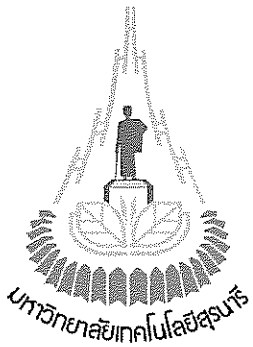


การจำลองกราฟฟิคเพื่อศึกษาการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค

(GUI for studying energy usage in MANETs)

โดย

นายไชยรัตน์ จันทร์บุญเรือง รหัสนักศึกษา B4602248

นายณัฐวุธ คณฑา รหัสนักศึกษา B4602767

รายงานนี้เป็นส่วนหนึ่งของการศึกษาวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม

ประจำภาคการศึกษาที่ 2 ปีการศึกษา 2549

หลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม หลักสูตรปรับปรุง พ.ศ. 2545

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

โครงงาน	การจำลองกราฟฟิคเพื่อศึกษาการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค (GUI for studying energy usage in MANETs)
จัดทำโดย	นายไชยรัตน์ จันทร์บุญเรือง
	นายณัฐวุธ คณฑา
อาจารย์ที่ปรึกษา	อาจารย์ ดร.วิภาวี หัตถกรรม
สาขาวิชา	วิศวกรรมโทรคมนาคม
ภาคการศึกษาที่	2/ 2549

บทคัดย่อ

(Abstract)

การจำลองกราฟฟิคเพื่อศึกษาการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค (GUI for studying energy usage in MANETs) คือโปรแกรมที่ จำลองการเคลื่อนที่ของโครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc networks) ไว้โดยใช้โปรแกรมวิชวลซีพลัสพลัสเวอร์ชั่นหกจุดศูนย์ (visual C++ 6.0) ซึ่งนอกจากจะเป็นการหาระยะทางที่สั้นที่สุดแล้วโปรแกรมยังสามารถใช้พิจารณาระดับพลังงาน ของระบบมาเนท (MANETs)ได้อีกด้วย พร้อมทั้งแสดงค่าระดับพลังงาน หรือ แบตเตอรี่ ปัจจุบัน ออกทางจอแสดงผล สามารถใช้ดูผลที่อาจจะกระทบต่อการเคลื่อนที่ของระบบได้ เพราะในสถานการณ์จริง การทดลองค่อน ข้างจะทำได้ยากผู้จัดทำจึงได้ สร้างแบบจำลองนี้ขึ้นมาเพื่อพิจาณาผลกระทบ ต่าง ๆ ที่มีผลต่อระบบ เช่น เวลาและความสำเร็จสะสม (time and accumuted of success path) จำนวนโหนดและความสำเร็จสะสม (node and accumuted of success path อายุขัยของเครือข่ายและจำนวนโหนด (network life time and node) อายุขัยของเครือข่ายและรัสมีการส่งสัญญาณ (networklife time and radius) เวลาและจำนวนโหนดที่เหลืออยู่ในเครือข่าย (time of alive node) พลังงานเฉลี่ยและเวลา (average residual energy in network and time) รัศมีการส่งสัญญาณและความสำเร็จสะสม (radius and accumuted of success path) จำนวนโหนดและความสำเร็จสะสม (node and accumuted of success path) ความเร็วและความสำเร็จสะสม (velocity and accumuted of success path) อายุขัยของเครือข่ายและจำนวนโหนด (network life time and node)

กิตติกรรมประกาศ

(Acknowledgement)

จากการที่คณะจัดทำรายงานได้รับมอบหมายให้ทำโครงงานเรื่อง การจำลองกราฟฟิคเพื่อศึกษาการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค (GUI for studying energy usage in MANETs) ส่งผลให้คณะจัดทำรายงานได้รับความรู้และประสบการณ์ต่างๆ เกี่ยวกับการเขียนโปรแกรมด้วยโปรแกรมวิชวลซีพลัสพลัสเวอร์ชั่นหกจุดศูนย์ (visual c++ 6.0) เป็นอย่างมาก บัดนี้โครงงานดังกล่าวพร้อมทั้งรายงานได้สำเร็จลงแล้ว ทั้งนี้ด้วยความร่วมมือและสนับสนุนจากบุคคลต่างๆ ดังนี้

1. อ.ดร. วิภาวี หัตถกรรม (อาจารย์ที่ปรึกษาโครงงาน)
2. อ.ดร. ปรเมศวร์ ห่อแก้ว (อาจารย์สาขาวิชาวิศวกรรมคอมพิวเตอร์)
3. นางสาววิภาดา นฤพิพัฒน์ (นักศึกษาปริญญาโท สาขาวิชาวิศวกรรมโทรคมนาคม)

ข้าพเจ้าใคร่ขอขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่มีส่วนร่วมในการให้ข้อมูลและเป็นที่ปรึกษาในการทำรายงานฉบับนี้จนเสร็จสมบูรณ์ ตลอดจนให้การดูแลและให้ความเข้าใจเกี่ยวกับพื้นฐานการใช้งานโปรแกรม ซึ่งข้าพเจ้าขอขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้ด้วย

นายณัฐวุธ คณฑา

นายไชยรัตน์ จันทร์บุญเรือง

สารบัญ

เรื่อง **หน้า**

สารบัญ (ต่อ)

เรื่อง **หน้า**

สารบัญรูป

รายการ **หน้า**

บทที่ 1
บทนำ

1. บทนำ

การติดต่อสื่อสารของระบบคอมพิวเตอร์เริ่มเปลี่ยนจากการใช้ระบบสายมาสู่การใช้สัญญาณวิทยุ (radio frequency: RF) เป็นสื่อกลางในการแลกเปลี่ยนข้อมูล ส่งผลให้เกิดการเปลี่ยนแปลงครั้งใหญ่ในโลกของการติดต่อสื่อสาร เนื่องจากผู้ใช้งานสามารถใช้งานคอมพิวเตอร์ในการเข้าถึงข้อมูลได้อย่างไม่จำกัดเวลาและสถานที่ ระบบเครือข่ายสามารถแบ่งได้หลายประเภท แต่ถ้าพิจารณาตามรูปแบบการเชื่อมต่อกับโครงข่ายสามารถแบ่งได้ 2 ประเภท คือ

แบบที่ 1 อาศัยโครงข่ายพื้นฐานในการแลกเปลี่ยนข้อมูล (infrastructure network) ซึ่งระบบโครงข่ายประกอบไปด้วยอุปกรณ์เครื่องให้บริการ (server) และเครื่องขอใช้บริการ (client) รวมถึงอุปกรณ์การติดต่อสื่อสารอื่น ๆ และต้องใช้อุปกรณ์เครือข่าย เช่น ฮับ (hub), สวิทชิ่ง (switching), แอคเซสพอยซ์ (access point (AP)) หรือเร้าเตอร์ (router) ทำหน้าที่เป็นเครื่องบริการ การแลกเปลี่ยนข้อมูล ถ้าอุปกรณ์เครือข่ายทำงานผิดพลาด จะส่งผลให้การติดต่อสื่อสาร ในระบบเครือข่ายนั้นใช้ไม่ได้เครือข่ายในปัจจุบันส่วนใหญ่ใช้โครงข่ายพื้นฐานนี้

แบบที่ 2 คือ แบบไม่มีโครงข่ายพื้นฐาน (infrastructure-less network) การติดต่อสื่อสารของอุปกรณ์ในระบบนี้ไม่จำเป็นต้องอาศัยอุปกรณ์เครือข่ายช่วนในการแลกเปลี่ยนข้อมูล ซึ่งในโครงข่ายแบบนี้ประกอบไปด้วยอุปกรณ์คอมพิวเตอร์ขนาดพกพาได้ (mobile devices) โดยใช้เทคโนโลยีสายเป็นตัวกลาง (transmission medium) ในการติดต่อสื่อสารและมีทำงานแบบกระจาย (distributed computing) กล่าวคือ ไม่มีจุดศูนย์กลางในการควบคุมการทำงานของระบบ แต่ใช้การทำงานร่วมกันเป็นหลัก ซึ่งเรียกอีกชื่อว่า โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network หรือ MANETs) โดยแบบจำลองโครงข่ายเคลื่อนที่แบบแอดฮอคนี้สามารถใช้เพื่อประกอบการศึกษาหลักการเลือกเส้นทางสื่อสารที่สั้นที่สุดและการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค และสำหรับพฤติกรรมจริงของโครงข่ายเคลื่อนที่แบบแอดฮอคนั้น แต่ละโหนดไม่ได้ทำการเชื่อมต่อกันทุกครั้งที่มีรัศมีถึงกัน แต่ละโหนดไม่ได้เปลี่ยนทิศทางในการเคลื่อนที่พร้อมกันทุกโหนด การสื่อสารระหว่างโหนดโดยผ่านโหนดอื่นซึ่งทำหน้าที่เป็นสถานีทวนสัญญาณ โหนดดังกล่าวนี้ไม่ได้อนุญาตให้โหนดอื่นใดๆ ใช้ในการทวนสัญญาณเสมอไป

2. ความเป็นมาและความสำคัญของปัญหา

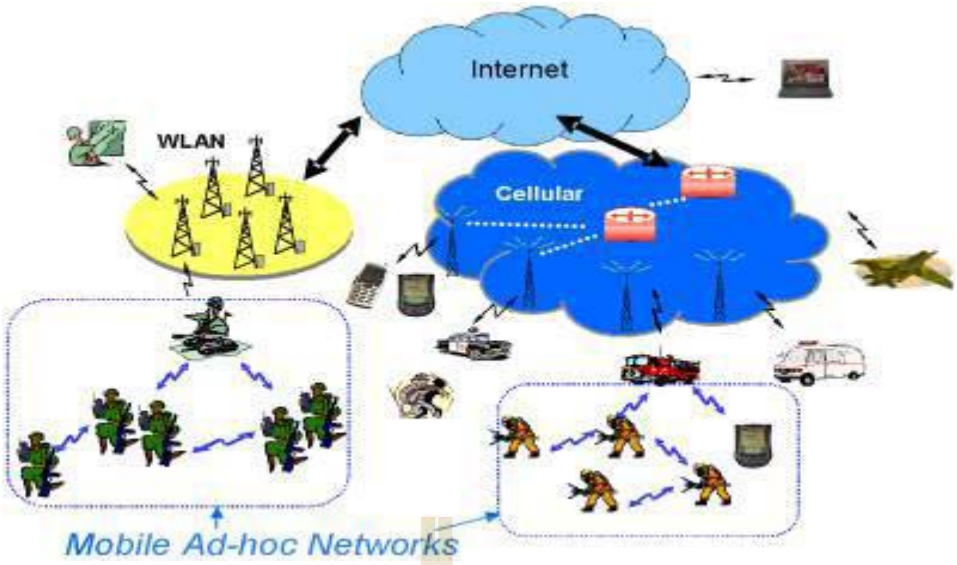
โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network หรือ MANETs) คือการติดต่อสื่อสารที่ไม่มีการอำนวยการจากส่วนกลาง ประกอบไปด้วยโหนดเคลื่อนที่ (mobile node) ซึ่งแต่ละโหนดจะมีทรานซ์สมิตเตอร์ (transmitters) และรีซีฟเวอร์ (receivers) เพื่อติดต่อกันอย่างไร้สาย ด้วยรัศมีการส่งที่จำกัด แต่ละโหนดจึงทำหน้าที่เป็นทั้งเราเตอร์ (router) และโฮสต์ (host) เพื่อส่งต่อแพคเกต (packet) และรันแอพลิเคชั่น (run application) ในเวลาเดียวกัน

เนื่องจากโหนดแต่ละโหนดสามารถเคลื่อนที่ได้อย่างเป็นอิสระ ทำให้รูปร่างของเครือข่าย (network topology) เปลี่ยนแปลงได้ตลอดเวลา จึงจำเป็นมีโปรโตคอล (protocol) เพื่อช่วยในการค้นหาเส้นทางระหว่างโหนดต้นทางและโหนดปลายทางเพื่อส่งข้อมูล การเลือกใช้เส้นทางที่ใช้ระดับพลังงานของแบตเตอรีเพื่อประหยัดพลังงาน มีความจำเป็นและน่าสนใจ เนื่องจากแต่ละโหนดมีระดับพลังงานที่จำกัดและลดลงตามระยะเวลาการใช้งาน การหาเส้นทางที่พิจารณาระดับพลังงานจะทำให้ได้มาซึ่งเส้นทางการส่งข้อมูลที่ดีระหว่างโหนด และสามารถประหยัดแบตเตอรีในระยะยาวอีกด้วย

3. โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) คืออะไร

โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) คือ ระบบที่ประกอบไปด้วยอุปกรณ์พกพาซึ่งสามารถแลกเปลี่ยนข้อมูลโดยใช้เทคโนโลยีไร้สาย และไม่จำเป็นต้องอาศัยสถานีแม่ (access point) ช่วยในการติดต่อสื่อสาร อุปกรณ์พกพาเหล่านี้ใช้พลังงานจากแบตเตอรี่และรูปแบบการเชื่อมต่อของโครงข่าย (network topology) สามารถเปลี่ยนแปลงได้ตลอดเวลา ตัวอย่างการประยุกต์ใช้งานเครือข่าย โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) เช่นในสนามรบ สถานที่เกิดภัยธรรมชาติ ห้องเรียน ห้องประชุม หรือสถานพยาบาล ซึ่งไม่มีโครงข่ายแบบพื้นฐานคงที่อยู่ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) แบ่งตามระยะทางการติดต่อสื่อสารได้ 4 ประเภท คือ

1. ระบบเครือข่ายไร้สายสำหรับเครื่องคอมพิวเตอร์แบบสวมใส่ (body area network: BAN)
2. ระบบเครือข่ายส่วนบุคคล (personal area network: PAN)
3. ระบบเครือข่ายงานเฉพาะที่ (local area network: LAN)
4. ระบบเครือข่ายงานบริเวณกว้าง(wide area network: WAN)



รูปที่ 1 ตัวอย่างโครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network)

