บทที่ 3

การออกแบบโพรโทคอลสำหรับการแพร่ที่มีความเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ

การแพร่ที่มีความเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะ (Density-Aware Reliable Broadcasting Protocol on Vehicular Ad-Hoc Networks : DECA) ถูกออกแบบโดยคำนึงถึงปัจจัยสำคัญ 3 ประการ ดังนี้ 1) ความเชื่อถือได้ (Reliability) ซึ่งเป็นจุดประสงค์หลักในการทำงานของโพรโทคอล 2) ค่าใช้จ่าย (Overhead) ที่เกิดขึ้นจากการแพร่ข้อมูล และการแพร่จาก beacon message ซึ่งมีผลต่อระบบสื่อสารไร้สายแบบแอดฮอกที่มีทรัพยากรอย่างจำกัด 3) ความเร็วในการแพร่ข้อมูล (Speed of Data Dissemination) ซึ่งยิ่งมีความเร็วสูงยิ่งทำให้ข้อมูลนั้นมีค่ามากขึ้น และส่งผลให้บริการในระดับผู้ใช้มีความแม่นยำมากขึ้น

3.1 แนวคิดในการออกแบบ

การออกแบบสำหรับโพรโทคอลการแพร่ที่มีความเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะมีแนวคิดในการออกแบบดังนี้

1) รถยนต์มักจะจับตัวกันเป็นกลุ่มบนถนน ดังนั้นการเลือกโหนดที่มีเพื่อนบ้านรอบข้างสูงสุดในการแพร่ข้อมูล ย่อมจะทำให้ข้อมูลนั้นครอบคลุมจำนวนโหนดได้มากกว่าโหนดอื่นๆ และหลีกเลี่ยงค่าใช้จ่ายที่จะเกิดขึ้นจากการแพร่ซ้ำในบริเวณเดิมเพื่อครอบคลุมจำนวนโหนดที่เท่ากัน

2) ความเร็วในการแพร่ข้อมูลขึ้นอยู่กับเวลารอ (Waiting Timeout) ดังนั้นการหลีกเลี่ยงการใช้งานเวลารอให้น้อยที่สุดจะสามารถลดความล่าช้าที่จะเกิดขึ้นในการแพร่แต่ละครั้งได้ ส่งผลให้การทำงานของโพรโทคอลทำงานได้เร็วขึ้น

3) โพรโทคอลสามารถทำงานได้อย่างยืดหยุ่นโดยใช้เพียงข้อมูลที่มีในการทำงานเท่านั้น ดังนั้นโพรโทคอลจะสามารถทำงานได้ดีแม้จะไม่มีข้อมูลทางด้านตำแหน่งและทิศทางหรือข้อมูลจากจีพีเอส

3.2 หลักการทำงานของโพรโทคอล

หลักการทำงานที่สำคัญของโพรโทคอลการแพร่ที่มีความเชื่อถือได้แบบรู้ข้อมูลความหนาแน่นสำหรับเครือข่ายไร้สายแบบแอดฮอกบนยานพาหนะประกอบด้วยส่วนต่างๆ ดังต่อไปนี้

1) Store-and-Forward : การทำงานแบบ Store-and-Forward เพื่อรองรับการทำงานในสภาพที่มีการเชื่อมต่อเป็นช่วง ๆที่เกิดขึ้นได้บ่อย โดยโหนดที่ได้รับข้อความจะเก็บข้อความนั้นไว้ในหน่วยความจำจนกว่าข้อความนั้นจะหมดอายุ โหนดที่เก็บข้อความสามารถส่งต่อข้อความนั้นให้แต่โหนดเพื่อนบ้านที่ยังไม่ได้รับข้อความนั้นได้

2) Beaconing with Adaptive Intervals : การใช้ Beacon Message ในการค้นพบเพื่อนบ้าน แลกเปลี่ยนข้อมูล และใช้ในการตรวจสอบหาข้อความที่ยังไม่ได้รับ โดยการแลกเปลี่ยน Beacon Message จะเกิดขึ้นภายในระยะสัญญาณการสื่อสารของโหนด (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 และเลือกโหนดส่งต่อใหม่จากรายชื่อเพื่อนบ้านของโหนดนั้น ๆเอง

