การออกแบบข้อกำหนดของระบบสื่อสารโดยใช้ไฟไนท์สเตทแมชชีน

DESIGN ON THE SPECIFICATION OF COMMUNICATION SYSTEM BY USING FINITE STATE MACHINE

ภูริชญ์ เชาว์ชนะพันธุ์

PURICH CHAOCHANAPHUN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2548

ISBN 974-19-1756-4

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การออกแบบข้อกำหนดของระบบสื่อสารโดยใช้ไฟไนท์สเตทแมชชีน

DESIGN ON THE SPECIFICATION OF COMMUNICATION SYSTEM BY USING FINITE STATE MACHINE

T060875

ภูริชญ์ เชาว์ชนพันธุ์
PURICH CHAOCHANAPHUN

เลขหมู่.........
เลขทะเบียน 60875
วัน,เดือน,ปี - 6 ก.ค. 2549
b..........
i..........

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2548

ISBN 974-15-1756-4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DESIGN ON THE SPECIFICATION OF COMMUNICATION SYSTEM BY USING FINITE STATE MACHINE

PURICH CHAOCHANAPHUN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN ELECTRONICS ENGINEERING
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2005

ISBN 974-15-1756-4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COPYRIGHT 2005

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อวิทยานิพนธ์	การออกแบบข้อกำหนดของระบบสื่อสารโดยใช้ไฟไนท์สเตทแมชชีน
นักศึกษา	นายภูริชญ์ เชาว์ชนพันธุ์
รหัสนักศึกษา	43061319
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมอิเล็กทรอนิกส์
พ.ศ.	2548
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ.ดร.รัตติกร วรากูลศิริพันธุ์

บทคัดย่อ

งานวิจัยนี้ได้นำเสนอการประยุกต์ใช้ไฟไนท์สเตทแมชชีนเป็นแบบจำลองในการออกแบบกลไกการทำงานของระบบสื่อสารที่เป็นไปตามข้อกำหนด ข้อกำหนดสำคัญของระบบสื่อสารในปัจจุบันที่งานวิจัยนี้นำมาพิจารณาได้แก่ ข้อจำกัดทางเวลาและข้อจำกัดทางภูมิศาสตร์ ด้วยการเพิ่มนิยามของข้อจำกัดเหล่านี้ในไฟไนท์สเตทแมชชีน ทำให้สามารถนำมาเป็นแบบจำลองในการออกแบบระบบสื่อสารที่มีการทำงานขึ้นกับเวลา มีการกำหนดพื้นที่และขอบเขตของการสื่อสารได้เป็นย่างดี และในทางกลับกันวิธีการนี้ยังสามารถใช้เป็นแบบจำลองในการทดสอบกลไกการทำงานของระบบสื่อสารที่ออกแบบนั้นๆว่าสามารถทำงานได้ตามข้อกำหนดหรือไม่

นอกจากนี้งานวิจัยนี้ได้พัฒนาซอฟท์แวร์ในการสร้างแบบจำลองของไฟไนท์สเตทแมชชีนที่มีข้อจำกัดทางเวลาและภูมิศาสตร์ เพื่อใช้เป็นเครื่องมือในการออกแบบกลไกการทำงานของระบบสื่อสารดังกล่าวข้างต้น ทำให้สะดวกต่อการจำลองการทำงานของระบบสื่อสาร และทดสอบกลไกการทำงานให้เป็นไปตามข้อกำหนดที่ต้องการ เป็นประโยชน์ต่อการตัดสินใจก่อนการนำไปใช้งานจริง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title	Design on the Specification of Communication System by Using Finite State Machine
Student	Mr. Purich Chaochanaphun
Student ID.	43061319
Degree	Master of Engineering
Programme	Electronics Engineering
Year	2005
Thesis Advisor	Assoc. Prof. Dr. Ruttikorn Varakulsiripunth

ABSTRACT

This research presented the application of finite state machine as a model for designing the operation mechanism of a communication system in according to its specification. The important specification items of a present communication system that have been considered in this research are the time and geographical constraints. By implementing the definition of these constraints into finite state machine, it can be used appropriately to be the model for designing the communication system that has to be operated with time, location and area of communication. And on the other hands, this methodology can be used as a simulation model to verify that operation mechanism of designed communication system will work properly or not.

Moreover, in this research we have also developed the software for constructing the finite state machine with time and geographical constraints. This software can be used as a tool to design the operation mechanism of above mentioned communication system. Therefore, it is convenient to model the operation of communication system and verify its operation mechanism to execute properly with required specification. This is useful for decision making before real implementation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้เป็นอย่างดี เนื่องด้วยการสนับสนุนของบุคคลหลายฝ่าย ผู้ทำวิจัย

ขอขอบพระคุณ รศ.ดร.รัตติกร วรากูลศิริพันธุ์ ซึ่งเป็นอาจารย์ที่ปรึกษา ที่กรุณาช่วยให้แนวความคิด ให้คำปรึกษาในการทำวิจัย ตลอดจนการวิเคราะห์เพื่อแก้ปัญหาต่างๆ อันเป็นประโยชน์ต่องานวิจัย และได้กรุณาให้การสนับสนุนโอกาสในการศึกษาวิจัยในระดับต่อไป

ขอขอบพระคุณ Prof. Yasushi Kato และ Prof. Kaoru Takahashi ที่ได้ให้โอกาสและความรู้ในการศึกษางานวิจัยต่อจากท่าน

ขอขอบพระคุณ ภาควิชาวิศวกรรมอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และ โครงการสำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ (ReCCIT) และ องค์การความร่วมมือระหว่างประเทศของญี่ปุ่น (JICA) ที่ได้ให้การสนับสนุนเครื่องมือและอุปกรณ์ในการทำวิจัยนี้ในห้องปฏิบัติการ โครงข่ายการสื่อสาร (Communication Network Laboratory)

ขอขอบคุณ คุณนพดล มณีรัตน์ ที่ช่วยให้ความรู้ แนะนำแนวทางในการทำวิจัย การเขียนและการนำเสนอบทความ การหาข้อมูลเกี่ยวกับงานวิจัย และให้คำปรึกษาที่ดีตลอดมา

ขอขอบคุณพี่ๆ เพื่อนๆ และน้องๆ ที่ช่วยให้คำปรึกษาและแก้ปัญหาต่างๆในงานวิจัย รวมทั้งทุกๆ คนที่มิได้เอ่ยนามในที่นี้

และสุดท้ายนี้ขอกราบขอบพระคุณคุณพ่อ คุณแม่ ผู้ซึ่งคอยอบรมเลี้ยงดู ให้โอกาสในการศึกษา และสนับสนุนในทุกๆ ด้าน รวมถึงช่วยเป็นกำลังใจให้เสมอมาและตลอดไป อีกทั้งช่วยผลักดันให้วิทยานิพนธ์ฉบับนี้สำเร็จได้อย่างดีเยี่ยม

ภูริชญ์ เชาว์ชนพันธุ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

รูปที่ หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในการออกแบบระบบการสื่อสารผู้ออกแบบต้องทราบความต้องการของระบบการสื่อสารที่จะออกแบบเสียก่อน ความต้องการของระบบการสื่อสารที่ได้มาจำเป็นต้องถูกเปลี่ยนให้เป็นข้อกำหนดของการสื่อสาร(Communication specification) ที่มีรูปแบบกระชับและชัดเจน เพื่อให้เกิดความถูกต้องในการออกแบบและตรวจสอบ ตรงต่อความต้องการของระบบการสื่อสารนั้น

เนื่องจากระบบการสื่อสารเป็นระบบที่ประกอบด้วยกระบวนการเชื่อมต่อถึงกัน ดังนั้นการออกแบบข้อกำหนดของระบบการสื่อสารจึงจำเป็นต้องพิจารณาถึงการทำงานร่วมกัน และการสื่อสารระหว่างกันภายในระบบ ที่ผ่านมาได้มีการนำเสนอวิธีการระบุข้อกำหนดอยู่หลากหลายรูปแบบ อาทิเช่น CCS[1] และ CSP[2] ได้จำลองระบบการสื่อสารเป็นขบวนการและแสดงข้อกำหนดต่างๆ ในรูปแบบของพีชคณิตหรือแบบจำลองคณิตศาสตร์ และต่อมาได้มีการนำเสนอข้อกำหนดการเคลื่อนที่และเวลาเพิ่มเข้าไป[3, 4, 5] แต่การออกแบบระบบสื่อสาร โดยใช้แบบจำลองคณิตศาสตร์มีข้อเสียคือ เห็นภาพการทำงานภายในระบบได้ยาก และมีความยุ่งยากในการออกแบบ ด้วยเหตุนี้จึงได้มีการนำเสนอวิธีการระบุข้อกำหนดแบบอื่นๆ ซึ่งไม่ยึดติดกับรูปแบบทางคณิตศาสตร์มากนัก เช่น การนำสเตทแมชชีนมาเป็นแบบจำลองเพื่อช่วยในการออกแบบ[6, 7, 8] ทำให้สามารถเข้าใจระบบการสื่อสารที่ออกแบบได้ง่ายขึ้น และมีความใกล้เคียงต่อการใช้งานจริงมากขึ้น ที่ผ่านมาได้มีการนำเสนอข้อกำหนดของเวลา[11, 12, 13, 14, 15, 16] พื้นที่การให้บริการและการเคลื่อนที่[9, 10] มาพิจารณาในการออกแบบระบบสื่อสาร แต่ข้อกำหนดเหล่านี้จะถูกแยกพิจารณาอย่างเป็นอิสระต่อกันในการออกแบบระบบการสื่อสาร

ในงานวิจัยนี้จึงได้นำเสนอข้อกำหนดเหล่านี้ในรูปแบบของนิยามพร้อมทั้งนำไฟไนท์สเตทแมชชีนมาเป็นแบบจำลองพื้นฐานในการออกแบบ และได้พยายามนำข้อกำหนดต่างๆมาพิจารณาร่วมกัน เพื่อให้ตรงกับความต้องการหรือความสามารถของระบบการสื่อสารที่มีอยู่ อาทิเช่น สามารถทำงานขึ้นกับเวลา สามารถระบุพื้นที่การติดต่อหรือการให้บริการ และสามารถเคลื่อนที่ได้ นอกจากนี้งานวิจัยยังได้พัฒนาโปรแกรมช่วยตรวจสอบ และจำลองการทำงานของระบบสื่อสาร เพื่อช่วยเพิ่มความสะดวกในการออกแบบ โดยโปรแกรมสามารถตรวจสอบความถูกต้องของข้อกำหนดต่างๆของระบบการสื่อสาร แสดงการทำงานของระบบการสื่อสารในรูปแบบของแผนผังสถานะการทำงาน และจำลองการทำงานของระบบการสื่อสาร ซึ่งช่วยให้ผู้ออกแบบสามารถตรวจสอบพฤติกรรมภายในของระบบการสื่อสารได้อีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

ความมุ่งหมายและวัตถุประสงค์ของการศึกษาแบ่งเป็นข้อๆ ได้ดังนี้

1. เพื่อศึกษาและหาวิธีการออกแบบระบบการสื่อสารให้มีความถูกต้อง และตรงกับความต้องการในปัจจุบัน
2. นำเสนอนิยามวิธีการระบุข้อกำหนดแบบใหม่ในการระบุระบบการสื่อสาร โดยนำไฟไนท์สเตทแมชชีนมาใช้เป็นแบบจำลองพื้นฐานในการออกแบบ
3. เพื่อนำเสนอวิธีการออกแบบระบบการสื่อสารที่มีข้อจำกัดของระบบการสื่อสารแบบต่างๆ โดยในงานวิจัยนี้ได้นำเสนอวิธีการระบุข้อกำหนดของระบบการสื่อสารทั้งแบบมีข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดการเคลื่อนที่
4. พัฒนาโปรแกรมช่วยในการออกแบบ วิเคราะห์ และจำลองการทำงานของระบบการสื่อสาร

1.3 สมมุติฐานของการศึกษา

ในปัจจุบันนี้มีวิธีการออกแบบระบบการทำงานของระบบสื่อสารอยู่หลายวิธี แต่ยังไม่สามารถรองรับกับความต้องการของระบบสื่อสารได้ ดังนั้นในงานวิจัยนี้จึงได้นำเสนอวิธีการระบุข้อกำหนดของระบบการสื่อสารที่มีข้อจำกัดของระบบการสื่อสารแบบต่างๆ ข้อจำกัดในระบบการสื่อสารที่นำมาพิจารณาได้แก่ ข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดทางการเคลื่อนที่ โดยการเพิ่มข้อจำกัดเหล่านี้เข้าไปในข้อกำหนดของระบบการสื่อสาร ทำให้ผู้ออกแบบสามารถออกแบบระบบการสื่อสารที่มีความหลากหลายมากขึ้น และโดยอาศัยไฟไนท์สเตทแมชชีนเป็นแบบจำลองพื้นฐานในการออกแบบ ทำให้ผู้ออกแบบสามารถออกแบบและทำความเข้าใจกับโครงสร้างของระบบการสื่อสารได้ง่ายกว่าแบบจำลองทางคณิตศาสตร์ นอกจากนี้ในงานวิจัยนี้ยังได้พัฒนาโปรแกรมช่วยตรวจสอบและจำลองการทำงานของระบบการสื่อสาร ซึ่งทำให้ผู้ออกแบบสามารถออกแบบ ตรวจสอบ และจำลองการทำงานของระบบการสื่อสารได้อย่างสะดวกอีกด้วย

1.4 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย

ในงานวิจัยนี้ได้นำเสนอการออกแบบและวิเคราะห์ระบบการสื่อสาร โดยนำความต้องการของระบบการสื่อสารที่ต้องการมาเปลี่ยนให้อยู่ในรูปแบบของข้อกำหนดที่ชัดเจน จากนั้นจึงนำข้อกำหนดต่างๆ ที่ได้มาทำการตรวจสอบความถูกต้องและจำลองการทำงานต่อไป สำหรับวิธีการระบุข้อกำหนดในงานวิจัยนี้ได้ถูกนำเสนอไว้ในรูปแบบของนิยาม โดยนำไฟไนท์สเตทแมชชีนมาประยุกต์ใช้เป็นแบบจำลองพื้นฐานของระบบการสื่อสาร ซึ่งวิธีการดังกล่าวได้ทำการนิยามข้อกำหนดให้ตรงกับความต้องการของระบบการสื่อสาร ด้วยการนิยามข้อจำกัดของระบบการสื่อสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้แก่ข้อกำหนด เช่น การเพิ่มข้อจำกัดทางเวลาทำให้สามารถออกแบบระบบการสื่อสารที่ขึ้นกับเวลาได้ การเพิ่มข้อจำกัดทางภูมิศาสตร์ทำให้สามารถออกแบบระบบการสื่อสารที่สามารถกำหนดพื้นที่การให้บริการภายในได้ และการเพิ่มข้อจำกัดการเคลื่อนที่ทำให้สามารถออกแบบระบบที่มีคุณสมบัติการเคลื่อนที่ได้

1.5 ขอบเขตการวิจัย

งานวิจัยนี้ได้นำเสนอการออกแบบระบบการสื่อสารด้วยวิธีการระบุข้อกำหนด โดยนำไฟไนท์สเตทแมชชีนมาประยุกต์ใช้เป็นแบบจำลองพื้นฐานของระบบการสื่อสาร ทั้งนี้ข้อกำหนดได้ถูกนิยามข้อจำกัดทางระบบการสื่อสารเข้าไว้ด้วย อันได้แก่ ข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดทางการเคลื่อนที่ จึงทำให้สามารถนำวิธีการนี้ไปใช้ออกแบบกับระบบการสื่อสารที่ทำงานขึ้นกับเวลา พื้นที่การให้บริการ และการเคลื่อนที่ได้ นอกจากนี้งานวิจัยนี้ยังได้พัฒนาโปรแกรมช่วยออกแบบและจำลองการทำงานของระบบสื่อสาร โดยอาศัยวิธีการออกแบบดังกล่าวเพื่อความสะดวกในการออกแบบและพัฒนาระบบอีกด้วย

1.6 ขั้นตอนการศึกษา

ขั้นตอนการศึกษาได้แสดงเป็นลำดับดังนี้

1. ศึกษาวิธีการออกแบบและวิธีการระบุข้อกำหนดแบบต่างๆที่มีอยู่
2. เลือกศึกษาวิธีการที่เหมาะสมสำหรับการออกแบบระบบการสื่อสาร ในงานวิจัยนี้ได้ใช้วิธีการระบุข้อกำหนด โดยอาศัยไฟไนท์สเตทแมชชีนเป็นแบบจำลองพื้นฐานในการออกแบบเนื่องจากเป็นแบบจำลองที่ง่ายต่อการทำความเข้าใจ ออกแบบ และตรวจสอบแก้ไข
3. เสนอวิธีการระบุข้อกำหนดต่างๆ ในรูปแบบของนิยามข้อกำหนด ตามลักษณะของระบบการสื่อสาร
4. นิยามข้อจำกัดต่างๆ ของระบบการสื่อสารเพื่อเป็นข้อกำหนด ทำให้ข้อกำหนดที่ได้สามารถถูกนำมาใช้ออกแบบระบบการสื่อสารที่มีข้อจำกัดได้ ข้อจำกัดที่นำมาพิจารณาในงานวิจัยนี้ ได้แก่ ข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดการเคลื่อนที่
5. พัฒนาโปรแกรมช่วยออกแบบและจำลองการทำงานของระบบโดยอาศัยวิธีการออกแบบที่นำเสนอ เพื่อให้สามารถนำมาใช้ออกแบบ ตรวจสอบความถูกต้อง วิเคราะห์ และจำลองการทำงานของระบบการสื่อสารได้
6. ทดสอบโปรแกรมที่พัฒนาขึ้นมากับระบบการสื่อสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

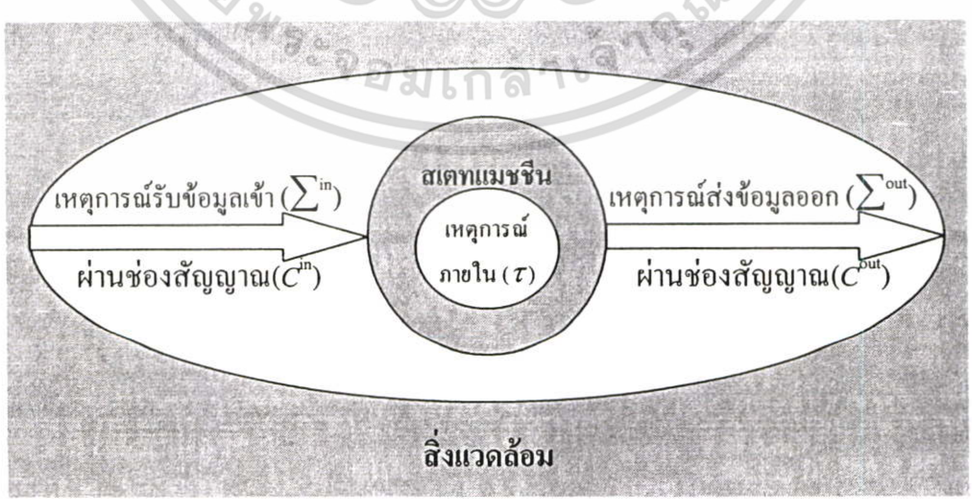
บทที่ 2
ไฟไนท์สเตทแมชชีนพื้นฐาน

2.1 บทนำ

ในบทนี้จะนิยามแบบจำลอง และการกำหนดรูปแบบของไฟไนท์สเตทแมชชีนพื้นฐาน โดยไฟไนท์สเตทแมชชีนพื้นฐานหมายถึง ไฟไนท์สเตทแมชชีนที่ปราศจากข้อจำกัดทางด้านการสื่อสารใดใด และต่อไปจะขอเรียกโดยย่อว่า "สเตทแมชชีน" ซึ่งแบบจำลองสเตทแมชชีนที่ได้นี้จะถูกนำมาใช้เป็นแบบจำลองในการออกแบบกลไกการทำงานภายในของระบบการสื่อสารต่อไป โดยระบบการสื่อสารประกอบไปด้วยสเตทแมชชีนหลายๆ ตัวทำงานร่วมกันอยู่

2.2 แบบจำลอง

ในหัวข้อนี้จะนิยามแบบจำลองสเตทแมชชีน ให้มีความสามารถสื่อสารกับสิ่งแวดล้อมหรือสเตทแมชชีนตัวอื่นๆได้ ในรูปแบบการรับข้อมูลเข้าและส่งข้อมูลออกผ่านทางช่องสัญญาณการสื่อสาร กำหนดให้เซตของช่องสัญญาณการสื่อสารที่รองรับเซตข้อมูลเข้า(I) เป็น C^{in} และเซตของช่องสัญญาณการสื่อสารที่รองรับเซตข้อมูลออก(O) เป็น C^{out} ในนิยามต่อจากนี้การรับส่งข้อมูลของสเตทแมชชีนผ่านช่องสัญญาณการสื่อสารจะถูกกำหนดให้เป็น *เหตุการณ์(event)* ซึ่งจะประกอบไปด้วย เหตุการณ์รับข้อมูลจากสิ่งแวดล้อม เหตุการณ์ส่งข้อมูลออกสู่สิ่งแวดล้อม และเหตุการณ์ภายใน ดังแสดงไว้ในรูปที่ 2.1 โดยเหตุการณ์เหล่านี้มีผลทำให้เกิดการเปลี่ยนแปลงขึ้นต่อสเตทแมชชีน



รูปที่ 2.1 แผนภาพแสดงเหตุการณ์รับ-ส่งข้อมูลกับสิ่งแวดล้อมและเหตุการณ์ภายใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นิยามที่ 2.1 สเตทแมชชีน M ประกอบด้วย 5 เซตหลัก คือ

$$M = \langle Q, V, \sum, T, \theta \rangle$$

- Q เป็นเซตจำกัดของสถานะควบคุม
- V เป็นเซตจำกัดของตัวแปร
- $\sum$ เป็นเซตของเหตุการณ์ ประกอบด้วย 3 เซตย่อย คือ
  - เซตเหตุการณ์รับข้อมูลเข้าจากสิ่งแวดล้อม, $\sum^{in}$ ($\sum^{in} \subseteq C^{in} \times I$),
  - เซตเหตุการณ์ส่งข้อมูลออกสู่สิ่งแวดล้อม, $\sum^{out}$ ($\sum^{out} \subseteq C^{out} \times O$) และ
  - เซตเหตุการณ์ที่เกิดขึ้นภายในสเตทแมชชีน M, $\{\tau\}$
- $T \subseteq (Q \times \mathbf{V}) \times \sum \times (Q \times \mathbf{V})$ เป็นเซตของการเปลี่ยนสถานะ
- $\theta \subseteq Q \times \mathbf{V}$ เป็นเซตของสถานะเริ่มต้น

กำหนดให้ $s \in Q \times \mathbf{V}$ เป็นสถานะของสเตทแมชชีน M โดยที่ V เป็นเซตของตัวแปร และ $\mathbf{V}$ เป็นเซตค่าของเซตตัวแปร V แต่เนื่องจากบางกรณีโดเมนของตัวแปรอาจเป็นเซตไม่จำกัด ดังนั้น จำนวนสถานะของ M อาจมีได้ไม่จำกัดด้วย การเปลี่ยนสถานะสามารถแสดงด้วย $s \xrightarrow{\sigma} s'$ โดยที่ $(s, \sigma, s') \in T$

ข้อกำหนดการแสดงที่ 1 แผนผังการเปลี่ยนสถานะของสเตทแมชชีนให้แสดงดังนี้

สถานะปัจจุบัน $\xrightarrow{guard \rightarrow event}$ สถานะถัดไป

guard คือเงื่อนไขของการเกิดเหตุการณ์(*event*) ซึ่งอ้างอิงถึงตัวแปรที่เกี่ยวข้องกับการเปลี่ยนสถานะ ถ้าไม่มีความจำเป็นหรือเงื่อนไขเป็นจริงตลอดสามารถเว้นไว้ได้

<u>เหตุการณ์รับข้อมูลเข้า</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ?ชื่อตัวแปร หรือ “ค่า”

เพื่อแสดงว่า สเตทแมชชีนรับข้อมูลจาก *ชื่อตัวแปร* หรือ “ค่า” ผ่านทางช่องสัญญาณ *ชื่อช่องสัญญาณ*

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>เหตุการณ์รับข้อมูลเข้าแบบข้อความ</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ?ชื่อข้อความ(ตัวแปร, … , ตัวแปร)

เพื่อแสดงว่า สเตทแมชชีนรับข้อมูลในรูปแบบข้อความจาก *ชื่อข้อความ* ผ่านทางช่องสัญญาณ *ชื่อช่องสัญญาณ* ในที่นี้ตัวแปรอาจเป็น “ค่า” ก็ได้

<u>เหตุการณ์ส่งข้อมูลออก</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ!ชื่อตัวแปร หรือ “ค่า”

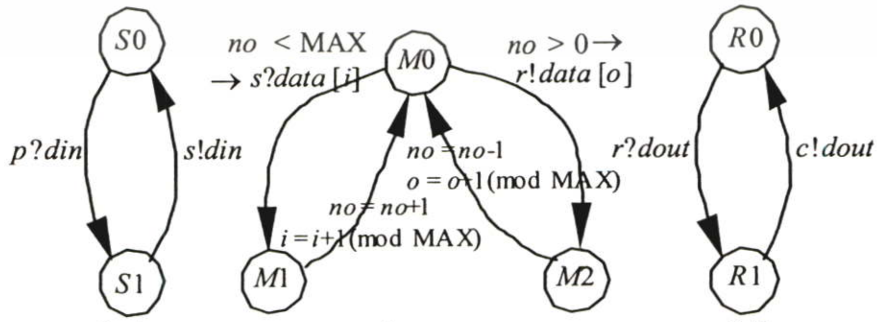
เพื่อแสดงว่า สเตทแมชชีนส่งข้อมูลจาก *ชื่อตัวแปร* หรือ “ค่า” ผ่านทางช่องสัญญาณ *ชื่อช่องสัญญาณ*

<u>เหตุการณ์ส่งข้อมูลออกแบบโครงสร้าง</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ!ชื่อข้อความ(ตัวแปร, … , ตัวแปร)

เพื่อแสดงว่า สเตทแมชชีนส่งข้อมูลในรูปแบบข้อความจาก *ชื่อข้อความ* ผ่านทางช่องสัญญาณ *ชื่อช่องสัญญาณ* ในที่นี้ตัวแปรอาจเป็น “ค่า” ก็ได้

สำหรับเหตุการณ์ภายในสเตทแมชชีน ถ้ามีกระบวนการเปลี่ยนค่าของตัวแปรควรแสดงกระบวนการนั้นกำกับไว้ด้วย



รูปที่ 2.2 ตัวอย่างระบบการรับ-ส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างที่ 2.1 รูปที่ 2.2 แสดงตัวอย่างการกำหนดรูปแบบการทำงานของระบบการสื่อสารด้วยแบบจำลองสเตทแมชชีน ซึ่งระบบนี้ประกอบไปด้วยสเตทแมชชีน 3 ตัว คือ **ตัวส่ง** ส่งข้อมูลให้แก่ **ตัวรับ** โดยผ่าน **ตัวกลาง** ตัวส่งประกอบด้วยสถานะควบคุม *S*0 และ *S*1 ทำงานโดยรับข้อมูลจากสิ่งแวดล้อมแล้วเก็บลงตัวแปรชื่อ *din* โดยผ่านทางช่องสัญญาณ *p* และส่งข้อมูลต่อให้กับตัวกลางผ่านทางช่องสัญญาณ *s* โดย เริ่มทำงานที่สถานะควบคุม *S*0 ตัวกลางประกอบด้วยสถานะควบคุม *M*0, *M*1, และ *M*2 เริ่มทำงานที่สถานะควบคุม *M*0 ประกอบด้วยตัวแปร *no*, *i*, *o*, และ *data* ซึ่ง *data* เป็นตัวแปรสำหรับเก็บข้อมูลหรือตัวเก็บข้อมูล, *no* แสดงจำนวนของข้อมูลในตัวเก็บข้อมูล, *i* แสดงตำแหน่งข้อมูลเข้าในตัวเก็บข้อมูล, และ *o* แสดงตำแหน่งข้อมูลออกในตัวเก็บข้อมูลโดยค่าเริ่มต้นของตัวแปรทั้ง 3 ถูกกำหนดให้เป็น 0, MAX เป็นค่าคงที่เพื่อระบุจำนวนข้อมูลที่มากที่สุดที่ตัวเก็บข้อมูลสามารถเก็บไว้ได้ ข้อมูลสามารถถูกเก็บลงตัวเก็บข้อมูลโดยผ่านทางช่องสัญญาณ *s* และถ้าในตัวเก็บข้อมูลมีข้อมูลอยู่ ตัวกลางก็สามารถส่งข้อมูลไปให้ตัวรับได้โดยผ่านทางช่องสัญญาณ *r* ซึ่งในระหว่างที่การรับส่งข้อมูลกระทำอยู่นั้น ค่าของตัวแปรในระบบก็จะถูกเปลี่ยนแปลงโดยเหตุการณ์ภายในไปด้วย ส่วนตัวรับประกอบด้วยสถานะควบคุม *R0* และ *R1* เริ่มทำงานที่สถานะควบคุม *R0* ทำงานโดยตัวรับรับข้อมูลผ่านทางช่องสัญญาณ *r* เก็บลงตัวแปร *dout* และส่งข้อมูลนั้นออกจากระบบสู่สิ่งแวดล้อมภายนอกระบบ โดยผ่านทางช่องสัญญาณ *c*

จากรูปที่ 2.2 เมื่อกำหนดให้

$\tau_1 = \{ no = no + 1; i = i + 1(\bmod \text{MAX}) \}$

$\tau_2 = \{ no = no - 1; o = o + 1(\bmod \text{MAX}) \}$

$\mathbf{v}$ = เซตค่าของเซตตัวแปร V ที่สถานะใดใด

สเตทแมชชีนตัวกลางประกอบไปด้วย 5 เซต ตามนิยามที่ 2.1 ดังนี้

$Q = \{ \text{M0, M1, M2} \}$

$V = \{ no, i, o, data \}$

$\sum = \{ s?data[i], r!data[o], \tau_1, \tau_2 \}$

โดยที่

$\sum^{\text{in}} = \{ s?data[i] \}$

$\sum^{\text{out}} = \{ r!data[o] \}$

$\tau = \{ \tau_1, \tau_2 \}$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$T = \{ ((M0, \mathbf{v})\ s?data[i]\ (M1, \mathbf{v}))\ ,\ ((M0, \mathbf{v})\ r!data[o]\ (M2, \mathbf{v})),$$
$$((M1, \mathbf{v})\ \tau_1\ (M0, \mathbf{v})), ((M2, \mathbf{v})\ \tau_2\ (M0, \mathbf{v})) \}$$
$$\theta = \{ (\mathrm{M0}, \mathbf{v}) \}$$

นิยามที่ 2.2 กำหนดให้ลำดับการทำงานของสเตทแมชชีน M เป็นลำดับของสถานะ และเหตุการณ์

$$\alpha = S_0 \sigma_0 S_1 \sigma_1 S_2 \sigma_2 S_3 \ldots$$

ซึ่งเป็นไปตามเงื่อนไขต่อไปนี้

1. $S_0 \in \theta$
2. สำหรับทุก $i \geq 0,\ S_i \xrightarrow{\sigma_i} S_{i+1}$

เซตลำดับการทำงานทั้งหมดของ M เขียนแทนด้วย $Comp(M)$

จากตัวอย่างที่ 2.1 สามารถแสดง สถานะ เหตุการณ์รับข้อมูลเข้า และเหตุการณ์ส่งข้อมูลออกได้ดังนี้

(สถานะควบคุม, $\{no, i, o, data\}$)

ชื่อช่องสัญญาณ?ข้อมูล

ชื่อช่องสัญญาณ!ข้อมูล

ถ้า MAX = 2 แล้ว ลำดับการทำงานที่เป็นไปได้ของตัวกลาง คือ

$\alpha =$ $(M0, \{no = 0, i = 0, o = 0, data[] = null\})\ s?$"d1"

$(M1, \{no = 0, i = 0, o = 0, data[] =$ "d1"$\})\ \tau$

$(M0, \{no = 1, i = 1, o = 0, data[] =$ "d1"$\})\ s?$"d2"

$(M1, \{no = 1, i = 1, o = 0, data[] =$ "d1"."d2"$\})\ \tau$

$(M0, \{no = 2, i = 0, o = 0, data[] =$ "d1"."d2"$\})\ r!$"d1"

$(M2, \{no = 2, i = 0, o = 0, data[] =$ "d1"."d2"$\})\ \tau$

$(M0, \{no = 1, i = 0, o = 1, data[] =$ "d1"."d2"$\})\ s?$"d3"

$(M0, \{no = 1, i = 0, o = 1, data[] =$ "d3". "d2"$\})$ …..

โดยที่ "d1", "d2", … **คือ ค่าข้อมูล**

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นิยามที่ 2.3 กำหนดให้ $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$ เป็นลำดับการทำงานของสเตทแมชชีน M ลำดับนี้ถ้าตัดสถานะและเหตุการณ์ภายในออกทั้งหมดแล้วจะเรียกว่า เส้นทางการเปลี่ยนแปลงของสเตทแมชชีน M

เส้นทางการเปลี่ยนแปลงเป็นเหมือนการสังเกตการทำงานของสเตทแมชชีนจากภายนอกหรือสิ่งแวดล้อม เนื่องจากเส้นทางการเปลี่ยนแปลงทั้งหมดของสเตทแมชชีน M จะแสดงเฉพาะเหตุการณ์ที่กระทำกับสิ่งแวดล้อมเท่านั้น เซตเส้นทางการเปลี่ยนแปลงทั้งหมดของสเตทแมชชีน M เขียนแทนด้วย *Trace*(M)

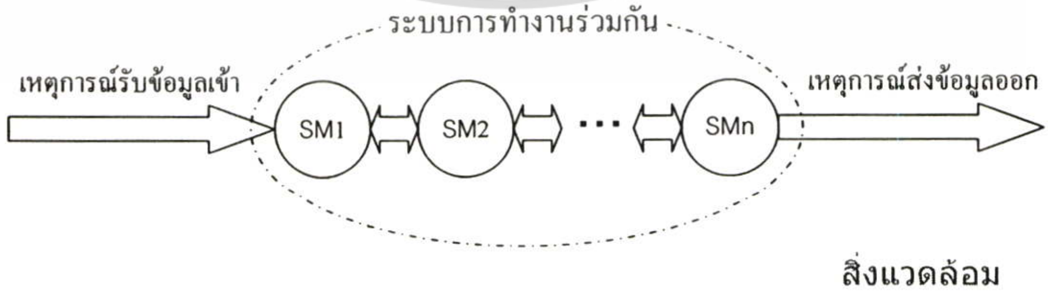
จากตัวอย่างที่ 2.1 สามารถเขียนเส้นทางการเปลี่ยนแปลงของตัวกลางได้ คือ

tr = *s*?“d1” *s*?“d2” *r*!“d1” *s*?“d3” …..

2.3 ระบบการทำงานร่วมกัน(Concurrent System)

ระบบการทำงานร่วมกัน คือ ระบบที่ประกอบด้วยตัวทำงานหรือสเตทแมชชีนตั้งแต่ 2 ตัวขึ้นไปที่สามารถสื่อสารและทำงานร่วมกันได้ ในหัวข้อนี้จะนิยามพฤติกรรมของระบบการทำงานร่วมกัน โดยการสร้างลำดับสถานะการทำงานของระบบ ซึ่งสามารถบ่งบอกพฤติกรรมของแต่ละสเตทแมชชีนย่อย และพฤติกรรมการติดต่อระหว่างกันได้

รูปที่ 2.3 แสดงแผนภาพของระบบการทำงานร่วมกัน ซึ่งประกอบไปด้วยสเตทแมชชีน n ตัว (SM1, SM2, … , SMn) ทำงานร่วมกันอยู่ โดยเหตุการณ์รับ-ส่งข้อมูลข้อมูลของแต่ละสเตทแมชชีนระหว่างกันจะเป็นเหตุการณ์ภายในของระบบการทำงานร่วมกัน และเหตุการณ์รับข้อมูลเข้าและส่งข้อมูลออกสู่สิ่งแวดล้อมภายนอก จะเป็นเหตุการณ์รับข้อมูลเข้า และส่งข้อมูลออกสู่สิ่งแวดล้อมภายนอกของระบบการทำงานร่วมกัน



รูปที่ 2.3 แผนภาพแสดงโครงสร้างของระบบการทำงานร่วมกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นิยามที่ 2.4 กำหนดให้ $M_1,\ldots, M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน ให้ S = $((c_1,\ldots, c_i,\ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_i \cup \ldots \cup \mathbf{v}_n)$ ซึ่งประกอบด้วยกลุ่มสถานะควบคุม และเซตค่าตัวแปรในแต่ละสเตทแมชชีนย่อย $M_i (1 \leq i \leq n)$ โดยที่ $V_i \cap V_j = \phi$ สำหรับทุก i และ j $(1 \leq i \leq n, 1 \leq j \leq n, i \neq j)$ เป็นสถานะของระบบการทำงานร่วมกัน และการถ่ายทอดสถานะของระบบ S ไปยังสเตทแมชชีน M_i ให้เขียนแทนด้วย $S^{(i)}$ ส่วนลำดับสถานะและเหตุการณ์ของระบบ α = $S_0\sigma_0S_1\sigma_1S_2\sigma_2S_3$ ... ให้เรียกว่า ลำดับการทำงานของระบบการทำงานร่วมกัน และถ้ามีการใช้ช่องสัญญาณ c ร่วมกันอย่างน้อย 2 สเตทแมชชีนแล้ว c จะถูกเรียกเป็น ช่องสัญญาณร่วม

เมื่อแทน c_i ด้วย c_i' และแทน $\mathbf{v}_i$ ด้วย $\mathbf{v}_i'$ ลงในสถานะของระบบ $S = ((c_1,\ldots, c_i,\ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_i \cup \ldots \cup \mathbf{v}_n)$แล้วให้แสดงด้วย $S[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$ ถ้ามีการแทนที่ตั้งแต่ 2 ตัวขึ้นไปให้แสดงต่อกันไปเรื่อยๆ ตัวอย่างเช่น ถ้าแทนที่ c_j ด้วย c_j' และแทน $\mathbf{v}_j$ ด้วย $\mathbf{v}_j'$ $(j \neq i)$ ต่อจากการแทนที่ข้างต้นแล้วจะแสดงเป็น $S[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$ $[c_j'/c_j, \mathbf{v}_j'/\mathbf{v}_j]$. ถ้ามีสถานะของระบบ S และเหตุการณ์ส่งข้อมูลออก $c!d$ และมี $S^{(i)}$ ที่ทำให้ $S^{(i)} \xrightarrow{c!d}$ แล้วกำหนดให้เซตของการเปลี่ยนสถานะของสเตทแมชชีนอื่นซึ่งทำให้ $S^{(j)} \xrightarrow{c?d} (c_j', \mathbf{v}_j')$ สำหรับบางสถานะ $(c_j', \mathbf{v}_j')$ เป็น $Peer(S^{(i)} \xrightarrow{c!d})$

นิยามที่ 2.5 กำหนดให้ $M_1,\ldots, M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน ลำดับการทำงานของระบบ $\alpha = S_0\sigma_0S_1\sigma_1S_2\sigma_2S_3\ldots$ จะเกิดขึ้นได้ ถ้าเงื่อนไขดังต่อไปนี้เป็นจริง

1. $S_0 = ((c_1,\ldots, c_i,\ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_i \cup \ldots \cup \mathbf{v}_n)$ ซึ่ง $S_0^{(i)} \in \theta_i$ สำหรับทุก i $(1 \leq i \leq n)$
2. ให้ $S_j = ((c_1,\ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_n)$ สำหรับแต่ละ $j \geq 0$ แล้ว σ_j และ S_{j+1} จะเป็นไปตามเงื่อนไข (a), (b), หรือ (c)

   (a) $\sigma_j = \tau$

   สำหรับบาง i $(1 \leq i \leq \text{n})$, ช่องสัญญาณร่วม c, ข้อมูล d, และ สถานะ $(c_i', \mathbf{v}_i')$ ของสเตทแมชชีน M_i,

   $S_j^{(i)} \xrightarrow{c!d} (c_i', \mathbf{v}_i')$

   $Peer(S_j^{(i)} \xrightarrow{c!d}) = \{ S_j^{(j1)} \xrightarrow{c?d} (c_{j1}', \mathbf{v}_{j1}'), \ldots, S_j^{(jm)} \xrightarrow{c?d} (c_{jm}', \mathbf{v}_{jm}')\} \neq \phi$ และ

   $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$ $[c_{j1}'/c_{j1}, \mathbf{v}_{j1}'/\mathbf{v}_{j1}] \ldots [c_{jm}'/c_{jm}, \mathbf{v}_{jm}'/\mathbf{v}_{jm}]$

   (b) σ_j เป็นเหตุการณ์รับหรือส่งข้อมูลผ่านช่องสัญญาณที่ไม่ใช่ช่องสัญญาณร่วม

   สำหรับบาง i $(1 \leq i \leq n)$,

   $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ และ

   $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(c) $\sigma_j = \tau$ (เหตุการณ์ภายในในสเตทแมชชีน)

สำหรับบาง i $(1 \leq i \leq n)$,

$S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ และ

$S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$

ลำดับการทำงานของระบบ แสดงถึง พฤติกรรมที่สามารถเกิดขึ้นภายในระบบการทำงานร่วมกันทั้งหมด ในนิยามนี้สมมุติว่าการสื่อสารกันระหว่างสเตทแมชชีนใดใดเกิดขึ้นพร้อมกัน สำหรับช่องสัญญาณที่ใช้ร่วมกันในสเตทแมชชีนตั้งแต่ 2 ตัวขึ้นไป การส่งข้อมูลออกจากสเตทแมชชีนหนึ่งจะเป็นการรับข้อมูลเข้าสู่สเตทแมชชีนตัวอื่นที่สามารถรับข้อมูลได้ในขณะเวลานั้น เช่น การส่งข้อมูลแบบกระจาย การรับหรือส่งข้อมูลแบบนี้เมื่อมองระบบจากสิ่งแวดล้อมจะเห็นเป็นเพียงเหตุการณ์ภายใน ส่วนข้อมูลที่ถูกส่งเข้าและออกโดยไม่ผ่านช่องสัญญาณร่วมจะไม่มีผลกระทบต่อสเตทแมชชีนตัวอื่น และเป็นการติดต่อกับสิ่งแวดล้อมเท่านั้น

