บทที่ 5
เจเนติกอัลกอริทึมสำหรับปัญหาการจัดสมดุลสายงานการประกอบแบบวัตถุประสงค์เดียว

เจเนติกอัลกอริทึมถูกนำไปประยุกต์ใช้ในการแก้ปัญหา Optimization Problem ต่างๆ เช่นการจัดตารางการผลิต (Murata และคณะ, 1996) การวางผังโรงงาน (ชนะ, 1998) รวมทั้งการจัดสมดุลของสายงานการประกอบซึ่งมีรูปแบบปัญหาต่างกันออกไป (Kim และคณะ,1996) โดยส่วนมากแล้ว เจเนติกอัลกอริทึมที่ใช้จะมีโครงสร้างหลักคล้ายคลึงกันตามแบบของเจเนติกอัลกอริทึมอย่างง่าย คือ มีการเข้ารหัสและสร้างประชากรเริ่มต้น (Representation) การรีโปรดักชั่น (Reproduction) การครอสโอเวอร์ และการมิวเตชั่น แต่อย่างไรก็ตาม เจเนติกอัลกอริทึมที่ใช้สำหรับแต่ละปัญหาก็จะมีรายละเอียดปลีกย่อยในโครงสร้างหลักที่แตกต่างกันออกไปขึ้นอยู่กับความเหมาะสมต่อรูปแบบปัญหานั้นๆ

เนื้อหาในบทนี้จะนำเสนอการประยุกต์เจเนติกอัลกอริทึมเพื่อใช้สำหรับปัญหาการจัดสมดุลของสายงานการประกอบอย่างง่าย โดยมีกฎเกณฑ์ในการจัด (Criteria) หรือในที่นี้เรียกว่าวัตถุประสงค์ (Objective) เพียงอย่างเดียว คือ เพื่อให้การกระจายของภาระงาน (Workload Variance) เท่ากัน

5.1 ลักษณะของปัญหาการจัดสมดุลสายงานการประกอบอย่างง่าย

ปัญหาการจัดสมดุลของสายงานการประกอบของงานวิจัยนี้เป็นปัญหาประเภท Single Model Deterministic แบบ Simple Case ซึ่งมีลักษณะดังนี้

1. เป็นปัญหาการกำหนดขั้นงาน (Work Element) ให้กับสถานีทำงานต่างๆ โดยมีวัตถุประสงค์ในการจัดเพื่อให้มีภาระงานในแต่ละสถานีเท่าๆกัน หรืออาจกล่าวได้อีกอย่างหนึ่งว่า เพื่อให้มีความแปรปรวนของภาระงานน้อยที่สุด ค่านี้สามารถหาได้จากสมการที่ 5.1

$$wv = \frac{\sum_{i=1}^{n}\left(Ti - \left(\frac{W}{n}\right)\right)^2}{n} \qquad (5.1)$$

เมื่อ wv = ความแปรปรวนของภาระงาน

n = จำนวนสถานีทำงานที่น้อยที่สุดเป็นจำนวนเต็ม

W = Total Processing Time

T_i = เวลาทำงานของสถานีที่ i

2. เป็นปัญหาการจัดสมดุลสายงานการประกอบของสายงานการประกอบสำหรับผลิตภัณฑ์ชนิดเดียว (Single Model Assembly Line)
3. เวลาทำงานคงที่ ไม่ขึ้นกับลำดับการจัดงาน และไม่ขึ้นกับสถานีที่ทำงานนั้นๆ
4. สายงานการประกอบมีลักษณะเป็นแบบ Machine Dominate ดังนั้นเวลาที่ใช้ในการทำงานแต่ละงานจะกำหนดเป็นเวลามาตรฐานตามเวลาทำงานของเครื่องจักร
5. ในการจัดงานให้กับสถานีต่างๆจะยึดหลักการจัดเพื่อให้มีจำนวนสถานีน้อยที่สุด ดังนั้นในการกำหนดงานให้กับสถานีจะพยายามกำหนดงานให้กับสถานีตามลำดับ กล่าวคือ จะกำหนดงานให้กับสถานีแรกจนเต็มความสามารถ (Capacity) ก่อนแล้วจึงค่อยกำหนดงานที่เหลือให้กับสถานีทำงานถัดไป
6. เป็นปัญหาการจัดสมดุลสายการประกอบให้กับสายงานการประกอบสายงานใหม่ ซึ่งยังไม่มีการติดตั้งอุปกรณ์ เครื่องจักรใดๆในสถานีการทำงาน ดังนั้นงานต่างๆจึงสามารถถูกจัดเข้าสถานีทำงานใดๆก็ได้ (ไม่มีข้อจำกัดของ Zoning Restriction)
7. สายงานการประกอบมีระบบการผลิตแบบ Flow Shop
8. สายงานการประกอบเป็นแบบอนุกรม คือแต่ละสถานีทำงานต่อเนื่องกันตามลำดับ ไม่มีสถานีที่ทำงานขนานกัน
9. ไม่มีการจำกัดขนาด Buffer Size ของแต่ละสถานีทำงานในสายงานการประกอบ
10. ทุกสถานีทำงานมีความสามารถในการทำงานเท่ากัน และความสามารถสูงสุดของแต่ละสถานีกำหนดจากรอบเวลาการผลิต
11. ข้อมูลเข้า(Input) ประกอบไปด้วย ขั้นการทำงาน (Work Element) เวลาทำงาน (Processing Time) ลำดับความสัมพันธ์ก่อนหลังของขั้นการทำงาน (Precedence Relationships) และรอบเวลาการผลิต (Cycle Time) ซึ่งทราบค่าแน่นอนอยู่แล้ว

นอกจากนี้ ในการจัดยังต้องพิจารณาถึงข้อจำกัดต่างๆ ได้แก่

- การกำหนดงานให้กับสถานีทำงานต้องไม่ขัดกับลำดับความสัมพันธ์ก่อน-หลังของงาน
- ในแต่ละสถานีทำงานสามารถทำงานได้หลายงาน แต่เวลารวมของการทำงานในสถานีทำงานนั้นๆต้องไม่เกินรอบเวลาการผลิตที่กำหนดไว้
- งานหนึ่งๆต้องทำบนสถานีใดสถานีหนึ่งเท่านั้น ไม่สามารถแยกงานไปทำคนละสถานีได้

- จำนวนสถานีทั้งหมดในสายงานการประกอบต้องไม่น้อยกว่าจำนวนสถานีทำงานที่น้อยที่สุดที่เป็นไปได้ และต้องไม่เกินจำนวนสถานีทำงานสูงสุดที่ยอมรับได้
- งานทุกงานต้องถูกจัดให้กับสถานีใดสถานีหนึ่งบนสายงานการประกอบ

12. Measure of Performance ที่ต้องคำนวณหานอกเหนือไปจากค่า Workload Variance ได้แก่ จำนวนสถานีที่ใช้ เวลาว่างงานรวม ประสิทธิภาพสายงานการประกอบ

5.2 โครงสร้างของเจเนติกอัลกอริทึมสำหรับปัญหาการจัดสมดุลของสายงานการประกอบแบบวัตถุประสงค์เดียว

5.2.1 โครงสร้างหลัก ประกอบด้วย 5 ส่วนคือ

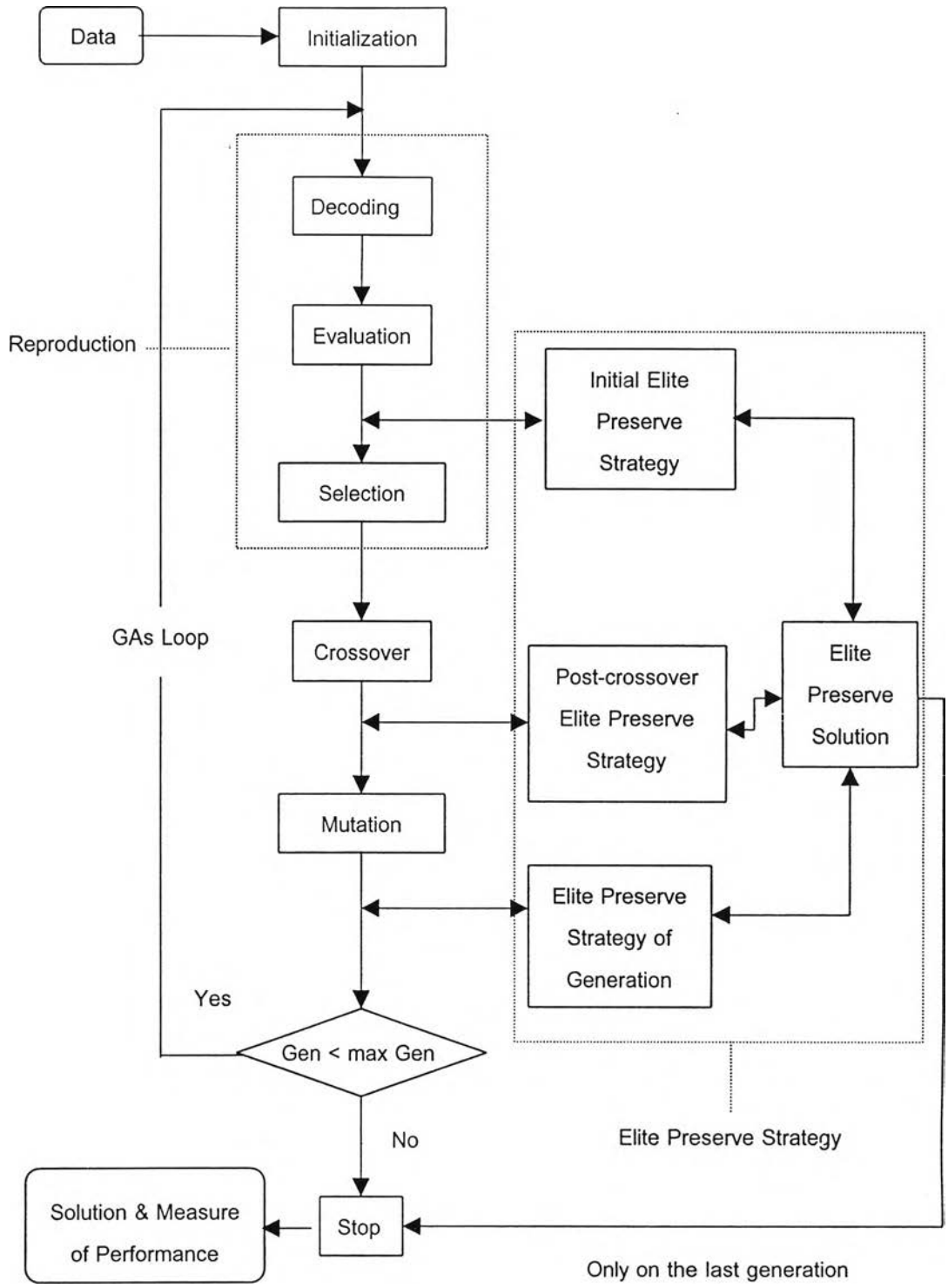
1. Initialization เป็นการใส่รหัสคำตอบและสร้างสตริงคำตอบเบื้องต้น
2. Reproduction เป็นการคัดเลือกสตริงคำตอบที่มีความเหมาะสม โดยดูจากค่าความเหมาะสม (Fitness) ของสตริงคำตอบแต่ละตัว ในส่วนของการ Reproduction จะแบ่งเป็นขั้นตอนย่อย 3 ขั้นตอน คือ
   2.1 Decoding เป็นการแปลความหมายของสตริงคำตอบ ในที่นี้การถอดรหัสคำตอบ
   2.2 Evaluation เป็นการคำนวณหาค่าต่างๆเพื่อนำไปสู่การหาค่าความเหมาะสม (Fitness) ของสตริงคำตอบ
   2.3 Selection เป็นกระบวนการในการคัดเลือกสตริงที่มีความเหมาะสมมากกว่า
3. Crossover เป็นการสร้างสตริงคำตอบตัวใหม่จากการแลกเปลี่ยนชิ้นส่วนระหว่างสตริงคำตอบ 2 ตัว
4. Mutation เป็นการสร้างสตริงคำตอบตัวใหม่โดยการย้ายค่าบางตำแหน่งภายในสตริงคำตอบ
5. Elite Preserve Strategy เป็นการเก็บค่าที่ดีที่สุดที่ได้จากการเปรียบเทียบระหว่างสตริงคำตอบที่สร้างใหม่ กับสตริงคำตอบตัวที่ดีที่สุดตัวเดิม

