บทที่ 4

ทฤษฎีการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์และเจนเนติกอัลกอริทึม

ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ เราจำเป็นที่จะต้องศึกษาทฤษฎีและเทคทิคต่าง ๆ ที่ใช่ในการแก้ปัญหาการหาค่าเหมาะสมที่สุด ดังนั้นเนื้อหาในบทนี้จึงเกี่ยวข้องกับหลักการพื้นฐานของการหาค่าเหมาะสมที่สุด ลักษณะปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ การแก้ปัญหาการหาค่าเหมาะสมที่สุดด้วยเมทธาฮิวริสติกรวมไปถึงการแก้ปัญหาด้วยวิธีการทางวิวัฒนาการหรือเอลโวลูชันนารี่อัลกอริทึมแบบหลายวัตถุประสงค์ ที่จัดว่าเป็นเมทธาฮิวริสติกหนึ่งที่มีความสามารถในการหาคำตอบได้อย่างมีประสิทธิภาพ และการวัดสมรรถนะของกลุ่มคำตอบที่ดีที่สุด ซึ่งมีรายละเอียดดังนี้

4.1 หลักการพื้นฐานของการหาค่าเหมาะสมที่สุด

การหาค่าเหมาะสมที่สุด (Optimization) เป็นวิธีการที่ใช้ในการหาคำตอบที่ดีที่สุดของปัญหาภายใต้เงื่อนไขหรือข้อจำกัดที่กำหนดขึ้น การหาค่าเหมาะสมที่สุดถือว่าเป็นสิ่งที่ช่วยในการแก้ปัญหาในด้านวิทยาการคอมพิวเตอร์ (Computer Science) ปัญญาประดิษฐ์ (Artificial intelligence) การวิจัยการดำเนินงาน (Operation Research) และสาขาอื่น ๆ ที่เกี่ยวข้องได้เป็นอย่างดี โดยจะแบ่งปัญหาการหาค่าเหมาะสมที่สุด ออกเป็น 2 รูปแบบตามการพิจารณาฟังก์ชันวัตถุประสงค์หรือฟังก์ชันเป้าหมาย คือการหาค่าเหมาะสมที่สุดที่พิจารณาฟังก์ชันพิจารณาฟังก์ชันวัตถุประสงค์เพียงวัตถุประสงค์ เรียกปัญหาลักษณะนี้ว่าเป็นปัญหาการหาค่าเหมาะสมที่สุดแบบวัตถุประสงค์เดียว (Single Objective Optimization Problem) ส่วนปัญหาที่พิจารณาฟังก์ชันวัตถุประสงค์มากกว่า 1 ฟังก์ชันพร้อม ๆ กัน ในรูปแบบปัญหาลักษณะนี้อาจมีฟังก์ชันวัตถุประสงค์มีความขัดแย้งกัน หรือเป็นไปในแนวทางเดียวกัน และเรียกว่าเป็นปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ (Multi-Objective Optimization Problem) หรือปัญหาการหาค่าเหมาะสมที่สุดที่พิจารณาหลายเกณฑ์ (Multi-Criteria Optimization Problem) หรือปัญหาการหาค่าเหมาะสมที่สุดแบบเวคเตอร์ (Vector Optimization Problem) (Osyczda, 1985) ซึ่งการหาค่าเหมาะสมที่สุดนี้จะประกอบด้วยเวคเตอร์ตัวแปรตัดสินใจ (Vector of Decision Variables) ข้อจำกัด (Constraints) และเวกเตอร์ฟังก์ชัน (Vector Functions) ที่สามารถเรียกว่าเป็น ฟังก์ชันวัตถุประสงค์(Objective Functions)

4.1.1 ฟังก์ชันวัตถุประสงค์

ในขั้นตอนการหาคำตอบที่ดีที่สุดนั้น จะมีเกณฑ์ (Criteria) ที่ใช้ในการคำนวณค่าเพื่อหาคำตอบที่ดที่สุด โดยที่เกณฑ์นั้จะเป็นตัวกำหนดเป้าหมายในค้นหาคำตอบว่าเป็นไปในลักษณะใดซึ่งเรียกว่า ฟังก์ชันวัตถุประสงค์ (Objective Function) โดยมากฟังก์ชันวัตถุประสงค์จะเกี่ยวข้องกับการหาค่ามากที่สุด (Maximization) น้อยที่สุด (Minimization) ของฟังก์ชันวัตถุประสงค์นั้น ๆ เช่น ในกรณีการจัดสมดุลสายการประกอบ ที่มีฟังก์วัตถุประสงค์เพื่อหาจำนวนสถานีงานที่น้อยที่สุด เป็นต้น

ในกรณีที่มีฟังก์ชันวัตถุประสงค์หลายฟังก์ชันที่ต้องถูกดำเนินการในการหาค่าเหมาะสมที่สุด กรณีเหล่านี้จะค่อนข้างยุ่งยากในการคำนวณ และต้องอาศัยเทคนิคและประสบการณ์ในการแก้ปัญหาเฉพาะหน้าในการกำหนดฟังก์ชันวัตถุประสงค์ อย่างไรก็ตามฟังก์ชันวัตถุประงค์เป็นเพียงเครื่องมือการออกแบบรูปแบบเบื้องต้นในการแก้ปัญหาเท่านั้น ดังนั้นฟังก์ชันวัตถุประสงค์ควรที่จะถูกเลือกในทางที่ทำให้วัตถุประสงค์บรรลุเป้าหมายที่วางไว้

4.1.2 ตัวแปรตัดสินใจ

ตัวแปรตัดสินใจ (Decision Variable) คือตัวแปรที่สามารถใช้ในการปรับเปลี่ยนหรือควบคุมขั้นตอนในการหาค่าเหมาะสมที่สุดที่ทำให้ฟังก์ชันวัตถุประสงค์เปลี่ยนแปลงได้ โดยทั่วไปแล้วปัญหาการหาค่าเหมาะสมที่สุดมักเกี่ยวข้องกับตัวแปรตัดสินใจมากกว่าหนึ่งตัวแปร ดังนั้นสิ่งสำคัญคือการคัดเลือกตัวแปรตัดสินใจที่มีผลต่อฟังก์ชันวัตถุประสงค์มากที่สุด ซึ่งการพิจารณาไว้ล่วงหน้าที่เหมาะสม จะช่วยเพิ่มประสิทธิภาพและความเร็วในการได้มาซึ่งคำตอบ

ตัวแปรตัดสินใจ จะเป็นตัวแปรที่สามารถวัดเป็นเชิงปริมาณได้ ซึ่งค่าที่นำมาใช้ในการเลือกในปัญหาการหาค่าเหมาะสมที่สุด สามารถแสดงเป็นตัวแปร x_j โดยที่ $j = 1,2,...,n$ และเวคเตอร์ของตัวแปรตัดสินใจ n ตัว สามารถเขียนได้เป็น

$$\bar{x} = \begin{bmatrix} x1 \\ x2 \\ \vdots \\ x_n \end{bmatrix} \tag{4.1}$$

หรือสามารถเขียนได้เป็น $\bar{x} = [x_1, x_2, ..., x_n]^T$ (4.2)

โดยปกติปัญหาต่าง ๆ ในทางวิศวกรรม สามารถนิยามเป็นฟังก์ชันของตัวแปรตัดสินใจได้ จำนวนของตัวแปรตัดสินใจจะขึ้นอยู่กับความซับซ้อนของปัญหา ซึ่งฟังก์ชันของตัวแปรตัดสินใจสามารถแบ่งได้เป็น 2 แบบ คือฟังก์ชันกำหนด (Deterministic Function) เป็นฟังก์ชันที่มีแนวทางการหาคำตอบอย่างแน่นอน คือฟังก์ชันที่สามารถนิยามเป็นสมการทางคณิตศาสตร์ได้ และฟังก์ชันเฟ้นสุ่ม (Stochastic Function) คือฟังก์ชันที่มีแนวทางในการหาคำตอบที่ไม่สามารถกำหนดได้อย่างชัดเจน หรือไม่สามารถนิยามเป็นสมการทางคณิตศาสตร์ได้ เช่น ปัญหาการหาเส้นทางเดินที่ดีที่สุดของพนักงานขาย (Traveling Salesman Problem: TSP) เป็นต้น

4.1.3 ข้อจำกัด

หลังจากที่ได้มีการเลือกตัวแปรตัดสินใจของปัญหาได้แล้ว จะพบว่าตัวแปรตัดสินใจจะต้องมีความสอดคล้องกับข้อเท็จจริงต่าง ๆ ทางกายภาพ ทางเคมี หรืออื่น ๆ ของระบบ โดยอาจกำหนดข้อจำกัดด้านทรัพยากร หรือสภาพแวดล้อม เช่น ข้อจำกัดทางวัตถุดิบเวลา เป็นต้น แล้วแต่กรณี สิ่งเหล่านี้จะเรียนว่าเป็นข้อจำกัด(Constraints) ในปัญหาการหาค่าเหมาะสมที่สุด ซึ่งข้อจำกัดนี้จะต้องอยู่ในความเป็นไปได้ของคำตอบที่สามารถยอมรับได้นอกจากนี้ข้อจำกัดยังขึ้นอยู่กับตัวแปรตัดสินใจ ซึ่งสามารถเขียนอยู่ในรูปข้อจำกัดที่แบบอสมการ (Inequality Constraints)

$$g_i(\bar{x}) \geq a_i, i = 1,2,...,m \tag{4.3}$$

หรือข้อจำกัดแบบสมการ(Equality Constraints)

$$h_i(\bar{x}) \geq b_i, i = 1,2,...,p \tag{4.4}$$

4.2 ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์

4.2.1 รูปแบบปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์

การแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ เป็นการค้นหาเซตคำตอบภายในพื้นที่ของคำตอบที่เป็นไปได้ เพื่อต้องการหาค่าที่ต่ำที่สุด หรือค่าสูงสุดของฟังก์ชันวัตถุประสงค์ในแต่ละฟังก์ชันพร้อม ๆ กัน ดังสมการที่ (4.5) โดยผลลัพธ์ที่ได้จากการแก้ปัญหาคือ *เซตกลุ่มคำตอบที่ดีที่สุด*

$$f(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}),..., f_k(\bar{x})]^T \tag{4.5}$$

ดังนั้นรูปแบบปัญหาการหาค่าเหมาะสมที่สุดทีมีหลายวัตถุประสงค์ จะเป็นการค้นหาเวกเตอร์คำตอบ $\bar{x}^* = [x_1^*, x_2^*, ..., x_n^*]^T$ ภายใต้ m ข้อจำกัดแบบอสมการ ดังสมการที่ (3.6) หรือภายใต้ p ข้อจำกัดแบบสมการ ดังสมการที่ (4.7)

$$g_i(\bar{x}) \geq 0, i = 1,2,...,m \tag{4.6}$$

$$h_i(\bar{x}) \geq 0, i = 1,2,...,p \tag{4.7}$$

ข้อจำกัดในสมการในสมการที่ (4.6) และ (4.7) จะเป็นการกำหนดขอบเขตพื้นที่คำตอบที่เป็นไปได้ (Feasible Region: Ω) และทุก ๆ จุดใน Ω ก็คือคำตอบที่เป็นไปได้ (Feasible Solution) โดยที่เวกเตอร์ฟังก์ชันวัตถุประสงค์ประสงค์ $f(\bar{x})$ ที่ใช้นี้จะทำการค้นหาคำตอบในเซต Ω ให้กลายเป็น เซต กลุ่มคำตอบที่ดีที่สุด (Λ) ภายใต้ค่าเป็นไปได้ทั้งหมดของฟังก์ชันวัตถุประสงค์ $f(\bar{x})$ที่มี k วัตถุประสงค์ และมี $\bar{x}^*$ แทนคำตอบที่ดีที่สุด

โดยทั่วไปแล้ว รูปแบบปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ มี 3 รูปแบบที่เป็นไปได้ดังนี้

- ทุกฟังก์ชันวัตถุประสงค์ต้องการหาค่าน้อยที่สุด
- ทุกฟังก์ชันวัตถุประสงค์ต้องการหาค่ามากที่สุด
- บางฟังก์ชันวัตถุประสงค์ต้องการหาค่าน้อยที่สุด บางฟังก์ชันวัตถุประสงค์ต้องการหาค่ามากที่สุด

ในการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์สามารถแปลงฟังก์ชันวัตถุประสงค์ให้มีรูปแบบเป็นการหาค่าน้อยที่สุด หรือมากที่สุดได้ ตัวอย่างเช่น การแปลงทุกฟังก์ชันวัตถุประสงค์ที่มีการหาค่าน้อยที่สุดให้เป็นรูปแบบการหาค่ามากที่สุด

$$\max f_i(\bar{x}) = -\min(-f_i(\bar{x})) \tag{4.8}$$

ส่วนรูปแบบของข้อจำกัดนั้นก็สามารถเปลี่ยนรูปแบบ

$$g_i(\bar{x}) \leq 0, i = 1,2,...,m \tag{4.9}$$

และให้อยู่ในรูป

$$-g_i(\bar{x}) \geq 0, i = 1,2,...,m \tag{4.10}$$

4.2.2 กลุ่มคำตอบที่ดีที่สุด

โดยทั่วไปแล้ว ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ (Multi-objective Optimization: MOP) จะประกอบด้วยฟังก์ชันวัตถุประสงค์ k วัตถุประสงค์ และตัวแปรตัดสินใจ n ตัว โดยรูปแบบปัญหาที่อาจเป็นการหาค่ามากที่สุด หรือการหาค่าน้อยที่สุดจะเป็นการกำหนดเป้าหมายในค้นหาคำตอบว่าเป็นไปในลักษณะใด หรือสามารถเขียนได้ดังสมการที่ (4.11)

$$Minimize\ /\ Maximize\ \{f_1(\bar{x}), f_2(\bar{x}),..., f_k(\bar{x})\} \tag{4.11}$$

สำหรับรูปแบบปัญหาการหาค่าฟังก์ชันวัตถุประสงค์ในที่นี้เป็นการหาค่าน้อยที่สุด และตลอดเนื้อหาในบทนี้ก็จะมีรูปแบบปัญหาเช่นเดียวกันกับสมการที่ (4.12) ซึ่งการค้นหาคำตอบจะถูกกำหนดจากเวกเตอร์ของตัวแปรตัดสินใจ ภายใต้ข้อจำกัดที่จะเป็นสิ่งที่ใช้ในการกำหนดขอบเขตคำตอบที่น้อยที่สุด สามารถเขียนได้ดังสมการที่ (4.12)

$$Minimize\ \{f_1(\bar{x}), f_2(\bar{x}),..., f_k(\bar{x})\} \tag{4.12}$$

ข้อจำกัด $g_i(\bar{x}) \le 0$

โดยที่ $\bar{x}$ คือ เวกเตอร์ของตัวแปรตัดสินใจ

$f_i(\bar{x})$ คือ ฟังก์ชันวัตถุประสงค์ที่ i

$g_i(\bar{x})$ คือ เวกแตอร์ข้อจำกัดที่ i

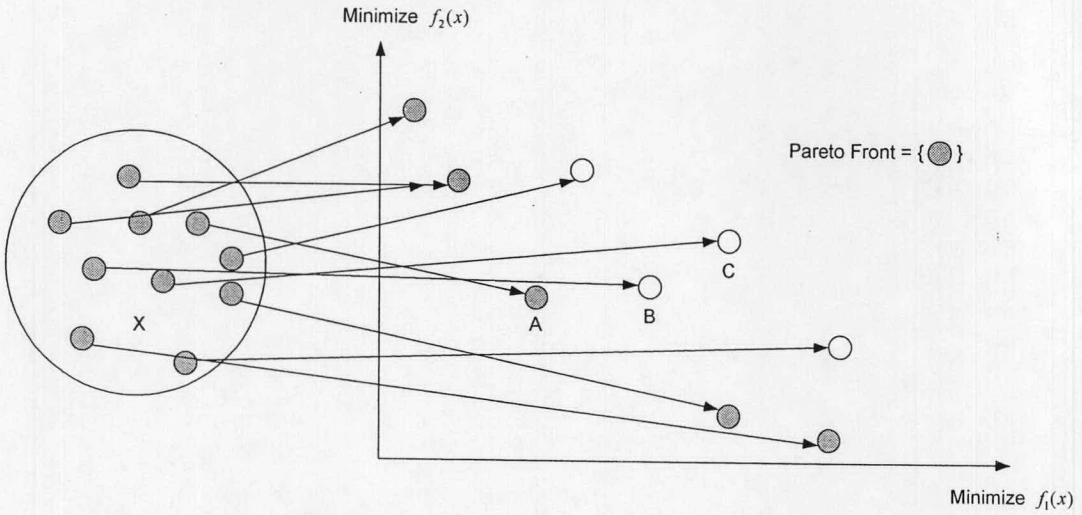
ถ้าเวกเตอร์ตัดสินใจ x ให้คำตอบที่ดีกว่าหรือครอบงำ (Dominated) เวกเตอร์ตัดสินใจ y (เขียนได้เป็น $x \succ y$) แล้ว

