โครงสร้างของหน่วยบริหารการจินตทัศน์อัลกอริทึม *

Structure of Algorithm Visualization Executive

สมชาย ประสิทธิ์จูตระกูล
ผู้ช่วยศาสตราจารย์
ภาควิชาวิศวกรรมคอมพิวเตอร์
จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพฯ

ชัชวาล วงศ์ศิริประเสริฐ
นิสิตปริญญาโท
ภาควิชาวิศวกรรมคอมพิวเตอร์
จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพฯ

Somchai Prasitjutrakul
Assistant. Professor
Dept. of Computer Engineering
Chulalongkorn University Bangkok,Thailand
email: somchaip@chulkn.chula.ac.th

Chatchawan Wongsiriprasert
Graduate Student
Dept. of Computer Engineering
Chulalongkorn University Bangkok,Thailand
email: u33cws@cpu.cp.chula.ac.th

บทคัดย่อ

การจินตทัศน์อัลกอริทึมเป็นกรรมวิธีหนึ่งในการศึกษาทำความเข้าใจการทำงานของอัลกอริทึม ด้วยการใช้ภาพและการเปลี่ยนแปลงของภาพ เป็นสื่อในการแสดงถึงขั้นตอนการทำงานของอัลกอริทึม บทความนี้นำเสนอโครงสร้างของหน่วยบริหารของระบบจินตทัศน์อัลกอริทึม ซึ่งทำงานบนสภาพปฏิบัติการไมโครซอฟต์วินโดวส์ หน่วยบริหารนี้ให้บริการทางด้านการส่งผ่านข้อความ การประสานจังหวะการทำงานของขบวนการต่างๆ การสอบถามข้อมูลต่างๆของสภาพการจินตทัศน์ และการจัดการความผิดพลาดในระบบ บทการจินตทัศน์หนึ่งๆประกอบด้วยองค์ประกอบจำแนกได้เป็นสี่ประเภทคือ ตัวสร้างข้อมูล อัลกอริทึม ตัวแปลง และมุมมอง การสื่อสารระหว่างองค์ประกอบต่างๆกระทำผ่านกลไกการส่งผ่านข้อความ และองค์ประกอบอัลกอริทึมทั้งหลายในระบบจะได้รับการประสานจังหวะการทำงานเพื่อให้การทำงานเป็นไปอย่างมีระเบียบ หน่วยบริหารนี้ถูกสร้างขึ้นเป็นคลังโปรแกรมเชื่อมโยงแบบพลวัต โดยประสานการทำงานกับตัวควบคุมของระบบพัฒนาโปรแกรมวิชวลเบสิกที่มีประจำแต่ละองค์ประกอบ ส่งผลให้การพัฒนาส่วนต่างๆของระบบจินตทัศน์อัลกอริทึมกระทำได้ง่ายโดยใช้ภาษาวิชวลเบสิกและระบบอื่นๆที่สนับสนุนตัวควบคุมดังกล่าว

---

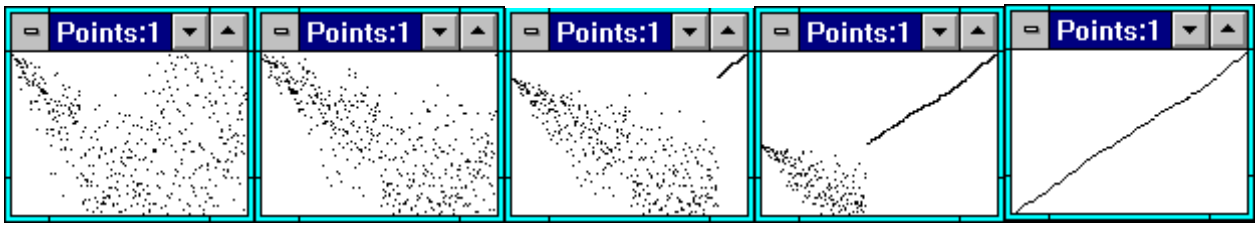
* งานนี้ได้รับการสนับสนุนจากศูนย์อิเล็กทรอนิกส์และคอมพิวเตอร์เทคโนโลยีแห่งชาติ

Abstract

Algorithm visualization is a means to study the behavior of how algorithms work. By using graphical views and animations of an algorithm in action. This paper presents the structure of the executive module of an algorithm visualization system implemented in Microsoft Windows operating enviroment. The executive message passing, process synchronization, environment-parameter inquiry, and error handling services through a set of application programming interfaces. Each visualization session consists several processes categorized into four component classes : data generator, algorithm, converter, and view. Communications among components are accomplished via the message passing mechanism using services provided by the executive. Algorithm components are synchronized so that the visualization session is orderedly carried out. The executive is implemented as a Dynamic Linking Library working in corporation with a Visual Basic custom control associated with each of the components. As a result, the algorithm visualization components can be easily developed using Visual Basic or other development tools supporting Visual Basic controls.

บทนำ

การจินตทัศน์อัลกอริทึม (Algorithm Visualization) เป็นกรรมวิธีหนึ่งในการศึกษาทำความเข้าใจการทำงานของอัลกอริธึม ด้วยการใช้ภาพ และการเปลี่ยนแปลงของภาพ เป็นสื่อในการแสดงถึงขั้นตอนการทำงานของอัลกอริทึม ในบางครั้งการใช้กรรมวิธีนี้จะเป็นเปิดโลกของผู้ทำการศึกษาเข้าสู่พฤติกรรมของการทำงานของอัลกอริธึมที่ไม่เคยจินตนาการและคาดหวังมาก่อน ตัวอย่างเช่นในรูปที่ 1 แสดงภาพบางภาพระหว่างการแสดงขั้นตอนการเรียงลำดับข้อมูลแบบฮีป บนข้อมูลที่มีค่าเริ่มต้นแบบสุ่ม โดยแสดงข้อมูลทั้งหมดในระนาบสองมิติที่ประกอบด้วยจุดต่างๆ จุดหนึ่งจุดแทนข้อมูลหนึ่งตัว มีพิกัด x ของจุดแทนตำแหน่งของข้อมูลนั้นในรายการ และพิกัด y ของจุดแทนค่าของข้อมูลนั้น ดังนั้นเมื่อเรียงลำดับเสร็จเรียบร้อยจากค่าน้อนไปค่ามาก จะได้ผลลัพธ์ดังรูปขวาสุดของรูปที่ 1



รูปที่ 1 ภาพการจินตทัศน์การเรียงลำดับข้อมูลแบบฮีปบนข้อมูลเริ่มต้นแบบสุ่ม

ด้วยตัวอย่างการจินตทัศน์ที่แสดงข้างต้นนี้ ผู้ใช้สามารถเรียนรู้และทำการทดลอง เพื่อศึกษาพฤติกรรมการทำงานของอัลกอริทึม โดยเฉพาะอย่างยิ่งการทำงานของอัลกอริทึมสำหรับสภาพของข้อมูลขาเข้าที่ต่างกัน การประยุกต์ระบบจินตทัศน์ที่เห็นเด่นชัดคือ การนำไปใช้ประกอบการเรียนการสอน ใช้เป็นเครื่องมือในการวิเคราะห์ความซับซ้อนของอัลกอริทึม ใช้ประกอบการออกแบบอัลกอริทึมใหม่ หรือใช้ปรับประสิทธิภาพของอัลกอริทึมให้เข้ากับลักษณะของปัญหา เป็นต้น