5.2.2 ขั้นตอนการทำงานของเจเนติกอัลกอริทึม

1. **Data Input :** รับข้อมูลเข้าต่างๆ ซึ่งได้แก่ งาน เวลาแต่ละงาน ความสัมพันธ์ตามลำดับก่อนหลังของงาน รอบเวลาการผลิต จำนวนสถานีสูงสุดที่ยอมรับได้

2. **Representation & Initialization :** นำข้อมูลต่างๆมาสร้างคำตอบเบื้องต้นแบบสุ่มจำนวน *popsize* ตัว โดยผ่านกระบวนการใส่รหัสคำตอบ (Representation) และการสร้างประชากรเบื้องต้น (Initial Population)
3. **Decoding :** นำรหัสคำตอบของประชากรเบื้องต้นทุกตัว มาถอดรหัสคำตอบเพื่อให้ได้คำตอบที่สมบูรณ์และสามารถนำไปคำนวณหาค่าต่างๆที่ต้องการได้
4. **Evaluation :** คำนวณหาค่าต่างๆที่ต้องการ เช่น เวลาในแต่ละสถานี จำนวนสถานีที่ต้องการ ค่าความแปรปรวนของภาระงาน ค่าประสิทธิภาพสายการประกอบ แล้วนำค่าเหล่านี้ไปคำนวณหาค่า Fitness ของประชากรเบื้องต้นทุกตัว
5. **Initial Elite Preserve Strategy :** หาค่าคำตอบที่ดีที่สุดจากประชากรเจนเนอเรชั่นแรกและเก็บค่าคำตอบที่ดีที่สุดนี้ไว้เป็น Elite Preserve Solution
6. **Selection :** คัดเลือกคำตอบที่ดีเข้าสู่ Mating Pool เพื่อเตรียมทำการจับคู่ โดยอาศัยวิธี Selection หาคำตอบที่มีความเหมาะสมมากกว่า
7. **Crossover :** ทำการจับคู่คำตอบที่อยู่ใน Mating Pool และทำการครอสโอเวอร์ด้วยความน่าจะเป็นในการครอสโอเวอร์ เท่ากับ Pc
8. **Post-crossover Elite Preserve Strategy :** หาค่าคำตอบที่ดีที่สุดจากการครอสโอเวอร์ แล้วนำมาเปรียบเทียบกับ Elite Preserve Solution ที่มีอยู่ ถ้าคำตอบที่ได้จากการ ครอสโอเวอร์ดีกว่า ก็ให้เก็บคำตอบนั้นเป็น Elite Preserve Solution แทน
9. **Mutation :** ทำการมิวเตชั่นสตริงคำตอบที่มีด้วยความน่าจะเป็นในการมิวเตชั่นเท่ากับ Pm
10. **Elite Preserve Strategy of Generation :** หาค่าคำตอบที่ดีที่สุดจากการมิวเตชั่นแล้วนำมาเปรียบเทียบกับ Elite Preserve Solution ที่มีอยู่ ถ้าคำตอบที่ได้จากมิวเตชั่นดีกว่า ก็ให้เก็บคำตอบนั้นเป็น Elite Preserve Solution แทน แต่ถ้า Elite Preserve Solution ดีกว่า ก็ให้แทนที่คำตอบที่แย่ที่สุดจากการมิวเตชั่นด้วย Elite Preserve Solution
11. **GAs-loop :** ดูว่า เจนเนอเรชั่น น้อยกว่าจำนวนเจนเนอเรชั่นสูงสุดหรือไม่ ถ้าน้อยกว่าให้กลับไปทำข้อที่ 7-13 ถ้าไม่ให้ทำข้อที่ 15
12. **Stop :** หยุดกระบวนการของเจเนติกอัลกอริทึม และนำค่า Elite Preserve Solution มาเป็นคำตอบ

โครงสร้างและวิธีการของเจเนติกอัลกอริทึมแสดงได้ดังรูปที่ 5.1



รูปที่ 5.1 แผนผังแสดงโครงสร้างและวิธีการของเจเนติกอัลกอริทึม

5.3 วิธีการของเจเนติกอัลกอริทึม

5.3.1 การใส่รหัสคำตอบ (Chromosome Representation / Coding)

ขั้นตอนแรกของเจเนติกอัลกอริทึม คือการกำหนดรูปแบบของการใส่รหัสคำตอบ ซึ่งถือว่าเป็นขั้นตอนที่สำคัญและมีผลอย่างมากต่อขั้นตอนอื่นๆของ GAs การใส่รหัสคำตอบ คือ การเปลี่ยนคำตอบของปัญหาให้อยู่ในรูปของสตริงคำตอบ (หรือที่เรียกว่า Chromosome) วิธีการใส่รหัสคำตอบมีทั้งแบบ Binary String และ Non-binary String ในกรณีของปัญหาการจัดสมดุลของสายงานการประกอบ คำตอบของปัญหาคือกลุ่มของงานที่ถูกมอบหมายให้กับสถานีทำงานสถานีต่างๆ ดังนั้น วิธีการใส่รหัสคำตอบที่ใช้จึงต้องสามารถแสดงลำดับของงานในรูปของสตริงได้ วิธีการใส่รหัสคำตอบที่ใช้จึงควรเป็นแบบ Non-binary String

ในปัญหาการจัดสมดุลของสายงานการประกอบอย่างง่าย งานต่างๆจะตกอยู่ภายใต้ข้อจำกัดของความสัมพันธ์ตามลำดับก่อนหลังของงาน จึงไม่สามารถกำหนดงานให้กับสถานีทำงานใดๆได้อย่างอิสระ ดังนั้นงานต่างๆควรจะถูกนำมาจัดลำดับ (Sequence) โดยพิจารณาความสัมพันธ์ก่อนหลังของงานเสียก่อนแล้วจึงค่อยนำไปเรียงใส่ให้กับสถานีทำงานต่างๆตามลำดับ จากแนวคิดดังกล่าว ลำดับของงานที่ถูกจัดเรียงสามารถนำมาใช้เป็นรูปแบบของการใส่รหัสคำตอบได้ วิธีนี้เรียกว่า Sequence-oriented Representation สตริงคำตอบที่ได้คือลำดับของงานทั้งหมดที่จะนำไปจัดให้กับสถานีตามลำดับนั่นเอง ลักษณะของสตริงคำตอบมีดังนี้

1) คำตอบ 1 คำตอบ แทนด้วยสตริงคำตอบ 1 ตัวที่เรียกว่า Chromosome
2) ใน 1 chromosome จะแบ่งเป็นหน่วยเล็กๆที่เรียกว่า bit เรียงกันอยู่ จำนวนของ bit จะเท่ากับจำนวนขั้นงานทั้งหมดที่ต้องทำบนสายงานการประกอบที่พิจารณา
3) ในแต่ละ bit จะมีค่าตัวเลขตั้งแต่ 1 ถึง m บรรจุอยู่ค่าหนึ่ง ค่านี้หมายถึงหมายเลขที่ใช้แทนขั้นงานต่างๆ
4) ตำแหน่งของ bit หมายถึงลำดับที่ของงานนั้นๆ
5) ตัวเลขในแต่ละ bit ต้องไม่ซ้ำกัน
6) ขั้นงานที่อยู่ใน bit ตำแหน่งก่อนหน้า จะต้องถูกจัดให้กับสถานีก่อนหน้าหรือสถานีเดียวกันกับ ขั้นงานที่อยู่ใน bit ตำแหน่งหลังเท่านั้น ไม่สามารถนำมาจัดให้กับสถานีหลังได้

ตัวอย่างเช่น สตริงคำตอบ [1 3 2 5 7 9 8 4 14 13 11 6 10 12 15] จะได้ว่า ใน 1 Chromosome มี 15 bit หมายถึง สายงานการประกอบที่พิจารณามี 15 ขั้นงาน งานแรกที่จะนำไปจัดให้กับสถานีคืองานในตำแหน่งแรก ซึ่งคือขั้นงานที่ 1 ขั้นงานถัดไปที่จะนำไปจัดให้กับสถานีคือ งานในตำแหน่งที่ 2 ซึ่งคือขั้นงานที่ 3 งานที่จะนำไปจัดอีกคืองานที่อยู่ในตำแหน่งถัดไปตามลำดับ

5.3.2 การสร้างกลุ่มประชากรเบื้องต้น (Initial Population Creating)

การสร้างกลุ่มประชากรเบื้องต้น คือ การสร้างคำตอบเบื้องต้นขึ้นมาจำนวนหนึ่งเพื่อนำไปใช้ในกระบวนการของ GAs โดยคำตอบ 1 คำตอบคือประชากร 1 ตัว จำนวนของประชากรที่ต้องการสร้างนั้นเป็นพารามิเตอร์ตัวหนึ่งที่ต้องมีการกำหนด ซึ่งในที่นี้กำหนดให้เท่ากับ เท่ากับ *popsize* ตัว