ตัวอย่างที่ 2. 2 พฤติกรรมที่สามารถเกิดขึ้นได้ของระบบการทำงานร่วมกันประกอบด้วย สเตทแมชชีน ตัวส่ง ตัวกลาง และตัวรับ ในตัวอย่างที่ 2.1 ถูกแสดงดังรูปที่ 2.4 โดยให้ค่า MAX = 1, สถานะควบคุมของระบบสามารถเขียนแทนด้วย (สถานะควบคุมตัวส่ง, สถานะควบคุมตัวกลาง, สถานะควบคุมตัวรับ) และแสดงร่วมกับค่าของตัวแปร *no* อนึ่งเหตุการณ์รับข้อมูลเข้าหรือส่งข้อมูลออกระหว่างสเตทแมชชีนจะกลายเป็นเหตุการณ์ภายใน ลำดับการทำงานของระบบที่เกิดขึ้นได้บ่อยคือ ลำดับที่เริ่มจากสถานะเริ่มต้นและสิ้นสุดที่สถานะเริ่มต้นอีกครั้ง เช่น

$((S0, M0, R0), \{no = 0\})\ p?dt$

$((S1, M0, R0), \{no = 0\})\ \tau(s\langle dt\rangle)$

$((S0, M1, R0), \{no = 0\})\ \tau$

$((S0, M0, R0), \{no = 1\})\ \tau(r\langle dt\rangle)$

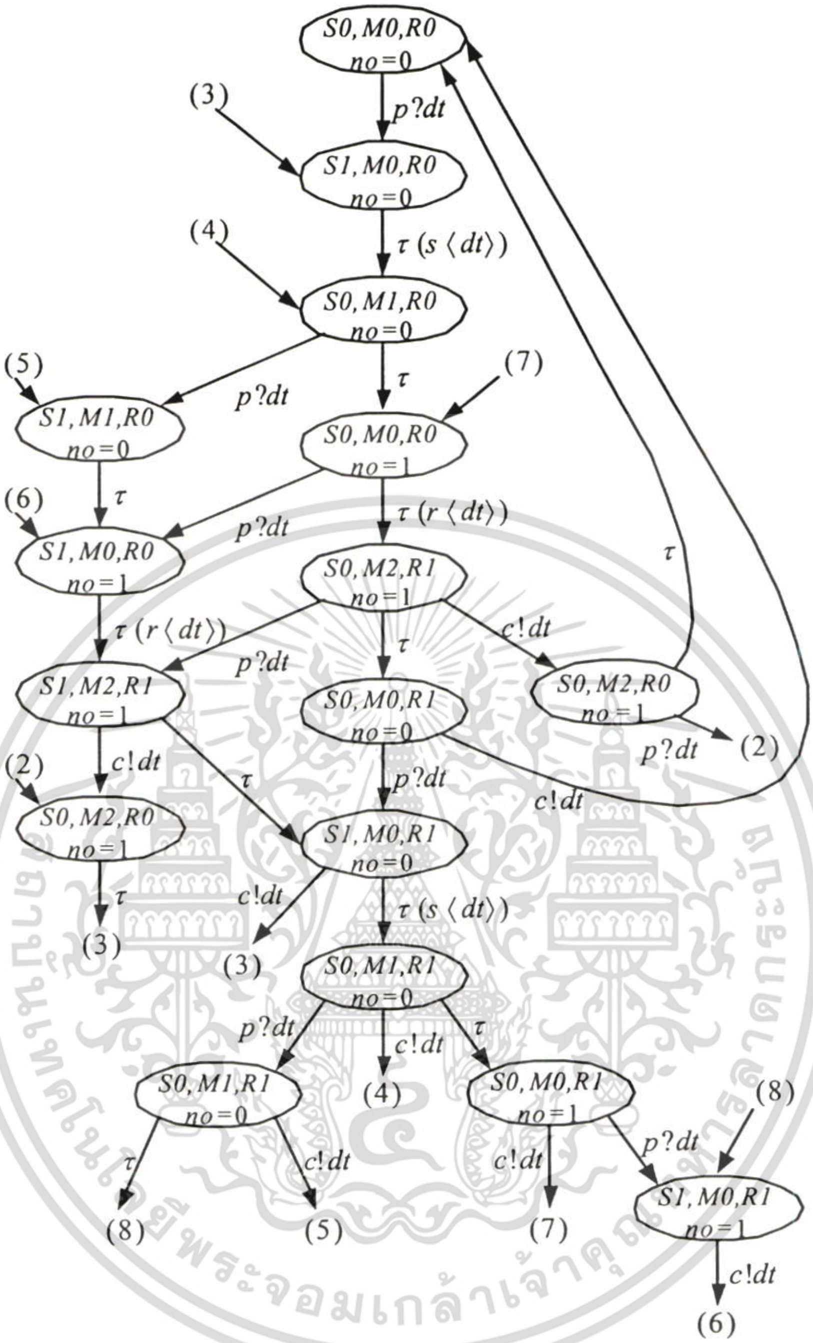
$((S0, M2, R1), \{no = 1\})\ c!dt$

$((S0, M2, R0), \{no = 1\})\ \tau$

$((S0, M0, R0), \{no = 0\})$

จากลำดับการทำงานนี้ สามารถอธิบายเป็นเส้นทางการเปลี่ยนแปลงได้คือ ตัวส่งเริ่มรับข้อมูลจากสิ่งแวดล้อมโดยผ่านช่องสัญญาณ p และส่งข้อมูลดังกล่าวให้กับตัวกลางโดยผ่านช่องสัญญาณ s และเมื่อตัวกลางได้รับข้อมูลแล้ว ตัวกลางส่งข้อมูลต่อให้กับตัวรับโดยผ่านช่องสัญญาณ r เมื่อตัวรับได้รับข้อมูลแล้ว ทำการส่งข้อมูลออกสู่สิ่งแวดล้อมโดยผ่านช่องสัญญาณ c

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.4 พฤติกรรมที่เป็นไปได้ของระบบการรับ-ส่งข้อมูล

เมื่อสังเกตลำดับเหตุการณ์ที่เกิดขึ้นจะเห็นว่า ในระบบการทำงานร่วมกันนี้ตัวส่งจะรับข้อมูลจากสิ่งแวดล้อม และทำการส่งข้อมูลภายในระบบไปสู่ตัวรับโดยผ่านตัวกลาง และสุดท้ายตัวรับจะส่งข้อมูลนั้นส่งออกไปสู่สิ่งแวดล้อมอีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
ไฟไนท์สเตทแมชชีนแบบมีข้อจำกัด

3.1 บทนำ

การออกแบบระบบการสื่อสารให้ตรงกับตามความต้องการของผู้ออกแบบนั้น แบบจำลองจะต้องมีความยืดหยุ่นในการออกแบบ คือสามารถนำมาใช้ในการออกแบบระบบได้หลากหลายซึ่งความหลากหลายมาจากข้อจำกัดและพฤติกรรมของระบบการสื่อสารที่แตกต่างกันนั่นเอง ดังนั้นเพื่อให้แบบจำลองที่ได้สามารถนำมาใช้ออกแบบกับระบบการสื่อสารได้เป็นอย่างดี จึงจำเป็นต้องมีวิธีการกำหนดข้อจำกัดทางด้านการสื่อสารไว้ในแบบจำลองด้วย โดยข้อจำกัดของการสื่อสารที่สำคัญและนำมาพิจารณานี้ได้แก่ ข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดการเคลื่อนที่ ซึ่งการเพิ่มข้อจำกัดทางเวลาทำได้โดยการเพิ่มข้อกำหนดทางเวลาให้แก่ไฟไนท์สเตทแมชชีนพื้นฐานที่นิยามไว้แล้วในบทที่ 2 ส่วนการเพิ่มข้อจำกัดทางภูมิศาสตร์และการเคลื่อนที่ ทำได้โดยการเพิ่มข้อกำหนดให้แก่แบบจำลองของระบบการสื่อสาร

3.2 ข้อจำกัดทางเวลา

ในหัวข้อนี้จะกล่าวถึงการเพิ่มข้อกำหนดทางเวลาในการเปลี่ยนสถานะของสเตทแมชชีนเพื่อให้สเตทแมชชีนที่ได้สามารถนำมาใช้ออกแบบกลไกภายในระบบการสื่อสารที่มีการทำงานหรือการเปลี่ยนแปลงที่ขึ้นกับเวลาได้ เช่น ระบบการประมวลผลตามเวลาจริง โปรโตคอลในระบบการสื่อสาร เป็นต้น

เงื่อนไขการเปลี่ยนสถานะในสเตทแมชชีนแบบมีข้อจำกัดทางเวลาจะถูกพิจารณาร่วมกับเวลาด้วย โดยเงื่อนไขนั้นจะถูกพิจารณาก็ต่อเมื่อเวลายังอยู่ภายในช่วงค่าหน่วงเวลาต่ำสุด l และค่าหน่วงเวลาสูงสุด u ให้ φ เป็น ฟังก์ชันซึ่งให้ทั้งค่าหน่วงเวลาน้อยสุดและมากสุดในแต่ละการเปลี่ยนสถานะเป็นข้อจำกัดทางเวลา กำหนดให้ฟังก์ชันค่าหน่วงเวลาน้อยสุดและมากสุดของการเปลี่ยนสถานะ μ เป็น $\varphi_l(\mu)$ และ $\varphi_u(\mu)$ โดยที่ $\varphi_l(\mu) \leq \varphi_u(\mu)$ และสมมุติว่าการเปลี่ยนสถานะใดใด เกิดขึ้นแบบทันที

นิยามที่ 3.1 กำหนดให้ $M = \langle Q, V, \sum, T, \theta \rangle$ เป็นสเตทแมชชีน และ φ เป็นฟังก์ชันรวมซึ่งให้ค่า $\varphi_l(\mu)$ และ $\varphi_u(\mu)$ ในแต่ละการเปลี่ยนสถานะ $\mu \in T$ โดยที่ $\varphi_l(\mu) \leq \varphi_u(\mu)$ แล้วดังนั้น $\langle M, \varphi \rangle$ จะถูกเรียกว่า สเตทแมชชีนแบบมีข้อจำกัดทางเวลา φ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

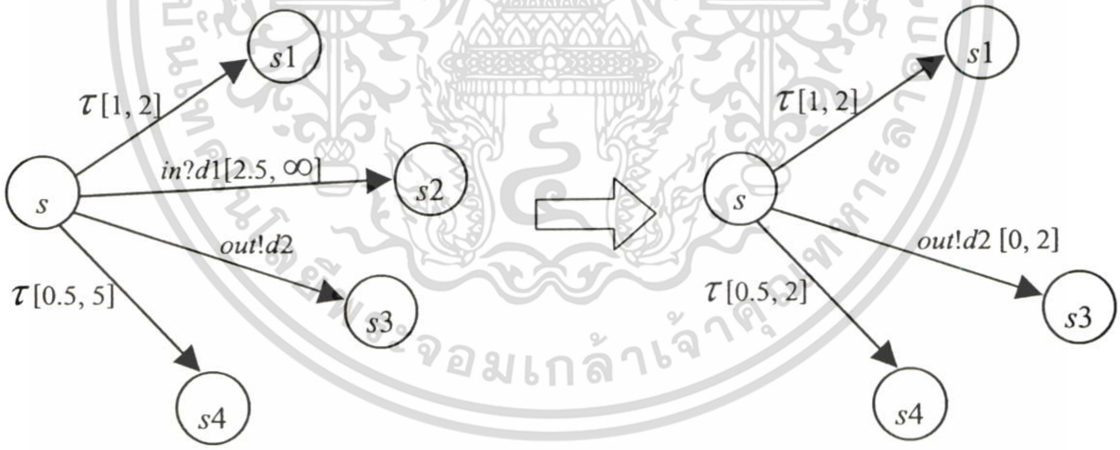
กำหนดให้สเตทแมชชีนแบบมีข้อจำกัดทางเวลามีความพิเศษคือ มีช่องสัญญาณ CLK ซึ่งสามารถให้ค่าเวลาปัจจุบันแก่สเตทแมชชีนได้ และช่องสัญญาณพิเศษนี้ไม่ส่งผลกระทบใดๆ ต่อข้อจำกัดทางภูมิศาสตร์ซึ่งจะอธิบายในหัวข้อถัดไป

ข้อกำหนดการแสดงที่ 2 ถ้ากำหนดค่าหน่วงเวลาน้อยสุด l และค่าหน่วงเวลามากสุด u ของการเปลี่ยนสถานะแล้ว แผนผังการเปลี่ยนสถานะสามารถแสดงในรูปแบบดังนี้:

$$\text{สถานะปัจจุบัน} \xrightarrow{guard \rightarrow event[l,u]} \text{สถานะถัดไป}$$

ในส่วนของ $[l, u]$ สามารถละไว้ได้ถ้า $l = 0$ และ $u = \infty$

เหตุการณ์ภายในสามารถเกิดขึ้นได้ โดยไม่จำเป็นต้องคำนึงถึงสิ่งแวดล้อมภายนอกเหมือนอย่างเหตุการณ์รับข้อมูลเข้าและส่งข้อมูลออก ดังนั้นการเปลี่ยนสถานะใดใดที่สัมพันธ์กับเหตุการณ์ภายในจะต้องทำงานภายในเวลาที่จำกัด ซึ่งนิยามต่อไปนี้จะให้ความหมายของการทำงานได้ของการเปลี่ยนสถานะ โดยพิจารณาจากค่าหน่วงเวลา



รูปที่ 3.1 การทำงานได้ของการเปลี่ยนสถานะ

นิยามที่ 3.2 ให้ s เป็นสถานะของสเตทแมชชีนแบบมีข้อจำกัดทางเวลา สำหรับการเปลี่ยนสถานะซึ่งมีสถานะเริ่มต้นเป็น s และเหตุการณ์เป็นเหตุการณ์ภายใน ค่าที่น้อยที่สุดของค่าหน่วงเวลามากสุด ให้เขียนแทนด้วย $mm(s)$ ถ้าการเปลี่ยนสถานะเหล่านั้นไม่มีเหตุการณ์ที่เป็นเหตุการณ์ภายในจะกำหนดให้ $mm(s) = \infty$ ถ้า μ เป็นการเปลี่ยนสถานะจากสถานะ s และเป็นไปตามเงื่อนไขดังต่อไปนี้แล้ว จะกล่าวได้ว่า μ สามารถเกิดขึ้นได้ที่เวลา t

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. $\varphi_l(\mu) \leq t \leq \varphi_u(\mu)$
2. $t \leq mm(s)$

รูปที่ 3.1 แสดงตัวอย่างของการทำงานได้ของการเปลี่ยนสถานะ ในกรณีที่ $mm(s) = 2$ การทำงานได้ของแต่ละการเปลี่ยนสถานะจากสถานะ s คือ

$s \xrightarrow{\tau} s_1$ สามารถเกิดขึ้นได้ในช่วงเวลา 1 ถึง 2.

$s \xrightarrow{in?d1} s_2$ ไม่สามารถเกิดขึ้นได้ในช่วงเวลาใดใด เนื่องจากค่าหน่วงเวลาน้อยสุดของเหตุการณ์นี้มีค่ามากกว่าค่า $mm(s)$.

$s \xrightarrow{out!d2} s_3$ สามารถเกิดขึ้นได้ในช่วงเวลา 0 ถึง 2.

$s \xrightarrow{\tau} s_4$ สามารถเกิดขึ้นได้ในช่วงเวลา 0.5 ถึง 2.

ดังนั้นการเปลี่ยนสถานะดังรูปที่ 3.1 ซ้ายมือ เขียนแทนได้ดังรูปที่ 3.1 ขวามือ

สำหรับลำดับการทำงานของสเตทแมชชีนแบบมีข้อจำกัดทางเวลาจะใช้ *เหตุการณ์ที่มีเวลากำกับ* และลำดับเวลา เป็นส่วนประกอบ เหตุการณ์ที่มีเวลากำกับจะแสดงอยู่ในรูป $\langle \sigma, t \rangle$ ซึ่งหมายถึงเหตุการณ์ σ ได้เกิดขึ้นเมื่อเวลา t

นิยามที่ 3.3 กำหนดให้ $\langle M, \varphi \rangle$ เป็นสเตทแมชชีนแบบมีข้อจำกัดทางเวลา ลำดับการทำงานของ $\langle M, \varphi \rangle$ คือ

$$\alpha = s_0 \langle \sigma_0, t_0 \rangle s_1 \langle \sigma_1, t_1 \rangle s_2 \langle \sigma_2, t_2 \rangle \ldots$$

ถ้าเงื่อนไขดังต่อไปนี้เป็นจริง

1. $s_0 \sigma_0 s_1 \sigma_1 s_2 \sigma_2 \ldots$ เป็นลำดับการทำงานของ M
2. $t_0 t_1 t_2 \ldots$ เป็นลำดับเวลา
3. $s_0 \xrightarrow{\sigma_0} s_1$ สามารถดำเนินการได้ที่เวลา t_0. และสำหรับทุก $i \geq 1$, $s_i \xrightarrow{\sigma_i} s_{i+1}$ สามารถดำเนินการได้ในเวลา $t_i - t_{i-1}$

เซตของลำดับการทำงานทั้งหมดของ $\langle M, \varphi \rangle$ เขียนแทนด้วย $Comp(\langle M, \varphi \rangle)$

นิยามที่ 3.4 กำหนดให้ $\alpha = s_0 \langle \sigma_0, t_0 \rangle s_1 \langle \sigma_1, t_1 \rangle s_2 \langle \sigma_2, t_2 \rangle \ldots$ เป็นลำดับการทำงานของสเตทแมชชีนแบบมีข้อจำกัดทางเวลา $\langle M, \varphi \rangle$ ลำดับที่ตัดสถานะและเหตุการณ์ที่มีเวลากำกับที่เป็นเหตุการณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภายใน(τ) ทั้งหมดออกจาก α จะถูกเรียกใหม่ว่า เส้นทางการเปลี่ยนแปลงของ $\langle M, \varphi\rangle$ เซตเส้นทางการเปลี่ยนแปลงทั้งหมดของ $\langle M, \varphi\rangle$ เขียนแทนด้วย $Trace(\langle M, \varphi\rangle)$

3.3 ข้อจำกัดทางภูมิศาสตร์

ระบบการทำงานร่วมกันสามารถจำลองการทำงานโดยอาศัยแบบจำลองไฟไนท์สเตทแมชชีนพื้นฐานหลายตัวประกอบกันร่วมกับการระบุเงื่อนไขการสื่อสารระหว่างกัน แต่หากต้องการระบุให้มีเงื่อนไขการสื่อสารในแต่ละพื้นที่แล้ว ระบบจำเป็นต้องมีข้อจำกัดทางภูมิศาสตร์ด้วย

3.3.1 เครือข่ายช่องสัญญาณ

ในหัวข้อนี้จะนิยามความหมายของ เครือข่ายช่องสัญญาณ(Field) [9, 10] เพื่อแสดงถึงขอบเขตการสื่อสารที่ครอบคลุมในแต่ละพื้นที่ของกลุ่มสเตทแมชชีน สมมุติว่าเซตจำกัด L เป็นเซตของพื้นที่ทั้งหมดซึ่งอาจมีสเตทแมชชีนอยู่ กำหนดให้แต่ละพื้นที่แทนด้วยตัวเลขจำนวนนับ(1, 2, …) และเซตทั้งหมดของช่องสัญญาณเป็น C และการสื่อสารระหว่างสเตทแมชชีนถูกจำกัดด้วยขอบเขตของช่องสัญญาณ

นิยามที่ 3.5 ถ้าเครือข่ายช่องสัญญาณ F มีสเตทแมชชีนอยู่แล้ว

$$F = \{(c, COM(c)) \mid c \in C, COM(c) \subseteq 2^L\}$$

สำหรับบาง c ซึ่ง $c \in C$, มี P ซึ่ง $P \subseteq L$ และ $P \in COM(c)$ หมายถึง สเตทแมชชีนทั้งหมดที่อยู่ในเซตพื้นที่ P สามารถติดต่อถึงกันได้โดยผ่านช่องสัญญาณ c หรืออีกนัยหนึ่งคือ สเตทแมชชีนบนพื้นที่อื่นซึ่งไม่ได้อยู่ในเซต P ไม่สามารถติดต่อกันได้โดยผ่านทางช่องสัญญาณ c และเรียก $COM(c)$ ว่าขอบเขตช่องสัญญาณ c

ตัวอย่างที่ 3.1 กำหนดให้ $L = \{1, 2, 3, 4\}$, $C = \{c_1, c_2\}$ และสมมุติให้เครือข่ายช่องสัญญาณ F คือ

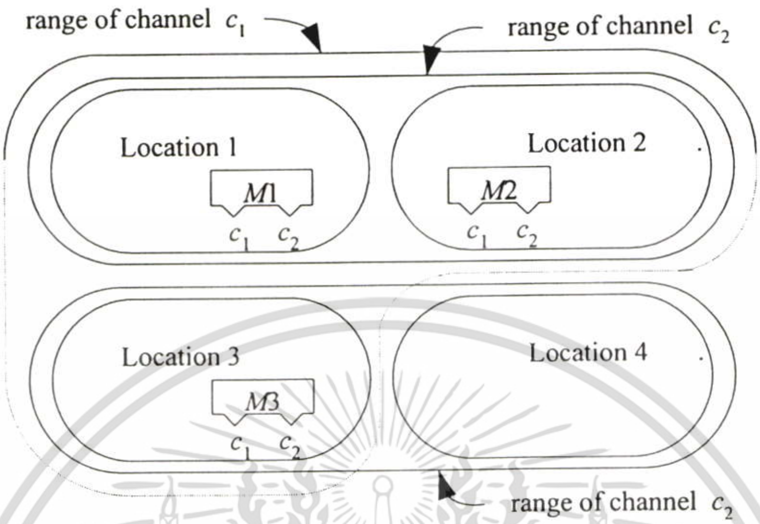
$$F = \{(c_1, \{\{1, 2, 3\}\}), (c_2, \{\{1, 2\}, \{3, 4\}\})\}$$

รูปที่ 3.2 แสดงสเตทแมชชีน $M1$ อยู่บนพื้นที่ 1, $M2$ อยู่บนพื้นที่ 2, และ $M3$ อยู่บนพื้นที่ 3 ในกรณีนี้สเตทแมชชีนทั้งหมดสามารถติดต่อกันได้โดยผ่านทางช่องสัญญาณ c_1, $M1$ และ $M2$ สามารถติดต่อกันได้โดยผ่านทางช่องสัญญาณ c_2 แต่อย่างไรก็ตาม $M3$ ไม่สามารถติดต่อกับ $M1$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

หรือ $M2$ โดยผ่านทางช่องสัญญาณ c_2 ได้ เนื่องจาก $M3$ ไม่ได้อยู่ในขอบเขตช่องสัญญาณ c_2 เดียวกันกับ $M1$ และ $M2$



รูปที่ 3.2 ตัวอย่างของเครือข่ายช่องสัญญาณ

สมมุติให้เครือข่ายช่องสัญญาณมีคุณสมบัติ 2 อย่างดังต่อไปนี้

1. ถ้าเซต P ไม่ได้อยู่ใน $COM(c)$ ซึ่ง $P \subseteq L$ และ $l \in P$ แล้วสเตทแมชชีนบนพื้นที่ l จะไม่สามารถติดต่อกับสเตทแมชชีนอื่นโดยผ่านช่องสัญญาณ c ได้
2. ถ้า P และ $P' \in COM(c)$ $(P \neq P')$ แล้วเซต P และ P' จะต้องไม่มีสมาชิกซ้ำกัน

คุณสมบัติข้อ 2 ทำให้สามารถระบุได้ว่าขอบเขตของช่องสัญญาณไหนกำลังถูกเลือกอยู่

3.3.2 พฤติกรรมของระบบการทำงานร่วมกัน

ในหัวข้อนี้จะนิยามพฤติกรรมของระบบการทำงานร่วมกันโดยการสร้างลำดับการทำงานของระบบ ซึ่งสามารถบ่งบอกถึงพฤติกรรมของสเตทแมชชีนย่อยแต่ละตัว และพฤติกรรมการติดต่อสื่อสารระหว่างกันได้ โดยในขั้นแรกจะทำการกำหนดพฤติกรรมของระบบการทำงานร่วมกันแบบไม่มีข้อจำกัดทางเวลาก่อน(แต่ยังมีข้อจำกัดทางภูมิศาสตร์อยู่) จากนั้นจึงเพิ่มข้อจำกัดทางเวลาเข้าไปในระบบ ระบบการทำงานร่วมกันบางระบบที่มีทั้งข้อจำกัดทางภูมิศาสตร์และเวลา ถ้าขอบเขตของช่องสัญญาณครอบคลุมพื้นที่ทั้งหมดแล้ว พฤติกรรมของระบบการทำงานร่วมกันที่ได้จะมีเพียงข้อจำกัดทางเวลาเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
60875

นิยามที่ 3.6 กำหนดให้ $M_1, \ldots , M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน และ F เป็นเครือข่ายช่องสัญญาณ แล้ว $\langle \{M_1, \ldots , M_n\}, F \rangle$ ถูกเรียกว่า ระบบการทำงานร่วมกันของสเตทแมชชีน $M_1, \ldots , M_n$ แบบมีข้อจำกัดทางภูมิศาสตร์ F

นิยามที่ 3.7 กำหนดให้ $\langle \{M_1, \ldots , M_n\}, F \rangle$ เป็นระบบการทำงานร่วมกันแบบมีข้อจำกัดทางภูมิศาสตร์ และ l_i เป็นพื้นที่ซึ่งมีสเตทแมชชีน M_i อยู่ ลำดับการทำงานของระบบ $\alpha = S_0 \sigma_0 S_1 \sigma_1 S_2 \ldots$ เป็นลำดับการทำงานที่มีเป็นไปได้ของ $\langle \{M_1, \ldots , M_n\}, F \rangle$ ถ้าเป็นไปตามเงื่อนไขดังต่อไปนี้

1. $S_0 = ((c_1, \ldots, c_i, \ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_i \cup \ldots \cup \mathbf{v}_n)$ ซึ่ง $S_0^{(i)} \in \theta_i$ สำหรับทุก i $(1 \le i \le n)$
2. ให้ $S_j = ((c_1, \ldots, c_n), \mathbf{v}_1 \cup \ldots \cup \mathbf{v}_n)$ สำหรับ $j \ge 0$ แล้ว σ_j และ S_{j+1} จะเป็นไปตามเงื่อนไข (a), (b), หรือ (c)

   (a) $\sigma_j = \tau$

   สำหรับบาง i $(1 \le i \le n)$, ช่องสัญญาณร่วม c, ข้อมูล d, และ สถานะ $(c_i', \mathbf{v}_i')$ ของสเตทแมชชีน M_i,

   $S_j^{(i)} \xrightarrow{c!d} (c_i', \mathbf{v}_i')$

   $Z = \{ S_j^{(j1)} \xrightarrow{c?d} (c_{j1}', \mathbf{v}_{j1}'), \ldots , S_j^{(jm)} \xrightarrow{c?d} (c_{jm}', \mathbf{v}_{jm}')\} \subseteq Peer(S_j^{(i)} \xrightarrow{c!d})$ และ

   $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i] [c_{j1}'/c_{j1}, \mathbf{v}_{j1}'/\mathbf{v}_{j1}] \ldots [c_{jm}'/c_{jm}, \mathbf{v}_{jm}'/\mathbf{v}_{jm}]$

   โดยที่ l_i อยู่ในขอบเขตหนึ่งของ c ใน F และ Z เป็นเซตย่อยที่ใหญ่ที่สุดของ $Peer(S_j^{(i)} \xrightarrow{c!d})$ ซึ่งมี $l_{j1}, \ldots, l_{jm}$ อยู่ในขอบเขตนี้

   (b) σ_j เป็นเหตุการณ์รับหรือส่งข้อมูลผ่านช่องสัญญาณที่ไม่ใช่ช่องสัญญาณร่วม

   สำหรับบาง i $(1 \le i \le n)$,

   $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ และ

   $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$ โดยที่ l_i อยู่ในขอบเขตหนึ่งของ c ใน F

   (c) $\sigma_j = \tau$ (เหตุการณ์ภายในในสเตทแมชชีนตัวเดียว)

   สำหรับบาง i $(1 \le i \le n)$,

   $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ และ

   $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$

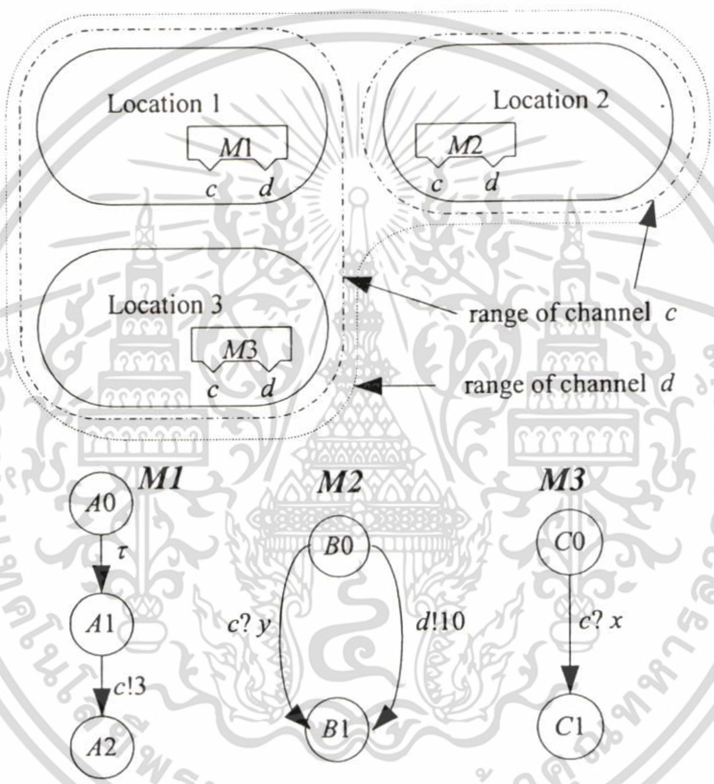
ตัวอย่างที่ 3.2 สมมุติให้ $L = \{1, 2, 3\}$, $C = \{c, d\}$, $F = \{(c, \{\{1, 3\}, \{2\}\}), (d, \{\{1, 2, 3\}\})\}$, ข้อกำหนดสเตทแมชชีนแสดงดังรูปที่ 3.3 และสเตทแมชชีน M_i $(1 \le i \le 3)$ อยู่บนพื้นที่ i แล้ว ลำดับการทำงานที่เป็นไปได้ของระบบการทำงานร่วมกันนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$\alpha 1 =$ $((A0, B0, C0), \{y = .., x = ..\})$ $\tau((A1, B0, C0), \{y = .., x = ..\})$ $\tau(c\langle 3\rangle)$ $((A2, B0,$ $C1), \{y = .., x = 3\})$ $d!10$ $((A2, B1, C1), \{y = .., x = 3\})$

$\alpha 2 =$ $((A0, B0, C0), \{y = .., x = ..\})$ $\tau((A1, B0, C0), \{y = .., x = ..\})$ $d!10$ $((A1, B1, C0),$ $\{y = .., x = ..\})$ $\tau(c\langle 3\rangle)$ $((A2, B1, C1), \{y = .., x = 3\})$

$\alpha 3 =$ $((A0, B0, C0), \{y = .., x = ..\})$ $d!10$ $((A0, B1, C0), \{y = .., x = ..\})$ $\tau((A1, B1, C0),$ $\{y = .., x = ..\})$ $\tau(c\langle 3\rangle)$ $((A2, B1, C1), \{y = .., x = 3\})$



รูปที่ 3.3 ตัวอย่างของระบบการทำงานร่วมกันแบบมีข้อจำกัดทางภูมิศาสตร์

จากตัวอย่างให้สังเกตว่า สเตทแมชชีน $M1$ และ $M2$ ไม่สามารถติดต่อสื่อสารถึงกันได้ เนื่องจาก $M1$ ไม่ได้อยู่ในขอบเขตช่องสัญญาณ c เดียวกันกับ $M2$

ตัวอย่างที่ 3.3 ให้พิจารณาระบบการให้งาน (Contract Net Protocol) [17] อย่างง่ายเป็นตัวอย่างระบบการสื่อสารแบบมีข้อจำกัดทางเวลาและภูมิศาสตร์ ระบบการให้งานเป็นการเรียกแบบให้งานตามตัวแทนในระบบหลายตัวแทน ทำงานโดยตัวแทนที่ถูกเรียกว่า ผู้จัดการ(Manager) ส่งงานให้ตัวแทนที่ถูกเรียกว่า ผู้รับงาน(Contractor) เรียกลักษณะการให้งานนี้ว่า *TA* (Task Annoucement) เมื่อผู้รับงานได้รับ TA มาแล้วผู้รับงานสามารถตัดสินใจเองได้ว่าจะรับหรือไม่รับงานนั้นก็ได้ ถ้ารับ

เอกสารนี้เป็นเอกสารที่สงวนสิทธิ์สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผู้รับงานจะเสนอการประมูล(*Bid*) กลับไปยังผู้จัดการ ผู้จัดการจะเปิดรับการประมูลภายในระยะเวลาที่จำกัด เมื่อหมดเวลาผู้จัดการจะมอบหมายงาน(*Award*) ให้แก่ผู้รับงานที่ให้การประมูลไว้สูงที่สุด ซึ่งในขณะเดียวกันผู้รับงานสามารถทำตัวเป็นผู้จัดการได้เช่นกัน ทำให้เกิดรูปแบบการทำงานเป็นลำดับชั้นขึ้นระหว่างผู้จัดการและผู้รับงาน

รูปที่ 3.4(ก) แสดงโครงสร้างของลำดับชั้นการทำงาน โดยให้เรียกผู้จัดการที่อยู่สูงที่สุดว่าผู้ควบคุม(*Controller*) ซึ่งผู้จัดการที่อยู่ในระดับล่างลงมาสามารถแสดงบทบาทเป็นผู้ควบคุมได้เช่นเดียวกัน กล่องข้อความแต่ละใบในรูปนี้แสดงถึงสเตทแมชชีนแต่ละตัว จุดทึบกลมและเส้นสีดำที่เชื่อมต่อระหว่างกล่องข้อความแสดงถึงช่องสัญญาณ และเส้นทางการสื่อสารที่เชื่อมต่อระหว่างสเตทแมชชีนแต่ละตัว ส่วนพื้นที่ที่ถูกล้อมรอบด้วยเส้นประแสดงถึงเครือข่ายช่องสัญญาณ ให้สังเกตว่าช่องสัญญาณ *c* ถูกนำมาใช้ซ้ำในการติดต่อของผู้จัดการไปยังผู้รับงานแต่ละตัว

รูปที่ 3.4(ข) แสดงข้อกำหนดต่างๆ ของสเตทแมชชีนแต่ละตัวใน (a) ส่วนข้อกำหนดสเตทแมชชีนของผู้ควบคุมคือข้อกำหนดสเตทแมชชีนของผู้จัดการอย่างง่ายนั่นเอง

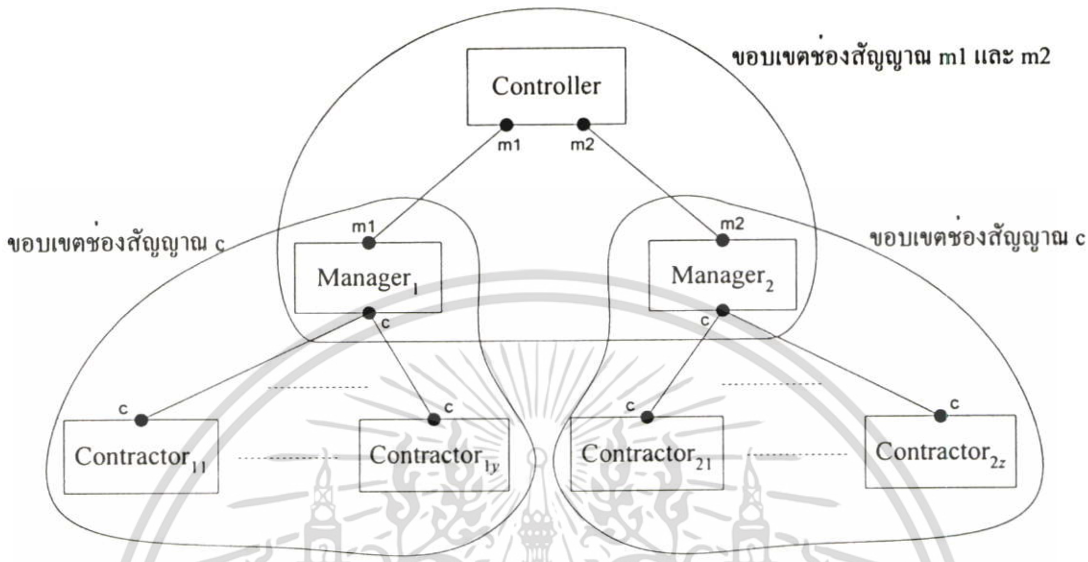
ข้อกำหนดของผู้จัดการแต่ละตัว Manger_i $(1 \leq i \leq 2)$ แสดงไว้ด้านซ้ายมือของรูป 10(b) ให้ค่าเริ่มต้นของตัวแปร *now* เป็น 0 การส่งข้อความระหว่างผู้จัดการแต่ละตัว Manger_i กับผู้รับงานที่เกี่ยวข้อง Contractor_{ij} (j = 1..y if i = 1; j = 1..z if i = 2) กระทำได้โดยส่งผ่านช่องสัญญาณ *c* และการส่งข้อความระหว่างผู้จัดการแต่ละตัว Manger_i กับผู้ควบคุม Controller กระทำได้โดยส่งผ่านช่องสัญญาณ *m* ซึ่งข้อความที่ถูกส่งประกอบไปด้วยชื่อประเภทข้อความตามด้วยตัวแปรหรือค่าคงที่ที่ต้องการส่งอยู่ในวงเล็บ สำหรับข้อกำหนดของผู้จัดการ การเปลี่ยนสถานะจากสถานะ *M2* เป็นสถานะ *M3* คือการตั้งค่าเวลาให้กับ *TA* ในสถานะ *M3* เวลาปัจจุบันในระบบจะถูกอ่านและเก็บไว้ที่ตัวแปร *now* ตลอดเวลา เพื่อนำมาใช้ในเงื่อนไขของการเปลี่ยนสถานะจากสถานะ *M3* เป็นสถานะ *M5* ซึ่งเงื่อนไขนี้จะเป็นจริงก็ต่อเมื่อสิ้นสุดเวลาการประมูล(*BP*) และถ้าเงื่อนไขเป็นจริงจะเข้าสู่ขบวนการมอบหมายงาน การประมูลจะต้องเกิดขึ้นภายในเวลา *BP* ซึ่งขบวนการประมูล *BidPoc* จะทำหน้าที่ในการตัดสินใจและเลือกผู้รับงานที่จะได้รับการมอบหมายงาน โดยการระบุค่าเฉพาะตัวของผู้รับงานในตัวแปร *ACID*

ข้อกำหนดของผู้รับงานแต่ละตัว แสดงไว้ด้านขวามือของรูปที่ 3.4(ข) ให้สังเกตการเปลี่ยนสถานะจากสถานะ *C3* เป็นสถานะ *C4* ถ้าผู้รับงานได้รับการมอบหมายงานจาก *TA* แล้ว ผู้รับงานจะทำงานด้วยขบวนการ *TaskProc* ซึ่งใช้เวลาทำงานตั้งแต่ P_{min} ถึง P_{max} และการเปลี่ยนสถานะจาก *C1* เป็น *C0* เกิดขึ้นได้ในกรณีที่ผู้รับงานไม่สนใจการประมูลงาน *TA*

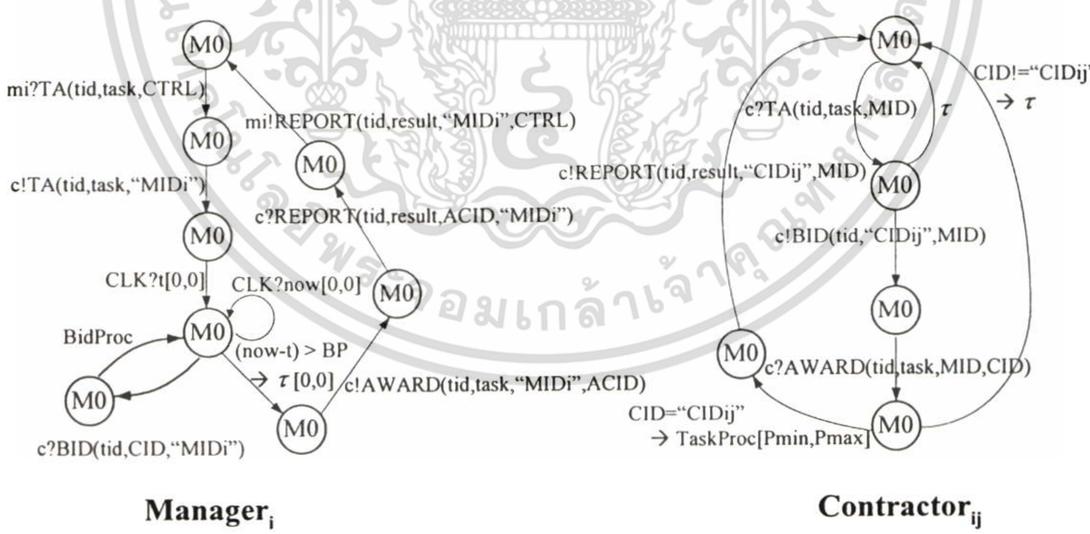
ลำดับการทำงานที่เป็นไปได้ของระบบการสื่อสารนี้ ขึ้นอยู่กับข้อจำกัดทางภูมิศาสตร์และข้อกำหนดของสเตทแมชชีน ตัวอย่างพฤติกรรมที่เป็นไปได้คือ ผู้ควบคุมส่งประกาศ *TA* ให้แก่ผู้จัดการ ผู้จัดการส่งต่อให้แก่ผู้รับงานต่างๆ ที่เกี่ยวข้อง หลังจากนั้นเมื่อผู้จัดการได้รับการเสนอการประมูลจากผู้รับงานและได้มอบหมายงานให้แก่ผู้รับงานที่เหมาะสมแล้ว ผู้รับงานที่ได้รับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มอบหมายจะทำงานจนเสร็จและส่งรายงานแจ้งกลับมายังผู้จัดการ สุดท้ายผู้จัดการส่งรายงานต่อให้กับผู้ควบคุม