ผู้พัฒนาบทการจินตทัศน์ (Visualization session) สำหรับอัลกอริทึมหนึ่งๆนั้น จะต้องพัฒนาหรือเรียกใช้ส่วนประกอบการจินตทัศน์ที่ประกอบด้วย ส่วนผลิตข้อมูลขาเข้า ส่วนอัลกอริทึม ส่วนมุมมอง และส่วนปรับเปลี่ยนคำสั่ง (เอกสารอ้างอิง [1], [2], [3]) ดังนั้นเพื่อสนับสนุนความสะดวกในการพัฒนาบทการจินตทัศน์อัลกอริทึม ตัวระบบจินตทัศน์ต้องจัดเตรียมบริการต่างๆที่จำเป็นอาทิเช่น การส่งผ่านข้อความคำสั่งระหว่างส่วนประกอบต่างๆ การประสานจังหวะการทำงานของส่วนประกอบต่างๆ การควบคุมภาวะการจินตทัศน์ และอื่นๆ อันเป็นหน้าที่หลักของหน่วยบริหารงานการจินตทัศน์ (AVISEXEC)

บทความนี้นำเสนอโครงสร้างและการทำงานของหน่วยบริหารงานการจินตทัศน์ ซึ่งทำงานบนสภาพปฏิบัติการไมโครซอฟต์วินโดวส์ (Microsoft Windows) โดยมีลำดับการนำเสนอดังนี้ หัวข้อที่ 2 สรุปงานวิจัยที่เกี่ยวข้องทางด้านการจินตทัศน์อัลกอริทึม หัวข้อที่ 3 บรรยายถึงลักษณะโครงสร้างของระบบการจินตทัศน์อัลกอริทึมที่เสนอในงานวิจัยนี้ โดยโครงสร้างของหน่วยบริหารการจินตทัศน์จะกล่าวในรายละเอียดในหัวข้อที่ 4 จากนั้นจะสรุปสาระของบทความนี้ในหัวข้อที่ 5

งานวิจัยอื่นๆที่เกี่ยวข้อง

ความคิดในการพัฒนาเครื่องมือเพื่อช่วยให้เข้าใจการทำงานของอัลกอริทึม นั้นมีประวัติย้อนหลังไปจนถึงช่วงกลางของทศวรรษที่ 70 แนวทางแรกที่มีการพัฒนาขึ้นก็คือการบันทึกภาพการทำงานของอัลกอริทึม ด้วยฟิล์มภาพยนต์ เช่นภาพการประมวลผลโครงสร้างข้อมูลแแบบโยง(list processing)ของ Bell Lab (เอกสารอ้างอิง [4]) ,ภาพยนต์แสดงการทำงานของอัลกอริทึม ที่ใช้กับ PQ-treesของ Booth (เอกสารอ้างอิง [5]) และ ภาพยนต์แสดงการทำงานของการเรียงข้อมูลของ Beacker (เอกสารอ้างอิง [6])

แนวทางถัดมาที่มีการพัฒนาขึ้นการสร้างระบบซึ่งมีความสามารถที่จะแสดงโครงสร้างข้อมูลที่เก็บอยู่ภายในตัว อัลกอริทึม หรือโปรแกรม ทำให้สามารถมองเห็นข้อมูลซึ่งถูกเก็บอยู่ รวมทั้งการลักษณะการเปลี่ยนแปลงของข้อมูลเหล่านี้ขณะที่โปรแกรมกำลังทำงานได้ การแสดงข้อมูลในลักษณะนี้มีประโยชน์มากในการแก้ไขข้อผิดผลาดของโปรแกรมเมื่อต้องการหาว่าเมื่อใดที่ข้อมูลภายในของโปรแกรมเริ่มไม่ถูกต้อง ตัวอย่างของระบบเหล่านี้ได้แก่ Insense, GDBX, PROVIDE (เอกสารอ้างอิง [7], [8] และ [9] ตามลำดับ)

ถึงแม้ว่าการแสดงภาพโครงสร้างข้อมูลแบบนี้จะสามารถแสดงลักษณะของโครงสร้างข้อมูลได้แต่ก็ไม่สามารถที่จะแสดงให้เห็นได้ว่าตัวข้อมูลมีความสัมพันธ์กับอัลกอริทึม อย่างไร นอกจากนี้วิธีนี้ยังไม่สามารถแสดงสถานะการทำงานของ อัลกอริทึม ออกมาให้ผู้ใช้เห็นอีกด้วย จึงมีผู้พัฒนาระบบการแสดงการทำงานของ อัลกอริทึม ออกมาอีกแนวทางหนึ่งโดยการยินยอมให้ผู้พัฒนาโปรแกรมหรือ อัลกอริทึม แทรกคำสั่งพิเศษบางอย่างเข้ามาในตัวโปรแกรมเมื่อต้องการแสดงค่าข้อมูลที่น่าสนใจ นอกจากนี้ยังยินยอมให้ผู้ใช้ทดลองเปลี่ยนค่าข้อมูลแล้วสังเกตผลที่เกิดขึ้น ตัวอย่างของระบบในลักษณะนี้ได้แก่ Yarwood, De Boer, BALSA-I, Zeus, Tango, Pavens (เอกสารอ้างอิง [10], [11], [12], [13], [14] และ [15] ตามลำดับ)

โครงสร้างของระบบจินตทัศน์อัลกอริทึม

การออกแบบระบบจินตทัศน์อัลกอริทึมที่กล่าวในบทความนี้ มีวัตถุประสงค์ผลักดังนี้

1. *ความเป็นอิสระจากกัน* ส่วนต่างๆของระบบจะต้องเป็นอิสระจากกัน เนื่องจากผู้พัฒนาส่วนต่างๆมีบทบาทและความถนัดต่างกัน การพัฒนาส่วนต่างๆโดยคำนึงถึงส่วนอื่นๆในระบบให้น้อยที่สุดจะยังผลให้สามารถมุ่งความสนใจเฉพาะส่วน ทั้งนี้ทำให้การกำหนดบทบาทของผู้พัฒนาแต่ละส่วนมีความชัดเจนยิ่งขึ้น
2. *การลดผลข้างเคียง* การเปลี่ยนแปลงการทำงานในเชิงรายละเอียดของแต่ละส่วนไม่ควรมีผลข้างเคียงต่อการทำงานของส่วนอื่นๆ ลักษณะเช่นนี้แสดงให้เห็นถึงการกำหนดหน้าที่ของแต่ละส่วนจากข้อมูลขาเข้าและข้อมูลขาออกที่แต่ละส่วนรับรู้และผลิตที่ชัดเจน
3. *การใช้คลังคำสั่งร่วม* ส่วนจำเพาะต่างๆที่พัฒนาขึ้นควรถูกเก็บอยู่ในคลัง ที่สามารถนำไปใช้ใหม่ได้ การเพิ่ม การเปลี่ยนแปลง และการแทนที่ส่วนต่างๆ ควรกระทำได้โดยไม่มีผลต่อส่วนประกอบอื่นๆในระบบ
4. *ความง่ายสำหรับผู้สร้างและผู้ใช้ระบบจินตทัศน์* ระบบจะต้องใช้กลไกการประสานงานระหว่างส่วนต่างๆที่ไม่ยุ่งยากซับซ้อน ใช้คุณสมบัติและกลไกต่างๆของระบบปฏิบัติการที่เอื้ออำนวยไว้แล้ว ยังผลให้การพัฒนาปรับปรุงเพิ่มเติมเป็นไปอย่างรวดเร็ว อีกทั้งการใช้งานของผู้ใช้จะต้องเป็นไปตามสามัญสำนึกและคล้องจองกับสภาพปฏิบัติการที่เป็นอยู่