ระบบเครือข่ายไร้สายสำหรับแลน (wireless local area network: WLAN) เริ่มมีบทบาทมากขึ้นในปัจจุบันเนื่องจากการเพิ่มจำนวนของเครื่องคอมพิวเตอร์ที่มีขนาดเล็ก เช่น เครื่องพีดีเอ (personal digital assistant (PDA)) เครื่องคอมพิวเตอร์ส่วนตัวแบบพกพา (notebook computer) และคอมพิวเตอร์แบบสวมใส่ (wearable computer) ซึ่งมีจอแสดงผลคล้ายแว่นตา หน่วยประมวลผลกลาง (central processing unit) ติดที่เข็มขัดและแป้นพิมพ์สวมไว้ที่แขน อุปกรณ์เหล่านี้ทำงานภายใต้สภาวะที่มีการเคลื่อนที่ และจำเป็นต้องอาศัยเครือข่ายไร้สายเป็นสื่อกลางในการแลกเปลี่ยนข้อมูล ส่งผลให้ความต้องการการใช้งานเครือข่ายไร้สายสำหรับแลนเพิ่มขึ้นอย่างมาก ระบบเครือข่ายไร้สายสำหรับแลนช่วยให้การทำงานมีความคล่องตัวสูงขึ้น เนื่องจากผู้ใช้งานมีการเคลื่อนย้ายอุปกรณ์พกพาไร้สายไปตามที่ต่าง ๆ เช่น ภายในมหาวิทยาลัย สนามบิน ร้านอาหาร หรือโรงแรม ซึ่งสามารถเชื่อมต่ออินเทอร์เนตได้ตลอดเวลา

ระบบเครือข่ายส่วนบุคคล เป็นเครือข่ายไร้สายที่มีรัศมีครอบคลุมพื้นที่การติดต่อสื่อสารประมาณ 10 เมตร โดยใช้เทคโนโลยีบลูทูท (bluetooth) เครือข่ายส่วนบุคคลเป็นเครือข่ายขนาดเล็กที่สามารถใช้ในการแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์อิเล็กทรอนิกส์ เช่น โทรศัพท์เคลื่อนที่ ชุดหูฟังไร้สาย เครื่องพีดีเอ (PDA) เครื่องคอมพิวเตอร์ส่วนตัวแบบพกพา และสมาร์ทการ์ด

4. หลักการพื้นฐานเกี่ยวกับระบบสื่อสารแบบ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network)

โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) มีลักษณะโดยทั่วไปดังต่อไปนี้

1. โมบิลิตี้ (mobility) โหนดแต่ละโหนดใน โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) นั้นสามารถเคลื่อนที่ได้ ซึ่งทิศทางในการเคลื่อนที่ของแต่ละโหนดนั้นอาจจะเป็นทิศทางที่สามารถคาดการณ์ได้หรืออาจจะเป็นการเคลื่อนที่แบบสุ่ม (random) ความเร็วในการเคลื่อนที่นั้นก็จะมีตั้ง หยุดนิ่งจนไปถึง การเคลื่อนที่อย่างเร็ว และยังสามารถเคลื่อนที่ไปเป็นกลุ่มได้อีกด้วย ซึ่งสามารถเกิดขึ้นได้ใน โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) ทั้งสิ้น

2. แอดฮอคโทโพโลยี (ad hoc topology) ระบบโครงข่ายแบบ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) นั้น โครงสร้างระดับล่าง (infrastructure) ของระบบ จะไม่อยู่กับที่ จึงทำให้โทโพโลยี (topology) ของโครงข่ายนั้นถูกกำหนดโดย ตำแหน่งของโหนดแต่ละโหนด ณ เวลาขณะใดขณะหนึ่ง

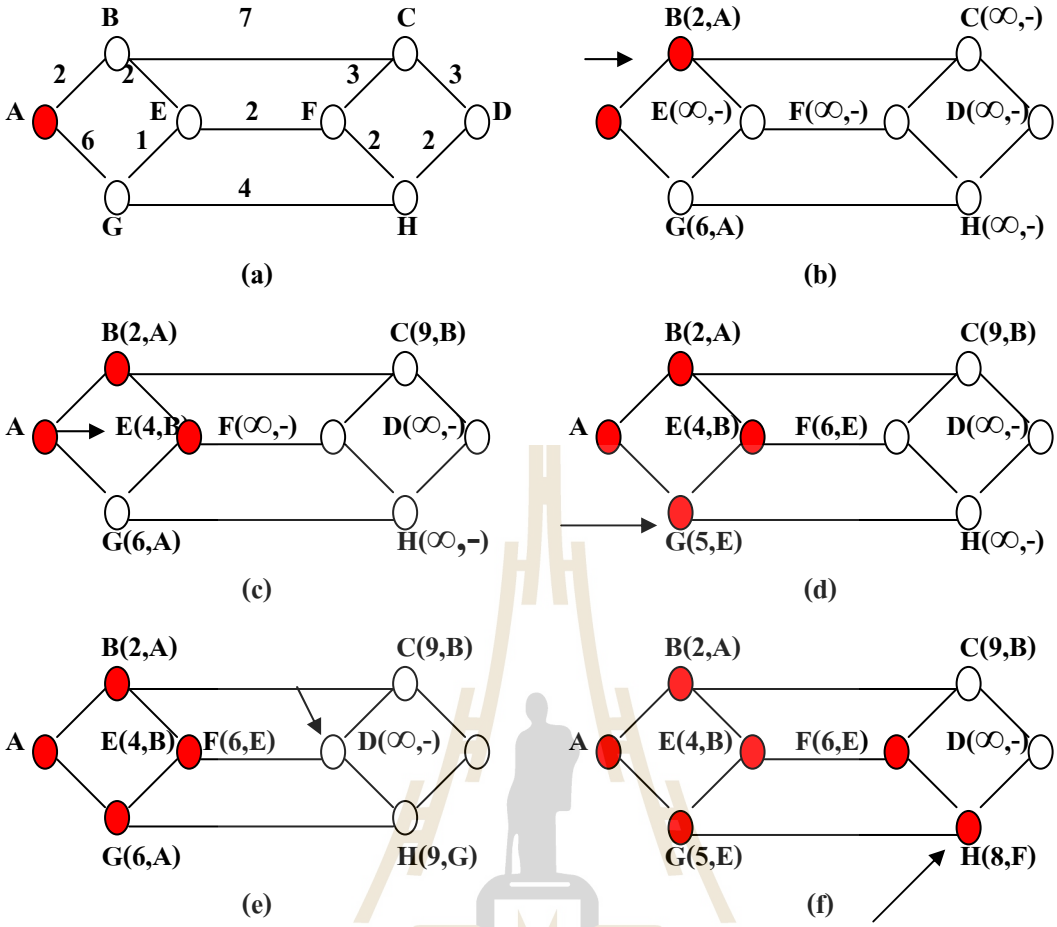
3. ไวเลสเน็ทเวอร์ค (wireless network) การเชื่อมต่อในระบบ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) เป็นแบบ ไวเลส (wireless) ซึ่งอ้างอิงตามมาตรฐาน IEEE 802.11

5. หลักการเลือกทางเดินที่สั้นที่สุด

การเลือกทางเดินที่สั้นที่สุด เป็นวิธีการที่ถูกนำไปใช้มากที่สุดแบบหนึ่ง หลักการทำงานเริ่มต้นด้วยการสร้างรูปกราฟของระบบเครือข่ายย่อย โดยให้แต่ละโหนดในรูปกราฟแทนเราเตอร์แต่ละตัวในเครือข่าย และให้เส้นเชื่อมโหนด (arc) แทนสายสื่อสารที่เชื่อมต่อระหว่างเราเตอร์ การเลือกเส้นทางเดินระหว่างเราเตอร์คู่หนึ่ง ทำได้โดยการค้นหาเส้นทางที่สั้นที่สุดในรูปกราฟ

นิยามของคำว่าเส้นทางที่สั้นที่สุดอาจมีได้หลายความหมาย เช่นการใช้จำนวนครั้งของการรับ-ส่ง ข้อมูล หรือจำนวนเส้นเชื่อมในรูปกราฟเป็นหลักพิจารณา เส้นทางเอบีซีดี (ABCD) และ เอบีอีเอฟ (ABEF) ในรูปจะถือว่ายาวเท่ากันแต่ถ้าใช้ระยะทาง (เช่น เป็นกิโลเมตร) มาพิจารณาแล้วเส้นทางเอบีอีเอฟ (ABEF) ย่อมสั้นกว่า

อีกวิธีหนึ่ง ที่สามารถนำมาใช้แทนการนับจำนวนเส้นเชื่อม หรือการวัดระยะทาง คือการกำหนด ให้ตัวเลขบนเส้นเชื่อมแต่ละเส้นเป็นตัวเลขที่บอกจำนวนแพ็กเก็ตที่รอการจัดส่ง และเวลารอคอยโดยเฉลี่ย ที่คำนวณมาจากการรับส่งแพ็กเก็ตมาตรฐาน ด้วยวิธีนี้เส้นทางที่สั้นที่สุดจึงหมายถึงเส้นทางที่ส่งข้อมูลได้เร็วที่สุด



รูปที่ 2 ขั้นตอน 5 ขั้นตอนแรกที่ใช้ในการคำนวณหาเส้นทาง
ที่สั้นที่สุดจากเราเตอร์เอ (A) ไปยังเราเตอร์บี (B) (ลูกศรชี้ตำแหน่งที่กำลังทำการคำนวณอยู่)

วิธีการที่นำมาใช้กับกรณีทั่วไปนั้นจะกำหนดให้ตัวเลขบนเส้นเชื่อมคือ ผลลัพธ์ที่ได้จากการคำนวณโดยมีระยะทาง ความเร็วในการส่งข้อมูล ปริมาณข้อมูลโดยเฉลี่ย ค่าใช้จ่าย ค่าเฉลี่ยมาตรฐานของจำนวณแพ็กเก็ตที่รอการจัดส่ง ระยะเวลารอคอย และอื่น ๆที่เป็นตัวประกอบ การเปลี่ยนแปลงค่าความสำคัญของตัวประกอบเหล่านี้ ทำให้แต่ละเราเตอร์สามารถใช้อัลกอริทึมเดียวกันในการคำนวณ แต่ให้ความสำคัญตัวประกอบไม่เหมือนกันได้

ไดจิคตรา (Dijikstra (1959)) ได้นำเสนออัลกอริทึมสำหรับการค้นหาเส้นทางที่สั้นที่สุดระหว่าง 2 จุดในรูปกราฟดังในรูปที่ 2 ต้องการเส้นทางสั้นที่สุดจากเอ (A) ไปยังบี (D) เริ่มด้วยการระบายสีแดงทึบที่จุดเอ (A) (เพื่อบอกให้ทราบว่าได้พิจารณาจุดนี้แล้ว) หาระยะทางของแต่ละจุดที่มีเส้นเชื่อมมาที่จุดเอ (A) แล้วใส่ป้ายบอกระยะทางและโหนด กำกับเส้นเชื่อมเหล่านั้น คือใส่ (2, A)ไว้ที่จุดบี (B) และใส่ (6, A) ไว้ที่จุดจี (G) แล้วเลือกเส้นทางที่สั้นที่สุดจากตัวเลขทั้งหมดที่หาได้

ซึ่งก็คือจุดบี (B) ให้ระบายสีแดงทึบที่จุดบี (B) แล้วใช้วิธีการเดิม จะได้ (4, B) ที่จุดอี (E), (9, B) ที่จุดซี (C), และของเดิม (6, A) ที่จุดจี (G) เลือกจุดอี (E) เป็นจุดต่อไปเพราะมีค่าต่ำสุด ระบายสีแดงทึบที่จุดอี (E) แล้วใช้วิธีการเดิม จะได้ (5,E) ที่จุดจี (G) ซึ่งใช้แทน(6,A) เพราะมีค่าต่ำกว่า (6, E) ที่จุดเอฟ (F), และของเดิม (9, B)ที่จุดซี (C) เลือกจุดจี (G) เป็นจุดต่อไป ระบายสีแดงทึบที่จุดจี (G) แล้วใช้วิธีการเดิม จะได้ (9,G) ที่จุดเอช (H),ของเดิม (6, E) ที่จุดเอฟ (F) ,ของเดิม(9, B)ที่จุดซี (C) เลือกจุดเอฟ (F) เป็นจุดต่อไป ระบายสีแดงทึบที่จุดเอฟ (F) แล้วใช้วิธีการเดิม จะได้ (8, F) ที่จุดเอช (H) ซึ่งแทน (9, G) เพราะมีค่าต่ำกว่า, และของ (9,B) ที่จุดซี (C) ,เลือกจุดเอช (H) เป็นจุดต่อไป ให้ระบายสีแดงทึบที่จุดเอช (H) แล้วใช้วิธีการเดิมจะได้ (10,H)ที่จุดดี (D) และของเดิม (9, B)ที่จุดซี (C) แม้ว่าจะได้เส้นทางมาถึงจุดปลายทางแล้วก็ตามแต่ค่า (9, B)ที่จุดซี (C) มีค่าต่ำกว่าจึงต้องคำนวณต่อไปโดยเลือกจุดซี (C) เป็นจุดต่อไป ระบายสีแดงทึบที่จุดซี (C) แล้วใช้วิธีการเดิมจะได้ (12, C) ที่จุดดี (D) แต่ค่าเดิมคือ (10, H) มีค่าต่ำกว่าจึงใช้ค่าเดิมผลลัพธ์ที่ได้คือเส้นทางเอบีอีเอฟเอชดี (ABEFHD) มีความยาว10 หน่วยเป็นเส้นทางที่สั้นที่สุด