(ก) โครงสร้าง



(ข) ข้อกำหนดสเตทแมชชีน

รูปที่ 3.4 ระบบการให้งาน(Contract Net Protocol)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4 ข้อจำกัดการเคลื่อนที่

ข้อจำกัดการเคลื่อนที่ หมายความถึง ข้อจำกัดใดใดที่เกิดขึ้นและมีผลต่อสเตทแมชชีนเนื่องมาจากการเคลื่อนที่ของสเตทแมชชีนนั้นๆในแต่ละพื้นที่ ดังนั้นข้อจำกัดการเคลื่อนที่จะขึ้นอยู่กับหรือถูกจำกัดโดยการกำหนดข้อจำกัดทางภูมิศาสตร์

นิยามต่อไปนี้จะกล่าวถึง การมีอยู่ของสเตทแมชชีนในแต่ละพื้นที่หรือตำแหน่งสเตทแมชชีน เพื่อนำมาใช้ในการอ้างอิงถึง ตำแหน่งของสเตทแมชชีนทั้งหมดที่อยู่ในระบบการสื่อสาร

นิยามที่ 3.8 กำหนดให้ M เป็นเซตของสเตทแมชชีนทั้งหมด และ L เป็นเซตของพื้นที่ แล้ว เซตตำแหน่งสเตทแมชชีน P คือ

$$P = \{(l, N \mid l \in L, N \subseteq M\}$$

เนื่องจากการกำหนดพื้นที่ไม่ทับซ้อนกัน ดังนั้นในแต่ละพื้นที่จะมีสเตทแมชชีนได้ไม่ซ้ำกัน

ตัวอย่างที่ 3.4 จากรูปที่ 3.5 สมมุติให้ $L = \{1, 2, 3, 4\}$, $C = \{c_1, c_2\}$ และ $F = \{(c_1, \{\{1, 2, 3\}\}), (c_2, \{\{1, 2\}, \{3, 4\}\})\}$

จากรูปที่ 3.5 ด้านบน ตำแหน่งสเตทแมชชีน คือ

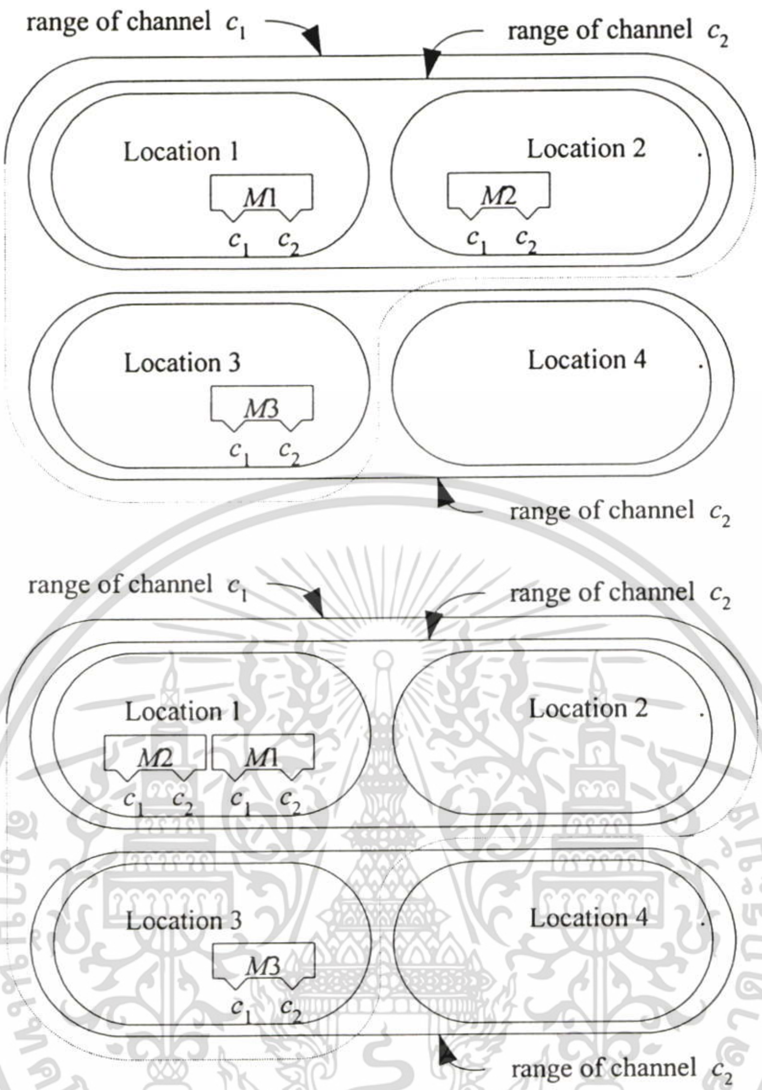
$$P = \{(1, \{\mathrm{M1}\}), (2, \{\mathrm{M2}\}), (3, \{\mathrm{M3}\}), (4, \phi)\}$$

จากรูปที่ 3.5 ด้านล่าง สเตทแมชชีน M2 เคลื่อนที่จากพื้นที่ 2 ไปยังพื้นที่ 1 ตำแหน่งสเตทแมชชีนคือ

$$P' = \{(1, \{\mathrm{M1}, \mathrm{M2}\}), (2, \phi), (3, \{\mathrm{M3}\}), (4, \phi)\}$$

ในข้อจำกัดการเคลื่อนที่ เรานิยามข้อกำหนดตามรูปแบบของระบบการสื่อสารแบบเคลื่อนที่ โดยสมมุติให้สเตทแมชชีนแต่ละตัวสามารถเคลื่อนที่เป็นอิสระจากกัน และสามารถเคลื่อนที่ไปยังที่ใดก็ได้ ดังนั้นข้อจำกัดทางด้านการสื่อสารที่เกิดขึ้นจะขึ้นอยู่กับข้อจำกัดทางภูมิศาสตร์ในแต่ละพื้นที่ที่สเตทแมชชีนนั้นๆ อยู่ จากการนิยามข้างต้นทำให้ทราบถึงตำแหน่งและพฤติกรรมการเคลื่อนที่ของสเตทแมชชีน ดังนั้นแบบจำลองดังกล่าวจึงสามารถนำมาประยุกต์ใช้ในการออกแบบระบบการสื่อสารแบบเคลื่อนที่ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.5 แสดงการเคลื่อนที่ของสเตทแมชชีน M2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

โปรแกรมช่วยตรวจสอบและจำลองการทำงาน

4.1 ลักษณะของโปรแกรม

จากวิธีการระบุข้อกำหนดที่ได้กล่าวมาแล้ว สามารถนำมาใช้ออกแบบระบบการสื่อสารทั้งแบบมีข้อจำกัดทางเวลา ทางภูมิศาสตร์ และการเคลื่อนที่ได้ แต่การนำวิธีการดังกล่าวมาใช้ออกแบบระบบได้จริงและมีประสิทธิภาพได้นั้นจำเป็นต้องอาศัยเครื่องมือช่วยในการออกแบบ ในงานวิจัยนี้จึงได้พัฒนาเครื่องมือช่วยตรวจสอบและจำลองระบบมีชื่อว่า MGraphGen ขึ้นมาเพื่อช่วยให้การออกแบบระบบมีความสะดวกและรวดเร็วมากขึ้น

MGraphGen ถูกพัฒนาขึ้นโดยอาศัยเครื่องมือพัฒนาโปรแกรม ไมโครซอฟท์ วิชวล เบสิก มีความสามารถทำงานได้ตามข้อกำหนดที่สร้างขึ้นด้วยวิธีการที่ได้นำเสนอมาแล้ว โปรแกรมจะเริ่มทำงานเมื่อได้รับการป้อนแฟ้มข้อกำหนด ตัวโปรแกรมประกอบไปด้วยเครื่องมือช่วยตรวจสอบทดสอบ และเครื่องมือจำลองการทำงานของระบบ เพื่อให้ผู้ใช้สามารถตรวจสอบความถูกต้องของข้อกำหนด ตรวจสอบพฤติกรรมภายใน และจำลองการทำงานของระบบได้

ความสามารถในการทำงานของโปรแกรม MGraphGen มีดังนี้

- สามารถตรวจสอบความถูกต้องของข้อกำหนดที่สร้างขึ้นได้
- สามารถแสดงผลแบบแผนภาพได้ ทำให้ง่ายต่อการทำความเข้าใจและใช้งาน
- สามารถเลือกแสดงลำดับการทำงาน และเส้นทางการเปลี่ยนแปลงของสเตทแมชชีน ตามเหตุการณ์ที่ต้องการได้
- สามารถค้นหาเส้นทางที่ใช้เวลาน้อยที่สุด โดยการระบุสถานะต้นทาง และสถานะปลายทางได้
- สามารถจำลองการทำงานของระบบเพื่อดูการทำงาน และพฤติกรรมต่างๆ ภายในระบบได้

ความต้องการขั้นพื้นฐานสำหรับการทำงาน

- ไมโครซอฟวินโดวส์ 95/98/ME/2000/XP/2003
- CPU Intel/AMD, RAM 64 MB, Hard Disk 20 MB

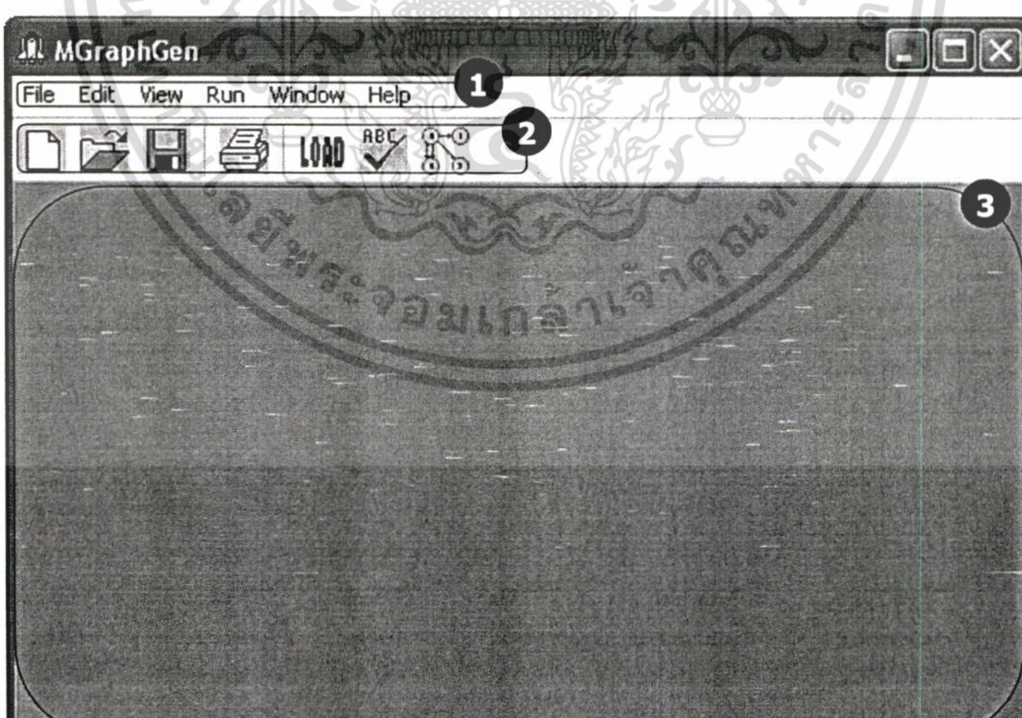
MGraphGen สามารถรับแฟ้มข้อกำหนดได้ 2 ประเภทคือ แฟ้มข้อกำหนดสเตทแมชชีน และแฟ้มข้อกำหนดระบบ ซึ่งมีรายละเอียดดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. แฟ้มข้อกำหนดสเตทแมชชีน คือ แฟ้มที่ประกอบไปด้วยข้อกำหนดต่างๆ ของสเตทแมชชีนซึ่งอาจจะมีข้อจำกัดทางเวลาหรือไม่มีก็ได้
2. แฟ้มข้อกำหนดระบบ คือ แฟ้มที่ประกอบไปด้วยข้อกำหนดต่างๆ ของการทำงานร่วมกันของสเตทแมชชีน ประกอบไปด้วยข้อกำหนดพื้นที่การให้บริการต่างๆ ข้อกำหนดเครือข่ายสัญญาณของช่องสัญญาณร่วม อนึ่งข้อกำหนดพื้นที่การให้บริการจะมีการอ้างอิงถึงแฟ้มข้อกำหนดสเตทแมชชีนเข้าไว้ด้วย ดังนั้นการป้อนแฟ้มข้อกำหนดระบบจึงเป็นเสมือนเป็นการป้อนแฟ้มข้อกำหนดสเตทแมชชีนที่อ้างอิงถึงทั้งหมดเข้าไปแล้ว

เมื่อผู้ใช้งานเรียกเปิดโปรแกรม MGraphGen แล้ว โปรแกรมจะแสดงหน้าต่างหลักซึ่งมีลักษณะและองค์ประกอบดังแสดงไว้ในรูปที่ 4.1 ดังต่อไปนี้

1. **แถบเมนู** เป็นที่สำหรับเรียกใช้คำสั่งต่างๆ ของโปรแกรม
2. **กล่องเครื่องมือ** เป็นชุดคำสั่งเครื่องมือทำงานต่างๆ ที่ใช้งานบ่อย ได้แก่ สร้างแฟ้มข้อกำหนดใหม่ เปิดแฟ้มข้อกำหนด บันทึกการแก้ไขแฟ้มข้อกำหนด พิมพ์ ป้อนแฟ้มข้อกำหนด ตรวจสอบไวยากรณ์แฟ้มข้อกำหนด และแสดงแผนผังสถานะสเตทแมชชีน
3. **พื้นที่ทำงาน** เป็นพื้นที่ที่ใช้สำหรับการออกแบบ และแสดงผล
4. **แถบแสดงสถานะ** เป็นแถบที่ใช้สำหรับสถานะของโปรแกรม



รูปที่ 4.1 หน้าต่างหลักของโปรแกรม MGraphGen

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.2 แสดงลักษณะการทำงานของโปรแกรมตามลำดับคำสั่งการใช้งาน จะเห็นได้ว่า เมื่อผู้ใช้งานเรียกเปิดโปรแกรมแล้ว ผู้ใช้งานจำเป็นต้องป้อนแฟ้มข้อกำหนดก่อนโปรแกรมจึงสามารถทำงานต่อไปได้ ซึ่งการทำงานของโปรแกรมขึ้นอยู่กับการป้อนประเภทของแฟ้มข้อกำหนด ถ้าป้อนแฟ้มข้อกำหนดระบบผู้ใช้จะสามารถจำลองการทำงานของระบบโดยอาศัยเครื่องมือจำลองการทำงานได้ และยังสามารถเลือกทดสอบสเตทแมชชีนภายในระบบโดยอาศัยเครื่องมือทดสอบสเตทแมชชีนได้อีกด้วย

รูปที่ 4.2 ลักษณะการทำงานของโปรแกรม MGraphGen

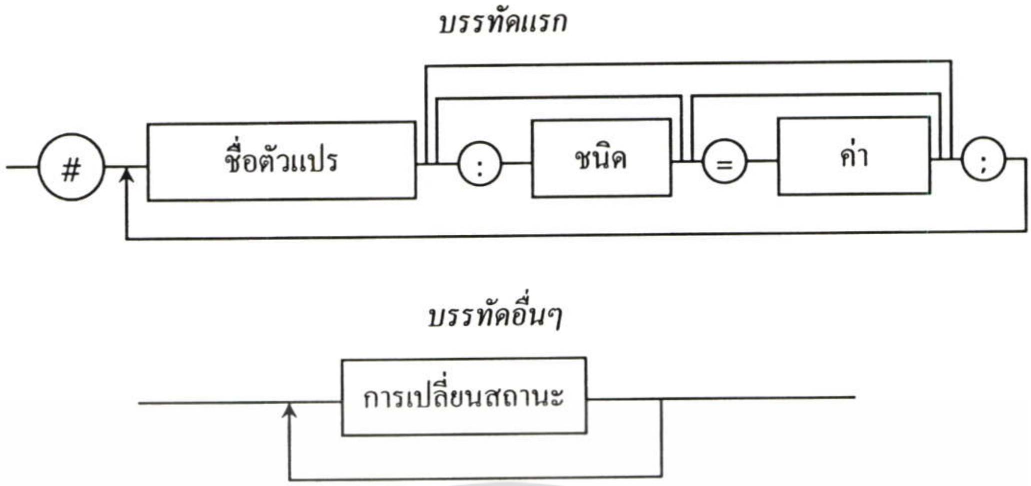
4.2 รูปแบบแฟ้มข้อกำหนด

4.2.1 แฟ้มข้อกำหนดสเตทแมชชีน

4.2.1.1 โครงสร้างแฟ้ม

โครงสร้างแฟ้มข้อกำหนดสเตทแมชชีนประกอบไปด้วย ส่วนการประกาศตัวแปร และส่วนข้อกำหนด ดังแสดงไว้ในรูปที่ 4.3 ส่วนการประกาศตัวแปรประกอบไปด้วย การระบุชื่อ ชนิด และค่าเริ่มต้นของตัวแปรต่างๆ ที่จะนำมาใช้ในข้อกำหนด ซึ่งต้องประกาศไว้ที่บรรทัดแรกสุดของแฟ้มเท่านั้น และต้องขึ้นต้นด้วยเครื่องหมายสี่เหลี่ยม(#) ส่วนข้อกำหนดประกอบไปด้วย ข้อกำหนดการเปลี่ยนสถานะต่างๆ ของสเตทแมชชีนตามวิธีการระบุข้อกำหนดที่ได้กล่าวมาแล้ว ซึ่งอาจมีข้อจำกัดทางเวลาหรือไม่มีก็ได้ รูปแบบข้อกำหนดการเปลี่ยนสถานะได้แสดงไว้ดังรูปที่ 4.4

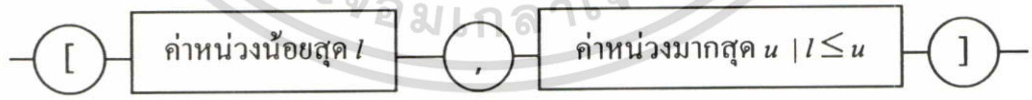
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.3 โครงสร้างแฟ้มข้อกำหนดสเตทแมชชีน

(ก) การกำหนดการเปลี่ยนสถานะ

(ข) การกำหนดเหตุการณ์



(ค) การกำหนดส่วนของเวลา

รูปที่ 4.4 รูปแบบข้อกำหนดการเปลี่ยนสถานะ

4.2.1.2 ข้อกำหนดการเปลี่ยนสถานะ

ข้อกำหนดการเปลี่ยนสถานะประกอบไปด้วย ชื่อสถานะต้นทาง เหตุการณ์ และชื่อสถานะปลายทาง ดังแสดงไว้ในรูปที่ 4.4(ก) โดยสถานะต้นทางของข้อกำหนดแรกสุดจะถูกกำหนดให้เป็นสถานะเริ่มต้นของสเตทแมชชีนนั้นๆ และชื่อสถานะต้องระบุเป็นตัวเลขทั้งหมด สำหรับเหตุการณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ของการเปลี่ยนสถานะให้อิงตามรูปแบบข้อกำหนดการแสดงที่ 1 ในหัวข้อ 2.2 ซึ่งเป็นไปได้ 3 เหตุการณ์ด้วยกัน คือ เหตุการณ์รับข้อมูลเข้าจากสิ่งแวดล้อม เหตุการณ์ส่งข้อมูลออกสู่สิ่งแวดล้อม และเหตุการณ์ภายใน ถ้าเหตุการณ์นั้นเป็นเหตุการณ์ภายในให้ใช้คำสำคัญแทนสัญลักษณ์เป็น "tau" โดยรูปแบบการกำหนดเหตุการณ์แสดงไว้ในรูปที่ 4.4(ข) และถ้าการเปลี่ยนแปลงดังกล่าวมีข้อจำกัดทางเวลาให้ใส่ส่วนของเวลาต่อท้ายเหตุการณ์นั้นไว้ด้วย ซึ่งค่าหน่วงน้อยสุดของเวลาจะต้องไม่มากไปกว่าค่าหน่วงมากสุดของเวลาเสมอ ดังแสดงไว้ในรูปที่ 4.4(ค)

ตัวอย่างที่ 4.1 แฟ้มข้อกำหนดสเตทแมชชีนตัวกลางในตัวอย่างที่ 2.1

```
#MAX:int=1; i:int=0; o:int=0; no:int=0; data[]:int
1{no<MAX-->s?data[i]}2
2{tau(no=no+1;i=(i+1)%MAX)}1
1{no>0-->r!data[o]}3
3{tau(no=no-1;o=(o+1)%MAX)}1
```

จากนิยามที่ 2.1 แฟ้มข้อกำหนดประกอบไปด้วย

Q = { 1, 2, 3 }

V = { no, i, o, data[] }

$\sum = \sum^{in} + \sum^{out} + \tau$

โดยที่

$\sum^{in}$ = { s?data[i] }

$\sum^{out}$ = { r!data[o] }

τ = { tau(no=no+1;i=(i+1)%MAX), tau(no=no-1;o=(o+1)%MAX) }

T = { (1, {no, i, o, data[]}) *s?data[i]* (2, {no, i, o, data[]}),

(1, {no, i, o, data[]}) *r!data[o]* (3, {no, i, o, data[]}),

(2, {no, i, o, data[]}) tau(no=no+1;i=(i+1)%MAX) (1, {no, i, o, data[]}),

(3, {no, i, o, data[]}) tau(no=no-1;o=(o+1)%MAX) (1, {no, i, o, data[]}) }

θ = { (1, no, i, o, data[]) }

จากแฟ้มข้อกำหนดสเตทแมชชีนในตัวอย่างที่ 4.1 อธิบายได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- บรรทัดแรกประกาศใช้ตัวแปรภายในแฟ้มข้อกำหนดให้เป็นแบบจำนวนเต็ม คือ MAX, i, o, no โดยกำหนดให้มีค่าเริ่มต้นเป็น 1, 0, 0, 0 ตามลำดับ และตัวแปร data[] มีรูปแบบเป็นอาเรย์ของจำนวนเต็ม
- บรรทัดที่ 2 กำหนดการเปลี่ยนสถานะจากสถานะ 1 ไปเป็นสถานะ 2 ด้วยเหตุการณ์รับข้อมูลจากช่องสัญญาณ s แล้วเก็บลงตัวแปร data[] ตำแหน่งที่ i แต่มีเงื่อนไขว่า ตัวแปร no จะต้องมีค่าน้อยกว่าค่าในตัวแปร MAX
- บรรทัดที่ 3 กำหนดการเปลี่ยนสถานะจากสถานะ 2 ไปเป็นสถานะ 1 ด้วยเหตุการณ์ภายใน คือ การแก้ไขค่าในตัวแปร no และ i
- บรรทัดที่ 4 กำหนดการเปลี่ยนสถานะจากสถานะ 1 ไปเป็นสถานะ 3 ด้วยเหตุการณ์ส่งข้อมูลที่เก็บไว้ในตัวแปร data[] ตำแหน่งที่ o ออกสู่ช่องสัญญาณ r แต่มีเงื่อนไขว่า ตัวแปร no จะต้องมีค่าน้อยกว่าค่าในตัวแปร MAX
- บรรทัดที่ 5 กำหนดการเปลี่ยนสถานะจากสถานะ 3 ไปเป็นสถานะ 1 ด้วยเหตุการณ์ภายใน คือ การแก้ไขค่าในตัวแปร no และ i

4.2.2 แฟ้มข้อกำหนดระบบ

โครงสร้างแฟ้มข้อกำหนดระบบประกอบไปด้วย ส่วนการประกาศพื้นที่การให้บริการ และส่วนการประกาศช่องสัญญาณร่วม โดยมีรูปแบบข้อกำหนดดังต่อไปนี้

การประกาศพื้นที่การให้บริการ

```
[Location]
    ชื่อพื้นที่การให้บริการ = ชื่อแฟ้มข้อกำหนดสเตทแมชชีน, …
    …
```

ในบรรทัดแรกของส่วนนี้ต้องใส่คำสำคัญ "[Location]" เพื่อระบุว่าเป็นส่วนการประกาศพื้นที่การให้บริการ บรรทัดต่อๆ มาให้แสดงพื้นที่การให้บริการที่ใช้ในระบบทั้งหมด โดยในแต่ละพื้นที่การให้บริการต้องระบุถึงสเตทแมชชีนทั้งหมดที่มีอยู่บนพื้นที่นั้นๆ ด้วย

การประกาศช่องสัญญาณร่วม

```
[SharedChannel]
    ชื่อช่องสัญญาณร่วม = เซตของพื้นที่ทั้งหมดที่อยู่ในขอบเขตช่องสัญญาณร่วมนี้
    …
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในบรรทัดแรกของส่วนนี้ต้องใส่คำสำคัญ "[SharedChannel]" เพื่อระบุว่าเป็นส่วนการประกาศช่องสัญญาณร่วม บรรทัดต่อๆ มาให้แสดงช่องสัญญาณร่วมที่ใช้ในระบบทั้งหมด โดยในแต่ละช่องสัญญาณร่วมต้องระบุถึง เซตของพื้นที่การให้บริการทั้งหมดที่อยู่ในขอบเขตของช่องสัญญาณนั้นๆ

ตัวอย่างที่ 4.2 แฟ้มข้อกำหนดระบบของระบบการให้งานในตัวอย่างที่ 3.3

```
[Location]
1=Controller.txt
2=manager1.txt,contractor11.txt,contractor12.txt
3=manager2.txt,contractor21.txt,contractor22.txt
[SharedChannel]
m1={1,2}
m2={1,3}
c={2},{3}
```

จากนิยามที่ 3.5 และ 3.8 สามารถแสดง เซตพื้นที่การให้บริการ L, เซตช่องสัญญาณร่วม C, ขอบเขตช่องสัญญาณ F, และตำแหน่งของสเตทแมชชีน P เป็นดังนี้

$L = \{1, 2, 3\}$

$C = \{\mathrm{m1}, \mathrm{m2}, \mathrm{c}\}$

$F = \{(\mathrm{m1}, \{1, 2\}), (\mathrm{m2}, \{1, 3\}), (\mathrm{c}, \{\{2\}, \{3\}\})\}$

$P = \{(1, \{\mathrm{Controller}\}), (2, \{\mathrm{manager1}, \mathrm{contractor11}, \mathrm{contractor12}\}),$
$(3, \{\mathrm{manager2}, \mathrm{contractor21}, \mathrm{contractor22}\})\}$

การระบุสเตทแมชชีนในแต่ละพื้นที่การให้บริการในบรรทัดที่ 2, 3 และ 4 ต้องระบุเป็นชื่อแฟ้มข้อกำหนดสเตทแมชชีนต่างๆ ที่มีอยู่ในระบบด้วย เพื่อใช้ในการอ้างอิงและการทำงานภายในระบบ ดังนั้นแฟ้มข้อกำหนดสเตทแมชชีนดังกล่าวควรถูกสร้างขึ้นก่อนการสร้างแฟ้มข้อกำหนดระบบและป้อนให้กับโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การทดสอบสเตทแมชชีน

การออกแบบระบบการสื่อสารให้ทำงานได้อย่างถูกต้องตามความต้องการนั้น จำเป็นต้องมีการทดสอบการทำงานของกลไกแต่ละกลไกภายในระบบเสียก่อน ซึ่งก็คือการทดสอบการทำงานของสเตทแมชชีนนั่นเอง และเมื่อกลไกภายในระบบทำงานได้อย่างถูกต้องดีแล้ว จึงทำการทดสอบการทำงานร่วมกันในระบบต่อไป ในหัวข้อนี้จะกล่าวถึงวิธีการตรวจสอบข้อกำหนด และการแสดงแผนผังสถานะสเตทแมชชีน จากนั้นจะกล่าวถึงวิธีการทดสอบแบบต่างๆ ด้วยเครื่องมือช่วยทดสอบสเตทแมชชีนกับข้อกำหนดของสเตทแมชชีนในระบบที่ออกแบบ ด้วยเครื่องมือช่วยนี้ทำให้นักออกแบบระบบสามารถทดสอบและวิเคราะห์การทำงานภายในระบบได้โดยง่าย

ความสามารถของเครื่องมือช่วยทดสอบสเตทแมชชีนมีดังนี้

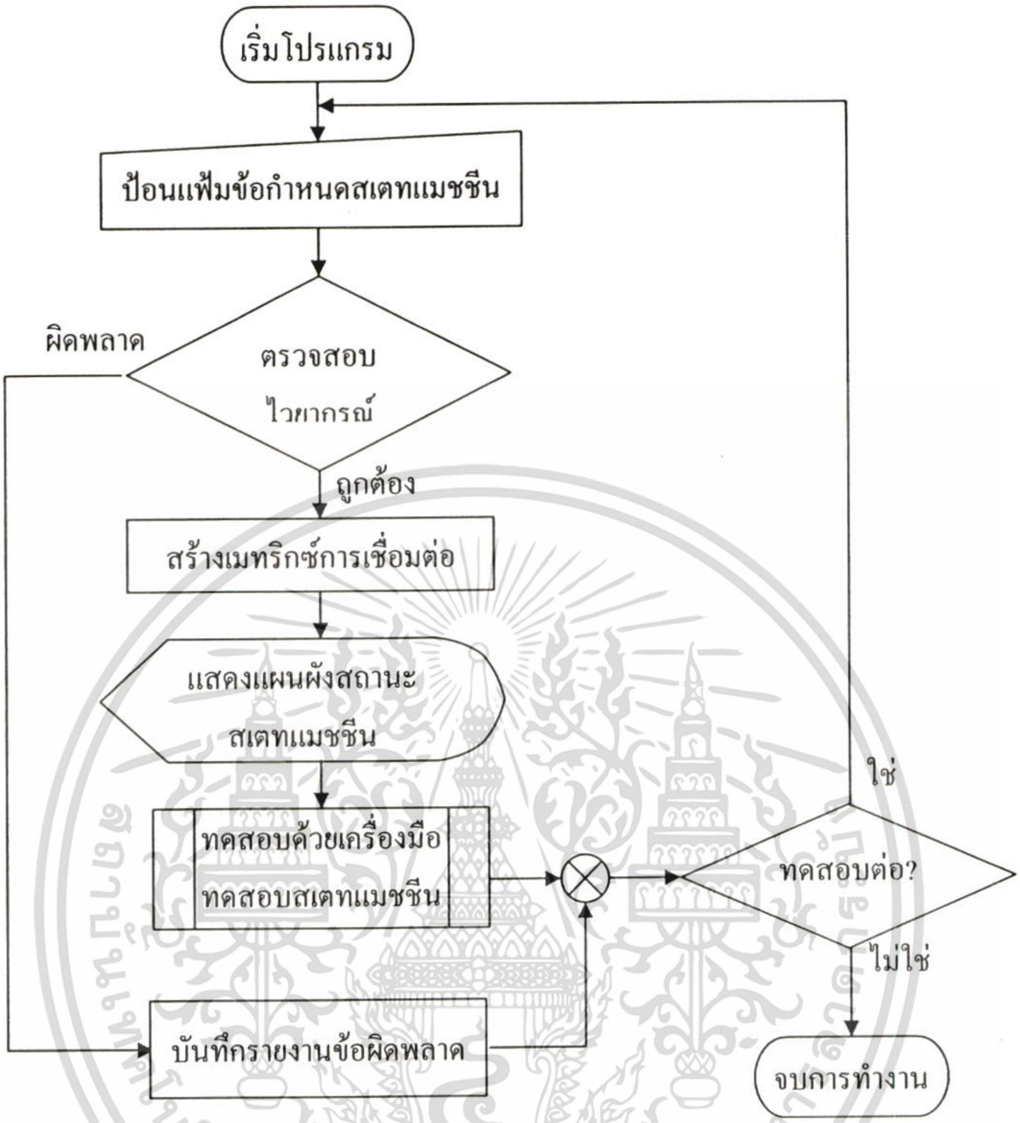
- ตรวจสอบความถูกต้องของแฟ้มข้อกำหนดสเตทแมชชีน
- แสดงข้อกำหนดของสเตทแมชชีนในรูปแบบของแผนผังสถานะ
- ค้นหาเส้นทางที่ใช้เวลาน้อยที่สุดจากสเตทแมชชีนที่ต้องการ
- แสดงลำดับการทำงานและเส้นทางการเปลี่ยนแปลงของสเตทแมชชีนตามลำดับที่ถูกเลือก

ข้อจำกัดของเครื่องมือช่วยทดสอบสเตทแมชชีนมีดังนี้

- สามารถแสดงสถานะภายในสเตทแมชชีน ได้สูงสุด 20 สถานะ
- การเปลี่ยนสถานะระหว่างสถานะใดใด กำหนดได้ไม่เกิน 5 เหตุการณ์
- ค่าหน่วงเวลาของการเกิดเหตุการณ์กำหนดได้ไม่เกิน 62,000 หน่วยเวลา

การทดสอบสเตทแมชชีนทำได้โดยการป้อนแฟ้มข้อกำหนดสเตทแมชชีน หรือการอ้างอิงแฟ้มข้อกำหนดสเตทแมชชีนจากแฟ้มข้อกำหนดระบบซึ่งจะกล่าวต่อไปในหัวข้อ 4.4 ให้กับโปรแกรม MGraphGen จากนั้นโปรแกรมจะทำการตรวจสอบความถูกต้องของแฟ้มข้อกำหนด ถ้าไม่ถูกต้องโปรแกรมจะบันทึกรายงานข้อผิดพลาดแล้วแจ้งเตือนผู้ใช้ทันที ถ้าถูกต้องโปรแกรมจะทำงานต่อโดยการสร้างเมทริกซ์เชื่อมต่อ และแสดงแผนผังสถานะของสเตทแมชชีน ในขั้นตอนนี้ผู้ใช้สามารถทดสอบสเตทแมชชีนได้ด้วยเครื่องมือทดสอบสเตทแมชชีน ซึ่งวิธีการทำงานในส่วนต่างๆ จะได้กล่าวถึงในตอนต่อไป การทำงานของโปรแกรมเมื่อผู้ใช้งานต้องการทดสอบสเตทแมชชีนแสดงไว้ในรูปที่ 4.5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



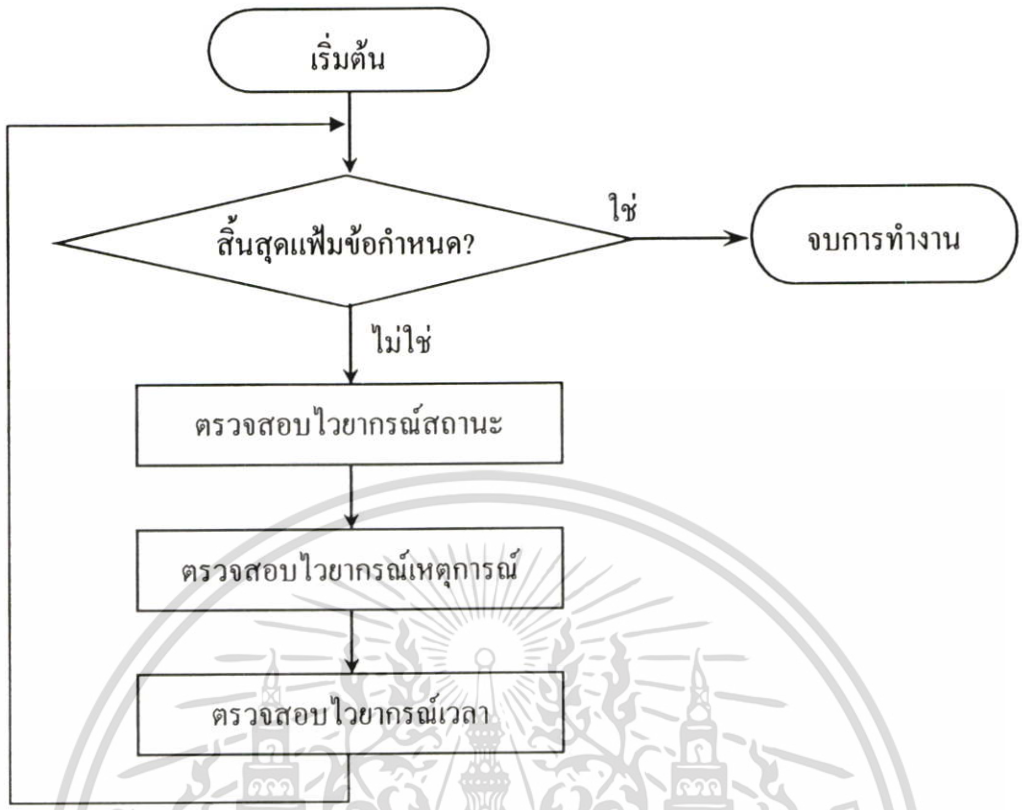
รูปที่ 4.5 ขั้นตอนการทำงานของโปรแกรมเมื่อต้องการทดสอบสเตทแมชชีน

4.3.1 การตรวจสอบไวยากรณ์

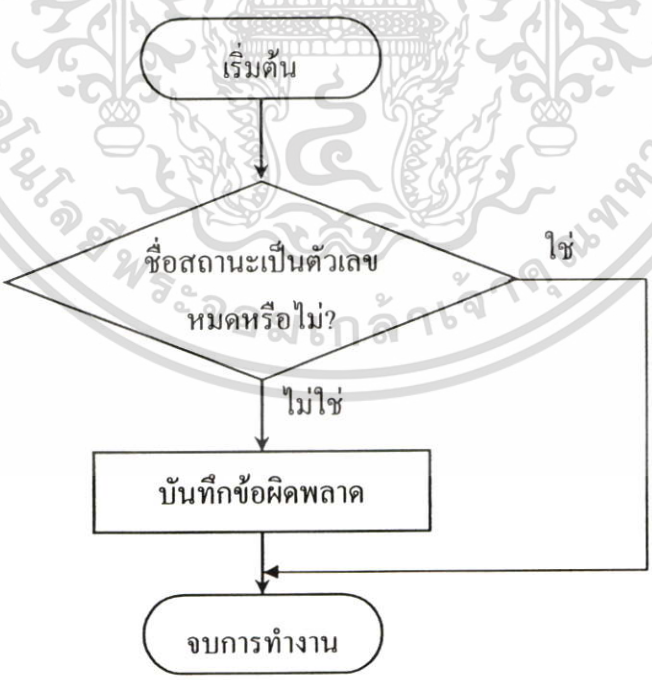
การตรวจสอบความถูกต้องในแฟ้มข้อกำหนดของโปรแกรม จะยึดตามไวยากรณ์ข้อกำหนดในหัวข้อ 4.4.2 หากกระบวนการตรวจสอบพบข้อผิดพลาดจากแฟ้มข้อกำหนดที่ป้อนเข้ามา โปรแกรมจะแจ้งข้อผิดพลาดต่างๆไว้ในแฟ้มรายงานข้อผิดพลาด(error.log) ซึ่งจะถูกใช้อ้างอิงในการแก้ไขความผิดพลาดของข้อกำหนดให้ถูกต้องต่อไป ตัวอย่างที่ 4.3 แสดงตัวอย่างแฟ้มข้อกำหนดสเตทแมชชีนที่ถูกต้องตามหลักไวยากรณ์

ขั้นตอนการตรวจสอบไวยากรณ์ของโปรแกรมแสดงไว้ในรูปที่ 4.6 เมื่อมีการป้อนแฟ้มข้อกำหนดสเตทแมชชีนเข้ามา โปรแกรมจะเริ่มตรวจสอบไวยากรณ์สถานะก่อน จากนั้นจึงตรวจสอบไวยากรณ์เหตุการณ์ และตรวจสอบไวยากรณ์เวลา โดยขั้นตอนการตรวจสอบไวยากรณ์สถานะ และเหตุการณ์ได้แสดงไว้ในรูปที่ 4.7 และ 4.8 ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

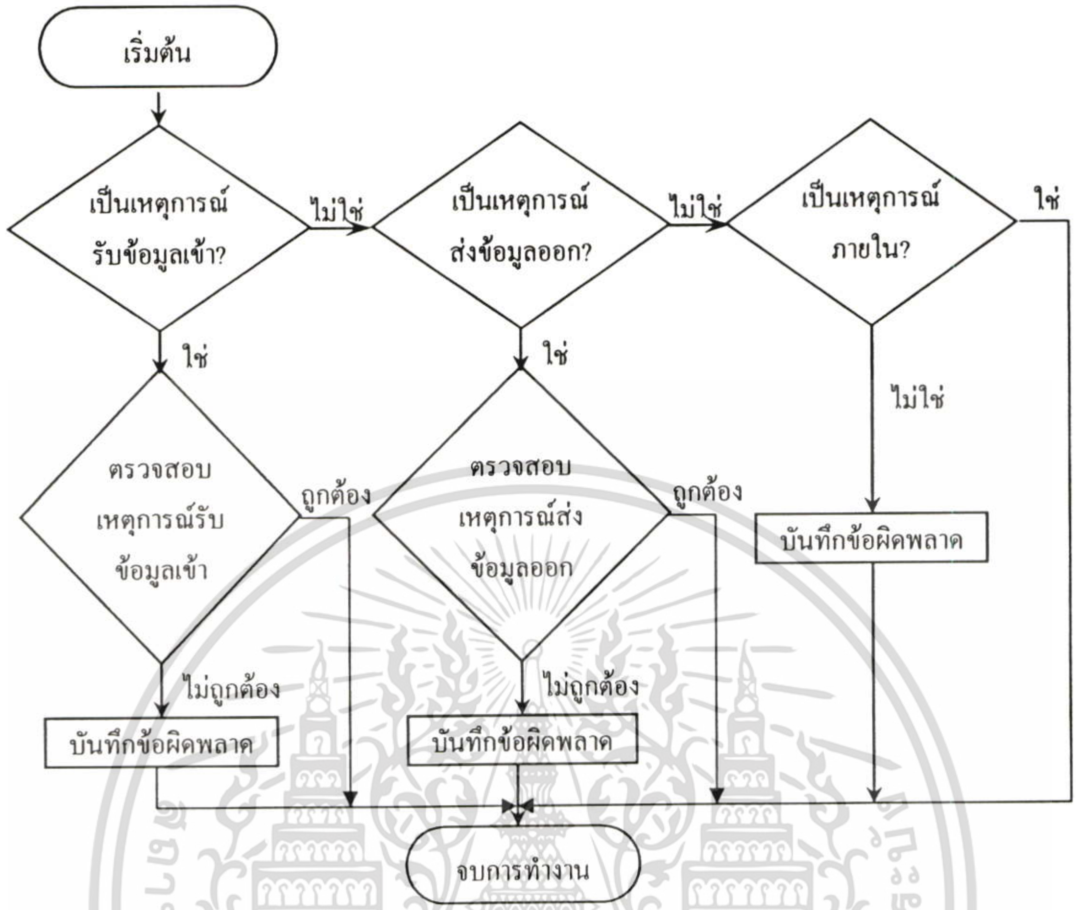


รูปที่ 4.6 ขั้นตอนการตรวจสอบไวยากรณ์แฟ้มข้อกำหนดสเตทแมชชีน



รูปที่ 4.7 ขั้นตอนการตรวจสอบไวยากรณ์สถานะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.8 ขั้นตอนการตรวจสอบไวยากรณ์เหตุการณ์

ตัวอย่างที่ 4.3 แฟ้มข้อกำหนดสเตทแมชชีนที่ถูกต้องตามหลักไวยากรณ์