สำหรับปัญหาการจัดสมดุลของสายงานการประกอบนั้น ประชากร 1 ตัว หมายถึงลำดับของงานทั้งหมดที่จะทำบนสายงานการประกอบนั้น ดังนั้นการสร้างประชากร 1 ตัวจึงทำได้โดยการใส่ตัวเลขตั้งแต่ 1 ถึง m (m คือจำนวนงานทั้งหมด) ลงไปในแต่ละ bit ของสตริงคำตอบจนครบทุก bit และทำเช่นนี้ไปเรื่อยๆจนกว่าจะได้ประชากรทั้งหมด *popsize* ตัว

ตามหลักการของ GAs การสร้างประชากรเบื้องต้นมักใช้วิธีการแบบสุ่ม ซึ่งหมายความว่าตัวเลขที่นำมาใส่ในแต่ละ bit จะต้องถูกเลือกแบบสุ่ม แต่อย่างไรก็ตาม ในกรณีของปัญหาการจัดสมดุลของสายงานการประกอบซึ่งมีข้อจำกัดด้านความสัมพันธ์ตามลำดับก่อนหลังของงาน การสร้างประชากรแบบสุ่มอาจทำให้เกิดคำตอบที่ขัดกับข้อจำกัดดังกล่าว หรือที่เรียกว่า คำตอบที่เป็นไปไม่ได้ (Infeasible Solution) ดังนั้นในการสร้างคำตอบเบื้องต้นสำหรับปัญหา ALB นี้จึงต้องใช้วิธีสุ่มโดยพิจารณาความสัมพันธ์ตามลำดับก่อนหลังของงานร่วมด้วย วิธีนี้จะช่วยรับประกันได้ว่าคำตอบเบื้องต้นที่สร้างขึ้นมาทั้งหมดจะเป็นคำตอบที่ไม่ขัดกับข้อจำกัดดังกล่าว หรือที่เรียกว่า คำตอบที่เป็นไปได้ (Feasible Solution)

5.3.2.1 การสร้างเมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงาน (Precedence Metrix)

การสุ่มโดยพิจารณาความสัมพันธ์ตามลำดับก่อนหลังของงานร่วมด้วย จำเป็นต้องอาศัยเครื่องมือช่วยที่เรียกว่า เมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงาน (Precedence Metrix) ซึ่งแสดงตัวอย่างไว้ในรูปที่ 5.2 เมตริกซ์นี้จะช่วยบอกให้เรารู้ว่า

งานใดต้องทำก่อนหรือหลังงานใด อีกทั้งยังช่วยบอกว่างานนั้นๆมีงานก่อนหน้า หรืองานที่ต้องทำตามหลังอีกกี่งาน

ก่อน \ หลัง	1	2	3	4	5	6
1	0	1	0	1	0	1
2	0	0	0	1	0	1
3	0	1	0	1	0	1
4	0	0	0	0	0	1
5	0	0	0	1	0	1
6	0	0	0	0	0	0

(หมายเหตุ: จำนวนขั้นงานทั้งหมด คือ 6

รูปที่ 5.2 ตัวอย่างเมตริกซ์แสดงความสัมพันธ์ตามลำดับก่อนหลังของงาน

ลักษณะของตารางจะเป็นเมตริกซ์ ขนาด m x m ซึ่ง m หมายถึงจำนวนขั้นงานทั้งหมด หมายเลขของแถว (Row) หมายถึงงานที่ทำก่อน และหมายเลขของหลัก (Column) หมายถึงงานที่ต้องทำทีหลัง ตำแหน่งแถวและคอลัมน์ที่ 1ถึง m จะประกอบไปด้วยตัวเลข 0 และ 1 ซึ่ง 0 จะหมายถึง งานที่ไม่มีความสัมพันธ์ก่อนหลังระหว่างกัน ส่วน 1 จะหมายถึงงานที่มีความสัมพันธ์ก่อนหลังระหว่างกัน เช่น
ที่แถวที่ 3 คอลัมน์ที่ 2 มีค่าเป็น 1 หมายความว่า งานที่ 3 ต้องทำก่อนงานที่ 2
ทีแถวที่ 3 คอลัมน์ที่ 5 มีค่าเป็น 0 หมายความว่างานที่ 3 และงานที่ 5 ไม่มีความสัมพันธ์ก่อนหลังกัน

นอกจากนี้ ถ้าต้องการดูว่างานใดตามหลังงานที่เราสนใจ ก็ให้ดูที่แถวของงานนั้น และถ้าต้องการดูว่ามีงานใดต้องทำก่อนงานที่เราสนใจ ก็ให้ดูที่คอลัมน์ของงานนั้น ตัวอย่างเช่น ถ้าต้องการรู้ว่ามีงานใดตามหลังงานที่ 3 ก็ดูที่แถวที่ 3 จะได้ว่างานตามหลัง คือ 2 4 และ 6 หรือถ้าต้องการรู้ว่ามีงานใดต้องทำก่อนหน้างานที่ 3 ก็ดูที่คอลัมน์ที่ 3 จะได้ว่าไม่มีงานก่อนหน้างานที่ 3

วิธีการสร้างตารางความสัมพันธ์ตามลำดับก่อนหลังของงาน มีดังนี้

1. สร้างเมตริกซ์ศูนย์ขนาด m x m
2. เอาความสัมพันธ์ตามลำดับก่อนหลังของงานซึ่งเป็นข้อมูลรับเข้ามาแปลงเป็นค่า 0 หรือ 1 ที่ตำแหน่งต่างๆของเมตริกซ์ เช่น ถ้ากำหนดให้ขั้นงานที่ 4 มีงานก่อนหน้าคืองาน 2,3 ก็จะได้ว่า ในคอลัมน์ที่ 4 ให้ใส่เลข 1 ที่ แถวที่ 2 และ 3
3. จากข้อ 1 และ 2 จะได้ตารางความสัมพันธ์ ซึ่งค่า 0-1 จะเป็นตัวบ่งชี้ถึงความสัมพันธ์ระหว่างงาน จากนั้นให้ทำการรวมค่าในแต่ละคอลัมน์ เพื่อดูว่างานต่างๆมีงานที่ต้องทำก่อนหน้าอีกกี่งาน

4. เมื่อได้ตารางแสดงความสัมพันธ์ตามลำดับก่อนหลังของงานแล้ว ก็สามารถสรุปได้ว่าแต่ละงานมีงานที่ต้องทำก่อนหน้ากี่งาน โดยดูจากผลรวมในแต่ละคอลัมน์ เช่นผลรวมในแต่ละคอลัมน์เป็น 0 2 0 2 0 1 หมายความว่า งานที่ 1 3 5 ไม่มีงานก่อนหน้า งานที่ 2 4 มีงานก่อนหน้า 2 งาน และงานที่ 6 มีงานก่อนหน้า 1 งาน

5.3.2.2 การสร้างคำตอบเบื้องต้นสำหรับปัญหา ALB โดยใช้เมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงาน

การสร้างคำตอบเบื้องต้น จะใช้วิธีกำหนดหมายเลขงานลงไปในแต่ละ bit ของสตริงคำตอบตามลำดับจนครบทุก bit โดยตัวเลขที่จะนำมาใส่ ต้องได้รับการตรวจสอบโดยใช้ Precedence Metrix แล้วว่าเป็นงานที่ไม่มีงานที่ต้องทำก่อนหน้าเหลืออยู่แล้ว ขั้นตอนการสร้างโดยละเอียด มีดังนี้

1. หางานที่ไม่มีงานก่อนหน้า (Task without Predecessor) โดยดูจากผลรวมของคอลัมน์ใน Precedence Metrix คอลัมน์หมายเลขใดที่ผลรวมเป็น 0 งานหมายเลขนั้นก็จะไม่มีงานก่อนหน้า
2. ถ้างานที่ไม่มีงานก่อนหน้ามีเพียงงานเดียว ให้เอางานนั้นใส่ลงไปในตำแหน่ง bit แรกของสตริงคำตอบได้เลย แต่ถ้ามีหลายงานก็ให้เลือกมาเพียง 1 งานโดยการสุ่ม แล้วจึงเอาไปใส่ใน bit แรก
3. งานหมายเลขใดที่ถูกกำหนดลงไปแล้ว ให้ตัดทิ้งโดยการเปลี่ยนตัวเลขแถวนั้นใน Precedence Metrix เป็น 0 ทั้งหมด
4. หาผลรวมในแต่ละคอลัมน์ใหม่อีกครั้ง
5. ทำซ้ำข้อที่ 1 ถึง 4 เพื่อกำหนดงานลงไปใน bit ถัดๆไป จนกระทั่งงานทุกงานถูกกำหนดลงไปในสตริงคำตอบจนหมด

5.3.2.3 จำนวนประชากรเบื้องต้น

จากขั้นตอนการสร้างประชากรเบื้องต้นในข้อ 5.3.2.2 เป็นการสร้างประชากรเพียง 1 ตัวเท่านั้น แต่ในวิธีการของ GAs จำเป็นที่จะต้องมีจำนวนประชากรมากกว่า 1 ตัวเพื่อให้สามารถดำเนินการตามวิธีการของ GAs ในขั้นต่อๆไปได้ จำนวนประชากรเบื้องต้นจะเท่ากับจำนวนประชากรในแต่ละเจนเนอเรชั่น และเป็นพารามิเตอร์ตัวหนึ่งที่มีผลต่อประสิทธิภาพของ GAs การกำหนดจำนวนประชากรเบื้องต้นที่เหมาะสมจะได้กล่าวในบทต่อไป แต่ในที่นี้ให้ใช้จำนวนประชากรเท่ากับ *popsize* ตัว

การสร้างประชากรเบื้องต้นให้ได้ครบ *popsize* ตัว สามารถทำได้โดยทำตามขั้นตอนที่ 1 ถึง 5 ในข้อ 5.3.2.2 จนครบ *popsize* ครั้ง ในแต่ละครั้งที่เริ่มสร้างประชากรตัวใหม่ Precedence Metrix จะต้องถูกเปลี่ยนให้กลับสู่ Precedence Metrix ดั้งเดิมเหมือนในข้อ 5.3.2.1 เสียก่อน

ประชากรเบื้องต้นทั้งหมดที่สร้างขึ้นจำนวน *popsize* ตัวต้องไม่ซ้ำกัน ทั้งนี้เพื่อให้เกิดความหลากหลายของคำตอบ และเป็นการป้องกันไม่ให้คำตอบที่ได้จากวิธีของเจเนติกอัลกอริทึมที่ใช้เป็นค่า Local Optima นอกจากนี้การสร้างประชากรเบื้องต้นให้แตกต่างกันยังช่วยให้สามารถกำหนดจำนวนประชากรน้อยลงได้

5.3.3 การถอดรหัสคำตอบ (Decoding)