บทที่ 2

โปรแกรมแบบจำลองโครงข่ายเคลื่อนที่แบบแอดฮอค

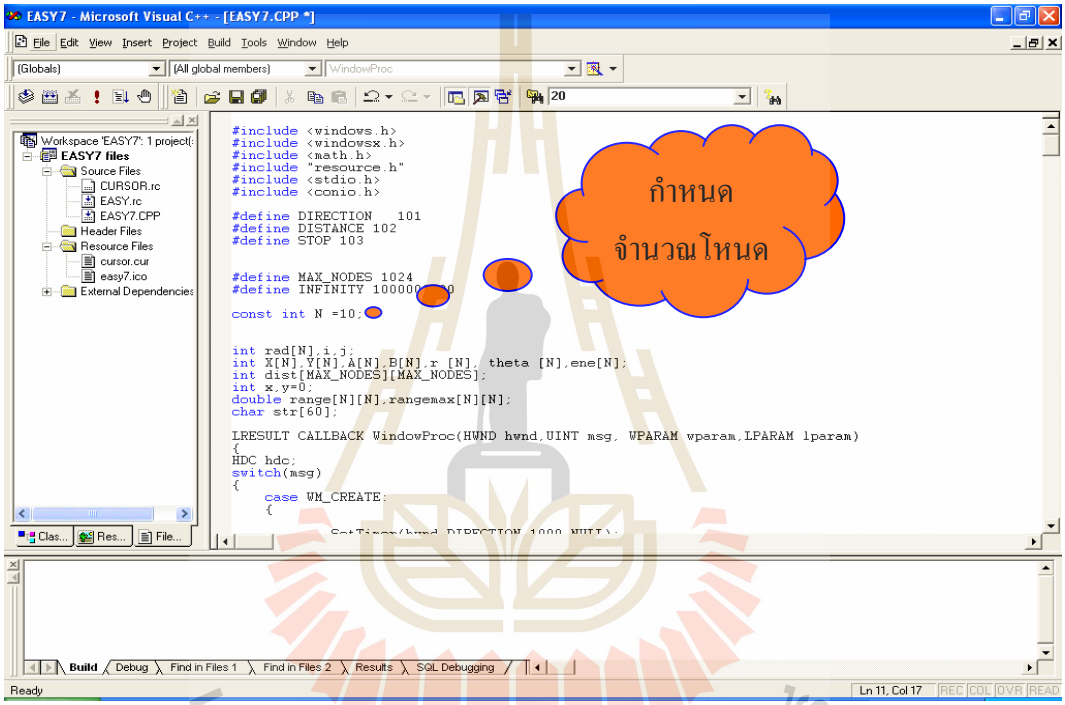
(mobile ad hoc network)

ด้วย โปรแกรมวิชวลซีพลัสพลัส หกจุดศูนย์ (Visual C++ 6.0)

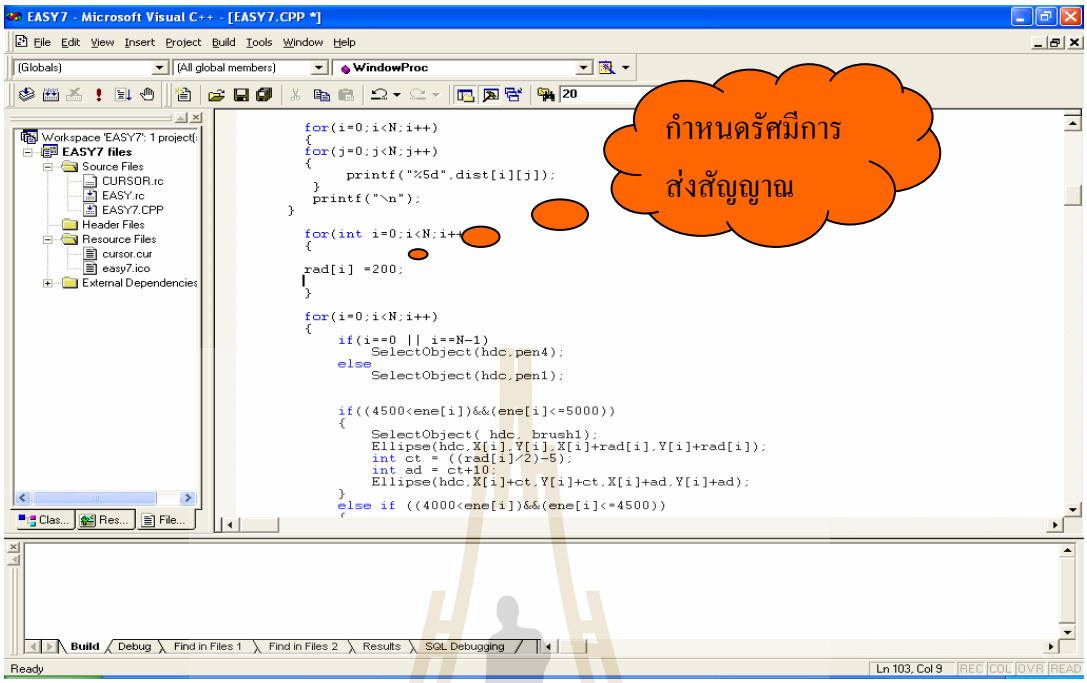
1. วิธีการใช้โปรแกรม

1. ดับเบิลคลิกไอคอนที่มีนามสกุล เป็น .dsw ดังรูป

2. กำหนดจำนวณโหนดในตัวโปรแกรม



3. กำหนดรัศมีการส่งสัญญาณ ค่าที่ได้คือค่าของเส้นผ่านศูนย์กลาง ค่ารัศมีต้องทำการหารสอง



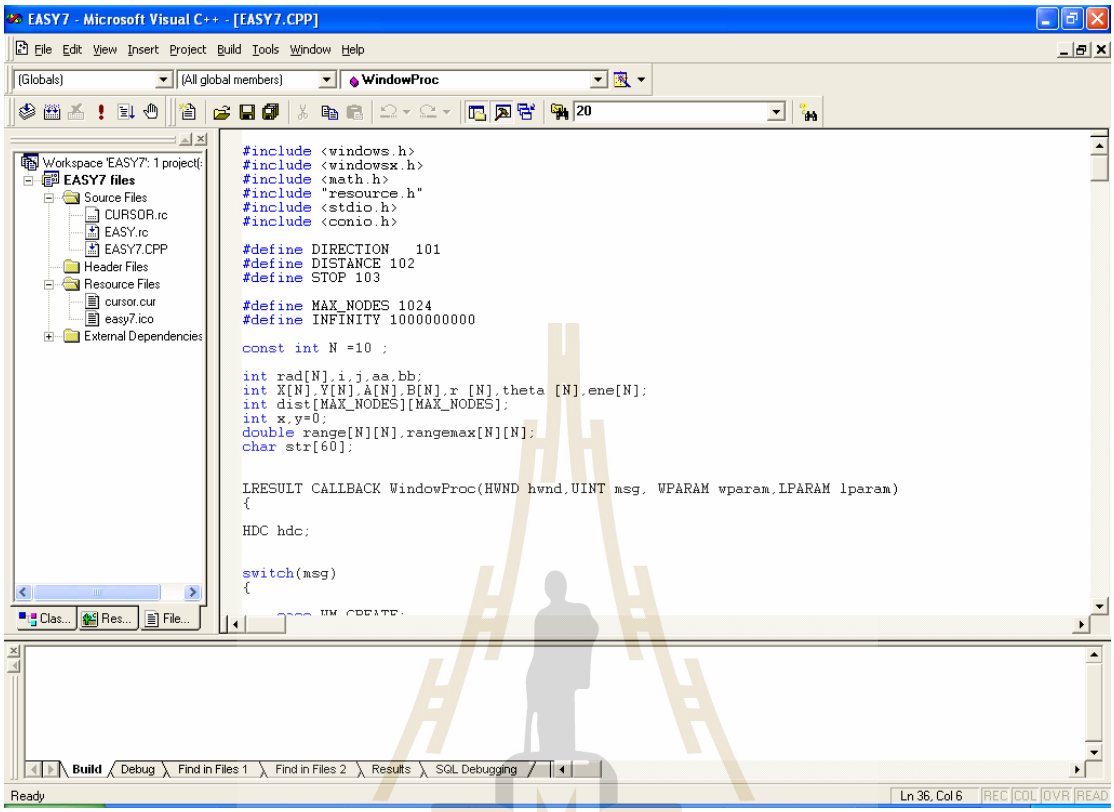
4. คลิกที่รูป  เพื่อทำการคอมไพล์ หรือกด ctrl+f7 หรือไปที่ build และ build easy7.exe หรือกด f7

5. ทำการรันโปรแกรมโดยกดที่  เพื่อรันโปรแกรม หรือไปที่ build และ execute easy7.exe หรือ กด ctrl+f5

6. เมื่อต้องการหยุดเพื่อดูผลก็กดที่ สเปสบาร์ (space bar) และเมื่อจะให้โปรแกรมดำเนินการต่อก็ให้กดปุ่มใด ๆ ก็ได้โปรแกรมก็จะทำงานต่อไป

7. เมื่อต้องการปิดโปรแกรมให้กด close เพื่อจบการทำงาน

2.โปรแกรมแบบจำลองโครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network)



รูปที่ 3 หน้าตาของโปรแกรมวิชวลซีพลัสพลัสหกจุดศูนย์ (Visual C++6.0)

1. ทำการรวม (include) ส่วนที่ใช้ในโปรแกรมไว้ที่ต้นไฟล์

```
#include <windows.h>
#include <windowsx.h>
#include <math.h>
#include "resource.h"
#include <stdio.h>
#include <conio.h>
```

2. กำหนดค่าคงที่ ให้กับโปรแกรม

```
#define DIRECTION   101
#define DISTANCE 102
#define STOP 103


#define MAX_NODES 1024
#define INFINITY 1000000000
```

3. ประกาศตัวแปร

```
const int N =10 ;
int rad[N],i,j;
int X[N],Y[N],A[N],B[N],r [N],theta [N],ene[N];
int dist[MAX_NODES][MAX_NODES];
int x,y=0;
double range[N][N],rangemax[N][N];
char str[60];
```

4. ส่วนเริ่มต้นของวินโดพรอค (WindowProc)

```
LRESULT CALLBACK WindowProc(HWND hwnd,UINT msg, WPARAM wparam,LPARAM
lparam)
{
HDC hdc;
switch(msg)
{
```

5. กำหนดเวลา และระดับพลังงานตั้งต้นของโปรแกรม

```
        case WM_CREATE:
        {
                        SetTimer(hwnd,DIRECTION,800,NULL);
                        SetTimer(hwnd,DISTANCE,40,NULL);
```

```
                    SetTimer(hwnd,STOP,1000,NULL);
                    for(i=0;i<N;i++)
                    {
                            X[i]=rand()%795;
                            Y[i]=rand()%549;
                            ene[i]=5000;
                    }
        }
        break;
```

6. ทำการตรวจจับเมสเซสดับเบิ้ลยูเอ็มคอมมาน (Message WM_COMMAND) และตรวจสอบว่า wparam เป็นเมนูที่เราเลือกหรือไม่ ถ้าใช่ก็สั่งให้โปรแกรมจบการทำงาน

```
        case WM_COMMAND:
        {
                    if(wparam==ID_FILE_EXIT) DestroyWindow(hwnd);break;

        }
        break;
```

7. เป็นส่วนที่ใช้วาดภาพและแสดงออกทางจอแสดงผล

```
        case WM_PAINT:
        {

         hdc=GetDC(hwnd);
        Rectangle (hdc, 0, 0, 795, 549);
        TextOut(hdc,200,530,"Please Key Space = STOP!  ::   Any Key = Continue",52  )
```

8. สร้างปากกาที่ใช้วาดเส้น

```
                    HPEN pen1=CreatePen(PS_DOT,1,RGB(0,0,0));
```

```
HPEN pen2=CreatePen(PS_SOLID,5,RGB(236,243,43));
HPEN pen3=CreatePen(PS_SOLID,6,RGB(0,0,255));
HPEN pen4=CreatePen(PS_DOT,5,RGB(0,0,255));
HPEN pen6=CreatePen(PS_SOLID,1,RGB(0,0,0));
HPEN dapen, pen5=CreatePen(PS_DOT,5,RGB(0,0,0));
dapen = (HPEN) SelectObject( hdc, pen5);
```

9. สร้างแปรงที่ใช้ลงสีให้กับวงกลม

```
HBRUSH brush1=CreateSolidBrush(RGB(5,7,101));
HBRUSH brush2=CreateSolidBrush(RGB(6,93,4));
HBRUSH brush3=CreateSolidBrush(RGB(104,5,12));
HBRUSH brush4=CreateSolidBrush(RGB(22,37,233));
HBRUSH brush5=CreateSolidBrush(RGB(23,218,32));
HBRUSH brush6=CreateSolidBrush(RGB(243,36,9));
HBRUSH brush7=CreateSolidBrush(RGB(14,217,247));
HBRUSH brush8=CreateSolidBrush(RGB(243,139,245));
HBRUSH brush9=CreateSolidBrush(RGB(248,240,27));
HBRUSH brush10=CreateSolidBrush(RGB(255,255,255));
HBRUSH dabrush;
dabrush = (HBRUSH) SelectObject (hdc, brush1);
```


10. กำหนดรัศมีของวงกลม

```
for(int i=0;i<N;i++)
{
        rad[i] = 200;
}
```

11. เลือกปากกาแท่งแรกกับแท่งสุดท้ายพิจารณา ให้ใช้ ปากา 4 นอกจากนั้นให้ใช้ ปากา 1 แล้วจึงเข้าสู่เงื่อนไขของการใช้ระดับพลังงาน

