บทที่ 5

การโปรแกรมจีนเนติก
Genetic Programming

ในปี 1992 John Koza ได้เขียนบทความเรื่อง "Genetic Programming: On the Programming of Computers by Means of Natural Selection" Koza เป็นผู้ที่อุทิศตนให้กับงานทางด้านโครงสร้างต้นไม้ (tree structure) ซึ่งถือว่าเป็นจุดเริ่มต้นและก่อให้เกิดความสนใจของงานทางด้านการโปรแกรมจีนเนติกในเวลาต่อมาการโปรแกรมจีนเนติก [Koza, 1992] (genetic programming หรือ GP) คือจีนเนติกอัลกอริทึมในรูปแบบพิเศษ GP มีขั้นตอนการวิวัฒนาการในรูปจีโนไทป์ (genotype) สิ่งที่ GP แตกต่างไปจาก GA คือรูปแบบของโครโมโซม ใน GA โครโมโซมมีรูปแบบเป็นสายอักขระของบิต (bit string) ในขณะที่ GP มีโครโมโซมในรูปรหัสโปรแกรมบนโครงสร้างต้นไม้ จุดประสงค์หลักของ GP คือการทำให้โปรแกรมคอมพิวเตอร์มีวิวัฒนาการไปสู่คำตอบที่ต้องการได้ ในแต่ละรุ่น โครโมโซมโปรแกรมจะถูกประเมินเพื่อวัดประสิทธิภาพในโดเมนของปัญหานั้นๆ ผลจากการวัดประสิทธิภาพของโปรแกรมคอมพิวเตอร์ที่มีวิวัฒนาการ จะถูกใช้ในการกำหนดค่าความเหมาะสมของโครโมโซมโปรแกรมนั้นๆ ปฏิบัติการทางสายพันธุ์ใน GP ยังคงเป็นคล้ายคลึงกันกับ GA ไม่ว่าจะเป็นการทำมิวเทชันและครอสโอเวอร์ เนื้อหาในบทนี้จะกล่าวถึงรายละเอียดของ GP ในเบื้องต้น (อ้างอิงจาก [Engelbrecht, 2002]) ดังต่อไปนี้

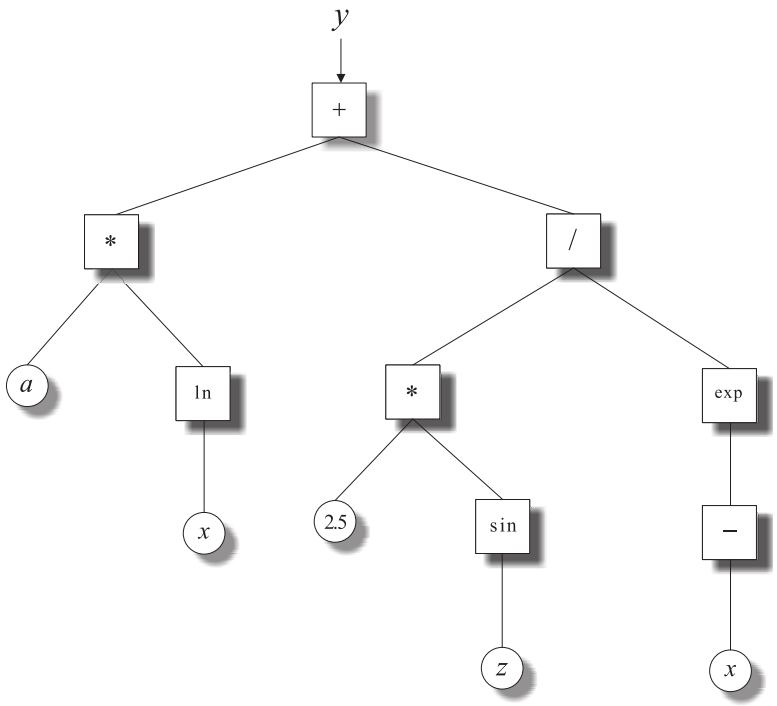
5.1 ตัวแทนโครโมโซม
Chromosome Representation

รูปแบบโครโมโซมของ GP เป็นส่วนที่แตกต่างไปจากโครโมโซมของ EC แบบอื่นๆ โครโมโซมของ GP ประกอบไปด้วยชิ้นส่วนของโปรแกรมคอมพิวเตอร์ ดังนั้นสำหรับ GP แล้ว วิวัฒนาการที่เกิดขึ้นจะทำให้โปรแกรมคอมพิวเตอร์ปรับตัวไปสู่คำตอบที่ต้องการ เราจะเรียกตัวแทนโครโมโซมของโปรแกรมคอมพิวเตอร์ดังกล่าวว่า*โครโมโซมโปรแกรม* รูปที่ 5.1 แสดงตัวอย่างโครโมโซมโปรแกรมของ GP ดังต่อไปนี้

$$\mathrm{y} = \mathrm{a} * \ln(\mathrm{x}) + 2.5 * \sin(\mathrm{z}) / \exp(-\mathrm{x})$$

โครโมโซมข้างต้นประกอบไปด้วยองค์ประกอบหลักๆ ดังต่อไปนี้

- **เซตปลายทาง** (terminal set) ประกอบไปด้วยตัวแปร (variable) และค่าคงที่ (constant) ทั้งหมด ตัวอย่างโครโมโซมโปรแกรมในรูปที่ 5.1 มีเซตปลายทางคือ $\{a, 2.5, x, z\}$ เป็นต้น โดยที่ $a, x, z \in \mathrm{R}$ องค์ประกอบของเซตปลายทางจะแสดงอยู่ในกรอบวงกลม



รูปที่ 5.1: ตัวอย่างโครโมโซมโปรแกรมของ $\mathtt{y} = \mathtt{a} * \ln(\mathtt{x}) + 2.5 * \sin(\mathtt{z}) / \exp(-\mathtt{x})$

- **เซตฟังก์ชัน** (function set) ประกอบไปด้วยฟังก์ชันที่สามารถใช้กับองค์ประกอบในเซตปลายทางได้ ตัวอย่างของฟังก์ชันเหล่านี้เช่น
  - ฟังก์ชันคณิตศาสตร์ $\{\sin, \cos, \tan, \exp, \ln, \log, \ldots\}$
  - ฟังก์ชันเลขคณิต $\{+, -, \times, \div, \ldots\}$
  - ฟังก์ชันบูลีน {AND,OR,NOT,XOR,...}
  - โครงสร้างการตัดสินใจ เช่น $\mathtt{IF} - \mathtt{THEN} - \mathtt{ELSE}$ หรือ $\mathtt{SWITCH} - \mathtt{CASE}$ เป็นต้น
  - โครงสร้างการวนรอบ เช่น $\mathtt{WHILE} \ldots \mathtt{DO}$ หรือ $\mathtt{FOR} \ldots \mathtt{DO}$ เป็นต้น

  ตัวอย่างเซตฟังก์ชันของโครโมโซมโปรแกรมในรูปที่ 5.1 คือ $\{+, -, *, /, \ln, \sin, \exp\}$ เป็นต้น องค์ประกอบของเซตฟังก์ชันจะแสดงอยู่ในกรอบสี่เหลี่ยม

เมื่อแต่ละส่วนของโครโมโซมโปรแกรมจากเซตปลายทางและเซตฟังก์ชันเชื่อมโยงกัน จะเรียกว่าเป็น*ไวยากรณ์* (grammar) ที่จำเป็นสำหรับใช้แก้ปัญหาที่ต้องการ โครงสร้างต้นไม้ (tree structure) มีความสะดวกในการแทนรูปแบบของโครโมโซมโปรแกรม องค์ประกอบต่างๆ ของเซตปลายทางจะอยู่ในส่วนบัพใบ (leaf node) และองค์ประกอบของเซตฟังก์ชันจะอยู่ในส่วนที่เหนือขึ้นไปจากบัพใบของโครงสร้างต้นไม้