3.2.1 การเก็บข้อมูลของโพรโทคอล

รถยนต์แต่ละคัน หรือโหนดจะเก็บข้อมูลสำคัญ 3 ชุด คือ ข้อมูลของเพื่อนบ้าน คิวของข้อความที่จะถูกส่งต่อ และข้อความที่ยังไม่หมดอายุ

1) ข้อมูลของเพื่อนบ้านจะประกอบ ด้วยหมายเลขประจำตัวของโหนด ความหนาแน่นของโหนด เพื่อใช้ในการเลือกโหนดที่จะส่งต่อข้อความ

2) คิวของข้อความที่จะถูกส่งต่อ เป็นรายละเอียดของข้อความพร้อมทั้งเวลาที่จะส่งข้อความออกไป คิวนี้ใช้เพื่อรอเวลาที่ข้อความนั้นจะถูกส่งต่อ แต่ในกรณีที่โหนดได้ยินโหนดเพื่อนบ้านส่งข้อความนั้นก่อน ข้อความก็จะถูกลบออกจากคิว เพื่อลดจำนวนการส่งของข้อความเดิมในบริเวณเดียวกัน ซึ่งคิวของข้อความที่จะถูกส่งต่อไม่ใช่หน่วยความจำที่ใช้ในการเก็บข้อมูล ซึ่งจะเก็บข้อความนั้นไว้จนกว่าข้อความนั้นหมดอายุ

3) ข้อความที่ยังไม่หมดอายุ จะถูกเก็บไว้ตามหลักการทำงานแบบ Store-and-Forward เพื่อใช้ในการส่งให้กับโหนดเพื่อนบ้านที่ยังไม่รับข้อมูลกรณีที่มีการเชื่อมต่อเป็นช่วง ๆ ไม่ต่อเนื่อง ข้อความจะถูกเก็บจนกว่าข้อความนั้นจะหมดอายุ

3.2.2 การแลกเปลี่ยนข้อมูลจากโหนดเพื่อนบ้าน (Beaconing)

ข้อมูลของโหนดเพื่อนบ้านจะได้จากการแลกเปลี่ยนข้อมูลผ่าน Beacon Message ซึ่งข้อมูลที่จะมีการส่งผ่านไปพร้อมกับ Beacon Message ประกอบด้วย

- หมายเลขเฉพาะตัวของโหนด

- ความหนาแน่นของพื้นที่ในบริเวณของโหนด โดยในที่นี้จะใช้จำนวนของเพื่อนบ้าน เป็นข้อมูลที่ใช้ในการเลือกโหนดที่จะส่งต่อข้อมูล โหนดที่มีความหนาแน่นสูงสุดจะถูกเลือก

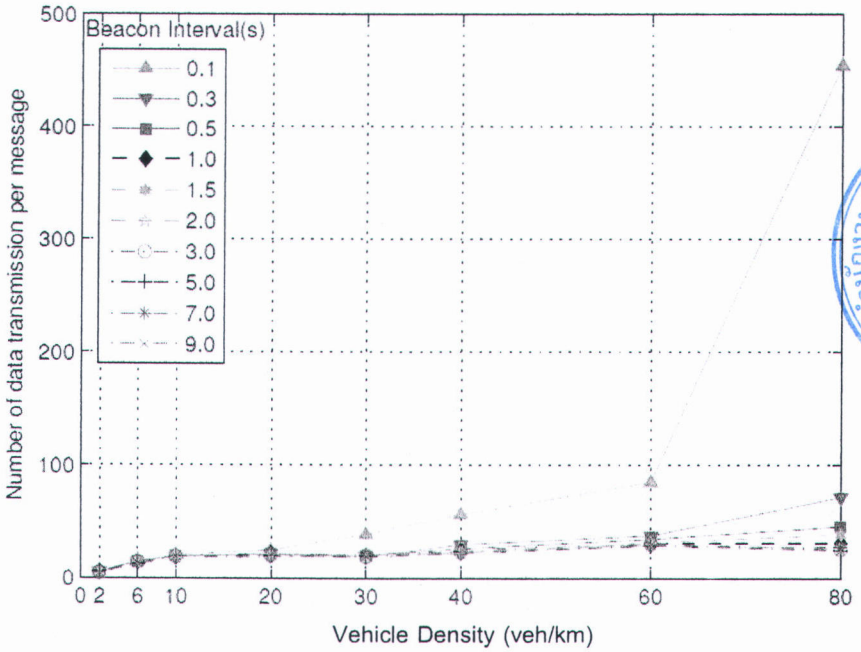
- รายการของข้อความที่ได้รับ ประกอบด้วยโหนดที่แพร่ข้อความ และหมายเลขเฉพาะของข้อความ ใช้เพื่อตรวจสอบว่ามีข้อความที่ยังไม่ได้รับหรือไม่ รายการของความที่ได้รับจะไม่ถูกเก็บลงในข้อมูลเพื่อนบ้าน เมื่อทำการตรวจสอบเสร็จจะถูกลบออก