4501-5000 ใช้ แปรง 1 ระบายสี
4001-4500 ใช้ แปรง 2 ระบายสี
3501-4000 ใช้ แปรง 3 ระบายสี
3001-3500 ใช้ แปรง 4 ระบายสี
2501-3000 ใช้ แปรง 5 ระบายสี
2001-2500 ใช้ แปรง 6 ระบายสี
1501-2000 ใช้ แปรง 7 ระบายสี
1001-1500 ใช้ แปรง 8 ระบายสี
501-1000 ใช้ แปรง 9 ระบายสี
น้อยกว่า 501ใช้ แปรง 10 ระบายสี

```
for(i=0;i<N;i++)
        {
                if(i==0 || i==N-1)
                        SelectObject(hdc,pen4);
                else
                        SelectObject(hdc,pen1);

                if((4500<ene[i])&&(ene[i]<=5000))
                {
                        SelectObject( hdc, brush1);
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
                else if ((4000<ene[i])&&(ene[i]<=4500))
                {
                        SelectObject( hdc, brush2);;
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
                else if ((3500<ene[i])&&(ene[i]<=4000))
```

```
{
        SelectObject( hdc, brush3);;
        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
        int ct = ((rad[i]/2)-5);
        int ad = ct+10;
        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
}
else if ((3000<ene[i])&&(ene[i]<=3500))
{
        SelectObject( hdc, brush4);;
        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
        int ct = ((rad[i]/2)-5);
        int ad = ct+10;
        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
}
else if ((2500<ene[i])&&(ene[i]<=3000))
{
        SelectObject( hdc, brush5);
        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
        int ct = ((rad[i]/2)-5);
        int ad = ct+10;
        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
}
else if ((2000<ene[i])&&(ene[i]<=2500))
{
        SelectObject( hdc, brush6);
        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
        int ct = ((rad[i]/2)-5);
        int ad = ct+10;
        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
}
else if ((1500<ene[i])&&(ene[i]<=2000))
```

```
                {
                        SelectObject( hdc, brush7);;
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
                else if ((1000<ene[i])&&(ene[i]<=1500))
                {
                        SelectObject( hdc, brush8);;
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
                else if ((500<ene[i])&&(ene[i]<=1000))
                {
                        SelectObject( hdc, brush9);;
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
                else
                {
                        SelectObject( hdc, brush10);
                        Ellipse(hdc,X[i],Y[i],X[i]+rad[i],Y[i]+rad[i]);
                        int ct = ((rad[i]/2)-5);
                        int ad = ct+10;
                        Ellipse(hdc,X[i]+ct,Y[i]+ct,X[i]+ad,Y[i]+ad);
                }
        }
```

12. คำนวณหาจุดศูนย์กลางของวงกลม

```
for(i=0;i<N;i++)
        {
                A[i] = ((X[i]+(X[i]+rad[i]))/2);
                B[i] = ((Y[i]+(Y[i]+rad[i]))/2);
```

13. คำนวณหาเส้นทางระหว่างจุดศูนย์กลางของวงกลมสองวง และกำหนดให้ เป็นอนันต์ (Infinity)

```
for (int s=0;s<N;s++)
{
range[i][s] = sqrt (((A[i]-A[s])*(A[i]-A[s]))+((B[i]-B[s])*(B[i]-B[s])));
rangemax[i][s]=((rad[i]/2)+(rad[s]/2));
dist[i][s]=INFINITY;
```

14. ถ้าหากว่าส่วนของวงกลมมีส่วนที่ซ้อนทับกันให้ลากเส้นจากจุดศูนย์กลางของวงกลมหนึ่งไปยังอีกวงกลมหนึ่ง และถ้ามีการเชื่อมต่อพลังงานจะลดลงทีละ1 จูล

```
if ((rangemax[i][s]>range[i][s])&&(ene[i] >500 )&&(ene[s]>500))
        {
                SelectObject( hdc, pen2);
                MoveToEx(hdc,A[s],B[s],NULL);
                LineTo(hdc,A[i],B[i]);
                ene[i]=ene[i]-1 ;
```

15. ส่วนของการเลือกเส้นทาง ไดจิคตราอัลกอริทึม (Dijkstra algorithm) ที่ใช้ในการคำนวณหาเส้นทางที่สั้นที่สุด

```
dist[i][s]=1;
struct state
{
        int predecessor;
        int lenght;
```

```
            int label;
}
state[MAX_NODES];
int n;
n=N;
  int i,k,min,t,s=0,path[100];
struct state *p;
t=n-1;
for(p=&state[0];p<&state[n];p++)
{
            p->predecessor = -1;
            p->lenght = INFINITY;
            p->label = 1;
}
state[t].lenght=0;
state[t].label = 0;
k=t;
do
{
            for(i=0;i<n;i++)
             {
                        if(dist[k][i]!=0 && state[i].label==1)
                        {
            if(state[k].lenght+ dist[k][i] < state[i].lenght)
                                    {
            state[i].predecessor = k;
            state[i].lenght = state[k].lenght + dist[k][i];
                                    }
                        }
            }
            k=0;min = INFINITY;
            for(i=0;i<n;i++)
```

```
                {
                 if(state[i].label==1 && state[i].lenght<min)
                        {
                          min = state[i].lenght;
                          k = i;
                        }
                }
           state[k].label = 0;
         }while(k!=s);

         i=0;k=s;
         do
         {
                path[i] = k;

                k = state[k].predecessor;
```

16. แสดงค่าระดับพลังงานปัจจุบันออกทาง จอแสดงผล

```
                        sprintf(str,"NODE 1  :%d",ene[0]);
TextOut(hdc,10,10,str,strlen(str));
                        sprintf(str,"NODE 2  :%d",ene[1]);
TextOut(hdc,10,25,str,strlen(str));
                        sprintf(str,"NODE 3  :%d",ene[2]);
TextOut(hdc,10,40,str,strlen(str));
                        sprintf(str,"NODE 4  :%d",ene[3]);
TextOut(hdc,10,55,str,strlen(str));
                        sprintf(str,"NODE 5  :%d",ene[4]);
TextOut(hdc,10,70,str,strlen(str));
                        sprintf(str,"NODE 6  :%d",ene[5]);
TextOut(hdc,10,85,str,strlen(str));
                        sprintf(str,"NODE 7  :%d",ene[6]);
```

```
TextOut(hdc,10,100,str,strlen(str));
                sprintf(str,"NODE 8  :%d",ene[7]);
TextOut(hdc,10,115,str,strlen(str));
                sprintf(str,"NODE 9  :%d",ene[8]);
TextOut(hdc,10,130,str,strlen(str));
                sprintf(str,"NODE10 :%d",ene[9]);
TextOut(hdc,10,145,str,strlen(str));
```

17. วาดรูปสี่เหลี่ยมเหลี่ยมเล็ก ๆ แสดงออกทางจอภาพทางด้านมุมขวาแสดงระดับพลังงาน ณ ระดับต่าง ๆ

```
        }
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush1);
                Rectangle(hdc,700,10,680,20);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush2);
                Rectangle(hdc,700,30,680,40);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush3);
                Rectangle(hdc,700,50,680,60);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush4);
                Rectangle(hdc,700,70,680,80);
        }
        {
```

```
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush5);
                Rectangle(hdc,700,90,680,100);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush6);
                Rectangle(hdc,700,110,680,120);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush7);
                Rectangle(hdc,700,130,680,140);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush8);
                Rectangle(hdc,700,150,680,160);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush9);
                Rectangle(hdc,700,170,680,180);
        }
        {
                SelectObject(hdc,pen6);
                SelectObject(hdc, brush10);
                Rectangle(hdc,700,190,680,200);
        }
        {
        TextOut(hdc,703,5,"en:4500-5000",15 );
        }
```

```
{
TextOut(hdc,703,25,"en:4000-4500",15 );
}
{
TextOut(hdc,703,45,"en:3500-4000",15 );
}
{
TextOut(hdc,703,65,"en:3000-3500",15  );
}
{
TextOut(hdc,703,85,"en:2500-3000",15  );
}
{
TextOut(hdc,703,105,"en:2000-2500",15 );
}
{
TextOut(hdc,703,125,"en:1500-2000",15  );
}
{
TextOut(hdc,703,145,"en:1000-1500",15 );
}
{
TextOut(hdc,703,165,"en:500-1000",15 );
}
{
TextOut(hdc,703,185,"en:<500",15 );
}
```

18. ทำการลากเส้นในส่วนที่เป็นเส้นทางที่สั้นที่สุด (Shortest-path) แล้วแสดงผลโหนดที่ถูกเลือก จำนวนโหนดที่ใช้ ออกทางจอแสดงผล

```
i++;
}
```

```
                                        while(k>=0);
                                                        for(x=0;x<i;x++)
                                                        {
                                                                if((x+1)<i)
                                                                 {


                                                                SelectObject( hdc, pen3);


        MoveToEx(hdc,A[path[x]],B[path[x]],NULL);
        LineTo(hdc,A[path[x+1]],B[path[x+1]]);
        sprintf(str,"NODE Choose :%d",path[x] );
        TextOut(hdc,150,10,str,strlen(str));
        sprintf(str,"Conecting Shortestpath   :%d",i );
        TextOut(hdc,450,10,str,strlen(str));
        sprintf(str,"N.Center Use  :%d",x );
        TextOut(hdc,300,10,str,strlen(str));



                                                                }
                                                        }
                                        }
                }
        }
```

19. ใช้ ปากกา5 และ แปรง 1 ในการวาดรูปสี่เหลี่ยม

```
                                        SelectObject( hdc, dapen);
                                        SelectObject( hdc, dabrush);
```

20. ทำการทำลายปากกาและแปรง หลังเลิกใช้งานแล้ว

```
                    DeleteObject(pen1);
                    DeleteObject(brush1);
                    DeleteObject(brush2);
                    DeleteObject(brush3);
                    DeleteObject(brush4);
                    DeleteObject(brush5);
                    DeleteObject(brush6);
                    DeleteObject(brush7);
                    DeleteObject(brush8);
                    DeleteObject(brush9);
                    DeleteObject(brush10);
                    DeleteObject(pen2);
                    DeleteObject(pen3);
                    DeleteObject(pen4);
                    DeleteObject(pen5);
                    DeleteObject(pen6);
                    ReleaseDC(hwnd,hdc);

        }
        break;
```

21. ตรวจสอบว่ามีการกดสเปส (SPACE) หรือเอสเคป (ESCAPE) หรือ ไม่ ถ้ากดสเปสบาร์ (SPACE BAR) ระบบจะหยุดการทำงานไว้ระยะหนึ่งก่อน ถ้ากดเอสเคป (ESCAPE) จะสั่งให้โปรแกรมจบการทำงาน

```
        case WM_KEYDOWN:
        {
                    if (wparam==VK_SPACE )
                    {
                            system ("pause");
                    }
```

```
                    if (wparam==VK_ESCAPE)
                    {
                            DestroyWindow(hwnd);
                    }
        }
        break;
```

22. ส่วนของการตรวจจับเวลาและการเคลื่อนที่ของโหนด พร้อมทั้ง บังคับไม่ให้เคลื่อนที่ออกนอกกรอบที่กำหนดไว้

```
        case WM_TIMER :
        {
                                for(i=0;i<N;i++)
                                {
                                    if (wparam == DIRECTION)

                                     theta [i] = rand ()%360;
                                    r [i] = ((rand ()%2)-8);
                                    if(X[i]<=0-(rad[i]/2))
                                    {
                                            X[i]++;
                                    }
                                    else
                                    if(X[i]>=(798-(rad[i]/2)))
                                    {
                                            X[i]--;
                                    }
                                    else
                                    if(Y[i]<=0-(rad[i]/2))
                                    {
                                            Y[i]++;
                                    }
```

```
                                        else
                                        if (Y[i]>=(550-(rad[i]/2)))
                                        {
                                                Y[i]--;
                                        }
                                        else
                                        {
                                        X[i] = X [i] + (r [i]*cos(theta[i]));
                                        Y[i] = Y [i] + (r [i]*sin(theta[i]));
                                        }
                        }
                for(i=0;i<N;i++)
                        {
                                if(wparam==STOP)
                                {
                                        if(ene[i]>0)
                                        {
                                        ene[i]=ene[i]-1;
                                        }
                                }
                        }


                                ::InvalidateRect (hwnd, NULL, FALSE);
        }
break;
```

23. ทำการ KillTimer และจบการทำงานส่วนของ windowProc

```
case WM_DESTROY:
```

```
        {

                                KillTimer(hwnd,DIRECTION);
                                KillTimer(hwnd,DISTANCE);
                                KillTimer(hwnd,STOP);
                                PostQuitMessage(0);

        }
        break;

}
return(DefWindowProc(hwnd,msg,wparam,lparam));
}
```

24. ส่วนของวินเมน (Winmain) ของโปรแกรม

```
int WINAPI WinMain( HINSTANCE hinstance, HINSTANCE hprevinstance, LPSTR lpcmdline,
int ncmdshow)
{
        WNDCLASSEX winclass;
        winclass.cbSize = sizeof(WNDCLASSEX);
        winclass.style = CS_VREDRAW|CS_HREDRAW| CS_OWNDC|CS_DBLCLKS;
        winclass.lpfnWndProc = WindowProc;
        winclass.cbClsExtra     = 0;
        winclass.cbWndExtra     = 0;
        winclass.hInstance      = hinstance;
        winclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
        winclass.lpszClassName = "MyWin";
        winclass.lpszMenuName  = MAKEINTRESOURCE(IDR_MAINMENU);
        winclass.hIcon = LoadIcon(hinstance,MAKEINTRESOURCE(IDI_EASY7));
        winclass.hCursor = LoadCursor(hinstance,MAKEINTRESOURCE(IDC_CURSOR));
        winclass.hIconSm = LoadIcon(hinstance,MAKEINTRESOURCE(IDI_EASY7));
```

```
RegisterClassEx(&winclass);


HWND hwnd;
MSG msg;
```

25. สร้างหน้าต่าง โดยประกาศตัวแปร hwnd ขึ้นมาก่อน โดยให้เป็นตัวแปรแบบ HWND Structure ซึ่งใช้ในการเก็บรายละเอียดของหน้าต่างหลักของโปรแกรม

```
hwnd = CreateWindowEx(WS_EX_TOPMOST, "Mywin", "GUI for studying energy usage in MANETs",
WS_OVERLAPPED|WS_CAPTION|WS_MINIMIZEBOX|WS_SYSMENU,0,0,800,600,NULL, NULL,hinstance,NULL);
```

26. เมื่อสร้างหน้าต่างแล้ว เราก็จะเรียกให้หน้าต่างประกฏ โดยเราจะเรียกใช้ฟังก์ชั่น ShowWindow

```
ShowWindow(hwnd,ncmdshow);
```

27. เมื่อ ShowWindow แล้วอันดับต่อไป เราก็จะเรียก UpdateWindow เพื่อเป็นการส่งสัญญาณให้หน้าต่างทำงาน

```
UpdateWindow(hwnd);
```