แต่ละโครโมโซมใน GP เป็นตัวแทนของส่วนโปรแกรม ที่ซึ่งเป็นองค์ประกอบของปริภูมิโปรแกรม (program space) ปริภูมิโปรแกรมประกอบไปด้วยส่วนของโปรแกรมทั้งหมด ที่สามารถประกอบกันเป็นไวยากรณ์ที่ต้องการได้ จุดประสงค์หลักของ GP คือการค้นหาโปรแกรมในปริภูมิโปรแกรมที่ให้ผลการประมาณที่ดีที่สุดของ*โปรแกรมวัตถุประสงค์* (objective program)

โครงสร้างต้นไม้ของประชากรใน GP สามารถมีขนาดคงที่หรือแปรผันได้ สำหรับโครงสร้างต้นไม้ที่มีขนาดคงที่แล้ว ต้นไม้ของทุกโครโมโซมโปรแกรมจะมีระดับความลึกเท่ากันหมด รวมไปถึงการขยายของต้นไม้ส่วนย่อย (subtree) ด้วย ซึ่งจะมีการขยายไปจนถึงระดับความลึกสูงสุดหนึ่งๆ เท่านั้น ส่วนโครงสร้างต้นไม้ที่มีขนาดแปรผัน

ได้จะมีการกำหนดเพียงระดับความลึกสูงสุดเท่านั้น ในบางระบบ อาจจะมีการปรับระดับความลึกสูงสุดนี้ตามจำนวนรุ่นที่เพิ่มมากขึ้นก็ได้ โครงสร้างต้นไม้ที่มีขนาดแปรผันได้นี้เป็นที่นิยมใช้มากกว่าแบบขนาดคงที่

การกำหนดค่าเริ่มต้นให้กับประชากรจะทำโดยขั้นตอนการสุ่ม ภายใต้เงื่อนไขของระดับความลึกของโครงสร้างต้นไม้ ความหมายของโครโมโซมโปรแกรมที่ได้จะถูกแสดงด้วยไวยากรณ์ที่กำหนดขึ้น สำหรับแต่ละโครโมโซมโปรแกรม จุดราก (root) ของต้นไม้ (จุดเริ่มต้น) จะถูกสุ่มเลือกจากเซตฟังก์ชัน ตัวประกอบการแตกกิ่ง (branching factor) หรือจำนวนของโนดลูกของรากนั้นๆ รวมไปถึงของโนดที่ไม่ใช่โนดปลายทาง จะถูกเลือกขึ้นมาตามลักษณะของฟังก์ชันรากที่ได้สุ่มเลือกไว้ แต่ละโนดถัดไปจะถูกสุ่มเลือกค่าเริ่มต้นได้ทั้งจากเซตปลายทางหรือเซตฟังก์ชัน เมื่อไรก็ตามที่โครงสร้างต้นไม้มาถึงส่วนที่เป็นองค์ประกอบปลายทาง ส่วนนี้จะกลายเป็นบัพใบและไม่มีการแตกกิ่งเป็นโนดลูกอีกต่อไป

5.2 ฟังก์ชันค่าความเหมาะสม Fitness Function

ฟังก์ชันค่าความเหมาะสมหรือ fitness function สำหรับ GP จะมีความแตกต่างกันไปตามแต่ละปัญหา ฟังก์ชันดังกล่าวใช้ในการประเมินค่าความเหมาะสมของแต่ละโครโมโซมโปรแกรม โดยมักจะทำการทดสอบกับเป้าหมายตัวอย่างในโลกของปัญหาจริงๆ พร้อมทั้งใช้ค่าการวัดประสิทธิภาพของตัวอย่างนั้นๆ มาเป็นค่าความเหมาะสม ยกตัวอย่างโครโมโซมโปรแกรมในรูปที่ 5.1 อีกครั้ง สมมุติว่าเราไม่มีข้อมูลหรือความรู้ใดๆ เกี่ยวกับโครงสร้างของโปรแกรมที่ต้องการค้นหาเลย *ยกเว้น*เซตปลายทางและเซตฟังก์ชันที่กำหนดให้ และข้อมูลเชิงตัวเลขของฟังก์ชันหรือโปรแกรมที่ต้องการ ในที่นี้ข้อมูลอินพุตประกอบไปด้วยข้อมูลของ 3 ค่าของตัวแปร a x และ z (a เป็นค่าคงที่) ส่วนข้อมูลเป้าหมายคือค่าเชิงตัวเลขของ y การประเมินค่าของโครโมโซมโปรแกรมมีอยู่สองขั้นตอนคือ

- คำนวณเอาต์พุตของแต่ละโครโมโซมด้วยค่าเชิงตัวเลขของ a x และ z
- คำนวณค่าความผิดพลาดระหว่างค่าเอาต์พุตที่ได้กับค่าเป้าหมาย y

ตัวอย่างค่าการวัดประสิทธิภาพจากค่าความผิดพลาดที่นิยมใช้ในกรณีนี้คือค่าความผิดพลาดแบบกำลังสองเฉลี่ย (mean-squared error หรือ MSE) ซึ่งจะถูกใช้เป็นค่าความเหมาะสมต่อไป

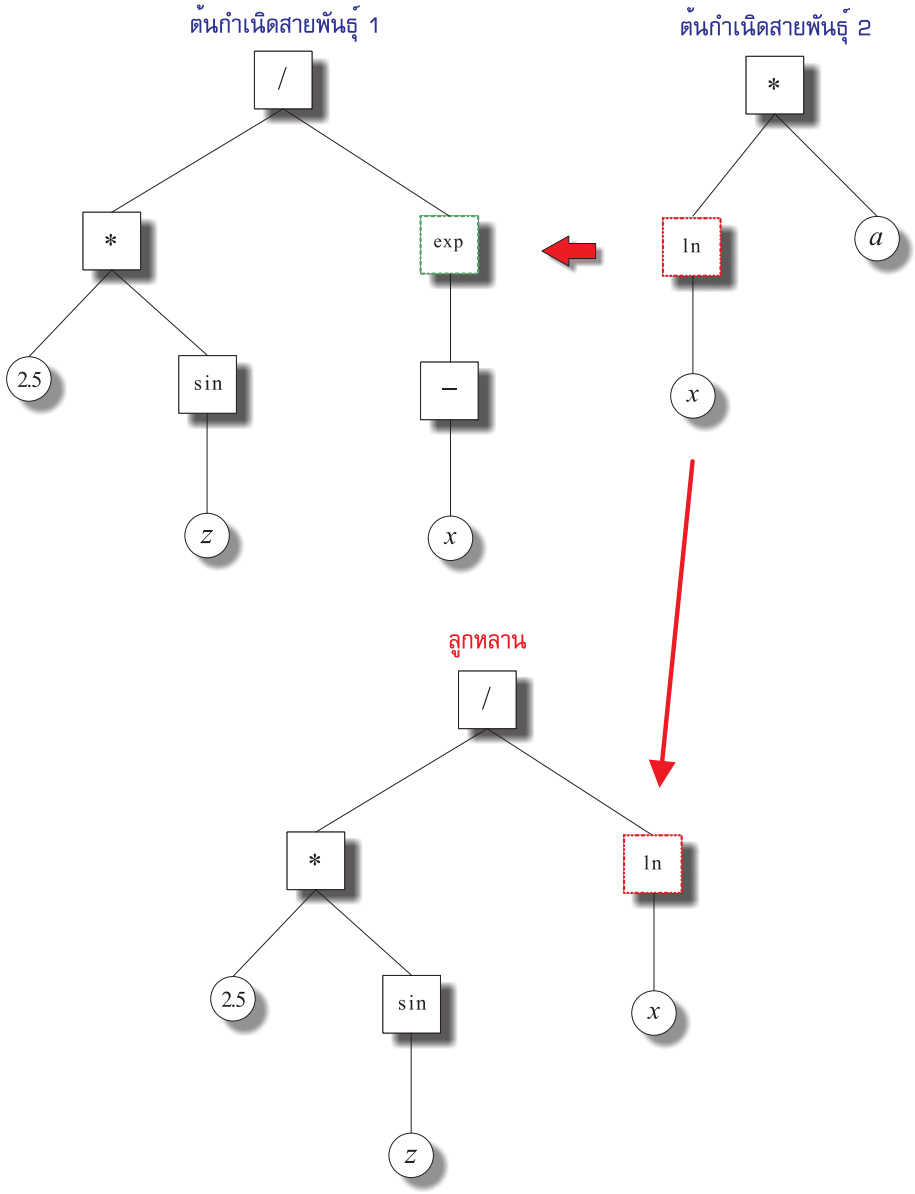
นอกเหนือไปจากการวัดค่าประสิทธิภาพของโครโมโซมโปรแกรมแล้ว ค่าความเหมาะสมยังสามารถใช้ในการลงโทษ (penalize) โครโมโซมที่มีโครงสร้างที่ไม่เหมาะสมได้ด้วย ยกตัวอย่างเช่น แทนที่จะทำการกำหนดระดับความลึกของโครงสร้างต้นไม้ไว้ค่าๆ หนึ่ง ค่าระดับความลึกนี้สามารถใช้เป็นเทอมสำหรับลงโทษเพื่อบวก (หรือลบ) ออกจากค่าความเหมาะสมจากฟังก์ชันความเหมาะสมก็ได้ ในทำนองเดียวกัน โครงสร้างต้นไม้ที่แผ่เป็นพุ่ม (bushy tree) อาจจะถูกใช้เป็นเทอมลงโทษได้ด้วยเช่นกัน