```
#a;b;c;
1{Tau[2.5,5]}2
1{ch5?a[15,30]}3
2{ch6?b[10,20]}3
2{ch6?c[14,28]}4
2{Tau[9,18]}5
3{ch4!b[12,24]}6
6{ch3!(2)[8,16]}7
6{Tau[10,20]}8
7{ch2!Acdf("23")[3,6]}8
5{ch7!Acdf((12),"23")}4
4{Tau[17,34]}8
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.2 การสร้างเมทริกซ์เชื่อมต่อ

เมทริกซ์เชื่อมต่อ เป็นเมทริกซ์ที่บ่งบอกถึงจำนวนเส้นทางการเปลี่ยนสถานะจากสถานะหนึ่งไปยังอีกสถานะหนึ่ง โปรแกรมจะสร้างเมทริกซ์เชื่อมต่อหลังจากได้ผ่านการตรวจสอบไวยากรณ์ข้อกำหนดสเตทแมชชีนเรียบร้อยแล้ว หลังจากนั้นจะนำเมทริกซ์เชื่อมต่อไปใช้เพื่อแสดงแผนผังสถานะของสเตทแมชชีนต่อไป วิธีการสร้างเมทริกซ์เชื่อมต่อนั้นได้อาศัยหลักการตามทฤษฎีของกราฟ โดยสมมุติให้เซตของสถานะในสเตทแมชชีนทั้งหมดเป็นเซตของจุดทั้งหมดบนกราฟ และเซตของเส้นทางการเปลี่ยนสถานะทั้งหมดในสเตทแมชชีนเป็นเซตของด้านทั้งหมดบนกราฟ เพื่อให้เข้าใจได้ดีขึ้นจึงขอกล่าวถึงทฤษฎีกราฟโดยย่อ ดังนี้

นิยามที่ 4.1 ถ้าให้ G เป็นสัญลักษณ์แทนความหมายของกราฟแล้วองค์ประกอบที่สำคัญของ G จะประกอบด้วยเซตของจุด (node) V และเซตของด้าน (edge) E กราฟ G จะเขียนแทนด้วย G(V, E) ถ้า $\{v_1, v_2, v_3\} \subseteq$ V และ $\{v_1, v_2, v_3\} \subseteq$ E แล้วสามารถเขียนความสัมพันธ์ของจุดและด้านได้ดังนี้

$$e_1 = (v_1, v_2)$$

โดยที่ e_1 หมายถึงด้านที่เชื่อมต่อระหว่างจุด v_1 และ v_2 และเรียก v_1, v_2 ว่าจุดเชื่อมต่อ(adjacent node) ดังรูปที่ 4.9

รูปที่ 4.9 จุดและด้านของกราฟ G(V, E)

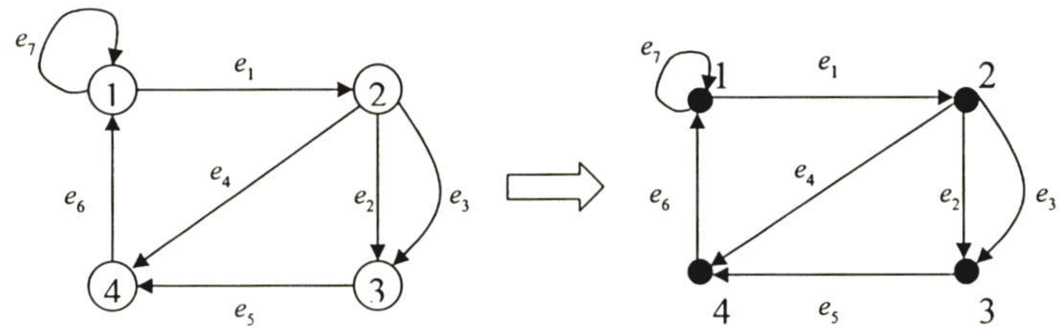
นิยามที่ 4.2 ให้ A เป็น m x n เมทริกซ์ และ $a_{ij} \in$ A โดยที่ $1 \leq i \leq m$ และ $1 \leq j \leq n$ แล้ว จะเรียก A ว่าเป็นเมทริกซ์เชื่อมต่อถ้า

$$a_{ij} = \begin{cases} d \text{ ถ้ามีเส้นทางจาก } v_i \text{ ไป } v_j \text{ จำนวน } d \text{ เส้นทาง} \\ 0 \text{ หากเป็นกรณีอื่น} \end{cases}$$

โดยที่ d เป็นจำนวนเส้นทางระหว่าสถานะ i และ j

ถ้า $a_{ij} = a_{ji}$ แสดงว่า A เป็นเมทริกซ์สมมาตร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.10 การมองสเตทแมชชีนให้อยู่ในรูปแบบของกราฟ

ตัวอย่างที่ 4.4 รูปที่ 4.10 แสดงสเตทแมชชีนในรูปแบบของกราฟ ดังนั้นเมทริกซ์เชื่อมต่อ A คือ

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

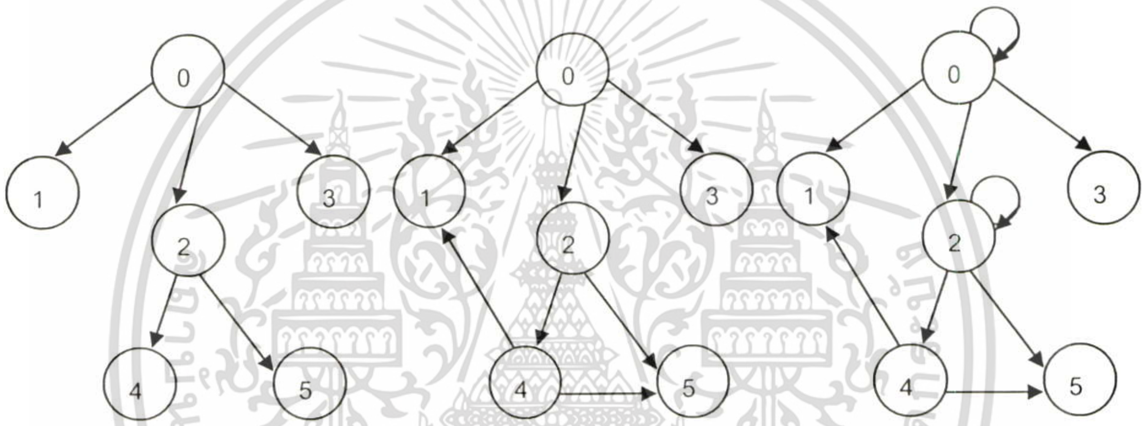
เมทริกซ์เชื่อมต่อนี้สามารถบอกถึงจำนวนเส้นทางการเชื่อมต่อระหว่างสถานะทั้งหมดภายในสเตทแมชชีนจากสถานะหนึ่งไปสถานะหนึ่ง เช่น จากสถานะ 2 มีเส้นทางการเปลี่ยนสถานะไปสถานะ 3 อยู่ 2 เส้นทาง และไปสถานะ 4 อยู่ 1 เส้นทาง ส่วนเส้นทางการเปลี่ยนสถานะจากสถานะ 1 ไปยังสถานะ 1 มีอยู่ 1 เส้นทาง ให้เรียกเส้นทางนี้ว่า เส้นทางวกกลับ (loop) เมทริกซ์เชื่อมต่อนี้มีประโยชน์มากในการแสดงแผนผังสถานะสเตทแมชชีน

4.3.3 การแสดงแผนผังสถานะสเตทแมชชีน

โปรแกรมสามารถแสดงแผนผังสถานะสเตทแมชชีน ด้วยการวาดแผนผังสถานะสเตทแมชชีนลงบนหน้าต่างแสดงผล ขั้นตอนการวาดแผนผังสถานะสเตทแมชชีนต้องอาศัยข้อมูลจากเมทริกซ์เชื่อมต่อ ดังนั้นจึงจำเป็นต้องสร้างเมทริกซ์เชื่อมต่อขึ้นมาก่อนโดยอาศัยข้อมูลจากแฟ้มข้อกำหนดสเตทแมชชีนที่ป้อนเข้ามา ซึ่งการสร้างแผนผังสถานะจากเมทริกซ์มีข้อดีคือ สามารถใช้การดำเนินการต่างๆ ที่มีอยู่ในเมทริกซ์เพื่อช่วยในการวาดแผนผังสถานะได้เช่น การสลับแถวและหลักในเมทริกซ์ เป็นต้น มีผลให้แผนผังสถานะที่ได้มีความยืดหยุ่นในการวาด และขจัดปัญหาการวาดเส้นทางการเปลี่ยนสถานะที่ทับซ้อนกัน เนื่องจากสามารถสลับลำดับการวาดสถานะและเส้นทางการเปลี่ยนสถานะได้จากการสลับแถวและหลักในเมทริกซ์นั้นเอง กระบวนการวาดแผนผังสถานะมีขั้นตอนดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. กำหนดและวาดสถานะเริ่มต้น
2. ค้นหาสถานะปลายทางจากเมทริกซ์เชื่อมต่อ เพื่อคัดแยกการวาดเส้นทางไปยังสถานะที่จะถูกสร้างใหม่ ออกจากการวาดเส้นทางไปยังสถานะที่มีอยู่เดิม โดยแยกเก็บสถานะไว้ในตัวแปรอาร์เรย์สถานะใหม่ และตัวแปรอาร์เรย์สถานะเก่า ตามลำดับ
3. วาดเส้นทาง และสถานะใหม่ทั้งหมด ดังแสดงในรูปที่ 4.11(ก) โดยอ่านจากตัวแปรอาร์เรย์สถานะใหม่
4. วาดเส้นทางไปยังสถานะที่มีอยู่เดิมทั้งหมด ดังแสดงในรูปที่ 4.11(ข) โดยอ่านจากตัวแปรอาร์เรย์สถานะเก่า
5. สุดท้ายค้นหา และวาดเส้นทางวกกลับ ดังแสดงในรูปที่ 4.11(ค)



(ก) วาดเส้นทางและสถานะใหม่ (ข) วาดเส้นทางและสถานะที่มีอยู่เดิม (ค) วาดเส้นทางวกกลับ

รูปที่ 4.11 หลักการวาดแผนผังสถานะของสเตทแมชชีน

4.3.4 เครื่องมือทดสอบสเตทแมชชีน

ผู้ใช้งานสามารถทำการทดสอบ และวิเคราะห์กลไกการทำงานของสเตทแมชชีนภายในระบบได้ด้วยเครื่องมือทดสอบสเตทแมชชีน ซึ่งสามารถเลือกวิธีทดสอบสเตทแมชชีนได้ 2 วิธี คือ ค้นหาเส้นทางที่ใช้เวลาน้อยที่สุดโดยการระบุสถานะต้นทางและปลายทางที่ต้องการค้นหา และแสดงลำดับการทำงาน หรือเส้นทางการเปลี่ยนแปลงของสเตทแมชชีนตามลำดับเหตุการณ์ที่ถูกเลือก

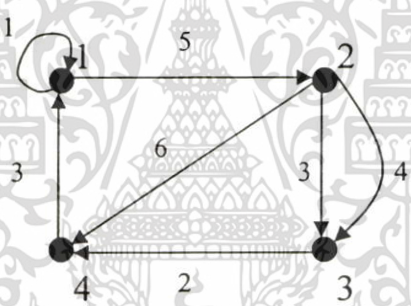
4.3.4.1 การค้นหาเส้นทางที่ใช้เวลาน้อยที่สุด

ผู้ใช้งานสามารถเรียกใช้คำสั่ง การค้นหาเส้นทางที่ใช้เวลาน้อยที่สุดของสเตทแมชชีนได้ในขณะที่โปรแกรมได้แสดงแผนผังสถานะสเตทแมชชีนเรียบร้อยแล้ว และผู้ใช้งานจำเป็นต้องป้อน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สถานะต้นทางและสถานะปลายทางที่ต้องการค้นหาให้กับโปรแกรมก่อนทำการค้นหา เมื่อผู้ใช้งานทำการค้นหา และโปรแกรมค้นพบเส้นทางที่ใช้เวลาน้อยที่สุดแล้ว โปรแกรมจะแสดงผลในรูปแบบของลำดับสถานะที่ใช้เวลาน้อยที่สุดพร้อมกับเวลาที่ใช้ทั้งหมด ผลลัพธ์ที่ได้จากการค้นหาเส้นทางคือ ทำให้ผู้ใช้งานทราบถึงเส้นทางและเวลาที่สั้นที่สุดในการทำงานที่เริ่มต้นจากสถานะหนึ่งและไปสิ้นสุดลงที่อีกสถานะหนึ่ง อัลกอลิทึมการค้นหาเส้นทางการเปลี่ยนสถานะที่ใช้เวลาน้อยสุดในสเตทแมชชีนนั้นได้ประยุกต์มาจากอัลกอลิทึมการหาเส้นทางของ Dijkstra โดยสมมุติให้ค่าหน่วงเวลาน้อยสุดของเหตุการณ์ในแต่ละเส้นทางการเปลี่ยนสถานะ เป็นค่าระยะทางระหว่างจุดใดใดบนกราฟ ตัวอย่างที่ 4.5 แสดงการหาเส้นทางการเปลี่ยนสถานะที่ใช้เวลาน้อยสุดของสเตทแมชชีนจากอัลกอลิทึมการหาเส้นทางที่สั้นที่สุดของ Dijkstra

ตัวอย่างที่ 4.5 สมมุติให้สเตทแมชชีนในตัวอย่างที่ 4.4 มีค่าหน่วงเวลาน้อยสุดในแต่ละเหตุการณ์มีค่าเป็น $e1=5$, $e2=3$, $e3=4$, $e4=6$, $e5=2$ $e6=3$ และ $e7=1$ ดังนั้นจากสเตทแมชชีนในรูปที่ 4.10 สามารถแสดงเป็นกราฟเวลาหน่วงน้อยสุดได้ดังรูปที่ 4.12



รูปที่ 4.12 กราฟแสดงค่าหน่วงเวลาน้อยสุด

จากกราฟในรูปที่ 4.12 สามารถแสดงเป็นเมทริกซ์หน่วงเวลา T ได้ คือ

$$
T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}
\begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 1 & 5 & 0 & 0 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 2 \\ 3 & 0 & 0 & 0 \end{bmatrix} \end{array}
$$

เมทริกซ์หน่วงเวลา T แสดงถึง ระยะเวลาน้อยสุดที่เหตุการณ์นั้นจะเกิดขึ้นได้ เช่น การเปลี่ยนสถานะจากสถานะ 2 ไปยังสถานะ 3 ใช้เวลาน้อยสุดคือ 3 หน่วยเวลา และเมื่อได้เมทริกซ์หน่วงเวลาแล้วสามารถค้นหาเส้นทางการเปลี่ยนสถานะที่ใช้ระยะเวลาน้อยที่สุดจากอัลกอริทึมของ Dijkstra ต่อไปได้ เช่น เมื่อค้นระยะเวลาน้อยที่สุดของการเปลี่ยนสถานะจากสถานะ 1 ไปยังสถานะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4 ได้ผลลัพธ์เป็น เส้นทางการเปลี่ยนสถานะคือ จากสถานะ 1 เกิดเหตุการณ์ *e*1 เปลี่ยนเป็นสถานะ 2 เกิดเหตุการณ์ *e*2 เปลี่ยนเป็นสถานะ 3 เกิดเหตุการณ์ *e*5 เปลี่ยนเป็นสถานะ 4 และใช้ระยะเวลาการเกิดเหตุการณ์ทั้งหมดคือ 5 + 3 + 2 = 10 หน่วยเวลา

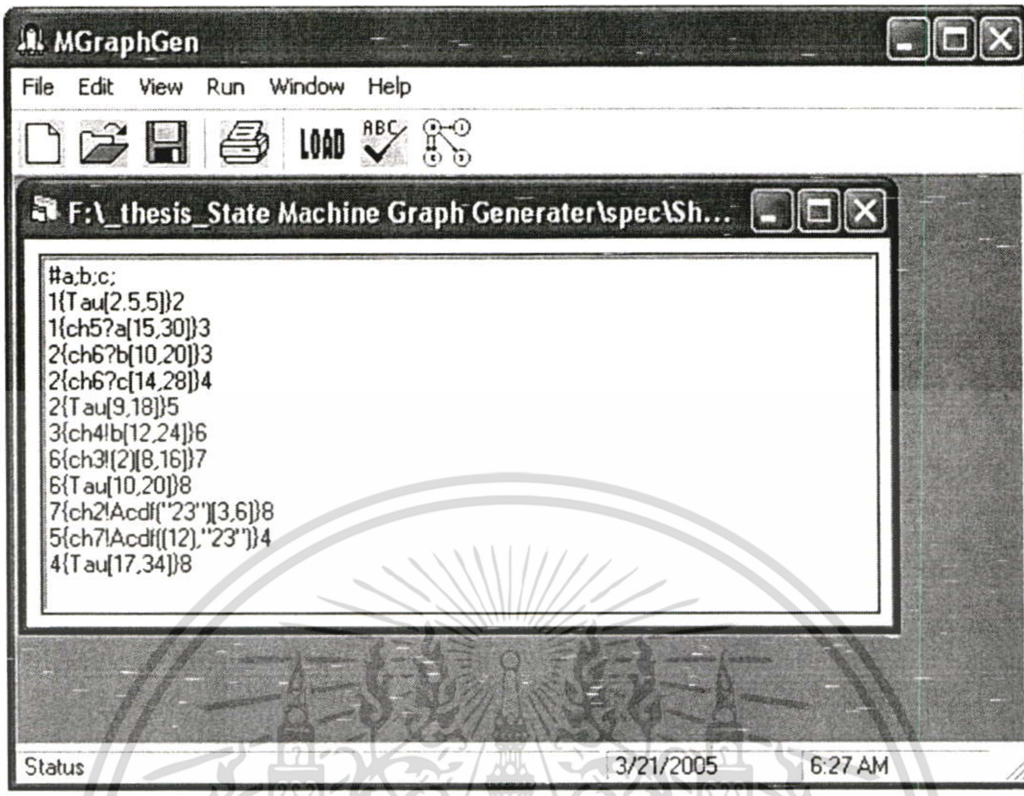
4.3.4.2 การแสดงลำดับการทำงานและเส้นทางการเปลี่ยนแปลงของสเตทแมชชีน

วิธีการทดสอบนี้ต้องกระทำหลังจากที่โปรแกรมได้แสดงแผนผังสถานะของสเตทแมชชีนเรียบร้อยแล้ว ผู้ใช้งานต้องทำการเลือกเส้นที่ต้องการวิเคราะห์ก่อนโดยการเลือกลำดับเหตุการณ์ที่ต้องการจากแผนผังสถานะของสเตทแมชชีน โปรแกรมจะแสดงกล่องข้อความให้กรอกค่าหน่วงเวลาการเกิดเหตุการณ์ทุกครั้งที่มีการเลือกเหตุการณ์ หลังจากเลือกลำดับเหตุการณ์ได้แล้วผู้ใช้งานสามารถเลือกแสดงลำดับการทำงาน หรือเส้นทางการเปลี่ยนแปลงของสเตทแมชชีนเพื่อวิเคราะห์ต่อไปได้

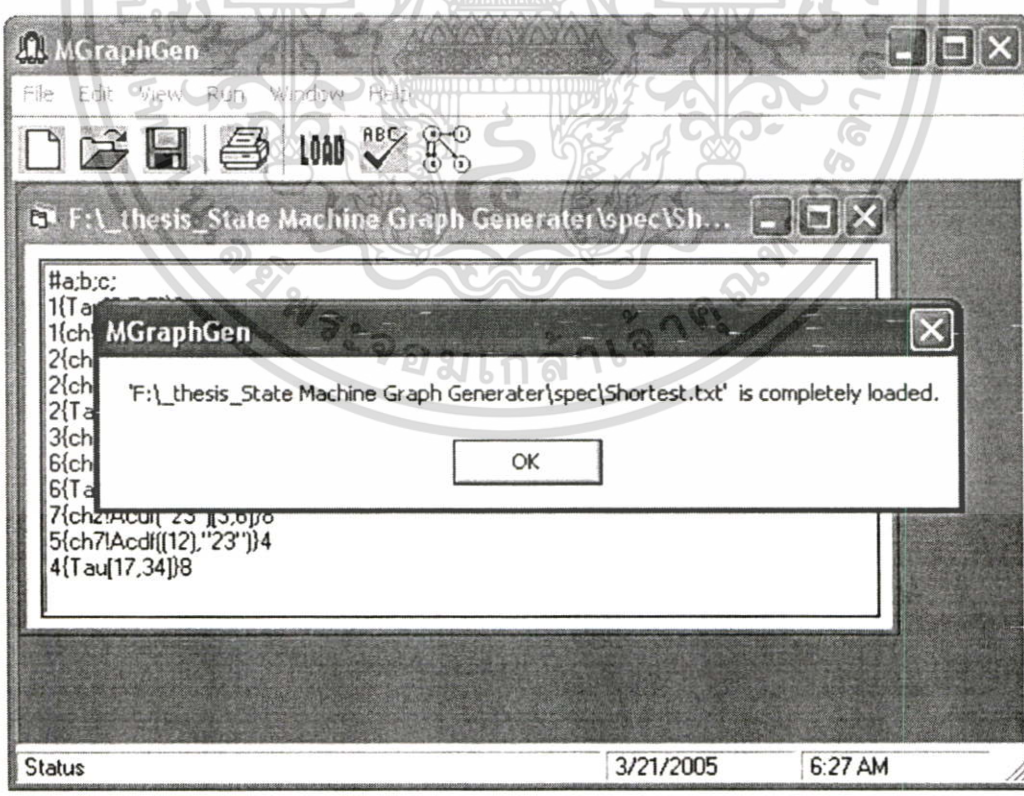
ตัวอย่างที่ 4.6 ตัวอย่างขั้นตอนการทดสอบสเตทแมชชีน ด้วยโปรแกรม MGraphGen มีดังนี้

1. เปิดแฟ้มข้อกำหนดสเตทแมชชีน ด้วยคำสั่ง File -> Open โปรแกรมจะแสดงข้อกำหนดต่างๆ ภายในแฟ้มข้อกำหนดสเตทแมชชีน ซึ่งในหน้าต่างนี้ผู้ใช้งานสามารถทำการแก้ไขและเก็บบันทึกได้ (ดูรูปที่ 4.13)
2. โหลดข้อกำหนดสเตทแมชชีนด้วยคำสั่ง Run -> Load (ดูรูปที่ 4.14)
3. แสดงแผนผังสถานะของสเตทแมชชีนด้วยคำสั่ง Run -> Graph (ดูรูปที่ 4.15)
4. เมื่อต้องการค้นหาเส้นทางที่ใช้เวลาน้อยที่สุด ให้กรอกสถานะต้นทางและสถานะปลายทางในช่อง From และ To จากนั้นให้คลิกที่ปุ่ม Search เพื่อค้นหา โปรแกรมจะแสดงเส้นทางที่ใช้เวลาน้อยที่สุดพร้อมกับเวลาทั้งหมดที่ใช้ในเส้นทางนั้น (ดูรูปที่ 4.16)
5. เมื่อต้องการแสดงลำดับการทำงานและเส้นทางการเปลี่ยนแปลงของสเตทแมชชีน ให้คลิกเลือกลำดับเหตุการณ์ที่ต้องการ โดยเมื่อคลิกที่เหตุการณ์แต่ละครั้งโปรแกรมจะแสดงกล่องใส่ข้อความเพื่อกรอกเวลาที่เหตุการณ์นั้นเกิดขึ้น(ดูรูปที่ 4.17) ให้กรอกค่าเวลาภายในช่วงเวลาหน่วงน้อยสุด และเวลาหน่วงมากสุดแล้วคลิกที่ปุ่ม OK เมื่อเลือกลำดับเหตุการณ์เสร็จแล้ว ถ้าต้องการแสดงลำดับการทำงานให้คลิกที่ปุ่ม Computation (ดูรูปที่ 4.18) แต่ถ้าต้องการแสดงเส้นทางการเปลี่ยนแปลงให้คลิกที่ปุ่ม Trace (ดูรูปที่ 4.19)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

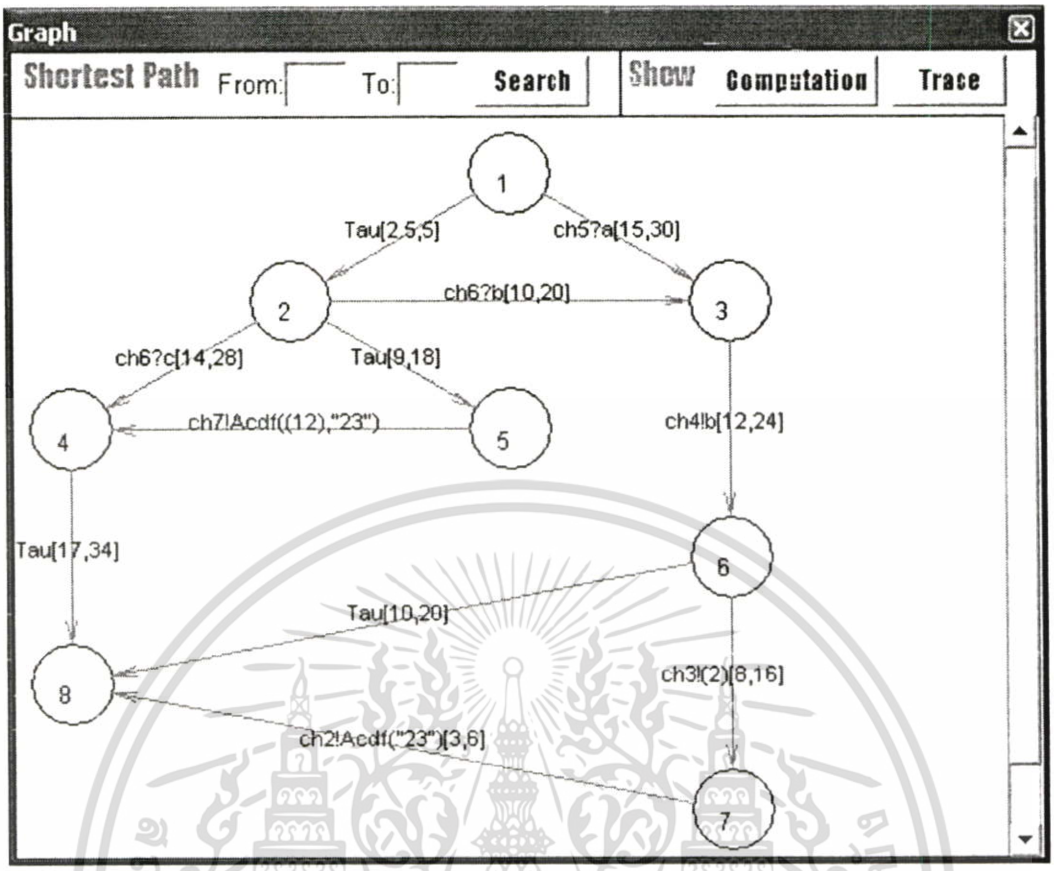


รูปที่ 4.13 ข้อกำหนดภายในแฟ้มข้อกำหนดสเตทแมชชีน

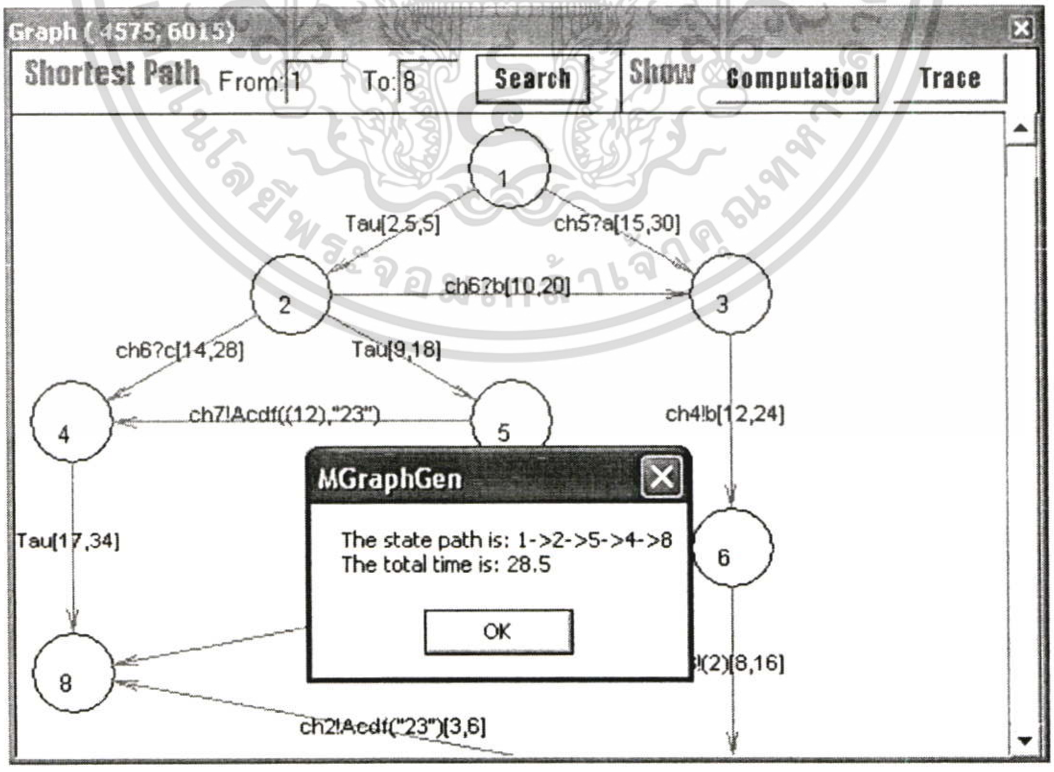


รูปที่ 4.14 กล่องข้อความแสดงแฟ้มข้อกำหนดถูกป้อนเรียบร้อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

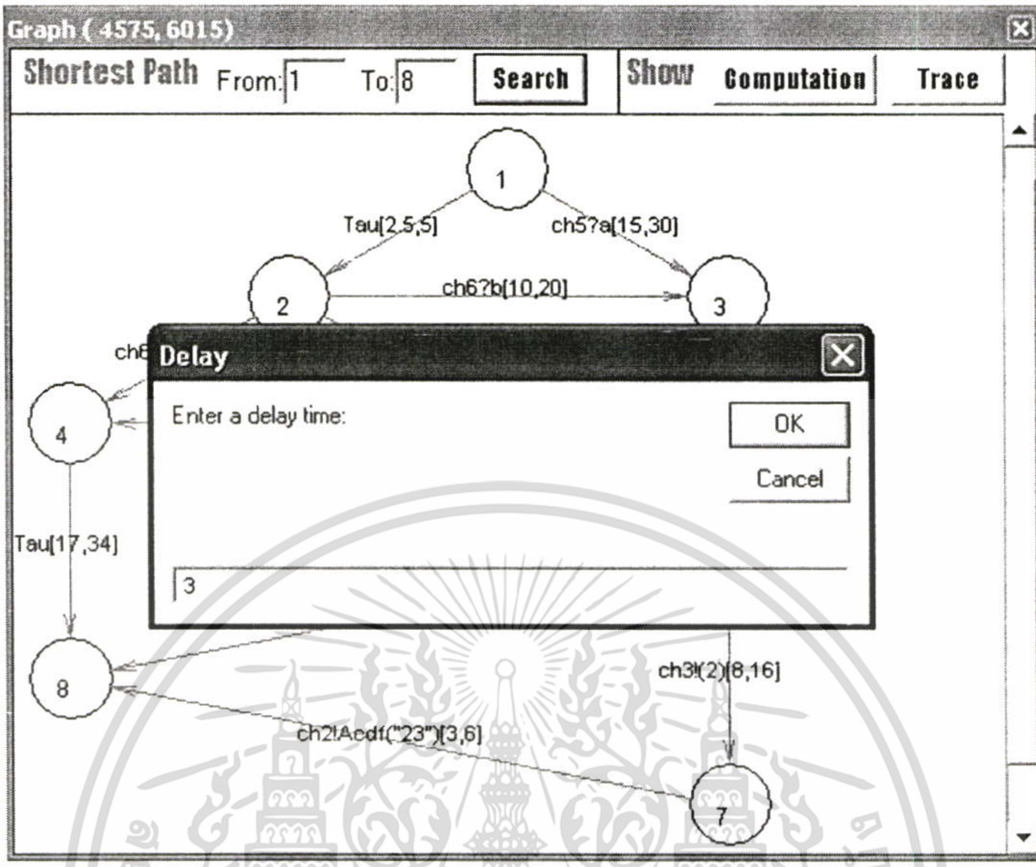


รูปที่ 4.15 แผนผังสถานะสเตทแมชชีน เมื่อใช้คำสั่งแสดงกราฟ

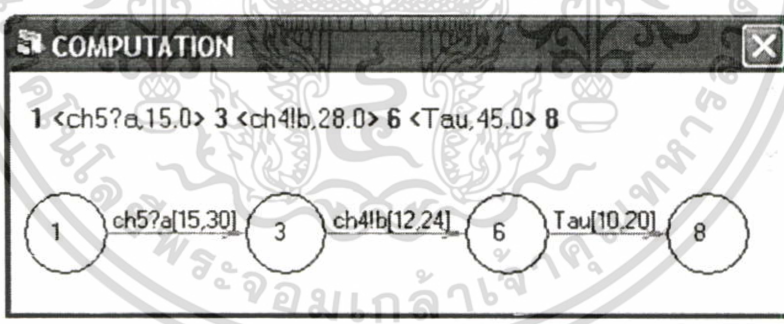


รูปที่ 4.16 ผลลัพธ์จากการค้นหาเส้นทางที่ใช้เวลาน้อยที่สุดจากสถานะ 1 ไปยังสถานะ 8

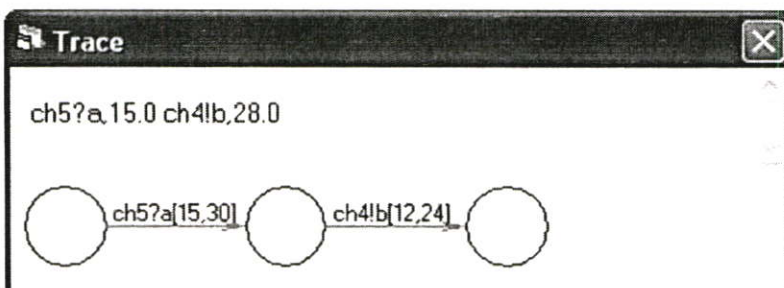
เอกสารนี้เป็นเอกสารที่เผยแพร่เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้เป็นประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.17 การกรอกเวลาหน่วงสำหรับเหตุการณ์ที่ผู้ใช้งานต้องการเลือก



รูปที่ 4.18 ลำดับการทำงานของลำดับเหตุการณ์ที่ถูกเลือก



รูปที่ 4.19 เส้นทางการเปลี่ยนแปลงของลำดับเหตุการณ์ที่ถูกเลือก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การจำลองการทำงานของระบบ

การจำลองการทำงานของระบบ มีความจำเป็นในการตรวจสอบพฤติกรรมการทำงานร่วมกันระหว่างสเตทแมชชีนภายในระบบ ในหัวข้อนี้จะกล่าวถึงวิธีการจำลองการทำงานของระบบการทำงานร่วมกันที่มีสเตทแมชชีนตั้งแต่ 2 ตัวขึ้นไป โปรแกรม MgraphGen สามารถจำลองการทำงานของระบบได้ด้วยเครื่องมือจำลองการทำงาน โดยการจำลองการทำงานจะขึ้นอยู่กับรูปแบบของข้อกำหนดระบบ และข้อกำหนดสเตทแมชชีนภายในระบบแต่ละตัว โดยจะแสดงผลการจำลองการทำงานในรูปแบบของลำดับการทำงานระหว่างกันของสเตทแมชชีน ทำให้ผู้ใช้งานสามารถตรวจสอบพฤติกรรมการทำงานร่วมกันภายในระบบการสื่อสารที่ออกแบบได้เป็นอย่างดี

ความสามารถของเครื่องมือจำลองการทำงานของระบบ มีดังนี้

- สามารถจำลองผลการทำงาน โดยอัตโนมัติ ตามข้อกำหนดได้
- ควบคุมการจำลองการทำงาน ได้ด้วยคำสั่งควบคุมเช่น เริ่มการจำลอง ยุติการจำลองชั่วขณะ และยุติการจำลอง
- สามารถเลือกปรับความเร็วการทำงานของระบบได้
- สามารถป้อนข้อมูลให้แก่ระบบ(จากสิ่งแวดล้อมภายนอก)ได้
- สามารถแสดงการจำลองการทำงานในรูปแบบลำดับการทำงานระหว่างกันของสเตทแมชชีนภายในระบบได้

ข้อจำกัดของเครื่องมือช่วยทดสอบสเตทแมชชีนมีดังนี้

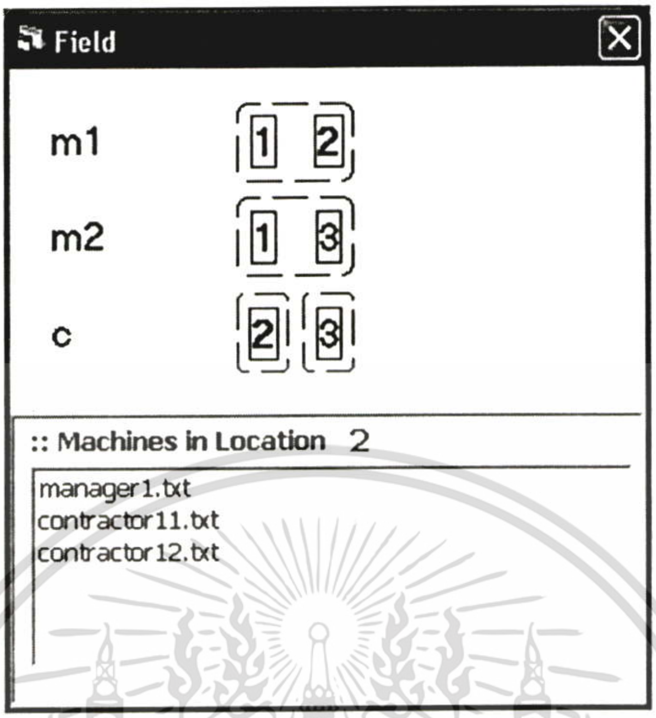
- ระบบจะต้องประกอบด้วยสเตทแมชชีนไม่เกิน 7 ตัว
- กำหนดพื้นที่บริการได้สูงสุด 18 พื้นที่
- เครือข่ายช่องสัญญาณร่วมหนึ่งครอบคลุมพื้นที่ได้ไม่เกิน 6 พื้นที่