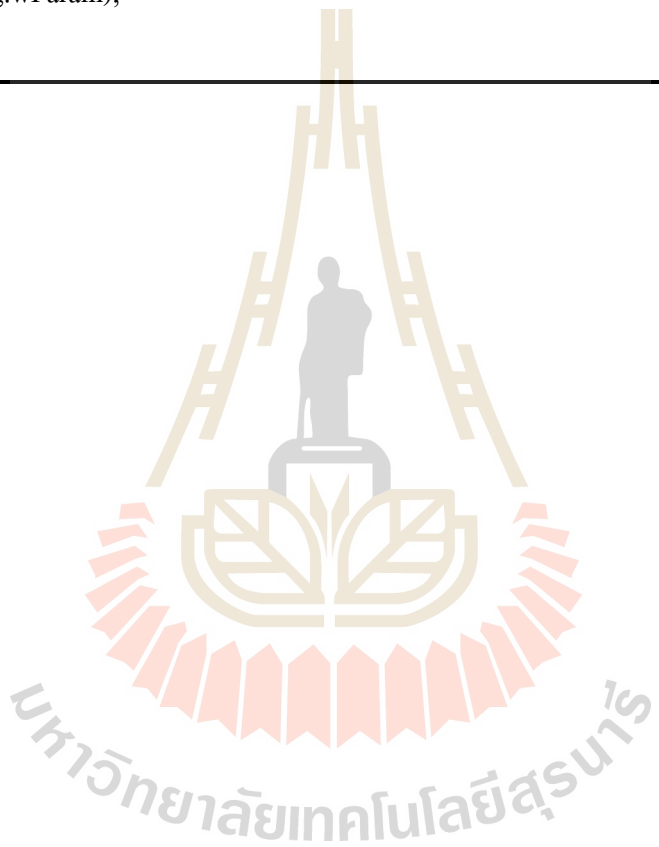
28. เข้าสู่ระบบการจัดการ Message ของโปรแกรม เมื่อหน้าต่างถูกสร้างและวาดที่จอภาพแล้ว ขั้นตอนต่อไปจะเป็นการเข้าสู่ระบบ Message ของโปรแกรมระบบ Message จะใช้ในการวนซ้ำเพื่อรับ Message จากนั้น เป็นการเข้าลูป Message โดย PeakMessage จะตรวจสอบว่ามี message มาหรือไม่ ถ้ามีมันจะ Translate และ Dispatch ไปยังเงื่อนไข switch case เพื่อจัดการ message ต่อไป

```
while(1)
{



        if (PeekMessage(&msg,NULL,0,0,PM_NOREMOVE))
        {
                if(!GetMessage(&msg,NULL,0,0))
```

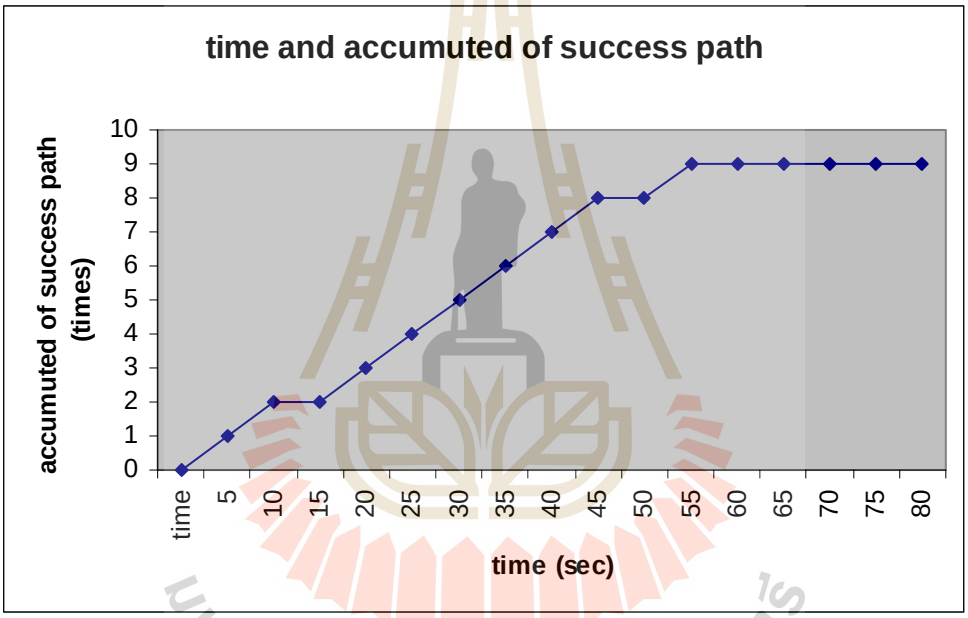
```
                    return msg.wParam;
                    TranslateMessage(&msg);
                    DispatchMessage(&msg);
                }
                else
                {
                }
        }
        return (msg.wParam);
}
```



บทที่ 3
ปัจจัยที่มีผลกระทบต่อระบบและวิเคราะห์ผล

ทำการทดสอบระบบจำลองเครือข่ายโดยใช้โปรแกรม การจำลองกราฟฟิคเพื่อศึกษาการใช้พลังงานในโครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc networks) ด้วยเงื่อนไขและข้อพิจารณาต่างๆ ดังต่อไปนี้ โดยกำหนดให้ 10 หน่วยพลังงาน = 1 จูล และ 1 pixel = 1 เมตร สถานที่ที่ใช้ในการทดสอบมีขนาด 800×600 ตารางเมตร

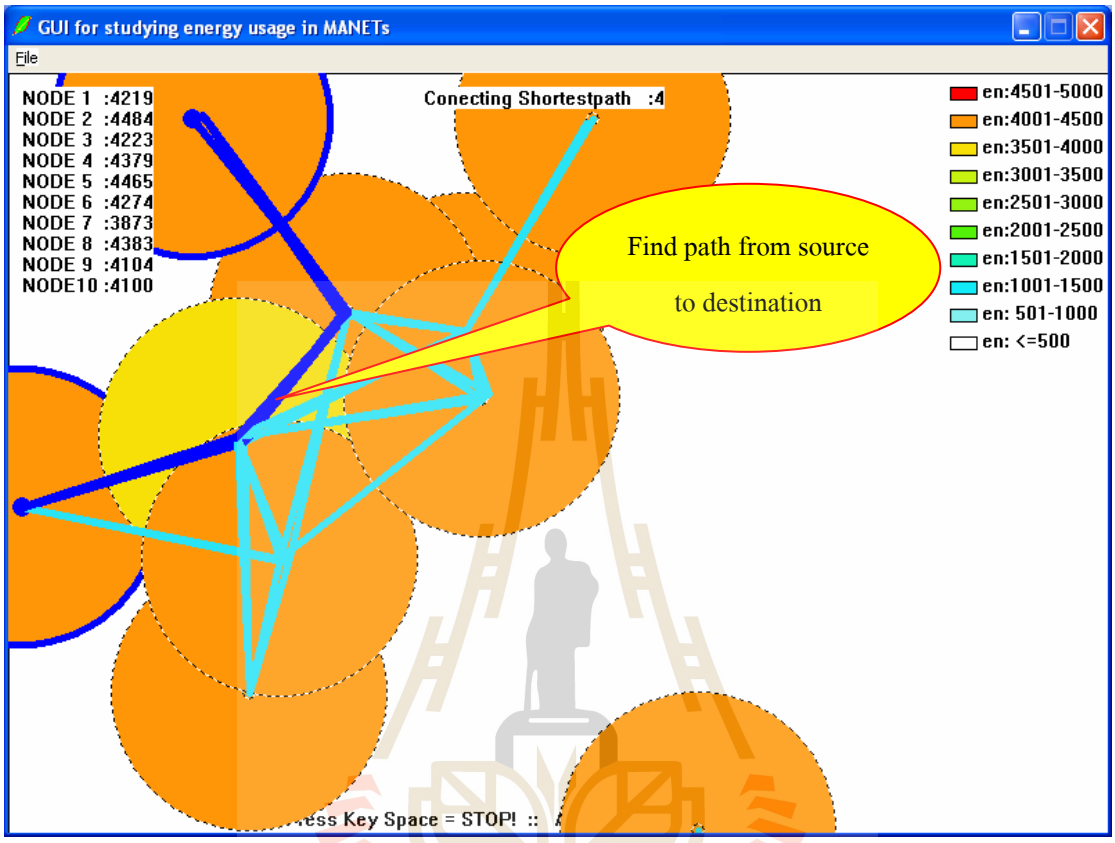
1. เวลาและความสำเร็จสะสม (time and accumuted of success path)



รูปที่ 4 กราฟแสดงความสัมพันธ์ระหว่าง เวลาและความสำเร็จสะสม (time and accumuted of success path)

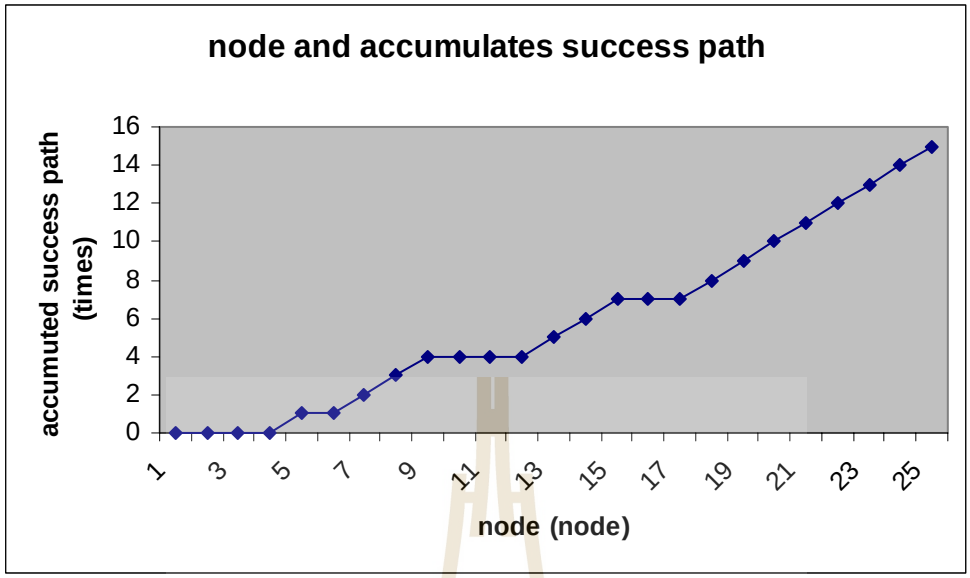
กรณีแรกทำการทดสอบระบบโดยจะหยุดโปรแกรมทุกๆ 5 วินาทีเพื่อตรวจสอบการเกิดเส้นทางที่สั้นที่จุด (shortest path) ระหว่างวงกลม2 วงที่พิจารณา จากกราฟจะพบว่าเมื่อเวลาเปลี่ยนไปโอกาสที่จะเกิดการเลือกเส้นทางที่สั้นที่สุด (shortest path) ก็ยังมีอยู่ในช่วงหนึ่งและเริ่มน้อยลงเรื่อย ๆ แต่พอเวลาผ่านไปจนกระทั่งถึงช่วงเวลาหนึ่งที่โหนดพิจาณาคือโหนดแรกและโหนดสุดท้ายตายกราฟจะคงที่ นั่นคือ ไม่มีการเกิดการเลือกเส้นทางที่สั้นที่สุด (shortest path) เกิดขึ้นอีกเลยแสดงว่าเวลามีผลต่อการเลือกเส้นทางเมื่อเวลาผ่านไป โอกาสในการเลือกเส้นทางก็จะ

ลดลง เพราะพลังงานในระบบลดลง (ระดับแบตเตอรี่ของโหนดที่ลดลงตามการใช้งานและเวลา) จนกระทั่งความสำเร็จสะสม (accumulates of success path) คงที่ เนื่องจากการเชื่อมต่อของโหนดไม่เกิดขึ้น



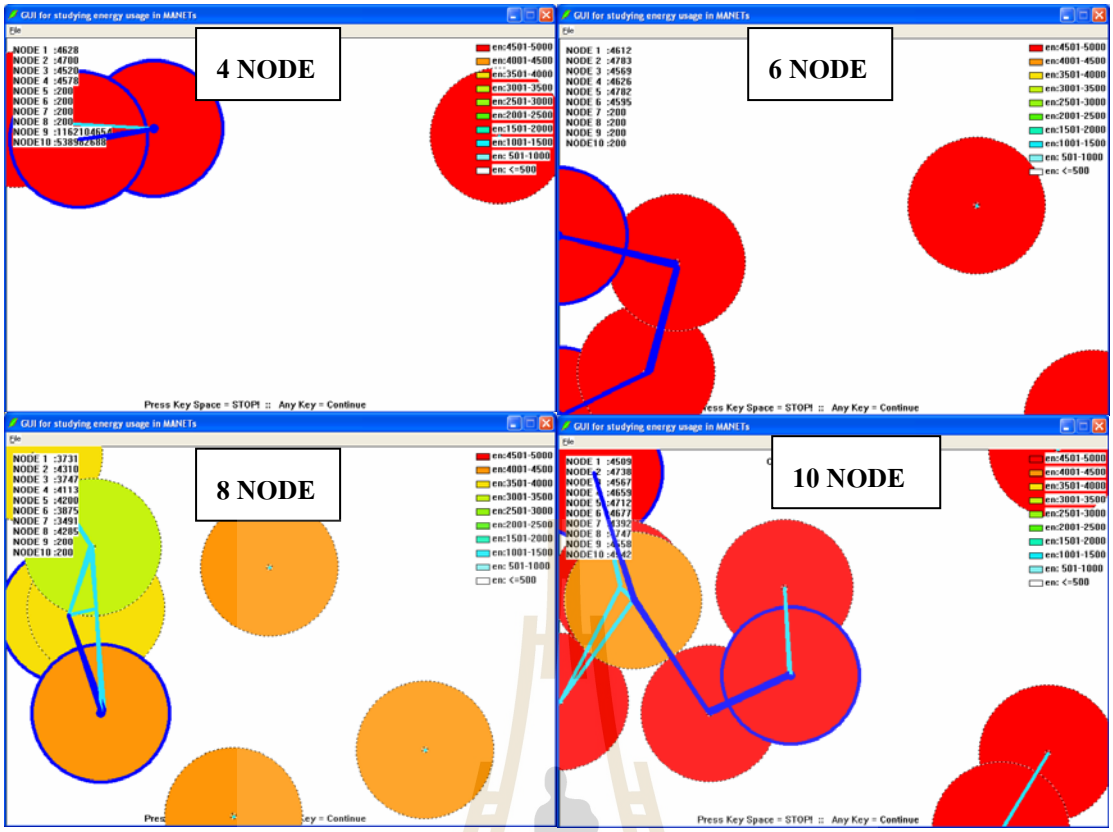
รูปที่ 5 แสดงการหาเส้นทางจากโหนดต้นทางสู่โหนดปลายทาง