5.3 ปฏิบัติการทางสายพันธุ์ Genetic Operators

ปฏิบัติการทางสายพันธุ์ของ GP ประกอบไปด้วยทั้งครอสโอเวอร์และมิวเทชัน หลักการของปฏิบัติการทางสายพันธุ์ทั้งสองยังคงมีแนวคิดเหมือนกันกับการคำนวณเชิงวิวัฒนาการทั่วๆ ไป ดังรายละเอียดต่อไปนี้

5.3.1 ครอสโอเวอร์ Crossover

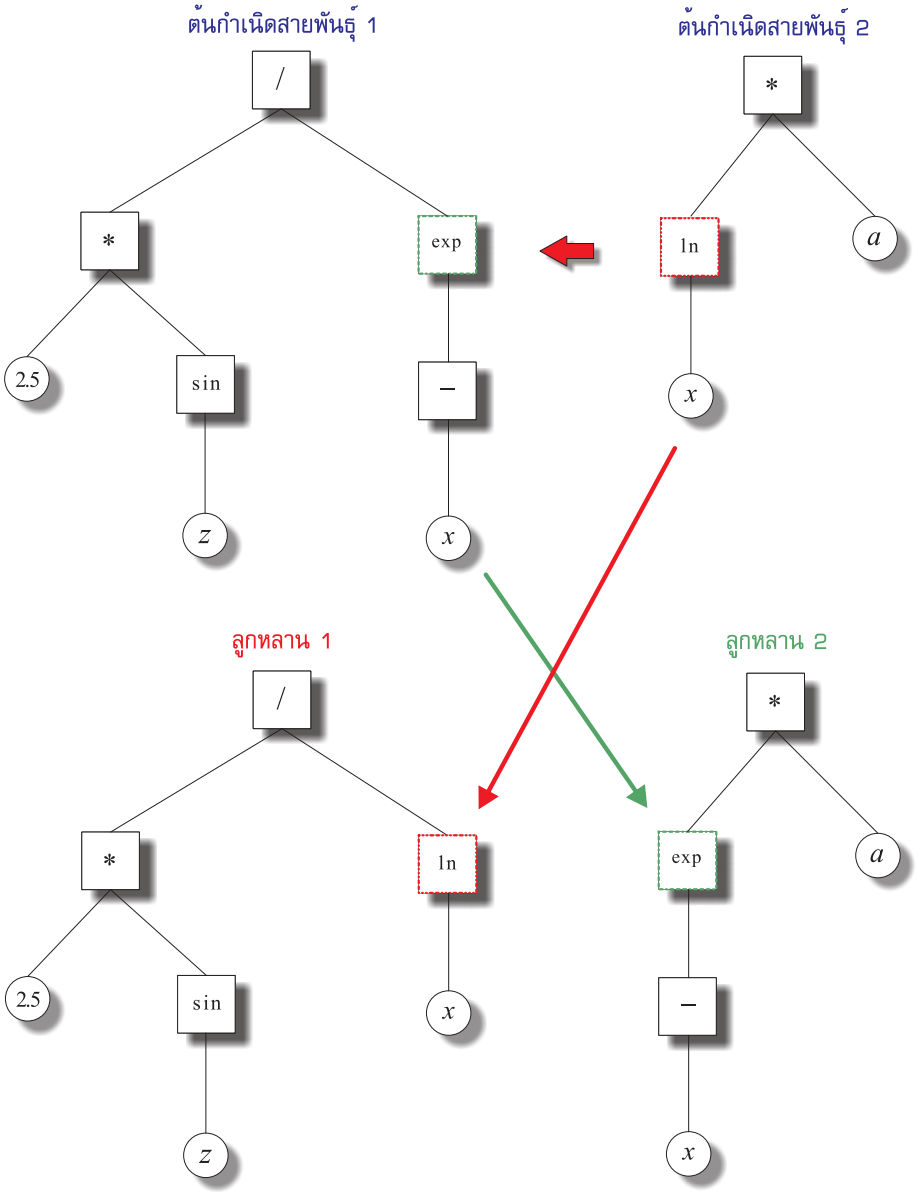
GP ทำการคัดเลือกต้นกำเนิดสายพันธุ์สำหรับทำครอสโอเวอร์จำนวน 2 โครโมโซม การทำครอสโอเวอร์ของ GP แบ่งออกได้เป็น 2 ประเภท ตามจำนวนของลูกหลานที่ได้



รูปที่ 5.2: ตัวอย่างการทำครอสโอเวอร์ให้กำเนิดลูกหลานโครโมโซมเดี่ยว

- **ครอสโอเวอร์ให้กำเนิดลูกหลานโครโมโซมเดี่ยว** โนดจะถูกสุ่มเลือกจากโครโมโซมของต้นกำเนิดสายพันธุ์ กระบวนการครอสโอเวอร์จะทำการ*แทนที่*ต้นไม้ส่วนย่อยของโนดนั้นๆ ด้วยต้นไม้ส่วนย่อยจากอีกต้นกำเนิดสายพันธุ์หนึ่ง ตัวอย่างของครอสโอเวอร์ดังกล่าวแสดงในรูปที่ 5.2 จะเห็นได้ว่า โครโมโซมลูกหลานที่ได้จะมีเพียงโครโมโซมเดียวเท่านั้น
- **ครอสโอเวอร์ให้กำเนิดลูกหลานโครโมโซมคู่** การทำครอสโอเวอร์แบบนี้จะคล้ายคลึงกันกับในวิธีแรก ข้อแตกต่างมีเพียงการ*แทนที่*ต้นไม้ส่วนย่อยจะเปลี่ยนเป็นการ*สลับ*ต้นไม้ส่วนย่อย ที่ได้จากแต่ละต้นกำเนิดสายพันธุ์ ดังแสดงในรูปที่ 5.3

การทำครอสโอเวอร์ใน GP เป็นอะไรที่ค่อนข้างตรงไปตรงมาและเห็นภาพได้ง่าย เพราะไม่ได้อยู่ในรูปแบบเชิงตัวเลข ในขณะที่การทำมิวเทชันก็มีลักษณะคล้ายๆ กัน ดังรายละเอียดต่อไปนี้