การแลกเปลี่ยนข้อมูลเพื่อนบ้านผ่านการทำ Beacon เป็นการเพิ่มค่าใช้จ่าย (Overhead) ให้กับการทำงานของโพรโทคอล ในปกติการส่ง Beacon Message จะเป็นเวลาคงที่ เช่น ทุกหนึ่งวินาทีหรือ 1 Hz. ซึ่งหากพิจารณาถึงความสำคัญในการทำงานของโพรโทคอลแล้ว การแลกเปลี่ยน Beacon Message เกิดขึ้นเพื่อค้นหาโหนดเพื่อนบ้านในบริเวณการสื่อสาร ในกรณีที่มีความหนาแน่นของโหนดน้อย การทำ Beacon จำเป็นจะต้องมีความถี่สูงเพื่อให้สามารถพบเพื่อนบ้านได้เร็วที่สุด และในการกรณีที่มีความหนาแน่นสูงการทำ Beacon ที่ความถี่สูงจะก่อให้เกิดปัญหาการชน และส่งผลต่อประสิทธิภาพของโพรโทคอลได้ ดังเช่นในกราฟรูปที่ 3.1 ซึ่งแสดงว่าการทำ Beacon ที่ความถี่สูง แม้ว่าจะทำให้โพรโทคอลมีข้อมูลที่ทันสมัยตลอดเวลา แต่ส่งผลเสียต่อการใช้งานทรัพยากรที่มีอยู่อย่างจำกัด ปัญหาการชนการของข้อมูล ทำให้มีจำนวนการส่งต่อข้อความสูงขึ้นเมื่อเทียบกับความถี่ที่ต่ำกว่า

ดังนั้นการเลือกใช้ช่วงเวลาที่เหมาะสมในการทำ Beacon สำหรับความหนาแน่นของเครือข่ายในแต่ละพื้นที่สามารถลดค่าใช้จ่ายที่เกิดขึ้นระหว่างการทำงานของโพรโทคอลได้ โดยที่ความน่าเชื่อถือจากการแพร่ของโพรโทคอลยังมีค่าเท่าเดิม สามารถแบ่งความหนาแน่นของเครือข่ายที่เกิดได้เป็น 2 ประเภท คือ

1) ความหนาแน่นของโหนด หากมีจำนวนโหนดหนาแน่น ในบริเวณนั้นจะมีจำนวน Beacon Message จำนวนมาก ซึ่งเพิ่มโอกาสที่จะเกิดปัญหาการชนกัน

2) จำนวนของข้อความที่มีการแพร่ ในบริเวณที่มีการแพร่ข้อความสูง ย่อมทำให้จำนวนการใช้ทรัพยากรในพื้นที่มีสูง ซึ่งเพิ่มโอกาสที่จะเกิดปัญหาการชนเช่นเดียวกัน