2. จำนวนโหนดและความสำเร็จสะสม (node and accumuted of success path)



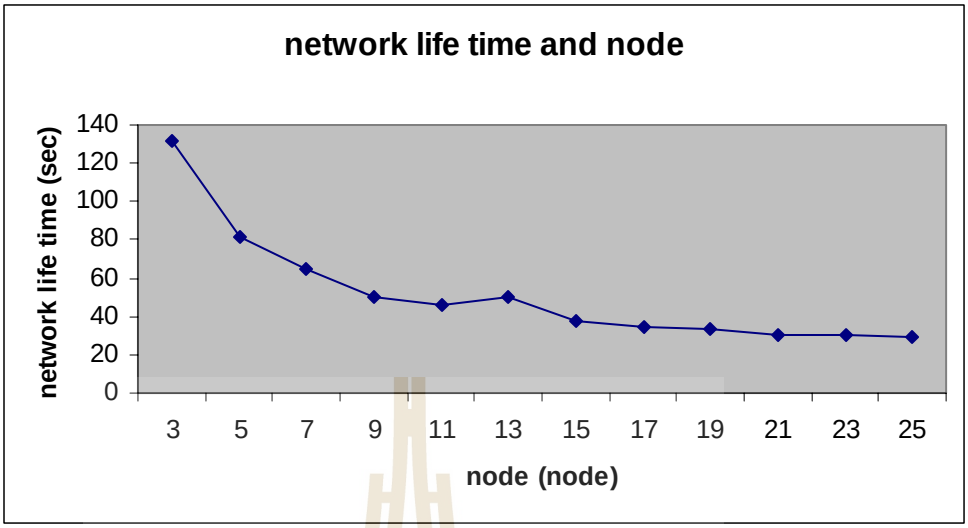
รูปที่ 6 กราฟความสัมพันธ์ ระหว่างจำนวนโหนดและความสำเร็จสะสม (node and accumuted of success path)

จากการเพิ่มจำนวนโหนดขึ้นเรื่อย ๆ ทีละ2 โหนดและทำการหยุดโปรแกรมที่ 5 วินาทีแรกของการรันโปรแกรม พบว่าโอกาสในการเลือกเส้นทางก็จะมากขึ้นเรื่อย ๆ สาเหตุเพราะว่าเมื่อเราเพิ่มจำนวนโหนดมากขึ้นตัวเลือกในการเชื่อมต่อก็มากขึ้นทราฟฟิค (traffic) ในระบบก็มากขึ้น แต่พื้นที่ว่างน้อยลงเรื่อย ๆ ดังนั้นโอกาสในการเลือกเส้นทางจึงมากขึ้นตามจำนวนโหนด จำนวนโหนดยิ่งมากเท่าไหร่ โอกาสการเชื่อมต่อระหว่างโหนดมีมากขึ้น ทำให้โอกาสทำให้โอกาสในการเลือกเส้นทางมีมากตามไปด้วย



รูปที่ 7 แสดงความแตกต่างเมื่อจำนวนโหนดมากขึ้น

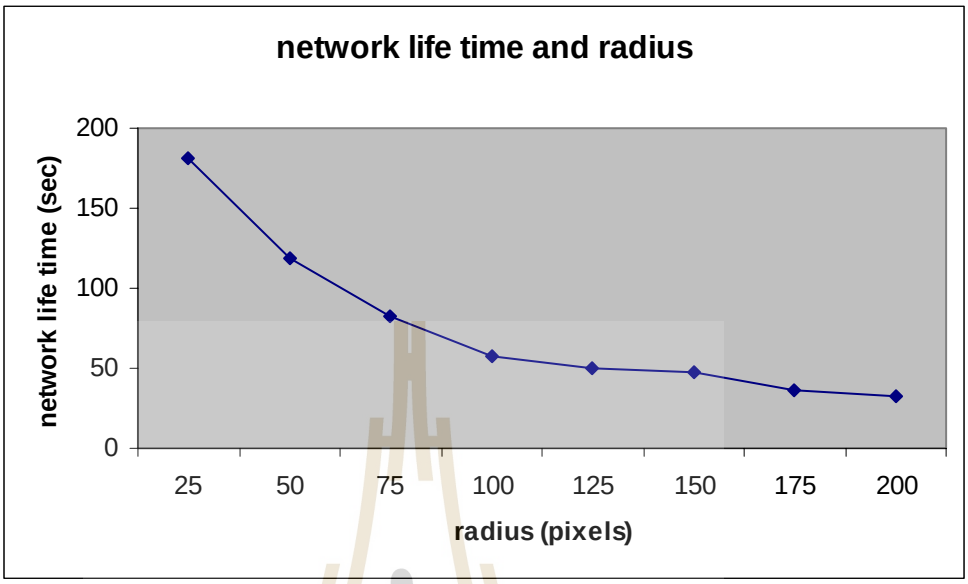
3. อายุขัยของเครือข่ายและจำนวนโหนด (network life time and node)



รูปที่ 8 กราฟแสดงความสัมพันธ์ระหว่างอายุขัยของเครือข่ายและจำนวนโหนด (network life time and node)

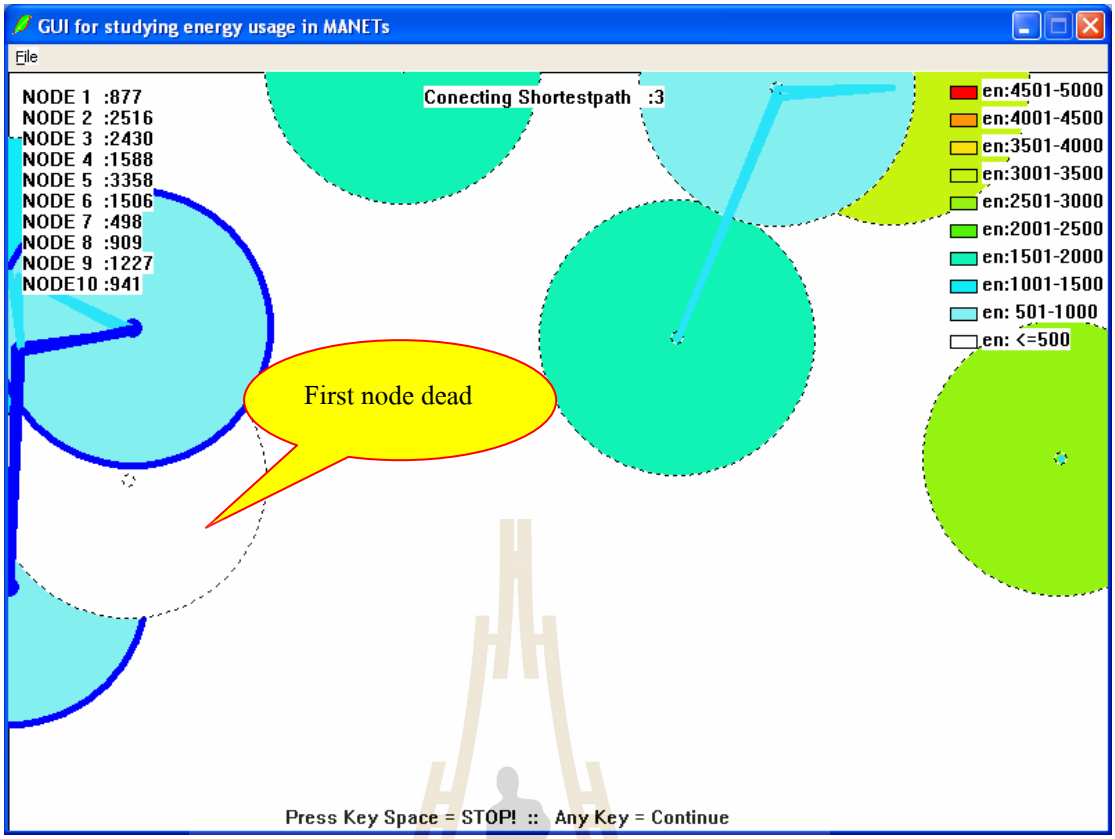
ทำการทดสอบโดยจับเวลาที่โหนดตัวแรกของเครือข่ายใช้พลังงานหมดไป จากการเพิ่มโหนดขึ้นเรื่อย ๆ จะพบว่าอายุของเครือข่ายมีแนวโน้มที่จะลดลงเรื่อย ๆ สาเหตุเนื่องมาจากเมื่อจำนวนโหนดเพิ่มขึ้น โอกาสที่แต่ละโหนดจะถูกต่อเชื่อมกันก็มีมากขึ้น และเมื่อโหนดแต่ละโหนดถูกต่อเชื่อมกันจะทำให้พลังงานลดลงอย่างรวดเร็ว และเมื่อจำนวนโหนดเพิ่มมากขึ้น ทำให้พลังงานถูกใช้มาก ทำให้อายุของเครือข่ายลดลงด้วย หรืออาจกล่าวได้ว่าโหนด ในเครือข่ายน้อย ๆ มีการเชื่อมต่อน้อยอ้างอิงจากการค้นพบที่สำเร็จน้อย เมื่อโอกาสการเชื่อมต่อน้อยลงโหนดก็ใช้พลังงานน้อย สำหรับเครือข่าย โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) เมื่อผู้ใช้งานมีความหนาแน่นมากขึ้นทำให้เกิดการเชื่อมต่อระหว่างโหนดมากขึ้น และสูญเสียพลังงานมากขึ้น โหนดจึงตายเร็ว แต่ในความเป็นจริงไม่ได้มีการเชื่อมต่อระหว่างโหนดทุกครั้งที่มีรัศมีของสัญญาณส่งถึงกัน

4. อายุขัยของเครือข่ายและรัสมีการส่งสัญญาณ (network life time and radius)



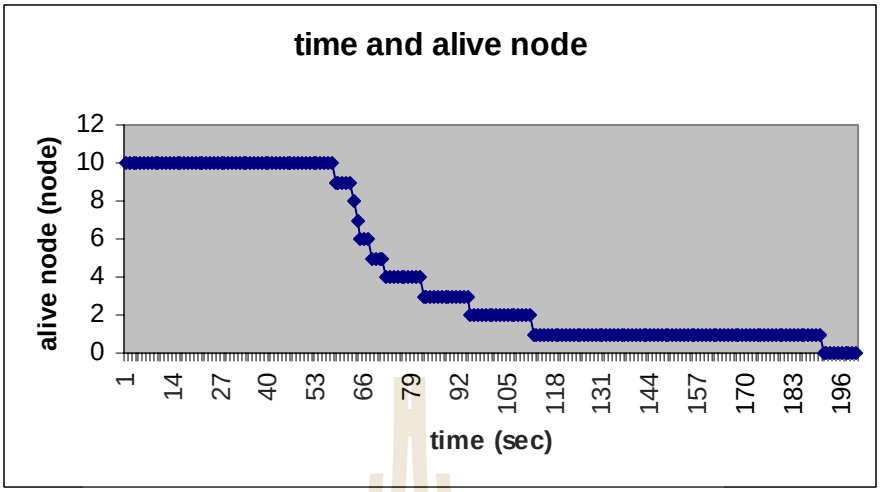
รูปที่ 9 กราฟแสดงความสัมพันธ์ระหว่างอายุขัยของเครือข่ายและรัสมีการส่งสัญญาณ (network life time and radius)

จากกราฟเป็นการทดสอบระบบโดยพิจารณาอายุขัยของเครือข่ายเมื่อแต่ละโหนดมีขนาดรัศมีต่างๆ พบว่าเมื่อโหนดมีรัศมีมากขึ้น แนวโน้มของอายุของเครือข่ายจะลดลง เนื่องจากเมื่อแต่ละโหนดมีขนาดรัศมีมากขึ้นทำให้โอกาสที่รัศมีของแต่ละโหนดสัมผัสหรือซ้อนทับกันก็มีมากขึ้นด้วย ทำให้เกิดการสร้างเส้นทางเชื่อมต่อระหว่างโหนดเป็นผลให้ระดับพลังงานของโหนดลดลงเร็วขึ้น เนื่องจากมีการใช้พลังงานเพื่องส่งต่อแพ็กเก็ต การทดสอบนี้กำหนดให้อายุของเครือข่ายคือเวลาขณะที่โหนดแรกตาย



รูปที่ 10 แสดงตัวอย่างโหนดแรกที่ตายในระบบ

5. กราฟแสดงความสัมพันธ์ระหว่างเวลาและจำนวนโหนดที่เหลืออยู่ในเครือข่าย (time of alive node)