รูปที่ 5.3: ตัวอย่างการทำครอสโอเวอร์ให้กำเนิดลูกหลานโครโมโซมคู่

5.3.2 มิวเทชัน Mutation

ปฏิบัติการมิวเทชันใน GP ได้รับการพัฒนาไปมากมายหลายอย่าง วิธีการทำมิวเทชันที่ได้รับความนิยมใช้ขั้นตอนตามที่แสดงในรูปที่ 5.4 และ 5.5 ดังรายละเอียดต่อไปนี้

- **มิวเทชันของโนดฟังก์ชัน** (function node mutation) คือการทำมิวเทชันกับโนดที่ไม่ใช่บัพใบ หรือฟังก์ชันโนดนั่นเอง โนดดังกล่าวจะถูกสุ่มเลือกฟังก์ชันใหม่จากเซตฟังก์ชัน (ที่สมมูลกันในแง่ของจำนวนอินพุต หรือจำนวนกิ่งของต้นไม้ส่วนย่อย) เพื่อมาแทนที่ฟังก์ชันในโนดนั้นๆ ดังแสดงในรูปที่ 5.4-(ข)

- **มิวเทชันของโนดปลายทาง** (terminal node mutation) คือการทำมิวเทชันกับโนดที่เป็นบัพใบ หรือโนด

ปลายทางนั่นเอง โนดดังกล่าวจะถูกสุ่มเลือกใหม่จากเซตปลายทาง แล้วนำมาแทนที่โนดนั้นๆ รูปที่ 5.4-(ค)

- **มิวเทชันแบบสลับที่** (swapping mutation) คือการทำมิวเทชันกับฟังก์ชันโนดที่สุ่มได้ โดยการสลับที่อาร์กิวเมนต์ของฟังก์ชันนั้นๆ ดังแสดงในรูปที่ 5.4-(ง)
- **มิวเทชันแบบแตกกิ่ง** (grow mutation) คือการทำมิวเทชันกับโนดที่สุ่มได้ แล้วแทนที่โนดนั้นๆ ด้วยต้นไม้ส่วนย่อยที่ได้จากการสุ่มเช่นกัน รูปที่ 5.5-(จ) แสดงการแทนที่ของโนดด้วยต้นไม้ส่วนย่อยใหม่
- **มิวเทชันแบบเกาส์เซียน** (Gaussian mutation) ปกติแล้วมิวเทชันแบบนี้จะกระทำกับโนดที่เป็นค่าคงที่ (โนดถูกเลือกแบบสุ่ม) แล้วทำมิวเทชันโดยการเพิ่มค่าสุ่มแบบเกาส์เซียนกับค่าคงที่ในโนดนั้นๆ ดังแสดงในรูปที่ 5.5-(ฉ)
- **มิวเทชันแบบตัดกิ่ง** (truncate mutation) คือการสุ่มตัดโนดฟังก์ชันแล้วแทนที่ด้วยโนด (สุ่ม) ปลายทาง การทำมิวเทชันแบบนี้เป็นการทำให้โครโมโซมพรุน ดังแสดงในรูปที่ 5.5-(ช)

โดยปกติแล้ว โครโมโซมโปรแกรมที่จะถูกทำมิวเทชัน จะมีการกำหนดค่าความน่าจะเป็นในการทำมิวเทชันด้วย p_m นอกเหนือไปจากนั้นแล้ว ยังมีการกำหนดค่าความน่าจะเป็น p_n ให้กับการทำมิวเทชันของโนดภายในโครงสร้างต้นไม้อีกด้วย ค่าความน่าจะเป็น p_n ที่สูง จะมีผลให้การเปลี่ยนแปลงของยีนในโครโมโซมโปรแกรมนั้นสูงไปด้วย ในทางตรงกันข้าม ค่าความน่าจะเป็น p_m ที่สูง จะมีผลให้จำนวนโครโมโซมโปรแกรมที่จะต้องถูกกระทำมิวเทชันสูงไปด้วย การทำมิวเทชันของ GP มีข้อแตกต่างไปจากการทำมิวเทชันของ EC แบบอื่นๆ คือสามารถทำเพียงอย่างเดียวหรือหลายๆ อย่างด้วยกันได้

5.4 ซอฟต์แวร์และแหล่งข้อมูลอื่นๆ

ตลอดเวลาที่ผ่านมา GP ได้รับความสนใจมากมาย นอกเหนือไปจากเนื้อหาที่ได้รับการถ่ายทอดและพัฒนาอย่างต่อเนื่องแล้ว เทคโนโลยีซอฟต์แวร์ที่ก้าวหน้า ทำให้การพัฒนา GP บนคอมพิวเตอร์ส่วนบุคคลเป็นไปอย่างทรงประสิทธิภาพ ผู้อ่านสามารถติดตามข้อมูลต่างๆ ที่เกี่ยวกับ GP ได้จากเว็บไซต์หลัก

`http://www.genetic-programming.org/`

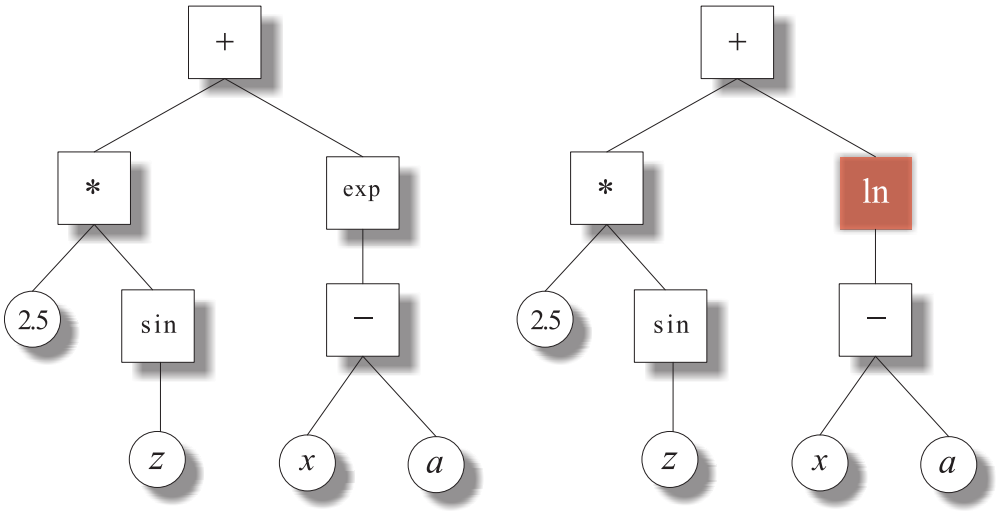
เว็บไซต์ดังกล่าวดูแลโดย John Koza ผู้ที่ทำให้ GP ได้รับความสนใจจนกระทั่งทุกวันนี้ ผู้อ่านสามารถติดตามข้อมูลความเคลื่อนไหวเกี่ยวกับเทคโนโลยีของ GP รวมไปถึงการพัฒนาซอฟต์แวร์ GP จากนักพัฒนาโปรแกรมทั่วทุกมุมโลก ตัวอย่างของซอฟต์แวร์ GP ที่ใช้ในหัวข้อนี้เป็นกล่องเครื่องมือที่ใช้กับชุดซอฟต์แวร์ MATLAB® เรียกว่า GPLAB ผู้อ่านที่สนใจสามารถค้นหาข้อมูลได้จากเว็บไซต์ต่อไปนี้ [Silva]