$f_i(x) \le f_i(y)$ สำหรับทุกค่า $i \in \{1,2,...,k\}$ และ

$f_i(x) < f_i(y)$ มีอย่างน้อย 1 ค่าของ $i \in \{1,2,...,k\}$

นั่นคือ ถ้าคำตอบที่ได้ในพื้นที่คำตอบที่เป็นไปได้ เป็นคำตอบที่ไม่มีคำตอบใดดีกว่าหรือไม่มีคำตอบใดที่สามารถครอบงำชุดคำตอบนี้ได้ จะเรียกคำตอบนี้ว่าเป็น กลุ่มคำตอบที่ดีที่สุด (Pareto Optimal) และเรียกสมาชิกคำตอบทุกคำตอบที่อยู่ในกลุ่มคำตอบที่ดีที่สุด ว่าเซตกลุ่มคำ

ตอบที่ดีที่สุด (Pareto Optimal Set) หรือเซตคำตอบที่ไม่ถูกครอบงำจากทุกคำตอบ (Non-dominated Set) หรือเซตคำตอบที่มีประสิทธิภาพ (Efficient Set) ซึ่งเซตคำตอบนี้จะใช้ในการกำหนดพื้นที่ขอบเขตของคำตอบ และเรียกว่า ขอบเขตของกลุ่มคำตอบที่ดีที่สุด (Pareto Optimal Frontier) หรือขอบเขตของคำตอบที่ไม่ถูกครอบงำจากทุกคำตอบ (Non-dominated Frontier) หรือขอบเขตของคำตอบที่มีประสิทธิภาพ (Efficient Frontier)



รูปที่ 4.1 การค้นหาพื้นที่คำตอบในปัญหาการหาค่าเหมาะสมที่สุดหลายวัตถุประสงค์

จากรูปที่ 4.1 แสดงตัวอย่างรูปแบบปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ ที่มีเป้าหมายเพื่อหาค่าที่น้อยที่สุดของ 2 ฟังก์ชันวัตถุประสงค์พร้อมกัน โดยกำหนดให้ X เป็นพื้นที่คำตอบ และ z เป็นเวกเตอร์ฟังก์ชันความแข็งแรง (Vector Fitness Function) ซึ่งจะทำการค้นหาในพื้นที่คำตอบ X ไปสู่คำตอบที่ได้มาจากเวกเตอร์วัตถุประสงค์ $f_1(x)$ และ $f_2(x)$ ให้มีค่าน้อยที่สุด โดยที่เวกเตอร์วัตถุประสงค์ที่ดีที่สุด (จุดทึบ) คือเวกเตอร์วัตถุประสงค์ที่ไม่มีค่าใดดีกว่า เรียกว่า เซตคำตอบที่ไม่ถูกครอบงำจากทุกคำตอบ และสมาชิกคำตอบที่ไม่ถูกครอบงำจากทุกคำตอบนี้ จะประกอบขึ้นเป็นขอบเขตกลุ่มคำตอบที่ดีที่สุด จะเห็นได้ว่าคำตอบที่ตรงกันกับจุดทึบ ก็คือ กลุ่มคำตอบที่ดีที่สุด(Non-dominated Optimal) ซึ่งจาก 3 เวกเตอร์คำตอบ A B และ C สามารถเขียนได้เป็น $A \succ B \succ C$

สำหรับเป้าหมายของปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ คือ การระบุเซตกลุ่มคำตอบที่ดีที่สุด ซึ่งการระบุถึงเซตกลุ่มคำตอบที่ดีที่สุดที่แท้จริง (True Pareto Optimal หรือ Reference Pareto Optimal) นั้น เป็นสิ่งที่เป็นไปไม่ได้ในทางปฏิบัติ เนื่องจากปัญหาที่มีความเฉพาะเจาะจง อย่างเช่นปัญหาการหาค่าเหมาะสมที่สุดในเชิงการจัด (Combinatorial Optimization Problem) ขนาดของปัญหาที่ใหญ่ขึ้น ส่งผลให้คำตอบที่เป็นไปได้มีจำนวนมากขึ้น ในลักษณะเอ็กโปเนนเชียล และเป็นปัญหา NP-hard ดังนั้นการหาค่าคำตอบที่ดี

ที่สุด จึงเป็นเพียงสิ่งที่เป็นไปได้ยาก นอกจากนี้ฟังก์ชันวัตถุประสงค์ของปัญหาที่มีความสัมพันธ์ตรงกันข้ามกัน ยังก่อให้เกิดความสับสน ดังนั้นในทางปฏิบัติวิธีการที่ใช้ปัญหาในการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ จะเป็นการสืบค้นถึงเซตกลุ่มคำตอบที่สามารถใช้แทนเซตกลุ่มคำตอบที่ดีที่สุดที่แท้จริงได้ โดยสิ่งที่ทำให้การแก้ปัญหาการหาค่าเหมาะสมที่สุดบรรลุเป้าหมายมีดังนี้

- ขอบเขตกลุ่มคำตอบที่ดีที่สุด ควรจะใกล้เคียงกับขอบเขตกลุ่มคำตอบที่ดีที่สุดแท้จริง และควรเป็นเซตย่อย (Sub Set) ของขอบเขตกลุ่มคำตอบที่ดีที่สุดที่แท้จริง
- เซตของกลุ่มคำตอบที่อยู่บนขอบเขตกลุ่มคำตอบที่ดีที่สุดนี้ควรมีลักษณะการกระจายแบบสม่ำเสมอ (Uniform Distribution) หรือมีคำตอบอยู่บนขอบเขตกลุ่มคำตอบอย่างทั่วไปถึง ไม่เกาะอยู่บริเวณใดบริเวณหนึ่งนั่นคือคำตอบนี้มีความสามารถครอบคลุมขอบเขตของกลุ่มคำตอบที่ดีที่สุด
- ขอบเขตกลุ่มคำตอบที่ดีที่สุด ควรจับสเปกตรัมของขอบเขตกลุ่มคำตอบได้ทั้งหมด หมายถึงมีความสามารถในการสืบค้นถึงคำตอบที่อยู่ปลายสุดของคำตอบในพื้นที่ฟังก์ชันวัตถุประสงค์

4.3 การแก้ปัญหาการหาค่าเหมาะสมที่สุดด้วยเมทธาฮิวริสติก

ส่วนใหญ่ปัญหาที่เกี่ยวข้องกับการตัดสินใจในด้านธุรกิจและเศรษฐศาสตร์รวมไปถึงด้านการผลิต เส้นทาง สถานที่ตั้ง และการจัดตาราง ล้วนแต่เป็นปัญหาการหาค่าเหมาะสมที่สุด (Optimization Problem) และยากเกินกว่าที่จะแก้ปัญหาได้อย่างถูกต้อง ภายใต้ระยะเวลาอันจำกัด ดังนั้นฮิวริสติก (Heuristic) จึงกลายเป็นวิธีที่เป็นตัวเลือกที่ใช้ในการแก้ปัญหาประเภทนี้ ฮิวริสติกเป็นวิธีการที่ไม่มีแนวทางหรือกฎเกณฑ์ที่แน่นอนตายตัว มักอาศัยประสบการณ์ที่ผ่านมาเข้ามาช่วย ซึ่งอาจได้คำตอบที่ไม่ดีนัก แต่โดยเฉลี่ยล้วคำตอบที่ได้จะเป็นคำตอบที่ดีถึงแม้ว่าแนวทางนี้จะไม่ได้รับประกันว่าสุดท้ายแล้วจะได้คำตอบที่ดีที่สุดก็ตาม แต่เนื่องจากเป็นวิธีที่ไม่ยุ่งยากในการคำนวณและเวลาในการคำนวณ แนวคิดนี้จึงได้รับการยอมรับจากนักวิชาการ

การได้มาของคำตอบที่เป็นไปได้อย่างง่ายแต่คุณภาพของคำตอบไม่ดีนะ ถือว่าเป็นสิ่งที่ยังต้องการวิธีการที่มีคุณภาพ เพื่อให้ได้มาซึ่งคำตอบที่เป็นไปได้ที่ดีที่สุดอยู่ภายใต้เวลาที่ใช้ในทางปฏิบัติที่จำกัด เนื่องจากส่วนใหญ่ปัญหาการหาค่าเหมาะสมที่สุดมักมีความซับซ้อนโดยเฉพาะในทางปฏิบัติขนาดปัญหาที่ใหญ่จะทำให้เกิดความยากในการเลือกวิธีการแก้ปัญหาที่เหมาะสม บ่อยครั้งที่การเลือกใช้วิธีการหาคำตอบแบบการประมาณค่า (Exact Algorithm) แล้วทำให้

สูญเสียเวลาในการหาคำตอบ ดังนั้นในทางปฏิบัติฮิวริสติกจึงเป็นวิธีที่ดีกว่าที่สามารถประยุกต์ใช้ในทางปฏิบัติได้อย่างเหมาะสม

ความหมายของฮิวริสติกและเมทธาฮิวริสติก

ฮิวริสติก ถูกแนะนำเป็นครั้งแรกโดย Polya (1945) หลักการพื้นฐานของการค้นหาแบบฮิวริสติก เป็นเทคนิคการประมาณคำตอบด้วยการใช้สามัญสำนึกและคาดหวังว่าคำตอบที่ได้รับจะเป็นคำตอบที่ดี ภายใต้เวลาที่จำกัด แต่สุดท้ายก็ไม่ได้รับประกันว่าคำตอบนั้นจะเป็นคำตอบที่ดีที่สุด นอกจากนี้ฮิวริสติกยังเป็นวิธีที่มีพื้นฐานมาจากการประยุกต์รูปแบบการค้นหา Greedy รวมไปถึงวิธีการแทรก (Insertion Procedure) และกฎการจ่ายงาน (Dispatching Rules) และยังเป็นแนวคิดที่ทำให้เกิดการพัฒนาวิธีการปรับปรุงคำตอบ ซึ่งถือว่าเป็นกำเนิดของการค้นหาเฉพาะที่ (Local Search) เพื่อไม่ให้คำตอบที่ได้เป็นคำตอบที่ดีที่สุดเฉพาะที่ (Local Optimum) และเป็นอุปสรรคหนึ่งของความสามารถของวิธีการนี้ ดังนั้นการพิจารณาเทคนิคของฮิวริสติกที่จะใช้เป็นแนวทางในการค้นหาคำตอบจะต้องมีความสามารถหลีกเลี่ยงผลเสียที่เกิดจากการปรับปรุงคำตอบด้วยการทำซ้ำ จึงเป็นการค้นหาวิธีที่มีความสามารถทำให้ได้คำตอบที่ดีและใช้เวลาในการหาคำตอบได้รวดเร็ว หรือเรียกว่าเป็นการค้นหาอย่างชาญฉลาด (Intelligent Search Method) ที่เรียกว่า "เมทธาฮิวริสติก" (Metaheuristic) ที่มีรูปแบบวิธีการหาคำตอบแบบการประมาณค่าและการรวมกันขั้นพื้นฐานของวิธีการทางฮิวริสติกที่สามารถช่วยในการค้นหาพื้นที่คำตอบได้อย่างมีประสิทธิภาพ

เมทธาฮิวริสติกได้รับการแนะนำครั้งแรกโดย Glover (1986) ได้กล่าวว่าเมทธาฮิวริสติกเป็นกลยุทธ์ระดับสูงที่ใช้ในการให้แนวทางในการหาคำตอบแบบการค้นหาซ้ำๆ ที่อาศัยพื้นฐานวิธีการฮิวริสติก นั้นคือเป็นกระบวนการให้แนวทางในการค้นหาพื้นที่คำตอบได้อย่างครอบคลุมและคาดหวังว่าคำตอบที่ได้นั่นเป็นคำตอบที่ดีที่สุด โดยที่อัลกอริทึมต่างๆที่จัดว่าเป็นเมทธาฮิวริสติก ได้แก่ ระบบโครงข่ายประสาทเทียม (Artificial Neural Network: ANN) เอนโคโลนี่ (Ant Colony System: ACO) เอลโวลูชันนารี่อัลกอริทึม (Evolutionaly Algorithms: EAs) เจนเนติกอัลกอริทึม (Ganetic Algorithms: GAs) ซิมูเลทแอนนีลลิ่ง (Simulated Annealing: SA) ทาบู เซิร์ท (Tabu Search: TS) และเมมเมติกอัลกอริทึม (Memetic Algorithms: MAs) เป็นต้น

รายละเอียดโดยสรุปของเมทธาฮิวริสติกที่ได้รับความนิยมใช้ในการแก้ปัญหาการหาค่าเหมาสมที่สุด ดังนี้

ซิมูเลทแอนเนลลิ่ง

SA (Kirkpatric, Gelatt and Vecchi, 1993) (William, et al., 1992) (Zomaya, 2001) เป็นเมทธาฮิวริสติกที่ใช้การสุ่มจุดคำตอบใหม่จากบริเวณข้างเตียงของจุดคำตอบเดิม โดยมีหลักการจากการเลียนแบบจากการตกผลึกของโลหะในช่วงที่เย็นตัว คือโมเลกุล (Molecule) สามารถมีการเคลื่อนที่อย่างอิสระที่อุณหภูมิสูงเปรียบเทียบกับการที่สามารถเลือกจุดคำตอบใหม่จากบริเวณจุดคำตอบเริ่มต้นได้อย่างอิสระ และจะเคลื่อนที่ช้าลงเมื่ออุณหภูมิลดต่ำลงอย่างช้าๆ เปรียบได้กับการที่มีข้อกำหนดมากขึ้นในการเลือกจุดคำตอบใหม่จากบริเวณจุดเดิม และในที่สุดอะตอม (Atom) จะมีการเรียงตัวจนกลายเป็นผลึกที่สมบูรณ์และมีเสถียรภาพเนื่องจากมีพลังงานน้อยที่สุดเปรียบได้กับคำตอบที่ดีที่สุดของฟังก์ชัน

ทาร์บู เซิร์ท

TS (Glover, 1989) (Glover, 1990) (Glover and Laguna, 1990) เป็นวิธีการหาคำตอบแบบสุ่ม มีหลักการมาจากการหาคำตอบที่ดีกว่าบริเวณข้างเคียงภายในพื้นที่การค้นหา โดยขั้นแรกจำกำหนดกลุ่มคำตอบเริ่มต้นในพื้นที่การค้นหาจากการสุ่มเช่นเดียวกันกับวิธีซิมูเลทแอนนีลลิ่ง จากนั้นทำการหาคำตอบที่ดีกว่าจากบริเวณข้างเคียงของกลุ่มคำตอบเริ่มต้น โดยมีการเปรียบเทียบไม่ให้ซ้ำกับกลุ่มคำตอบที่เคยหาผ่านมาแล้วซึ่งอยู่ใน (Tabu List) เมื่อได้กลุ่มคำตอบใหม่จากกลุ่มคำตอบเดิมแล้วทำการหากลุ่มคำตอบใหม่ต่อไปเรื่อยๆ จากกลุ่มคำตอบเดิม

เอลโวลูชันนารี่อัลกอริทึม

EAs เป็นเทคนิคการหาค่าเหมาะสมที่สุดที่อาศัยการค้นหาแบบอิงฐานประชากร (Population Base Search) และเป็นเมทธาฮิวริสติกตัวหนึ่งที่ได้รับความนิยมเป็นอย่างมากในปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ EAs จะทำการค้นหาสมาชิกคำตอบแบบหลายจุดพร้อมกัน ซึ่งทำให้เกิดคำตอบที่เป็นไปได้ นอกจากนี้ยังเป็นเทคนิคที่สามารถใช้แก้ปัญหาที่มีความซับซ้อนได้อย่างมีประสิทธิภาพ และใช้ได้อย่างกว้างขวางกับทุกปัญหาการหาค่าเหมาะสมที่สุด เมทธาฮิวริสติกที่จัดได้ว่าเป็น EAs ได้แก่ Ganetic Algorithms Evolutionary Strategy, Evolutionary Program, Scatter Search และ Memetic Algorithms