ด้วยวัตถุประสงค์หลักที่กล่าวไว้ข้างต้นนี้ ระบบจินตทัศน์อัลกอริทึมถูกแบ่งเป็นส่วนประกอบย่อยๆ 3 ส่วนดังนี้ (ดูรูปที่ 2 ประกอบ)

1. *หน่วยบริหารการจินตทัศน์* (AVISEXEC) เป็นแกนกลางสำหรับการจินตทัศน์ ส่วนนี้ให้บริการงานต่างๆดังนี้ การประสานจังหวะการทำงานขององค์ประกอบต่างๆในระบบ การส่งผ่านข้อความคำสั่งต่างๆในระบบ การให้บริการการสอบถามข้อมูลต่างๆของสภาพการทำงาน และการจัดการความผิดพลาดที่อาจเกิดขึ้นในระบบ
2. *หน่วยควบคุมบทจินตทัศน์* (AVISSESSIONCONTROL) คือส่วนควบคุม ออกแบบ ตั้งค่า และสั่งงานการจินตทัศน์ ส่วนนี้จะเป็นส่วนที่ติดต่อประสานงานกับผู้ใช้โดยตรง เพื่อให้เกิดความคล่องตัวระหว่างการจินตทัศน์
3. *องค์ประกอบการจินตทัศน์* (AVISCOMPONENT) อันประกอบด้วยองค์ประกอบต่างๆที่ถูกนำเข้า นำออก ควบคุมสั่งงานจาก AVISSESSIONCONTROL และใช้บริการการจินตทัศน์ต่างๆจาก AVISEXEC เพื่อการจินตทัศน์อัลกอริทึมที่ได้ออกแบบไว้ ดังนี้

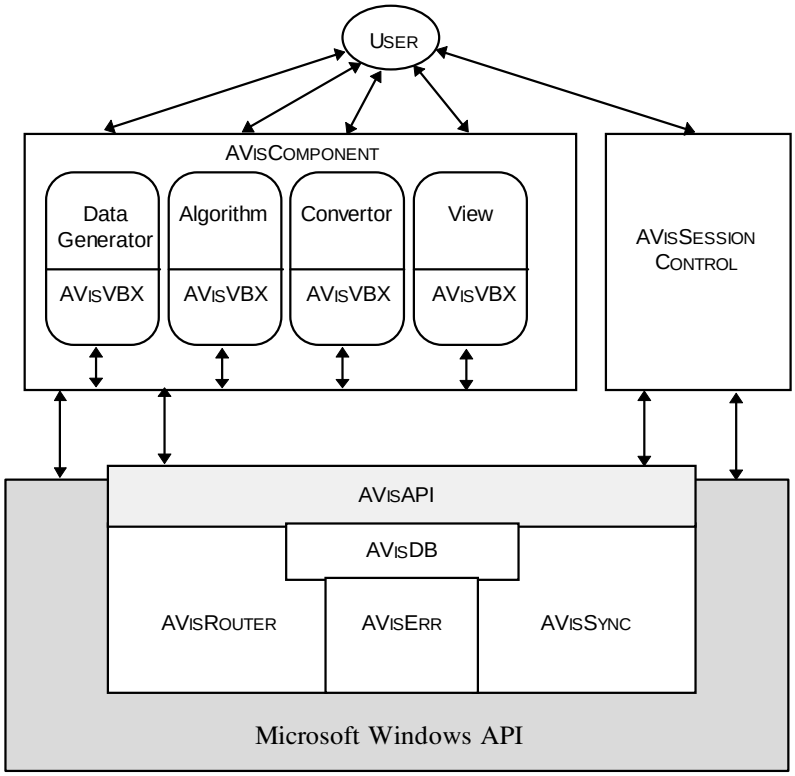
   3.1 DATAGENERATOR คือองค์ประกอบในการผลิตข้อมูลขาเข้าสำหรับการทำงานของอัลกอริทึม

   3.2 ALGORITHM คือองค์ประกอบที่เก็บอัลกอริทึม ที่ต้องการจินตทัศน์ไว้

   3.3 VIEW คือองค์ประกอบมุมมองเพื่อการแสดงผลทางจอภาพ โดยใช้ภาพและการเปลี่ยนแปลงภาพ เป็นสื่อในการจินตทัศน์ ในระบบจะมีมุมมองได้หลายแบบทั้งนี้ขึ้นกับความเหมาะสมของการตีความและการทำความเข้าใจการจินตทัศน์

   3.4 CONVERTER คือองค์ประกอบการแปลงข้อความคำสั่งจาก ALGORITHM สู่ VIEW ที่ได้เลือกไว้ ทั้งนี้เพื่อให้ภาพที่แสดงออกทางมุมมองสื่อความหมายตามสภาวะการทำงานของอัลกอริทึม

โดยทั่วไป การพัฒนาองค์ประกอบเหล่านี้กระทำได้ง่ายและรวดเร็วโดยใช้ระบบพัฒนาโปรแกรมวิชวลเบสิก (Visual Basic Development System เอกสารอ้างอิง [16]) แต่ละองค์ประกอบจะมี AVISVBX ซึ่งเป็นตัวควบคุมเฉพาะงานของวิชวลเบสิก (Visual Basic Custom Control) เพื่อประสานการทำงานกับ AVISEXEC ได้สะดวก เนื่องจาก AVISVBX ซ่อนความซับซ้อนของกลไกการประสานงานเหล่านี้ จากผู้พัฒนาองค์ประกอบ



รูปที่ 2 โครงสร้างของระบบจินตทัศน์อัลกอริทึม

หน่วยบริหารการจินตทัศน์ (AVISEXEC)

หน่วยบริหารการจินตทัศน์ถือได้ว่าเป็นส่วนที่สำคัญมากและเป็นแกนหลักในการทำงานของระบบจินตทัศน์ เป็นส่วนที่บริหารการทำงาน ข้อมูล และสภาพแวดล้อมต่างๆระหว่างการจินตทัศน์ AVISEXEC ประกอบด้วยส่วนย่อยที่ให้บริการต่างๆกันคือ (ดูรูปที่ 2) การประสานจังหวะการทำงานขององค์ประกอบต่างๆในระบบ (AVISSYNC) การส่งผ่านข้อความคำสั่งต่างๆในระบบ (AVISROUTER) ที่จัดเก็บในรูปของ Message Routing Graph การให้บริการการสอบถามข้อมูลต่างๆของสภาพการทำงาน (AVISDB) การจัดการความผิดพลาดและเหตุการณ์ที่ไม่คาดคิดที่อาจเกิดขึ้นในระบบ (AVISERR) โดยบริการทั้งหมดนี้ถูกเรียกใช้ผ่านส่วนเชื่อมประสานโปรแกรม (AVISAPI) ที่คอยตรวจสอบสิทธิ์และความถูกต้องของการขอใช้บริการ รายละเอียดการทำงานของส่วนต่างๆจะได้กล่าวในหัวข้อย่อยต่อไปนี้

AVISAPI

AVISAPI (Algorithm Visualization Application Programming Interface) เป็นส่วนประกอบส่วนเดียวของ AVISEXEC ที่ส่วนอื่นๆของระบบ จะเรียกเพื่อขอใช้บริการของ AVISEXEC ได้ AVISAPI มีหน้าที่หลักอยู่สามประการคือ