เครื่องมือจำลองการทำงานของระบบประกอบไปด้วย

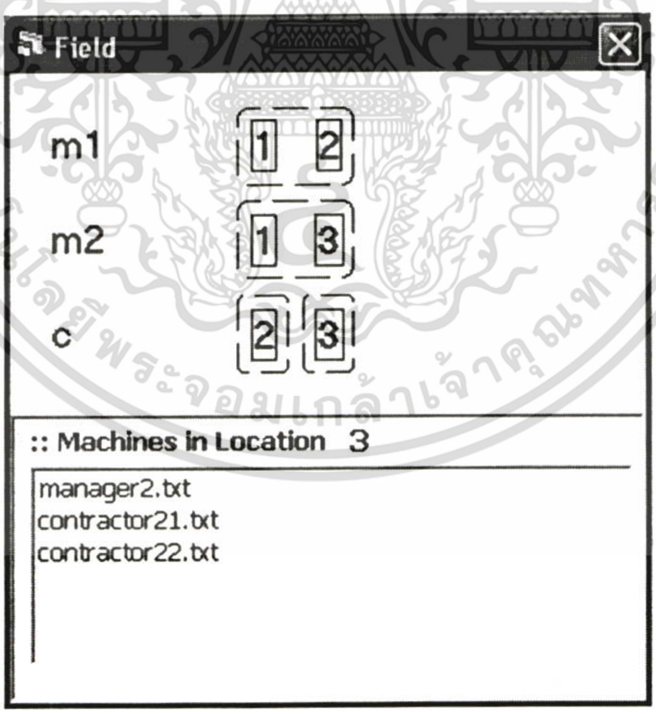
4.4.1 ตัวแสดงเครือข่ายช่องสัญญาณร่วม

ตัวแสดงเครือข่ายช่องสัญญาณร่วมแสดงขอบเขตช่องสัญญาณร่วมทั้งหมดที่ใช้ในระบบในรูปแบบของแผนภาพ และแสดงสเตทแมชชีนที่อยู่ในแต่ละพื้นที่ได้โดยการคลิกเลือกพื้นที่ที่ต้องการในแผนภาพนี้ จากนั้นผู้ใช้งานยังสามารถคลิกเลือกที่สเตทแมชชีนแต่ละตัวเพื่อแสดงแผนผังสถานะสเตทแมชชีนนั้นๆ ได้อีกด้วย รูปที่ 4.19 และ 4.20 แสดงตัวอย่างของตัวแสดงเครือข่ายช่องสัญญาณร่วมของระบบการให้งานตามแฟ้มข้อกำหนดระบบในตัวอย่างที่ 4.3 เมื่อผู้ใช้ต้องการเลือกดูสเตทแมชชีนในพื้นที่ 2 และพื้นที่ 3 ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.20 ตัวแสดงเครือข่ายช่องสัญญาณร่วม เมื่อผู้ใช้เลือกดูสเตทแมชชีนในพื้นที่ 2



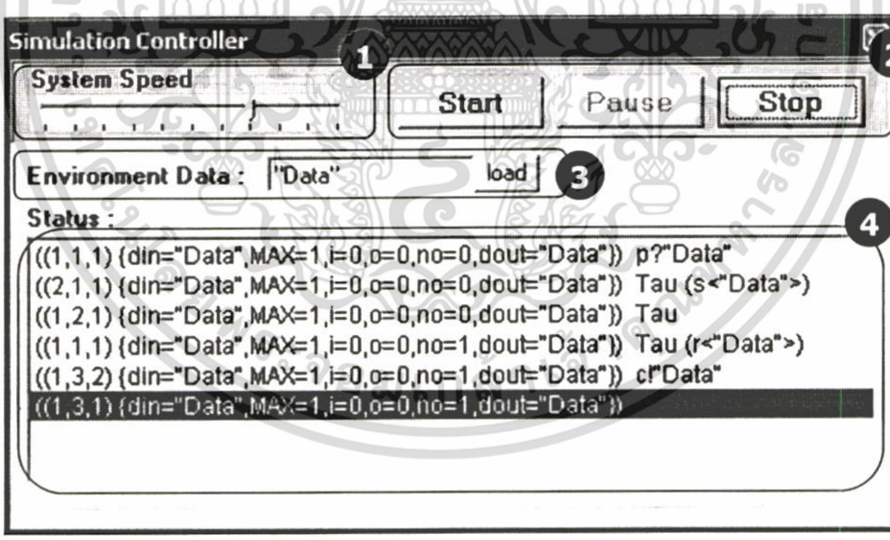
รูปที่ 4.21 ตัวแสดงเครือข่ายช่องสัญญาณร่วม เมื่อผู้ใช้เลือกดูสเตทแมชชีนในพื้นที่ 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4.2 ตัวควบคุมการจำลอง

ตัวควบคุมการจำลอง มีหน้าที่ในการควบคุมการจำลองการทำงาน และแสดงผลการจำลองในรูปแบบลำดับการทำงานของระบบ รูปที่ 4.22 แสดงตัวควบคุมการจำลอง ซึ่งประกอบไปด้วย

1. **แถบควบคุมความเร็วระบบ** ใช้ควบคุมความเร็วในการทำงานของระบบที่ทำการจำลองอยู่ โดยเมื่อปรับแถบให้เลื่อนไปทางขวาระบบจะทำงานเร็วขึ้น และเมื่อปรับแถบให้เลื่อนไปทางซ้ายระบบจะทำงานช้าลง
2. **ชุดคำสั่งควบคุมการจำลอง** ผู้ใช้งานสามารถควบคุมการจำลองการทำงานได้โดยผ่านชุดคำสั่งควบคุมการทำงาน ซึ่งชุดคำสั่งควบคุมการทำงานประกอบไปด้วยคำสั่ง เริ่มการจำลอง ยุติการจำลองชั่วขณะ และยุติการจำลอง
3. **กล่องป้อนค่าให้ระบบ** ใช้เพื่อป้อนข้อมูลจากสิ่งแวดล้อมเข้าสู่ระบบ โดยการกรอกค่าที่ต้องการลงในช่อง Environment Data แล้วคลิกที่ปุ่ม load โปรแกรมจะนำค่าที่ได้ป้อนให้ระบบโดยอัตโนมัติเมื่อระบบต้องการค่าจากสิ่งแวดล้อม
4. **กล่องแสดงผลการทำงาน** ใช้แสดงลำดับการทำงานของระบบทั้งหมดที่ทำการจำลองอยู่ โดยที่บรรทัดสุดท้ายของลำดับการทำงานจะเน้นสีเข้ม เพื่อบอกถึงสถานะปัจจุบันของการจำลองการทำงานของระบบ



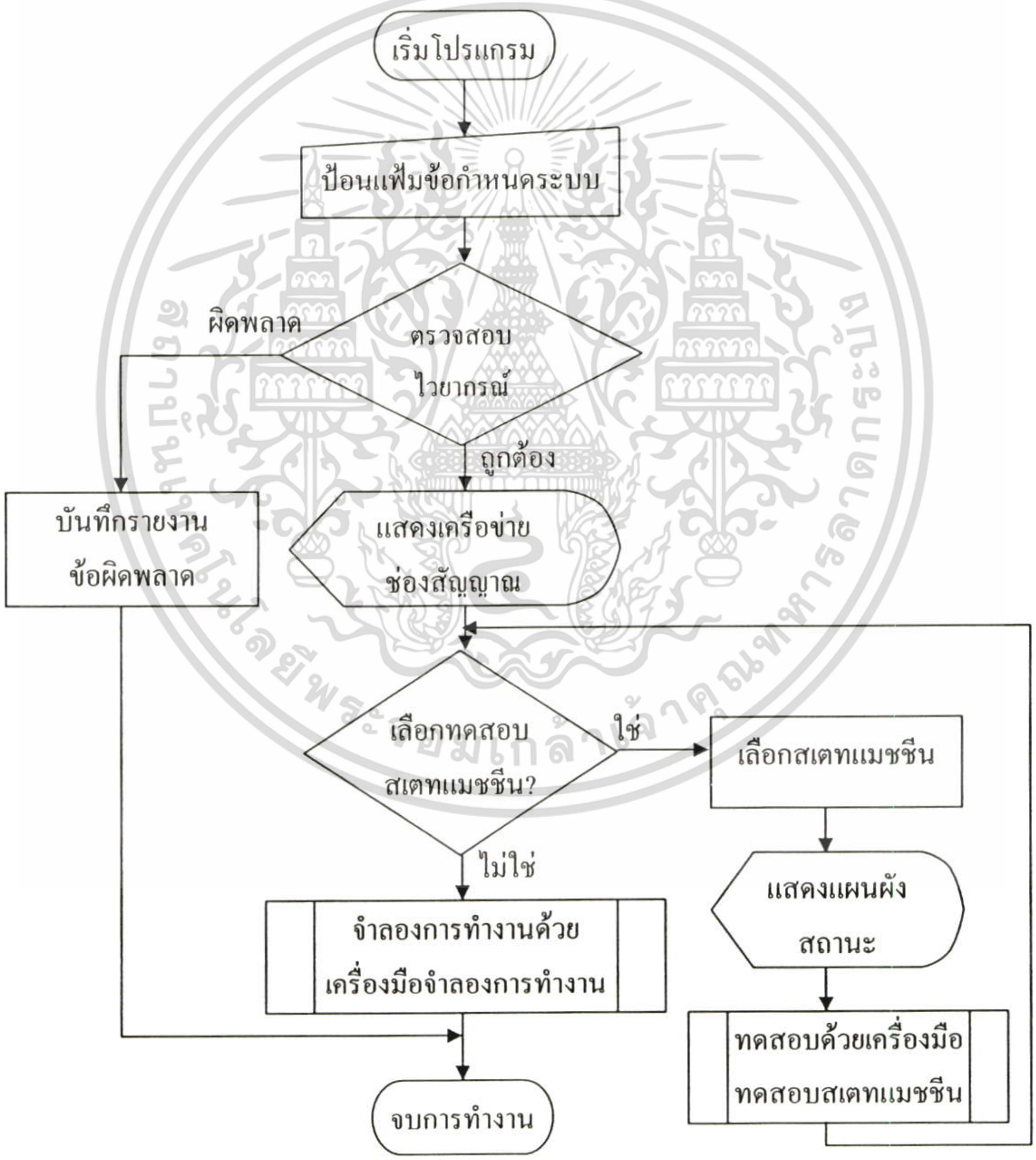
รูปที่ 4.22 ตัวควบคุมการจำลองและองค์ประกอบ

4.4.3 ตัวแสดงลำดับการสื่อสารระหว่างสเตทแมชชีน

ตัวแสดงลำดับการสื่อสารระหว่างสเตทแมชชีน เป็นหน้าต่างแสดงผลการทำงานในรูปแบบของลำดับการสื่อสารระหว่างสเตทแมชชีน ลำดับการสื่อสารที่ได้นี้มีประโยชน์ในการตรวจสอบพฤติกรรมการทำงานร่วมกันของสเตทแมชชีน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นตอนการจำลองการทำงานเริ่มจากการป้อนแฟ้มข้อกำหนดระบบให้แก่โปรแกรม ถ้าแฟ้มดังกล่าวไม่มีข้อผิดพลาดโปรแกรมจะแสดงหน้าต่างเครือข่ายช่องสัญญาณของระบบ จากหน้าต่างนี้ผู้ใช้งานสามารถเลือกให้โปรแกรมแสดงแผนผังสถานะของสเตทแมชชีน และทดสอบสเตทแมชชีนตัวใดภายในระบบได้ จากนั้นให้เลือกเครื่องมือการจำลองการทำงานของระบบเพื่อจำลองการทำงาน ซึ่งจะแสดงผลการจำลองการทำงานในรูปแบบของลำดับการทำงานของระบบและลำดับการสื่อสารระหว่างสเตทแมชชีน การทำงานของโปรแกรมเมื่อผู้ใช้งานต้องการจำลองการทำงานของระบบแสดงในรูปที่ 4.23



รูปที่ 4.23 ขั้นตอนการทำงานของโปรแกรมเมื่อต้องการจำลองการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างที่ 4.6 ตัวอย่างขั้นตอนการจำลองการทำงานของระบบการรับส่งข้อมูล(จากตัวอย่างที่ 2.1) โดยให้ค่า MAX = 1 ก่อนอื่นจำเป็นต้องสร้างแฟ้มข้อกำหนดสเตทแมชชีนที่จะใช้งานในระบบทั้งหมดเสียก่อน ซึ่งในระบบการรับส่งข้อมูลนี้ต้องประกอบด้วยแฟ้มข้อกำหนดสเตทแมชชีน ตัวส่ง ตัวกลาง และตัวรับ จากนั้นให้สร้างแฟ้มข้อกำหนดระบบ เพื่อระบุถึงพื้นที่การให้บริการและเครือข่ายสัญญาณ ตัวอย่างแฟ้มข้อกำหนดในระบบทั้งหมดแสดงได้ดังต่อไปนี้

แฟ้มข้อกำหนดสเตทแมชชีนของตัวส่ง (Sender.txt)

```
#din:int;
1{p?din}2
2{s!din}1
```

แฟ้มข้อกำหนดสเตทแมชชีนของตัวกลาง (Medium.txt)

```
#MAX:int=1; i:int=0; o:int=0; no:int=0; data[]:int
1{no<MAX-->s?data[i]}2
2{tau(no=no+1;i=(i+1)%MAX)}1
1{no>0-->r!data[o]}3
3{tau(no=no-1;o=(o+1)%MAX)}1
```

แฟ้มข้อกำหนดสเตทแมชชีนของตัวรับ (Receiver.txt)

```
#dout:int;
1{r?dout}2
2{c!dout}1
```

แฟ้มข้อกำหนดระบบของระบบการรับส่งข้อมูล (DataTrans.ini)

```
[Location]
1=Sender.txt,Medium.txt,Receiver.txt
[SharedChannel]
s={1}
r={1}
```

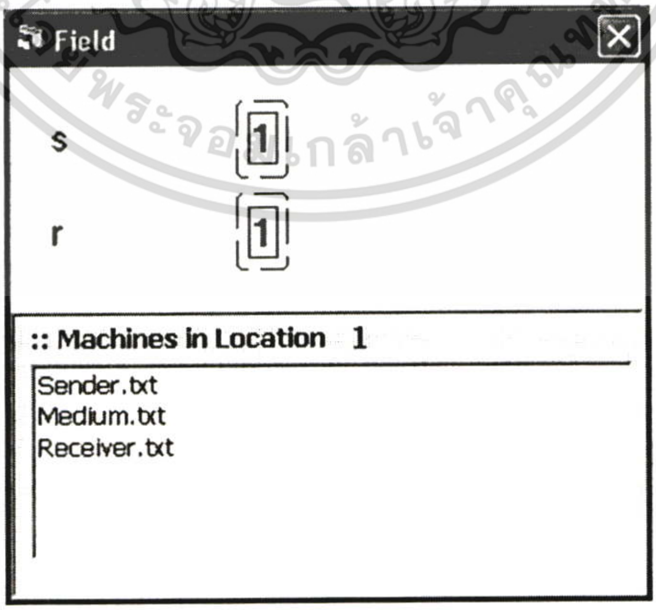
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อสร้างแฟ้มข้อกำหนดสเตทแมชชีนและแฟ้มข้อกำหนดระบบเสร็จแล้ว ผู้ใช้งานสามารถจำลองการทำงานของระบบได้โดยป้อนแฟ้มข้อกำหนดระบบ(DataTrans.ini) ถ้าแฟ้มข้อกำหนดระบบมีความถูกต้องโปรแกรมจะแสดงตัวแสดงเครือข่ายช่องสัญญาณร่วม ดังรูปที่ 4.24 จากหน้าต่างนี้ผู้ใช้งานสามารถเลือกดูแผนผังสถานะสเตทแมชชีนในพื้นที่ได้ รูปที่ 4.25, 4.26, และ 4.27 แสดงแผนผังสถานะสเตทแมชชีนตัวส่ง ตัวกลาง และตัวรับ ตามลำดับ จากแผนผังสถานะสเตทแมชชีนดังกล่าว ผู้ใช้งานสามารถทำการทดสอบด้วยเครื่องมือทดสอบสเตทแมชชีนแต่ละตัวได้ จากนั้นถ้าต้องการจำลองการทำงานของระบบให้คลิกเลือกคำสั่ง Run -> Simulation โปรแกรมจะเรียกตัวควบคุมการจำลองขึ้นมา จากขั้นตอนนี้ผู้ใช้งานสามารถเริ่มจำลอง หยุดการจำลองชั่วขณะ หรือยุติการจำลองการทำงานของระบบได้โดยคลิกที่ปุ่ม "Start" "Pause" และ "Stop" ตามลำดับ ถ้าผู้ใช้งานเริ่มทำการจำลองการทำงาน โปรแกรมจะเปิดหน้าต่างตัวแสดงลำดับการสื่อสารระหว่างสเตทแมชชีนขึ้นมา และแสดงลำดับการส่งข้อมูลระหว่างกันของสเตทแมชชีนทั้งสาม จากหน้าต่างนี้ผู้ใช้งานสามารถตรวจสอบพฤติกรรมการทำงานร่วมกันที่เป็นไปได้ของ ตัวส่ง ตัวกลาง และตัวรับ ในระบบการสื่อสารนี้ได้

รูปที่ 4.28 และ 4.29 แสดงตัวควบคุมการจำลอง และลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 3 หน่วยเวลา

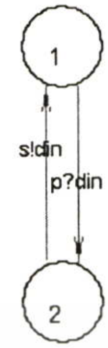
รูปที่ 4.30 และ 4.31 แสดงตัวควบคุมการจำลอง และลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 6 หน่วยเวลา

รูปที่ 4.32 และ 4.33 แสดงตัวควบคุมการจำลอง และลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 7 หน่วยเวลา

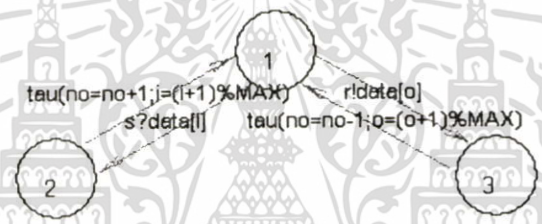


รูปที่ 4.24 ตัวอย่างเครือข่ายช่องสัญญาณของระบบการรับส่งข้อมูล

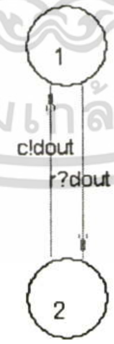
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.25 แผนผังสถานะสเตทแมชชีนของตัวส่ง

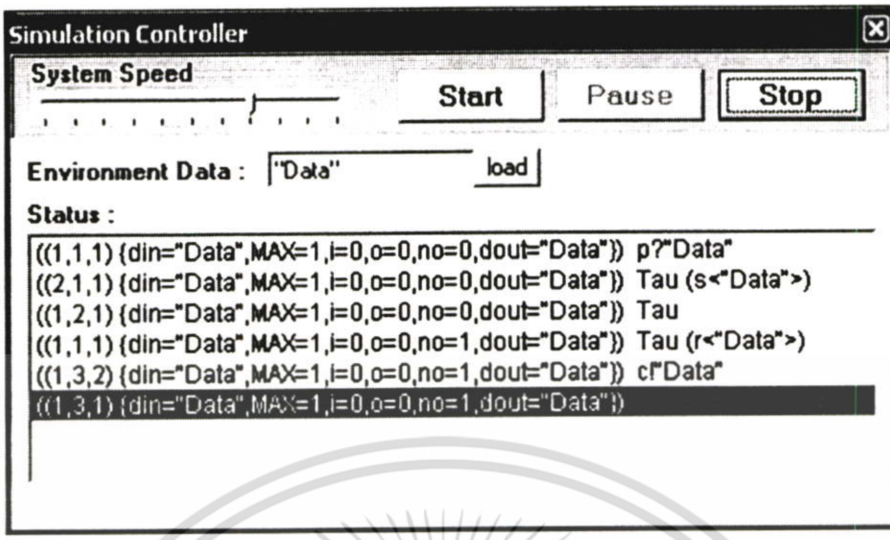


รูปที่ 4.26 แผนผังสถานะสเตทแมชชีนของตัวกลาง

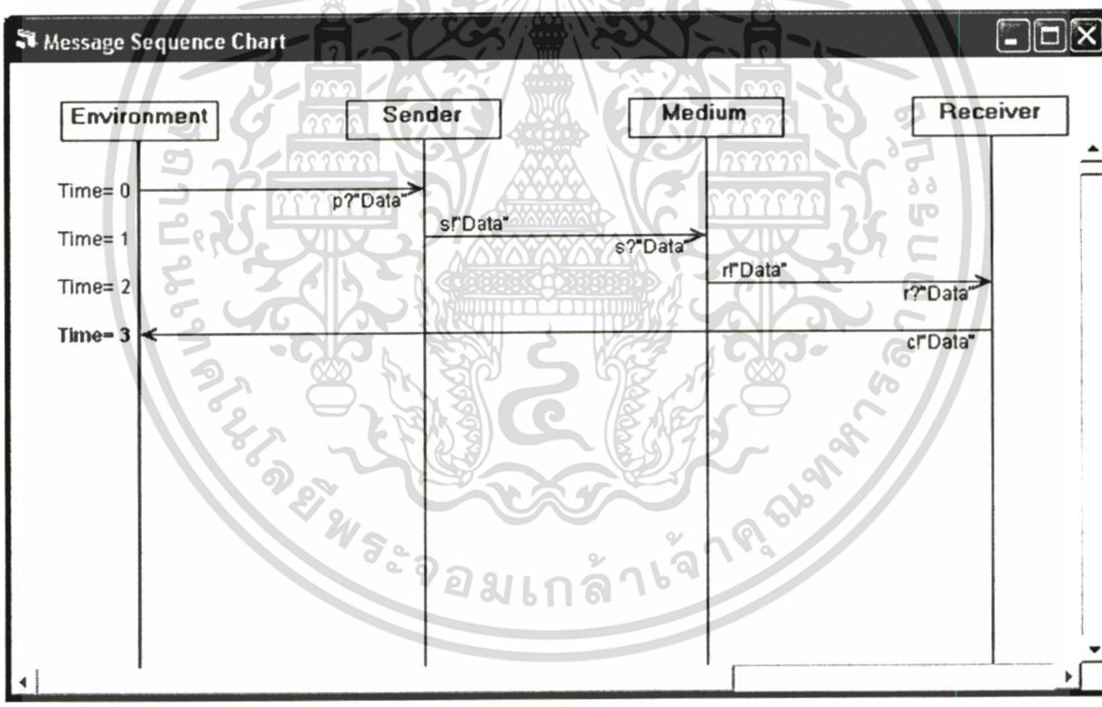


รูปที่ 4.27 แผนผังสถานะสเตทแมชชีนของรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

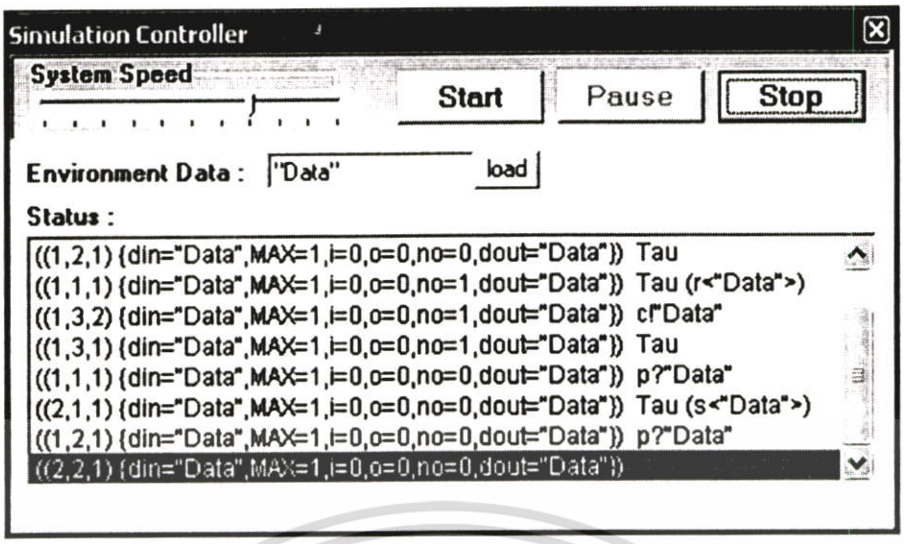


รูปที่ 4.28 ตัวควบคุมการจำลอง เมื่อเวลาผ่านไป 3 หน่วยเวลา

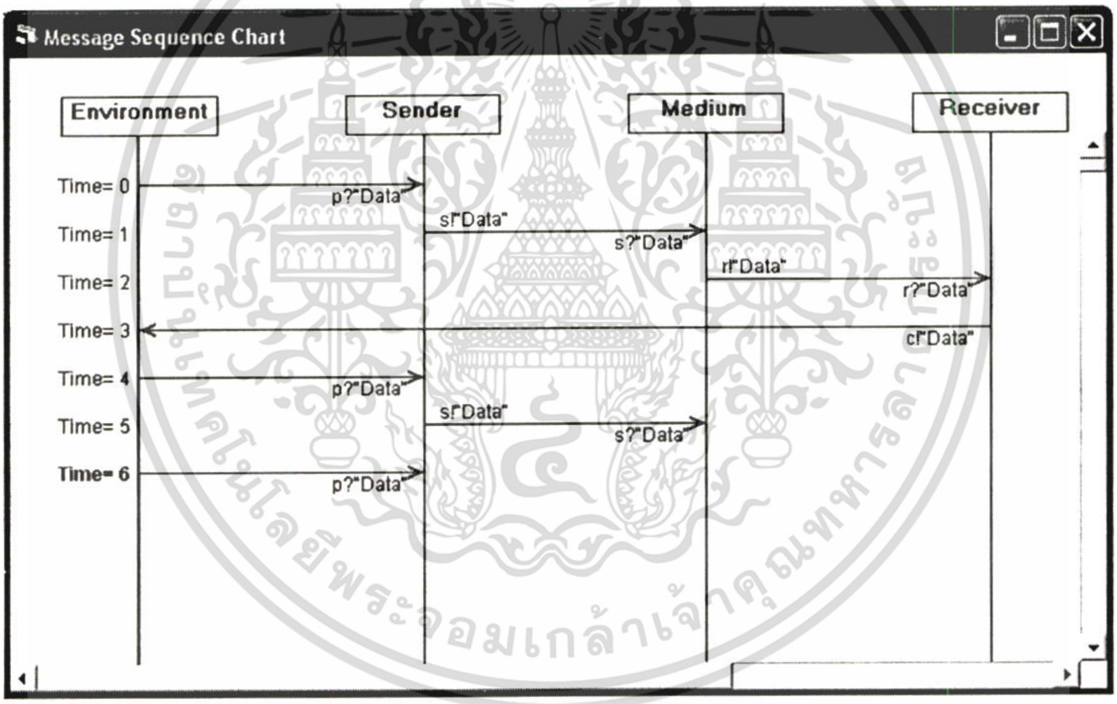


รูปที่ 4.29 ลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 3 หน่วยเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

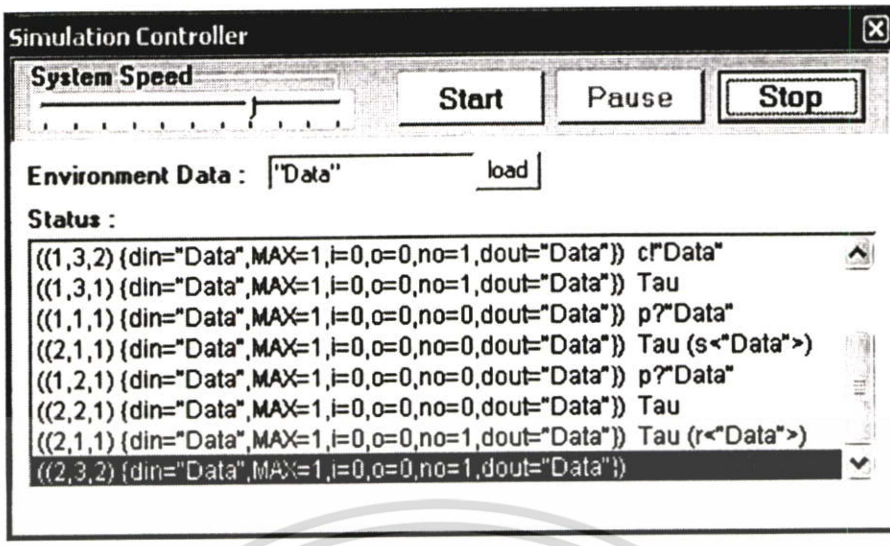


รูปที่ 4.30 ตัวควบคุมการจำลอง เมื่อเวลาผ่านไป 6 หน่วยเวลา

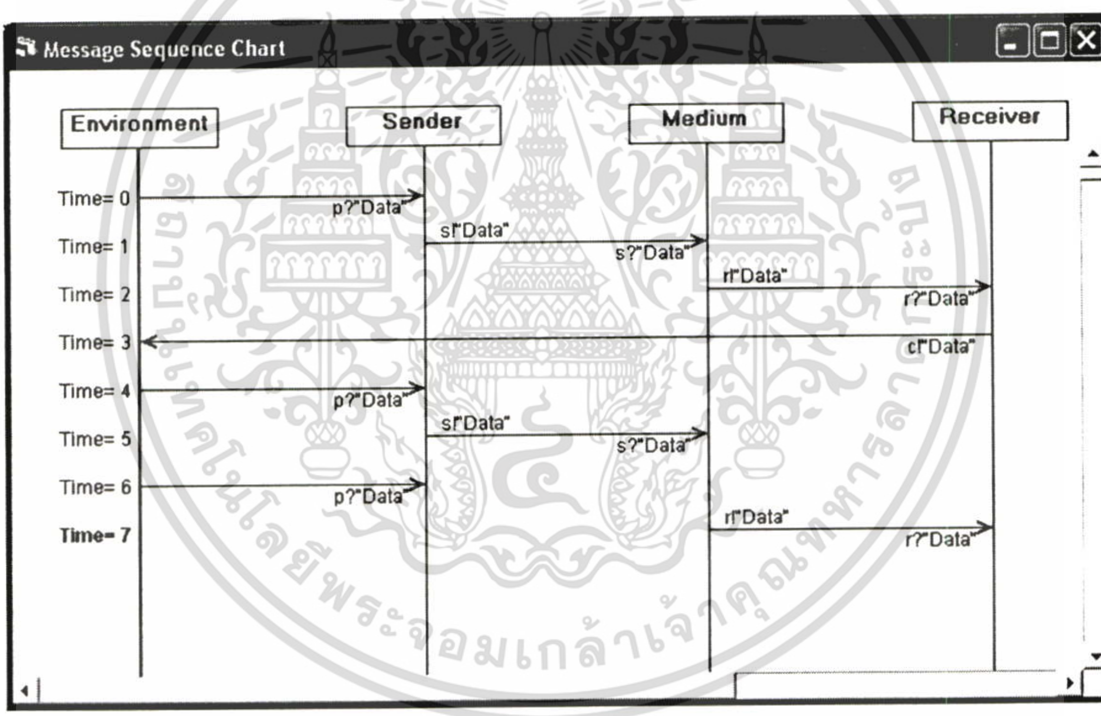


รูปที่ 4.31 ลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 6 หน่วยเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.32 ตัวควบคุมการจำลอง เมื่อเวลาผ่านไป 7 หน่วยเวลา



รูปที่ 4.33 ลำดับการสื่อสารระหว่างสเตทแมชชีน เมื่อเวลาผ่านไป 7 หน่วยเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปผลการวิจัย

5.1 บทสรุป

ในงานวิจัยนี้ได้นำเสนอวิธีการระบุข้อกำหนดของระบบการสื่อสาร โดยนำไฟไนท์สเตทแมชชีนมาเป็นแบบจำลองในการออกแบบ สาเหตุที่เลือกไฟไนท์สเตทแมชชีนมาเป็นแบบจำลองคือ

1. เนื่องจากระบบการสื่อสาร หรือ โปรโตคอลการสื่อสารนั้น มีลักษณะการทำงานทั่วไปเป็นกระบวนการลำดับขั้น สเตทแมชชีนก็มีลักษณะการทำงานและสามารถแสดงการทำงานเป็นกระบวนการลำดับขั้นได้เช่นกัน
2. เนื่องจากกระบวนการในระบบการสื่อสารประกอบ ด้วยกลไกภายในหลายส่วนทำงานร่วมกัน สเตทแมชชีนสามารถแทนกลไกการทำงานเหล่านั้นได้ เนื่องจากสเตทแมชชีนสามารถทำงานร่วมกันกับสเตทแมชชีนตัวอื่นๆได้
3. สเตทแมชชีนเป็นแบบจำลองที่แสดงพฤติกรรมภายในระบบได้เป็นอย่างดี
4. สามารถดัดแปลงพฤติกรรมภายในสเตทแมชชีนได้ ด้วยการนิยามข้อกำหนด
5. สเตทแมชชีนมีความสามารถในการปรับขยายระบบได้เป็นอย่างดี

เราได้นิยามไฟไนท์สเตทแมชชีนให้เป็นสเตทแมชชีนที่ประกอบด้วย 5 เซตหลัก คือ เซตของสถานะควบคุม เซตของตัวแปร เซตของเหตุการณ์ เซตของการเปลี่ยนสถานะ และเซตของสถานะเริ่มต้น การเปลี่ยนแปลงภายในสเตทแมชชีนเกิดขึ้นได้จากเหตุการณ์ซึ่งเป็นไปได้ 3 แบบ คือ เหตุการณ์ส่งข้อมูลออก เหตุการรับข้อมูลเข้า และเหตุการณ์ภายใน ด้วยเหตุการณ์รับ-ส่งข้อมูลเข้า-ออกทำให้สเตทแมชชีนแต่ละตัวสามารถติดต่อถึงกันได้โดยผ่านช่องสัญญาณร่วม ทำให้การออกแบบกลไกการทำงานภายในระบบ หรือการออกแบบสเตทแมชชีนสามารถแยกส่วนกันออกแบบได้ตามการทำงานของสเตทแมชชีนแต่ละตัว ส่งผลให้การปรับปรุงหรือเพิ่มเติมส่วนการทำงานของระบบสามารถทำได้โดยง่าย หลังจากได้นิยามสเตทแมชชีนใหม่แล้ว เพื่อให้สเตทแมชชีนที่ได้สามารถนำมาใช้ออกแบบระบบการสื่อสารได้ จำเป็นต้องเพิ่มข้อกำหนดเกี่ยวกับข้อจำกัดต่างๆ ที่มีอยู่ในระบบการสื่อสารเข้าไป ในงานวิจัยนี้ได้พิจารณาถึงข้อจำกัดที่จำเป็นในระบบการสื่อสารส่วนใหญ่ อันได้แก่ ข้อจำกัดทางด้านเวลา ข้อจำกัดทางภูมิศาสตร์ และข้อจำกัดการเคลื่อนที่ เมื่อเพิ่มข้อจำกัดเหล่านี้กับข้อกำหนดแล้ว ทำให้สามารถนำวิธีการออกแบบนี้มาใช้ออกแบบระบบการสื่อสารที่มีข้อจำกัดดังกล่าวได้ เช่น ระบบโปรโตคอล ระบบการสื่อสารแบบเคลื่อนที่ เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และเนื่องจากวิธีการดังกล่าวอาศัยแบบจำลองสเตทแมชชีนจึงทำให้กลไกภายในระบบการสื่อสารที่ได้ง่ายต่อการทำความเข้าใจ ออกแบบ และปรับปรุงต่อในอนาคต

เพื่อช่วยให้การออกแบบระบบการสื่อสารมีความสะดวก และรวดเร็ว มากขึ้น ในงานวิจัยนี้จึงได้พัฒนาโปรแกรมช่วยในการทดสอบและจำลองการทำงานของระบบมีชื่อว่า MGraphGen โดยโปรแกรมสามารถทำงานสอดคล้องกับข้อกำหนดตามวิธีที่ได้นำเสนอเป็นอย่างดี และประกอบไปด้วยเครื่องมือช่วยที่สำคัญคือ เครื่องมือทดสอบสเตทแมชชีน และเครื่องมือจำลองการทำงานของระบบ ด้วยเครื่องมือทดสอบสเตทแมชชีน ทำให้ผู้ใช้งานสามารถทดสอบความถูกต้องของข้อกำหนด ค้นหาเส้นทางที่สั้นที่สุดจากสถานะต้นทางไปยังสถานะปลายทางที่ต้องการ และเลือกแสดงลำดับการทำงานหรือเส้นทางการเปลี่ยนแปลงที่ต้องการวิเคราะห์ได้ และด้วยเครื่องมือจำลองการทำงานของระบบ ทำให้ผู้ใช้งานสามารถตรวจสอบพฤติกรรมการทำงานร่วมกันของสเตทแมชชีนภายในระบบได้

5.2 การประยุกต์ใช้งาน

จากวิธีการระบุข้อกำหนด และโปรแกรมช่วยที่นำเสนอมานี้ สามารถประยุกต์ใช้งานกับระบบการสื่อสารที่มีหรือไม่มีข้อจำกัดได้ ข้อจำกัดทางการสื่อสารที่สามารถนำมาใช้ออกแบบได้ได้แก่ ข้อจำกัดทางด้านเวลา ข้อจำกัดทางด้านภูมิศาสตร์ และข้อจำกัดทางด้านการเคลื่อนที่ ตัวอย่างระบบการสื่อสารที่สามารถประยุกต์ใช้ได้ เช่น

1. ระบบการประมวลผลตามเวลาจริง
2. ระบบโปรโตคอลการสื่อสาร
3. ระบบการสื่อสารแบบเคลื่อนที่

5.3 ข้อเสนอแนะ

เนื่องจากวิธีการระบุข้อกำหนดที่ได้เสนอมานี้ อาศัยแบบจำลองสเตทแมชชีน จึงทำให้สามารถแก้ไข และปรับแต่งพฤติกรรมภายในของแบบจำลองได้ เช่น การเพิ่มข้อกำหนด หรือเพิ่มข้อจำกัดทางด้านต่างๆ เป็นต้น ดังนั้นข้อจำกัดด้านอื่นๆ ในระบบการสื่อสารสามารถถูกเพิ่มให้เป็นข้อกำหนดใหม่ๆ ให้กับวิธีการออกแบบได้ ดังนั้นจะเห็นได้ว่าวิธีการออกแบบที่นำเสนอมานี้สามารถถูกปรับปรุงให้มีประสิทธิภาพในการออกแบบระบบการสื่อสารได้มากยิ่งขึ้นได้ ด้วยการเพิ่มนิยามวิธีการระบุข้อกำหนดตัวใหม่ลงไป อนึ่งโปรแกรมช่วยในการออกแบบควรถูกปรับปรุงให้มีประสิทธิภาพมากขึ้นตามไปด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] R. Milner. **Communication and Concurrency.** Prentice-Hall. 1989.

[2] C. A. R. Hoare. **Communication Sequential Process.** Prentice-Hall. 1985.

[3] R. Milner, J. Parrow and D.Walker. "A Calculus of Mobile Processes, Part I and Part II." Journal of Information and Computation, Vol.100, 1992. pp. 1-77.

[4] J. Davies and S. Schneider. "A Brief History of Timed CSP." Theoret ical Computer Science, Vol. 138, 1995.

[5] ISO/IEC. "Info rmation Technology—Enhancements to LOTOS (E-LOTOS)." ISO/IEC 15437, Aug. 2001.

[6] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. S. Trauring and M. Trakhtenbrot. "Statemate: a Working Environment for the Development of Complex Reactive System." IEEE Trans. Software Eng., Vol. 16, No. 4, 1990. pp. 403-414.

[7] D. Harel and A. Naamad. "The STATEMATE Semantics of Statecharts." ACM Trans. Software Engineering and Methodology, Vol. 5, No. 4, Oct. 1996. pp. 293-333.

[8] E. Battiston, F. D. Cindio and G. Mauli. "Modular Algebraic Nets to Specify Concurrent Syatems." IEEE Trans. Software Eng., Vol. 22, No. 10, 1996. pp. 689-705.

[9] K. Takahashi, G. Itabashi and Y. Kato. "Specification of a Mobile System based on Finite State Model." Proc. 1st International Conference on Parallel and Distributed Computing, Applications and Technologies, 2000. pp. 209-14.

[10]G. Itabashi, K. Takahashi,Y. Kato, T. Kato and B. B. Bista. "A Specification Mothod for Mobile and Concurrent Systems based on the Concept of Field and Finite State Machine, and Its Application." (in Japanese) Trans. Information Processing Society of Japan, Vol. 42, No. 12, Dec. 2001. pp. 3041-3053.

[11]A. Gabrielian and M. Franklin. "Multilevel Specification of Real-Time Systems." Commun. ACM, Vol. 34, No. 5, May 1991. pp. 50-60.

[12]F. Jahanian and A. K. Mok. "Modechart: A Specification Language for Real-Time Systems." IEEE Trans. Software Eng., Vol. 20, No. 12, Dec. 1994. pp. 933-947.

[13]A. Coen-Porisini, C. Ghezzi and R. A. Kemmerer. "Specification of Realtime Systems Using ASTRAL." IEEE Trans. Software Eng., Vol. 23, No. 9, Sept. 1997. pp. 572-598.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง (ต่อ)

[14]S. Yamane. "Specification and Verification of Real-time Concurrent Software." (in Japanese) Trans. Information Processing Society of Japan, Vol. 37, No. 2, Feb. 1996. pp. 188-203.

[15]R. Alur and D. Dill. "Automata for Modeling Real-Time Systems." Lecture Notes in Computer Science 443, 1990. pp. 322-335.

[16]A. C. Shaw. "Communication Real-Time State Machines." IEEE Trans. Software Eng., Vol. 18, No. 9, Sept. 1992. pp. 805-816.

[17]R. G. Smith. "The Contact Net Protocol: High-Level Communication and Control in a Distributed Problem Solver." IEEE Trans. Comp., Vol. C-29, No. 12, 1980. pp. 1104-1113.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

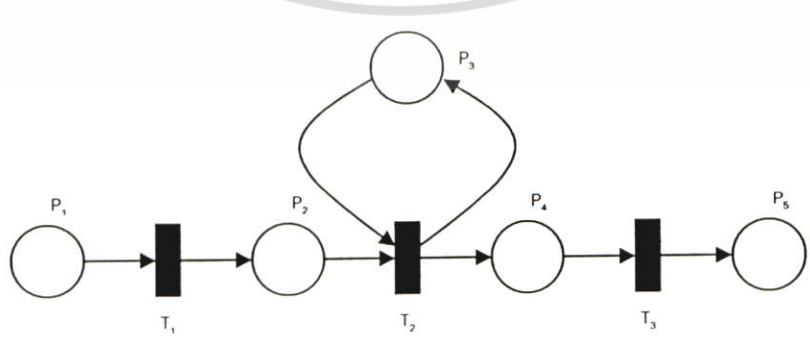
ภาคผนวก ก.