รูปที่ 3.1 กราฟแสดงจำนวนการส่งข้อความที่ช่วงเวลาการทำ Beacon ต่างๆกัน

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

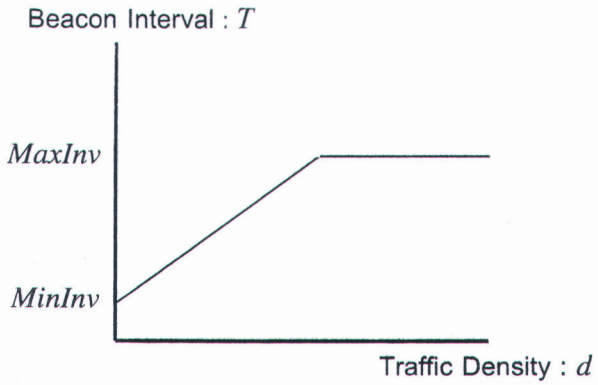
$$d = (w_1 \times n) + (w_2 \times m) \qquad (1)$$

ช่วงเวลาสำหรับการส่ง Beacon Message ที่มีการเปลี่ยนแปลงตามความหนาแน่นของเครือข่าย (Adaptive Beacon Interval) สามารถใช้สมการเชิงเส้นอย่างง่ายมาใช้ในการคำนวณค่าที่เหมาะสมโดยใช้ค่าความหนาแน่นของเครือข่ายเป็นตัวกำหนด และมีการกำหนดช่วงเวลาสั้นสุดและยาวสุด เพื่อให้โพรโทคอลยังสามารถทำงานได้อย่างมีประสิทธิภาพ และสมรรถภาพคงเดิม เรียกการคำนวณช่วงเวลาแบบนี้ว่า การคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น (Linear Adaptive Algorithm: LIA)

การคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น (LIA) สามารถอธิบายได้ตามสมการที่ (2) โดย T คือ ช่วงเวลาสำหรับการทำ Beacon ครั้งถัดไป $MinInv$ คือ ช่วงเวลาสั้นสุด c เป็นค่าคงที่ในการเพิ่มช่วงเวลา d ความหนาแน่นของเครือข่าย และ $MaxInv$ คือ ช่วงเวลายาวสุด

สามารถนำมาเขียนกราฟได้ตามรูปที่ 3.2 การคำนวณช่วงเวลาจะทำหลังจากมีการแพร่ Beacon Message ไปแล้ว และจะใช้ช่วงเวลาที่คำนวณได้เป็นเวลาที่จะทำ Beacon ในครั้งถัดไป

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



รูปที่ 3.2 กราฟแสดงการคำนวณช่วงเวลาปรับตัวแบบเชิงเส้น

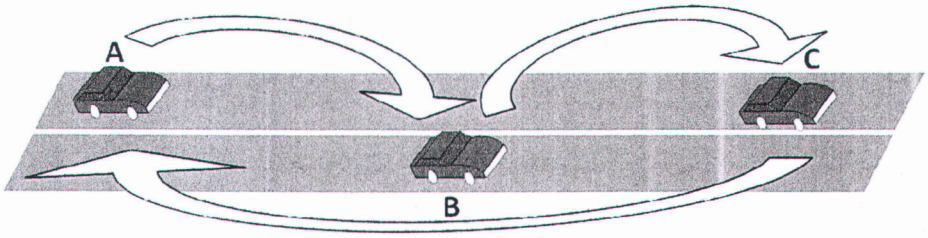
3.2.3 การเลือกโหนดส่งต่อข้อความ (Preferred Node Selection Algorithm)

การทำงานเมื่อเริ่มการแพร่ข้อความ โหนดที่เริ่มต้นการแพร่จะเลือกโหนดที่มีความหนาแน่นสูงที่สุดจากข้อมูลเพื่อนบ้านที่เก็บไว้ จากนั้นจึงแนบหมายเลขของโหนดนั้นพร้อมกับส่งข้อความออกไป โดยโหนดที่ได้รับข้อความนั้นแล้วพบว่าตัวเองเป็นโหนดที่ถูกเลือก ก็จะทำการเลือกโหนดที่ความหนาแน่นสูงสุดจากรายชื่อของตัวเอง แล้วแนบหมายเลขของโหนดนั้นลงไปกับข้อความก่อนส่งข้อความออกไป ซึ่งกระบวนการนี้จะเกิดขึ้นจนกว่ารถยนต์ในบริเวณได้รับข้อความนั้นทั้งหมด หรือข้อความนั้นหมดอายุ