1. ตรวจสอบสิทธิ์ของผู้เรียกใช้ฟังก์ชัน ทั้งนี้บางฟังก์ชันจะถูกเรียกใช้ได้จากองค์ประกอบบางประเภทเท่านั้น ตัวอย่างเช่น AVISSESSIONCONTROL เท่านั้นที่มีสิทธิ์ลงหรือถอนทะเบียนองค์ประกอบอื่นๆในระบบเป็นต้น
2. ตรวจสอบความถูกต้องของข้อมูลซึ่งผู้เรียกใช้ส่งมาประกอบการขอบริการ (parameter validation) ถ้าข้อมูลที่ส่งมาถูกต้องก็จะเรียกโปรแกรมย่อยของ AVISDB, AVISROUTER หรือ AVISSYNC ที่เหมาะสมมาทำงานต่อไป แต่หากพบข้อผิดผลาดจะให้ค่าความผิดผลาดกลับไปยังผู้ขอใช้บริการโดยไม่ทำการเรียกโปรแกรมย่อยที่เกี่ยวข้องนั้นๆมาทำงาน
3. เป็นตัวกั้นระหว่างฟังก์ชันซึ่งผู้ขอใช้บริการมองเห็นจากฟังก์ชันและข้อมูลภายในของระบบ (encapsulation) การทำเช่นนี้มีประโยชน์คือ หากเมื่อใดเกิดการเปลี่ยนแปลงโครงสร้างข้อมูลและการทำงานภายในของระบบ ก็จะไม่มีผลต่อผู้ขอใช้บริการ
4. ให้บริการฟังก์ชันบางอย่างซึ่งเป็นฟังก์ชันที่ไม่เกียวกับระบบจินตทัศน์ อัลกอริทึม โดยตรง แต่จะทำให้ผู้ใช้พัฒนาส่วน AVISCOMPONENT ได้ง่ายขึ้น เช่นการบริการการส่งผ่านข้อมูลที่เป็นแถวลำดับ เพื่อหลีกเลี่ยงการส่งข้อมูลทีละตัว การตั้งข้อความประจำวินโดว์ขององค์ประกอบที่ทำงานอยู่ เพื่อให้ผู้ใช้รับรู้ เป็นต้น

ฟังก์ชันต่างๆที่ AVISEXEC มีไว้บริการผ่านทาง AVISAPI นั้น สามารถแบ่งออกเป็นกลุ่มย่อยๆตามหน้าที่การทำงานได้ดังนี้

1 บริการการจัดการองค์ประกอบที่ทำงานในระบบ

ตารางที่ 1 แสดงฟังก์ชันซึ่งใช้จัดการองค์ประกอบที่ทำงานในระบบ

ชื่อฟังก์ชัน	หน้าที่

• ฟังก์ชั่นที่ใช้เปิดและปิด AVISDB
`AVisOpenSession`
`AVisCloseSession`
	
	จัดเตรียมระบบก่อนเริ่มสั่งการให้องค์ประกอบต่างๆในระบบเริ่มทำงาน
	บอกเลิกการจินตทัศน์ขององค์ประกอบต่างๆในระบบ

• ฟังก์ชั่นที่ใช้เพิ่มข้อมูล
`AVisRegisterComponent`
`AVisRegisterController`
	
	ลงทะเบียนองค์ประกอบเข้าสู่ระบบ
	แจ้งในระบบทราบว่าผู้เรียก จะทำหน้าที่เป็นหน่วยควบคุมบทจินตทัศน์ (AVISSESSIONCONTROL)

• ฟังก์ชั่นที่ใช้ลบข้อมูล	

AVisUnregisterComponent	นำองค์ประกอบที่กำหนดออกจากระบบ
AVisUnregisterController	ยกเลิกหน้าที่หน่วยควบคุมบทจินตทัศน์
• ฟังก์ชั่นที่ใช้สืบค้นข้อมูล	
AVisEnumComponent	ค้นหาองค์ประกอบทุกตัวที่มีในระบบ
AVisIsValidID	ตรวจสอบว่าองค์ประกอบว่ายังอยู่ในระบบหรือไม่
AVisGetComponentWindow	ให้ค่าหมายเลขประจำวินโดว์ขององค์ประกอบ
AVisGetComponentID	ค้นหาหมายเลขประจำองค์ประกอบจากหมายเลขวินโดว์(HWND) และหมายเลขโปรแกรม(HINSTANCE)
AVisGetComponentType	ให้บริการสอบถามประเภทขององค์ประกอบ
AVisGetSyncCount	ให้ค่าจำนวนครั้งของขั้นตอนการทำงานที่ถูกประสานจังหวะ

จากฟังก์ชันทั้งหมดจะเห็นว่าไม่มีฟังก์ที่สามารถใช้แก้ไขข้อมูลของ AVISDB ปรากฏอยู่เลยทั้งนี้เพราะผู้ขอใช้บริการไม่สามารถแก้ไขข้อมูลเหล่านี้ได้โดยตรง แต่จะต้องทำการแก้ไขผ่าน AVISROUTER หรือ AVISSYNC โดยเรียกใช้ AVISAPI ที่เหมาะสม

2 บริการการจัดการการรับส่งข้อความคำสั่งของAVISROUTER

ตารางที่ 2. แสองฟังก์ชันซึ่งให้บริการการจัดการการรับส่งข้อความคำสั่งของ AVISROUTER

ชื่อฟังก์ชัน	หน้าที่
• ฟังก์ชั่นที่ใช้สร้างและลบ และสืบค้นข้อมูลของ Message Routing Graph	
AVisBind	เชื่อมความสัมพันธ์ในการรับส่งข้อมูลระหว่างองค์ประกอบ ผู้ส่งข้อมูล และองค์ประกอบผู้รับข้อมูล
AVisUnbind	ลบความสัมพันธ์ในการรับส่งข้อมูลระหว่างองค์ประกอบผู้ส่งข้อมูลและองค์ประกอบผู้รับข้อมูล ที่เคยถูกเชื่อมไว้ด้วยฟังก์ชัน AVisBind
AVisEnumLink	ค้นหาองค์ประกอบทุกตัวที่เชื่อมโยงกับองค์ประกอบที่กำหนด
• ฟังก์ชันที่ใช้ส่งข้อความคำสั่งไปยังองค์ประกอบอื่น	
AVisInputRequest	ส่งข้อความคำสั่งไปยังองค์ประกอบตัวแรกที่เป็นตัวส่งข้อมูล ขององค์ประกอบที่กำหนด
AVisInputRequestEx	ส่งข้อความคำสั่งจากองค์ประกอบหนึ่งไปยังอีกองค์ประกอบหนึ่งที่เป็นตัวส่งข้อมูล
AVisOutputNotify	ส่งข้อความคำสั่งไปยังองค์ประกอบทุกๆตัวที่เป็นตัวรับข้อมูล ของ

	องค์ประกอบที่กำหนด
AVisOutputNotifyEx	ส่งข้อความคำสั่งจากองค์ประกอบหนึ่งไปยังอีกองค์ประกอบหนึ่งที่เป็นตัวรับข้อมูล
• ฟังก์ชันที่ใช้เพื่อสร้าง กลไกของ Parallel Call (เอกสารอ้างอิง [13])	
AVisReplyMessage	แจ้งเตือนว่าองค์ประกอบที่ได้รับคำสั่งทำงานเสร็จแล้ว เมื่อองค์ประกอบใดทำงานตามข้อความคำสั่งที่ได้รับแล้วเสร็จ จะต้องเรียกฟังก์ชันนี้เพื่อแจ้งให้ผู้ส่งข้อความคำสั่งดังกล่าวทราบ
• ฟังก์ชั่นที่ใช้เพื่อป้องกันการเกิด ข้อความคำสั่งซ้อน	
AVisSuspendMessage	แจ้งเตือนAVISวAองคWประกอบฤมSพรษอมทีSจะรับขษอความคำสัSงรดๆ
AVisResumeMessage	ยกเลิกสภาวะไม่รับข้อความคำสั่งที่เกิดจากการเรียกฟังก์ชัน AVisSuspendMessage