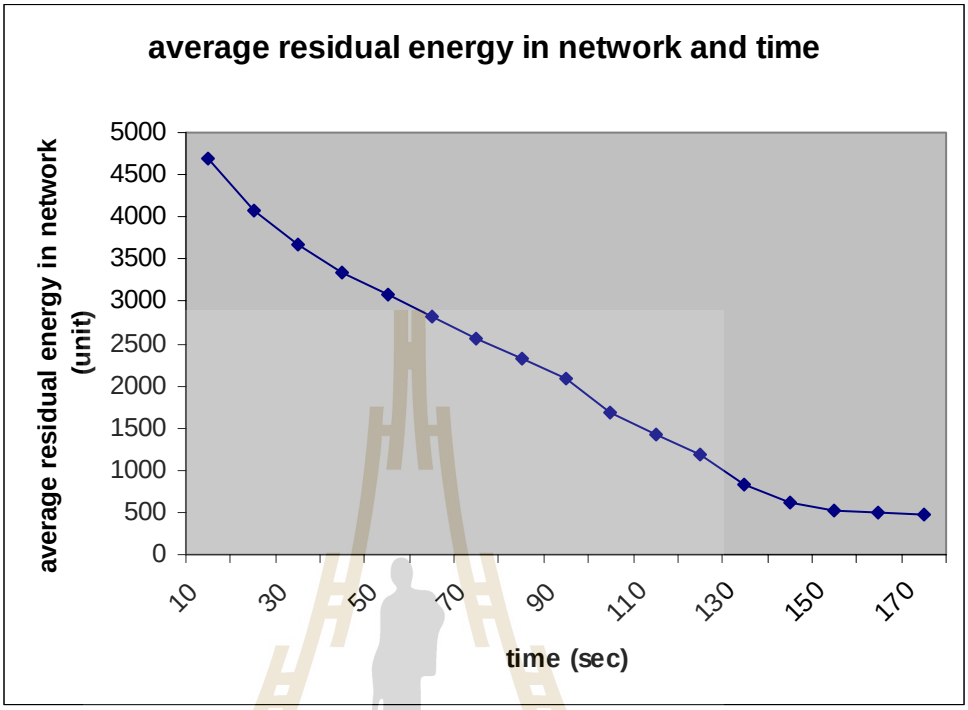


รูปที่ 11 กราฟแสดงความสัมพันธ์ระหว่างเวลาและจำนวนโหนดที่เหลืออยู่ในเครือข่าย (time of alive node)

ทำการทดสอบโดยการจับเวลาที่แต่ละโหนดตาย โดยจะสังเกตได้ว่าในช่วงแรกๆ แต่ละโหนดจะตายในเวลาไล่เลี่ยกัน ส่วนช่วงหลังๆ จะเริ่มห่างออกไปเนื่องจากว่าจำนวนโหนดลดน้อยลงทำให้โอกาสในการเชื่อมต่อกับโหนดต่างๆ ลดน้อยลง ดังนั้นโหนดหลังๆ จึงอยู่ได้นานขึ้น แต่เหลืออีกปัจจัยหนึ่งที่มีผลกับการลดระดับพลังงานนั่นคือระยะเวลาทำให้ระดับพลังงานยังคงลดลงเหมือนเดิมแต่จะค่อย ๆ ลดลง

6. กราฟแสดงความสัมพันธ์ระหว่างพลังงานเฉลี่ยและเวลา

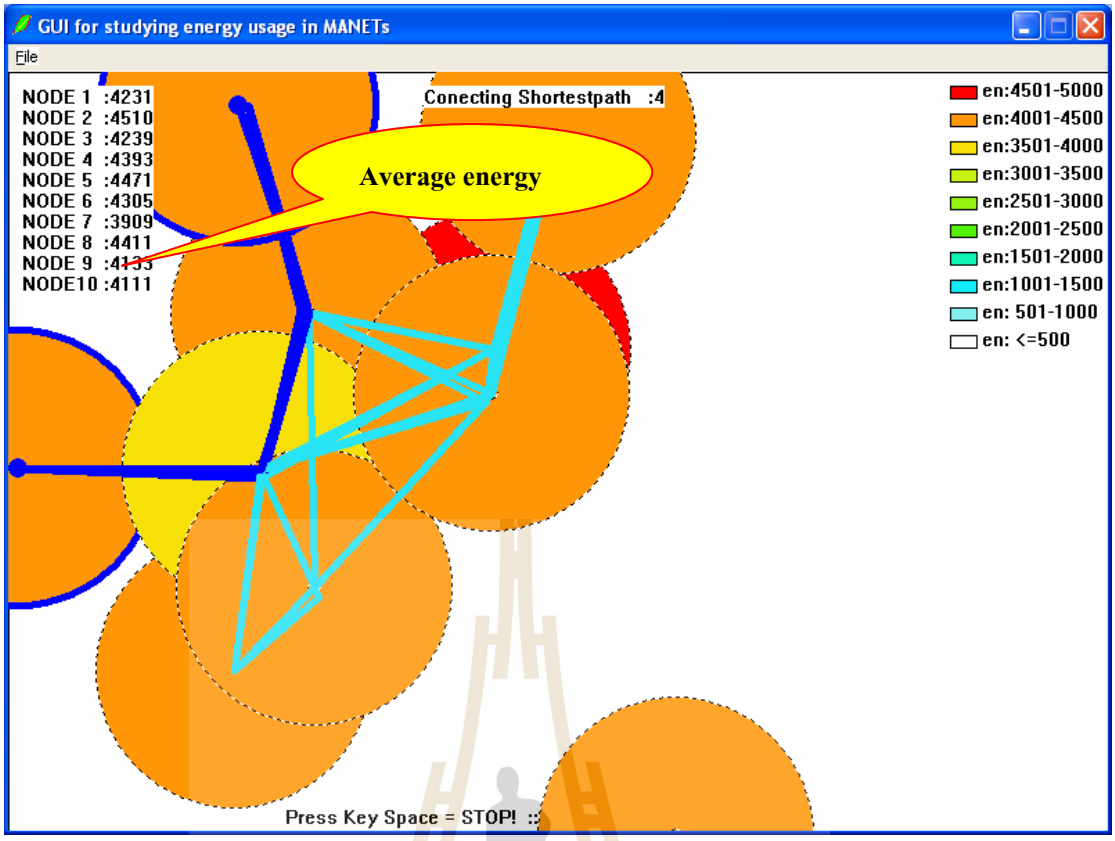
(average residual energy in network and time)



รูปที่ 12 กราฟแสดงความสัมพันธ์ระหว่างพลังงานเฉลี่ยและเวลา

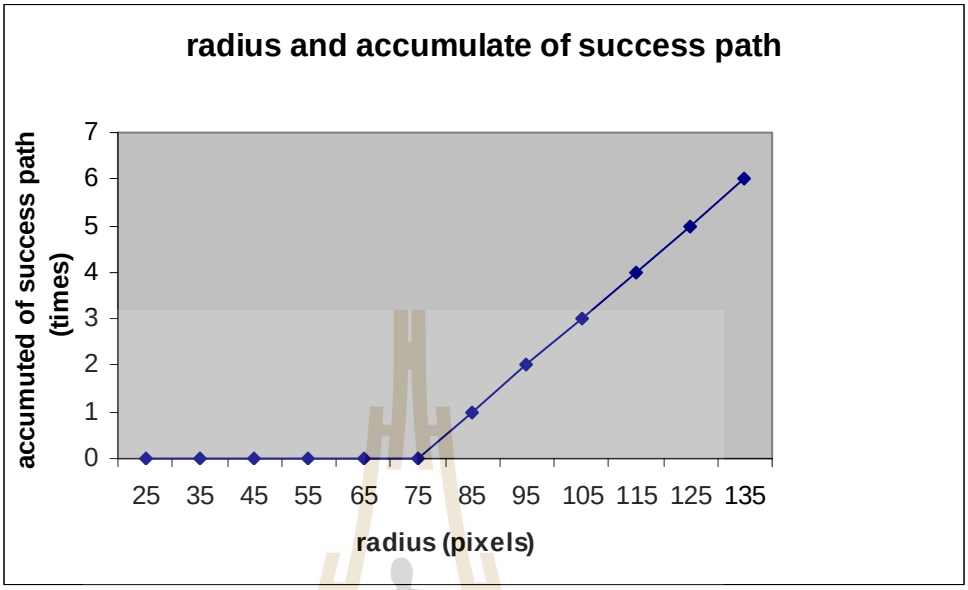
(average residual energy in network and time)

ทำการทดสอบระดับพลังงานเฉลี่ยของเครือข่ายโดยกำหนดให้เครือข่ายมีทั้งหมด 10 โหนด แล้วทำการหาค่าเฉลี่ยระดับพลังงานของเครือข่ายทุกๆ 10 วินาที พบว่าระดับพลังงานเฉลี่ยของเครือข่ายมีแนวโน้มลดลงอย่างต่อเนื่อง และในช่วงหลังๆ กราฟมีความชันน้อยลงเพราะว่าระดับพลังงานลดช้าลงเนื่องจากระดับแบตเตอร์รี่ลดลงจากการใช้งาน เมื่อเวลาเพิ่มมากขึ้นมีการใช้โหนดเพื่อส่งต่อแพ็กเก็ตมากขึ้น ดังนั้นพลังงานสะสมเฉลี่ย (residual energy) จึงลดลง



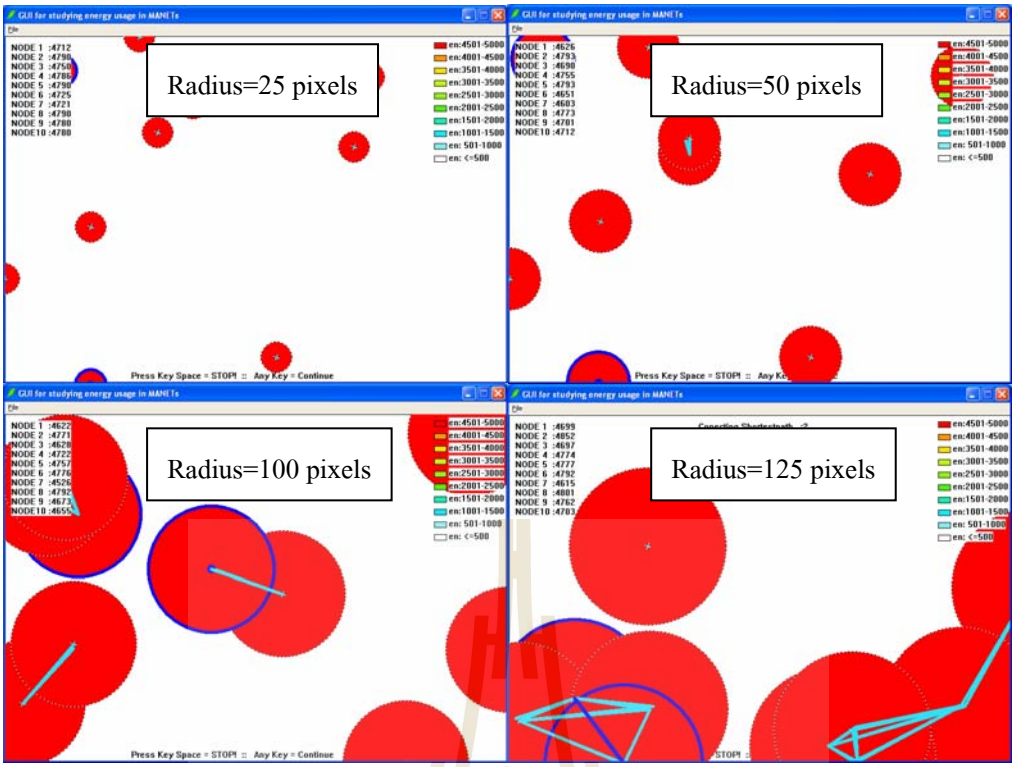
รูปที่ 13 แสดงระดับพลังงานที่เหลืออยู่ในระบบ

7. กราฟแสดงความสัมพันธ์ระหว่างรัศมีการส่งสัญญาณและความสำเร็จสะสม (radius and accumuted of success path)



รูปที่ 14 กราฟแสดงความสัมพันธ์ระหว่างรัศมีการส่งสัญญาณและความสำเร็จสะสม (radius and accumuted of success path)

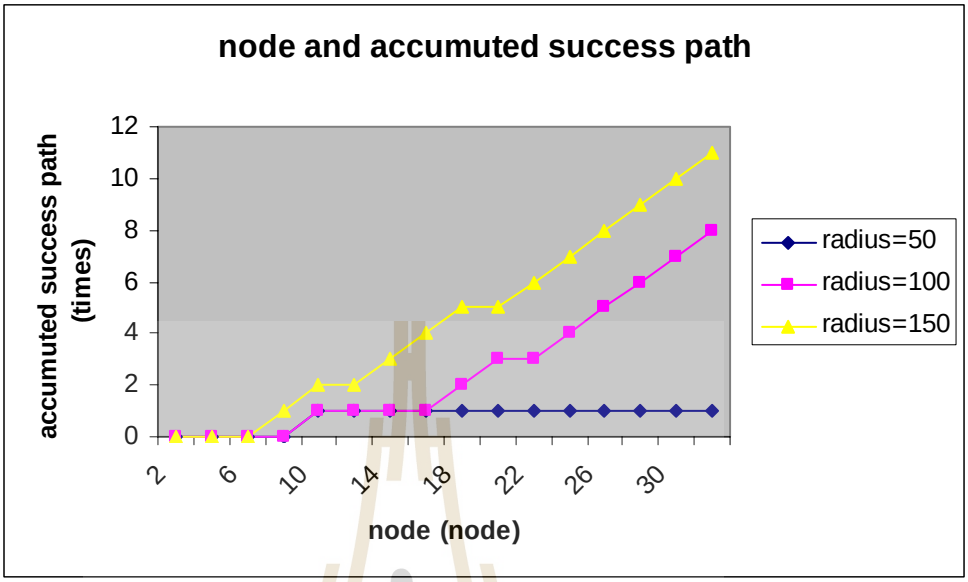
เมื่อทำการทดลองเพิ่มขนาดรัศมีขึ้นเรื่อยๆ และเก็บผลการทดลองที่เวลาทุก ๆ 10 วินาที พบว่ามีการเชื่อมต่อถึงกันระหว่างสองโหนดที่กำหนดให้ติดต่อสื่อสารกันโดยผ่านการเลือเส้นทางที่สั้นที่สุด (shortest-path) บ่อยครั้งขึ้น ทั้งนี้ก็เนื่องจากขนาดรัศมีของโหนดที่ใหญ่ขึ้นนั่นเอง