การเปรียบเทียบระหว่างไฟไนท์สเตทแมชชีนและเพทรี่เน็ทเพื่อนำมาใช้ออกแบบระบบสื่อสาร

บทความในส่วนนี้จะกล่าวถึงการเปรียบเทียบการนำแบบจำลองไฟไนท์สเตทแมชชีน และเพทรี่เน็ทมาใช้ออกแบบระบบสื่อสาร เพื่อช่วยในการตัดสินใจว่าแบบจำลองแบบไหนเหมาะสมที่สุดที่จะถูกนำมาใช้เป็นแบบจำลองในการออกแบบระบบสื่อสาร เนื่องจากข้อกำหนดในระบบการสื่อสารนั้นมีมากมาย จึงได้นำเฉพาะข้อกำหนดการสื่อสารที่สำคัญและจำเป็นมาเปรียบเทียบ อันได้แก่ การทำงานร่วมกัน การทำงานที่ต้องขึ้นกับเวลา และการทำงานที่ต้องขึ้นกับพื้นที่บริการ โดยการทำงานร่วมกันเป็นข้อกำหนดพื้นฐานที่จำเป็นต้องมีในระบบสื่อสาร การทำงานที่ต้องขึ้นกับเวลาเป็นข้อกำหนดที่จะถูกนำมาใช้ในระบบงานที่ขึ้นกับเวลา เช่น ระบบโปรโตคอลการสื่อสาร ระบบการประมวลผลในเวลาจริง เป็นต้น ส่วนการทำงานที่ต้องขึ้นกับพื้นที่บริการเป็นข้อกำหนดที่จะถูกนำมาใช้ในระบบที่มีการระบุพื้นที่การสื่อสาร เช่น ระบบเครือข่ายแบบไร้สาร เป็นต้น

แบบจำลอง

ส่วนประกอบพื้นฐานของเพทรี่เน็ทประกอบไปด้วย เซตสถานที่(place) เซตการเปลี่ยนแปลง(transition) เซตเส้นทาง(arc) และเซตโทเคน(token) โดยเซตของเส้นทางถูกนิยามจากกระบวนการเข้า และกระบวนการออก ซึ่งเส้นทางที่ได้จากกระบวนการทั้งสองนั้นจะบังคับทิศทางการเคลื่อนที่ของโทเคนจากสถานที่ไปยังการเปลี่ยนแปลง และจากการเปลี่ยนแปลงไปยังสถานที่ ตัวอย่างแบบจำลองเพทรี่เน็ทแสดงไว้ดังรูปที่ 1



รูปที่ 1 ตัวอย่างแบบจำลองเพทรี่เน็ท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

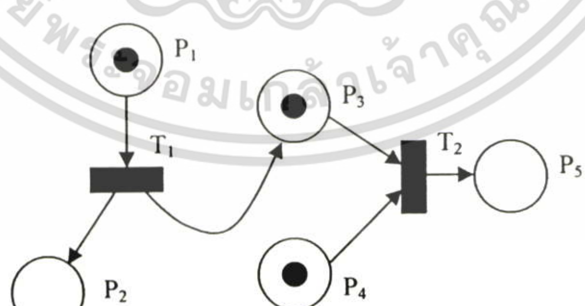
แบบจำลองเพทรี่เน็ทประกอบด้วย 4 องค์ประกอบหลักดังนี้คือ

- P เป็นเซตจำกัดของพื้นที่
- T เป็นเซตจำกัดของการเปลี่ยนแปลง
- $I : (P \times T)$ เป็นกระบวนการเข้าซึ่งนิยามเซตเส้นทางจากสถานที่ไปยังการเปลี่ยนแปลง
- $O : (P \times T)$ เป็นกระบวนการออกซึ่งนิยามเซตเส้นทางจากการเปลี่ยนแปลงไปยังสถานที่

เมื่อเปรียบเทียบแบบจำลองเพทรี่เน็ทและแบบจำลองไฟไนท์สเตทแมชชีน(ดังที่ได้กล่าวไว้แล้วในบทที่ 2) แบบจำลองทั้งสองมีหลักการทำงานที่ต่างกันคือ แบบจำลองเพทรี่เน็ทอาศัยหลักการเคลื่อนที่ของโทเคนเป็นเงื่อนไขของการเปลี่ยนแปลง และสถานะปัจจุบันในระบบต้องอาศัยตำแหน่งของโทเคนเป็นตัวบอก แต่สำหรับการเปลี่ยนแปลงที่เกิดขึ้นในแบบจำลองไฟไนท์สเตทแมชชีนใช้เหตุการณ์เป็นเงื่อนไขเป็นตัวกำหนด และสามารถสังเกตสาถานะปัจจุบันได้จากสถานะสุดท้ายของการเปลี่ยนแปลง ดังนั้นแบบจำลองไฟไนท์สเตทแมชชีนมีความง่ายต่อการกำหนดเงื่อนไขการเปลี่ยนแปลงมากกว่าเพราะสามารถระบุเงื่อนไขการเปลี่ยนแปลงที่เหตุการณ์ได้เลย ทั้งยังสามารถสังเกตสถานะปัจจุบันของระบบได้โดยง่าย และเนื่องจากความสามารถในการควบคุมการเปลี่ยนแปลงนี้เอง จึงทำให้แบบจำลองไฟไนท์สเตทแมชชีนมีความยืดหยุ่นในการระบุข้อกำหนดการเปลี่ยนแปลงได้มากกว่าแบบจำลองเพทรี่เน็ทอีกด้วย

การทำงานร่วมกัน

โดยพื้นฐานแล้วเพทรี่เน็ทถูกออกแบบมาให้สามารถ จำลองระบบการทำงานร่วมกันได้อยู่แล้ว โดยการใส่โทเคนในสถานที่ที่เหมาะสม ดังรูปที่ 2



รูปที่ 2 การทำงานร่วมกันของแบบจำลองเพทรี่เน็ท

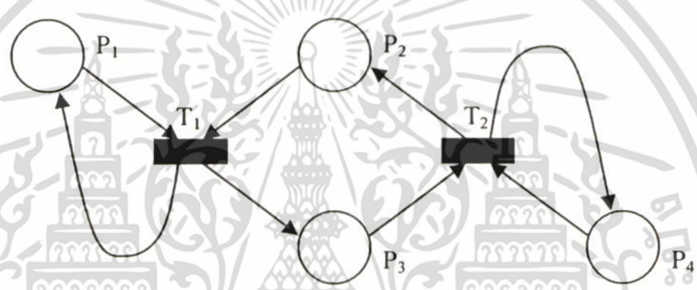
ไฟไนท์สเตทแมชชีนสามารถนำมาใช้ออกแบบกับการทำงานร่วมกันได้เช่นเดียวกัน แต่จำเป็นต้องกำหนดช่องสื่อสาร และพฤติกรรมร่วมตามหัวข้อที่ 2.3 ให้กับไฟไนท์สเตทแมชชีน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดังนั้นเพทรี้เน็ทมีความสะดวกกว่าในการนำมาออกแบบการทำงานร่วมกัน เนื่องจากไม่จำเป็นต้องมีการเพิ่มข้อกำหนดใดใดเหมือนไฟไนท์สเตทแมชชีน แต่การเพิ่มข้อกำหนดเองได้ของไฟไนท์สเตทแมชชีนก็มีประโยชน์ เพราะทำให้แบบจำลองที่ได้มีความยืดหยุ่นต่อการใช้งาน

การทำงานที่ต้องขึ้นกับเวลา

เพทรี้เน็ทสามารถนำมาใช้ออกแบบกับระบบงานที่ขึ้นกับเวลาได้ โดยการเพิ่มข้อกำหนดทางเวลาให้กับเพทรี้เน็ทพื้นฐาน ทำได้โดยการตั้งชื่อของสถานที่และการเปลี่ยนแปลงทุกตัว แล้วระบุค่าหน่วงเวลาการทำงานสูงสุด และต่ำสุดของทุกการเปลี่ยนแปลงที่ต้องการให้ขึ้นกับเวลา เรียกเพทรี้เน็ทแบบนี้ว่า เพทรี้เน็ทเวลา รูปที่ 3 แสดงตัวอย่างเพทรี้เน็ทเวลา โดยระบุค่าหน่วงเวลาการเปลี่ยนแปลงตามตาราง



การเปลี่ยนแปลง	สถานที่(เข้า)	ค่าหน่วงต่ำสุด	ค่าหน่วงสูงสุด
T1	P1, P2	0	P1 + 200
T2	P3, P4	P3 + 10	∞

รูปที่ 3 ตัวอย่างเพทรี้เน็ทเวลา

ส่วนไฟไนท์สเตทแมชชีนสามารถนำมาใช้ออกแบบกับ ระบบงานที่ขึ้นกับเวลาได้เช่นเดียวกันดังที่ได้กล่าวไว้แล้วในหัวข้อ 3.2

ทำงานที่ต้องขึ้นกับพื้นที่บริการ

การออกแบบระบบที่ต้องทำงานขึ้นกับพื้นที่บริการได้นั้น จำเป็นต้องมีการระบุพื้นที่การให้บริการ ช่องสัญญาณร่วม และขอบเขตของช่องสัญญาณเสียก่อน ซึ่งเพทรี้เน็ทนั้นทำได้ยาก เนื่องจากข้อจำกัดของการสร้างเงื่อนไขต่างๆ จำเป็นต้องสร้างจากเส้นทางการเคลื่อนที่ของโทเคน

ส่วนไฟไนท์สเตทแมชชีนสามารถนำมาใช้ออกแบบกับ ระบบที่ต้องทำงานขึ้นกับพื้นที่บริการได้ ดังที่ได้กล่าวไว้แล้วในหัวข้อ 3.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข.

บทความเกี่ยวข้องที่ได้รับการตีพิมพ์

[1] N. Maneerat, **P. Chaochanaphun**, R. Varakulsiripunth, K. Takahashi, and Y. Kato. "Specification and Verification of Communication Systems Based on State Machine Model with Time Constraint." The 3rd International Symposium on Communications and Information Technologies (ISCIT 2003), Songkhla, Thailand, 3-5 September, 2003. pp. 252-257.

[2] **ภูริชญ์ เชาว์ชนพันธุ์**, นพดล มณีรัตน์, รศ.ดร.รัตติกร วรากูลศิริพันธุ์. "วิธีการระบุข้อกำหนดสำหรับระบบการทำงานร่วมกันโดยใช้แบบจำลองสเตทแมชชีนและเครื่องมือช่วย." Published in Journal of King Mongkut's Institute of Technology Ladkrabang, Vol. 12, No. 1, April 2004. pp. 19-29.

[3] **P. Chaochanaphun**, N. Maneerat, R. Varakulsiripunth, K. Takahashi, and Y. Kato. "Specification Method for Constrained Concurrent System and Its Support Tool." The 1st ECTI Annual Conference (ECTI-CON2004), Pattaya, Thailand, 13-14 May, 2004. pp. 65-68.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Specification and Verification of Communication Systems Based on State Machine Model with Time Constraint

*Noppadol Maneerat**,†, Purich Chaochanaphun*, Ruttikorn Varakulsiripunth*, Kaoru Takahashi**, and Yasushi Kato***

*Faculty of Engineering, and Research Center for Communications and Information Technology (ReCCIT), and †Computer Research and Service Center, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand. (email: kvruttik@kmitl.ac.th)

** Sendai National College of Technology
4-16-1, Chuo, Ayashi, Aoba-ku, Sendai, 989-3128 Japan.
(email: kato@info.sendai-ct.ac.jp)

Abstract

An approach of design and analysis of communication systems is proposed by using a specification method based on the state machine model. The time constraint is introduced into the model in order to make it possible to specify the time sequential processing of communication systems. We have also constructed design and analysis tool of this method that is very helpful for investigating and verifying the specified system before real implementation.

1. Introduction

Most of communication procedure is a concurrent system that consists of a number of communicating entities. To specify such systems it needs to consider their concurrency together with communication. The *process calculi* such as CCS [1] and CSP [2] have been proposed as specification methods by modeling the entities as process represented by algebraic expressions. Though it can deal with the concurrency, it is an abstract one. Several specification methods based on a *state machine model* have been proposed [3~11] that are not as abstract as the *process calculi* and are closer to implementation.

This paper deals with specification methods based on state machine models by adding time constraint into the models. This makes it possible to specify sequential processing of real-time communication systems.

First, we introduce a single basic state machine and modeled it so that it can interact with the environment via channels. By defining a communication system as a collection of such state machines, the whole behavior of the communication system is given in terms of interactions among the state machines and their individual internal activities. Next, we introduce constrained state machines that can adapt to various specification purposes. Finally, we have developed a design and analysis tool based on our specification method. This tool can be used to analyze the communication system in state diagram form. By using this tool, the behaviors of a specified system can be investigated and verified before the implementation.

2. Basic state machine

A (*basic*) *state machine* has *variables* in addition to the usual (control) states. Each variable has its own *type* like *integer* or *string*. Let V be a set of variables, then a *valuation* of V is a mapping that associates to each variable of V, a value in its domain. We write $\mathbf{V}$ for the set of valuations of V.

A state machine can perform interaction with the environment or other state machines. For a state machine, this interaction appears as an input or output of data via a *channel*. We denote the set of channels accepting an input set I as C^{in} and the set of channels accepting an output set O as C^{out}. In the following definition, a data input/output of a state machine via a channel is formulated as an *event* that causes a state transition of state machine.

Definition 1: A *state machine* M is a 5-tuple:

$$M = \langle Q, V, \Sigma, T, \theta \rangle$$

- Q is a finite set of *control states*.
- V is a finite set of *variables*.
- Σ is a set of three *events*, those are Σ^{in} ($\Sigma^{in} \subseteq C^{in} \times I$), Σ^{out} ($\Sigma^{out} \subseteq C^{out} \times O$) and $\{\tau\}$ which are a set of input event from environment, a set of output event to the environment and a set of *internal* event, respectively.
- $T \subseteq (Q \times \mathbf{V}) \times \Sigma \times (Q \times \mathbf{V})$ is a *transition relation*.
- $\theta \subseteq Q \times \mathbf{V}$ is a set of *initial state*.

We say $s \in Q \times \mathbf{V}$ is a *state* of the state machine M. Since there may be a case where the domain of a variable is infinite, the number of states of M may not be finite.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

We write $s \xrightarrow{\sigma} s'$ for a $(s, \sigma, s') \in T$. If there exists a state s' such that $s \xrightarrow{\sigma} s'$ for a state s and an event σ, we write $s \xrightarrow{\sigma}$.

Notation 1: We use a usual state transition diagram for a visual representation of a state machine. If an explicit specification of variable values is needed, it is attached to the control state. A basic structure of state transition is represented as follows:

$$current\_state \xrightarrow{guard \rightarrow event} next\_state$$

The *guard* is a Boolean expression that qualifies the values of the variables related with the transition. If it is not necessary or it is always *true*, it can be omitted.

The *event* is written as follows if it is an input.

channel_name?*variable_name* "value"

This represents that the state machine inputs data with the value specified by "value" via the channel specified by *channel_name*, and assigns it to the variable specified by *variable_name*. If data is structured, for example, a protocol message whose *type* is defined and which has some parameters, it is written as follows.

channel_name?(*parameter*, ..., *parameter*)

Each *parameter* is represented as the above described *variable_name*:*type* "value".

The *event* is written as follows if it is an output.

channel_name!*expression*

This represents that the state machine outputs data with the value obtained by interpreting the specified *expression* via the channel specified by *channel_name*. For a structured data, its form conforms to the above description as follows.

channel_name!(*parameter*, ..., *parameter*)

An expression is written in each *parameter* part.

If the *event* is internal, the processing statements that may change the variable should be written. We use the symbol τ to represent the internal event.

Example 1: Figure 1 shows a notation example of state machines for data transmission in a communication system in which Sender transmits data to Receiver via Medium. The control states of Sender are *S*0 and *S*1. Sender inputs data into the variable *din* via the channel *p* from the environment and delivers it to Medium via the channel *s*. Sender's initial state is *S*0. The control states of Medium are *M*0, *M*1 and *M*2, and its initial control state is *M*0. There are four variables: *no*, *i*, *o* and *data*. *data* is a buffer for storing data. *no* represents the number of data in the buffer, *i* indicates the position in the buffer into which the next data is input, and *o* indicates the position in the buffer from which the next data is output. The initial values of these three variables are all 0's. *MAX* is a constant that represents the maximum of the number of data capable of storing. Within this range, data can be input into the buffer via the channel *s*. If there are one or more data in buffer, Medium can output data to Receiver via the channel *r*. While inputting and/or outputting data, the necessary update of the variable values is performed, with internal transitions. The control states of Receiver are *R*0 and *R*1, and its initial control state is *R*0. Receiver inputs data into the variable *dout* via the channel *r*, and outputs it to the environment via the channel *c*.

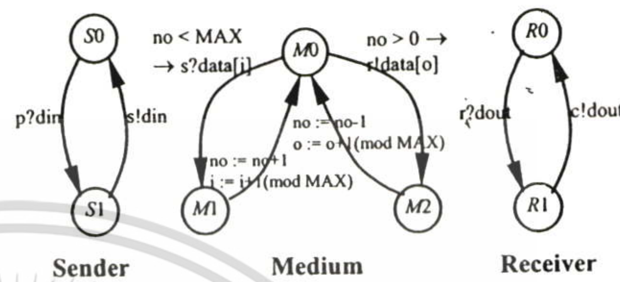


Figure 1. Data Transfer in a Communication System.

Definition 2: A *computation* of a state machine *M* is an alternating sequence of states and events

$$\alpha = s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 \ldots$$

which satisfies the following conditions:

1. $s_0 \in \theta$.
2. For all $i \geq 0$, $s_i \xrightarrow{\sigma_i} s_{i+1}$.

The set of all computations of a state machine *M* is denoted as *Comp*(*M*).

If $MAX = 2$ in Medium of Example 1, then the following sequence α is a computation of Medium:

α =(*M*0, {*no*=0, *i*=0, *o*=0, *data*[]=*null*}) *s*?"*d*1"
(*M*1, {*no*=0, *i*=0, *o*=0, *data*[]="*d*1"}) τ
(*M*0, {*no*=1, *i*=1, *o*=0, *data*[]="*d*1"}) *s*?"*d*2"
(*M*1, {*no*=1, *i*=1, *o*=0, *data*[]="*d*1" · "*d*2"}) τ
(*M*0, {*no*=2, *i*=0, *o*=0, *data*[]="*d*1" · "*d*2"}) *r*!"*d*1"
(*M*2, {*no*=2, *i*=0, *o*=0, *data*[]="*d*1" · "*d*2"}) τ
(*M*0, {*no*=1, *i*=0, *o*=1, *data*[]="*d*1" · "*d*2"}) *s*?"*d*3"
(*M*1, {*no*=1, *i*=0, *o*=1, *data*[]="*d*3" · "*d*2"}) ·····

where "*d*1", "*d*2", "*d*3" ,..... represent data values.

Definition 3: Let $\alpha = s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 \ldots$ be a computation of a state machine *M*. The sequence obtained by removing all states and all τ-events from α is said to be a *trace* of *M*.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

We can think of a trace as an observation of a computation of a state machine from the outside (environment). The set of all traces of state machine M is denoted as *Trace*(M).

From the above mentioned example, a trace of Medium is written as follows.

$$tr = s?\text{“}d1\text{”}\ s?\text{“}d2\text{”}\ r!\text{“}d1\text{”}\ s?\text{“}d3\text{”}\ \cdots$$

3. State machine with time constraint

Basic state machines described above do not have the concept of time. Here, we have introduced a time constraint into the transitions of a state machine to deal with time-dependent computations and traces. Consequently, it becomes possible to specify real-time systems, communication protocols and so on.

Here, a *time domain* Ψ is assumed to contain the minimum element 0 representing the starting time and have a usual +-operation over its elements. Ψ is also assumed to be totally ordered. In this paper, we think of the non-negative real numbers $\boldsymbol{R}\geq 0$ as Ψ. We extend Ψ to $\Psi^{\infty} = \Psi \cup \{\infty\}$ where $\infty \geq t$ for all $t \in \Psi^{\infty}$. Then the time sequence ψ is defined as follows.

Definition 4: A *time sequence* ψ is a sequence of time represented by,

$$\psi = t_0\, t_1\, t_2\, t_3\, t_4 \ldots\ldots\ldots$$

which satisfies the following requirements.

- $t_i \in \Psi$ for all $i \geq 0$.
- $t_i \leq t_{i+1}$ for all $i \geq 0$.
- if ψ is an infinite sequence, there exists $i \geq 0$ such that $t_i > t$ for all $t \in \Psi$.

In our state machine model without time concept, the firing (enabling) condition of a transition is that the current state of the state machine is at the source of the transition and the corresponding event is possible to occur. Considering the time constraint however, the *minimal delay l* and the *maximal delay u* until its firing are also taken into account in addition to that firing condition. Let φ be a function that gives transitions their minimal delays and maximal delays, as a time constraint. Then we denote the minimal delay of a transition μ as $\varphi_l(\mu)$ and the maximal delay as $\varphi_u(\mu)$, where $\varphi_l(\mu) \leq \varphi_u(\mu)$. We assume that the firing of each transition is instantaneous (i.e., takes no time).

Definition 5: Let $M = \langle Q, V, \Sigma, T, \theta\rangle$ be a state machine, and $\varphi\,(\varphi : T \rightarrow [\Psi \times \Psi^{\infty}])$ be a total function which gives each $\mu \in T$ a pair of $\varphi_l(\mu)$ and $\varphi_u(\mu)$, where $\varphi_l(\mu) \leq \varphi_u(\mu)$. Then a pair $\langle M, \varphi\rangle$ is said to be a *state machine with a time constrain* φ.

We assume that each state machine with a time constraint has a special, implicit channel *CLK* which can provide the state machine with the *current time* at any timing. And this channel is not affected by other constraints.

Notation 2: When describing the minimal delay l and maximal delay u of a transition in a state diagram, it takes the following form:

$$current\_state \xrightarrow{guard \,\rightarrow\, event[l,\,u]} next\_state$$

The part of $[l, u]$ can be omitted if $l = 0$ and $u = \infty$.

Definition 6: Let s be a state of a state machine with a time constraint. For some transitions whose source states are s and events are internal, the minimum of their maximal delays is denoted as $mm(s)$. If the transitions whose events are internal do not exist, it is defined to be $mm(s) = \infty$. If μ is a transition from s and satisfies the following conditions, then μ is said to be *firable at (relative) time t*:

1. $\varphi_l(\mu) \leq t \leq \varphi_u(\mu)$.
2. $t \leq mm(s)$.

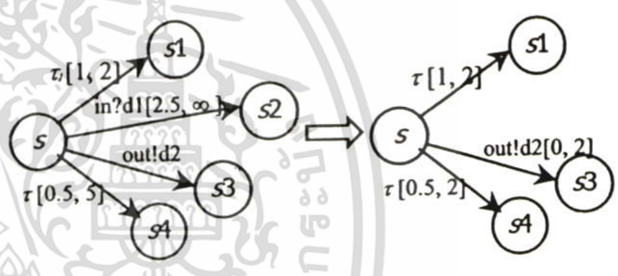


Figure 2. Firability of transitions.

Figure 2 shows an example of firability of transitions. In this case, since $mm(s) = 2$, the firability of each transition from state s is described as follows.

$s \xrightarrow{\tau} s_1$ is firable at any time between 1 and 2.
$s \xrightarrow{in?d1} s_2$ is not firable at any time.
$s \xrightarrow{out!d2} s_3$ is firable at any time between 0 and 2.
$s \xrightarrow{\tau} s_4$ is firable at any time between 0.5 and 2.

Thus the transitions on the left part can be interpreted by the right part as shown in the Figure 2.

We model a computation of a state machine with a time constraint by using *time-stamped events* and a *time sequence*. A time-stamped event takes the form of $\langle \sigma, t\rangle$ and it represents that an event σ occurs at time t.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Definition 7: Let $\langle M, \varphi\rangle$ be a state machine with a time constraint. An alternating sequence of states and time-stamped events

$$\alpha = s_0\langle \sigma_0, t_0\rangle\, s_1\langle \sigma_1, t_1\rangle\, s_2\langle \sigma_2, t_2\rangle \cdots$$

is a *computation* of $\langle M, \varphi\rangle$ if α satisfies the following conditions:

1. $s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 \ldots$ is a computation of M.
2. $t_0\, t_1\, t_2 \ldots$ is a time sequence.
3. $s_0 \xrightarrow{\sigma_0} s_1$ is firable at t_0. And for all $i \geq 1$.

   $s_i \xrightarrow{\sigma_i} s_{i+1}$ is firable at $t_i - t_{i-1}$.

The set of all computations of $\langle M, \varphi\rangle$ is denoted as $Comp(\langle M, \varphi\rangle)$.

Definition 8: Let $\alpha = s_0\langle \sigma_0, t_0\rangle\, s_1\langle \sigma_1, t_1\rangle\, s_2\langle \sigma_2, t_2\rangle \cdots$ be a computation of a state machine with time constraint $\langle M,\varphi\rangle$. The sequence obtained by removing all states and all time-stamped τ-events from α is said to be a *trace* of $\langle M, \varphi\rangle$. The set of all traces of $\langle M, \varphi\rangle$ is denote as $Trace(\langle M, \varphi\rangle)$.

4. Design and analysis tool

In this section we will present design and analysis tool that we have developed based on the concept of the basic and constrained state machine described in sections 2 and 3. This tool, "*MgraphGen*", can be used to design and specify the specification of communication systems. The developers can construct specification obviously by the state machine model with a time constraint. They can investigate and analyze the system behaviors easily and simply in the form of graphical design. The detail description is explained as follows.

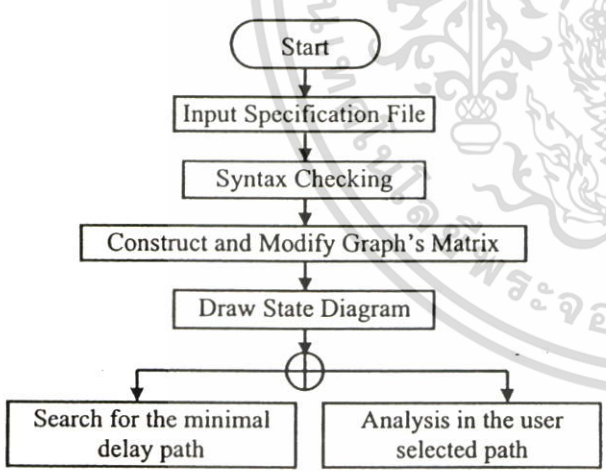


Figure 3. Flowchart of *MgraphGen*.

4.1 General features of *MgraphGen*

MGraphGen can show a state machine model in the form of state diagram corresponding to the specification of the communication system. As shown in Figure 2, firstly, the input specification in the text format will be checked for the correctly specified syntax. Then, the tool will construct and modify the *graph's matrix* in order to show its state diagram as the first result. From this step, user can investigate the processes of the specified system through this state diagram by selecting the following modes. Those are (i) the search for the shortest path or minimal delay path by choosing source and destination states, and (ii) the analysis in the user selected path.

4.2 Operation modules in *MgraphGen*

There are 5 main operation modules in this tool operated sequentially as shown in Figure 3. They are *Syntax Checking Module, Matrix Generating Module, Display Sate Diagram Module, Search for the minimal delay path Module, and Analysis in the user selected path Module.*

4.2.1 Syntax checking module

This module is used for checking the specification syntax. The syntax checking process occurs after the input specification file is loaded. If the input file has any syntax errors, it will create the error log file that contains the error or warning messages as an error report. User must correct those errors and re-input the corrected file again. Figure 4 shows a sample of specification file.

Line		Line	
1	5{Tau[2.5,5]}6	7	3{ch3!(2)[8,16]}2
2	5{ch5?a:str"2"[15,30]}4	8	3{Tau[10,20]}1
3	6{ch6?a:int(2)[10,20]}4	9	2{ch2!Acdf("23")[3,6]}1
4	6{ch6?a:int(2)[14,28]}8	10	7{ch7!Acdf((12),"23")}8
5	6{Tau[9,18]}7	11	8{Tau[17,34]}1
6	4{ch4!ab12[12,24]}3		

Figure 4. A sample of specification file.

4.2.2 Matrix generating module

This module is operated after the syntax checking has completed. First, it constructs a graph's matrix from the initial state based on the corrected specification. Then this matrix is adjusted by matrix operation until each state is posted in the proper position. Figure 5 shows modified graph's matrix of specification file shown in Figure 4. The values of this matrix show the number of path from state *i* to state *j*. For example, there is one path from state *5* to state *6* .

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์เชิงการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	5	6	4	8	7	3	1	2
5	0	1	1	0	0	0	0	0
6	0	0	1	1	1	0	0	0
4	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	1	0
7	0	0	0	1	0	0	0	0
3	0	0	0	0	0	0	1	1
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	0

Figure 5: Graph's Matrix of specification file in Figure 4.

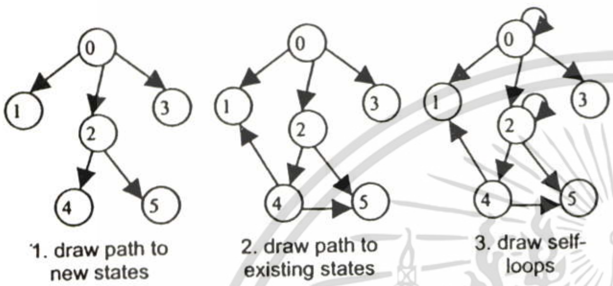


Figure 6. Sample of drawing procedure for state diagram.

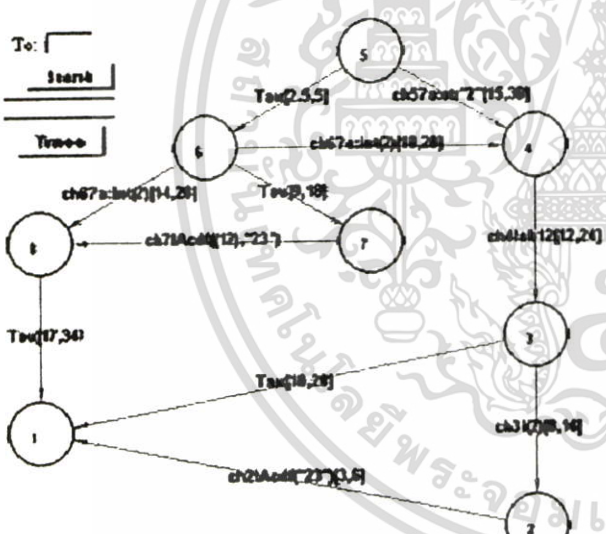


Figure 7. A sample of state diagram of the specification file.

4.2.3 Display state diagram module

This module uses to draw state diagram of the specified system. It is operated after graph's matrix has been constructed and adjusted. This module starts to draw from the initial state to a next state and from the present state to a next state. Then it finds and draws the rest of transition paths until the diagram has been completed. Finally, it draws all of the self-loop transitions. The module will record the position of all the paths for the routing of the next path. It has an ability to avoid the crossing path by finding the existing path and drawing the suitable detour path. Figure 6 shows the drawing procedure of state diagram. Figure 7 shows a sample of state diagram of the specification file in Figure 4.

4.2.4 Search for the minimal delay path module

User chooses this module in order to find the minimal delay path. The Dijkstra's algorithm is introduced into this module for searching the shortest path between specified *source* and *destination states* by the assumption that every link (or event) between two nodes (or states) has a certain minimum delay cost. Then the algorithm will search the path between the two given nodes with the lowest delay cost. The result shows the shortest path as a sequence of states and its total time consumption. Figure 8 shows a shortest path and total delay from *state 5* to *state 1* of state diagram in Figure 7.

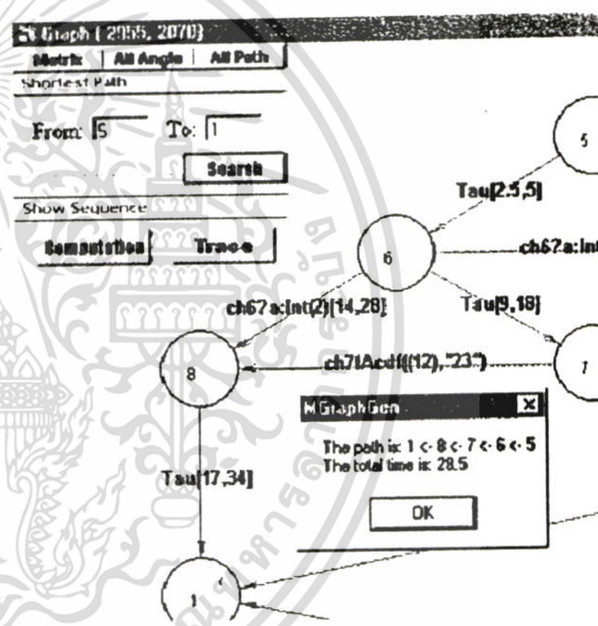


Figure 8. Result of searching for the minimal delay path.

4.2.5 Analysis in the user selected path module

User chooses this module in order to find the computation or the trace the selected path. By clicking on the expected event, the input box will ask user to enter the delay time for enabling that event (see Figure 9). The delay time must be numerical value between minimal and maximal delay of that event. Figure 10 and 11 show a sample of computation and trace results, respectively.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. Conclusions

In this paper, we have discussed the specification method for a communication system based on the state machine model with a time constraint. Several definitions and notations used in this specification method are described in definite details. Finally, the design and analysis tool, *MgraphGen*, has developed based on those specified definitions and notations.

In the future work, we will consider on the application of our method to a mobile system modeling and specification. In this case, the geographical constraints have to be introduced in the state machine models. And we will also extend the *MGraphGen* to be able to describe the behavior of a communication system in the form of component state machine.

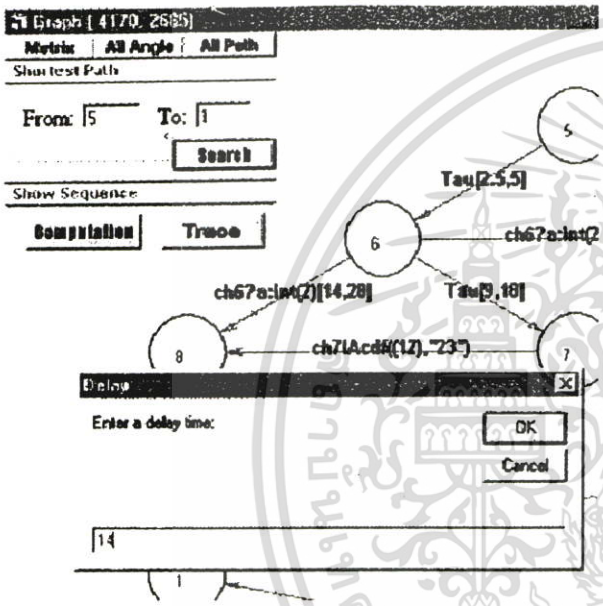


Figure 9. Delay time setting for selected path.

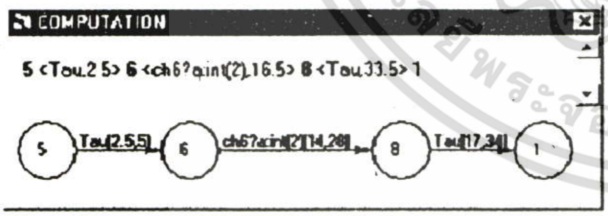


Figure 10. A sample of computation of the user selected path.

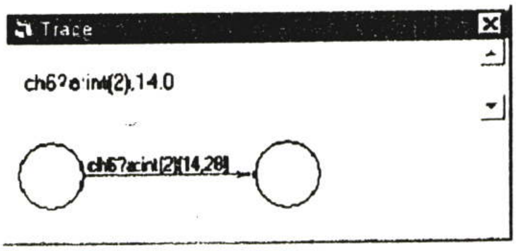


Figure 11. A sample of trace of the user selected path.

6. References

[1] R. Milner, "Communication and Concurrency," Prentice-Hall, 1989.

[2] C. A. R. Hoare, "Communication Sequential Process," Prentice-Hall, 1985.

[3] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. S. Trauring and M. Trakhtenbrot, "Statemate: a Working Environment for the Development of Complex Reactive System," IEEE Trans. Software Eng., Vol.16, No.4, pp.403-414, 1990.

[4] D. Harel and A. Naamad, "The STATEMATE Semantics of Statecharts," ACM Trans. Software Engineering and Methodology, Vol.5, No.4, pp.293-333, Oct. 1996.

[5] E. Battiston, F. D. Cindio and G. Mauli, "Modular Algebraic Nets to Specify Concurrent Syatems," IEEE Trans. Software Eng., Vol.22, No.10, pp.689-705, 1996.

[6] A. Gabrielian and M. Franklin, "Multilevel Specification of Real-Time Systems," Commun. ACM, Vol.34, No.5, pp.50-60, May. 1991.

[7] F. Jahanian and A. K. Mok, "Modechart: A Specification Language for Real-Time Systems," IEEE Trans. Software Eng., Vol.20, No.12, pp.933-947, Dec. 1994.

[8] A. Coen-Porisini, C. Ghezzi and R. A. Kemmerer, "Specification of Realtime Systems Using ASTRAL," IEEE Trans. Software Eng., Vol.23, No.9, pp.572-598, Sept. 1997.

[9] S. Yamane, "Specification and Verification of Real-time Concurrent Software," (in Japanese) Trans. Information Processing Society of Japan, Vol.37, No.2, pp.188-203, Feb. 1996.

[10] R. Alur and D. Dill, "Automata for Modeling Real-Time Systems," Lecture Notes in Computer Science 443, pp.322-335, 1990.

[11] A. C. Shaw, "Communication Real-Time State Machines," IEEE Trans. Software Eng., Vol.18, No.9, pp.805-816, Sept. 1992.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิธีการระบุข้อกำหนดสำหรับระบบการทำงานร่วมกัน โดยใช้แบบจำลองสเตทแมชชีนและเครื่องมือช่วย

Specification Method for Concurrent System Using State Machine Model and Supported Tool

ภูริชญ์ เชาว์ชนพันธุ์* นักศึกษาปริญญาโท

นพดล มณีรัตน์*·** วิศวกร

รัตติกร วรากูลศิริพันธุ์* รองศาสตราจารย์, ดร.

* ภาควิชาอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

* สำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ (ReCCIT) สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

** สำนักวิจัยและบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

บทความนี้นำเสนอการออกแบบและวิเคราะห์ระบบการสื่อสารด้วยวิธีการระบุข้อกำหนดโดยอาศัยแบบจำลองสเตทแมชชีน ซึ่งวิธีการนี้สามารถใช้ได้กับทั้งระบบที่มีข้อจำกัดทางเวลาและภูมิศาสตร์ โดยการเพิ่มข้อจำกัดทางเวลาทำให้สามารถควบคุมเวลาการเปลี่ยนสถานะของสเตทแมชชีนได้ และการเพิ่มข้อจำกัดทางภูมิศาสตร์ทำให้สามารถกำหนดพื้นที่ให้บริการและขอบเขตการสื่อสารภายในระบบได้ ในส่วนท้ายของบทความนี้ยังได้นำเสนอเครื่องมือช่วยวิเคราะห์และจำลองการทำงานของระบบ ซึ่งสามารถช่วยในการตรวจสอบและวิเคราะห์ระบบให้ถูกต้องก่อนถูกนำไปใช้งานจริงได้อีกด้วย

คำสำคัญ : การระบุข้อกำหนด แบบจำลองสเตทแมชชีน ข้อจำกัดทางเวลา ข้อจำกัดทางภูมิศาสตร์ เครื่องมือช่วยวิเคราะห์และจำลองการทำงาน

Abstract

An approach of design and analysis of communication systems is proposed by using a specification method based on the state machine model. The time and geographical constraints are introduced into the model. With time constraint, transitions of a state machine can be controlled by assigned transition delay. A geographical constraint enhances the model ability to specify the local communication among state machines. Furthermore, the analysis and simulation tool of this method has been constructed that is very helpful for investigating and analysis the specified system before the real implementation.