โดยทั่วไป Component ที่พัฒนาด้วย Microsoft Visual Basic จะไม่ต้องเรียกใช้ฟังก์ชันเหล่านี้โดยตรงแต่จะขอใช้บริการเหล่านี้ผ่านทาง AVISVBX ที่เตรียมไว้ให้แล้ว(ดูรูปที่ 1)

3 บริการการประสานจังหวะการทำงานขององค์ประกอบอัลกอริทึมของAVISSYNC

ตารางที่ 3. แสดงฟังก์ชันซึ่งให้บริการการประสานจังหวะการทำงานขององค์ประกอบอัลกอริทึมของ AVISSYNC

ชื่อฟังก์ชัน	หน้าที่
• ฟังก์ชันที่องคWประกอบเรียกเพื่อประสานจังหวะการทำงานกับองคWประกอบอื่น	
AVisOpenSync	แจ้งเตือนส่วนประสานจังหวะให้ตรียมโครงสร้างข้อมูลสำหรับเก็บข้อมูลการประสานจังหวะขององค์ประกอบที่เรียกใช้ฟังก์ชันนี้
AVisCloseSync	แจ้งเตือนส่วนประสานจังหวะให้ลบโครงสร้างข้อมูลที่เตรียมไว้ด้วยฟังก์ชัน AVisOpenSync เนื่องจากองค์ประกอบไม่ต้องการประสานจังหวะอีกต่อไป
AVisSync	แจ้งเตือนให้ส่วนประสานจังหวะทราบว่าองค์ประกอบอัลกอริทึมได้ทำงานมาแล้ว เพื่อให้ส่วนประสานจังหวะอนุญาติให้องค์ประกอบอัลกอริทึมอื่นทำงานได้
• ฟังก์ชั่นที่ใช้สำหรับตั้งและอ่านค่าสถานะการทำงานของส่วนประสานจังหวะ	
AVisGetSyncStatus	อ่านค่าสถานะส่วนประสานจังหวะ

AVisGetSyncMode	อานค่าภาวะการทำงานของส่วนประสานจังหวะ
AVisSetSyncMode	ตั้งภาวะการทำงานของส่วนประสานจังหวะให้อยู่ในสภาวะที่ต้องการ
AVisStepNextSync	สั่งเริ่มการทำงานรอบต่อไป หลังจากหยุดรอการทำงานทุกๆ รอบเมื่อทำงานในสภาวะทำทีละคำสั่ง
AVisEnumSync	แจงองค์ประกอบทุกๆตัวที่ทำงานแบบประสานจังหวะกันในระบบ

4 บริการพิเศษอื่นๆแก่องค์ประกอบต่างๆที่ทำงานในระบบ

ตารางที่ 4. แสองฟังก์ชันซึ่งให้บริการพิเศษอื่นๆแก่องค์ประกอบต่างๆที่ทำงานในระบบ

ชื่อฟังก์ชัน	หน้าที่
AVisVBCreateArrayParam	การสร้างหมายเลขค่าขนาด long เพื่อแทนอาเรย์ในภาษา Visual Basic
AVisVBGetArrayParam	ทำสำเนาข้อมูลในอาเรย์ซึ่งแทนด้วยค่าที่ได้จากฟังก์ชัน AVisVBCreateArrayParam มาเก็บไว้ในอาเรย์ของ Visual Basic ที่
AVisVBGetMainWindow	ให้ค่าหมายเลขกำกับวินโดว์หลักของโปรแกรมที่พัฒนาด้วย Visual Basic
AVisVBSetMainWindowCaption	ตั้งข้อความหัวเรื่องของโปรแกรมที่พัฒนาจาก Visual Basic
AVisLoadComponent	เรียกองค์ประกอบซึ่งเป็นโปรแกรมบน Microsoft Windows มาทำงาน
AVisUnloadComponent	ยุติการทำงานของโปรแกรมองค์ประกอบ(terminate program)

ฟังก์ชันในกลุ่มนี้จะเป็นฟังก์ชันที่ช่วยให้ผู้พัฒนาองค์ประกอบด้วย Microsoft Visual Basic พัฒนาองค์ประกอบได้ง่ายขึ้น

AVISDB

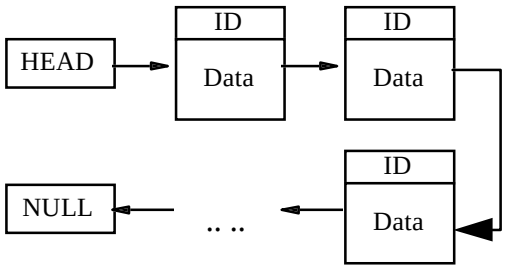
AVISDB (Algorithm Visualization Data Base) เป็นส่วนประกอบของ AVISEXEC ที่จัดเก็บข้อมูลทั้งหมดที่ต้องร่วมกันใช้ระหว่างส่วนประกอบย่อยต่างๆ นอกจากเป็นที่จัดเก็บข้อมูลแล้ว AVISDB จะให้บริการการสืบค้น แก้ไข เพิ่ม และลบข้อมูลเหล่านี้แก่ส่วนประกอบส่วนอื่นด้วย การแยกส่วนข้อมูลออกเป็นหน่วยย่อยต่างหากจากส่วนประกอบอื่นๆ มีข้อดีคือเราสามารถแยกการพัฒนาวิธีการจัดเก็บข้อมูลออกจากการพัฒนาส่วนประกอบอื่นๆ ทำให้สามารถปรับปรุงแก้ไขโครงสร้างข้อมูลให้มีประสิทธิภาพมากขึ้นโดยไม่กระทบกับส่วนประกอบอื่นในระบบ

ข้อมูลที่เก็บไว้ใน AVISDB ประกอบดฃวยรายละเอียดต่างๆของตัว องค์ประกอบที่ถูกนำเข้าสู่ระบบ อันได้แก่

1. หมายเลขประจำองค์ประกอบ (ID) ซึ่งเป็นค่าเฉพาะตัวองค์ประกอบที่ไม่ซ้ำกัน องค์ประกอบที่ขอใช้บริการใดๆผ่านทาง AVISAPI ต้องแสดงค่า ID ของตัวเองเสมอ
2. หมายเลขประจำวินโดว์ขององค์ประกอบ (Window Handle) ซึ่งเป็นหมายเลขที่ส่วนประกอบอื่นๆของ AVISEXEC ต้องใช้ เมื่อต้องการขอใช้บริการของ Microsoft Windows API
3. ค่าสถานะการทำงานขององค์ประกอบ ค่านี้ถูกเปลี่ยนแปลงโดยส่วน AVISROUTER AVISSYNC และ AVISERR

เนื่องจากโครงสร้างที่ใช้เก็บข้อมูลของ AVISDB สามารถถูกเพิ่ม ลบ และแก้ไขได้ในขณะทำงาน การจัดเก็บข้อมูลดังกล่าวจะอยู่รูปแบบของโครงสร้างข้อมูลแบบรายการโยง (linked list) ดังแสดงในรูปที่ 3 ข้อมูลประจำองค์ประกอบต่างๆจะถูกเก็บอยู่ในรายการโยง การค้นหาข้อมูลที่ต้องการจะใช้ ID เป็นตัวค้นหา [1]