Gas เป็นการค้นหาคำตอบแบบเฟ้นสุ่ม (Stochastic Search) หรือเป็นเทคนิคการหาค่าเหมาะสมที่สุดที่ลักษณะการทำงานในรูปแบบของการค้นหาแบบฮิวริสติกซึ่งมีรากฐานแนวความคิดมาจากทฤษฎีวิวัฒนาการชาร์ล ดาร์วิน (Charles Darwin) โดยอิงจากแนวความคิดการอยู่รอดของผู้ที่แข็งแรงที่สุด (Survival of the fittest) ในสภาวะแวดล้อมที่เหมือนกัน การทำงานของ Gas นี้จะเป็นไปในลักษณะการค้นหาคำตอบแบบคู่ขนาน (Parallel Search) โดยคำตอบที่ได้จากการหาคำตอบในหนึ่งรุ่น (Generation) จะผ่านการแปลง (Transformation) เพื่อที่จะนำไปสู่การค้นหาคำตอบที่ดีขึ้นในรุ่นถัดไป การเปลี่ยนแปลงที่เกิดขึ้นกับคำตอบ (Solution) หรือสมาชิกของประชากร (Individual) ภายในประชากร (Population) หนึ่งรุ่นนั้นจะเป็นไปเพื่อการสำรวจพื้นที่ในการค้นหา (Search Space) และส่งเสริมให้มีการถ่ายทอดคุณสมบัติที่ดี (Fit Characteristics) ของคำตอบที่ค้นพบในรุ่นปัจจุบันไปยังรุ่นถัดไป สมาชิกของประชากรที่มีคุณลักษณะที่ดีจะมีหลายคำตอบด้วยกันในประชากรที่ได้มาจากการค้นหาโดย Gas ซึ่งจะนำไปสู่การหาคำตอบที่ดีที่สุด (Optimal Solution) นั้นคือสมาชิกของประชากรที่ลักษณะดีที่สุด (Fittest Individual)

MAs มีพื้นฐานวิธีการค้นหาคำตอบมาจาก EAs เช่นเดียวกันกับ GAs แต่มีความแตกต่างที่ MAs เป็นการรวมกันของ EAs และการประยุกต์ใช้กระบวนการค้นหาคำตอบแบบเฉพาะที่ (Local Search Procedure) โดยที่ EAs จะใช้ Evolutionary Search ในการสำรวจพื้นที่คำตอบที่เป็นไปได้อย่างกว้างๆ ในขณะที่กระบวนการค้นหาคำตอบแบบเฉพาะที่จะทำการขยายคำตอบที่ดี (Zoom-in) ในพื้นที่คำตอบให้ออกมาเป็นคำตอบที่น่าสนใจและคาดว่าคำตอบนั้นจะเป็นคำตอบที่ดี นอกจากนี้ MAs ยังเป็นวิธีการที่ดีรับความนิยม และได้รับการพิสูจน์ว่าเป็นวิธีการที่สามารถนำไปประยุกต์ใช้ได้อย่างหลากหลายในด้านเทคนิคการหาคำตอบที่เหมาะสมที่สุดเชิงจัด การหาค่าเหมาะสมที่สุดของฟังก์ชั้นไม่คงที่ (Optimization of Non-stationary Functions) การหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ และยังเป็นที่รู้จักกันในชื่อที่เรียกว่า hybrid EA, Ganetic Local Search, Baldwinian EAs, Lamarckian EAs เป็นต้น

ความแตกต่างของ SA TS Gas และ MAs คือ SA นั้นจะมีการค้นหาคำตอบทีละจุด แต่ GAs และ MAs จะเป็นการค้นหาแบบคู่ขนาน คือ ค้นหาคำตอบหลายจุดพร้อมกัน ทำให้โอกาสที่จะได้ค่าคำตอบที่เป็นค่าดีที่สุดเฉพาะที่ (Local Optimal Value) นั้นลดลง ส่วนการหาคำตอบโดย TS นั้นจำเป็นต้องเก็บข้อมูลเก่า (Tabu List) มาเปรียบเทียบในการหาคำตอบทำให้ต้องใช้หน่วยความจำมากขึ้นเรื่อยๆ ในกรณีที่มีการค้นหาคำตอบหลายรอบ แต่ GAs นี้จะใช้หน่วยความจำเท่าเดิมตลอดในการค้นหาแต่ละรุ่น ทั้ง SA และ TS จะใช้การหาคำตอบบริเวณข้างเคียงกับคำตอบเดิมเท่านั้น ทำให้ไม่เหมาะกับการหาคำตอบกับฟังก์ชั่นที่มีพื้นที่การค้นหาขนาดใหญ่

4.4 การแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ด้วยเอลโวลูชั่นนารี่อัลกอริทึมแบบหลายวัตถุประสงค์

เอลโวลูชันนารี่อัลกอริทึมแบบหลายวัตถุประสงค์ Multi-Objective Evolutionary Algorithm: MOEAs) ได้นำมาประยุกต์ใช้ในปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ โดยอัลกอริทึมนี้ได้มีวัตถุประสงค์เพื่อค้นหาขอบเขตกลุ่มคำตอบ ที่ดีที่สุดที่มีลักษณะการกระจายแบบสม่ำเสมอ และใกล้เคียงกับขอบเขตกลุ่มคำตอบที่ดีที่สุดที่แท้จริง จะเป็นเซตคำตอบที่มี high-dimensional ซึ่งถือว่ามีความซับซ้อนมากกว่าปัญหาการหาค่าเหมาะสมที่สุดแบบวัตถุประสงค์เดียว โดยทั่วไปแล้วการประมาณเซตกลุ่มคำตอบที่ดีที่สุดจะเกี่ยวข้องกับสองเป้าหมายในการหาคำตอบในปัญหาการหาค่าเหมาะสมที่สุดแบบหลายวัตถุประสงค์ เป้าหมายแรก คือ การกำหนดค่าความแข็งแรงแบบวิธีเชิงกลุ่มที่ดีที่สุด (Pareto-based Fitness Assignment) ซึ่งจะใช้เป็นแนวทางในการค้นหาขอบเขตของกลุ่มคำตอบมีความสม่ำเสมอตลอดทั้งขอบเขตกลุ่มคำตอบ เป้าหมายที่สองคือ วิธีการประมาณความหนาแน่นของประชากรคำตอบ (Population Diversity) เพื่อใช้รักษาความหลากหลายให้กับคำตอบ ทำให้ลักษณะการกระจายของคำตอบบนขอบเขตกลุ่มคำตอบมีความสม่ำเสมอตลอดทั้งขอบเขตของกลุ่มคำตอบ ไม่เกาะอยู่บริเวฯใดบริเวณหนึ่ง ซึ่งทั้งสองเป้าหมายนี้จะกล่าวในหัวข้อถัดไป

วิธีการแก้ปัยหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ที่เป็นที่ยอมรับในความสามารถในการหาคำตอบ ก็คือ Multi-Objective GAs ซึ่งเป็นวิธีที่อ้างอิงมาจาก MOEAs และได้มีการพัฒนาเป็นอัลกอริทึมต่างๆ ได้แก่

- Vector Evaluated Genetic Algorithm (VEGA) โดย Shaffer (1985)
- Multi-Objective Genetic Algorithm (MOGA) โดย Fonseca และ Fleming (1993)
- Niched-Pareto Genetic Algorithm (NPGA) โดย Horn, Nafpliotis และ Goldberg (1993)
- Non-dominate Sorting Genetic Algorithm (NSGA) พัฒนาโดย Srinivas และ Deb (1995)
- Strength Pareto Evolutionary Algorithm (SPEA) โดย Zitzler และ Thiele (1999)

- Pareto-Archived Evolutionary Strategy (PEAS) โดย Knowles และ Corne (2000)
- Niched-Pareto Genetic Algorithm (NPGA-II) โดย Erickson, Mayer และ Horn (2001)
- Strength Pareto Evolutionary Algorithm 2 (SPEA 2) โดย Zitzler, Laumanns และ Thiele (2001)
- Non-dominated Sorting Ganetic Algorithm II (NSGA-II) โดย Deb, Agrawal, Prawal, Pratab และ Meyarivan (2002)
- Rank Density Genetic Algorithm (RDGA) โดย Lu, Yen และ Member (2003)

4.4.1 การกำหนดค่าความแข็งแรง

การกำหนดค่าความแข็งแรง (Fitness Assignment) ในปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์จะมีความแตกต่างกับการหาค่าเหมาะสมที่สุดที่มีวัตถุประสงค์เดียว โดยสิ่งที่สำคัญสำหรับปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์คือการกำหนดค่าความแข็งแรงให้กับสมาชิกของกลุ่มประชากรแต่ละตัวได้อย่างเหมาะสม และสอดคล้องกับกลุ่มคำตอบที่ดีที่สุด วิธีการกำหนดค่าความแข็งแรงในปัญหาที่มีหลายวัตถุประสงค์ได้แก่ วิธีการคำนวณแบบเวกเตอร์ (Vector Evaluation Approach) โกล์โปรแกรมมิ่ง (Goal Programming Approach) วิธีคอมโพรไมส์ (Compromise Approach) วิธีการรวมฟังก์ชั่นโดยอาศัยการให้น้ำหนัก (Weighted Sum Approach) วิธีเชิงกลุ่มที่ดีที่สุด (Pareto-based Approach) เป็นต้น ซึ่งในงานวิจัยนี้ได้ใช้การกำหนดค่าแข็งแรงด้วยวิธีเชิงกลุ่มที่ดีที่สุดส่วนวิธีอื่นจะขอกล่าวถึงโดยสรุป (Gen และ Cheng, 2000) ดังนี้

4.4.1.1 วิธีการคำนวณค่าแบบเวกเตอร์

วิธีการนี้เป็นวิธีการกำหนดค่าแข็งแรงวิธีแรกที่ขยายผลให้ GAs มีความสามารถในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ จากเดิมที่ใช้ตัววัดความแข็งแรงแบบสเกลลาร์ (Scalar Fitness) ในแต่ละโครโมโซม จะใช้ตัววัดความแข็งแรงแบบเวคเตอร์ (Vector Fitness) เพื่อสร้างคำตอบในเจเนอเรชั่นต่อไป สำหรับปัญหาที่มี k วัตถุประสงค์ ในการเลือกคำตอบในแต่ละเจนเนอเรชั่นจะเป็นการทำซ้ำจำนวน k ครั้ง ซึ่งสัดส่วนของคำตอบในเจนเนเรชั่นต่อไปจะถูกเลือกด้วยพื้นฐานของแต่ละฟังก์ชั่นวัตถุประสงค์

4.4.1.2 โกล์โปรแกรมมิ่ง

เป็นเทคนิคหนึ่ง ที่ใช้ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ได้อย่างมีประสิทธิภาพ โดย Gen และ Liu (1997) นำมาประยุกต์ใช้กับ GAs เพื่อแก้ปัญหาโกล์โปรแกรมมิ่งแบบไม่เชิงเส้น ซึ่งวิธีการกำหนดค่าความแข็งแรงแบบนี้จะอาศัยการจัดอันดับ เพื่อประเมินความสามารถของสมาชิกคำตอบ โดยการให้อันดับจะใช้ค่าของวัตถุประสงค์ จากนั้นจะทำการเรียงลำดับของอันดับในแต่ละสมาชิกคำตอบ โดยการให้อันดับจะใช้ค่าของวัตถุประสงค์ จากนั้นจะทำการเรียงลำดับของอันดับในแต่ละสมาชิกคำตอบ และใช้หลักการให้ความสำคัญมากกว่ากัน โดยสมาชิกคำตอบที่มีความสำคัญมากกว่าจะได้รับอันดับแรก ถ้าสมาชิกคำตอบที่ได้รับการเรียงลำดับแล้วมีค่าวัตถุประสงค์เท่ากันจะให้สมาชิกคำตอบนั้นมีอันดับความสำคัญที่สอง ซึ่งค่าความแข็งแรงของสมาชิกคำตอบนี้จะถูกกำหนดด้วยการประมาณในช่วงจากช่วงที่ดีที่สุดถึงแย่ที่สุดตามฟังก์ชั่นเอ็กโปเนนเชียล

4.4.1.3 วิธีคอมไพรไมส์

Chang และ Gen (1997) ได้แนะนำวิธีการกำหนดค่าความแข็งแรงด้วยวิธีคอมโพรไมส์ เพื่อใช้ในการแก้ปัญหาการหาเส้นทางที่สั้นที่สุด จากการพิจารณา 2 เกณฑ์ โดยที่พื้นฐานแนวคิดของเทคนิคนี้มาจากการลอกเลียนแบบการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์แบบทั่วไป (Conventional Multi-objective Optimizations) ซึ่งวิธีคอมไพล์ไมส์นี้จะทำการระบุคำตอบที่ใกล้เคียงกับคำตอบในอุดมคติ (Ideal Solution) ซึ่งกำหนดด้วยตัววัดระยะทางบางตัววัด สำหรับคำตอบในอุดมคติปกติแล้วจะไม่สามารถค้นพบได้ แต่สำหรับวิธีคอมโพรไมส์นี้จะช่วยให้การคำนวณค่ากลุ่มคำตอบที่พบได้นี้สามารถอ้างอิงไปสู่จุดคำตอบในอุดมคติ

ได้ โดยพิสูจน์ให้เห็นว่าการค้นหาคำตอบให้เข้าใกล้คำตอบในอุดมคตินี้มีเหตุผลที่เป็นไปได้ โดยหลักการของจุดแทนคำตอบในอุดมคติได้ถูกแนะนำให้ใช้แทนหลักการเดิม ซึ่งจุดแทนคำตอบในอุดมคตินี้จะเป็นคำตอบที่สอดคล้องกับคำตอบในเจนเนอเรชั่นปัจจุบัน ที่ได้คำนวณบนพื้นที่การสำรวจพื้นที่คำตอบบางส่วน ไม่ใช่พื้นที่คำตอบทั้งหมด นั้นคือจุดแทนคำตอบในอุดมคตินี้จึงสามารถหาค่าได้ง่ายในแต่ละเจนเนอเรชั่น

4.4.1.4 วิธีการรวมฟังก์ชั่นโดยอาศัยการให้น้ำหนัก

วิธีการรวมฟังก์ชั่นโดยอาศัยการให้น้ำหนัก (Weighted Sum Approach) เป็นวิธีการที่ไม่ยุ่งยากและง่ายต่อการนำไปใช้ วิธีการนี้อาศัยการให้น้ำหนักกับฟังก์ชั่นวัตถุประสงค์แต่ละอย่าง โดยที่น้ำหนักนี้อาจจะได้มาจากการประมาณความสำคัญของวัตถุประสงค์นั้นๆ แล้วนำเอาน้ำหนักและฟังก์ชั่นวัตถุประสงค์ทั้งหมดมารวมกัน (Weighted Sum) เป็นเส้นตรง (Scalar Fitness Function) เพื่อสร้างค่าวัตถุประสงค์ใหม่เพียงค่าเดียว โดยมีน้ำหนักของวัตถุประสงค์เป็นสัมประสิทธิ์ของฟังก์ชั่นวัตถุประสงค์นั้นๆ ดังสมการที่ (4.13) และคำตอบที่ได้จากการรวมวัตถุประสงค์จะมีเพียงคำตอบเดียวที่ให้ค่าเหมาะสมที่สุดระหว่างแต่ละวัตถุประสงค์ วิธีการหาคำตอบที่ใช้วิธีรวมวัตถุประสงค์ ได้แก่ วิธีการรวมฟังก์ชั่นโดยอาศัยการให้น้ำหนัก (Weighted Sum Approach) โดย Jakob et al. วิธีลดฟังก์ชั่นไปสู่ฟังก์ชั่นเดียว (Reduction to Single Objective) โดย Ritzel และ Wayland วิธีการบรรลุเป้าหมาย (Goal Attainment) เป็นต้น

$$f(x) = w_1 f_1(x) + w_2 f_2(x) + \ldots + w_k f_k(x) \qquad (4.13)$$

จากสมการที่ (4.13) ถ้ากำหนดน้ำหนักให้กับแต่ละวัตถุประสงค์เป็นค่าเฉพาะหนึ่งๆ จะเห็นได้ว่าทิศทางการหาคำตอบจะมุ่งสู่จุดใดจุดหนึ่งเพียงจุดเดียวเท่านั้น ดังนั้นวิธีการนี้จะให้คำตอบที่ดีที่สุด (Trade-off) สำหรับการกำหนดค่าน้ำหนักแบบหนึ่งๆ เพียงคำตอบเดียว จึงไม่ต้องพึ่งพาการตัดสินใจของผู้ตัดสินใจ (Decision Maker) อีกครั้งหนึ่ง