คำตอบที่ปรากฏอยู่ในประชากรหรือสตริงคำตอบที่สร้างขึ้นยังเป็นคำตอบที่ไม่สมบูรณ์ กล่าวคือเป็นเพียงลำดับของงานที่จะต้องนำไปจัดให้กับสถานีตามลำดับเท่านั้น ดังนั้นจึงต้องมีการนำงานตามลำดับที่ได้ในสตริงคำตอบไปจัดให้กับสถานีทำงานให้เรียบร้อยเสียก่อน ซึ่งเราจะเรียกขั้นตอนนี้ว่าการถอดรหัสคำตอบ (Decoding) แต่อย่างไรก็ตาม สตริงคำตอบที่เรามีสามารถบอกได้แต่เพียงว่างานที่อยู่ในลำดับแรกๆ ควรจะถูกจัดลงไปในสถานีทำงานต้นๆ และงานที่อยู่ในลำดับหลังก็ควรจะถูกจัดให้อยู่ในสถานีเดียวกันหรือสถานีหลังเท่านั้น ดังนั้นจึงสามารถนำงานในสตริงคำตอบมาจัดได้หลายแบบ เพื่อให้ได้การจัดที่ดีที่สุด สตริงคำตอบที่ได้จะต้องถูกถอดรหัสด้วยวิธีที่เหมาะสม

สำหรับปัญหา ALB ที่พิจารณา การถอดรหัสคำตอบทำได้ดังนี้

1. นำงานที่อยู่ในลำดับแรกในสตริงคำตอบไปจัดให้กับสถานีทำงานแรก
2. นำงานที่อยู่ในลำดับถัดไปจัดให้กับสถานีทำงานแรกเช่นกัน แล้วดูว่าเวลาทำงานรวมในสถานีเกินรอบเวลาการผลิตที่กำหนดให้หรือไม่ ถ้าเกินให้ตัดงานล่าสุดที่จัดให้สถานีทิ้งไป แล้วนำงานที่ตัดออกไปจัดให้กับสถานีถัดไป แต่ถ้าเวลารวมในสถานีน้อยกว่ารอบเวลาผลิต ก็ให้เอางานที่อยู่ในลำดับต่อๆมาไปจัดให้กับสถานีนั้นจนกว่าเวลารวมของสถานีจะมากกว่ารอบเวลาการผลิต
3. เมื่อนำงานที่ตัดออกมาจากสถานีเดิม มาจัดให้กับสถานีถัดไปแล้ว ก็ให้นำงานลำดับต่อๆมาไปจัดให้กับสถานีทำงานนั้น จนกว่าเวลาทำงานรวมของสถานีจะเกินรอบเวลาการผลิต ถ้าเกินก็ให้ตัดงานนั้นออกจากสถานีเดิม และนำไปจัดให้กับสถานีใหม่ต่อไป
4. ทำตามข้อที่ 3 จนกว่างานทุกงานจะถูกจัดให้กับสถานีทำงานจนหมด

จากการถอดรหัสคำตอบ สตริงที่มีลักษณะการจัดเรียงลำดับของงานที่ต่างกัน เมื่อนำมาผ่านกระบวนการถอดรหัสแล้วอาจให้คำตอบหรือผลการจัดงานที่เหมือนกันก็ได้ ดังตารางที่ 5.1

ตารางที่ 5.1 สตริงคำตอบตัวอย่าง 2 ตัวที่มีการจัดเรียงลำดับงานต่างกันแต่ให้ผลการจัดเหมือนกัน

สตริงคำตอบที่	1	2
สตริงคำตอบ เวลาทำงานของแต่ละงาน รอบเวลาการผลิต	1-3-5-4-2-6-7-8-10-9 2 4 3 4 3 3 2 2 2 1 10	1-5-3-2-4-6-7-8-10-9 2 3 5 3 4 3 2 2 2 1 10
ผลการจัด	สถานีที่ 1 = 1 3 5 สถานีที่ 2 = 4 2 6 สถานีที่ 3 = 7 8 10 9	สถานีที่ 1 = 1 5 3 สถานีที่ 2 = 2 4 6 สถานีที่ 3 = 7 8 10 9

จะเห็นได้ว่าสตริงตัวอย่าง 2 ตัวมีลักษณะการจัดเรียงที่แตกต่างกัน แต่เมื่อถอดรหัสคำตอบแล้วได้ผลเหมือนกัน ดังที่ได้กล่าวมาแล้วข้างต้นว่า ในการสร้างประชากรคำตอบเบื้องต้นจะไม่ยอมให้เกิดคำตอบที่ซ้ำกันขึ้นมา ดังนั้นภายหลังจากการถอดรหัสคำตอบแล้ว จึงต้องมีการตรวจสอบดูว่าผลการถอดรหัสที่ได้เหมือนกันหรือไม่ ถ้าเหมือนกันก็ให้สร้างประชากรเบื้องต้นตัวใหม่ที่ไม่ซ้ำกับตัวเดิมขึ้นมาแทน

ในกรณีที่เป็นการถอดรหัสคำตอบของสตริงคำตอบที่ได้จากการครอสโอเวอร์หรือการมิวเตชั่น ไม่จำเป็นต้องพิจารณาผลการถอดรหัสว่าให้คำตอบที่เหมือนกันหรือไม่ ทั้งนี้เนื่องจากในขั้นตอนของการครอสโอเวอร์ และมิวเตชั่นเรายอมให้เกิดคำตอบที่ซ้ำกันขึ้นมาได้

การถอดรหัส ไม่เพียงแต่ให้คำตอบว่าแต่ละสถานีงานทำงานอะไรบ้าง แต่ยังให้ค่าของ จำนวนสถานีทำงานที่ต้องการ (จำนวนสถานีทำงานที่ได้จะเป็นจำนวนสถานีทำงานที่น้อยที่สุดอยู่แล้ว) และเวลาทำงานรวมในแต่ละสถานีด้วย

5.3.4 การประเมินค่า (Evaluation)

ก่อนที่จะเข้าสู่ขั้นตอนการคัดเลือกของเจเนติกอัลกอริทึม จำเป็นที่จะต้องมีการประเมินค่าประชากรแต่ละตัวเสียก่อนว่ามีความเหมาะสมมากหรือน้อยเพียงใด ความเหมาะสมนี้จะวัดจากค่า Fitness ของสตริงคำตอบแต่ละตัว ตัวใดที่มีค่า Fitness มากก็หมายความว่ามีความเหมาะสมมากตามไปด้วย โดยที่ค่า Fitness ดังกล่าวหมายถึง ค่าของฟังก์ชั่นวัตถุประสงค์ที่เราต้องการทำให้ต่ำที่สุดหรือสูงที่สุดนั่นเอง

ค่าฟังก์ชั่นวัดถุประสงค์ในที่นี้คือ ค่าความแปรปรวนของภาระงาน (Workload Variance) ที่ต้องการทำให้มีค่าต่ำที่สุดนั่นเอง การหาค่าความแปรปรวนของภาระงานของสตริงแต่ละตัวสามารถหาได้จากสมการที่ 5.1

สมการดังกล่าวอยู่ในรูปแบบของการหาค่าต่ำที่สุด ดังนั้นต้องทำการแปลงให้อยู่ในรูปแบบการหาค่าสูงที่สุดของ Fitness Function เสียก่อน ดังสมการที่ 5.2

$$f(Xi) = \left(\sum_{i=1}^{popsize} Xi \right) - Xi \qquad (5.2)$$

เมื่อ Xi คือค่า Workload Variance ของสตริงคำตอบแต่ละตัว

นอกเหนือไปจากการคำนวณหาค่า Workload Variance และค่า Fitness ของสตริงคำตอบแต่ละตัวแล้ว ในขั้นตอนของการประเมินค่ายังสามารถหาค่าอื่นๆที่เกี่ยวข้อง นอกเหนือไปจากค่าที่ได้จากการถอดรหัส ได้ คือ

ก. รอบเวลาการผลิตจริง จากการถอดรหัสอาจได้ว่าสถานีทำงานบางสถานีใช้เวลาทำงานรวมน้อยกว่ารอบเวลาการผลิต ดังนั้นจึงสามารถกำหนดรอบเวลาการผลิตจริงได้จากเวลาทำงานรวมของแต่ละสถานีที่มากที่สุด ดังสมการที่ 5.3

$$real_ct = \max(Ti) \qquad i = 1,2,\ldots,N \qquad (5.3)$$

ข. เวลาว่างงานรวมจริง คือผลรวมของเวลาว่างงานในแต่ละสถานี หรือผลรวมของส่วนต่างระหว่างรอบเวลาการผลิตจริงกับเวลาทำงานในแต่ละสถานี สามารถเขียนเป็นสมการได้ดังสมการที่ 5.4

$$Total\ Idle\ Time = \sum_{i=1}^{N} (real_ct - Ti) \qquad (5.4)$$

ค. ประสิทธิภาพของสายงานการประกอบ เป็นตัววัดว่าสายงานการประกอบที่ออกแบบมีประสิทธิภาพมากน้อยแค่ไหน หาได้จากสมการ 5.5

$$Line\ Eff. = \frac{\sum_{i=1}^{n} \left(\frac{Ti \times 100}{real_CT} \right)}{n} \qquad (5.5)$$

หมายเหตุ ค่าเวลาว่างงานรวม ประสิทธิภาพของสายการผลิต รวมทั้งจำนวนสถานีที่ต้องการ สามารถนำมาใช้เป็นค่าวัตถุประสงค์ที่ต้องการทำให้ต่ำที่สุดหรือสูงที่สุดได้เช่นกัน และสามารถนำไปแทนเป็นตัวแปร Xi ใน Fitness Function เพื่อเปลี่ยนให้เป็นปัญหาการจัดสมดุลของสายงานการประกอบที่มีวัตถุประสงค์ของการจัดงานแบบอื่นๆ ได้ แต่ในกรณีของค่าประสิทธิภาพการผลิตซึ่งเป็นค่าฟังก์ชั่นของการหาค่าสูงสุดอยู่แล้ว ให้ใช้สมการการหาค่าประสิทธิภาพของสายงานป็น Fitness Function ได้เลย

5.3.5 การคัดเลือกคำตอบ (Selection)

การคัดเลือกคำตอบทำโดยนำเอากลุ่มสตริงคำตอบเบื้องต้นทั้งหมดมาผ่านวิธีการคัดเลือกโดยดูจากค่า Fitness ของสตริงคำตอบแต่ละตัวเป็นหลัก สตริงตำตอบตัวที่มีค่า Fitness มากก็มีโอกาสที่จะถูกคัดเลือกไว้มากกว่าตัวที่มีค่า Fitness น้อย สตริงคำตอบที่ผ่านการคัดเลือกจำนวน *popsize* ตัวจะผ่านเข้าสู่ Mating Pool เพื่อรอการจับคู่และการดำเนินการของ GAs ในขั้นต่อไป