`http://gplab.sourceforge.net/`

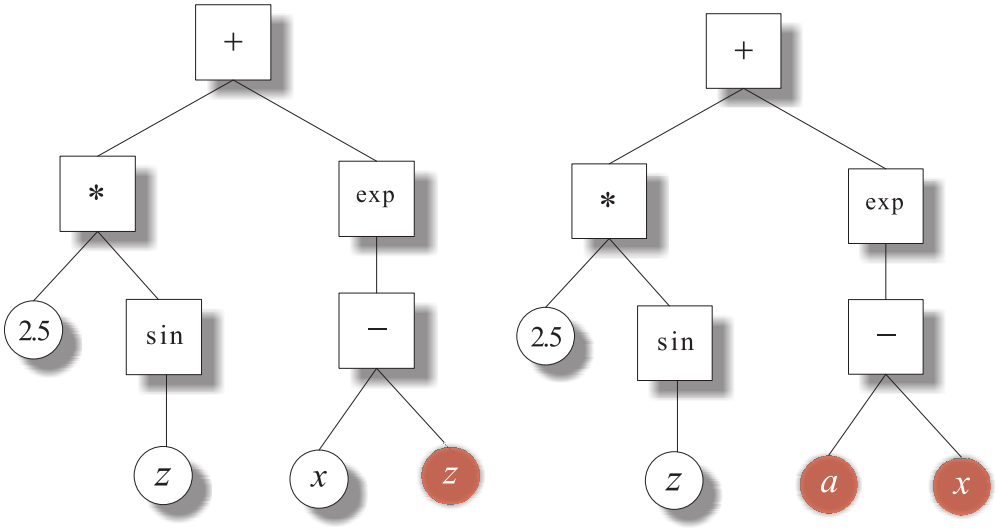
กล่องเครื่องมือข้างต้นเป็นชุดซอฟต์แวร์ที่ครบถ้วนในการใช้งาน GP บน MATLAB® นอกจากนั้นแล้วยังเป็นเวอร์ชันที่ใช้งานได้ฟรี และมีตัวอย่างการใช้งานอย่างเพียบพร้อม เหมาะแก่การเริ่มต้นในการศึกษาและพัฒนางานทางด้าน GP ตัวอย่างการทำงานของ GP จากชุดซอฟต์แวร์ดังกล่าวมีดังต่อไปนี้

■ **ตัวอย่างที่ 5.1** การหาฟังก์ชันการถดถอยด้วย GP

พิจารณาการหาฟังก์ชันการถดถอยของข้อมูลในตารางที่ 5.1 ข้อมูลดังกล่าวได้มาจากฟังก์ชัน $y = 2x^2$ ดังนั้นคำตอบของ GP ที่ต้องค้นหาคือฟังก์ชัน $y = f(x) = 2x^2$ นั่นเอง ปัญหาดังกล่าวนี้เรียกว่าเป็นปัญหาการหาฟังก์ชันการถดถอย (regression) ในปัญหาของ GP ค่าคู่อินพุต/เอาต์พุตสำหรับหาฟังก์ชันการถดถอยนั้นเรียกว่าค่าความเหมาะสมกรณี (fitness case) ในตัวอย่างพารามิเตอร์ที่สำคัญของ GP ถูกตั้งค่าดังต่อไปนี้



(ก) โครโมโซมดั้งเดิมก่อนทำมิวเทชัน (ข) มิวเทชันของโนดฟังก์ชัน

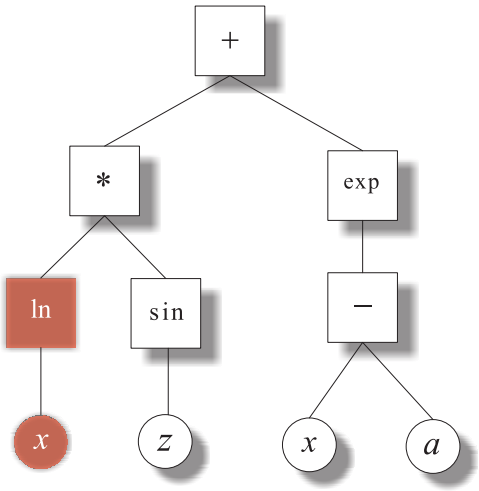


(ค) มิวเทชันของโนดปลายทาง (ง) มิวเทชันแบบสลับที่ (swapping)

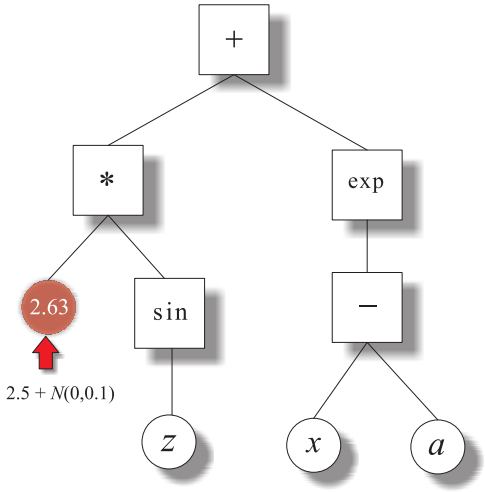
รูปที่ 5.4: ตัวอย่างการทำมิวเทชัน

- จำนวนประชากร = 10
- จำนวนรุ่น = 20
- ปฏิบัติการทางสายพันธุ์ = ครอสโอเวอร์และมิวเทชัน
- ฟังก์ชันเซต = {`plus`, `minus`, `times`, `sin`, `cos`, `log`}
- การคัดเลือกสายพันธุ์ = จัดการแข่งขัน
- การแทนที่ = แทนที่ประชากรทั้งรุ่น

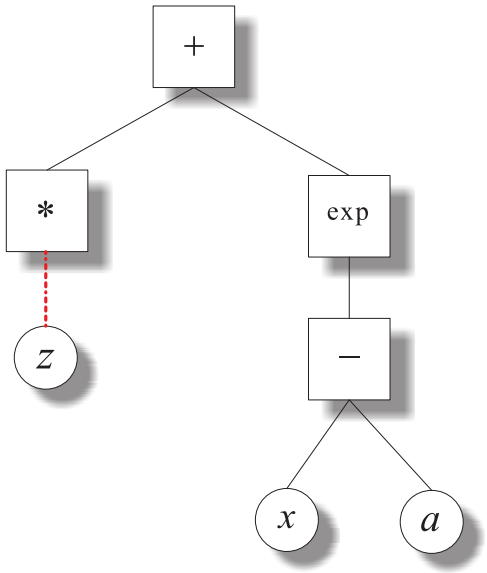
พิจารณาเมื่อ GP เริ่มต้นทำงาน (รุ่นที่ 0 เป็นค่าสุ่ม) รูปที่ 5.6-(ก) แสดงโครงสร้างต้นไม้ของโครโมโซมที่ดีที่สุดในกลุ่มประชากร ค่าฟังก์ชันที่อ่านได้จากโครงสร้างต้นไม้คือ $f_0(x) = x \times x = x^2$ ในขณะที่รูปที่ 5.6-(ข) และ (ค) ได้



(จ) มิวเทชันแบบแตกกิ่ง (growing)



(ฉ) มิวเทชันแบบเกาส์เซียน (Gaussian)

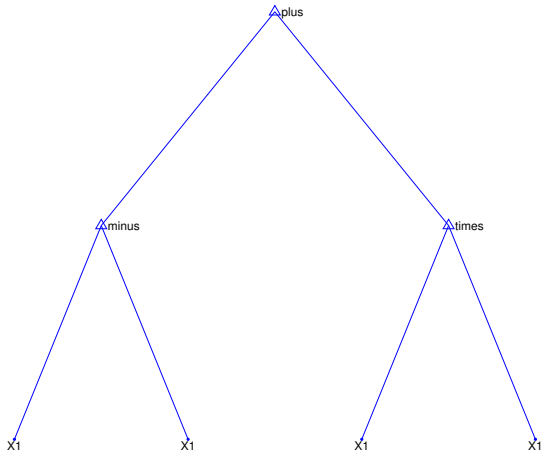


(ช) มิวเทชันแบบตัดกิ่ง (truncate)