การเลือกโหนดที่จะส่งข้อความต่อนั้น จะเลือกโหนดที่มีความหนาแน่นสูงสุด โดยที่โหนดนั้นจะไม่ใช่โหนดที่ส่งข้อความก่อนหน้า (Precursor Node) และโหนดที่ถูกเลือกจะทำงานก็ต่อเมื่อโหนดนั้นไม่เคยได้รับข้อความมาก่อน เพื่อป้องกันการเลือกโหนดวนซ้ำเดิม ดังเช่นในรูปที่ 3.3 ซึ่งสามารถเกิดขึ้นได้ในกรณีนี้ที่ใช้ข้อมูลความหนาแน่นเท่านั้น เนื่องจากโหนดจะไม่ทราบตำแหน่งของเพื่อนบ้านในการเลือก และไม่เก็บข้อมูลว่าโหนดเพื่อนบ้านมีข้อความใดที่ได้รับแล้ว

รูปที่ 3.3 ปัญหาการเลือกซ้ำ เกิดขึ้นโดย A เลือกโหนด B เป็นโหนดที่ส่งต่อข้อมูล จากนั้น B จึงเลือกโหนด C เป็นโหนดที่มีความหนาแน่นสูงสุดที่ไม่ใช่โหนดที่ส่งก่อนหน้า (Precursor Node) แต่เมื่อ C ทำการเลือกโหนดในข้อมูลเพื่อนบ้าน ซึ่งหากมี A อยู่ A จะเป็นโหนดที่ถูกเลือกโดยที่ A เป็นโหนดที่มีความหนาแน่นสูงสุดและ A ไม่ใช่โหนดที่มีการส่งข้อมูล

ก่อนหน้า ในกรณีนี้การทำงานของโพรโทคอลจะไม่ให้ A ส่งข้อมูลซ้ำ ซึ่งจะเกิดความสูญเปล่าในการส่งข้อความเดิมในบริเวณที่โหนดได้รับข้อความแล้ว แต่จะให้โหนดที่ได้รับข้อความนั้นเป็นครั้งแรกซ่อมแซมการส่งข้อความแทน จะไม่เกิดปัญหานี้หากโพรโทคอลทำงานร่วมกับข้อมูลจีพีเอส