Keywords : Specification, State Machine Model, Time Constraint, Geographical Constraint, Analysis and Simulation Tools

1. บทนำ

ขบวนการสื่อสารส่วนใหญ่คือ ระบบการทำงานร่วมกันซึ่งมีการเชื่อมต่อถึงกัน ดังนั้นในการออกแบบระบบดังกล่าวจึงต้องคำนึงถึงการเชื่อมต่อหรือการสื่อสารร่วมกันเหล่านี้ด้วย

วิธีการระบุข้อกำหนดให้กับระบบนั้นสามารถใช้ขบวนการทางคณิตศาสตร์สร้างแบบจำลองการสื่อสารขึ้นมา เช่น CCS[1] และ CSP[2] ซึ่งนำเสนอการแสดงผลในรูปแบบพีชคณิต และสามารถใช้แสดงเหตุการณ์ที่เกิดร่วมกันได้ แต่ก็มีวิธีการระบุข้อกำหนดแบบอื่นที่อาศัยสเตทแมชชีนเป็นแบบจำลองและได้ถูกนำเสนอไปแล้ว [3-5] ข้อดีของวิธีนี้คือไม่ได้ติดอยู่ในรูปแบบทางพีชคณิต จึงมีความเหมาะสมกับการนำไปใช้งานจริงมากกว่า

บทความนี้นำเสนอวิธีการระบุข้อกำหนดของระบบแบบใหม่โดยใช้แบบจำลองสเตทแมชชีนเป็นพื้นฐาน ซึ่งเนื้อหาส่วนแรกจะกล่าวถึงแบบจำลองสเตทแมชชีน และวิธีการระบุข้อกำหนดของระบบที่ใช้ จากนั้นกล่าวถึงการเพิ่มข้อจำกัดให้แก่สเตทแมชชีน อันได้แก่ ข้อจำกัดทางเวลา และทาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภูมิศาสตร์ ด้วยข้อจำกัดนี้ทำให้ข้อกำหนดมีความใกล้เคียงกับระบบจริง และสามารถนำไปใช้กับงานได้หลากหลายขึ้น ส่วนสุดท้ายได้นำเสนอเครื่องมือหรือโปรแกรมที่ถูกพัฒนาขึ้นมาเพื่อตรวจสอบและวิเคราะห์ระบบซึ่งถูกออกแบบตามวิธีการระบุข้อกำหนดนี้ โดยโปรแกรมสามารถแสดงกราฟของแบบจำลองสเตทแมชชีนในระบบ ทำการแสดงผลเพื่อวิเคราะห์และจำลองการทำงานและพฤติกรรมต่าง ๆ ภายในระบบได้

2. สเตทแมชชีนพื้นฐาน

2.1 แบบจำลอง

สเตทแมชชีนสามารถติดต่อสื่อสารกับสิ่งแวดล้อมภายนอกหรือสเตทแมชชีนตัวอื่นๆได้ ซึ่งการติดต่อนี้เป็นการรับหรือส่งข้อมูลเข้าและออกผ่านช่องทางสื่อสาร กำหนดให้เซ็ตของช่องสื่อสารที่รองรับเซ็ตข้อมูลเข้า (I) และออก (O) เป็น C^{in} และ C^{out} ในนิยามต่อจากนี้การเข้าและออกของข้อมูลในสเตทแมชชีนผ่านช่องสื่อสารจะถูกกำหนดเป็น *เหตุการณ์ (event)* ซึ่งเป็นสาเหตุให้เกิดการเปลี่ยนแปลงในสเตทแมชชีน

นิยามที่ 1 สเตทแมชชีน M ประกอบด้วย 5 เซ็ตหลัก คือ

$M = \langle Q, V, \Sigma, T, \theta \rangle$

- Q เป็นเซ็ตจำกัดของสถานะควบคุม
- V เป็นเซ็ตจำกัดของตัวแปร
- Σ เป็นเซ็ตของเหตุการณ์ ประกอบด้วย 3 เซ็ต คือ เซ็ตเหตุการณ์รับข้อมูลเข้าจากสิ่งแวดล้อม Σ^{in} ($\Sigma^{in} \subseteq C^{in} \times I$), เซ็ตเหตุการณ์ส่งข้อมูลออกสู่สิ่งแวดล้อม Σ^{out} ($\Sigma^{out} \subseteq C^{out} \times O$) และเซ็ตเหตุการณ์ที่เกิดขึ้นภายใน สเตทแมชชีน M คือ $\{\tau\}$
- $T \subseteq (Q \times \mathrm{V}) \times \Sigma \times (Q \times \mathrm{V})$ เป็นเซ็ตของการเปลี่ยนสถานะ
- $\theta \subseteq Q \times \mathrm{V}$ เป็นเซ็ตของสถานะเริ่มต้น

กำหนดให้ $s \in Q \times V$ เป็นสถานะของสเตทแมชชีน M โดยที่ V เป็นเซ็ตของตัวแปร และ V เป็นเซ็ตค่าของเซ็ตตัวแปร V แต่เนื่องจากบางกรณีโดเมนของตัวแปรอาจเป็นเซ็ตไม่จำกัด ดังนั้นจำนวนสถานะของ M อาจมีได้ไม่จำกัดด้วย

การเปลี่ยนสถานะสามารถแสดงด้วย $s \xrightarrow{\sigma} s'$ ซึ่งสามารถเข้าใจได้ง่ายกว่า $(s, \sigma, s') \in T$ ถ้ามีสถานะ s' ที่เกิดจากสถานะ s และเหตุการณ์ σ ให้เขียนแทนด้วย $s \xrightarrow{\sigma}$

ข้อกำหนดการแสดงที่ 1 แผนผังการเปลี่ยนสถานะของสเตทแมชชีนให้แสดงดังนี้

สถานะปัจจุบัน $\xrightarrow{guard \rightarrow event}$ สถานะถัดไป

guard คือ เงื่อนไขของการเกิดเหตุการณ์ (event) ซึ่งอ้างอิงถึงตัวแปรที่เกี่ยวข้องกับการเปลี่ยนสถานะ ถ้าไม่มีความจำเป็นหรือเงื่อนไขเป็นจริงตลอดก็สามารถละไว้ได้

<u>เหตุการณ์รับข้อมูลเข้า</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ?ชื่อตัวแปร หรือ "ค่า"

ซึ่งแสดงว่าสเตทแมชชีนรับข้อมูลที่ระบุโดย ชื่อตัวแปร หรือ "ค่า" ผ่านช่องสื่อสารที่ระบุโดย ชื่อช่องสื่อสาร และสำหรับ

<u>เหตุการณ์รับข้อมูลแบบโครงสร้าง</u> ให้แสดงเป็น

ชื่อช่องสัญญาณ?ชื่อข้อความ (ตัวแปร,...,ตัวแปร)

ในที่นี้ตัวแปรอาจเป็น "ค่า" ก็ได้

<u>เหตุการณ์ส่งข้อมูลออก</u> ให้แสดงเป็น

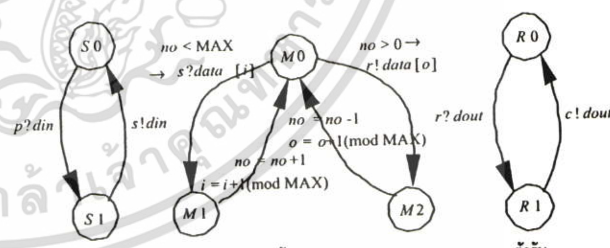
ชื่อช่องสื่อสาร!ชื่อตัวแปร หรือ "ค่า"

ซึ่งแสดงว่าสเตทแมชชีนรับข้อมูลที่ระบุโดย ชื่อตัวแปร หรือ "ค่า" ผ่านช่องสื่อสารที่ระบุโดย ชื่อช่องสื่อสาร และสำหรับ

<u>เหตุการณ์ส่งข้อมูลแบบโครงสร้าง</u> ให้แสดงเป็น

ชื่อช่องสื่อสาร!ชนิดตัวแปร (ตัวแปร,...,ตัวแปร)

สำหรับเหตุการณ์ภายในสเตทแมชชีน ถ้ามีกระบวนการเปลี่ยนค่าตัวแปร ควรแสดงกระบวนการนั้นกำกับไว้ด้วย



รูปที่ 1 ตัวอย่างระบบการรับ-ส่งข้อมูล

ตัวอย่างที่ 1 รูปที่ 1 แสดงตัวอย่างขั้นตอนการทำงานของระบบการสื่อสารด้วยแบบจำลองสเตทแมชชีน ซึ่งประกอบด้วย **ตัวส่ง** ส่งข้อมูลให้ **ตัวรับ** ผ่าน **ตัวกลาง** ตัวส่งประกอบด้วยสถานะควบคุม $S0$ และ $S1$ ทำงานโดยรับข้อมูลจากสิ่งแวดล้อมเก็บลงตัวแปร din โดยผ่านช่องสื่อสาร p และส่งข้อมูลต่อให้กับตัวกลางผ่านช่องสื่อสาร s โดยเริ่มต้นที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สถานะควบคุม $S0$ ตัวกลางประกอบด้วยสถานะควบคุม $M0$, $M1$, และ $M2$ เริ่มต้นที่สถานะควบคุม $M0$ ประกอบด้วยตัวแปร no, i, o, และ $data$ ซึ่ง $data$ เป็นตัวแปรสำหรับเก็บข้อมูลหรือตัวเก็บข้อมูล, no แสดงจำนวนของข้อมูลในตัวเก็บข้อมูล, i แสดงตำแหน่งข้อมูลเข้าในตัวเก็บข้อมูล, และ o แสดงตำแหน่งข้อมูลออกในตัวเก็บข้อมูล โดยค่าเริ่มต้นของตัวแปรทั้ง 3 ถูกกำหนดให้เป็น 0, MAX เป็นค่าคงที่เพื่อระบุจำนวนข้อมูลที่มากที่สุดที่ตัวเก็บข้อมูลสามารถเก็บได้ ข้อมูลสามารถถูกเก็บลงตัวเก็บข้อมูลโดยผ่านช่องสื่อสาร s และถ้าในตัวเก็บข้อมูลมีข้อมูลอยู่ตัวกลางก็สามารถส่งข้อมูลไปให้ตัวรับได้โดยผ่านช่องสื่อสาร r ซึ่งในระหว่างที่การรับส่งข้อมูลกระทำอยู่นั้น ค่าของตัวแปรในระบบก็จะถูกเปลี่ยนแปลงโดยเหตุการณ์ภายในไปด้วย ส่วนตัวรับประกอบด้วยสถานะควบคุม $R0$ และ $R1$ เริ่มต้นที่สถานะควบคุม $R0$ ทำงานโดยตัวรับรับข้อมูลผ่านช่องสื่อสาร r เก็บลงตัวแปร $dout$ และส่งข้อมูลนั้นออกจากระบบสู่สิ่งแวดล้อมภายนอกระบบโดยผ่านช่องสื่อสาร c

นิยามที่ 2 กำหนดให้ลำดับการทำงานของสเตทแมชชีน M เป็นลำดับของสถานะและเหตุการณ์

$$\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$$

ซึ่งเป็นไปตามเงื่อนไขต่อไปนี้

1. $S_0 \in \theta$
2. สำหรับทุก $i \geq 0, S_i \xrightarrow{\sigma_i} S_{i+1}$

เซ็ตลำดับการทำงานทั้งหมดของ M เขียนแทนด้วย $Comp(M)$

จากตัวอย่างที่ 1 สามารถแสดง สถานะ เหตุการณ์รับข้อมูลเข้า และเหตุการณ์ส่งข้อมูลออกได้ดังนี้

(สถานะควบคุม, $\{no, i, o, data\}$)
ชื่อช่องสื่อสาร?ข้อมูล
ชื่อช่องสื่อสาร!ข้อมูล

ถ้า MAX = 2 แล้ว ลำดับการทำงานที่เป็นไปได้ของตัวกลางคือ

$\alpha = (M0, \{no=0, i=0, o=0, data[\,]=null\})\, s?\text{"d1"}$
$(M1, \{no=0, i=0, o=0, data[\,]=\text{"d1"}\})\, \tau$
$(M0, \{no=1, i=1, o=0, data[\,]=\text{"d1"}\})\, s?\text{"d2"}$
$(M1, \{no=1, i=1, o=0, data[\,]=\text{"d1"."d2"}\})\, \tau$
$(M0, \{no=2, i=0, o=0, data[\,]=\text{"d1"."d2"}\})\, r!\text{"d1"}$
$(M2, \{no=2, i=0, o=0, data[\,]=\text{"d1"."d2"}\})\, \tau$
$(M0, \{no=1, i=0, o=1, data[\,]=\text{"d1"."d2"}\})\, s?\text{"d3"}$
$(M0, \{no=1, i=0, o=1, data[\,]=\text{"d3"."d2"}\})\, ...$

โดยที่ "d1", "d2", ... คือ ค่าข้อมูล

นิยามที่ 3 กำหนดให้ $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$ เป็นลำดับการทำงานของสเตทแมชชีน M ลำดับนี้ถ้าตัดสถานะและเหตุการณ์ภายในออกทั้งหมดแล้วจะเรียกว่า เส้นทางการเปลี่ยนแปลงของสเตทแมชชีน M

เส้นทางการเปลี่ยนแปลงเป็นเหมือนการสังเกตการทำงานของสเตทแมชชีนจากภายนอกหรือสิ่งแวดล้อม เนื่องจากเส้นทางการเปลี่ยนแปลงทั้งหมดของสเตทแมชชีน M จะแสดงเฉพาะเหตุการณ์ที่กระทำกับสิ่งแวดล้อมเท่านั้น เซ็ตเส้นทางการเปลี่ยนแปลงทั้งหมดของสเตทแมชชีน M เขียนแทนด้วย $Trace(M)$

จากตัวอย่างลำดับการทำงานข้างต้น เส้นทางการเปลี่ยนแปลงของตัวกลางคือ

$$tr = s?\text{"d1"}\ s?\text{"d2"}\ r!\text{"d1"}\ s?\text{"d3"}\ .....$$

2.2 ระบบการทำงานร่วมกัน

ระบบการทำงานร่วมกันประกอบด้วยกลุ่มของสเตทแมชชีนที่มีการสื่อสารกัน ในหัวข้อนี้จะนิยามพฤติกรรมของระบบการทำงานร่วมกันโดยการสร้างลำดับสถานะการทำงานของระบบ ซึ่งสามารถบ่งบอกพฤติกรรมของแต่ละสเตทแมชชีนย่อย และพฤติกรรมการติดต่อระหว่างกันได้

นิยามที่ 4 กำหนดให้ $M_1, \ldots, M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน ให้ $S = ((c_1, \ldots, c_i, \ldots, c_n), v_1 \cup \ldots \cup v_i \cup \ldots \cup v_n)$ ซึ่งประกอบด้วยกลุ่มสถานะควบคุม และเซ็ตค่าตัวแปรในแต่ละสเตทแมชชีนย่อย $M_i (1 \leq i \leq n)$ โดยที่ $V_i \cap V_j = \varnothing$ สำหรับทุก i และ j $(1 \leq i \leq n, 1 \leq j \leq n, i \neq j)$ เป็นสถานะของระบบการทำงานร่วมกัน และการถ่ายทอดสถานะของระบบ S ไปยังสเตทแมชชีน M_i ให้เขียนแทนด้วย $S^{(i)}$ ส่วนลำดับสถานะและเหตุการณ์ของระบบ $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$ ให้เรียกว่า ลำดับการทำงานของระบบการทำงานร่วมกัน

ถ้ามีการใช้ช่องสื่อสาร c ร่วมกันอย่างน้อย 2 สเตทแมชชีนแล้ว c จะถูกเรียกเป็นช่องสื่อสารร่วม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อแทน c_i ด้วย c_i' และแทน v_i ด้วย v_i' ลงในสถานะของระบบ $S = ((c_1,..., c_i,..., c_n), \mathrm{v}_1 \cup ... \cup \mathrm{v}_i \cup ... \cup \mathrm{v}_n)$ แล้วให้แสดงด้วย $S[c_i'/c_i, \mathrm{v}_i'/\mathrm{v}_i]$ ถ้ามีการแทนที่ตั้งแต่ 2 ตัวขึ้นไป ให้แสดงต่อกันไปเรื่อย ๆ ตัวอย่างเช่น ถ้าแทนที่ c_j ด้วย c_j' และแทน v_j ด้วย v_j' ($j \neq i$) ต่อจากการแทนที่ข้างต้นแล้วจะแสดงเป็น $S\,[c_i'/c_i, \mathrm{v}_i'/\mathrm{v}_i]\,[c_j'/c_j, \mathrm{v}_j'/\mathrm{v}_j]$. ถ้ามีสถานะของระบบ S และเหตุการณ์ส่งข้อมูลออก $c!d$ และมี $S^{(i)}$ ที่ทำให้ $S^{(i)} \xrightarrow{c!d}$ แล้วกำหนดให้เซ็ตของการเปลี่ยนสถานะของสเตทแมชชีนอื่นซึ่งทำให้ $S^{(j)} \xrightarrow{c?d} (c_j', \mathrm{v}_j')$ สำหรับบางสถานะ (c_j', v_j') เป็น $Peer(S^{(i)} \xrightarrow{c!d})$

นิยามที่ 5 กำหนดให้ $M_1,..., M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน ลำดับการทำงานของระบบ $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 ...$ จะเกิดขึ้นได้ ถ้าเงื่อนไขดังต่อไปนี้เป็นจริง

1. $S_0 = ((c_1,..., c_i,..., c_n), \mathrm{v}_1 \cup ... \cup \mathrm{v}_i \cup ... \cup \mathrm{v}_n)$ ซึ่ง $S_0^{(i)} \in \theta_i$ สำหรับทุก i $(1 \leq i \leq n)$
2. ให้ $S_j = ((c_1,..., c_n), \mathrm{v}_1 \cup ... \cup \mathrm{v}_n)$ สำหรับแต่ละ $j \geq 0$ แล้ว σ_j และ S_{j+1} จะเป็นไปตามเงื่อนไข (a), (b), หรือ (c)
   (a) $\sigma_j = \tau$
   สำหรับบาง i $(1 \leq i \leq n)$, ช่องสื่อสารร่วม c, ข้อมูล d, และสถานะ (c_i', v_i') ของสเตทแมชชีน M_i,
   $S_j^{(i)} \xrightarrow{c!d} (c_i', \mathrm{v}_i')$, $Peer(S_j^{(i)} \xrightarrow{c!d}) = \{S_j^{(j1)} \xrightarrow{c?d} (c_{j1}', \mathrm{v}_{j1}'), ..., S_j^{(jm)} \xrightarrow{c?d} (c_{jm}', \mathrm{v}_{jm}')\} \neq \varnothing$, และ $S_{j+1} = S_j[c_i'/c_i, \mathrm{v}_i'/\mathrm{v}_i][c_{j1}'/c_{j1}, \mathrm{v}_{j1}'/\mathrm{v}_{j1}] ... [c_{jm}'/c_{jm}, \mathrm{v}_{jm}'/\mathrm{v}_{jm}]$
   (b) σ_j เป็นเหตุการณ์รับข้อมูลเข้าหรือส่งข้อมูลออกผ่านช่องสื่อสารที่ไม่ใช่ช่องสื่อสารร่วม
   สำหรับบาง i $(1 \leq i \leq n)$,
   $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathrm{v}_i')$ และ $S_{j+1} = S_j[c_i'/c_i, \mathrm{v}_i'/\mathrm{v}_i]$
   (c) $\sigma_j = \tau$ (เหตุการณ์ภายในในสเตทแมชชีนตัวหนึ่ง)
   สำหรับบาง i $(1 \leq i \leq n)$,
   $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathrm{v}_i')$ และ $S_{j+1} = S_j[c_i'/c_i, \mathrm{v}_i'/\mathrm{v}_i]$

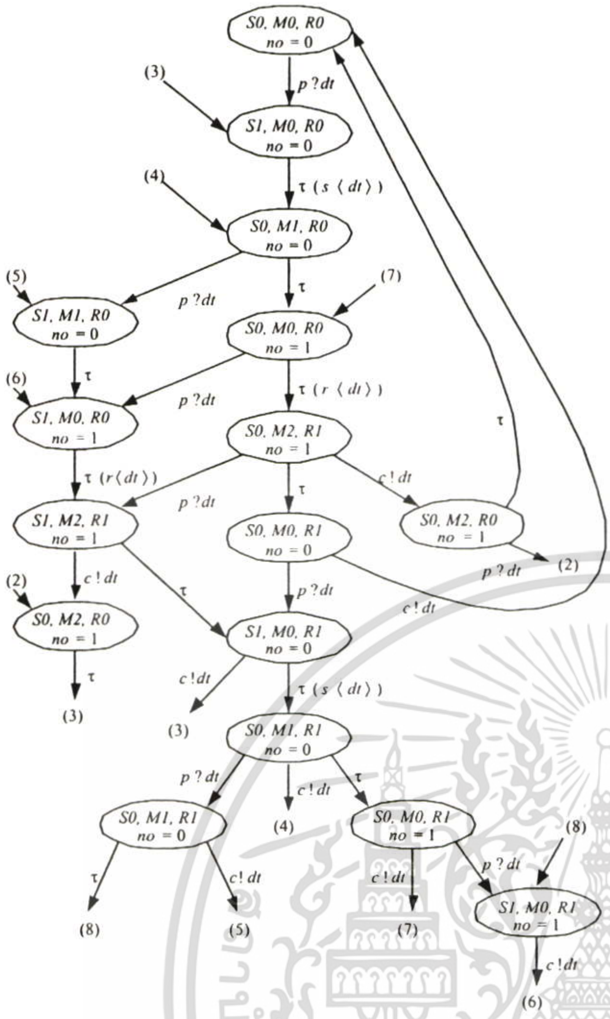
ลำดับการทำงานของระบบแสดงถึงพฤติกรรมที่สามารถเกิดขึ้นภายในระบบการทำงานร่วมกันทั้งหมด

ในนิยามนี้สมมุติว่าการสื่อสารกันระหว่างสเตทแมชชีนใด ๆ เกิดขึ้นพร้อมกัน สำหรับช่องสื่อสารที่ใช้ร่วมกันในสเตทแมชชีนตั้งแต่ 2 ตัวขึ้นไป การส่งข้อมูลออกจากสเตทแมชชีนหนึ่งจะเป็นการรับข้อมูลเข้าสู่สเตทแมชชีนตัวอื่นที่สามารถรับได้ในขณะเวลานั้น เช่น การส่งข้อมูลแบบกระจาย การรับหรือส่งข้อมูลแบบนี้เมื่อมองจากสิ่งแวดล้อมจะเห็นเป็นเพียงเหตุการณ์ภายใน ส่วนข้อมูลที่ถูกส่งเข้าและออกโดยไม่ผ่านช่องสื่อสารร่วมจะไม่กระทบต่อสเตทแมชชีนอื่นและเป็นการติดต่อกับสิ่งแวดล้อมเท่านั้น

ตัวอย่างที่ 2 พฤติกรรมที่สามารถเกิดขึ้นได้ของระบบการทำงานร่วมกัน ประกอบด้วยสเตทแมชชีน **ตัวส่ง, ตัวกลาง,** และ**ตัวรับ** ในตัวอย่างที่ 1 ถูกแสดงในรูปที่ 2 โดยให้ MAX = 1, สถานะควบคุมของระบบสามารถเขียนแทนด้วย (สถานะควบคุมตัวส่ง, สถานะควบคุมตัวกลาง, สถานะควบคุมตัวรับ) และแสดงร่วมกับค่าของตัวแปร *no* อนึ่งเหตุการณ์รับข้อมูลเข้าหรือส่งข้อมูลออกระหว่างสเตทแมชชีนจะกลายเป็นเหตุการณ์ภายใน ลำดับการทำงานของระบบที่เกิดขึ้นบ่อยที่สุดคือ ลำดับที่เริ่มจากสถานะเริ่มต้นและสิ้นสุดที่สถานะเริ่มต้นอีกครั้ง เช่น

$((S0, M0, R0), \{no = 0\})$ $p?dt$
$((S1, M0, R0), \{no = 0\})$ $\tau(s\langle dt\rangle)$
$((S0, M1, R0), \{no = 0\})$ τ
$((S0, M0, R0), \{no = 1\})$ $\tau(r\langle dt\rangle)$
$((S0, M2, R1), \{no = 1\})$ $c!dt$
$((S0, M2, R0), \{no = 1\})$ τ
$((S0, M0, R0), \{no = 0\})$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 พฤติกรรมที่เป็นไปได้ของระบบการรับ-ส่งข้อมูล

เมื่อสังเกตเหตุการณ์ในลำดับ จะเห็นว่าในระบบการทำงานร่วมกันนั้นตัวส่งจะรับข้อมูลจากสิ่งแวดล้อม และทำการส่งข้อมูลภายในระบบไปสู่ตัวรับโดยผ่านตัวกลาง และสุดท้ายตัวรับจะส่งข้อมูลนั้นออกไปสู่สิ่งแวดล้อม

3. สเตทแมชชีนแบบมีข้อจำกัด

ในหัวข้อนี้จะพัฒนาแบบจำลองสเตทแมชชีนและระบบการทำงานร่วมกันให้สามารถใช้งานได้กว้างขวางมากขึ้น โดยการเพิ่มข้อจำกัดทางเวลาและทางภูมิศาสตร์เข้าไป

3.1 ข้อจำกัดทางเวลา

ในส่วนนี้จะกล่าวถึงการเพิ่มข้อจำกัดทางเวลาเข้าไปในการเปลี่ยนสถานะของสเตทแมชชีนเพื่อให้ใช้งานได้กับลำดับการทำงานและเส้นทางการเปลี่ยนแปลงที่ขึ้นกับเวลา ทำให้สามารถนำมาประยุกต์ใช้กับระบบการประมวลผลตามเวลาจริง โปรโตคอลที่ใช้สำหรับการสื่อสาร หรือระบบอื่น ๆ ได้

เงื่อนไขการเปลี่ยนสถานะในสเตทแมชชีนแบบมีข้อจำกัดทางเวลาจะถูกพิจารณาร่วมกับเวลาด้วย โดยเงื่อนไขนั้นจะถูกพิจารณาก็ต่อเมื่อเวลายังอยู่ภายในช่วงเวลาหน่วงต่ำสุด l และเวลาหน่วงสูงสุด u ให้ φ เป็นฟังก์ชันซึ่งให้ทั้งค่าหน่วงเวลาต่ำสุดและสูงสุดในแต่ละการเปลี่ยนสถานะเป็นข้อจำกัดทางเวลา กำหนดให้ฟังก์ชันเวลาหน่วงต่ำสุดและสูงสุดของการเปลี่ยนสถานะ μ เป็น $\varphi_l(\mu)$ และ $\varphi_u(\mu)$ โดย $\varphi_l(\mu) \le \varphi_u(\mu)$ และสมมุติว่าการเปลี่ยนสถานะใด ๆ เกิดขึ้นแบบทันที

นิยามที่ 6 กำหนดให้ $M = \langle Q, V, \Sigma, T, \theta \rangle$ เป็นสเตทแมชชีน และ φ $(\varphi : T \rightarrow [\Psi \times \Psi^{\infty}])$ เป็นฟังก์ชันรวมซึ่งให้ $\varphi_l(\mu)$ และ $\varphi_u(\mu)$ ในแต่ละการเปลี่ยนสถานะ $\mu \in T$ โดยที่ $\varphi_l(\mu) \le \varphi_u(\mu)$ แล้วดังนั้น $\langle M, \varphi \rangle$ จะถูกเรียกเป็นสเตทแมชชีนแบบมีข้อจำกัดทางเวลา φ

กำหนดให้สเตทแมชชีนแบบมีข้อจำกัดทางเวลามีความพิเศษคือ มีช่องสัญญาณ CLK ซึ่งสามารถให้ค่าเวลาปัจจุบันแก่สเตทแมชชีนได้ และช่องสัญญาณพิเศษนี้ไม่ส่งผลกระทบใด ๆ ต่อข้อจำกัดทางภูมิศาสตร์ซึ่งจะอธิบายต่อไป

ข้อกำหนดการแสดงที่ 2 ถ้ากำหนดเวลาหน่วงน้อยสุด l และเวลาหน่วงมากสุด u ของการเปลี่ยนสถานะแล้ว แผนผังการเปลี่ยนสถานะสามารถแสดงในรูปแบบดังนี้:

$$\text{สถานะปัจจุบัน} \xrightarrow{guard \rightarrow event\,[l,\,u]} \text{สถานะถัดไป}$$

ในส่วนของ $[l, u]$ สามารถละได้ถ้า $l = 0$ และ $u = \infty$

สำหรับลำดับการทำงานของสเตทแมชชีนแบบมีข้อจำกัดทางเวลา จะใช้ *เหตุการณ์ที่มีเวลากำกับ* และลำดับเวลาเป็นส่วนประกอบ เหตุการณ์ที่มีเวลากำกับจะแสดงอยู่ในรูป $\langle \sigma, t \rangle$ ซึ่งหมายถึงเหตุการณ์ σ ได้เกิดขึ้นเมื่อเวลา t

นิยามที่ 7 กำหนดให้ $\langle M, \varphi \rangle$ เป็นสเตทแมชชีนแบบมีข้อจำกัดทางเวลา ลำดับการทำงานของ $\langle M, \varphi \rangle$ คือ

$$\alpha = S_0 \langle \sigma_0, t_0 \rangle S_1 \langle \sigma_1, t_1 \rangle S_2 \langle \sigma_2, t_2 \rangle \ldots$$

ถ้าเงื่อนไขดังต่อไปนี้เป็นจริง

1. $S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 \ldots$ เป็นลำดับการทำงานของ M
2. $t_0\, t_1\, t_2 \ldots$ เป็นลำดับเวลา
3. $S_0 \xrightarrow{\sigma_0} S_1$ สามารถดำเนินการได้ที่เวลา t_0. และสำหรับทุก $i \ge 1, S_i \xrightarrow{\sigma_i} S_{i+1}$ สามารถดำเนินการได้ที่เวลา $t_i - t_{i-1}$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เซ็ตของลำดับการทำงานทั้งหมดของ $\langle M, \varphi\rangle$ เขียนแทนด้วย $Comp(\langle M, \varphi\rangle)$

นิยามที่ 8 กำหนดให้ $\alpha = S_0 \langle \sigma_0, t_0\rangle S_1 \langle \sigma_1, t_1\rangle S_2 \langle \sigma_2, t_2\rangle \ldots$ เป็นลำดับการทำงานของสเตทแมชชีนแบบมีข้อจำกัดทางเวลา $\langle M, \varphi\rangle$ ลำดับที่ตัดสถานะและเหตุการณ์ที่มีเวลากำกับที่เป็นเหตุการณ์ภายใน (τ) ทั้งหมดออกจาก α จะถูกเรียกใหม่ว่า เส้นทางการเปลี่ยนแปลงของ $\langle M, \varphi\rangle$ เซ็ตเส้นทางการเปลี่ยนแปลงทั้งหมดของ $\langle M, \varphi\rangle$ เขียนแทนด้วย $Trace(\langle M, \varphi\rangle)$

3.2 ระบบการทำงานร่วมกัน

ระบบการทำงานร่วมกันสามารถจำลองการทำงานโดยอาศัยแบบจำลองสเตทแมชชีนพื้นฐานหลายตัวประกอบกันร่วมกับการระบุเงื่อนไขการสื่อสารระหว่างกัน แต่หากต้องการระบุให้มีเงื่อนไขการสื่อสารในแต่ละพื้นที่แล้ว ระบบจำเป็นต้องมีข้อจำกัดทางภูมิศาสตร์ด้วย

3.2.1 ข้อจำกัดทางภูมิศาสตร์

ในหัวข้อนี้จะนิยามความหมายของ ฟิลด์[4] เพื่อแสดงถึงการสื่อสารในแต่ละพื้นที่ของกลุ่มสเตทแมชชีน สมมุติว่าเซ็ตจำกัด L เป็นเซ็ตของพื้นที่ทั้งหมดซึ่งอาจมีสเตทแมชชีนอยู่ กำหนดให้แต่ละพื้นที่แทนด้วยตัวเลขจำนวนนับ (1, 2, ...) และเซ็ตทั้งหมดของช่องสื่อสารเป็น C ซึ่งการสื่อสารระหว่างสเตทแมชชีนจะถูกจำกัดด้วยขอบเขตของช่องสื่อสาร

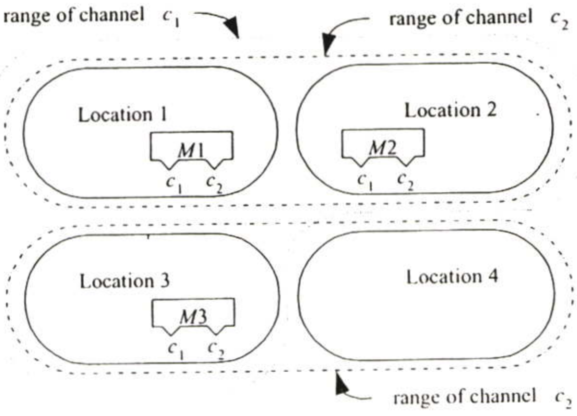
นิยามที่ 9 ถ้าฟิลด์ F มีสเตทแมชชีนอยู่แล้ว

$F = \{(c, COM(c)) \mid c \in C, COM(c) \subseteq 2^L\}$

สำหรับบาง c ซึ่ง $c \in C$, มี P ซึ่ง $P \subseteq L$ และ $P \in COM(c)$ หมายถึง สเตทแมชชีนทั้งหมดที่อยู่ในเซ็ตพื้นที่ P สามารถติดต่อถึงกันได้โดยผ่านช่องสื่อสาร c หรืออีกนัยหนึ่งคือ สเตทแมชชีนบนพื้นที่อื่นซึ่งไม่ได้อยู่ในเซ็ต P ไม่สามารถติดต่อกันได้โดยผ่านช่องสื่อสาร c และเรียก $COM(c)$ ว่า ขอบเขตของ c

ตัวอย่างที่ 3 ให้ $L = \{1, 2, 3, 4\}$ และ $C = \{c_1, c_2\}$ ตัวอย่างหนึ่งของฟิลด์ F คือ

$F = \{(c_1, \{\{1, 2, 3\}\}), (c_2, \{\{1, 2\}, \{3, 4\}\})\}$



รูปที่ 3 ตัวอย่างของฟิลด์

รูปที่ 3 แสดงสเตทแมชชีน $M1$ อยู่บนพื้นที่ 1, $M2$ อยู่บนพื้นที่ 2, และ $M3$ อยู่บนพื้นที่ 3 ในกรณีนี้สเตทแมชชีนทั้งหมดสามารถติดต่อกันได้โดยผ่านช่องสื่อสาร c_1, $M1$ และ $M2$ ติดต่อกันได้โดยผ่านช่องสื่อสาร c_2 แต่อย่างไรก็ตาม $M3$ ไม่สามารถติดต่อกับ $M1$ หรือ $M2$ โดยผ่านช่องสื่อสาร c_2 ได้ เนื่องจาก $M3$ ไม่ได้อยู่ในขอบเขตช่องสื่อสาร c_2 เดียวกันกับ $M1$ และ $M2$

สมมุติให้ฟิลด์มีคุณสมบัติ 2 อย่างดังต่อไปนี้

1. ถ้าเซ็ต P ไม่ได้อยู่ใน $COM(c)$ ซึ่ง $P \subseteq L$ และ $l \in P$ แล้ว สเตทแมชชีนบนพื้นที่ l จะไม่สามารถติดต่อกับสเตทแมชชีนอื่นโดยผ่านช่องสื่อสาร c ได้
2. ถ้า P และ $P' \in COM(c)$ $(P \neq P')$ แล้ว เซ็ต P และ P' จะต้องไม่มีสมาชิกซ้ำกัน

คุณสมบัติข้อ 2 ทำให้ระบุได้ว่าขอบเขตของช่องสื่อสารไหนกำลังถูกเลือกอยู่

3.2.2 พฤติกรรมของระบบการทำงานร่วมกัน

ในหัวข้อนี้จะนิยามพฤติกรรมของระบบการทำงานร่วมกันโดยสร้างลำดับการทำงานของระบบ ซึ่งสามารถบ่งบอกถึงพฤติกรรมของแต่ละสเตทแมชชีนย่อย และพฤติกรรมการติดต่อสื่อสารระหว่างกันได้ ในขั้นแรกต้องทำการกำหนดพฤติกรรมของระบบการทำงานร่วมกันโดยปราศจากข้อจำกัดทางเวลาก่อน (แต่ยังมีข้อจำกัดทางภูมิศาสตร์อยู่) ต่อมาจึงเพิ่มข้อจำกัดทางเวลาเข้าไป ในระบบการทำงานร่วมกันบางระบบที่มีทั้งข้อจำกัดทางภูมิศาสตร์และเวลา ถ้าขอบเขตของช่องสื่อสารครอบคลุมพื้นที่ทั้งหมดแล้ว พฤติกรรมของระบบการทำงานร่วมกันที่ได้จะมีเพียงข้อจำกัดทางเวลาเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นิยามที่ 10 กำหนดให้ $M_1, \dots, M_n$ เป็นสเตทแมชชีนย่อยของระบบการทำงานร่วมกัน และ F เป็นฟิลด์ แล้ว $\langle\{M_1, \dots, M_n\}, F\rangle$ ถูกเรียกว่า ระบบการทำงานร่วมกันของสเตทแมชชีน $M_1, \dots, M_n$ แบบมีข้อจำกัดทางภูมิศาสตร์ F

นิยามที่ 11 กำหนดให้ $\langle\{M_1, \dots, M_n\}, F\rangle$ เป็นระบบการทำงานร่วมกันแบบมีข้อจำกัดทางภูมิศาสตร์ และ l_i เป็นพื้นที่ซึ่งมีสเตทแมชชีน M_i อยู่ ลำดับการทำงานของระบบ $\alpha = S_0\, \sigma_0\, S_1 \sigma_1 S_2 \dots$ เป็นลำดับการทำงานที่มีเป็นไปได้ของ $\langle\{M_1, \dots, M_n\}, F\rangle$ ถ้าเป็นไปตามเงื่อนไขดังต่อไปนี้

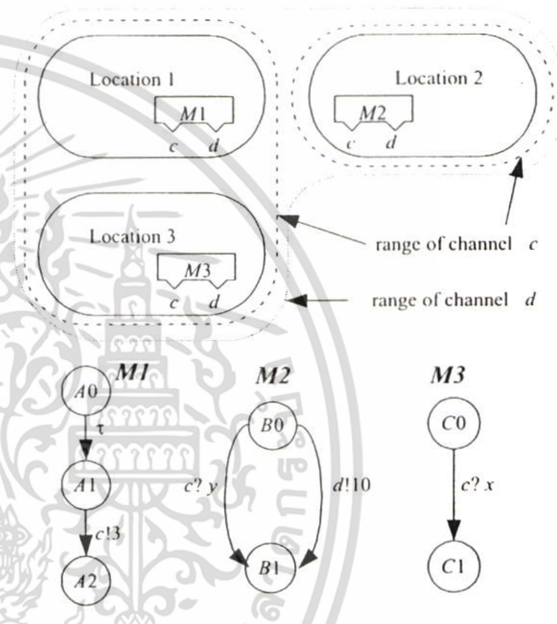
1. $S_0 = ((c_1, \dots, c_i, \dots, c_n), v_1 \cup \dots \cup v_i \cup \dots \cup v_n)$ ซึ่ง $S_0^{(i)} \in \theta_i$ สำหรับทุก i $(1 \le i \le n)$
2. ให้ $S_j = ((c_1, \dots, c_n), v_1 \cup \dots \cup v_n)$ สำหรับ $j \ge 0$ แล้ว σ_j และ S_{j+1} จะเป็นไปตามเงื่อนไข (a), (b), หรือ (c)
   - (a) $\sigma_j = \tau$
     สำหรับบาง i $(1 \le i \le n)$, ช่องสื่อสารร่วม c, ข้อมูล d, และสถานะ (c_i', v_i') ของสเตทแมชชีน M_i, $S_j^{(i)} \xrightarrow{c!d} (c_i', v_i')$, $Z = \{ S_j^{(j1)} \xrightarrow{c?d} (c_{j1}', v_{j1}'), \dots, S_j^{(jm)} \xrightarrow{c?d} (c_{jm}', v_{jm}')\} \subseteq Peer(S_j^{(i)} \xrightarrow{c!d})$ และ $S_{j+1} = S_j[c_i'/c_i, v_i'/v_i]\,[c_{j1}'/c_{j1}, v_{j1}'/v_{j1}] \dots [c_{jm}'/c_{jm}, v_{jm}'/v_{jm}]$ โดยที่ l_i อยู่ในขอบเขตหนึ่งของ c ใน F และ Z เป็นเซ็ตย่อยที่ใหญ่ที่สุดของ $Peer(S_j^{(i)} \xrightarrow{c!d})$ ซึ่งมี $l_{j1}, \dots, l_{jm}$ อยู่ในขอบเขตนี้
   - (b) σ_j เป็นเหตุการณ์รับข้อมูลเข้าหรือส่งข้อมูลออกผ่านช่องสื่อสารที่ไม่ใช่ช่องสื่อสารร่วม
     สำหรับบาง i $(1 \le i \le n)$,
     $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', v_i')$ และ $S_{j+1} = S_j[c_i'/c_i, v_i'/v_i]$ โดยที่ l_i อยู่ในขอบเขตหนึ่งของ c ใน F
   - (c) $\sigma_j = \tau$ (เหตุการณ์ภายในในสเตทแมชชีนตัวเดียว)
     สำหรับบาง i $(1 \le i \le n)$,
     $S_j^{(i)} \xrightarrow{\sigma_j} (c_i', v_i')$ และ $S_{j+1} = S_j[c_i'/c_i, v_i'/v_i]$

ตัวอย่างที่ 4 สมมุติให้ $L = \{1, 2, 3\}$, $C = \{c, d\}$, $F = \{(c, \{\{1, 3\}, \{2\}\}), (d, \{\{1, 2, 3\}\})\}$, ข้อกำหนดสเตทแมชชีนแสดงดังรูปที่ 4 และสเตทแมชชีน M_i $(1 \le i \le 3)$ อยู่บนพื้นที่ i แล้ว ลำดับการทำงานที่เป็นไปได้ของระบบการทำงานร่วมกันนี้คือ

$\alpha 1$ = $((A0, B0, C0), \{y = .., x = ..\})\ \tau\ ((A1, B0, C0), \{y = .., x = ..\})\ \tau\ (c\langle 3\rangle)\ ((A2, B0, C1), \{y = .., x = 3\})\ d!10\ ((A2, B1, C1), \{y = .., x = 3\})$

$\alpha 2$ = $((A0, B0, C0), \{y = .., x = ..\})\ \tau\ ((A1, B0, C0), \{y = .., x = ..\})\ d!10\ ((A1, B1, C0), \{y = .., x = ..\})\ \tau\ (c\langle 3\rangle)\ ((A2, B1, C1), \{y = .., x = 3\})$

$\alpha 3$ = $((A0, B0, C0), \{y = .., x = ..\})\ d!10\ ((A0, B1, C0), \{y = .., x = ..\})\ \tau\ ((A1, B1, C0), \{y = .., x = ..\})\ \tau\ (c\langle 3\rangle)\ ((A2, B1, C1), \{y = .., x = 3\})$



รูปที่ 4 ตัวอย่างของระบบการทำงานร่วมกันกับข้อจำกัดทางภูมิศาสตร์

จากตัวอย่างให้สังเกตว่า สเตทแมชชีน $M1$ และ $M2$ ไม่สามารถติดต่อสื่อสารถึงกันได้ เนื่องจาก $M1$ ไม่ได้อยู่ในขอบเขตช่องสื่อสาร c เดียวกันกับ $M2$