การคัดเลือกคำตอบที่ใช้ คือ วิธี Tournament Selection (Goldberg,1991) ซึ่งเป็นวิธีที่ดัดแปลงมาจากวิธี Roulette Wheel Selection ดังนั้นจึงต้องมีการสร้างวงล้อรูเล็ดขึ้นมาก่อน

5.3.5.1 การสร้างวงล้อรูเล็ต

วงล้อรูเล็ต คือวงกลมที่มีพื้นที่ขนาด 1 หน่วยซึ่งพื้นที่ถูกแบ่งออกเป็นส่วนๆ ตามจำนวนของประชากรในแต่ละเจนเนอเรชั่น (เท่ากับ *popsize* ส่วน) พื้นที่แต่ละส่วนจะมีขนาดเท่ากับความน่าจะเป็นในการถูกเลือกของสตริงคำตอบแต่ละตัว วิธีการสร้างมีดังนี้

1. หาค่า Fitness รวมของสตริงคำตอบทั้งหมด *popsize* ตัว ดังสมการที่ 5.6

$$F = \sum_{i=1}^{popsize} f(Xi) \qquad (5.6)$$

โดย $f(Xi)$ = ค่า Fitness ของสตริงตัวที่ i

2. หาค่าความน่าจะเป็นในการถูกคัดเลือก (Probability of Selection) ของสตริงคำตอบแต่ละตัว ตามสมการที่ 5.7

$$p_i = \frac{f(Xi)}{F} \qquad i = 1, 2, \ldots, popsize \qquad (5.7)$$

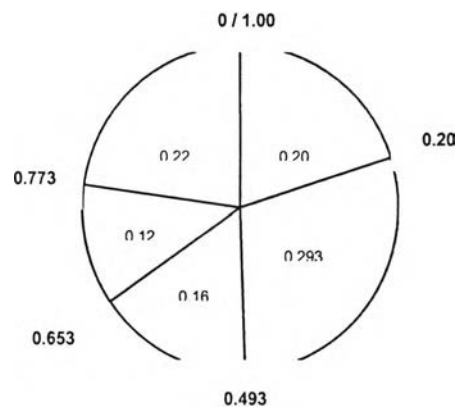
3. หาค่าความน่าจะเป็นในการถูกคัดเลือกสะสม (Cumulative Probability of Selection) ของสตริงคำตอบแต่ละตัว ตามสมการที่ 5.8

$$q_i = \sum_{j=1}^{i} p_j \qquad (5.8)$$

ตัวอย่างของวงล้อรูเล็ตแสดงได้ดังตารางที่ 5.2 และรูปที่ 5.3

ตารางที่ 5.2 *ตัวอย่างตารางแสดงการสร้างวงล้อรูเล็ต*

String No.	Fitness	*pi*	*qi*
1	15.000	0.200	0.200
2	22.000	0.293	0.493
3	12.000	0.160	0.653
4	9.000	0.120	0.773
5	17.000	0.227	1.000
รวม	75.000	1.000	



รูปที่ 5.3 *วงล้อรูเล็ต*

5.3.5.2 วิธี Tournament Selection

การคัดเลือกสตริงคำตอบโดยวิธี Roulette Wheel Selection ซึ่งใช้กันอยู่ทั่วไป จะใช้สุ่มสตริงคำตอบจากวงล้อรูเล็ต ซึ่งมีโอกาสที่จะสุ่มได้สตริงคำตอบที่มีค่า Fitness น้อยๆด้วย แต่สำหรับการคัดเลือกสตริงคำตอบโดยวิธี Tournament Selection เป็นการ

สุ่มสตริงคำตอบจากวงล้อรูเล็ตมา 2 ตัว แล้วนำค่า Fitness มาเปรียบเทียบกันอีกครั้งหนึ่ง สตริงคำตอบที่ถูกเลือกจึงเป็นตัวที่มีความเหมาะสมมากกว่า สำหรับขั้นตอนการเลือกมีดังนี้

1. สร้างตัวเลขสุ่ม r ซึ่งมีค่าระหว่าง 0 ถึง 1 ขึ้นมา 1 ค่า คือ r_1
2. ถ้า $r_1 < q_1$ ให้เลือกสตริงคำตอบตัวแรก แต่ถ้า $q_{i-1} < r_1 < q_i$ (เมื่อ $2< i <$ *popsize*) ให้เลือกสตริงคำตอบตัวที่ i มาเป็นสตริงคำตอบตัวแรก
3. สร้างตัวเลขสุ่ม r ซึ่งมีค่าระหว่าง 0 ถึง 1 ขึ้นมาอีก 1 ค่า คือ r_2
4. ถ้า $r_2 < q_1$ ให้เลือกสตริงคำตอบตัวแรก แต่ถ้า $q_{i-1} < r2 < q_i$ (เมื่อ $2< i <$ *popsize*) ให้เลือกสตริงคำตอบตัวที่ i มาเป็นสตริงคำตอบตัวที่สอง
5. นำค่า Fitness ของสตริงคำตอบทั้ง 2 ตัวมาเปรียบเทียบกัน ตัวใดมีค่า Fitness มากกว่าก็ให้เลือกตัวนั้นเข้าสู่ Mating Pool
6. ทำตามขั้นตอนข้อที่ 1 – 5 จนกว่าจะได้สตริงคำตอบใน Mating Pool ครบ *popsize* ตัว

จากวิธีดังกล่าวจะเห็นได้ว่า สตริงคำตอบที่มีค่า Fitness มากก็จะมีพื้นที่มากจึงมีโอกาสที่ตัวเลขสุ่มที่สร้างขึ้นมาจะตกอยู่ภายในบริเวณของสตริงคำตอบตัวนั้นมากกว่าตัวที่มีค่า Fitnessน้อย (มีพื้นที่น้อย) ทำให้สตริงคำตอบที่ถูกเลือกเข้าสู่ Mating Pool เป็นสตริงคำตอบที่มีค่า Fitness โดยเฉลี่ยสูงกว่าสตริงคำตอบเดิม

ตารางที่ 5.3 *ตัวอย่างการคัดเลือกด้วยวิธี Tournament Selection*

ครั้งที่	ประชากรตัวที่ 1				ประชากรตัวที่ 2				หมายเลขประชากรที่เลือก
	r_1	$qi > r_1$	หมายเลขประชากร	ค่า Fitness	r_2	$qi > r_2$	หมายเลขประชากร	ค่า Fitness	
1	0.320	0.493	2	22.000	0.951	1.000	5	17.000	2
2	0.178	0.200	1	15.000	0.607	0.653	3	12.000	1
3	0.891	1.000	5	17.000	0.762	0.773	4	9.000	5
4	0.457	0.493	2	22.000	0.018	0.200	1	15.000	2
5	0.936	1.000	5	17.000	0.406	0.493	2	22.000	2

(หมายเหตุ: qi ที่มากกว่าค่า r ในคอลัมน์ที่ 3 และ7 ได้มาจากตารางที่ 5.2)

ตารางที่ 5.3 แสดงตัวอย่างการคัดเลือกด้วยวิธี Tournament Selection ซึ่งจะเห็นได้ว่าสตริงคำตอบหมายเลข 2 ซึ่งมีค่า Fitness มากที่สุด จะถูกสุ่มเลือกขึ้นมามากครั้งที่สุด ในขณะที่สตริงคำตอบซึ่งมีค่า Fitness น้อยก็จะถูกสุ่มเลือกน้อยครั้งเช่นกัน ข้อสังเกตประการหนึ่งจากตัวอย่างก็คือ ในการสุ่ม ก็สุ่มได้สตริงคำตอบหมายเลข 4 ที่มี

ค่า Fitness น้อยที่สุดด้วย ซึ่งถ้าใช้วิธี Roulette Wheel สตริงคำตอบหมายเลข 4 นี้ก็จะมีโอกาสที่จะถูกเลือกเข้าสู่ Mating Pool และจะได้รับการดำเนินการตามกระบวนการ GAs ต่อไป แม้ว่าสตริงคำตอบตัวนี้จะมีความเหมาะสมต่ำก็ตาม แต่เมื่อใช้วิธีคัดเลือกแบบ Tournament Selection สตริงคำตอบหมายเลข 4 นี้จะต้องถูกนำไปเปรียบเทียบกับสตริงคำตอบอีกตัวก่อน ดังนั้นโอกาสที่สตริงคำตอบตัวนี้จะถูกเลือกเข้าสู่ Mating Pool ก็จะลดลง

5.3.6 การครอสโอเวอร์ (Crossover)

5.3.6.1 การจับคู่สตริงคำตอบ

จากสตริงคำตอบจำนวน *popsize* ตัวที่ได้มาจากกระบวนการคัดเลือก จะมีสตริงคำตอบเพียงบางส่วนเท่านั้นที่จะถูกนำมาจับคู่เพื่อเตรียมสำหรับกระบวนการครอสโอเวอร์ สตริงคำตอบที่ไม่ได้ถูกนำไปจับคู่ก็จะยังคงสภาพเดิมและอยู่ใน Mating Pool (เป็นประชากรในเจนเนอเรชั่น) ต่อไป จำนวนสตริงคำตอบที่จะถูกนำมาจับคู่ (*Nc*) ขึ้นอยู่กับความน่าจะเป็นในการครอสโอเวอร์ (*Pc*) การจับคู่สตริงคำตอบเพื่อที่จะนำไป ครอสโอเวอร์ มีขั้นตอนดังนี้

1. สร้างตัวเลขสุ่ม *r* ซึ่งมีค่าระหว่าง 0 ถึง 1 ให้กับสตริงคำตอบแต่ละตัว
2. สตริงคำตอบตัวใดที่ตัวเลขสุ่มมีค่าน้อยกว่า *Pc* จะถูกเลือกไปจับคู่และทำการครอสโอเวอร์
3. ถ้าไม่มีสตริงคำตอบตัวใดที่มีค่า *r* น้อยกว่า *Pc* ให้เริ่มทำข้อ 1 และ 2 อีกครั้ง
4. ถ้ามีสตริงคำตอบที่มีค่า *r* น้อยกว่า *Pc* ทั้งหมดจำนวน *Nc* ตัว โดยที่ *Nc* เป็นจำนวนคี่ ต้องทำการปรับให้เป็นจำนวนคู่ก่อน โดยมีเงื่อนไขในการปรับ ดังนี้