รูปที่ 3.3 ปัญหาการเลือกโหนดส่งต่อแบบวนซ้ำ

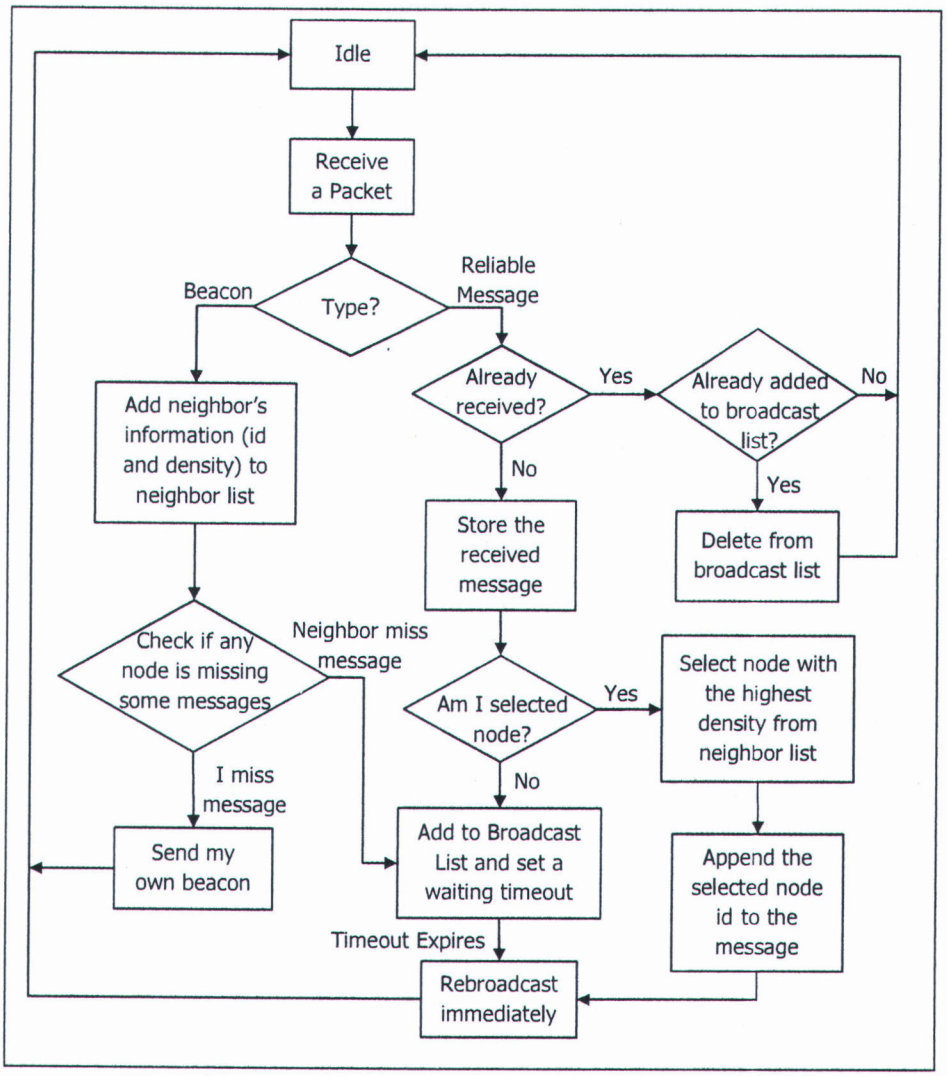
การซ่อมแซมการส่งข้อความเป็นกระบวนการที่เกิดขึ้นเมื่อโหนดที่ถูกเลือกไม่ทำงานตามที่กำหนด เกิดขึ้นได้เมื่อโหนดที่ถูกเลือกได้รับข้อความนั้นแล้วเช่นกรณีข้างต้น หรือเมื่อเกิดการชนทำให้โหนดที่ถูกเลือกไม่ได้รับข้อความนั้น หรือเกิดจากโหนดที่เห็นแก่ตัว ซึ่งได้รับข้อความแล้วไม่ทำการส่งข้อความนั้นต่อ กระบวนการซ่อมแซมนั้นจะเกิดขึ้นทันทีหลังจากที่โหนดที่ไม่ใช่โหนดที่ถูกเลือกได้รับข้อความใหม่ โหนดจะตั้งเวลาเพื่อรอการส่งข้อความนั้นซ้ำอีกครั้ง ซึ่งหากมีการส่งข้อความนั้นซ้ำ ข้อความที่ถูกตั้งเวลาไว้จะถูกลบออกจากคิว แต่ในกรณีที่โหนดนั้นไม่ได้ยินการส่งข้อความซ้ำอีกครั้งจนกระทั่งเวลาที่ตั้งไว้หมด โหนดจะทำการส่งข้อความนั้นซ้ำอีกครั้ง พร้อมทั้งเลือกโหนดที่จะส่งต่อข้อความใหม่จากข้อมูลเพื่อนบ้านที่มีอยู่ ซึ่งเมื่อโหนดหนึ่งได้ทำการซ่อมแซมโดยส่งต่อข้อความนั้นแล้ว โหนดอื่นๆที่อยู่ในบริเวณเดียวกันก็จะทำการลบข้อความนั้นออกจากคิว

เนื่องจากลักษณะเฉพาะของรถยนต์ที่ทำให้เกิดการเชื่อมต่อเป็นช่วงๆได้ ดังนั้นการใช้ Beacon Message จึงสามารถตรวจสอบได้เมื่อโหนดได้รับข้อความไม่ครบ โดยที่โหนดอื่นที่ได้รับ Beacon Message สามารถตรวจสอบได้ว่าข้อความแพร่ใดที่โหนดเพื่อนบ้านยังไม่ได้รับ รวมทั้งข้อความใดที่ตนเองยังไม่ได้รับเช่นกัน