4. เครื่องมือช่วยวิเคราะห์และจำลองการทำงาน

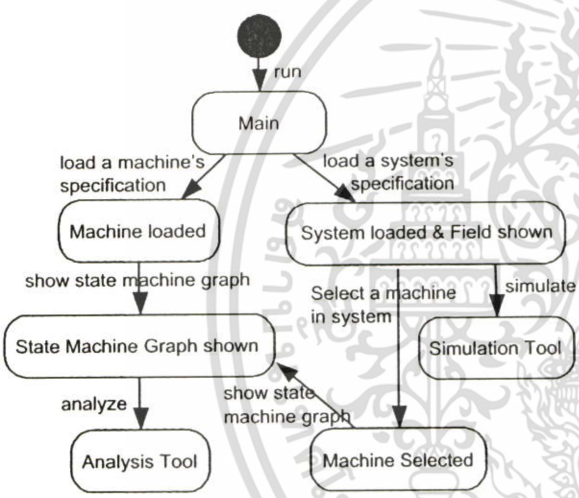
การตรวจสอบระบบให้ละเอียดได้นั้นจำเป็นต้องใช้แบบจำลองสเตทแมชชีนช่วยในการจำลองการทำงานของระบบตามข้อกำหนดที่นำเสนอมาก่อน จึงจะสามารถทำการวิเคราะห์ระบบต่อไปได้ ในงานวิจัยนี้ได้พัฒนาเครื่องมือช่วยวิเคราะห์และจำลองระบบชื่อ MGraphGen ขึ้นมาโดยอาศัยข้อกำหนดดังกล่าว เพื่อให้นักพัฒนาระบบสามารถวิเคราะห์และตรวจสอบพฤติกรรมต่าง ๆ ของระบบให้ถูกต้องได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1 ลักษณะของโปรแกรม MGraphGen

MGraphGen เป็นโปรแกรมซึ่งอาศัยรูปแบบวิธีการระบุข้อกำหนดที่ได้นำเสนอมาแล้ว โดยโปรแกรมจะเริ่มทำงานด้วยการรับแฟ้มข้อความ (Text File) ที่มีรูปแบบไวยากรณ์ตามข้อกำหนด (แฟ้มข้อกำหนด) เพื่อนำมาใช้แสดงผลและวิเคราะห์ระบบ ซึ่งลักษณะการทำงานของโปรแกรมแสดงไว้ดังรูปที่ 5 ข้อดีของโปรแกรม MGraphGen สำหรับผู้ใช้งานคือ

- สามารถเลือกวิเคราะห์ดูเส้นทางการเปลี่ยนแปลง และการทำงานของสเตทแมชชีน ตามลำดับเหตุการณ์ที่ถูกเลือก
- สามารถใช้ตรวจหาลำดับเหตุการณ์ที่ใช้เวลาหน่วงน้อยสุดจากการระบุสถานะต้นทางและปลายทางของผู้ใช้
- สามารถจำลองการทำงานของระบบเพื่อดูพฤติกรรมต่าง ๆ ของระบบโดยใช้เครื่องมือช่วยจำลองระบบ



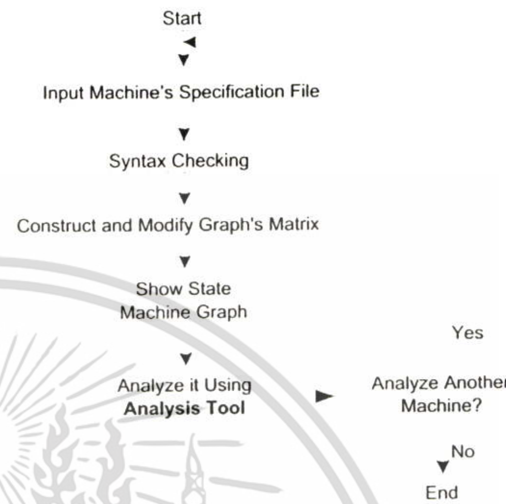
รูปที่ 5 ลักษณะการทำงานของ MGraphGen

จากรูปที่ 5 จะเห็นได้ว่าแฟ้มข้อกำหนดที่ถูกป้อนให้แก่โปรแกรมมีอยู่ 2 ประเภทคือ แฟ้มข้อกำหนดสเตทแมชชีน และแฟ้มข้อกำหนดระบบ โดยมีรายละเอียดดังนี้

- แฟ้มข้อกำหนดสเตทแมชชีนคือ แฟ้มที่ใช้ระบุข้อกำหนดต่าง ๆ ของสเตทแมชชีนพื้นฐาน และข้อจำกัดทางเวลา
- แฟ้มข้อกำหนดระบบคือ แฟ้มที่ใช้ระบุข้อกำหนดการทำงานร่วมกันของสเตทแมชชีน ซึ่งก็คือข้อกำหนดต่าง ๆ ที่เกี่ยวกับข้อจำกัดทางภูมิศาสตร์ และเนื่องจากข้อกำหนดดังกล่าวได้รวมเอาสเตทแมชชีนเข้าไว้แล้ว ดังนั้นการป้อนแฟ้มข้อกำหนดระบบจึงเป็นเสมือนการป้อนแฟ้มข้อกำหนดสเตทแมชชีนทั้งหมดที่อยู่ในระบบเข้าไปด้วย

4.2 การวิเคราะห์ระบบโดยอาศัยแบบจำลองสเตทแมชชีน

ในหัวข้อนี้จะกล่าวถึงเครื่องมือช่วยวิเคราะห์ระบบโดยอาศัยแบบจำลองสเตทแมชชีน ซึ่งจะช่วยให้นักพัฒนาระบบสามารถตรวจสอบและวิเคราะห์ระบบได้โดยง่ายและถูกต้อง รูปที่ 6 แสดงผังลำดับงานเครื่องมือช่วยวิเคราะห์ระบบ



รูปที่ 6 แผนผังลำดับงานเครื่องมือช่วยวิเคราะห์ระบบ

4.2.1 กระบวนการตรวจสอบไวยากรณ์

กระบวนการนี้มีหน้าที่ในการตรวจสอบความถูกต้องของแฟ้มข้อกำหนด หากการตรวจสอบพบข้อผิดพลาดจะแจ้งมาในรูปของแฟ้มรายงานข้อผิดพลาด ซึ่งจะถูกใช้อ้างอิงสำหรับแก้ไขแฟ้มข้อกำหนดให้ถูกต้องก่อนนำกลับมาป้อนแก่โปรแกรมอีกครั้ง ตัวอย่างแฟ้มข้อกำหนดสเตทแมชชีนที่ถูกต้องแสดงดังตารางที่ 1 โดยบรรทัดแรก:ประกาศตัวแปรที่จะถูกใช้งานภายในแฟ้มบรรทัดอื่นๆ:ระบุการเปลี่ยนสถานะของสเตทแมชชีน เช่น บรรทัดที่ 1:ระบุการเปลี่ยนสถานะจาก 5 เป็น 6 เป็นต้น

ตารางที่ 1 แสดงตัวอย่างของแฟ้มข้อกำหนดของสเตทแมชชีน

Line		Line	
0	#a;ab12;	6	4{ch4!ab12[12,24]}3
1	5{Tau[2.5,5]}6	7	3{ch3!(2)[8,16]}2
2	5{ch5?a[15,30]}4	8	3{Tau[10,20]}1
3	6{ch6?a[10,20]}4	9	2{ch2!Acdf("23")[3,6]}1
4	6{ch6?a[14,28]}8	10	7{ch7!Acdf((12),"23")}8
5	6{Tau[9,18]}7	11	8{Tau[17,34]}1

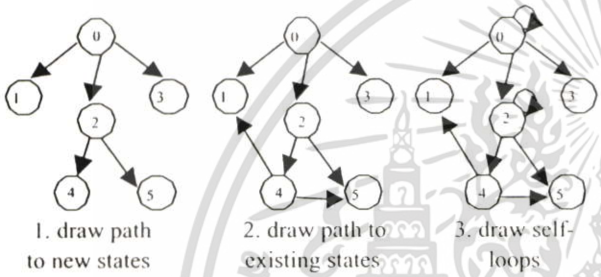
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2 กระบวนการสร้างและแก้ไขกราฟแมทริกซ์

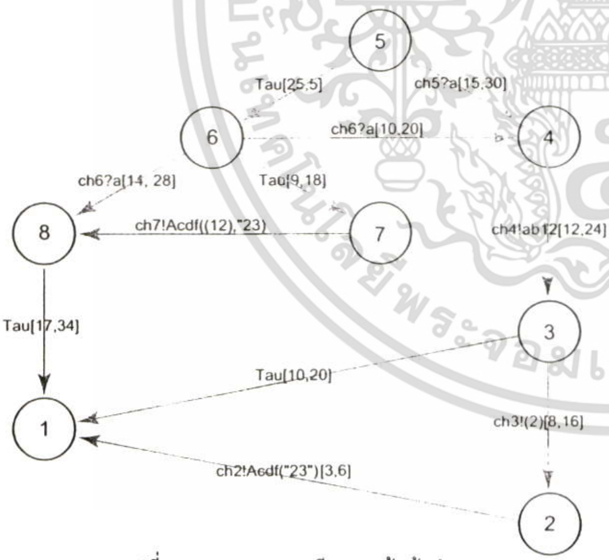
กระบวนการนี้ถูกทำหลังผ่านการตรวจสอบไวยากรณ์ข้อกำหนดแล้ว ทำหน้าที่สร้างแมทริกซ์เพื่อใช้ในการวาดกราฟสเตทแมชชีน ซึ่งแมทริกซ์นี้สามารถบอกถึงการเชื่อมต่อทั้งหมดภายในกราฟสเตทแมชชีนจากสถานะหนึ่งไปสถานะหนึ่ง

4.2.3 กระบวนการแสดงกราฟสเตทแมชชีน

กระบวนการนี้เป็นกระบวนการวาดกราฟสเตทแมชชีนลงบนฟอร์มโดยอาศัยกราฟแมทริกซ์ที่ถูกสร้างขึ้น โดยการวาดกราฟเริ่มจากสถานะเริ่มต้นไล่ไปจนถึงสถานะสิ้นสุด จากนั้นก็หาและวาดเส้นทางระหว่างสถานะที่ยังไม่ได้วาด และสุดท้ายหาและวาดเส้นทางวกกลับดังแสดงในรูปที่ 7 และรูปที่ 8 แสดงกราฟสเตทแมชชีนจากแฟ้มข้อกำหนดในตารางที่ 1



รูปที่ 7 หลักการวาดกราฟสเตทแมชชีน

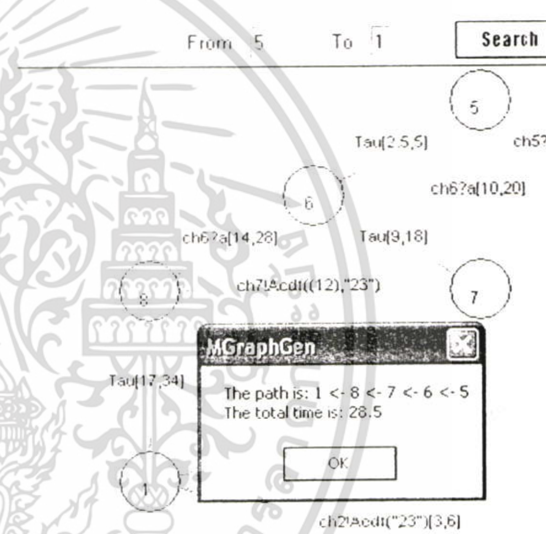


รูปที่ 8 กราฟสเตทแมชชีนจากแฟ้มข้อกำหนด

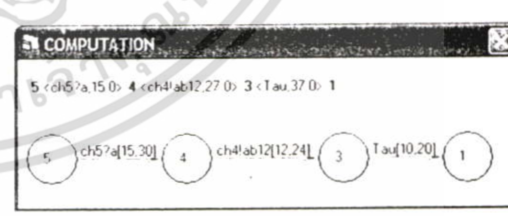
4.2.4 เครื่องมือวิเคราะห์สเตทแมชชีน

เครื่องมือนี้ผู้ใช้งานสามารถเลือกวิเคราะห์ได้ 2 แบบคือ

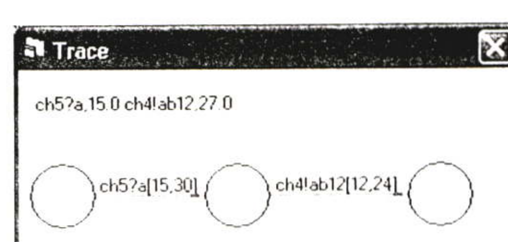
1. การตรวจหาลำดับเหตุการณ์ที่ใช้เวลาหน่วงน้อยสุด กระบวนการนี้ใช้หาระยะที่สั้นที่สุดจากสถานะเริ่มต้นไปยังสถานะปลายทาง โดยอ้างอิงค่าระยะด้วยค่าหน่วงเวลาน้อยสุด แสดงผลในรูปแบบของลำดับสถานะและเวลาการประมวลผลทั้งหมด รูปที่ 9 แสดงเส้นทางที่เกิดได้เร็วที่สุดจากสถานะ 5 ไปยังสถานะ 1 ของกราฟในรูปที่ 8
2. การวิเคราะห์ตามลำดับเหตุการณ์ที่ถูกเลือก กระบวนการนี้ผู้ใช้งานต้องทำการเลือกเส้นที่จะวิเคราะห์ก่อน โดยการเลือกเหตุการณ์ซึ่งอยู่ในกราฟสเตทแมชชีน โปรแกรมจะแสดงกล่องให้กรอกค่าหน่วงเวลาสำหรับการเกิดเหตุการณ์นั้น หลังจากเลือกลำดับเหตุการณ์แล้ว โปรแกรมสามารถแสดงลำดับการทำงานหรือเส้นทางการเปลี่ยนแปลงของสเตทแมชชีนให้วิเคราะห์ต่อไปได้ ดังตัวอย่างในรูปที่ 10 และ 11 ตามลำดับ



รูปที่ 9 ผลจากการตรวจหาลำดับเหตุการณ์ที่ใช้เวลาหน่วงน้อยสุด



รูปที่ 10 ลำดับการทำงานของลำดับเหตุการณ์ที่ถูกเลือก



รูปที่ 11 เส้นทางการเปลี่ยนแปลงของลำดับเหตุการณ์ที่ถูกเลือก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การจำลองการทำงานของระบบ

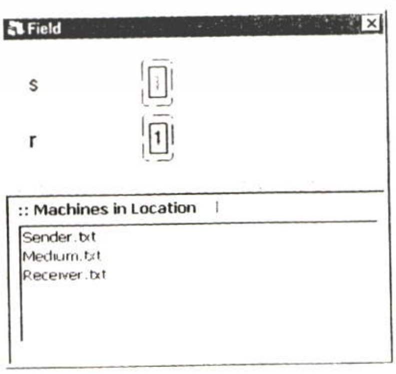
หัวข้อนี้จะกล่าวถึงการวิเคราะห์การทำงานร่วมกันของสเตทแมชชีนย่อยตั้งแต่ 2 ตัวขึ้นไปจากผลการจำลองการทำงานของระบบ เพื่อสังเกตพฤติกรรมต่าง ๆ ของระบบได้อย่างใกล้ชิด

ลักษณะของตัวจำลองการทำงานของระบบ

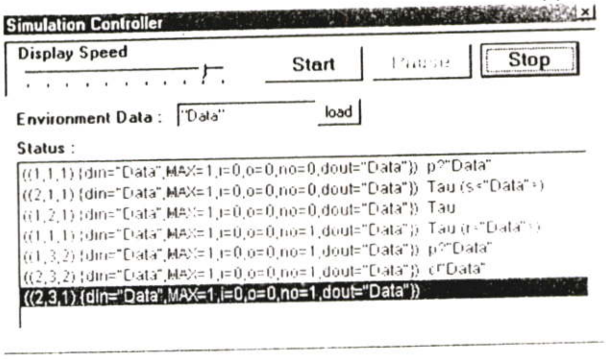
- จำลองผลการทำงานโดยอัตโนมัติ ตามข้อกำหนด
- ควบคุมการจำลองการทำงานได้ด้วยคำสั่งควบคุม เช่น เริ่มการจำลอง หยุดการจำลองชั่วขณะ และหยุดการจำลอง
- สามารถปรับเลือกความเร็วในการจำลองการทำงานได้
- สามารถกำหนดข้อมูลที่ป้อนให้แก่ระบบ (ข้อมูลที่ส่งจากสิ่งแวดล้อม) ได้เอง
- แสดงผลในรูปแบบลำดับการทำงานของระบบ

ขั้นตอนการจำลองการทำงานเริ่มจากการป้อนแฟ้มข้อกำหนดระบบให้แก่โปรแกรม ถ้าแฟ้มดังกล่าวไม่มีข้อผิดพลาดโปรแกรมจะแสดงฟิลด์ของระบบดังรูปที่ 12 ซึ่งในหน้าต่างนี้ผู้ใช้สามารถเลือกโหลดดูกราฟและวิเคราะห์สเตทแมชชีนแต่ละตัวในระบบได้ จากนั้นให้เลือกเครื่องมือการจำลองการทำงานของระบบเพื่อจำลองการทำงาน ซึ่งผลที่ได้จะแสดงในรูปของลำดับการทำงานของระบบ

รูปที่ 13 แสดงการจำลองการทำงานของระบบการรับส่งข้อมูล (ตัวอย่างที่ 1) โดยให้ค่า MAX = 1 ซึ่งสามารถตรวจดูพฤติกรรมที่เป็นไปได้ทั้งหมดตามรูปที่ 2 จากรูปในกล่อง "status" ด้านล่างเป็นลำดับการทำงานหนึ่งของระบบซึ่งเกิดจากการสุ่มเลือกการทำงานจากการทำงานที่เป็นไปได้ทั้งหมด โดยเริ่มจากสถานะเริ่มต้น ส่วนข้อความในแถบสีดำแสดงสถานะปัจจุบัน



รูปที่ 12 ตัวอย่างฟิลด์ของระบบการรับส่งข้อมูล



รูปที่ 13 ตัวอย่างการจำลองของระบบการรับส่งข้อมูล

5. สรุป

ในบทความนี้ได้นำเสนอวิธีการระบุข้อกำหนดของระบบการสื่อสารหรือระบบการทำงานร่วมกัน ซึ่งข้อกำหนดที่เสนอมานี้สามารถใช้กับระบบงานที่มีข้อจำกัดทางด้านเวลาและ/หรือข้อจำกัดทางภูมิศาสตร์ได้ และเนื่องจากวิธีการดังกล่าวอาศัยแบบจำลองสเตทแมชชีน จึงทำให้ง่ายต่อการออกแบบและพัฒนาระบบ นอกจากนี้ได้นำเสนอเครื่องมือช่วยในการวิเคราะห์และจำลองผลการทำงานของระบบเพื่อให้ผู้ใช้งานออกแบบและวิเคราะห์ระบบได้โดยง่าย

เอกสารอ้างอิง

[1] R. Milner, "Communication and Concurrency," **Prentice-Hall**, 1989.

[2] C. A. R. Hoare, "Communication Sequential Process," **Prentice-Hall**, 1985.

[3] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. S. Trauring and M. Trakhtenbrot, "Statemate: a Working Environment for the Development of Complex Reactive System," **IEEE Trans. Software Eng.,** Vol.16, No.4, pp. 403-414, 1990.

[4] G. Itabashi, K. Takahashi, Y. Kato, T. Kato and B. B. Bista, "A Specification Method for Mobile and Concurrent Systems based on the Concept of Field and Finite State Machine, and Its Application," (in Japanese) **Trans. Information Processing Society of Japan,** Vol.42, No.12, pp. 3041-3053, Dec. 2001.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[5] F. Jahanian and A. K. Mok, "Modechart: A Specification Language for Real-Time Systems," **IEEE Trans. Software Eng.,** Vol.20, No.12, pp. 933-947, Dec. 1994.

ภูริชญ์ เชาว์ชนพันธุ์ จบการศึกษาระดับปริญญาตรีสาขาวิศวกรรมไฟฟ้า จากมหาวิทยาลัยเกษตรศาสตร์ ในปี พ.ศ. 2542 ปัจจุบันกำลังศึกษาระดับปริญญาโทสาขาวิศวกรรมอิเล็กทรอนิกส์ที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สนใจด้านเครือข่ายสื่อสารคอมพิวเตอร์

นพดล มณีรัตน์ จบการศึกษาระดับปริญญาตรีสาขาวิทยาศาสตร์ฟิสิกส์ จากมหาวิทยาลัยศรีนครินทรวิโรฒ พิษณุโลก ในปี พ.ศ. 2533 ระดับปริญญาโทสาขาวิศวกรรมไฟฟ้า จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปี พ.ศ. 2539 ปัจจุบันรับราชการตำแหน่งวิศวกร ระดับ 5 สำนักวิจัยและบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง มีผลงานวิจัยด้านการควบคุมเครื่องมือวัดผ่านสายสัญญาณ IEEE-488 และเครือข่ายสื่อสารคอมพิวเตอร์

รศ. ดร.รัตติกร วรากูลศิริพันธุ์ จบการศึกษาระดับปริญญาตรีสาขาวิศวกรรมอิเล็กทรอนิกส์ จากมหาวิทยาลัย Kyoto ประเทศญี่ปุ่น ในปี ค.ศ.1978 ระดับปริญญาโทและปริญญาเอกสาขาวิศวกรรมไฟฟ้าและสื่อสารจากมหาวิทยาลัย Tohoku ประเทศญี่ปุ่น ในปี ค.ศ.1983 และ 1986 ตามลำดับ ปัจจุบันเป็นอาจารย์ประจำภาควิชาอิเล็กทรอนิกส์ที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง มีผลงานการวิจัยด้านเครือข่ายสื่อสารคอมพิวเตอร์ มัลติมีเดีย การประมวลผลภาษาธรรมชาติ เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Specification Method for Constrained Concurrent System and Its Support Tool

Purich Chaochanaphun*, Noppadol Maneerat*,†, Ruttikorn Varakulsiripunth*,

Kaoru Takahashi, and Yasushi Kato****

* Faculty of Engineering, and Research Center for Communications and Information Technology (ReCCIT), and †Computer Research and Service Center, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand. (email: kvruttik@kmitl.ac.th)

** Sendai National College of Technology, 4-16-1, Ayashi-chuo, Aoba-ku, Sendai, 989-3128 Japan.

ABSTRACT

An approach to design and analysis of communication systems is proposed by using a specification method based on the state machine model. The constraints are also introduced into the model. Furthermore, the analysis and simulation tool of this method has been developed that is very helpful for investigating and analyzing the specified system before the real implementation.

Keywords: Specification, State Machine Model, Time and Geographical Constraint, Analysis and Simulation Tool

1. INTRODUCTION

Most of communication procedure is a concurrent system that consists of a number of communicating entities. To specify such systems, it needs to consider their concurrency together with communication. The *process calculi* such as CCS [1] has been proposed as specification method by modeling the entities as processes represented by algebraic expressions. Though it can deal with the concurrency, it is an abstract one. Several specification methods based on a *state machine model* have been proposed [2, 3] that are not as abstract as the *process calculi* and are closer to real implementation.

This paper deals with one of specification methods based on state machine, and discusses a composite specification method for concurrent systems. First, we introduce a single basic state machine and model it to be able to interact with the environment via channels. By defining a communication system as a collection of such state machines, the whole behavior of the communication system is given in terms of interactions among the state machines and their individual internal activities. Next, we introduce constrained state machines that can adapt to various specification purposes. Finally, we have developed an analysis and simulation tool based on our specification method. By using this tool, the behaviors of a specified system can be investigated and verified before the real implementation.

2. BASIC STATE MACHINE

2.1. Model

A (*basic*) *state machine* has *variables* in addition to the usual (control) states. Each variable has its own *type* like *integer* or *string*. Let V be a set of variables and $\mathbf{V}$ be a set of valuations of V.

A state machine can perform interaction with the environment or other state machines. For a state machine, this interaction appears as an input or output of data via a *channel*. We denote the set of channels accepting an input set I as C^{in} and the set of channels accepting an output set O as C^{out}. In the following definition, a data input/output of a state machine via a channel is formulated as an *event* that causes a state transition of state machine.

Definition 1: A *state machine* M is a 5-tuple:

$M = \langle Q, V, \Sigma, T, \theta \rangle$

- Q is a finite set of *control states*.
- V is a finite set of *variables*.
- Σ is a set of three *events*, those are Σ^{in} ($\Sigma^{in} \subseteq C^{in} \times I$), Σ^{out} ($\Sigma^{out} \subseteq C^{out} \times O$) and $\{\tau\}$ which are a set of input events from the environment, a set of output events to the environment and a set of *internal* events, respectively.
- $T \subseteq (Q \times \mathbf{V}) \times \Sigma \times (Q \times \mathbf{V})$ is a *transition relation*.
- $\theta \subseteq Q \times \mathbf{V}$ is a set of *initial states*.

We say $s \in Q \times \mathbf{V}$ is a *state* of the state machine M and write $s \xrightarrow{\sigma} s'$ for a $(s, \sigma, s') \in T$. If there exists a state s' such that $s \xrightarrow{\sigma} s'$, we write $s \xrightarrow{\sigma}$.

Notation 1: We use a usual state transition diagram for a visual representation of a state machine. If an explicit specification of variable values is needed, it is attached to the control state. A basic structure of state transition is represented as follows:

$$current\_state \xrightarrow{guard \rightarrow event} next\_state$$

The *guard* is a boolean expression that qualifies the values of the variables related with the transition.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

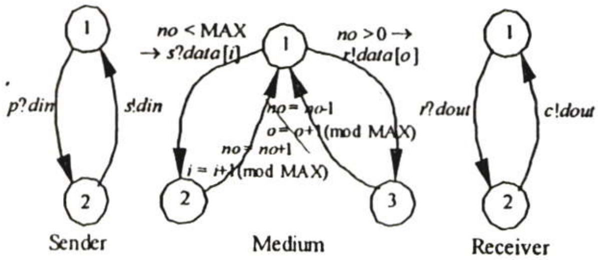


Fig.1: An Example of Data Transfer System.

The *event* is written as follows if it is an input.

channel_name?variable_name "value"
channel_name?(parameter, , parameter)

The *event* is written as follows if it is an output.
channel_name!expression
channel_name!(parameter, , parameter)

If the *event* is internal, the processing statements that may change the variable should be written. We use the symbol τ to represent the internal event.

Example 1: *Fig.1* shows a notation example of state machines for data transmission in a communication system in which Sender transmits data to Receiver via Medium. The Sender inputs data into the variable *din* via the channel *p* from the environment and delivers it to Medium via the channel *s*. The initial control state of Medium is "1 . There are four variables: *no, i, o* and *data*. *data* is a buffer for storing data. *no* represents the number of data in the buffer, *i* indicates the position in the buffer into which the next data is input, and *o* indicates the position in the buffer from which the next data is output. The initial values of these three variables are all 0 s. MAX is a constant that represents the maximum number of data storing capable. Within this range, data can be input into the buffer via the channel *s*. If there are one or more data in buffer, Medium can output data to Receiver via the channel *r*. While inputting and/or outputting data, the necessary update of the variable values is performed, with internal transitions. The Receiver inputs data into the variable *dout* via the channel *r*, and outputs it to the environment via the channel *c*.

Definition 2: A *computation* of a state machine *M* is an alternating sequence of states and events

$\alpha = s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 s_3 \ldots$

which satisfies the following conditions:

1. $s_0 \in \theta$.
2. For all $i \geq 0$, $s_i \xrightarrow{\sigma_i} s_{i+1}$.

The set of all computations of a state machine *M* is denoted as *Comp*(*M*).

If *MAX* = 2 in Medium of Example 1, then the following sequence α is a computation of Medium:

α = (1, {*no*=0, *i*=0, *o*=0, *data*[]=*null*}) *s*?"*d*1
(2, {*no*=0, *i*=0, *o*=0, *data*[]="*d*1 }) τ
(1, {*no*=1, *i*=1, *o*=0, *data*[]="*d*1 }) *s*?"*d*2
(2, {*no*=1, *i*=1, *o*=0, *data*[]="*d*1 "*d*2 }) τ
(1, {*no*=2, *i*=0, *o*=0, *data*[]="*d*1 "*d*2 }) *r*!"*d*1
(3, {*no*=2, *i*=0, *o*=0, *data*[]="*d*1 "*d*2 }) τ
(1, {*no*=1, *i*=0, *o*=1, *data*[]="*d*1 "*d*2 }) *s*?"*d*3
(2, {*no*=1, *i*=0, *o*=1, *data*[]="*d*3 "*d*2 })

where "*d*1 , "*d*2 , "*d*3 ,.. represent data values.

Definition 3: Let $\alpha = s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 s_3 \ldots$ be a computation of a state machine *M*. The sequence obtained by removing all states and all τ-events from α is said to be a trace of *M*. The set of all traces of *M* is denoted as *Trace*(*M*).

2.2 Concurrent System

A concurrent system consists of a number of communicating state machines. In this subsection, we define the behavior of the concurrent system by constructing *system state sequences* that can reflect the behavior specific to each of the component state machines and the interactive behavior among them.

Definition 4: Let M_1, M_n be component state machines of a concurrent system. A pair $S = ((c_1, \ , c_i, \ , c_n), \mathbf{v}_1 \cup \ \cup \mathbf{v}_i \cup \ \cup \mathbf{v}_n)$ comprising a tuple of control states and a set of valuations from each component state machine M_i $(1 \leq i \leq n)$ is called a *system state* of the concurrent system. The projection $(c_i, \mathbf{v}_i)$ of system state S into the state of state machine M_i is denoted as $S^{(i)}$. An alternating sequence $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$ of system states and events is called a system state sequence of the concurrent system. If a channel *c* is common among at least two state machines, then *c* is said to be a *shared channel*.

For a system state $S = ((c_1, \ , c_i, \ , c_n), \mathbf{v}_1 \cup \ \cup \mathbf{v}_i \cup \ \cup \mathbf{v}_n)$, a new system state obtained by substituting c_i' for c_i and v_i' for v_i is denoted as $S[c_i'/c_i, v_i'/v_i]$. If there are two or more such substitutions, they are put side by side. If for a system state *S* and some output event *c*!*d*, there is $S^{(i)}$ such that $S^{(i)} \xrightarrow{c!d}$, then we denote the set of transitions of the other state machines such that $S^{(j)} \xrightarrow{c?d} (c_j', \mathbf{v}_j')$ as $Peer(S^{(i)} \xrightarrow{c!d})$.

Definition 5: Let M_1, M_n be component state machines of a concurrent system. A system state sequence $\alpha = S_0\sigma_0 S_1\sigma_1 S_2\sigma_2 S_3 \ldots$ is a *valid system state sequence* of the concurrent system if the following conditions are satisfied

1. $S_0 = ((c_1, \ , c_i, \ , c_n), \mathbf{v}_1 \cup \ \cup \mathbf{v}_i \cup \ \cup \mathbf{v}_n)$ where $S_0^{(i)} \in \theta_i$ for all i $(1 \leq i \leq n)$.
2. Let $S_j = ((c_1, \ , c_n), \mathbf{v}_1 \cup \ \cup \mathbf{v}_n)$ for each $j \geq 0$. Then σ_j and S_{j+1} satisfy one of the following conditions:
   (a) $\sigma_j = \tau$
   for some i $(1 \leq i \leq n)$, shared channel *c*, data *d*, and state $(c_i', \mathbf{v}_i')$ of state machine M_i,
   $S_j^{(i)} \xrightarrow{c!d} (c_i', \mathbf{v}_i')$, $Peer(S_j^{(i)} \xrightarrow{c!d}) =$
   $\{ S_j^{(j1)} \xrightarrow{c?d} (c_{j1}', \mathbf{v}_{j1}'), \ , S_j^{(jm)} \xrightarrow{c?d} (c_{jm}', \mathbf{v}_{jm}')\} \neq \phi$, and $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$ $[c_{j1}'/c_{j1}, \mathbf{v}_{j1}'/\mathbf{v}_{j1}]$ $[c_{jm}'/c_{jm}, \mathbf{v}_{jm}'/\mathbf{v}_{jm}]$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(b) σ_j is an input or output event via a non-shared channel. for some i ($1 \le i \le n$),

$S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ and $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$

(c) $\sigma_j = \tau$ (internal event within a state machine). for some i ($1 \le i \le n$),

$S_j^{(i)} \xrightarrow{\sigma_j} (c_i', \mathbf{v}_i')$ and $S_{j+1} = S_j[c_i'/c_i, \mathbf{v}_i'/\mathbf{v}_i]$

3. STATE MACHINE WITH CONSTRAINTS

3.1. Time Constraint

Here, we have introduced a time constraint into the transitions of a state machine to deal with time-dependent computations and traces. Consequently, it becomes possible to specify real-time systems, communication protocols and so on.

Let φ be a function that gives transitions their *minimal delays* and *maximal delays*, as a time constraint. Then we denote the minimal delay of a transition μ as $\varphi_l(\mu)$ and the maximal delay as $\varphi_u(\mu)$, where $\varphi_l(\mu) \le \varphi_u(\mu)$. We assume that the firing of each transition is instantaneous.

Definition 6: Let $M = \langle Q, V, \Sigma, T, \theta\rangle$ be a state machine, and φ be a function which gives each $\mu \in T$ a pair of $\varphi_l(\mu)$ and $\varphi_u(\mu)$, where $\varphi_l(\mu) \le \varphi_u(\mu)$. Then a pair $\langle M, \varphi\rangle$ is said to be a *state machine with a time constrain* φ.

Notation 2: When describing the minimal delay l and maximal delay u of a transition in a state diagram, it takes the following form:

$$current\_state \xrightarrow{guard \rightarrow event[l,\, u]} next\_state$$

The part of $[l, u]$ can be omitted if $l = 0$ and $u = \infty$.

Definition 7: Let $\langle M, \varphi\rangle$ be a state machine with a time constraint. An alternating sequence of states and time-stamped events is a *computation* of $\langle M, \varphi\rangle$

$$\alpha = s_0\langle\sigma_0, t_0\rangle\, s_1\langle\sigma_1, t_1\rangle\, s_2\langle\sigma_2, t_2\rangle$$

if α satisfies the following conditions:

1. $s_0\sigma_0 s_1\sigma_1 s_2\sigma_2 \ldots$ is a computation of M.
2. $t_0 t_1 t_2 \ldots$ is a time sequence.
3. $s_0 \xrightarrow{\sigma_0} s_1$ is enable at t_0. And for all $i \ge 1$, $s_i \xrightarrow{\sigma_i} s_{i+1}$ is enable at $t_i - t_{i-1}$.

3.2. Geographical Constraint

We use the concept of a *field* which we introduced in [3], for representing locality of communication among state machines. We assume a finite set L of locations where state machines may exist. We represent concrete locations by natural numbers (1, 2, etc.). The set of all channels is denoted as C. The communication among state machines is restricted to limited ranges.

Definition 8: A *field* F is defined as follows:

$$F = \{(c, COM(c)) \mid c \in C, COM(c) \subseteq 2^L\}$$

For some c ($c \in C$) and P ($P \subseteq L$), $P \in COM(c)$ means that the state machines on the locations belonging to P can interact with each other via a channel c. We call an element of $COM(c)$ a range of c.

Example 2: Given $L=\{1, 2, 3, 4\}$ and $C=\{c1, c2\}$, an example of a field F is:

$$F = \{(c_1, \{\{1, 2, 3\}\}), (c_2, \{\{1, 2\}, \{3, 4\}\})\}$$

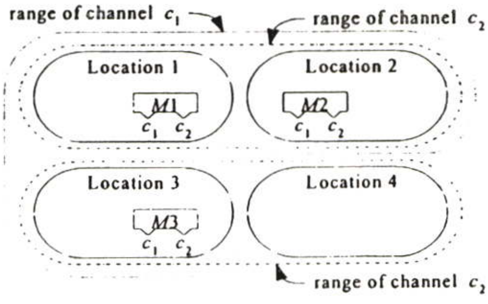


Fig.2: An Example of a Field.

Fig.2 shows an example in which a state machine $M1$ is on location 1, $M2$ on 2 and $M3$ on 3. In this situation, all the state machines can interact with each other via a channel c_1. $M1$ and $M2$ can interact via a channel c_2. However, $M3$ cannot interact with $M1$ or $M2$ via c_2.

4. ANALYSIS AND SIMULATION TOOL

We have developed the analysis and simulation tool (*MGraphGen*) based on our proposed specification method mentioned above. By using this tool the developers can construct specifications of concurrent system and investigate system behaviors easily in the form of graphical design.

4.1. Feature of *MGraphGen*

MGraphGen needs the specification in text file format as an input and it shows the graphical result as an output.

Key advantages of *MGraphGen* are as follows:

- Analysis of the computation and trace of user selected path in state machine model.
- Investigation of the shortest path in the state machine by specified source and destination states.
- Investigation of behaviors of a concurrent system by using simulation tool

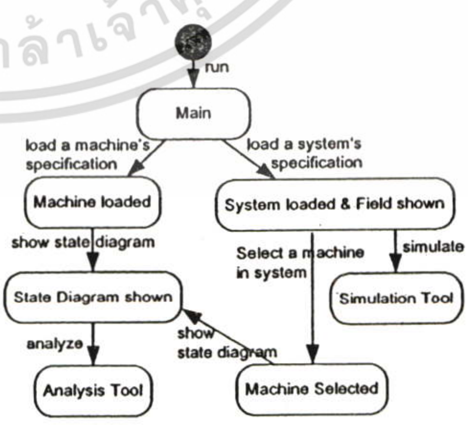


Fig.3: The Activity Diagram of MGraphGen.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลง

In *Fig.3*, the input specification can be one of two types: machine's specification and system's specification. The detail of the both type is as follows:

- Machine's specification is the specification file that contains the specification of state machine with or without time constraint.
- System's specification is the specification file. It not only contains the specification of a concurrent system such as geographical constraint but also refers to all state machines that need to run in the system.

4.2. Analysis Tool

This tool deals with the machine's specification and shows the state machine model in form of state diagram as first result. From this step, user can investigate the paths or processes of the specified machine through this state diagram by selecting the following modes. Those are (i) the search for the shortest path or minimal delay path by choosing source and destination states as shown in *Fig.4*, and (ii) the analysis in the user selected path.

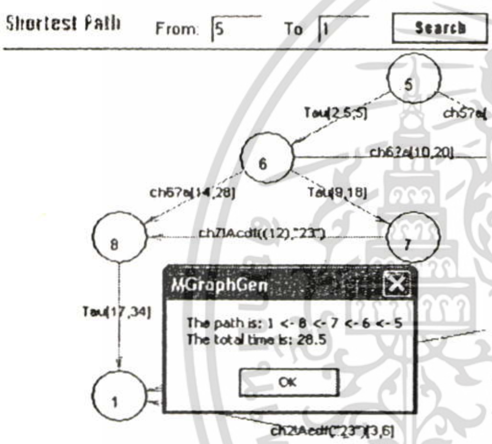


Fig.4: A Searching for the Minimal Delay Path.

4.3. Simulation Tool

To simulate the concurrent system, this tool needs the system's specification as input file and the machine's specification as reference file. The result of this tool can help developer to easily understand the behaviors of the concurrent system.

The feature of simulation tool is as follows:

- Automatic simulation followed by input specification.
- Simulation command consists of start, pause, and stop.
- Manually input data from environment.
- Display of result as system state sequence.
- Display of result as message sequence chart.

This tool can help developer to easily understand the behaviors of the concurrent system. We demonstrate the system simulation by using the specification of a data transfer system of ***Example 1*** with MAX=1. First the user must load this system's specification file as input file. Next run the simulation tool, then choose the start command to start simulation. While simulating, user can change the data that produced by environment at any time. *Fig.5* shows a simulation result while simulating.

5. CONCLUSION

In this paper, we have discussed the specification method for a communication system based on the state machine model with time and geographical constraints. Several definitions and notations used in this specification method are described in definite details. Finally, the analysis and simulation tool has been developed based on those specified definitions and notations.

In the future work, we will consider the application of our method to a mobile system.

6. References

[1] R. Milner, *Communication and Concurrency*, Prentice -Hall, 1989.

[2] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. S. Trauring and M. Trakhtenbrot, "Statemate: a Working Environment for the Development of Complex Reactive System," *IEEE Trans. Software Eng.*, Vol.16, No.4, pp.403-414, 1990.

[3] G. Itabashi, K. Takahashi, Y. Kato, T. Kato and B. B. Bista, "A Specification Method for Mobile and Concurrent Systems based on the Concept of Field and Finite State Machine, and Its Application," (in Japanese) *Trans. Information Processing Society of Japan*, Vol.42, No.12, pp.3041-3053, Dec. 2001.

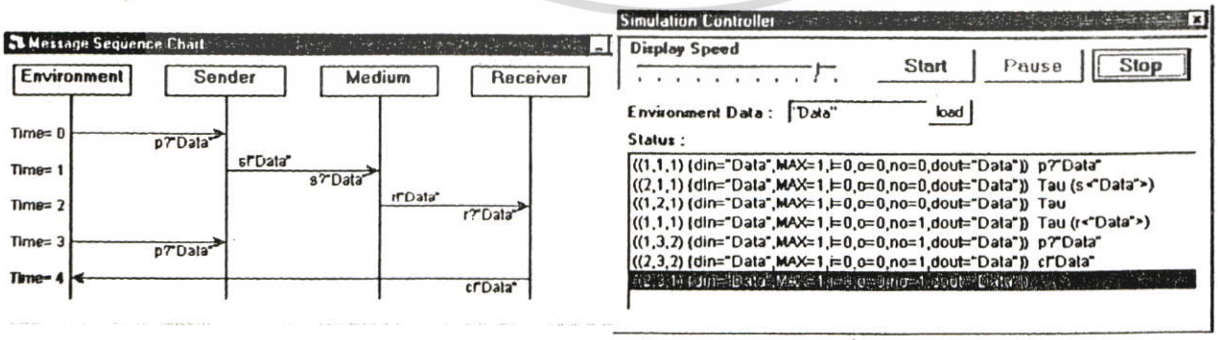


Fig.5: Simulation of the Behavior of a Data Transfer System.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

...และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประวัติผู้เขียน

นาย ภูริชญ์ เชาว์ชนพันธุ์ เกิดเมื่อวันที่ 11 มกราคม พ.ศ. 2521 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาระดับมัธยมศึกษาจากโรงเรียนปทุมคงคา และสำเร็จการศึกษาระดับปริญญาตรี วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า (ไฟฟ้าสื่อสาร) จากมหาวิทยาลัยเกษตรศาสตร์ ในปีการศึกษา 2542 และเข้าศึกษาต่อระดับปริญญาโท วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังเมื่อปี พ.ศ. 2543

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้