ปัญหาประการหนึ่งของการกำหนดน้ำหนักเป็นค่าเฉพาะหนึ่งๆ นั้นก็คือถ้าหากไม่มีข้อมูลเพียงพอก็จะเกิดความยุ่งยากในการกำหนดน้ำหนักของแต่ละวัตถุประสงค์ในกรณีนี้อาจใช้วิธีการสร้างกลุ่มคำตอบที่ดีที่สุด แทนโดยการเปลี่ยนค่าน้ำหนักของวัตถุประสงค์ไปเรื่อยๆเพื่อหาขอบเขตของคำตอบที่ดี แล้วค่อยให้ผู้ตัดสินใจเลือก และสำหรับกรณีที่วัตถุประสงค์

เข้าด้วยกันได้โดยตรง ต้องทำการเปลี่ยนรูปฟังก์ชั่นวัตถุประสงค์ (Normailzed) ทั้งหมดให้สอดคล้องและเป็นหน่วยพื้นฐานเดียวกันเสียก่อน

โดยทั่วไปวิธีการปรับน้ำหนัก จะมีการนำมาเพื่อเพิ่มประสิทธิภาพในการค้นหาแบบพันธุกรรม คือ วิธีการกำหนดแบบตายตัว (Fixed-Weighted) Approach) วิธีการกำหนดน้ำหนักแบบปรับเปลี่ยนได้ (Adaptive Weighted Approach)

วิธีการกำหนดแบบตายตัว เป็นวิธีการกำหนดให้น้ำหนักมีค่าไม่เปลี่ยนแปลงไประหว่างกระบวนการวิวัฒนาการ โดยที่น้ำหนักนี้สามารถกำหนดได้โดยการได้รับความรู้มาก่อน หรือถ้าไม่มีการได้รับความรู้มาก่อน จะใช้การสุ่มน้ำหนัก และวิธีการนี้จะใช้ทิศทางที่แน่นอนในการกำหนดน้ำหนัก

วิธีการกำหนดน้ำหนักแบบสุ่ม เป็นวิธีการสุ่มน้ำหนักในการเลือกกระบวนการที่ทำให้เกิดคำตอบที่เป็นไปได้ โดยวิธีการนี้จะทำให้ได้ขอบเขตกลุ่มคำตอบที่ดีที่สุดที่มีลักษณะการกระจายสม่ำเสมอ แต่ยังเป็นวิธีที่ละเลยกลุ่มคำตอบที่ดีที่สุดที่ได้ในแต่ละเจนเนอเรชั่น

วิธีการกำหนดน้ำหนักแบบปรับเปลี่ยนได้ วิธีการนี้จะมีการปรับให้น้ำหนักในเจนเนอเรชั่นปัจจุบันให้สามารถค้นหาคำตอบในอุดมคติได้ เนื่องจากวิธีการนี้เป็นการใช้น้ำหนักที่ปรับปรุงใหม่มาใช้ในแต่ละเจเนอเรชั่น จึงทำให้ได้คำตอบนี้มีโอกาสนี้มีโอกาสที่จะเป็นคำตอบในอุดมคติหรือคำตอบที่ดีที่สุด

4.4.1.5 **วิธีเชิงกลุ่มที่ดีที่สุด**

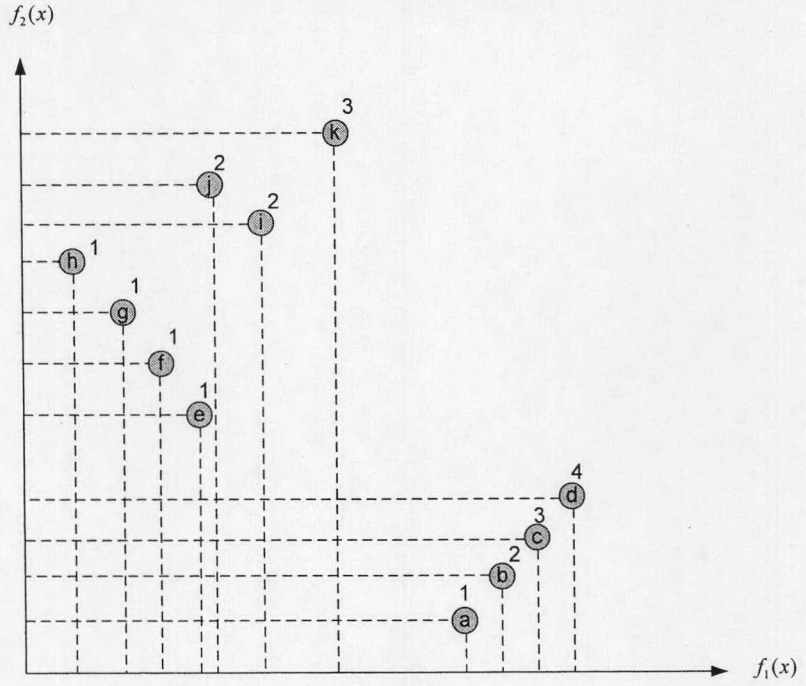
วิธีการนี้จะใช้การจัดอันดับแบบพาเรโต (Pareto Ranking Approach) ในการสร้างความสัมพันธ์ระหว่างค่าฟังก์ชันวัตถุประสงค์และค่าความแข็งแรง โดยใช้หลักการของ Pareto dominance เพื่อคำนวณค่าแข็งแรง หรือใช้ความน่าจะเป็นในการเลือกคำตอบ ซึ่งประชากรจะถูกจัดอันดับตามหลัก Dominance Rule แต่ละคำตอบจะถูกกำหนดค่าความแข็งแรงภายใต้พื้นฐานอันดับคำตอบ ซึ่งไม่ใช่ค่าฟังก์ชันวัตถุประสงค์จริง หรืออาจกล่าวได้ว่าเป็นค่าแข็งแรงไม่แท้จริง (Dummy Fitness Value) นั่นเองเปรียบเสมือนการแยกกลุ่มคำตอบ โดยใช้ค่าความแข็งแรงไม่แท้จริงเป็นตัวกำหนด นอกจากนี้สมาชิกคำตอบที่มีอันดับเดียวกันจะถูกจัดให้อยู่ในกลุ่มคำตอบเดียวกัน ซึ่งถือว่าสมาชิกคำตอบนี้จะมีโอกาสหรือมีความน่าจะเป็นที่เท่าเทียมกันในการที่จะถูกเลือกไปทำให้เกิดคำตอบใหม่

สำหรับวิธีการกำหนดค่าแข็งแรงแบบนี้ จะทำให้มีคำตอบที่ดีที่สุดของปัญหามากกว่าหนึ่งคำตอบ และอยู่ในรูปแบบที่เป็นเซตหรือกลุ่มคำตอบที่ดีที่สุด นั่นคือกลุ่มคำตอบที่ดีที่สุด นั่นคือกลุ่มคำตอบที่ไม่มีคำตอบตัวใดที่ดีกว่ากลุ่มคำตอบนี้ หรือคำตอบที่ไม่ถูกครอบงำจากคำตอบอื่นเลย ตัวอย่างวิธีการกำหนดค่าความแข็งแรงที่ใช้วิธีเชิงกลุ่มที่ดีที่สุด ได้แก่

- วิธีการจัดอันดับของ Goldberg หรือ Non-dominated Sorting
- วิธีการจัดลำดับของ Fonseca และ Fleming
- วิธีการจัดอันดับแบบ Accumulate Ranking Density Strategy (AARS)
- วิธีการจัดอันดับแบบ Strength of Dominators

วิธีการจัดอันดับของ Goldberg

การกำหนดค่าความแข็งแรงของคำตอบ ด้วยการจัดอันดับที่แบบพาเรโต ถูกนำเสนอเป็นครั้งแรกด้วย Goldberg (1989) และเรียกวิธีนี้ว่าวิธีการจัดลำดับของ Goldberg (Goldberg's Ranking) หรือ Non-dominated Sorting และเป็นเทคนิคหนึ่งในการบรรลุเป้าหมายแรกในการแก้ปัญหาการหาค่าที่เหมาะสมที่สุดที่มีหลายวัตถุประสงค์ คือการได้มาของขอบเขตกลุ่มคำตอบที่ดีที่สุด (Pareto Frontier) แนวคิดพื้นฐานของเทคนิคนี้ คือการจัดลำดับเซตของสตริงคำตอบในประชากรคำตอบทั้งหมด โดยจะพิจารณาคำตอบที่ไม่มีคำตอบใดดีกว่าเซตคำตอบนี้เป็นอันดับแรก และจัดอันดับ (Rank) เป็นอันดับที่หนึ่ง จากนั้นจะถูกตัดออกจากการพิจารณาของประชากรคำตอบทั้งหมด เซตของสตริงคำตอบที่เหลือจะถูกจัดให้เป็นอันดับต่อมา โดยที่กระบวนการหาคำตอบที่ดีที่สุดของเทคนิคนี้จะค้นหาคำตอบจนกระทั่งคำตอบในประชากรคำตอบทั้งหมดถูกจัดอันดับ แสดงได้ดังรูปที่ 4.2



รูปที่ 4.2 วิธีการจัดลำดับของ Goldberg

จากรูปที่ 4.2 จะเห็นได้ว่าสมาชิกประชากรคำตอบที่ไม่ถูกครอบงำจากคำตอบอื่น จะถูกจัดให้มีอันดับที่หนึ่ง จากนั้นจะไม่พิจารณาสมาชิกดังกล่าวชั่วคราว เพื่อกำหนดอันดับที่ ให้กับสมาชิกประชากรคำตอบที่เหลืออยู่ โดยถ้าสมาชิกประชากรคำตอบนั้นไม่ถูกครอบงำจากคำตอบอื่น จะถูกจัดอันดับที่ให้เป็นอันดับต่อมา และพิจารณาไปเรื่อย ๆ จนครบสมาชิกคำตอบทุกคำตอบ โดยเริ่มต้นจากสมาชิกตัวที่แข็งแรงที่สุด จนถึงสมาชิกตัวที่มีความอ่อนแอที่สุด สำหรับการจัดอันดับของ Goldberg นี้จะเป็นการกำหนดค่าแข็งแรงที่ได้นำไปประยุกต์ใช้กับเจนเนติกอัลกอริทึมที่มีชื่อว่า Non-dominated Sorting Genetic Algorithm II (NSGAII)

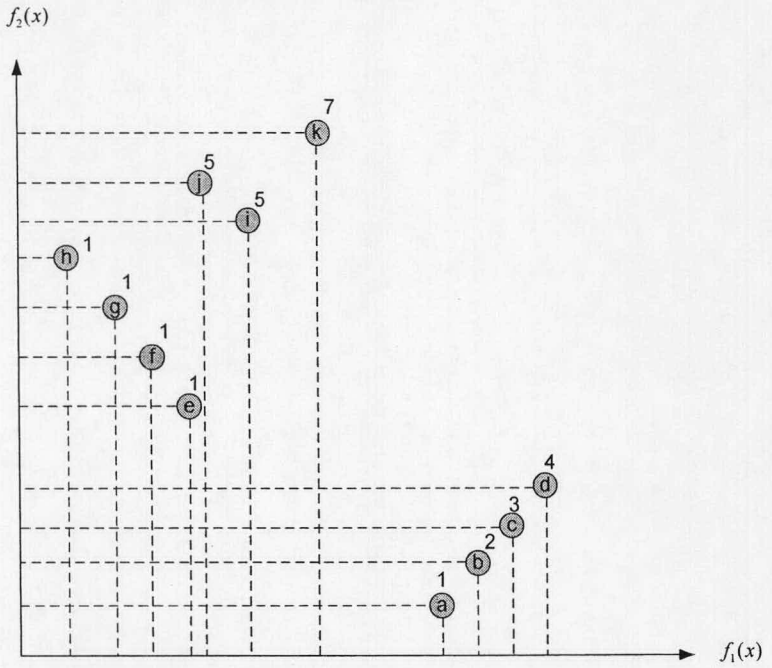
วิธีการจัดลำดับของ Fonseca และ Fleming

Fonseca และ Fleming (1993) ได้เสนอความผันแปรเกี่ยวกับเทคนิคการจัดอันดับของ Goldberg โดยที่การจัดอันดับแบบ Fonseca และ Fleming นี้เป็นเทคนิคที่ใช้การจัดอันดับสมาชิกประชากรคำตอบปัจจุบันเช่นเดียวกันกับวิธีของ Goldberg แต่มีความแตกต่างในการพิจารณาจำนวนคำตอบที่ถูกครอบงำ (Dominated Solution) ซึ่งได้จากเปรียบเทียบคำตอบในแต่ละสมาชิกคำตอบ คำตอบที่ไม่ถูกครอบงำจากคำตอบอื่น (Non-Dominated Solution) จะถูกจัดอันดับเป็นอันดับแรก จากนั้นจะพิจารณาเปรียบเทียบคำตอบที่

เหลือ ถ้าคำตอบนั้นถูกครอบงำจากคำตอบในอันดับแรกเพียงหนึ่งคำตอบ จะได้อันดับของคำตอบเป็นอันดับที่สอง ถ้าคำตอบนั้นถูกครอบงำจากคำตอบอันดับแรกจำนวนสองคำตอบ จะได้อันดับของคำตอบนั้นเป็นอันดับที่สาม นั่นคือสมาชิกคำตอบ x_i ในเจนเนอเรชั่นที่ t จะถูกการครอบงำด้วยจำนวนคำตอบในเจนเนอเรชั่นปัจจุบัน $p_i^{(t)}$ กำหนดได้จากสมการที่ (4.14)

$$rank(x,t) = 1 + p_i^{(t)} \tag{4.14}$$

จากสมการที่ (4.14) จะเห็นได้ว่าสมาชิกคำตอบของกลุ่มประชากรที่มีอันดับสูงที่สุดจะมีค่าความแข็งแรงอันดับน้อยที่สุด และสมาชิกคำตอบของกลุ่มประชากรที่มีอันดับต่ำที่สุดจะมีค่าความแข็งแรงอันดับที่สูงที่สุด อธิบายดังรูปที่ 4.3



รูปที่ 4.3 วิธีการจัดลำดับของ Fonseca และ Fleming (1993)

จากรูปที่ 4.3 จะเห็นได้ว่าค่าอันดับที่ของสมาชิกที่สนใจเท่ากับการนำค่า 1 บวกกับจำนวนสมาชิกในประชากรคำตอบที่สามารถครอบงำคำตอบนั้นได้ โดยค่าอันดับที่ของสมาชิกประชากรคำตอบสามารถคำนวนได้ดังนี้

สมาชิกประชากรคำตอบ a จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ b จะมีอันดับที่ = 1+(1) = 2

สมาชิกประชากรคำตอบ c จะมีอันดับที่ = 1+(2) = 3

สมาชิกประชากรคำตอบ d จะมีอันดับที่ = 1+(3) = 4

สมาชิกประชากรคำตอบ e จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ f จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ g จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ h จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ i จะมีอันดับที่ = 1+(4) = 5

สมาชิกประชากรคำตอบ j จะมีอันดับที่ = 1+(4) = 5

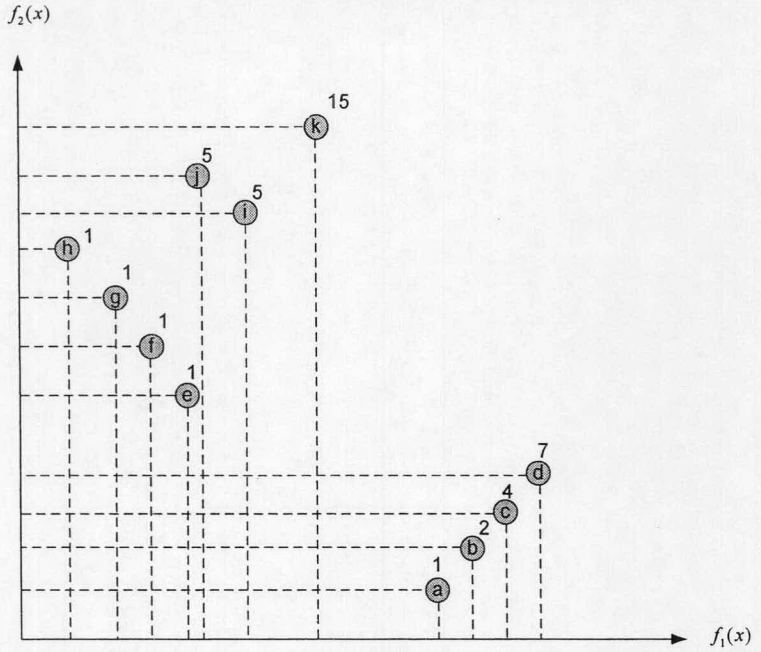
สมาชิกประชากรคำตอบ k จะมีอันดับที่ = 1+(6) = 7