การเพิ่มและลบข้อมูลใน AVISDB นั้นเกิดขึ้นจากการลงและการถอนทะเบียนของตัวองค์ประกอบที่ถูกนำเข้าสู่หรือนำออกจากระบบ โดยใช้ฟังก์ชัน `AvisRegisterComponent` และ `AVisUnregisterComponent` ของ AVISAPI ตามลำดับ



รูปที่ 3 โครงสร้างข้อมูลของ AVISDB

AVISROUTER

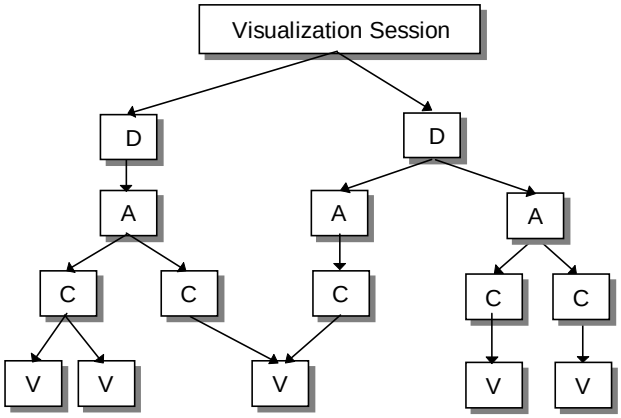
องค์ประกอบต่างๆที่ถูกนำเข้าสู่ระบบ เพื่อประกอบกันเป็นบทการจินตทัศน์นั้น จะมีการรับส่งข้อความคำสั่งและข้อมูลซึ่งกันและกัน เพื่อแยกองค์ประกอบให้เป็นอิสระจากกัน องค์ประกอบเหล่านี้จะไม่ส่งข้อมูลถึงกันและกันโดยตรง แต่จะใช้บริการของ AVISROUTER ในการรับส่งข้อความ แล้วผลักภาระให้ระบบเป็นผู้จัดการความสัมพันธ์ของการรับส่งข้อความดังกล่าว AVISROUTER มีหน้าที่ในการส่งผ่านข้อความไปยังจุดหมายที่มีความสัมพันธ์กับองค์ประกอบที่ใช้บริการ ควบคุมไม่ให้เกิดการซ้อนทับกันของข้อความที่จัดส่ง และรวมถึงการรับประกันว่าในกรณีที่ส่งให้ผู้รับหลายราย ผู้รับทุกๆรายจะได้รับข้อความที่ส่งพร้อมๆกัน

ความสัมพันธ์ของการรับส่งข้อความดังกล่าวนี้ จะบรรยายด้วยกราฟทางเดินของข้อความ (Message Routing Graph) ดังตัวอย่างในรูปที่ 4 ระบบจินตทัศน์อัลกอริทึมในงานวิจัยนี้แบ่งการเชื่อมต่อองค์ประกอบต่างๆเหล่านี้ออกเป็นสองประเภทคือ การเชื่อมต่อไปองค์ประกอบอื่นเพื่อขอข้อมูล (Input route : เส้นเชื่อมที่มีลูกศรเข้าหาองค์

---

[1] เพื่อการเข้าถึง node ต่างๆในรายการโยงได้ทันทีเมื่อต้องการอ้างอิงข้อมูล ค่าของ ID ประจำ node นั้นก็คือ ที่อยู่เริ่มต้นของหน่วยความจำของ node นั้น

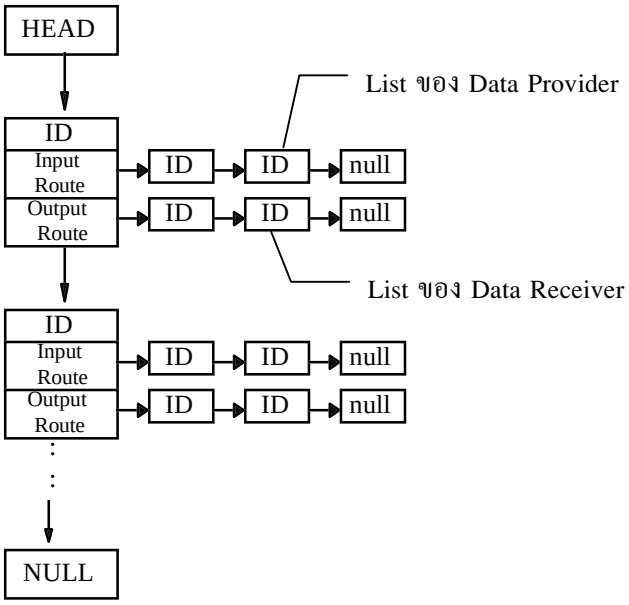
ประกอบ) และ การเชื่อมต่อไปองค์ประกอบอื่นเพื่อส่งข้อมูล (Output route : เส้นเชื่อมที่มีลูกศรออกจากองค์ประกอบ)



D : Data generator, A : Algorithm, C : Converter, V : View

รูปที่ 4 ตัวอย่าง Message Routing Graph ของบทการจินตทัศน์หนึ่ง

Message Routing Graph นี้ถูกเก็บในระบบโดยใช้ adjacency linked list ซึ่งสามารถเพิ่ม และลดได้ในขณะทำงานอย่างมีประสิทธิภาพ โดยแต่ละองค์ประกอบนั้นจะเก็บรายการโยงของผู้ส่งข้อความ และอีกหนึ่งรายการโยงสำหรับผู้รับข้อความขององค์ประกอบนั้นๆ ดังแสดงในรูปที่ 5 ข้อมูลของแต่ละ node ในรายการนี้จะเก็บเฉพาะ ID ขององค์ประกอบเท่านั้น เมื่อต้องการรายละเอียดขององค์ประกอบนั้นๆก็สามารถสอบถามได้จากบริการของ AVISDB



รูปที่ 5 โครงสร้างข้อมูลที่ใช้เก็บ Message Routing Graph

Message Routing Graph จะถูกสร้างและเปลี่ยนแปลงภายใต้การควบคุมของ AVISSESSIONCONTROL เมื่อนำองค์ประกอบเข้าหรือออกจากระบบ เพื่อเชื่อมความสัมพันธ์ และเพื่อลบความสัมพันธ์ระหว่างองค์ประกอบต่างๆ โดยใช้ฟังก์ชัน `AVisBind` และ `AVisUnbind` ของ AVISAPI ตามลำดับ

การเรียกใช้บริการส่งผ่านข้อความของ AVISROUTER นั้นกระทำได้โดยใช้ฟังก์ชัน AVisInputRequest และ ฟังก์ชัน AVisOutputNotify การทำงานของทั้งสองฟังก์ชันเหมือนกันในแง่ที่ว่าฟังก์ชันทั้งคู่มีไว้เพื่อส่งข้อความคำสั่งไปยังองค์ประกอบอื่นๆ จะต่างกันก็ตรงที่ฟังก์ชัน AVisInputRequest นั้นจะส่งข้อความไปยังองค์ประกอบผู้ส่ง ในขณะที่ฟังก์ชัน AVisOutputNotify จะส่งข้อความไปยังทุกองค์ประกอบผู้รับ