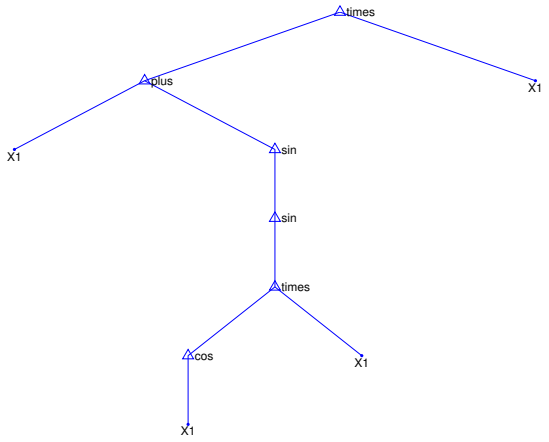
รูปที่ 5.5: ตัวอย่างการทำมิวเทชัน (ต่อ)

จากโครโมโซมที่ดีที่สุดของรุ่นที่ 15 และ 19 ที่ซึ่งทั้งสองฟังก์ชันอ่านค่าได้เป็น $f_{15}(x) = x(x + \sin(\sin x \cos x))$ และ $f_{19} = x(x+\sin x)$ ตามลำดับ GP ทำการค้นหาคำตอบและเจอคำตอบที่ถูกต้องในประชากรรุ่นที่ 20 ดังแสดงในรูปที่ 5.7 (อ่านค่าจากโครงสร้างต้นไม้ได้เป็น $(x + x) \times x = (2x) \times x = 2x^2$ เราจะสังเกตเห็นวิวัฒนาการของคำตอบได้ รูปที่ 5.8 แสดงกราฟจากฟังก์ชันที่ได้จากโครโมโซมที่ดีที่สุดในรุ่นประชากรต่างๆ ค่าความเหมาะสมของโครโมโซมที่ดีที่สุดในรุ่นต่างๆ แสดงในรูปที่ 5.9 (สังเกตการลู่เข้า) ส่วนกราฟในรูปที่ 5.10 แสดงกราฟค่าความน่าจะเป็นและความถี่ของปฏิบัติการทางสายพันธุ์ต่างๆ ■

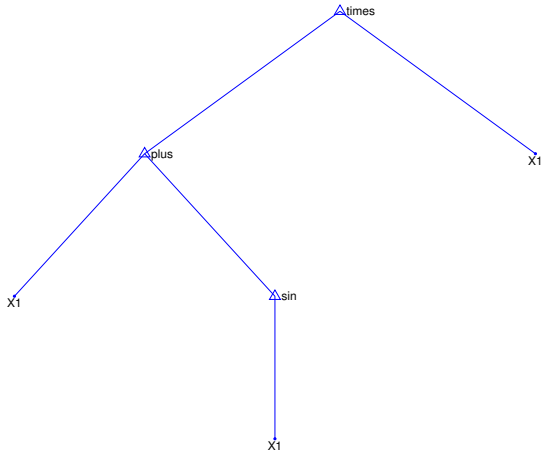
ตัวอย่างข้างต้นเป็นตัวอย่างง่ายๆ ที่แสดงขั้นตอนการทำงานของ GP ในการประยุกต์ใช้งานจริงนั้น เราต้องทำการ



(ก)



(ข)

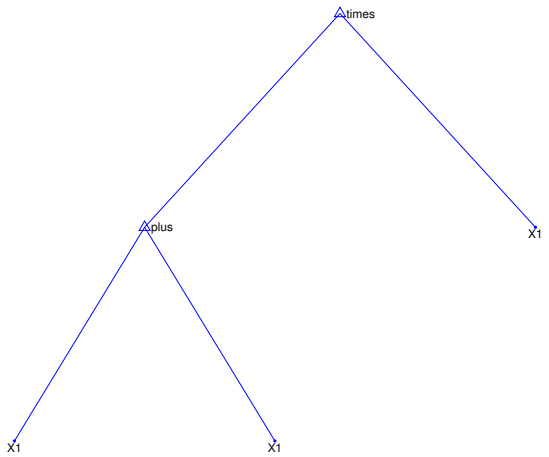


(ค)

รูปที่ 5.6: โครงสร้างต้นไม้ของฟังก์ชันที่อ่านได้จากโครโมโซมที่ดีที่สุด (ก) รุ่นที่ 0 (ข) รุ่นที่ 15 (ค) รุ่นที่ 19

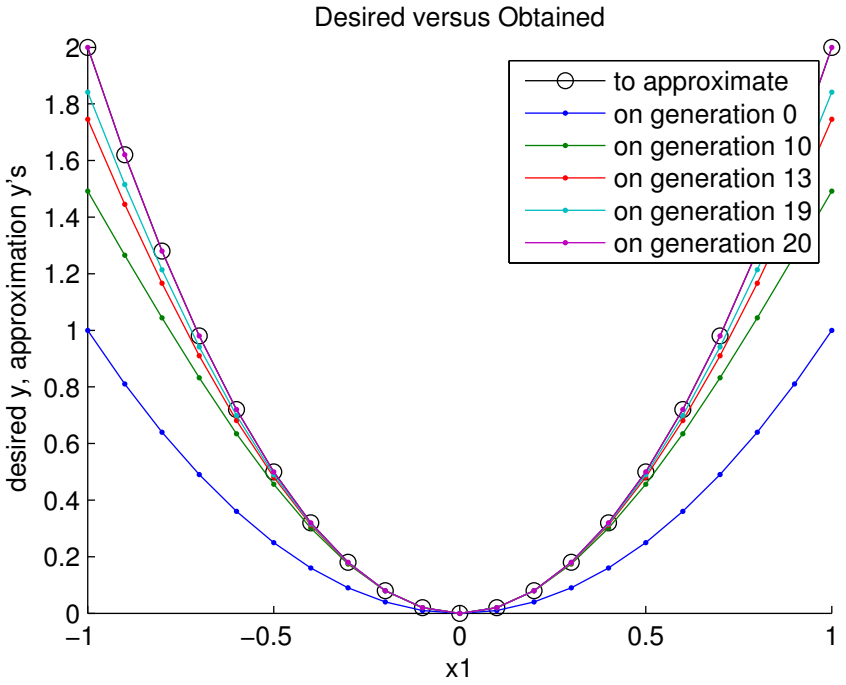
ตารางที่ 5.1: คู่อินพุต/เอาต์พุตของฟังก์ชันที่ต้องการค้นหาด้วย GP (สำหรับใช้ฝึกสอน)

	อินพุต (x)	เอาต์พุต (y)
ค่าความเหมาะสมกรณี 1	-1.0000	2.0000
ค่าความเหมาะสมกรณี 2	-0.9000	1.6200
ค่าความเหมาะสมกรณี 3	-0.8000	1.2800
ค่าความเหมาะสมกรณี 4	-0.7000	0.9800
ค่าความเหมาะสมกรณี 5	-0.6000	0.7200
ค่าความเหมาะสมกรณี 6	-0.5000	0.5000
ค่าความเหมาะสมกรณี 7	-0.4000	0.3200
ค่าความเหมาะสมกรณี 8	-0.3000	0.1800
ค่าความเหมาะสมกรณี 9	-0.2000	0.0800
ค่าความเหมาะสมกรณี 10	-0.1000	0.0200
ค่าความเหมาะสมกรณี 11	0.0000	0.0000
ค่าความเหมาะสมกรณี 12	0.1000	0.0200
ค่าความเหมาะสมกรณี 13	0.2000	0.0800
ค่าความเหมาะสมกรณี 14	0.3000	0.1800
ค่าความเหมาะสมกรณี 15	0.4000	0.3200
ค่าความเหมาะสมกรณี 16	0.5000	0.5000
ค่าความเหมาะสมกรณี 17	0.6000	0.7200
ค่าความเหมาะสมกรณี 18	0.7000	0.9800
ค่าความเหมาะสมกรณี 19	0.8000	1.2800
ค่าความเหมาะสมกรณี 20	0.9000	1.6200
ค่าความเหมาะสมกรณี 21	1.0000	2.0000