ผลลัพธ์ของการกำหนดอันดับที่ให้กับสมาชิกคำตอบในกลุ่มประชากรได้ครบทุกตัวแล้ว คือค่าความแข็งแรงไม่แท้จริงของคำตอบ โดยสมาชิกตัวที่แข็งแรงที่สุดจะมีอันดับที่น้อยที่สุด และสมาชิกตัวที่มีค่าความอ่อนแอที่สุด จะมีอันดับทีสูงที่สุด สำหรับการจัดอันดับของ Fonseca และ Fleming (1993) นี้จะเป็นการกำหนดค่าความแข็งแรงที่ได้นำไปประยุกต์ใช้กับเจนเนติกอัลกอริทึมที่มีชื่อว่าเจนเนติกอัลกอริทึมที่มีหลายวัตถุประสงค์ (Multi-Objective Genetic Algorithm: MOGA)

วิธีการจัดอันดับแบบ Accumulate Ranking Density Strategy

วิธีการจัดอันดับแบบ Accumulate Ranking Density Strategy (AARS) เป็นวิธีการจัดอันดับที่มีพื้นฐานแนวคิดมาจากวิธีการจัดลำดับของ Fonseca และ Fleming ดังนั้นอันดับที่ของสมาชิกคำตอบที่กำหนดให้แต่ละสมาชิกประชากรคำตอบ จะเท่ากับ 1 บวกกับผลรวมอันดับที่ได้จากการจัดลำดับแบบ Fonseca และ Fleming (1993) นั่นคือพิจารณาสามชิกคำตอบ y_i ในเจนเนอเรชั่นที่ t จะถูกการครอบงำด้วยอันดับที่ของสมาชิกคำตอบที่ได้จากการจัดลำดับแบบ Fonseca และ Fleming ในเจนเนอเรชั่นปัจจุบัน $p_i^{(t)}$ โดยการจัดลำดับของสมาชิกคำตอบสามารถกำหนดได้จากสมการที่ (4.15)

$$rank(y,t) = 1 + \sum_{j=1}^{p_i^{(t)}} rank(y,t) \qquad (4.15)$$



รูปที่ 4.4 วิธีการจัดอันดับแบบ Automatic Accumulated Ranking Strategy

จากรูปที่ 4.4 จะเห็นได้ว่าค่าอันดับที่ของสมาชิกคำตอบที่พิจารณาเท่ากับผลรวมของอันดับที่ของสมาชิกคำตอบที่สามารถครอบงำคำตอบนั้นได้ และคำตอบใดที่ไม่ถูกครอบงำจากคำตอบอื่นเลย จะถูกจัดให้อยู่ในอันดับแรกเช่นเดียวกับวิธีการจัดอันดับแบบ Goldberg และวิธีการจัดลำดับแบบ Fonseca และ Fleming โดยค่าอันดับที่ของสมาชิก ประชากรคำตอบสามารถคำนวณได้ดังนี้

สมาชิกประชากรคำตอบ a จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ b จะมีอันดับที่ = 1+(1) = 2

สมาชิกประชากรคำตอบ c จะมีอันดับที่ = 1+(1+2) = 4

สมาชิกประชากรคำตอบ d จะมีอันดับที่ = 1+(1+2+3) = 7

สมาชิกประชากรคำตอบ e จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ f จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ g จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ h จะมีอันดับที่ = 1

สมาชิกประชากรคำตอบ i จะมีอันดับที่ = 1+(1+1+1+1) = 5

สมาชิกประชากรคำตอบ j จะมีอันดับที่ = 1+(1+1+1+1) = 5

สมาชิกประชากรคำตอบ k จะมีอันดับที่ = 1+(1+1+1+1+5+5) = 7

ผลลัพธ์ของการกำหนดอันดับที่ให้กับสมาชิกคำตอบในกลุ่มประชากรได้ครบทุกตัวแล้ว คือค่าความแข็งแรงไม่แท้จริงของคำตอบ โดยสมาชิกตัวที่แข็งแรงที่สุดจะมีอันดับที่น้อยที่สุด และสมาชิกตัวที่มีความอ่อนแอที่สุด จะมีอันดับที่สูงที่สุด สำหรับวิธีการจัดอันดับแบบ Automatic Accumulated Ranking Strategy นี้จะเป็นการกำหนดค่าความแข็งแรงที่ได้นำไปประยุกต์ใช้กับเจนเนติกอัลกอริทึมที่มีชื่อว่า Rank Density Genetic Algorithm (RDGA)

วิธีการจัดอันดับแบบ Strength of Dominators

การจัดอันดับแบบ Strength of Dominators เป็นวิธีการกำหนดค่าความแข็งแรงของสมาชิกคำตอบที่แตกต่างจากวิธีการจัดอันดับของ Goldberg วิธีการจัดอันดับของ Fonseca และ Fleming และวิธีการจัดลำดับของ Automatic Accumulated Ranking Strategy เนื่องจากในวิธีนี้จะกำหนดค่าความแข็งแรงให้กับสมาชิกคำตอบด้วยการพิจารณาจำนวนสมาชิกในประชากรคำตอบที่ดีกว่าคำตอบที่กำลังพิจารณาอยู่ (Raw Fitness Value: $R(i)$) รวมกับการพิจารณาความหนาแน่นในบริเวณใกล้เคียงกับคำตอบนั้น(Density Information: $D(i)$) ไปพร้อมกัย วิธีการจัดอันดับแบบ Strength of Dominators ยังพิจารณาจำนวนคำตอบที่แย่กว่าคำตอบที่พิจารณาอยู่ เรียกว่าค่า Strength (Strength Value: $S(i)$) โดยผลรวมของค่า $S(i)$ นี้จะทำให้สามารถหาค่า $R(i)$ ได้ สมาชิกคำตอบที่มีค่า $R(i)=0$ แสดงว่าคำตอบนั้นเป็นคำตอบที่ไม่ถูกครอบงำจากคำตอบอื่นเลย และสมาชิกคำตอบนี้จะถูกเก็บไว้สถานที่เก็บคำตอบที่ดี(Archive of Non-dominated Solution) (Zitzler, Laumanns และ Thiele, 2002) ดังนั้นวิธีนี้จะมีค่าความแข็งแรงตามสมการที่ (4.16) ดังนี้

$$F(i) = R(i) + D(i) \tag{4.16}$$

กำหนดให้ $S(i) = \left|\left\{ j \,\middle|\, j \in P_t + E_t \wedge i \prec j \right\}\right|$ (4.17)

โดยที่ P_t แทนประชากรคำตอบ

E_t แทนประชากรคำตอบที่เป็น Non-dominated Solution

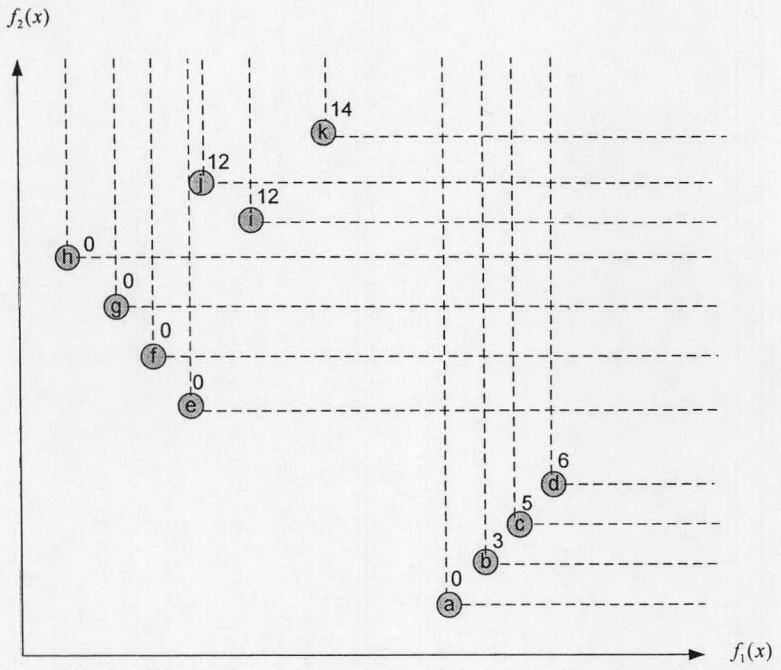
$|\,|$ แทนจำนวนเซตคำตอบ

$+$ แทนการรวมกันของหลายเซตคำตอบ

$\succ$ แทน Pareto Dominance Relation

และสามารถคำนวณค่า $R(i)$ ได้จาก

$$R(i) = \sum_{j \in P_t + \overline{P}_t,\, j \succ i} S(j) \tag{4.18}$$



รูปที่ 4.5 วิธีการจัดอันดับแบบ Strength of Dominators

จากรูปที่ 4.5 แสดงการคำนวณค่า $R(i)$ จะเห็นได้ว่า ในกลุ่มประชากรหนึ่งจำเป็นต้องมีการพิจารณาสมาชิกคำตอบด้วยการพิจารณาจำนวนสมาชิกในประชากรคำตอบที่ดีกว่าคำตอบที่พิจารณาอยู่ และจำนวคำตอบที่แย่กว่าคำตอบที่พิจารณาอยู่โดยในแต่ละสมาชิกคำตอบสามารถคำนวณค่าดังกล่าวได้ดังนี้

สมาชิกประชากรคำตอบ a จะมีอันดับที่ = $S(j)=3, R(i)=0$

สมาชิกประชากรคำตอบ b จะมีอันดับที่ = $S(j)=2, R(i)=3$

สมาชิกประชากรคำตอบ c จะมีอันดับที่ = $S(j)=1, R(i)=3+2=5$

สมาชิกประชากรคำตอบ d จะมีอันดับที่ = $S(j)=0, R(i)=3+2+1=6$

สมาชิกประชากรคำตอบ e จะมีอันดับที่ = $S(j)=3, R(i)=0$

สมาชิกประชากรคำตอบ f จะมีอันดับที่ = $S(j)=3, R(i)=0$

สมาชิกประชากรคำตอบ g จะมีอันดับที่ = $S(j)=3, R(i)=0$

สมาชิกประชากรคำตอบ h จะมีอันดับที่ = $S(j)=3, R(i)=0$

สมาชิกประชากรคำตอบ i จะมีอันดับที่ = $S(j)=1, R(i)=3+3+3+3+3=12$

สมาชิกประชากรคำตอบ j จะมีอันดับที่ = $S(j)=1, R(i)=3+3+3+3+3=12$

สมาชิกประชากรคำตอบ k จะมีอันดับที่ = $S(j)=1, R(i)=3+3+3+3+1+1=14$

ซึ่งในตัวอย่างการคำนวณนี้ไม่ได้แสดงถึงค่าความแข็งแรงของประชากรคำตอบ $F(i)$ เนื่องจากยังไม่ได้มีการคำนวณความหนาแน่น $D(i)$ และจะกล่าวในหัวข้อถัดไป

ผลลัพธ์ของการกำหนดค่า Raw Fitness Value จะมีลักษณะคล้ายคลึงกับรูปแบบการจัดอันดับค่าความแข็งแรงวิธีอื่น ๆ คือ จะได้กลุ่มสมาชิกคำตอบที่มีอันดับที่น้อยที่สุด นั่นคือสมาชิกที่มีความแข็งแรงมากที่สุด นั่นคือ Raw Fitness Value มีค่าเท่ากับศูนย์ จะนำคำตอบเหล่านี้ไปเก็บไว้ในสถานที่เก็บคำตอบที่ดี

สำหรับวิธีการจัดลำดับแบบ Strength of Dominators นี้จะเป็นการกำหนดค่าความแข็งแรงที่ได้นำไปประยุกต์ใช้กับเจนเนติกอัลกอริทึมที่มีชื่อว่า Strength Pareto Evolutionary Algorithm 2 (SPEA2)

4.4.2 การแบ่งปันค่าความแข็งแรง

ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์นั้น มีเป้าหมายเพื่อให้ได้ขอบเขตกลุ่มคำตอบที่ดีที่สุด และมีลักษณะรูปแบบการกระจายของเซตคำตอบบนขอบเขตกลุ่มคำตอบที่ดีที่สุดแบบสม่ำเสมอ ซึ่งวิธีเชิงกลุ่มที่ดีที่สุดสามารถทำให้บรรลุเป้าหมายในข้อแรก ส่วนเป้าหมายข้อที่สองนี้เป็นวิธีที่ใช้ในการสร้างความหลากหลายให้กับคำตอบ เนื่องจากสมาชิกคำตอบที่อยู่บนขอบเขตคำตอบเดียวกันมีการเกาะกลุ่มอยู่บริเวณใดบริเวณหนึ่ง ส่งผลให้บริเวณอื่น ๆ ไม่มีสมาชิกคำตอบอยู่เลย (ไม่พบคำตอบอื่น ๆ เลย) ปรากฏการณ์เช่นนี้ เรียกว่าการลอยเลื่อนเชิงพันธุกรรม (Genetic Drift) ซึ่งมักเกิดขึ้นกับปัญหาที่มีคำตอบหลายคำตอบ (Multi-model Problem) หรือปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ดังนั้นจึงต้องอาศัยเทคนิคที่เรียกว่า วิธีการสร้างความหลากหลายให้กับประชากรคำตอบ (Diversity Population) ซึ่งจัดว่าเป็นการแบ่งปันค่าความแข็งแรง (Fitness Sharing) ในการลดทอนค่าความแข็งแรงของสมาชิกคำตอบที่เกาะอยู่เป็นกลุ่ม ซึ่งขึ้นอยู่กับจำนวนสมาชิกที่เกาะบริเวณคำตอบนั้น ซึ่งมีเทคนิคต่าง ๆ ดังนี้

Niched Fitness Sharing Technique

แนวคิดของ Fitness Sharing ได้ถูกนำเสนอเป็นครั้งแรกด้วย Goldberg และ Richardson(1987) เนื่องจากการคงไว้ของความหลากหลายของประชากร คือการแบ่งสมาชิกในประชากรให้มีการแบ่งปันค่าความแข็งแรงไม่แท้จริงร่วมกัน และการแบ่งปันนี้จะทำให้ค่าความแข็งแรงลดลง ผลที่ได้จากการแบ่งปันความแข็งแรง จะทำให้กลุ่มคำตอบที่ได้มีการ

กระจายแบบสม่ำเสมอมากขึ้น สำหรับ Niching เป็นวิธีหนึ่งที่ใช้ในการแบ่งปันค่าความแข็งแรง และนำมาประยุกต์ใช้ใน MOGA โดยจะทำการคำนวณขนาดของ Niche เพื่อใช้เป็นสัดส่วนในการแบ่งปันค่าความแข็งแรงของสมาชิกคำตอบที่อยู่ใน Niche เดียวกัน ขั้นตอนการคำนวณการแบ่งปันความแข็งแรงแสดงได้ดังนี้

ขั้นตอนที่ 1 คำนวณค่า Euclidean Distance ระหว่างประชากรคำตอบ x และ y ที่มีประชากรคำตอบเป็น Normalized Objective ดังนั้นค่าของประชากรคำตอบจะมีค่าอยู่ระหว่าง 0 และ 1

$$df(x,y) = \sqrt{\sum_{k=1}^{K}\left(\frac{f_k(x) - f_k(y)}{f_k^{\max} - f_k^{\min}}\right)^2} \tag{4.19}$$

โดยที่ $df(x,y)$ คือระยะทางจากจุด x และ y ที่ได้รับการ Normalized

$f_k^{\max}$ และ $f_k^{\min}$ คือ ค่ามากที่สุดและน้อยที่สุดของค่าฟังก์ชันวัตถุประสงค์ $f_k(\cdot)$ ตามลำดับ

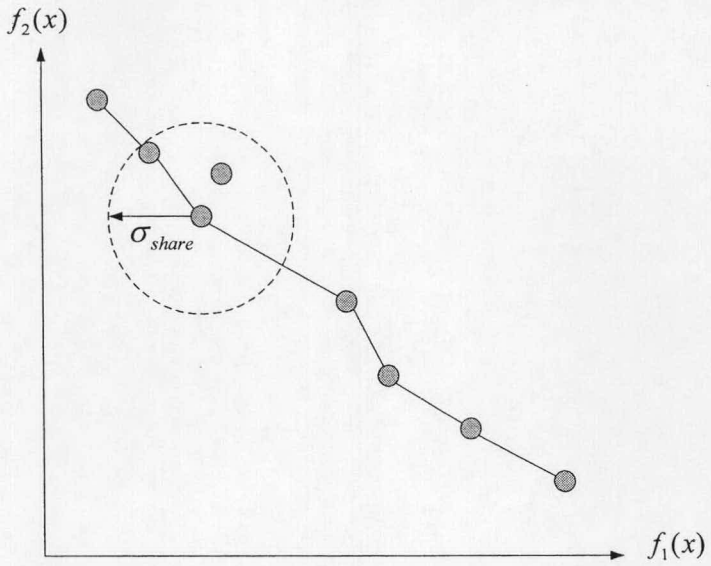
ขั้นตอนที่ 2 เมื่อได้ระยะทางระหว่างคำตอบที่ได้รับการ Normalized แล้วจะทำการคำนวณ Niche Count ซึ่งถือว่าเป็นการคำนวณขนาดของ Niche เพื่อใช้เป็นขอบเขตพื้นที่ในการระบุจำนวนสมาชิกคำตอบที่อยู่ใน Niche เดียวกัน

$$nc(x,t) = \sum_{y \in P, r(y,t)=r(x,t)} \max\left\{\frac{\sigma_{share} - df(x,y)}{\sigma_{share}}, 0\right\} \tag{4.20}$$

