability is not 100% because some devices are far apart from others in some period of time. Slightly lower reliability of Google Nexus One devices compared to that of laptops may be related to their inferior resources. Still, the result shows that DECA can also perform nicely in real devices.

Figure 6 White dashed line shows our walking path in the second scenario

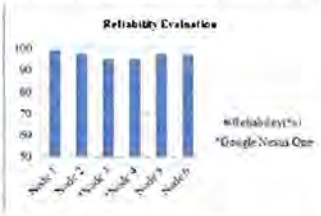


Figure 7 Bar graph for reliability evaluation

V. CONCLUSION

In this paper, we have implemented DECA on Android platform. We have chosen Google Nexus One as a testing device. To create wireless ad hoc on Google Nexus One, we need to install a customized ROM. Implementing and testing the DECA has introduced two important points to consider. Firstly, Java does not let us implement the protocol in the Network Layer, so we have to implement it in the Application Layer instead. Secondly, time difference between devices might cause unwanted rebroadcasting when a message in one node expires while the same message in another node does not. We have mitigated this problem by setting expired message's status as inactive and extending its lifetime in the node; after the second expiration, the message will be truly deleted. After that, we have conducted unit test and functional test to ensure correctness of our implementation. Moreover, we have also evaluated the reliability of this protocol on real devices. The testing results show that our implementation works correctly and this protocol is highly reliable on real devices.

VI. FUTURE WORK

DECA's performance has only been seriously evaluated in Network Simulator 2 [4]. However, when using DECA in real devices, there are some limitations that do not occur in the simulator. Therefore, performance of this protocol in real devices may be different from performance evaluated in the simulator. In the future, we plan to measure performance of this protocol in real devices using 3 metrics: reliability, overhead, and speed of data dissemination and compare the result with the one measured in the simulator.

Another issue we plan to address is the IP Address limitation. An algorithm that can set each device's IP Address automatically while avoiding IP Address conflict will make our work more practical.

Lastly, we also have a plan to implement DECA on other platforms such as iOS to make it more widespread.

REFERENCES

[1] Rabie Khodr Jradi and Lasse Seligmann Reedtz. *Ad-hoc network on Android* [online]. [cited 18 Mar, 2012]. Available from: http://adhoc-on-android.googlecode.com/files/bachelorthesis.pdf

[2] N. Na Nakorn, and K. Rojviboonchai, "DECA: Density-Aware Reliable Broadcasting in Vehicular Ad-Hoc Networks," *IEEE the 7th ECTI-CON2010*, Chiang Mai, Thailand, May 19-21, 2010.

[3] Google Inc. *Android Developers* [online]. [cited 18 Mar, 2012]. Available from: http://developer.android.com/index.html

[4] *ns-2* [online]. [cited 18 Mar, 2012]. Available from: http://nsnam.isi.edu/nsnam/index.php/Main_Page

[5] Tim Bray. *Nexus One Developer Phone* [online]. [cited 18 Mar, 2012]. Available from: http://android-developers.blogspot.com/2010/08/nexus-one-developer-phone.html

[6] *CyanogenMod* [online]. [cited 18 Mar, 2012]. Available from: http://www.cyanogenmod.com/

[7] Harold, E. R. (2005). *Java Network Programming* (3rd ed.) (pp. 5). Cambridge, MA: O'Reilly.

[8] Jordi Cucurull and Paul Gardner-Stephen. *Reception of UDP packets in sleep mode* [online]. [cited 18 Mar, 2012]. Available from: https://groups.google.com/forum/#!msg/android-platform/OpbSdp9FTmA/Y0zK8AIbAYIJ