รูปที่ 15 แสดงความแตกต่างกันของโหนดในระบบเมื่อเพิ่มรัศมีการส่งสัญญาณ

8. กราฟแสดงความสัมพันธ์ระหว่างจำนวนโหนดและความสำเร็จ

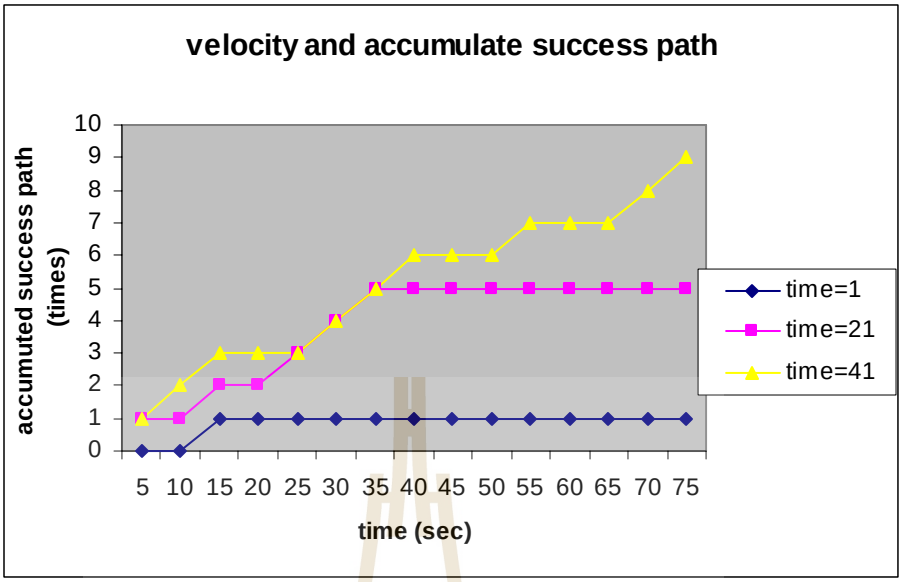
(node and accumuted of success path)



รูปที่ 16 กราฟแสดงความสัมพันธ์ระหว่างจำนวนโหนดและความสำเร็จสะสม (node and accumuted of success path)

จากกราฟเป็นการเปรียบเทียบให้เห็นว่าโหนดที่มีรัศมีโตกว่าจะเกิดการเลือกเส้นทางที่สั้นที่สุด (shortest path) ได้มากกว่าโหนดที่มีรัศมีขนาดเล็ก และในขณะเดียวกันจำนวนโหนดหรือจำนวนผู้ใช้งานในเครือข่ายก็มีผลในทำนองเดียวกันกับขนาดรัศมีของโหนด กล่าวคือเมื่อมีจำนวนโหนดมากขึ้นโอกาสที่จะเกิดการเลือกเส้นทางที่สั้นที่สุด (shortest path) ก็มีมากขึ้นด้วยเช่นกัน

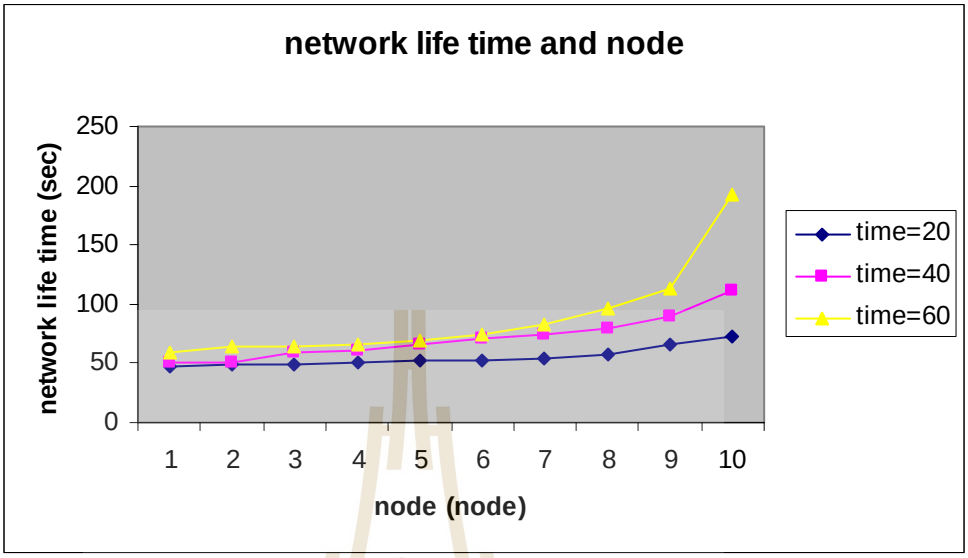
9. กราฟแสดงความสัมพันธ์ระหว่างความเร็วและความสำเร็จสะสม (velocity and accumuted of success path)



รูปที่ 17 กราฟแสดงความสัมพันธ์ระหว่างความเร็วและความสำเร็จสะสม (velocity and accumuted of success path)

ทำการทดสอบโดยพิจารณาความเร็วในการเคลื่อนที่ของโหนดกับ Accumuted of success path โดยความเร็วที่ทำการทดสอบมี 3 ระดับ คือ time=1 msec, 21 msec และ 41 msec กล่าวคือ โปรแกรมจะทำการเคลื่อนที่โหนดทุกๆ 1 msec, 21 msec และ 41 msec ดังนั้น Velocity ของโหนดที่เร็วที่สุดเรียงจากมากไปน้อยในที่นี้คือ time=1 msec, time=21 msec และtime= 41 msec ตามลำดับ จากการทดสอบจะเห็นได้ว่าการที่โหนดมีการเคลื่อนที่เร็วจะมีการเชื่อมต่อกันระหว่างโหนดบ่อยครั้งทำให้พลังงานหมดเร็ว โอกาสที่จะเกิด accumuted of success path น้อย ส่วนการที่โหนดเคลื่อนที่ช้าจะมีการเชื่อมต่อกันระหว่างโหนดน้อยครั้งจึงสูญเสียพลังงานในการเชื่อมต่อน้อยและยังส่งผลให้อายุขัยของเครือข่ายยาวขึ้นด้วย ทำให้ค่า accumuted of success path สูง

10. กราฟแสดงความสัมพันธ์ระหว่างอายุขัยของเครือข่ายและจำนวนโหนด (network life time and node)



รูปที่ 18 กราฟแสดงความสัมพันธ์ระหว่างอายุขัยของเครือข่ายและจำนวนโหนด (network life time and node)

การทดสอบนี้ทำการจับเวลาที่แต่ละโหนดตายเมื่อโหนดมีความเร็วในการเคลื่อนที่ 20 msec, 40 msec และ 60 msec จากกราฟจะเห็นได้ว่าถ้าโหนดมีการเคลื่อนที่เร็ว (20 msec) ทำให้อายุขัยของเครือข่ายน้อย แต่ถ้าหากโหนดมีการเคลื่อนที่ช้าลง (40 msec และ 60 msec) ทำให้อายุขัยของเครือข่ายยาวขึ้น ที่เป็นเช่นนี้เนื่องจากว่าการที่โหนดเคลื่อนที่เร็วทำให้เกิดการเชื่อมต่อระหว่างโหนดบ่อยครั้งจึงสูญเสียพลังงานมากว่าการที่โหนดเคลื่อนที่ช้าๆ ซึ่งก็จะมีการเชื่อมต่อระหว่างโหนดน้อยครั้งลง แต่อย่างไรก็ตามในระบบเครือข่ายจริงนั้นแต่ละโหนดไม่ได้ทำการเชื่อมต่อกันทุกครั้งที่มีรัศมีของสัญญาณส่งถึงกัน

บทที่ 4
สรุปผลและข้อดีข้อเสียของเครือข่าย โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network)

จากการทดลองเพื่อดูผลกระทบต่าง ๆ ซึ่งอาจส่งผลกระทบต่อระบบเครือข่าย โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) พบว่ามีปัจจัยหลายอย่างที่ส่งผลต่อการเชื่อมต่อไม่ว่าจะเป็นเวลา เพราะว่าเมื่อเวลาผ่านไปการสูญเสียพลังงานก็ยังคงมีอยู่ซึ่งประกอบไปด้วย 2 ส่วน

1. การสูญเสียพลังงานจากการเชื่อมต่อ
2. การสูญเสียพลังงานตามเวลา

ทั้ง 2 ส่วนนี้มีผลทำให้เครือข่ายมีอายุการใช้งานที่น้อยลงและการเชื่อมต่อที่น้อยลงตามไปด้วยแต่ถ้าเรามีโหนดหรือผู้ใช้งาน (USER) ของผู้ใช้งานเพิ่มขึ้นโอกาสในการเชื่อมต่อก็ย่อมมีมากขึ้นเพราะมีตัวเลือกในเครือข่ายให้เลือกมากกว่า ยิ่งถ้ารัศมีการส้งสัญญาณมากโอกาสในการเชื่อมต่อก็ยิงมากขึ้น

ข้อดีของ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) ได้แก่

โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) มีข้อได้เปรียบกว่าโครงข่ายไร้สายแบบอื่น ๆ ที่มีอยู่เดิมหลายประการยกตัวอย่างเช่น โครงข่ายสามารถสร้างขึ้นได้อย่างรวดเร็ว ไม่ยุ่งยากซับซ้อน ลดการก่อสร้างในส่วนของโครงข่ายที่ต้องติดตั้งอยู่กับที่ทำให้งบประมาณการสร้างโครงข่ายลดลงและยังทำให้การใช้พลังงานลดลงด้วย ดังนั้น โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) สามารถสร้างขึ้นได้ทันที โดยไม่ต้องมีโครงสร้างของสถานีฐานและไม่ต้องมีผู้ดูแลระบบโครงข่าย

โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) สามารถนำไปประยุกต์ใช้กับระบบต่าง ๆ ได้อย่างเหมาะสม ได้แก่ ใช้ติดต่อส่วนบุคคล ซึ่งอุปกรณ์ที่ใช้ติดต่อ เช่น เครื่องโทรศัพท์มือถือ คอมพิวเตอร์พกพา หรือ เครื่องพีดีเอ (PDA) ใช้ติดต่อภายในกลุ่ม เช่น สร้างโครงข่ายติดต่อสื่อสารภายในงานแสดงนิทรรศการ งานสัมนาต่าง ๆ ใช้ในงานด้านการทหาร ใช้ในกรณีฉุกเฉินต่าง ๆ เช่น กรณีที่เกิดภัยพิบัติ หรือใช้เป็นโครงข่ายสื่อสารภายในเมือง เป็นต้น

ข้อเสียของ โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) ได้แก่

1. โทโพโลยีแบบพลวัตร (dynamic Topology) โหนดต่าง ๆเคลื่อนที่ได้และสามารถติดต่อกับโหนดใด ๆก็ได้ในโครงข่าย ดังนั้น การเชื่อมโยงโหนดต่าง ๆในโครงข่ายไม่ถาวร มีการเปลี่ยนแปลงไปตามเวลา ขึ้นอยู่กับตำแหน่งของโหนดต่าง ๆ ณ เวลาขณะใดขณะหนึ่งซึ่งทำให้การแก้ปัญหาเรื่องการสื่อสารที่ขาดการติดต่อไปเมื่อโหนดมีการเคลื่อนที่

2. ข้อจำกัดด้านพลังงาน พลังงานที่ใช้ในโหนดเคลื่อนที่นั้นก็คือแบตเตอรี่ ซึ่งเป็นทรัพยากรที่จำกัด กรณีพิจารณาโหนด ที่ไม่มีการชาร์ทพลังงานเข้าไปใหม่ ดังนั้น การออกแบบระบบโครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) จะต้องเกี่ยวข้องกับเทคโนโลยีด้านการอนุรักษ์พลังงานด้วย

3. ข้อจำกัดการใช้แถบความถี่ ความจุในการส่งสัญญาณข้อมูลทางการเชื่อมโยงไร้สายมีน้อยกว่าการเชื่อมโยงโดยใช้สายเป็นอย่างมาก และการเชื่อมโยงไร้สายยังมีปัญหาในเรื่องของสัญญาณรบกวน ทำให้ข้อมูลด้านรับประกันคุณภาพลดลง ซึ่งมีอัตราบิตผิดพลาด (bit error rate) อยู่ในช่วง 10^{-4}-10^{-5}

4. ข้อจำกัดทางด้านการรักษาความปลอดภัย โครงข่ายที่โหนดมีการเคลื่อนที่และมีโทโพโลยีแบบพลวัตร มีความเสี่ยงด้านความปลอดภัยของข้อมูลมากกว่าโครงข่ายแบบถาวร เนื่องจากอุปกรณ์สื่อสารที่โหนดหรือของผู้ใช้อาจถูกขโมยได้ หรือสัญญาณข้อมูลอาจถูกส่งไปในการเชื่อมโยงไร้สายที่ไม่ปลอดภัย เป็นต้น ดังนั้นจะต้องมีการป้องกันปัญหาในเรื่องการลักลอบหรือดักสัญญาณข้อมูล และปัญหาด้านความปลอดภัยของโครงข่าย

5. โครงข่ายอิสระ คือ ไม่มีการควบคุมสั่งการ หรือการจัดการกับโครงข่ายที่เป็นส่วนกลาง โครงข่ายเคลื่อนที่แบบแอดฮอค (mobile ad hoc network) สามารถดำเนินการด้วยตนเองได้ ดังนั้นการออกแบบระบบจะต้องมีความซับซ้อนมากขึ้นด้วย

ประวัติผู้เขียน

นายไชยรัตน์ จันทร์บุญเรือง เกิดเมื่อวันที่ 10 เมษายน พ.ศ. 2527 ภูมิลำเนาอยู่ที่ ตำบลห้วยหิน อำเภอหนองหงส์ จังหวัดบุรีรัมย์ สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนบุรีรัมย์พิทยาคม อำเภอเมือง จังหวัดบุรีรัมย์ เมื่อปี พ.ศ. 2546 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นายณัฐวุธ คณฑา เกิดเมื่อวันที่ 4 มีนาคม พ.ศ. 2528 ภูมิลำเนาอยู่ที่ ตำบลห้วยแห้ง อำเภอบ้านไร่ จังหวัดอุทัยธานี สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนบ้านไร่วิทยา อำเภอบ้านไร่ จังหวัดอุทัยธานี เมื่อปี พ.ศ. 2546 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

บรรณานุกรม

นิรุธ อำนวยศิลป์.(2545).**เขียนเกมส์อย่างมืออาชีพด้วย Visual C++ และ Directx.** อินโฟรเพลส.นนทบุรี.