โดยที่ σ_{share} คือขนาดของ Niche

ขั้นตอนที่ 3 หลังจากได้ Niche Count แล้ว จะทำให้สามารถคำนวณค่าความแข็งแรงที่ถูกปรับตามตำแหน่งที่อยู่ใน Niche ดังนี้

$$f'(x,t) = \frac{f(x,t)}{nc(x,t)} \tag{4.21}$$



รูปที่ 4.6 Niched Fitness Sharing Technique

Crowding Distance

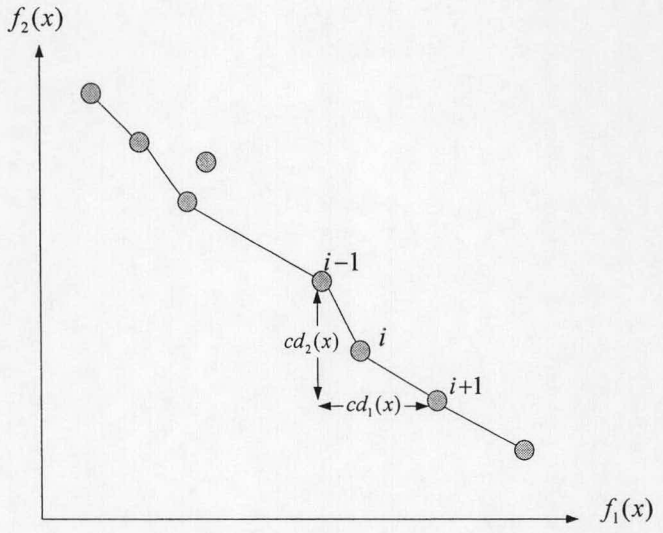
Crowding Distance เป็นเทคนิกหนึ่งที่มีความสามารถในการทำให้ประชากรคำตอบบนขอบเขตกลุ่มคำตอบที่ดีมีลักษณะการกระจายอย่างสม่ำเสมอมากยิ่งขึ้นและนำมาใช้ในการบรรลุเป้าหมายที่สองดังกล่าวของ NSGA II โดยเทคนิคนี้จะถูกนำมาใช้คำนวณระยะทางระหว่างสมาชิกประชากรคำตอบที่อยู่ใน Front เดียวกันเท่านั้น ขั้นตอนการคำนวณ Crowding Distance แสดงได้ดังนี้

ขั้นตอนที่ 1 ถ้าสมาชิกประชากรคำตอบมีอันดับที่เท่ากันแล้วให้คำนวณขั้นตอนที่ 2 และ 3 ดังนี้

ขั้นตอนที่ 2 กำหนดให้ l แทนจำนวนประชากรคำตอบทั้งหมดใน Front ที่ $j, j = 1,..., R$ และ $x_{[i,k]}$ แทน สมาชิกประชากรคำตอบที่ i ในฟังก์ชันวัตถุประสงค์ k ที่ได้รับการเรียงลำดับฟังก์ชันวัตถุประสงค์จากน้อยไปมาก (Sort List) โดยสมาชิกประชากรคำตอบที่มีลำดับที่ 1 (ค่าฟังก์ชันวัตถุประสงค์น้อยที่สุด) และลำดับสุดท้าย (ค่าฟังก์ชันวัตถุประสงค์มากที่สุด) จะถูกกำหนดให้มี Crowding Distance เป็นค่ามาก ๆ (Infinity) นั่นคือ $cd_k(x_{[l,k]}) = \infty$ และ $cd_k(x_{[l,k]}) = \infty$ ส่วนสมาชิกประชากรคำตอบรายการเรียงลำดับที่ 2 ถึง ลำดับที่ $l-1$ จะคำนวณ Crowding Distance จาก

$$cd_k(x_{[i,k]}) = \frac{f_k(x_{[i+1,k]}) - f_k(x_{[i-1,k]})}{f_k^{\max} - f_k^{\min}} \tag{4.22}$$

ขั้นตอนที่ 3 คำนวณผลรวมของ Crowding Distance ทั้ง k ฟังก์ชันวัตถุประสงค์ จะได้ว่า $cd(x) = \sum cd_k(x)$ นั่นคือค่า Crowding Distance ของสมาชิกคำตอบนั้น ๆ โดยค่านี้จะแสดงถึงระยะห่างระหว่างจุดที่อยู่ต่อเนื่องบนคำตอบใน Front เดียวกัน ค่า Crowding Distance น้อยจะแสดงให้เห็นถึงกลุ่มคำตอบใน Front นั้นมีการเกาะกลุ่มกัน ส่วนค่า Crowding Distance มากจะแสดงให้เห็นว่ากลุ่มคำตอบใน Front นั้นมีการกระจายอย่างชัดเจน



รูปที่ 4.7 Crowding Distance

Adaptive Density Estimation

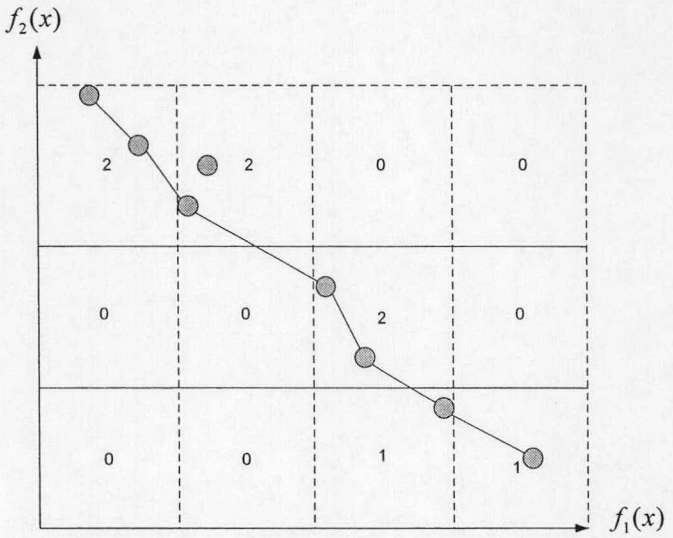
Adaptive Density Estimation เป็นเทคนิคหนึ่งที่ถูกนำมาใช้เพื่อแสดงให้เห็นความแตกต่างของสมาชิกคำตอบที่มีค่าความแข็งแรงเท่ากัน และทำให้ประชากรคำตอบบนขอบเขตกลุ่มคำตอบที่ดีมีลักษณะการกระจายอย่างสม่ำเสมอมากยิ่งขึ้น โดยเทคนิคนี้จะทำการแบ่งพื้นที่วัตถุประสงค์ (Objective Space) แบ่งออกเป็นเซลล์ (Cell) จำนวนสมาชิกประชากรคำตอบที่อยู่ในเซลล์เดียวกัน จะเป็นสิ่งที่กำหนดความหนาแน่นของเซลล์ โดยข้อมูลความหน่าแน่น (Density Information) นี้ที่นำมาใช้ในการเพิ่มความหลากหลายให้กับประชากรคำตอบ ขั้นตอนในการคำนวณของ Adaptive Density Estimation แสดงได้ดังนี้

ขั้นตอนที่ 1 สร้างเซลล์เพื่อใช้เป็นขอบเขตในการพิจารณาความหนาแน่นด้วยการแบ่งพื้นที่วัตถุประสงค์ด้วยความกว้างของเซลล์ที่คำนวณได้จากสมการที่ (4.23) ดังนี้

$$d_i = \frac{\max f_i(x) - \min f_i(x)}{K_i}, i = 1, 2, ..., k \qquad (4.23)$$

และ K แทนจำนวนฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 2 ในแต่ละสมาชิกในประชากรคำตอบเบื้องต้นจะทำการค้นหาจุดใกล้เคียงกับจุดกึ่งกลางของเซลล์ และกำหนดให้เป็นที่อยู่ของสมาชิกคำตอบนั้น (Home Address) และพิจารณาจากสมาชิกประชากรคำตอบอื่นที่อยู่ในที่อยู่เดียวกัน เรียกว่า เป็นสมาชิกในครอบครัว (Family Member) และจะถูกนับพร้อมบันทึกจำนวนคำตอบที่ตกอยู่ในขอบเขตของเซลล์เดียวกันว่าเป็นความหนาแน่นภายในเซลล์นั้น ๆ โดยมีการเก็บค่าเป็นลักษณะเมทริกซ์ พิจาณาได้ดังนี้ 4.8



รูปที่ 4.8 Adaptive Density Estimation

k-nearest neighbor

เทคนิคการประมาณค่าความหนาแน่นของคำตอบที่นำมาใช้ใน SPEA2 คือวิธีการ k-nearest neighbor วิธีการนี้จะคำนวณความหนาแน่นด้วยการหาระยะทางที่ใกล้เคียงกับสมาชิกคำตอบที่พิจารณาอยู่ k คำตอบ แทนด้วย σ_i^k และสามารถกำหนด k ได้จากรากที่สองของผลรวมจำนวนประชากรคำตอบและจำนวนของพื้นที่เก็บคำตอบที่ดี (Archive Size) $k = \sqrt{N_p + N_E}$ ดังนั้นจะได้ว่าความหนาแน่นของสมาชิกประชากรคำตอบที่ i จากสมการที่ (4.24) ดังนี้

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{4.24}$$

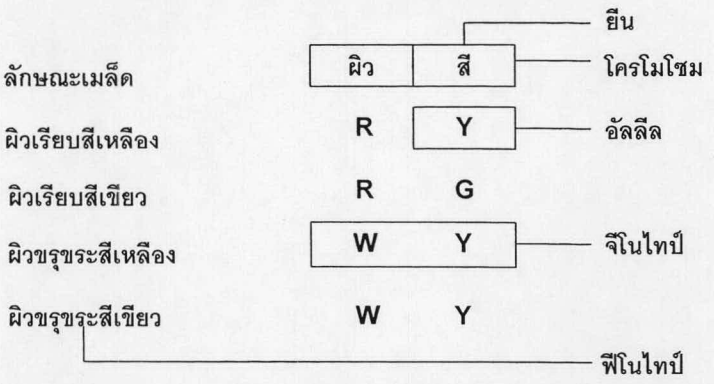
โดยที่การบวกค่า 2 เข้าไปนั้นดพื่อทำให้แน่ใจได้ว่าค่า $D(i)$ นี้จะไม่มากกว่าศูนย์และน้อยกว่า 0.5

4.5 เจนเนติกอัลกอริทึม (Genetic Algorithms: GAs)

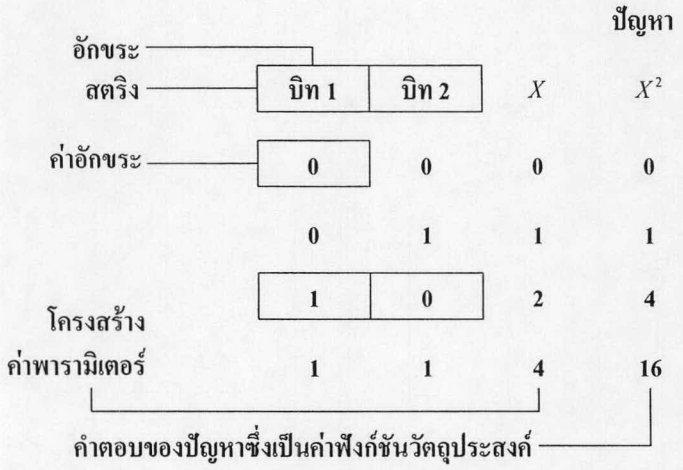
ในปัจจุบันการหาคำตอบของปัญหาบางประเภท เช่น ปัญหาการจัดสรรทรัพยากรที่มีอยู่อย่างจำกัดเพื่อให้เกิดประสิทธิภาพสูงสุดและปัญหาในการคำนวณต้นทุนต่ำสุด เป็นต้น สามารถหาคำตอบได้หลายวิธี วิธีการที่ง่ายที่สุดในการหาคำตอบคือวิธีการทางฮิวริสติกต่างๆ ซึ่งอาจได้คำตอบที่ไม่ดีนัก ในปัจจุบันนักวิทยาศาสตร์ได้นำความรู้เกี่ยวกับทฤษฎีหรือกฎเกณฑ์ทางธรรมชาติมาช่วยในการหาคำตอบหรือศึกษาวิจัย โดยมีเป้าหมายหลักในการใช้ประโยชน์ของความคงทน(Robustness) ต่อความไม่เที่ยงตรงแม่นยำ (Accuracy) ความไม่แน่นอน (Uncertainty) หรือความคลุมเครือของปัญหา (Vague) หลักการเหล่านี้สามารถพบได้จากวิธีการต่างๆ เช่น ระบบโครงข่ายประสาทเทียม (Neural Network) ฟัซซี่ลอจิก (Fuzzy Logic) และ GAs (Goldberg, 1989) ปัญหาที่พบส่วนใหญ่เป็นปัญหาที่ไม่เที่ยงตรงและคลุมเครือ ซึ่งหากต้องการคำตอบที่เที่ยงตรงและมีความแน่นอนสูงมากก็ย่อมมีค่าใช้จ่ายที่สูงมาก ดังนั้นวิธีการที่สามารถแก้ปัญหาที่คลุมเครือโดยได้คำตอบที่ใกล้เคียงสามารถยอมรับได้ ใช้เวลาในการหาคำตอบไม่มากนัก และมีค่าใช้จ่ายพอประมาณ ย่อมดีกว่าวิธีที่ได้ความเที่ยงตรงสูงแต่มีค่าใช้จ่ายที่สูง วิธีการหาคำตอบที่ดีอย่างหนึ่งได้แก่วิธีการของ GAs โดยอาศัยทฤษฎีในการถ่ายทอดลักษณะต่างๆทางกรรมพันธ์ไปสู่ยังลูกหลานซึ่งสามารถนำมาพัฒนาใช้ในการหาคำตอบที่ต้องการได้

4.5.1 พันธุศาสตร์กับเจนเนติกอัลกอริทึม

Mendel บิดาแห่งวิชาพันธุศาสตร์ ค้นพบว่าลักษณะต่างๆของสิ่งมีชีวิต เช่น ลักษณะผิวของเมล็ดพืช สีของเมล็ดพืช ฯลฯ ที่ถูกถ่ายทอดไปยังลูกหลานนั้นถูกควบคุม โดยหน่วยควบคุมลักษณะที่เรียกว่ายีน (Gene) และลักษณะย่อยของยีนเรียกว่าอัลลีล (Allele) เช่นยีนควบคุมลักษณะผิวของเมล็ดจะมีอัลลีลเป็นผิวเรียบและผิวขรุขระ เป็นต้น ซึ่งแต่ละยีนจะเรียงตัวอยู่บนโครโมโซม (Chromosome) ภายในเซลล์ ตำแหน่งของยีนแต่ละยีนบนโครโมโซมเรียกว่าโลกัส (Logus) และแต่ละแบบของชุดยีนเรียกว่า จีโนไทป์(Genotype) ซึ่งแสดงลักษณะภายนอกที่ปรากฏ ซึ่งเรียกว่า ฟีโนไทป์ (Phenotype) ดังรูปที่ 4.9 ก



ก) ลักษณะทางพันธุศาสตร์ของโครโมโซมควบคุมลักษณะของเมล็ดถั่ว ซึ่งมียีนลักษณะของเมล็ดคือ มีลักษณะเรียบ (R) ขรุขระ (W) และยีนลักษณะสีของเมล็ดคือสีเหลือง (Y) และสีเขียว (G)



ข) ลักษณะทางเจนเนติก แสดงถึงการแก้ปัญหาในการหาค่าสูงสุดของฟังก์ชัน $f(x)=x^2$ โดยที่ x มีค่าอยู่ระหว่าง $[0,4]$ และค่าของ x ถูกแปลงให้อยู่ในรูปไบนารี่สตริง