- กรณีที่โหนดเพื่อนบ้านได้รับข้อความไม่ครบ โหนดจะนำข้อความที่เก็บไว้ใส่ในคิวเพื่อรอการส่งต่อ และตั้งเวลาโดยการสุ่ม โหนดที่มีเวลารอน้อยที่สุดเท่านั้นจะทำการส่งข้อความให้โหนดเพื่อนบ้าน โหนดอื่นๆที่ได้ยินจะลบข้อความออกจากคิว ในการส่งข้อความในกรณีนี้โหนดที่ส่งข้อความจะไม่เลือกโหนดที่จะส่งต่อข้อมูล เนื่องจากโหนดเพื่อนบ้านน่าจะมีข้อมูลของโหนดเพื่อนบ้านในบริเวณนั้นที่ดีกว่า

- กรณีที่โหนดพบว่าตนเองมีข้อความที่ขาดไป จะส่ง Beacon Message ทันที เพื่อขอรับข้อความจากโหนดที่มีข้อความที่ตนเองไม่มีอยู่ ก่อนที่โหนดนั้นจะหายไปจากบริเวณนั้น

การทำงานของโพรโทคอลสามารถสรุปได้ตามผังงานรูปที่ 3.4

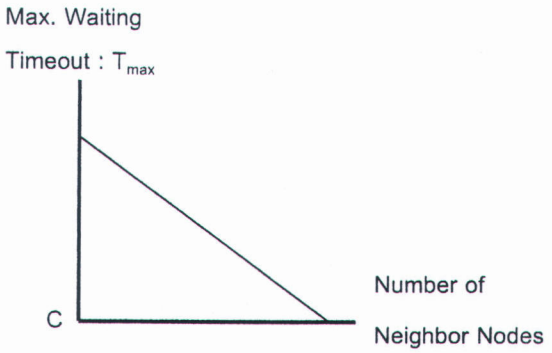


รูปที่ 3.4 ผังงานแสดงการทำงานของโพรโทคอล DECA

3.2.4 การคำนวณเวลารอ (Waiting Timeout Calculation)

เมื่อโหนดได้รับข้อความใหม่ แต่พบว่าหมายเลขโหนดที่ถูกเลือกไม่ตรงกับตัวเอง โหนดนั้นจะตั้งเวลารอเพื่อคอยฟังโหนดที่จะส่งข้อความต่อ แต่ในกรณีที่ไม่มีโหนดใดแพร่ข้อความนั้น ซึ่งอาจจะเกิดได้ในกรณีโหนดที่ถูกเลือกเปลี่ยนตำแหน่ง หรือเกิดความผิดพลาดในส่งข้อมูล หรืออาจจะเป็นเครือข่ายที่มีโหนดเห็นแก่ตัวอยู่ โหนดที่มีเวลารอสั้นสุดจะทำหน้าที่ส่งต่อข้อความนั้นแทน ดังนั้นระยะเวลาที่โหนดจะต้องรอนั้นจึงมีความสำคัญต่อประสิทธิภาพของโพรโทคอล คือ ในกรณีที่โหนดตั้งเวลารอนานเกินไปจะทำให้การแพร่ครั้งถัดไปช้าลงทำให้ความเร็วในการแพร่ลดลง หรือกรณีที่ตั้งเวลารอสั้นเกินไปก็อาจจะทำให้เกิดการแพร่ซ้ำที่บริเวณเดียวกัน ก่อให้เกิดค่าใช้จ่ายที่เพิ่มขึ้นได้