   - ถ้า *Nc* เป็นจำนวนคี่ซึ่งมีค่าระหว่าง 1 ถึง *popsize* ให้ทำการสุ่มตัวเลข 0 หรือ 1 ขึ้นมา 1 ค่า ถ้าสุ่มได้เลข 1 ให้เพิ่มสตริงคำตอบเข้าไปอีก 1 ตัว โดยสุ่มเลือกจากตัวที่เหลืออยู่ใน Mating Pool แต่ถ้าสุ่มได้เลข 0 ให้ตัดสตริงคำตอบทิ้ง 1 ตัว โดยสุ่มเลือกจากตัวที่ได้เลือกเอาไว้
   - ถ้า *Nc* มีค่าเท่ากับ 1 การปรับให้ใช้วิธีเพิ่มสตริงเข้าไปอีก 1 ตัวเท่านั้น
   - ถ้า *Nc* มีจำนวนเท่ากับ *popsize* ซึ่งเป็นจำนวนคี่ การปรับให้ใช้วิธีตัดสตริงคำตอบที่เตรียมได้ลง 1 ตัวเท่านั้น
5. เมื่อได้สตริงคำตอบที่จะนำมาจับคู่ทั้งหมด *Nc* ตัว ให้นำมาจับคู่ตามลำดับ ซึ่งจะได้ทั้งหมด *Nc*/2 คู่

5.3.6.2 การครอสโอเวอร์

สตริงคำตอบที่เตรียมไว้ $Nc/2$ คู่จะถูกนำมาผ่านกระบวนการครอสโอเวอร์ ซึ่งเป็นกระบวนการที่นำสตริงคำตอบที่ถูกจับคู่ไว้มาแลกเปลี่ยนส่วนซึ่งกันและกันเพื่อให้เกิดสตริงใหม่ขึ้นโดยจะเรียกสตริงคำตอบ 2 ตัวที่ถูกจับคู่นี้ว่า *"สตริงคำตอบรุ่นพ่อแม่ (Parent)"* และจะเรียกสตริงคำตอบ 2 ตัวที่ได้จากการครอสโอเวอร์นี้ว่า *"สตริงคำตอบรุ่นลูก (Offspring)"* วิธีการครอสโอเวอร์มีหลายวิธี ในที่นี้เสนอวิธีการครอสโอเวอร์ไว้ทั้งหมด 6 วิธี คือ

1) วิธี Modified One-point Crossover

วิธีการครอสโอเวอร์แบบตำแหน่งเดียวทำได้โดยการหาจุดครอสโอเวอร์มา 1 ตำแหน่ง แล้วนำส่วนท้ายของสตริงตัวหนึ่งต่อเข้ากับส่วนหัวของสตริงอีกตัวหนึ่ง ผลการครอสโอเวอร์จะได้สตริงคำตอบรุ่นลูกจำนวน 2 ตัว แต่วิธีนี้อาจทำให้สตริงมีลักษณะที่ไม่ถูกต้อง กล่าวคืออาจทำให้ยีนบางตำแหน่งเกิดค่าซ้ำกันซึ่งอาจทำให้เกิดการจัดงานซ้ำในสถานีทำงานได้ ดังนั้นจึงได้ดัดแปลงวิธีการครอสโอเวอร์แบบตำแหน่งเดียวเพื่อป้องกันการเกิดค่าซ้ำ โดยการครอสโอเวอร์วิธีนี้จะเริ่มจากการกำหนดจุดครอสโอเวอร์ (Crossover Point : Xp) ขึ้นมา 1 ตำแหน่งโดยการสุ่มค่า ระหว่าง [1, m - 1] โดยที่ m คือความยาวของสตริง ตำแหน่งการครอสโอเวอร์แทนด้วย " | " และกำหนดให้ตำแหน่ง bit ที่ 1 ถึง Xp ของ Parent ตัวที่ 1 เป็น H1 ตำแหน่ง bit ที่ 1 ถึง Xp ของ Parent ตัวที่ 2 เป็น H2

$$p_1 = [\underbrace{9\ 8\ 4}_{H1} |\ 5\ 6\ 7\ 1\ 3\ 2\ 10\ ]$$

$$p_2 = [\underbrace{8\ 7\ 1}_{H2} |\ 2\ 3\ 10\ 9\ 5\ 4\ 6\ ]$$

นำ H1 มาเป็นส่วนหัวของ Offspring ตัวที่ 1 และนำ H2 มาเป็นส่วนหัวของ Offspring ตัวที่ 2 ตำแหน่งที่เหลือเป็นตำแหน่งว่างแทนด้วย #

$$o_1 = [\ 9\ 8\ 4\ |\ \#\ \#\ \#\ \#\ \#\ \#\ \#\ ]$$

$$o_2 = [\ 8\ 7\ 1\ |\ \#\ \#\ \#\ \#\ \#\ \#\ \#\ ]$$

ลบค่าใน p_1ที่ซ้ำกับค่าใน o_2 ออกโดยการแทน x ลงในตำแหน่งที่ซ้ำใน p_2 และลบค่าใน p_2ที่ซ้ำกับค่าใน o_1 ออกโดยการแทน x ลงในตำแหน่งที่ซ้ำใน p_1

$$o_2 = [\ 8\ 7\ 1\ |\ \#\ \#\ \#\ \#\ \#\ \#\ \#\ ]$$

$$p_1 = [9\ x\ 4\ 5\ 6\ x\ x\ 3\ 2\ 10]$$

$$o_1 = [\ 9\ 8\ 4\ |\ \#\ \#\ \#\ \#\ \#\ \#\ \#\ ]$$

$$p_2 = [\ x\ 7\ 1\ |\ 2\ 3\ 10\ x\ 5\ x\ 6\ ]$$

นำค่าที่เหลืออยู่ใน p_1 มาแทนที่ # ใน o_2 ตามลำดับเดิม และนำค่าที่เหลืออยู่ใน p_2 มาแทนที่ # ใน o_1 ตามลำดับเดิมเช่นกัน ก็จะได้สตริงคำตอบรุ่นลูกที่สมบูรณ์

o_1 = [9 8 4 7 1 2 3 10 5 6]

o_2 = [8 7 1 9 4 5 6 3 2 10]

เนื่องจากสตริงคำตอบรุ่นพ่อแม่เป็นคำตอบที่เป็นไปได้ (เป็นคำตอบที่ไม่ขัดกับหลักความสัมพันธ์ตามลำดับก่อนหลังของงาน) อยู่แล้ว ดังนั้นสตริงคำตอบรุ่นลูกที่ได้จากการครอสโอเวอร์วิธีนี้จะไม่ขัดกับหลักความสัมพันธ์ก่อนหลังของงานด้วยเช่นกัน

2) วิธี Partial Match Crossover (PMX)

วิธีการของ PMX อย่างแรกคือการเลือกคู่สตริงพ่อแม่ขึ้นมาอย่างสุ่ม จากนั้นทำการเลือกตำแหน่งที่จะทำการครอสโอเวอร์อย่างสุ่มที่อยู่ในช่วง [1, m - 1] โดยที่ m คือความยาวของสตริง และยีนตัวแรกคือยีนหมายเลข 1 และยีนตัวสุดท้ายคือยีนหมายเลข m ขอบเขตของการครอสโอเวอร์อยู่ในช่วงเครื่องหมาย " | "

p_1 = [9 8 4 | 5 6 7 | 1 3 2 0]

p_2 = [8 7 1 | 2 3 0 | 9 5 4 6]

ในขั้นตอนต่อไปจะทำการสลับค่าระหว่างสตริงที่อยู่ในช่วง " | " นั่นคือตำแหน่งสุ่มอยู่ในช่วง [4, 6] ของโครโมโซมลูกหลานทั้งสอง โดยที่ค่าที่อยู่นอกเครื่องหมาย " | " และเป็นค่าที่ซ้ำกันกับค่าที่อยู่ในช่วงที่กำหนดให้เป็น x

จะเห็นได้ว่าตำแหน่งที่สองของสตริง p_2 มีค่าเป็น 7 ซึ่งซ้ำกับค่าที่อยู่ในช่วง " | " ของสตริง p_1 ก่อนทำการสลับ จึงเปลี่ยนค่าที่ซ้ำให้เป็น x ก่อนและตำแหน่งอื่นๆที่มีค่าซ้ำกันก็จะเปลี่ยนเป็น # ด้วยเช่นกัน

o_1 = [9 8 4 | 2 3 0 | 1 # # #]

o_2 = [8 x 1 | 5 6 7 | 9 5 4 #]

แล้วทำการแทนค่า (Map) ดังต่อไปนี้ 2 เป็น 5 และ 5 เป็น 2; 3 เป็น 6 และ 6 เป็น 3; 0 เป็น 7 และ 7 เป็น 0 การแทนค่าเหล่านี้ได้มาจากค่าที่อยู่ในช่วง " | " โดยพิจารณาตำแหน่งของสตริงที่ตรงกัน

สตริงที่ได้เมื่อทำการแทนที่แล้วคือ

o_1 = [9 8 4 2 3 0 1 6 5 7]

o_2 = [8 0 1 5 6 7 9 2 4 3]

3) วิธี Cycle crossover (CX)

ขั้นตอนแรกเลือกคู่สตริงพ่อแม่ขึ้นมาอย่างสุ่ม พิจารณาดังต่อไปนี้

p_1 = [2 0 3 4 8 5 9 1 7 6]

p_2 = [5 9 2 1 0 6 7 3 8 4]

พิจารณาที่ตำแหน่งแรกของสตริง p_1 และ p_2 คงเดิมไว้

o_1 = [2 - - - - - - - - -]

o_2 = [5 - - - - - - - - -]

ค่าตำแหน่งแรกของสตริง o_2 เป็น 5 นำมาพิจารณาค่าที่เป็น 5 (ตำแหน่งที่ 6) ของสตริง o_1 จะคงเดิมไว้ ส่วนค่าที่เป็น 6 (ตำแหน่งที่ 6) ของสตริง o_2 จะคงเดิมไว้เช่นกัน จะได้เป็น

o_1 = [2 - - - - 5 - - - -]

o_2 = [5 - - - - 6 - - - -]

จากค่าที่เป็น 6 ของสตริง o_2 นำไปพิจารณาต่อในสตริง o_1 ซึ่งจะคงเดิมไว้ ได้สตริงเป็น

o_1 = [2 - - - - 5 - - - 6]

o_2 = [5 - - - - 6 - - - 4]

ด้วยกระบวนการเดิมจะได้สตริงเป็น

o_1 = [2 - - 4 - 5 - - - 6]

o_2 = [5 - - 1 - 6 - - - 4]

↓

o_1 = [2 - - 4 - 5 - 1 - 6]

o_2 = [5 - - 1 - 6 - 3 - 4]

↓

o_1 = [2 - 3 4 - 5 - 1 - 6]

o_2 = [5 - 2 1 - 6 - 3 - 4]