รูปที่ 4.9 เปรียบเทียบลักษณะระหว่างเจนเนติกอัลกอริทึมกับลักษณะทางพันธุศาสตร์

ตารางที่ 4.1 เปรียบเทียบคำศัพท์ระหว่างพันธุศาสตร์และเจนนาติกอัลกอริทึม

พันธุศาสตร์	เจนเนติกอัลกอริทึม
โครโมโซม (Chromosome)	สตริง (String)
ยีน (Gene)	คุณลักษณะ, บิท (Character, bit)
อัลลีล (Allele)	ค่าของคุณลักษณะ, ค่าบิท(Character value, bit value)
โลกัส (Locus)	ตำแหน่ง (String position)
จีโนไทป์ (Genotype)	โครงสร้าง (Structure)
ฟีโนไทป์ (Phenotype)	โครงสร้างคำตอบ (A decode structure)

การแก้ปัญหาทางด้านคณิตศาสตร์ด้วย GAs พารามิเตอร์ต่างๆจะถูกแปลงให้อยู่ในรูปของสตริง (String) หรือโครโมโซมประกอบด้วยอักขระ (Character) หรือ (Bit) แต่ละตำแหน่งของโครโมโซมจะเก็บค่าอักขระหรือค่าของบิทที่แสดงโครงสร้างของแต่ละโครโมโซมที่ให้คำตอบของปัญหาแตกต่างกัน ดังรูปที่ 4.9 ข ซึ่งเป็นการประยุกต์ใช้ GAs กับการแก้ปัญหาการหาค่าสูงสุดของ $f(x)=x^2$ โดยที่ x อยู่ในช่วง [0, 4] และสามารถสรุปความหมายทางพันธุศาสตร์เทียบกับ GAs ได้ดังตารางที่ 4.1

Darwin (1859) ได้เสนอความคิดการเกิดสปีชีส์ของสิ่งมีชีวิต (The Origin of Species) โดยเสนอหลักการของวิวัฒนาการที่ผ่านกระบวนการคัดเลือกตามธรรมชาติ แม้ในตอนแรกทฤษฎีจะเป็นที่โต้แย้งกันมากต่อมาก็ได้เป็นที่ยอมรับในหมู่นักวิทยาศาสตร์ (Winston, 1992)

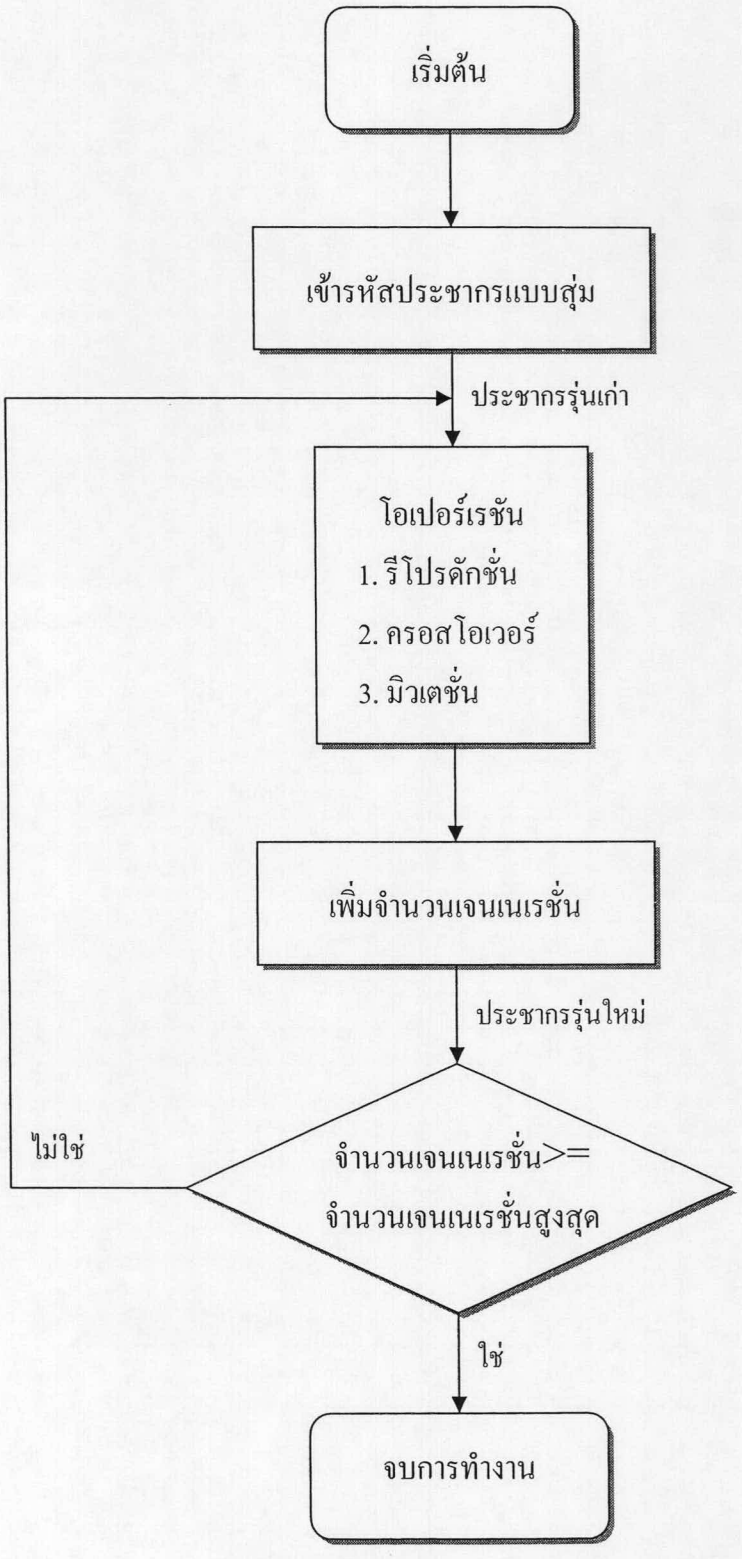
- สิ่งมีชีวิตแต่ละชนิดมีแนวโน้มที่จะถ่ายทอดลักษณะของมันไปสู่ลูกหลานของมัน
- ธรรมชาติทำให้สิ่งมีชีวิตมีลักษณะต่าง ๆ กัน
- สิ่งมีชีวิตมีความเหมาะสม ซึ่งมีลักษณะที่เหมาะสมที่สุด มีแนวโน้มที่จะมีลูกหลานมากกว่าสิ่งมีชีวิตที่มีลักษณะไม่เหมาะสม ซึ่งจะทำให้ประชากรอยู่รอดต่อไป
- เมื่อระยะเวลาผ่านไปยาวนาน จะเกิดการกลายพันธุ์ (Variation) ขึ้น และเกิดสปีชีส์ใหม่ที่มีลักษณะเหมาะสมกับระบบนิเวศน์นั้น

4.5.2 ความหมายของเจนเนติกอัลกอริทึม

GAs เป็นวิธีการค้นหาคำตอบโดยมีพื้นฐานมาจากกระบวนการคัดเลือกทางธรรมชาติ (Natural Selection) และ กระบวนการคัดเลือกทางพันธุศาสตร์ (Natural Genetics Selection) โดยการคัดเลือกสตริง (String) ที่มีความเหมาะสมจากกลุ่มของสตริงทั้งหมดด้วยวิธีการสุ่ม จากการนำสตริงเหล่านี้ไปผ่านกระบวนการคัดเลือกสตริงที่มีความเหมาะสม ซึ่งสตริงที่มีความเหมาะสมนี้คือคำตอบที่ดีที่สุดหรือใกล้เคียงคำตอบที่ดีที่สุด GAs ไม่ใช่การสุ่มแบบง่ายๆ แต่มันเป็นการใช้ข้อมูลในอดีตอย่างมีประสิทธิภาพเพื่อพิจารณาจุดที่จะต้องค้นหาใหม่โดยคาดหวังว่าสมรรถนะของการค้นหาจะดีขึ้น GAs ถูกพัฒนาขึ้นโดย Holland (1975) และคณะ โดยมีเป้าหมายในการวิจัย 2 อย่าง คือ ข้อแรก เพื่อสรุปและดัดแปลงการใช้กระบวนการทางธรรมชาติให้ถูกต้องมากที่สุด สองเพื่อออกแบบและสร้างซอฟแวร์ที่รักษากลไกที่สำคัญของธรรมชาติ และ GAs แตกต่างกับวิธีการค้นหาและการทำ Optimization แบบอื่นๆ คือ

- GAs ทำงานโดยการเข้ารหัสสตริงเป็นชุดพารามิเตอร์
- GAs เป็นการค้นหาจากทั้งประชากรไม่ใช่ค้นหาจากเพียงตำแหน่งๆเดียว
- GAs ใช้ข่าวสารที่เป็นผลลัพธ์ (ฟังก์ชันเป้าหมาย) โดยไม่ใช้การอนุพันธ์หรือความรู้อื่นๆ
- GAs จะเป็นวิธี Probabilistic ไม่ใช่ Deterministic

4.6 เจนเนติกอัลกอริทึมอย่างง่าย (Simple Genetic Algorithms)



รูปที่ 4.10 ขั้นตอนของ GAs อย่างง่าย

4.6.1 การเข้ารหัสและสร้างประชากรเริ่มต้นอย่างสุ่ม

ขั้นตอนแรกของ GAs คือ การเข้ารหัสหรือแปลงค่าพารามิเตอร์ให้อยู่ในรูปของสตริงที่มีความยาวแน่นอน ซึ่งวิธีการเข้ารหัสนี้ขึ้นอยู่กับรูปแบบของปัญหาแต่ละปัญหาสำหรับ SGA ใช้การเข้ารหัสแบบไบนารี่ (Binary Coding) ตัวอย่างเช่น ต้องการหาค่าสูงสุดของฟังก์ชัน $f(x)=x^2$ โดยที่ x มีค่าอยู่ระหว่าง [0,31] ในที่นี่ฟังก์ชันวัตถุประสงค์ (Objective Function) คือ $f(x)$ หรือ x^2 ซึ่งวิธีการเข้ารหัสแบบไบนารี่ โดยแปลงค่าพารามิเตอร์ x ให้อยู่ในรูปไบนารี่ 5 บิท จะได้ ค่าพารามิเตอร์ของ x จะมีค่าอยู่ในช่วง 00000 จนถึง 11111 (0 ถึง 31) เมื่อกำหนดวิธีการเข้ารหัสแล้ว จำเป็นที่จะต้องสร้างประชากรเริ่มต้น (Initial Population) โดยวิธีการสุ่มเพื่อที่จะผ่านขั้นตอนของ SGA ต่อไป สมมุติว่าสุ่มประชากรเริ่มต้น 4 สตริงได้เป็น

01101

11000

01000

10011

ค่าสตริงของประชากรเริ่มต้นนี้ เกิดจากการสุ่มค่า ทั้งหมด 20 ครั้งหรือ สตริงแต่ละตัวทำการสุ่ม 5 ครั้ง

4.6.2. ประชากรรุ่นเก่า (Old Population)

ประชากรรุ่นเก่า คือสตริงที่จะถูกคัดเลือกไปเป็นต้นแบบสำหรับสร้างประชากรรุ่นใหม่ (New Population) โดยประชากรรุ่นเก่าชุดแรกคือประชากรเริ่มต้นนั่นเอง

4.6.3 การดำเนินการของ SGA

SGA ประกอบไปด้วยตัวปฏิบัติการ 3 อย่างได้แก่ รีโปรดักชั่น (Reproduction) การครอสโอเวอร์ (Crossover) และการมิวเตชั่น (Mutation) ดังมีรายละเอียดดังต่อไปนี้

- **รีโปรดักชั่น** คือกระบวนการที่สตริงแต่ละตัวเลียนแบบค่าฟังก์ชันเป้าหมาย $f(x)$ โดยที่ฟังก์ชันนี้อาจเป็นการวัด ผลตอบแทน ค่าอรรถประโยชน์ หรือสิ่งที่ต้องการให้เป็นค่าสูงสุด หรือค่าความเหมาะสม (Fitness) สตริงที่มีความเหมาะสมสูงกว่าก็จะมีความน่าจะเป็นในการสนับสนุนลูกหลานรุ่นต่อไปสูงด้วย ตัวปฏิบัติการนี้เกิดขึ้นจาก

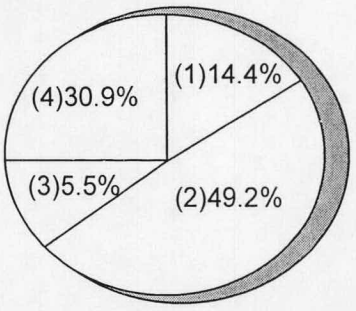
กระบวนการคัดเลือกตามธรรมชาติตามทฤษฎีผู้อยู่รอดที่มีความเหมาะสม (Survival of Fittest) ของ ชาล์ส ดาร์วิน ประชากรที่มีความเหมาะสมในธรรมชาติจะมีความสามารถในการรอดพ้นผู้ล่า โรคภัยไข้เจ็บ อุปสรรคอื่นๆที่ต่อต้านการเจริญเติบโตเป็นผู้ใหญ่และสามารถสืบพันธุ์ต่อไปได้ ส่วนฟังก์ชันเป้าหมายจะเป็นสิ่งที่ใช้พิจารณาว่าสตริงที่สร้างขึ้นจะมีชีวิตอยู่หรือตายจากไป

ตัวปฏิบัติการรีโปรดักชั่นสามารถสร้างขึ้นได้หลายวิธี วิธีที่ง่ายวิธีหนึ่งคือ สร้างจากวงล้อรูลเล็ทที่มีจำนวนช่องเท่ากับจำนวนประชากรสตริง และขนาดของช่องก็เป็นสัดส่วนกับค่าความเหมาะสม ดังรูปที่ 4.11 และค่าความเหมาะสมของฟังก์ชันเป้าหมายของประชากรทั้งสี่แสดงอยู่ในตาราง 4.2

ค่าความเหมาะสมทั้งหมดโดยรวมจะได้ 1170 และค่ารายละเอียดต่างๆแสดงดังในตารางที่ 4.2 แสดงถึงวงล้อรูลเล็ทสำหรับการรีโปรดักชั่น ซึ่งสร้างจากสัดส่วนของค่าฟิตเนสของสตริงทั้งหมด เช่นสตริงหมายเลข 1 มีค่าความเหมาะสมเป็น 169 หรือ 14.4% (169/1170) ของค่า Fitness โดยรวมของทั้งประชากร ในการทำการรีโปรดักชั่นจะหมุนวงล้อเป็นจำนวน 4 ครั้งหรือเท่ากับจำนวนสตริง เช่นสตริงหมายเลข 1 มีค่าเป็น 169 คิดเป็น 14.4% ของ ค่าความเหมาะสมทั้งหมด ดังนั้นเมื่อหมุนรูลเล็ท 1 ครั้งก็จะมีความน่าจะเป็นที่จะถูกเลือกเท่ากับ 0.144 ในการหมุนรูลเล็ทแต่ละครั้งจะได้ตัวแทนในการสืบพันธุ์ (Reproduction Candidate) สตริงที่มีความเหมาะสมสูงจะถูกคัดเลือกสำหรับการสืบพันธุ์การรีโปรดักชั่นสำหรับสตริงลูกหลานในรุ่นต่อไป เมื่อสตริงมีรูปร่างที่แน่นอนแล้วก็จะถูกส่งไปเข้าเมทติ้งพูล (Mating Pool) เพื่อที่จะผ่านกระบวนการของตัวปฏิบัติการอื่นต่อไป

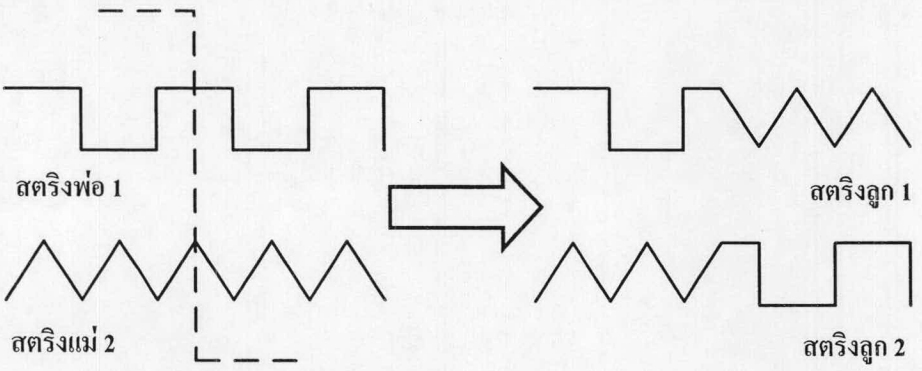
ตารางที่ 4.2 กลุ่มประชากรตัวอย่างและค่าเหมาะสม

No.	สตริง	ค่าความเหมาะสม	%โดยรวม
1	01101	169	14.40
2	11000	576	49.20
3	01000	64	5.50
4	10011	361	30.90
รวม		1170	100