ในกรณีของการเรียกใช้ฟังก์ชัน AVisOutputNotify หากมีองค์ประกอบผู้รับหลายรายที่มีช่องทางต่อจากองค์ประกอบที่ส่งข้อความ AVISROUTER จะต้องส่งข้อความไปยังองค์ประกอบผู้รับต่างๆเหล่านั้นพร้อมๆกัน ทั้งนี้เพื่อให้ผู้รับทุกๆตัวตอบสนองการทำงานตามข้อความที่ส่งไปพร้อมๆกัน (หรือเกือบพร้อมกัน) จึงเสมือนกับการที่องค์ประกอบผู้ส่งสั่งให้องค์ประกอบผู้รับทำงานแบบขนาน (Parallel Call) โดยไม่ต้องรอให้ผู้รับทำงานเสร็จทีละรายๆ ยดยมีขัษนตอนการจัดการการสัศงงานมบบขนานดังนีษ

```
AVisOutputNotify( compID, message )
{
  AVisSetWaitCount( # of data receivers of compID )
  for each data receiver i of compID {
    AVisSendMessage( i, message )
  }
  AVisWaitforReply()
}

AVisSendMessage( compID, message )
{
  wait until compID is ready to receive
  SendMessage( compID, message ) /* Windows API */
}
AVisWaitforReply (  )
{
  while ( nWaitCount > 0 ) {
    YieldControl()
  }
}
```

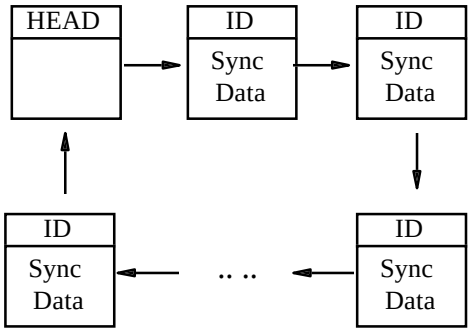
จากฟังก์ชันที่แสดงข้างบนนี้ มีการใช้ semaphore ชื่อว่า nWaitCount เพื่อเก็บจำนวนองค์ประกอบที่ได้รับข้อความคำสั่งแล้วยังทำงานไม่เสร็จ นั่นคือ nWaitCount จะถูกตั้งค่าเท่ากับจำนวนผู้รับข้อความคำสั่งเสียก่อน แล้วจึงทยอยส่งข้อความ จากนั้นเข้าสู่วงวนรอจนกว่า nWaitCount จะมีค่าเป็นศูนย์ (อนึ่งระหว่างการรอนั้นจะเรียกฟังก์ชัน YieldControl เพื่อเปิดโอกาสให้โปรแกรมอื่นๆในระบบได้ทำงาน) องค์ประกอบผู้รับตัวใดที่ทำงานเสร็จก็จะลดค่าของ nWaitCount ลงหนึ่งโดยอัตโนมัติภายใต้การควบคุมการทำงานของ AVISVBX (ที่เป็นตัวควบคุมการจินตทัศน์ที่มีกำกับองค์ประกอบต่างๆ) เมื่อ nWaitCount มีค่าเป็นศูนย์ก็แสดงว่าการสั่งการผ่านการเรียกใช้ฟังก์ชัน AVisVBOutputNotify ได้ทำงานเสร็จเรียบร้อย นี้จึงเปรียบเสมือนกับการเรียกใช้โปรแกรมย่อยในองค์ประกอบผู้รับข้อความต่างๆแบบขนาน

นอกจากนี้การส่งข้อความต่างๆใน AVISROUTER ยังรับประกันว่าองค์ประกอบที่รับข้อความ จะต้องพร้อมที่รับข้อความ นั่นคือไม่ได้อยู่ในขั้นตอนการทำงานเพื่อข้อความอื่นๆอยู่ หรืออีกนัยหนึ่งคือการป้องกันการส่งข้อความซ้อนทับกัน การควบคุมเงื่อนไขดังกล่าวนี้กระทำได้โดยการสร้างฟังก์ชัน AVisSendMessage (แสดงไว้ข้างบน) ขึ้นใช้แทนการใช้ฟังก์ชันการส่งข้อความของ Windows โดยตรง โดยใน AVisSendMessage นี้จะมีการตรวจสอบความพร้อมขององค์ประกอบที่จะรับข้อความก่อนที่จะส่ง โดยประสานการทำงานกับ AVISVBX ประจำองค์ประกอบนั้นๆเพื่อตรวจสอบสภาวะดังกล่าว

AVISSYNC

ในระบบจินตทัศน์อัลกอริทึมนั้น ผู้ใช้งานอาจต้องการสังเกตการทำงานของหลายๆอัลกอริทึมในเวลาเดียวกัน ทั้งนี้เพื่อเปรียบเทียบพฤติกรรมของอัลกอริทึม อันรวมถึงเวลาการทำงานเชิงเปรียบเทียบของอัลกอริทึม ดังนั้นระบบจะต้องมีส่วนให้บริการการประสานจังหวะการทำงาน เพื่อให้แต่ละอัลกอริทึมมีโอกาสได้ทำงานอย่างยุติธรรม เนื่องจากสภาพปฏิบัติการไมโครซอฟต์วินโดวส์ทำงานแบบหลายภารกิจในลักษณะร่วมกันทำงาน (corporative multitasking หรือที่เรียกว่า non pre-emptive) นั่นคือโปรแกรมต่างๆในระบบจะต้องคอยแบ่งเวลาการทำงานให้กันและกัน (หากผู้ใดไม่ยอมปล่อยให้ผู้อื่นทำงานผู้อื่นก็ทำงานไม่ได้) ส่วนให้บริการนี้คือ AVISSYNC ซึ่งเป็นส่วนบริการส่วนหนึ่งที่องค์ประกอบอัลกอริทึมขอใช้บริการ เพื่อรับประกันว่าเวลาการทำงานสัมพัทธ์ของอัลกอริทึมต่างๆเป็นไปอย่างถูกต้อง

โครงสร้างข้อมูลที่ใช้ประกอบการประสานจังหวะการทำงานของอัลกอริทึมคือ Synchronization list (ดูรูปที่ 6) ซึ่งสร้างแบบรายการโยงแบบวน (circular linked list) ทั้งนี้เพื่อให้สอดคล้องกับวิธีการจัดลำดับการทำงานขององค์ประกอบอัลกอริทึม ที่ใช้วิธี round robin (นั่นคือวิธีจัดลำดับการทำงานแบบผลัดกันทำ) เพื่อให้โอกาสทุกอัลกอริทึมได้ทำงานโดยเท่าเทียมกัน องค์ประกอบอัลกอริทึมจะถูกเพิ่มเข้าหรือลบออกจาก synchronization list เมื่อองค์ประกอบนั้นมีการเรียกใช้บริการ `AVisOpenSync` หรือ `AVisCloseSync` ของ AVISAPI ตามลำดับ



รูปที่ 6 โครงสร้างข้อมูลของ Synchronization list

นอกจาก AVISSYNC จะมีหน้าที่ในการประสานจังหวะการทำงานแล้วนั้น AVISSYNC ยังมีหน้าที่ในการควบคุมความเร็วในการทำงานของอัลกอริทึมได้ ตั้งแต่การให้ทำงานทีละหนึ่งคำสั่งพื้นฐาน (single step) หยุดการทำงานชั่วคราว และทำงานตามปกติ ลักษณะการทำงานเหล่านี้เรียกว่าภาวะการทำงานของการประสานจังหวะ (synchronization mode) อันเป็นค่าที่เก็บใน AVISSYNC ที่ผู้ใช้กำหนดผ่าน AVISSESSIONCONTROL การจินตทัศน์จะมีสถานะในการหยุดรอ หรือ ทำงานต่อ (visualization status) ทั้งนี้ขึ้นกับภาวะการทำงานและการประสานจังหวะการทำงานในขณะนั้น ซึ่งทั้งหมดนี้อยู่ภายใต้การควบคุมของ AVISSYNC