กระบวนการในการเลือกตัวที่จะคงเดิมไว้จะสิ้นสุดต่อเมื่อ ค่าในตำแหน่งใดตำแหน่งหนึ่งของ o_2 มีค่าตรงกับค่าในตำแหน่งแรกของ สตริง o_1 จากนั้นจะทำการสลับค่าที่ยังคงเหลืออยู่ทั้งสองสตริงแบบ ตำแหน่งต่อตำแหน่ง ได้สตริงลูกหลานเป็น

o_1 = [2 9 3 4 0 5 7 1 8 6]

o_2 = [5 0 2 1 8 6 9 3 7 4]

4) วิธี Order Crossover (OX)

เช่นเดียวกับวิธีการอื่นๆ อย่างแรกคือการเลือกคู่สตริงพ่อแม่ขึ้นมาอย่างสุ่ม จากนั้นทำการเลือกตำแหน่งที่จะทำการครอสโอเวอร์อย่างสุ่ม ขอบเขตของการครอสโอเวอร์อยู่ในช่วงเครื่องหมาย " | "

p_1 = [8 9 4 | 5 6 7 | 1 3 2 0]

p_2 = [8 7 1 | 2 3 0 | 9 5 4 6]

สุ่มตำแหน่งที่จะทำการครอสโอเวอร์ได้อยู่ในช่วง [4, 6] จากนั้นเลือกคู่แม็ปที่อยู่ในช่วงการครอสโอเวอร์ถ้าค่าที่ตรงกันให้ทำเครื่องหมาย # เครื่องหมาย # หมายความว่าปล่อยตำแหน่งนั้นให้ว่าง สตริงลูกหลานจะเป็น

o_1 = [8 9 4 | 5 6 7 | 1 # # #]

o_2 = [8 # 1 | 2 3 0 | 9 # 4 #]

จากนั้นจะทำการเลื่อนตำแหน่ง โดยยึดถือตำแหน่งหลังช่วง " | " เป็นจุดอ้างอิง (สำหรับสตริง o_1 เป็น 1 และ สำหรับ o_2 เป็น 9) ได้สตริงลูกหลานดังต่อไปนี้

o_1 = [5 6 7 | # # # | 1 8 9 4]

o_2 = [2 3 0 | # # # | 9 4 8 1]

ขั้นตอนต่อไปจะเป็นการสลับตำแหน่งที่อยู่ในช่วงของการครอสโอเวอร์ของสตริงทั้งสองที่เหลือ สตริงที่ได้เมื่อทำการครอสโอเวอร์แล้วคือ

o_1 = [5 6 7 | 2 3 0 | 1 8 9 4]

o_2 = [2 3 0 | 5 6 7 | 9 4 8 1]

5) วิธี Position-Base Method

พัฒนาโดย Syswerda โดยดัดแปลงมาจากวิธีการครอสโอเวอร์แบบ OX (Gen, M. and Chen, R., 1997) โดยเริ่มจากการเลือกสตริงพ่อแม่

p_1 = [2 0 3 4 8 5 9 1 7 6]

p_2 = [5 9 2 1 0 6 7 3 8 4]

เลือกตำแหน่งจากสตริงพ่อแม่ $p1$ อย่างสุ่ม แล้วนำค่าในตำแหน่งที่เลือกของสตริงพ่อแม่ p_1 ไปใส่ในตำแหน่งเดียวกันของสตริงลูก o_1 ในที่นี้เลือกตำแหน่งที่ 4 และ 7

p_1 = [2 0 3 4 8 5 9 1 7 6]

p_2 = [5 9 2 1 0 6 7 3 8 4]

o_1 = [# # # 4 # # 9 # # # #]

ตัดค่าที่อยู่ในตำแหน่งที่เลือกของสตริงพ่อแม่p_1 ออกจากสตริงพ่อแม่ p_2 ดังนั้นค่าที่ตัดออกคือ 4และ9

p_2 = [5 # 2 1 0 6 7 3 8 #]

นำค่าที่เหลืออยู่ในสตริงพ่อแม่p_2 มาใส่ในสตริงลูกo_1 ตามลำดับ

p_2 = [5 # 2 1 0 6 7 3 8 #]

O_1 = [5 2 1 4 0 6 9 7 3 8]

6) วิธี Order-Base Method

พัฒนาโดย Syswerda เช่นกัน มีความใกล้เคียงกับPosition-Base แต่แตกต่างกันเล็กน้อย โดยมีวิธีการดังนี้

เลือกตำแหน่งอย่างสุ่มจากสตริงพ่อแม่p_1 ในที่นี้เลือกตำแหน่งที่ 4 และ 7

p_1 = [2 0 3 4 8 5 9 1 7 6]

p_2 = [5 9 2 1 0 6 7 3 8 4]

พิจารณาว่า นำค่าจากสตริงพ่อแม่p_1 ไปใส่สตริงลูกo_1 ที่ตำแหน่งเดียวกันกับตำแหน่งของสตริงพ่อแม่ p_2 ที่มีค่าตรงกับค่าที่ถูกเลือกในสตริงพ่อแม่ p_1 โดยเรียงตามลำดับก่อนหลัง

p_1 = [2 0 3 4 8 5 9 1 7 6]

O_1 = [# 4 # # # # # # # 9]

p_2 = [5 9 2 1 0 6 7 3 8 4]

นำค่าในตำแหน่งที่เหลือของสตริงพ่อแม่ p_2 ไปใส่ในตำแหน่งที่ว่างอยู่ของสตริงลูก ตามลำดับ

O_1 = [5 4 2 1 0 6 7 3 8 9]

p_2 = [5 9 2 1 0 6 7 3 8 4]

ในขั้นสุดท้ายจะทำการคืนสตริงเข้าสู่เมทติ้งพูล

5.3.6.3 การซ่อมแซมคำตอบ (Repair Method)

จากการครอสโอเวอร์ทั้ง 6 วิธี มีเพียงวิธีแรกเท่านั้นที่สตริงคำตอบรุ่นลูกที่ได้เป็นคำตอบที่เป็นไปได้ ส่วนวิธีที่ 2-5 นั้น คำตอบที่ได้จะขัดกับหลักความสัมพันธ์ก่อนหลังของงาน ดังนั้นจึงต้องทำการซ่อมแซมคำตอบเพื่อให้ได้คำตอบที่เป็นไปได้เสียก่อน

การซ่อมแซมคำตอบที่เสนอโดย Kim และคณะ (1996) เป็นการซ่อมแซมคำตอบโดยพยายามรักษาลำดับของค่าในสตริงให้เหมือนเดิมมากที่สุด วิธีการทำได้ดังนี้ คือ

1) หาว่าขั้นงานใดที่ไม่มีงานก่อนหน้า จาก Precedence Metrix
2) นำขั้นงานที่ไม่มีงานก่อนหน้าที่ได้ทั้งหมดไปตรวจดูว่าอยู่ในตำแหน่งใดในสตริงคำตอบที่ต้องการซ่อมแซม

3) นำขั้นงานที่ไม่มีงานก่อนหน้า ซึ่งอยู่ในตำแหน่งที่ด้นที่สุด ไปใส่ในตำแหน่งแรกของสตริงคำตอบใหม่
4) ตัดงานที่ใส่ลงไปในสตริงคำตอบแล้วออกจาก Precedence Metrix
5) ทำตามขั้นตอนที่ 1-4 จนกว่าจะครบทุกขั้นงาน

เมื่อทำการครอสโอเวอร์และซ่อมแซมคำตอบเสร็จแล้ว สตริงคำตอบรุ่นลูกที่เป็นคำตอบที่เป็นไปได้จะถูกส่งกลับเข้าสู่ Mating Pool เพื่อไปรวมกับสตริงคำตอบรุ่นพ่อแม่ที่ไม่ได้ถูกเลือกมาครอสโอเวอร์ จากนั้นก็จะเข้าสู่กระบวนการของ GAs ลำดับถัดไป

5.3.7 การมิวเตชั่น (Mutation)

คือการสลับตำแหน่งของค่าภายในสตริงคำตอบตัวเดียว วิธีการมิวเตชั่นมีหลายวิธี แต่ในกรณีของปัญหาการจัดสมดุลของสายงานการประกอบ การสลับตำแหน่งของค่าในสตริงมีโอกาสทำให้ได้สตริงตัวใหม่ที่เป็นคำตอบที่เป็นไปไม่ได้สูงมาก ดังนั้นจึงใช้วิธีการมิวเตชั่นแบบ Random-Sequence Mutation ซึ่งสตริงที่ได้จากการมิวเตชั่นแบบนี้จะไม่ขัดกับหลักความสัมพันธ์ตามลำดับก่อนหลังของงาน

การพิจารณาว่าสตริงตัวใดจะถูกนำมามิวเตชั่นขึ้นอยู่กับค่าความน่าจะเป็นในการมิวเตชั่น (Pm) โดยการพิจารณาจะเริ่มจากการสุ่มค่า r ซึ่งมีค่าระหว่าง [0,1] ให้กับสตริงคำตอบทุกตัวใน Mating Pool จากนั้น ทำการเลือกเฉพาะสตริงที่มีค่า r น้อยกว่าค่าความน่าจะเป็นในการมิวเตชั่น (Pm) ไปทำการมิวเตชั่น

เมื่อได้สตริงตัวที่จะทำการมิวเตชั่นแล้ว ให้ทำการสุ่มตำแหน่งที่จะทำการมิวเตชั่น (Mp) ขึ้นมา 1 ตำแหน่ง ซึ่งเป็นค่าระหว่าง [1,m-1] โดยที่ m คือความยาวของสตริงคำตอบ ตำแหน่งที่จะทำการมิวเตชั่นแทนด้วย " | "

จากนั้นให้นำค่าในตำแหน่งที่ 1 ถึง Mp มาเป็นค่าในตำแหน่งที่ 1 ถึง Mp ของสตริงคำตอบตัวใหม่ ตำแหน่งอื่นๆซึ่งยังไม่มีค่า ให้แทนด้วย #

p = [9 8 5 3 4 2 1 6 10 7]

o = [9 8 5 3 # # # # # #]

ตำแหน่งที่เหลือในสตริงคำตอบตัวใหม่จะได้มาจากการสุ่มแบบมีเงื่อนไขเช่นเดียวกับการสร้างสตริงคำตอบเบื้องต้น โดยในตอนแรกต้องตัดงานที่กำหนดลงไปในสตริงตัวใหม่แล้ว (ในที่นี้ คือ งานที่ 9 8 5 3) ออกจาก Precedence Metrix เสียก่อนแล้วจึงทำตามวิธีการสร้างสตริงคำตอบเบื้องต้นต่อไป

สตริงคำตอบที่ได้จากการมิวเตชั่นและสตริงคำตอบที่ไม่ได้ถูกเลือกมาทำการมิวเตชั่นจะถูกนำมารวมกัน เพื่อเตรียมเข้าสู่เจนเนอเรชั่นต่อไป