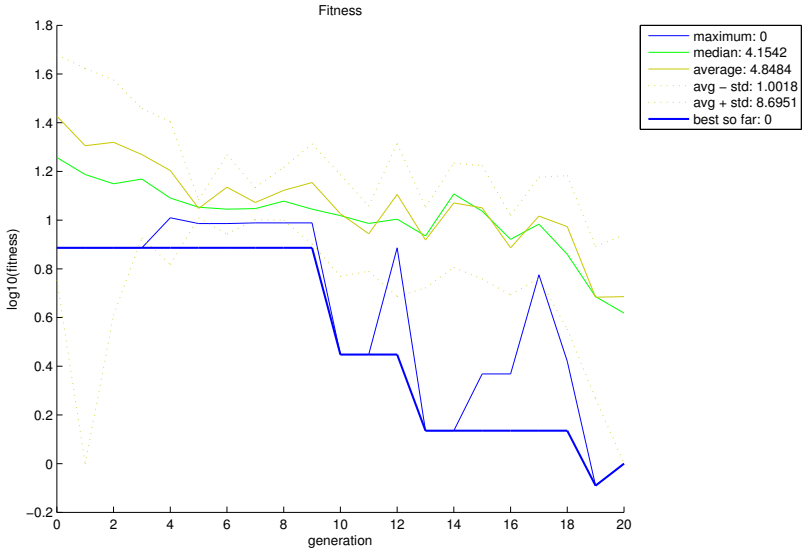


รูปที่ 5.7: โครงสร้างต้นไม้ของฟังก์ชันคำตอบที่ถูกต้องจากโครโมโซมรุ่นที่ 20

ศึกษาและกำหนดพารามิเตอร์ของ GP ให้เหมาะสมกับปัญหาที่ต้องการค้นหาคำตอบ ด้วยเทคโนโลยีของซอฟต์แวร์ในปัจจุบัน ทำให้เราสามารถศึกษา GP ได้อย่างสะดวกสบาย



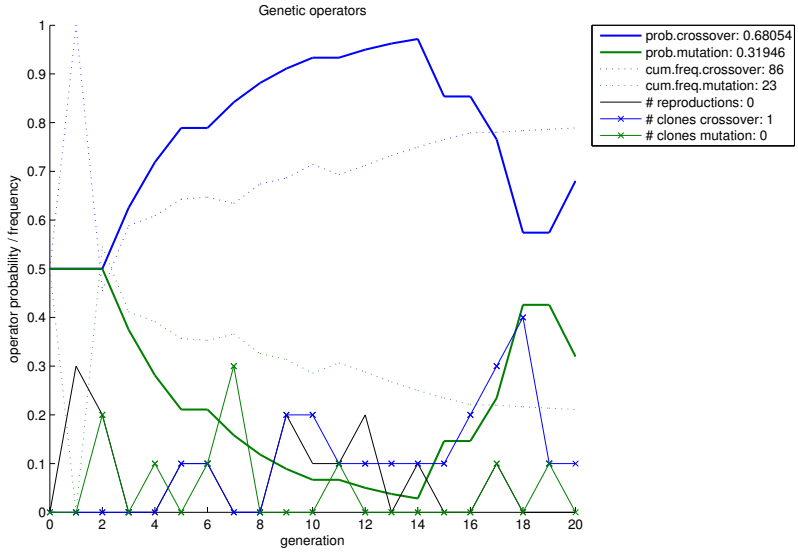
รูปที่ 5.8: กราฟจากฟังก์ชันของโครโมโซมที่ดีที่สุดในรุ่นประชากรต่างๆ



รูปที่ 5.9: ค่าความเหมาะสมจากโครโมโซมที่ดีที่สุดในรุ่นประชากรต่างๆ

5.5 สรุป

เนื้อหาของ GP ที่นำเสนอในหนังสือเล่มนี้เป็นเพียงระดับพื้นฐาน โดยมีจุดประสงค์เพื่อที่จะให้ผู้อ่านได้คุ้นเคยกับ GP เท่านั้น รายละเอียดเชิงลึกของ GP เองนั้นยังมีอีกมาก ผู้อ่านสามารถศึกษาเพิ่มเติมได้จากเอกสารอ้างอิงที่ได้ระบุไว้ ตัวอย่างของ GP ในรูปแบบต่างๆ เช่น [Banzhaf et al., 1998]



รูปที่ 5.10: ค่าความน่าจะเป็นและความถี่ของปฏิบัติการทางสายพันธุ์

- GP แบบจีโนมต้นไม้ (tree genomes) [Koza, 1992][Iba et al., 1995]
- GP แบบจีโนมเชิงเส้น (linear genomes) [Banzhaf, 1993][Cramer, 1985]
- GP แบบจีโนมกราฟ (graph genomes) [Jacob, 1996][Teller and Veloso, 1995]
- STROGANOFF (STructured Representation On Genetic Algorithms for NOnlinear Function Fitting) [Ivakhnenko, 1971]
- GP ใช้ไวยากรณ์ไม่พึ่งบริบท (context-free grammars) [Whigham, 1995a][Whigham, 1995b]
- GP ของระบบ L (L-systems)[Koza, 1993][Jacob, 1994][Memmi et al., 1994]

นอกไปจากนั้นแล้ว ยังมีงานอีกมากมายที่ได้ทำการปรับแต่ง GP ให้มีประสิทธิภาพในด้านหลักๆ ต่อไปนี้

- ความเร็วของ GP
- การวิวัฒนาการของโปรแกรม
- ประสิทธิภาพในการค้นหาของ GP

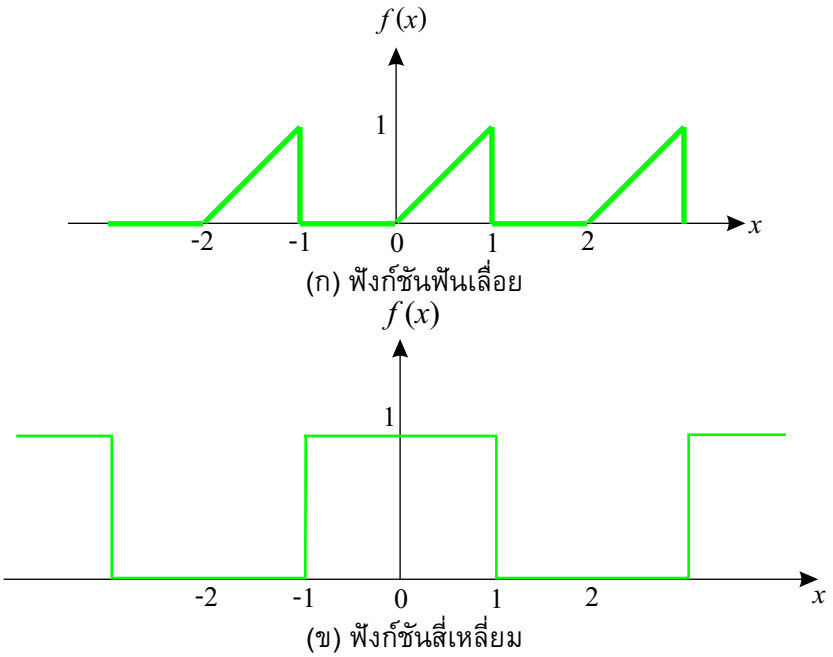
การโปรแกรมจีนเนติกหรือ GP เป็นการคำนวณเชิงวิวัฒนาการ ที่มีรูปแบบและโครงสร้างที่แตกต่างไปจาก GA หรือ ES โดย GP มีจุดประสงค์ที่จะค้นหาคำตอบในรูปของฟังก์ชันหรือรหัสต้นฉบับของโปรแกรม แล้วนำเอาฟังก์ชันหรือรหัสต้นฉบับของโปรแกรมนั้นๆ ไปใช้แก้ปัญหาที่ต้องการได้ เราอาจกล่าวได้ว่า GP เป็นโปรแกรมที่เขียนโปรแกรมเองได้ นอกเหนือไปจากการค้นหาฟังก์ชันการถดถอยแล้ว GP ยังสามารถนำไปประยุกต์ใช้งานด้านอื่นๆ อีกมากมายเช่นการจัดกลุ่ม (clustering) ชีววิทยา คอมพิวเตอร์กราฟิค ระบบควบคุม วิศวกรรมไฟฟ้า การเงิน การประมวลผลภาพ แบบจำลองระบบ การหาค่าเหมาะที่สุด การจดจำรูปแบบ การประมวลผลสัญญาณดิจิตอล เป็นต้น งานวิจัยด้านต่างๆ มากมาย ที่ได้รับการพิสูจน์ถึงประสิทธิภาพการทำงานของ GP จนกระทั่งทุกวันนี้ GP จึงได้รับความสนใจและมีการขยายขอบเขตการใช้งานเพิ่มขึ้นอย่างต่อเนื่อง