AVISERR

เนื่องจากระบบจินตทัศน์อัลกอริทึมที่ออกแบบและพัฒนาขึ้นนี้ทำงานบน Microsoft Windows ซึ่งมีลักษณะการทำงานของโปรแกรมหลายโปรแกรมแบบร่วมกันทำงาน ดังนั้นหากเกิดความผิดพลาดใดๆกับองค์ประกอบอาจเป็นผลให้การทำงานของทั้งระบบหยุดชะงักไปได้ AVISERR เป็นหน่วยพิเศษภายใน AVISEXEC ที่มีหน้าที่คอยตรวจสอบว่า AVISCOMPONENT ต่างๆ รวมทั้ง AVISSESSIONCONTROL ยังทำงานอยู่ในระบบอย่างปกติหรือไม่

โดยจะตรวจสอบข้อผิดพลาดของการที่ AVISCOMPONENT หรือ AVISSESSIONCONTROL เลิกการทำงานไปโดยไม่ได้ถอนทะเบียนออกจากระบบหรือไม่ หากเกิดเหตุการณ์ดังกล่าว จะมีผลต่อระบบดังนี้

1. ทำให้โครงสร้างข้อมูลภายใน AVISDB, AVISROUTER, AVISSYNC ไม่ตรงกับสภาพความเป็นจริง
2. หากข้อผิดผลาดเหล่านี้เกิดขึ้นกับผู้รับข้อความขณะทำงานตามข้อความที่ได้รับ จะทำให้ผู้ส่งข้อความต้องรออย่างไม่มีที่สิ้นสุดเพราะกลไกของการสั่งการแบบขนาน ป้องกันไม่ให้ผู้ส่งข้อความทำงานอื่นจนกว่าผู้รับข้อความจะตอบรับข้อความกลับมาหมดทุกราย
3. หากองค์ประกอบอัลกอริทึมที่ขอใช้บริการการประสานจังหวะหยุดทำงานโดยไม่แจ้งให้ AVISSYNC ทราบ ทำให้ AVISSYNC ไม่สามารถส่งต่อการทำงานให้กับองค์ประกอบอัลกอริทึมอื่นๆในระบบได้ ยังผลให้เกิดการหยุดรออย่างไม่สิ้นสุดขององค์ประกอบอัลกอริทึมอื่นๆเหล่านั้น

เหตุการณ์เช่นนี้จะเกิดขึ้นบ่อยมากในช่วงของการพัฒนาองค์ประกอบต่างๆระหว่างการทดสอบ หากผู้พัฒนาเขียนโปรแกรมผิดพลาด อาจเกิด abnormal termination โดยที่ AVISSESSIONCONTROL ไม่รับรู้ได้ สำหรับรายละเอียดของ AVISERR (ซึ่งเกี่ยวข้องกับสภาพปฏิบัติการไมโครซอฟต์วินโดวส์มาก) จะขอละไว้ในบทความนี้ และจะได้นำเสนอในบทความอื่นต่อไป

สรุป

บทความนี้ได้นำเสนอโครงสร้างของหน่วยบริหารการจินตทัศน์ (AVISEXEC) อันเป็นแก่นกลางของระบบจินตทัศน์อัลกอริทึม ซึ่งทำงานบนสภาพปฏิบัติการไมโครซอฟต์วินโดวส์ หน่วยบริหารนี้ให้บริการทางด้านการส่งผ่านข้อความ (AVISROUTER) การประสานจังหวะการทำงานของอัลกอริทึมต่างๆ (AVISSYNC) การสอบถามข้อมูลต่างๆของสภาพการจินตทัศน์ (AVISDB) และการจัดการความผิดพลาดในระบบ (AVISERR) โดยผ่านชุดเชื่อมประสานกับโปรแกรมประยุกต์ (AVISAPI) บทการจินตทัศน์หนึ่งๆประกอบด้วยองค์ประกอบต่างๆจำแนกได้เป็นสี่ประเภทคือ ตัวสร้างข้อมูล อัลกอริทึม ตัวแปลง และมุมมอง หน่วยบริหารนี้ถูกสร้างขึ้นเป็นคลังโปรแกรมเชื่อมโยงแบบพลวัต (AVISDLL) และเป็นตัวควบคุมของระบบพัฒนาโปรแกรมวิชวลเบสิก (AVISVBX) ที่มีประจำแต่ละองค์ประกอบ โดยทั้งสองส่วนนี้จะทำงานประสานกัน ส่งผลให้การพัฒนาส่วนต่างๆของระบบจินตทัศน์อัลกอริทึมกระทำได้ง่ายโดยใช้ภาษาวิชวลเบสิก และระบบอื่นๆที่สนับสนุนตัวควบคุมดังกล่าว

เอกสารอ้างอิง

1. S. Prasitjutrakul and W. Watcharawittayakul, Algorithm Visualization : a Revisit to Sorting Algorithm, Thai Journal of Development Administration, Vol. 33, No. 2., pp. 193-201., April-June 1993.
2. S. Prasitjutrakul and W. Watcharawittayakul, Algorithm Visualization System : An Overview of Internal Structure, Proc. of Third ASEAN Regional Seminar on Microelectronics and Information Technology, Bangkok, August 1994.
3. สมชาย ประสิทธิ์จูตระกูล และ วิทยา วัชระวิทยากูล, ระบบจินตทัศน์อัลกอริธึมภายใต้สภาพปฏิบัติการไมโครซอฟต์วินโดวส์, การประชุมทางวิชาการคอมพิวเตอร์ 2537
4. Kenneth C. Knowlton. L6:Bell Telephone Laboratories Low-Level Linked List Language, two 16mm black and white sound film,1966

5. Kellog S. Booth, PQ Trees, 16mm color silent film,12 minutes,1975

6. Ronal M. Baecker and David Sherman, Sorting Out Sorting, 16mm color sound film, 30 minutes,1981

7. Brad A. Myers, “Incense: A System for Displaying Data Structures,” Computer Graphics, Vol 17, No.3,115-125, July1983

8. David B. Baskerville, “Graphic Presentation of Data Structure in the DBX Debugger,” Report No.UCB/CSD 86/260, University of California at Berkeley, Berkeley, CA, October 1985

9. Thomas G. Moher, PROVIDE: a Process Visualization and Debugging Environment, Technical Report, University of Illinois at Chicago, Chicago, IL July 1985

10.Edward Yarwood, Toward Program Illustration, M.Sc. Thesis, Dept. of Computer Science, University of Toronto, Toronto, ON, 1974.

11.James M. De Boer, A System for the Animation of Micro-PL/I Programs, M.Sc. Thesis, Dept. of Computer Science, University of Torinto, Toronto, ON, 1974.

12.Marc H. Brown, Algorithm Animation, The MIT Press, Cambridge, MA, 1988

13.Marc H. Brown, Zeus: A System for Algorithm Animation and Multi View Editing, Proc. IEEE Workshop Visual Language, IEEE CS Press, CA, pp. 27-39, 1991.

14.Stasko, Tango: A Framwork and System for Algorithm Animation, Computer,Vol. 23,No. ,pp.27-399, Sept. 1990.

15.Roman et al.,Pavene: A System for Declarative Visualization of Concurrent Computations, J. Visual Language and Computing, Vol. 3, No. 2,pp. 161-163,June 1992.

16.Microsoft, Microsoft Visual Basic Programming System for Windows Programmer’s Guide version 3.0, Microsoft Inc., 1993.