5.3.8 เทคนิคการเก็บค่าที่ดีที่สุด (Elite Preserve Strategy)

เนื่องจากสตริงคำตอบที่ได้จากการครอสโอเวอร์และการมิวเตชั่น อาจเป็นคำตอบที่แย่กว่าคำตอบที่เคยปรากฏในเจนเนอเรชั่นที่ผ่านๆมา ดังนั้นจึงต้องมีการเก็บค่าที่ดีที่สุด (Elite Preserve Solution) เอาไว้เพื่อใช้เปรียบเทียบกับค่าที่ดีที่สุดของสตริงคำตอบชุดใหม่ ถ้าหาก Elite Preserve Solution ให้ค่า Fitness ที่ดีกว่าค่าที่ดีที่สุดของสตริงชุดใหม่ก็ให้เอา Elite Preserve Solution แทนที่ค่าที่แย่ที่สุด ทั้งนี้เพื่อให้สตริงคำตอบที่ดีที่สุดเท่าที่พบยังคงอยู่ในกระบวนการของ GAs ต่อไป

เทคนิคการเก็บค่าที่ดีที่สุดจะถูกนำไปใช้ระหว่างกระบวนการของ GAs 3 ครั้ง คือ ดังนั้นจึงแบ่งเทคนิคการเก็บค่าที่ดีที่สุดออกเป็น 3 เทคนิค คือ

1) Initial Elite Preserve Strategy

เป็นจุดเริ่มต้นของเทคนิคการเก็บค่าที่ดีที่สุด ซึ่งจะกระทำเพียงครั้งเดียวภายหลังจากการสร้างสตริงคำตอบเบื้องต้นในตอนต้นของกระบวนการ GAs และผ่านการถอดรหัสรวมทั้งการประเมินค่าเรียบร้อยแล้ว ค่า Fitness ของสตริงแต่ละตัวที่ได้จากการประเมินค่าจะถูกเรียงลำดับจากมากไปน้อย สตริงคำตอบเพียงหนึ่งตัวที่ให้ค่า Fitness มากที่สุดก็จะถูกเลือกไปเป็น คำตอบที่ดีที่สุดที่เก็บไว้ (Elite Preserve Solution) จากนั้นสตริงคำตอบทั้งหมดรวมทั้งตัวที่เลือกไปเป็น Elite Preserve Solution จะเข้าสู่ขั้นตอนต่างๆของ GAs ต่อไป

2) Post-crossover Elite Preserve Strategy

เป็นเทคนิคการเก็บค่าที่ดีที่สุดที่ใช้ภายหลังจากที่เสร็จสิ้นกระบวนการครอสโอเวอร์แล้ว ทั้งนี้เนื่องจากว่าสตริงคำตอบที่ได้จากการครอสโอเวอร์อาจเป็นคำตอบที่ดีกว่าคำตอบอื่นๆที่เคยพบมา แต่เมื่อนำไปมิวเตชั่นแล้ว สตริงคำตอบตัวนี้จะเปลี่ยนไปและอาจให้คำตอบที่ดีน้อยกว่าเดิม ดังนั้น เพื่อป้องกันไม่ให้สตริงคำตอบที่ดีที่ได้หลังจากการครอสโอเวอร์สูญหายไปจึงต้องทำการการถอดรหัสและประเมินค่าสตริงคำตอบภายหลังการครอสโอเวอร์ทั้งหมด *popsize* ตัว แล้วนำสตริงคำตอบที่ดีที่สุดภายหลังการครอสโอเวอร์ไปเปรียบเทียบกับ Elite Preserve Solution ถ้าหากสตริงคำตอบภายหลังการครอสโอเวอร์ดีกว่า ก็ให้เอาสตริงคำตอบที่ดีที่สุดนั้นไปเป็น Elite Preserve Solution แทน แต่ถ้า Elite Preserve Solution ดีกว่า ก็ให้นำสตริงคำตอบภายหลังจากการครอสโอเวอร์ทั้งหมด *popsize* ตัวไปผ่านกระบวนการมิวเตชั่นตามปกติ

ยกตัวอย่างเช่น ภายหลังการครอสโอเวอร์ มีสตริงคำตอบ 10 ตัว ซึ่งเมื่อนำไปผ่านกระบวนการถอดรหัสและประเมินค่าแล้วได้ค่า Fitness ของสตริงคำตอบ

แต่ละตัวเป็น 2 6 8 7 9 4 5 12 6 4 ค่า Fitness ที่ดีที่สุดใน 10 ตัวนี้คือค่า 12 ของสตริงคำตอบตัวที่ 8 ก็ให้เอาค่า 12 นี้ไปเปรียบเทียบกับค่า Fitness ของ Elite Preserve Solution ถ้าหากดังกล่าวน้อยกว่า 12 ก็ให้เอาสตริงคำตอบตัวที่ 8 นี้ไปใช้เป็น Elite Preserve Solution ตัวใหม่แทน แต่ถ้าค่าดังกล่าวมากกว่าหรือเท่ากับ 12 ก็ให้คงคำตอบของ Elite Preserve Solution ไว้ แล้วนำสตริงคำตอบทั้ง 10 ตัวนี้ไปทำการมิวเตชั่นต่อไป

3) Elite Preserve Strategy of Generation

เป็นเทคนิคการเก็บค่าที่ดีที่สุดที่ใช้ภายหลังการมิวเตชั่น ซึ่งถือว่าเป็นการเก็บค่าที่ดีที่สุดของเจนเนอเรชั่นนั้นๆด้วย การเก็บค่าที่ดีที่สุดของเจนเนอเรชั่นจะช่วยให้คำตอบที่ดีที่สุดเท่าที่เคยปรากฏขึ้นมายังคงมีอยู่ในเจนเนอเรชั่นต่อไป การเก็บค่าในขั้นตอนนี้จะทำหลังจากที่มีการมิวเตชั่นเรียบร้อยแล้ว สตริงคำตอบที่ได้ภายหลังการมิวเตชั่นจำนวน *popsize* ตัว จะถูกถอดรหัสและประเมินค่า จากนั้นก็ให้เอาสตริงคำตอบหลังที่ดีที่สุดจากการมิวเตชั่น มาเปรียบเทียบกับ Elite Preserve Solution เช่นเดียวกับในขั้นตอนของ Post-crossver Elite Preserve Strategy แต่แตกต่างกันตรงที่ จะมีการนำเอา Elite Preserve Solution มาแทนที่คำตอบที่แย่ที่สุดของสตริงคำตอบชุดนี้เมื่อ Elite Preserve Solution เป็นคำตอบที่ดีกว่า

ยกตัวอย่างเช่น ภายหลังการมิวเตชั่นได้สตริงคำตอบ 10 ตัว ที่มีค่า Fitness เป็น 5 6 8 3 1 9 4 6 7 7 จะได้ว่าค่า Fitness ที่ดีที่สุดคือ 9 ของสตริงคำตอบตัวที่ 6 ซึ่งถ้าค่า Fitness ของ Elite Preserve Solution น้อยกว่า 9 สตริงคำตอบตัวที่ 6 จะกลายเป็น Elite Preserve Solution ตัวใหม่ แต่ถ้าค่า Fitness ของ Elite Preserve Solution มากกว่า 9 ก็ให้ตัดสตริงคำตอบตัวที่ 5 ซึ่งมีค่า Fitness ต่ำที่สุดทิ้งไป เอาสตริงคำตอบที่เป็น Elite Preserve Solution ขณะนั้นไปใส่แทน

สตริงคำตอบที่ได้ภายหลังจากขั้นตอนนี้จะกลายเป็นสตริงคำตอบพ่อแม่ที่แท้จริงในเจนเนอเรชั่นต่อไป

5.4 สรุปท้ายบท

วิธีการเจเนติกอัลกอริทึมที่ใช้สำหรับปัญหาการจัดสมดุลของสายงานการประกอบ แบ่งเป็น 5 ส่วนหลักคือการสร้างประชากรเบื้องต้น (Initialization) การรีโปรดักชั่น (Reproduction)

การครอสโอเวอร์ (Crossover) การมิวเตชั่น (Mutation) และเทคนิคการเก็บค่าที่ดีที่สุด (Elite Preserve Strategy)

การสร้างประชากรเบื้องต้นทำได้โดยการสร้างคำตอบเบื้องต้นจำนวนหนึ่งในรูปของสตริงคำตอบที่ได้จากการใส่รหัสแบบ Sequence-oriented Representation โดยพิจารณาลำดับความสัมพันธ์ก่อนหลังร่วมด้วย ในส่วนของการรีโปรดักชั่นจะแบ่งเป็น 3 ส่วนย่อยคือการถอดรหัส (Decoding) ซึ่งเป็นการนำขั้นงานในสตริงคำตอบมาจัดให้กับสถานีทำงานตามลำดับ การประเมินค่า (Evaluation) เป็นการคำนวณหาค่าความเหมาะสมของสตริงคำตอบแต่ละตัว และการคัดเลือก (Selection) ซึ่งเป็นการคัดเลือกสตริงตัวที่มีความเหมาะสมมากกว่าเพื่อเข้าสู่กระบวนการถัดไปโดยวิธีการคัดเลือกแบบ Tournament Selection เมื่อเข้าสู่กระบวนการครอสโอเวอร์ สตริงจะถูกสุ่มเลือกด้วยความน่าจะเป็นในการครอสโอเวอร์เพื่อมาจับคู่และทำการครอสโอเวอร์ วิธีครอสโอเวอร์ที่ใช้มีหลายวิธีคือ MOX PMX CX OX PBX OBX ร่วมกับวิธีซ่อมแซมคำตอบเพื่อให้ได้สตริงคำตอบที่เป็นไปได้ จากนั้นสตริงคำตอบจะถูกสุ่มเลือกไปทำการมิวเตชั่นด้วยความน่าจะเป็นในการมิวเตชั่น วิธีมิวเตชั่นที่ใช้คือวิธี Random sequence Mutation ซึ่งสามารถให้คำตอบที่เป็นไปได้โดยไม่ต้องผ่านการซ่อมแซมคำตอบ สตริงคำตอบที่ได้จะกลายเป็นประชากรพ่อแม่ในเจนเนอเรชั่นต่อไป กระบวนการของ GAs จะเกิดขึ้นซ้ำไปเรื่อยๆจนกว่าจะถึงเจนเนอเรชั่นสูงสุดที่กำหนด

ในระหว่างกระบวนการของ GAs มีการนำเทคนิคการเก็บค่าที่ดีที่สุดเข้าไปใช้ภายหลังการสตริงคำตอบเบื้องต้น ภายหลังการครอสโอเวอร์ และภายหลังการมิวเตชั่น ทั้งนี้เพื่อให้สตริงคำตอบที่ดียังคงอยู่ต่อไปในเจนเนอเรชั่นถัดไป