โจทย์คำถาม

5.1. จงอธิบายองค์ประกอบสำคัญในโครงสร้างของ GP

5.2. จงอธิบายหลักการที่นำเอา GP ไปใช้ในการทำให้โปรแกรมควบคุมหุ่นยนต์เคลื่อนที่ มีวิวัฒนาการในการเคลื่อนที่จากจุดหนึ่ง ไปยังอีกจุดหนึ่ง โดยมีสิ่งกีดขวางระหว่างทางเดิน

5.3. จงออกแบบ GP ในการใช้หาอนุกรมฟูริเยร์ของฟังก์ชัน $f(x)$ ที่แสดงในรูปที่ 5.11 โดยทำการกำหนดจำนวนเทอมของอนุกรม พร้อมทั้งแสดงเซตฟังก์ชันและเซตปลายทางทั้งหมด เลือกค่าการชักตัวอย่างให้เหมาะสม



(ก) ฟังก์ชันฟันเลื่อย

(ข) ฟังก์ชันสี่เหลี่ยม

รูปที่ 5.11: ฟังก์ชันสำหรับใช้หาอนุกรมฟูริเยร์

5.4. พิจารณาข้อมูลคู่อินพุต/เอาต์พุตสำหรับใช้หาฟังก์ชันการถดถอยในตารางที่ 5.2 โดยใช้ซอฟต์แวร์ GPLAB จงออกแบบ GP สำหรับค้นหาฟังก์ชันของข้อมูลดังกล่าว เลือกพารามิเตอร์ต่างๆ ของ GP พร้อมทั้งอธิบายผลการค้นหาที่ได้ (*หมายเหตุ*: ข้อมูลเป็นของฟังก์ชัน $f(x) = \exp x/2$

5.5. จากการออกแบบ GP ในคำถามที่ 1.4 จงอภิปรายและเปรียบเทียบผลลัพธ์จากการใช้พารามิเตอร์ที่แตกต่างกัน ดังต่อไปนี้ (ศึกษาเพิ่มเติมจากคู่มือการใช้ซอฟต์แวร์ GPLAB)

- จำนวนประชากร
- การกำหนดค่าเริ่มต้นของโครงสร้างต้นไม้ (โครโมโซมประชากร)
- ค่าความลึกที่มากที่สุดของโครงสร้างต้นไม้
- การแทนที่
- เซตฟังก์ชัน
- จำนวนค่าชักตัวอย่างของฟังก์ชัน (ในคำถามที่ 1.4 มีค่าชักตัวอย่าง 21 ค่า)

ตารางที่ 5.2: คู่อินพุต/เอาต์พุตของฟังก์ชันที่ต้องการค้นหาด้วย GP (สำหรับใช้ฝึกสอน)

	อินพุต (x)	เอาต์พุต (y)
ค่าความเหมาะสมกรณี 1	-1.0000	0.1839
ค่าความเหมาะสมกรณี 2	-0.9000	0.2033
ค่าความเหมาะสมกรณี 3	-0.8000	0.2247
ค่าความเหมาะสมกรณี 4	-0.7000	0.2483
ค่าความเหมาะสมกรณี 5	-0.6000	0.2744
ค่าความเหมาะสมกรณี 6	-0.5000	0.3033
ค่าความเหมาะสมกรณี 7	-0.4000	0.3352
ค่าความเหมาะสมกรณี 8	-0.3000	0.3704
ค่าความเหมาะสมกรณี 9	-0.2000	0.4094
ค่าความเหมาะสมกรณี 10	-0.1000	0.4524
ค่าความเหมาะสมกรณี 11	0.0000	0.5000
ค่าความเหมาะสมกรณี 12	0.1000	0.5526
ค่าความเหมาะสมกรณี 13	0.2000	0.6107
ค่าความเหมาะสมกรณี 14	0.3000	0.6749
ค่าความเหมาะสมกรณี 15	0.4000	0.7459
ค่าความเหมาะสมกรณี 16	0.5000	0.8244
ค่าความเหมาะสมกรณี 17	0.6000	0.9111
ค่าความเหมาะสมกรณี 18	0.7000	1.0069
ค่าความเหมาะสมกรณี 19	0.8000	1.1128
ค่าความเหมาะสมกรณี 20	0.9000	1.2298
ค่าความเหมาะสมกรณี 21	1.0000	1.3591

บรรณานุกรม

W. Banzhaf. Genetic programming for pedestrians. In S. Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, page 628, San Francisco, CA, 1993. University of Illinois at Urbana-Champaign, Morgan Kaufmann.

Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming - An Introduction: on the automatic evolution of computer programs and its applications*. Morgan Kaufmann, 1998.

N. L. Cramer. A representation for the adaptive generation of simple sequential programs. In J. J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and the Applications*, pages 183--187, Pittsburgh, PA, 1985. Carnegie-Mellon University.

Andries P. Engelbrecht. *Computational Intelligence: An Introduction*. Wiley, 2002. ISBN 0-470-84870-7.

H. Iba, T. Sato, and H. de Garis. Numerical genetic programming for system identification. In J.P. Rosca, editor, *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pages 64--75, Tahoe City, CA, 1995.

A. Ivakhnenko. Polynomial theory of complex systems. In *IEEE Transactions on Systems, Man and Cybernetics*, volume 1, pages 364--378, 1971.

C. Jacob. Genetic l-system programming. In Y. Davidor, H.-P. Schwefel, and M. R. Schwefel, editors, *Parallel Problem Solving from Nature III, Lecture Notes in Computer Science*, volume 866, 1994.

C. Jacob. Evolving evolution programs: Genetic programming and l-systems. In H. Voigt, W. Ebeling, I Rechenberg, and H. Schwefel, editors, *Parallel Problem Solving From Nature IV. Proceedings of the International Conference on Evolutionary Computation*, pages 42--51, Berlin, 1996. Springer-Verlag.

J. R. Koza. Discovery of rewrite rules in lindenmayer systems and state transition rules in cellular automata via genetic programming. In *Symposium on Pattern Formation (SPF-93)*, Claremont, CA, 1993.

J. R. Koza. *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT Press, Cambridge, MA, 1992.

H. Memmi, J. Mizoguchi, and K. Shimohara. Hardware evolution -- an hdl approach. In *Proceedings of the Japan-US Symposium on Flexible Automation*. The Institute of Systems, Control and Information Engineers, 1994.

Sara Silva. Gplab: A genetic programming toolbox for matlab. URL `http://gplab.sourceforge.net/`.

A. Teller and M. Veloso. Pado: Learning tree structured algorithms for orchestration into an object recognition system. Technical Report CMU-CS-95-101, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1995.

P. A. Whigham. Grammatically-based genetic programming. In J. P. Rosca, editor, *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pages 33--41, Tahoe City, CA, 1995a.

P. A. Whigham. Inductive bias and genetic programming. In A. M. S. Zalzala, editor, *First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA*, volume 414, pages 461--466, Sheffield, UK, 1995b. IEE, London, UK.