กรณีที่มีการใช้งานเวลารอ แบ่งออกเป็น 2 กรณี คือ เมื่อโหนดที่ถูกเลือกไม่ทำงานดังที่กล่าวข้างต้น และกรณีที่โหนดเพื่อนบ้านมีข้อความที่ยังไม่ได้รับ ซึ่งทั้ง 2 กรณีมีวิธีการคำนวณเวลาเช่นเดียวกัน เนื่องจากโหนดใน DECA ทราบข้อมูลของความหนาแน่น หรือจำนวนโหนดเพื่อนบ้านเท่านั้น ดังนั้นหากโหนดที่มีจำนวนเพื่อนบ้านที่สุดทำการส่งต่อข้อมูล จะทำให้โอกาสในการแพร่ครั้งนั้นครอบคลุมโหนดจำนวนมากขึ้นด้วย ดังนั้นการคำนวณเวลารอสูงสุดจะคำนวณโดยให้โหนดที่มีเพื่อนบ้านจำนวนมากที่เวลารอสั้นที่สุด เพื่อป้องกันการแพร่ที่เวลาเดียวกันจึงใช้การสุ่มค่าในช่วง (C, Tmax] โดยที่ C มีค่าเป็นสองเท่าของค่าเวลาที่ใช้ในการส่ง (Propagation Delay) และ Tmax เป็นเวลารอสูงสุดที่คำนวณได้จากจำนวนโหนดเพื่อนบ้าน สามารถอธิบายการคำนวณได้จากกราฟในรูปที่ 3.5

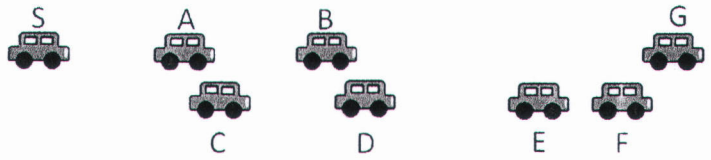


รูปที่ 3.5 กราฟแสดงการคำนวณเวลารอสูงสุด

3.3 ตัวอย่างการทำงานของโพรโทคอล

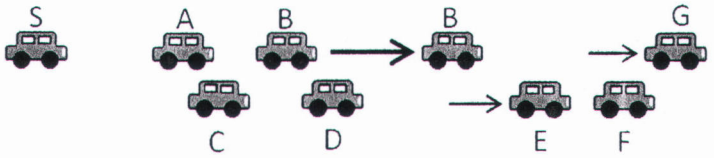
ในรูปที่ 3.6 เป็นการทำงานในลักษณะปกติ เมื่อโหนด S ต้องการเริ่มต้นแพร่ข้อมูลออกไป โหนด S มีโหนด A, B, C และ D เป็นโหนดเพื่อนบ้าน ในกรณีนี้ให้ D เป็นโหนดที่มีความหนาแน่นสูงสุด ดังนั้นในการแพร่ข้อความ S จึงเลือก D แนบไปกับข้อความ หลังจากการแพร่ข้อความของ S โหนด A, B, C และ D ได้รับข้อความพร้อมกัน เมื่อ 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 3.6 ลักษณะของรถในการเชื่อมต่อแบบปกติ

ในรูปที่ 3.7 แสดงการทำงานในกรณีที่มีการเชื่อมต่อเป็นช่วงๆ (Intermittent Connectivity) ในกรณีนี้สมมติให้ E, F และ G อยู่ห่างออกไปจากระยะสัญญาณสื่อสารของ D เมื่อ S เป็นโหนดเริ่มต้นการแพร่ข้อความ และ D ถูกเลือก จากนั้น D จะเลือกโหนดส่งต่อถัดไป ซึ่งอาจจะเป็น A, B และ C ซึ่งในกรณีนี้ A, B และ C จะไม่ส่งข้อความต่อ เนื่องจากโหนดเหล่านี้ได้รับข้อความนั้นแล้ว แต่เมื่อเวลาผ่านไป B แซง C และ D ทำให้พบกับรถยนต์บริเวณด้านหน้าซึ่งคือ E, F และ G หลังจากได้รับ Beacon Message B จะทราบว่าทั้ง 3 โหนดได้รับข้อความไม่ครบถ้วน B จะส่งต่อข้อความแต่ไม่ระบุโหนดส่งต่อ เนื่องจากว่าในพื้นที่ใหม่นั้น E, F และ G น่าจะมีข้อมูลของโหนดเพื่อนบ้านที่ดีกว่า B ดังนั้นหลังจากที่ E, F และ G ได้รับข้อความก็จะตั้งเวลา โหนดที่มีเวลาสั้นสุดก็จะเลือกโหนดที่จะส่งต่อถัดไป แทนที่การเลือกของ B



รูปที่ 3.7 ลักษณะของรถในการเชื่อมต่อเป็นช่วงๆ