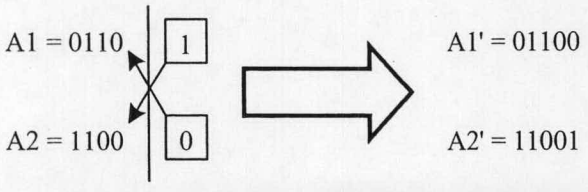
รูปที่ 4.11 การรีโปรดักชันอย่างง่ายด้วยวิธีการใช้วงล้อรูเล็ทที่มีขนาดของแต่ละช่องเป็นสัดส่วนกับค่าความเหมาะสม

- **การครอสโอเวอร์** หลังจากประชากรทั้งหมดผ่านกระบวนการรีโปรดักชั่นแล้ว จะทำการจับคู่สมาชิกในเมทติ้งพูลหรือกลุ่มประชากรทั้งหมดอย่างสุ่มและทำการไขว้สลับค่าที่อยู่หลังตำแหน่งที่เลือกไว้จากการสุ่มหรือ ทำการแลกเปลี่ยนสวนกัน



รูปที่ 4.12 การครอสโอเวอร์อย่างง่ายเพื่อให้เห็นถึงการเปลี่ยนแปลงสตริงและการแลกเปลี่ยนข่าวสารโดยเลือกตำแหน่งไขว้แบบสุ่ม

การเลือกตำแหน่งที่จะทำการครอสโอเวอร์ จะทำโดยการสุ่มค่าที่เป็นจำนวนเต็มตำแหน่งที่ k ช่วงของสตริงที่เลือกจะอยู่ในช่วง $[2,t-1]$ โดยที่ t คือตำแหน่งสุดท้ายของสตริงใหม่ทั้งสองก็จะมีการสลับอักขระตั้งแต่ตำแหน่งที่ $k+1$ จนถึง t ยกตัวอย่างเช่น พิจารณาสตริง A_1, A_2 จากประชากรเริ่มต้น



สมมติว่าเลือกจำนวนสุ่มระหว่าง 1 ถึง 4 และได้ค่า $k=$ 4 (แสดงโดยใช้สัญลักษณ์ " | " แทนการแยก) ผลของการครอสโอเวอร์สตริงที่เป็นประชากรรุ่นใหม่จึงมีสัญลักษณ์ " ' "

- **การมิวเตชั่น** มิวเตชั่นเป็นสิ่งที่จำเป็นถึงแม้ว่ารีโปรดักชั่นและครอสโอเวอร์ช่วยให้การค้นหาเป็นไปอย่างมีประสิทธิภาพในบางครั้งก็มีการสูญเสียส่วนที่สำคัญไป (ค่า 1 หรือ 0 ในบางตำแหน่ง) การมิวเตชั่นจะป้องกันส่วนที่สูญเสียที่ไม่อาจเรียกคืนได้ (I recovery Loss) ในบางครั้งการหาคำตอบของ SGA คำตอบอาจติดอยู่ใน Local Optimal การมิวเตชั่นด้วยอัตราส่วนที่เหมาะสมจะทำให้คำตอบสามารถหลุดออกจาก Local Optimal หรืออาจกล่าวได้ว่าโอเปอร์เรเตอร์ของการมีมิวเตชั่นเป็นการ เปลี่ยนแปลงค่าตำแหน่งสตริงแบบสุ่ม จากปัญหาที่พิจารณาค่าจะเปลี่ยนแปลงจาก 0 เป็น 1 หรือ 1 เป็น 0 โดยการเลือกตำแหน่งที่จะทำการมิวเตชั่นอย่างสุ่ม อัตราการมิวเตชั่นในธรรมชาติจะมีค่าค่อนข้างต่ำ ในการนำไปใช้งานจะต้องมีการพิจารณาอย่างเหมาะสม

4.6.4 ประชากรรุ่นใหม่ (New population)

สตริงทั้งหมดที่ได้จากกระบวนการของ GAs เรียกว่าประชากรรุ่นใหม่หรือเจนเนอเรชั่น (Generation) รุ่นใหม่ซึ่งจะกลายเป็นประชากรรุ่นเก่า สำหรับการดำเนินการต่อไป กระบวนการของ SGA จะทำซ้ำไปเรื่อยๆ จนกว่าจำนวนเจนเนอเรชั่นจะมากกว่าจำนวนเจนเนอเรชั่นที่กำหนดไว้สูงสุด

Surech (1995) ได้พิจารณาถึงการหาขนาดของประชากร จากอัตราส่วนของวิธีการที่ทั้งหมดของคำตอบที่เป็นไปไม่ได้และอัตราส่วนของวิธีการที่ทั้งหมดของคำตอบที่เป็นไปได้

$$\lim_{n\to\infty}\frac{n^n}{n!}\approx\lim_{n\to\infty}\frac{(2\pi)^{\frac{1}{2}}\left(\frac{n}{e}\right)^n}{n^n} \qquad (4.26)$$

$$=\lim_{n\to\infty}\frac{(2\pi)^{\frac{1}{2}}}{e^n}=0$$

จากสมการ (4.26) กำหนดให้

n คือขนาดของปัญหา

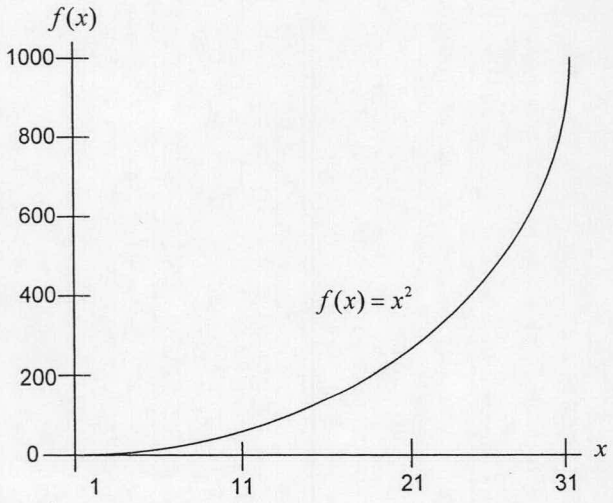
$n!$ คือจำนวนวิธีการที่จัดเรียงหรือจำนวนวิธีการจัดเรียงที่เป็นไปได้

n^n คือจำนวนวิธีการที่จัดเรียงหรือจำนวนวิธีการจัดเรียงทั้งหมด

สามารถสรุปได้ว่า ความน่าจะเป็นของในการสร้างประชากรคำตอบอย่างสุ่ม มีค่าเป็นศูนย์เมื่อ n มีค่ามากขึ้น สมมติว่า n มีค่าเป็น 8 ดังนั้น $8!/8^8 = 1/416.1 = 2.403x10^{-3}$ หรืออาจกล่าวได้ว่า โอกาสที่จะได้คำตอบที่ถูกต้องเป็น 1 ใน 416 ของคำตอบที่เป็นไปไม่ได้ ถ้ากำหนดจำนวนประชากรเป็น 100 และทำการคำนวณเพียงแค่ 1 เจนเนอเรชั่นก็ไม่อาจคาดได้ว่าจะได้คำตอบที่ดี การกำหนดจำนวนประชากรเริ่มต้นและจำนวนเจนเนเรชั่นทั้งหมดจึงมีผลในการหาคำตอบ

4.7 ตัวอย่างการใช้เจนเนติกอัลกอริทึมในการหาคำตอบของฟังก์ชัน

เนื้อหาในส่วนนี้จะเป็นการประยุกต์ใช้ GAs ในการแก้ปัญหา Optimization หาค่าสูงสุดของฟังก์ชัน $f(x) = x^2$ ทีละขั้นตอน โดย x เป็นตัวแปรที่มีค่าเปลี่ยนแปลงระหว่าง 1 ถึง 31 ดังรูปที่3.5 แสดงถึงลักษณะฟังก์ชัน $f(x)$ สำหรับปัญหานี้ตัวแปร x จะถูกเข้ารหัสให้เป็นไบนารี่ ที่มีความยาวสตริง 5 บิท



รูปที่ 4.13 ฟังก์ชันวัตถุประสงค์ $f(x) = x^2$

วิธีการทำเริ่มจากเลือกประชากรแรกขึ้นมาอย่างสุ่ม โดยประชากรเริ่มแรกจะได้มาจากการโยนเหรียญ 20 ครั้ง จากตาราง 4.3 จะเห็นได้ว่าสตริงหมายเลข 3 ซึ่งมีค่าเป็น 01000 (นำมาเข้าแปลงเป็นเลขฐานสิบ คือ 23 = 8) จากนั้นก็จะแปลงให้อยู่ในฟังก์ชันเป้าหมาย $f(x) = x^2$ จะได้ค่าเป็น 64สำหรับค่า x และ $f(x)$ อื่นๆก็คิดในลักษณะเดียวกัน

ตารางที่ 4.3 การคำนวณหาคำตอบของบ SGA กับฟังก์ชัน $f(x)=x^2$

ก) การสุ่มสตริงเริ่มต้นและการรีโปรดักชั่น

หมายเลขสตริง	ประชากรเริ่มต้น (สร้างขึ้นแบบสุ่ม)	ค่า x (unsigned integer)	$f(x)=x^2$	P select $f_i/\sum f$	Expected Count $f_i/\bar{f}$	Actual Count (จากวงล้อรูลเล็ท)
1	01101	13	169	0.14	0.58	1
2	11000	24	576	0.49	1.97	2
3	01000	8	64	0.06	0.22	0
4	10011	19	361	0.31	1.23	1
ผลรวม			1170	1.00	4.00	4.0
ค่าเฉลี่ย			293	0.25	1.00	1.0
ค่าสูงสุด			576	0.49	1.97	2.0

ข) การครอสโอเวอร์

เมทติ้งพูลหลังจากการรีโปรดักชั่น	สตริงจับคู่ (เลือกแบบสุ่ม)	ตำแหน่งครอสโอเวอร์(เลือกแบบสุ่ม)	ประชากรใหม่	ค่า x	$f(x)=x^2$
01101	2	4	01100	12	144
11000	1	4	11001	25	625
11000	4	2	11011	27	729
10011	3	2	10000	16	256
ผลรวม					1754
ค่าเฉลี่ย					439
ค่าสูงสุด					729

หมายเหตุ

1. ประชากรเริ่มแรกทั้งสี่ตัว ในแต่ละตัวได้มาจากการสุ่มโยนเหรียญ 5 ครั้ง (มี 5 บิท)
2. รีโปรดักชั่นได้จากการหมุนวงล้อรูลเล็ท
3. ครอสโอเวอร์ได้จากการโยนเหรียญสองเหรียญแล้วทำการถอดรหัส ($TT=00_2$ = 0 = ตำแหน่งที่ไขว้ คือ 1 และ $HH=11_2$ = 3 = ตำแหน่งที่ไขว้คือ 4)
4. ความน่าจะเป็นของครอสโอเวอร์กำหนดให้เป็นหนึ่ง $p_c=1.0$
5. ความน่าจะเป็นของมิวเตชั่นเป็น 0.001 , p =0.001 , Expected Mutation = $5\times4\times0.001=0.2$ ไม่มีค่า Expected Mutation ระหว่างประชากรเดี่ยว

ประชากรรุ่นต่อไป จะเริ่มต้นกระบวนการรีโปรดักชั่นจากเมทติ้งพูล โดยการหมุนวงล้อรูลเล็ท 4 ครั้ง ได้สตริงหมายเลข 1 และ 4 ได้รับการคัดเลือกไปยังรุ่นต่อไป 1 ครั้ง สตริง 2 ได้รับการคัดเลือกไปยังรุ่นต่อไป 2 ครั้ง สตริง 3 ไม่ได้รับการคัดเลือกไปยังรุ่นต่อไปเลย เมื่อเปรียบเทียบจำนวนครั้งที่ถูกคัดเลือกที่คาดหวัง หรือ Expected Count (หาได้จาก $f_i/\bar{f}$) กับจำนวนครั้งที่ถูกคัดเลือกจริงจะเห็นได้ว่ามีค่าใกล้เคียงกัน ค่าที่ดีที่สุดจะมีโอกาสที่จะถูกคัดเลือกมากกว่า ส่วนค่าที่ไม่ดีก็จะตายจากไป

ขั้นตอนต่อไปคือการครอสโอเวอร์ ซึ่งจะต้องมีการจับคู่กันระหว่างสตริง โดยมีสองขั้นตอน คือ (1) สตริงจะถูกจับคู่อย่างสุ่มโดยใช้วิธีการโยนเหรียญจับคู่ (2) สตริงจะทำการครอสโอเวอร์โดยการโยนเหรียญเพื่อเลือกตำแหน่งที่จะไขว้ (Crossing sites) เมื่อพิจารณาตาราง 4.3ข อีกครั้ง จะเห็นได้ว่าจากการสุ่มจับคู่ในเมทติ้งพูล สตริงหมายเลข 2 จะจับคู่กับสตริงหมายเลข 1 และมีตำแหน่งการไขว้คือ 4 สตริงทั้งสองคือ 01101 และ 11000 เมื่อทำการไขว้จะได้สตริงตัวใหม่คือ 01100 และ 11001 สตริงที่เหลือในเมทติ้งพูลจะทำการไขว้กันในตำแหน่งที่สองดังแสดงในตารางที่4.3ข

ตัวปฏิบัติการสุดท้ายคือมิวเตชั่นซึ่งจะเปลี่ยนค่าเป็นบิทต่อบิท สมมุติความน่าจะเป็นของการมิวเตชั่นในการทดสอบเป็น 0.001 ตำแหน่งที่จะเปลี่ยนแปลงทั้งหมดมี 20 บิท (ได้จากจำนวนสตริง*จำนวนบิทของสตริงแต่ละตัว $5\times 4 = 20$) เพราะฉะนั้นตำแหน่งบิทที่จะมิวเตชั่นของประชากรรุ่นนี้คือ $20\times 0.001 = 0.02$ บิท จากการคำนวณจะเห็นได้ว่าไม่มีบิทใดต้องทำการมิวเตชั่นสำหรับค่าความน่าจะเป็นนี้ นั่นก็คือไม่มีบิทใดที่จะต้องเปลี่ยนค่าจาก 1 เป็น 0 หรือ 0 เป็น 1 สำหรับประชากรรุ่นนี้ แต่สมมติว่าถ้าตำแหน่งบิทที่จะมิวเตชั่นของประชากรรุ่นนี้คือ 5 ดังนั้นตำแหน่งบิทที่ 5 จะต้องทำการเปลี่ยนค่าจาก 0 เป็น 1 หรือ 1 เป็น 0

หลังจากผ่านการรีโปรดักชั่น ครอสโอเวอร์และมิวเตชั่น ประชากรรุ่นใหม่ก็พร้อมที่จะถูกทดสอบ โดยทำการเข้ารหัสสตริงใหม่คำนวณหาค่า x และค่าฟังก์ชัน $f(x)$ ตารางที่ 3.3 ข. แสดงถึงผลจากการทดลองจะเห็นได้ว่ากระบวนการที่เกี่ยวข้องกับความน่าจะเป็นทำให้ค่าสมรรถนะดีขึ้น ค่าความเหมาะสมของประชากรโดยเฉลี่ยมีค่าเพิ่มขึ้นจาก 293 เป็น 439 ในขณะที่ค่าความเหมาะสม
สูงสุดมีค่าเพิ่มขึ้นจาก 576 เป็น 729 ถึงแม้ว่ากระบวนการสุ่มจะช่วยทำให้ค่าต่างๆสูงขึ้นแต่ค่าต่างๆ ที่เพิ่มขึ้นเหล่านี้ไม่ใช่ความบังเอิญ ค่าสตริงที่ดีที่สุดของประชากรเริ่มแรกคือ (11000) จะมีการเลียนแบบ 2 ครั้งเนื่องจากเป็นค่าที่สูงเกินกว่าค่าเฉลี่ย เมื่อรวมกับค่าสตริงตัวต่อไป (10011) แบบสุ่มและทำการไขว้แบบสุ่มในตำแหน่งที่สองก็จะได้ผลลัพธ์เป็น (11011) ซึ่งก็จะเป็นค่าที่ดีเช่นกัน

ค่าพารามิเตอร์ของ SGA มีความสำคัญอย่างมาก ค่าพารามิเตอร์เหล่านี้ในบางครั้งจำเป็นต้องมีการปรับเปลี่ยนไปตามรูปแบบของปัญหาเพื่อให้ได้คำตอบที่ดี แต่ในบางครั้งก็ไม่อาจที่จะหาคำตอบที่ดีได้เนื่องจาก (Michalewicz, 1992)

1. การเข้ารหัสของปัญหาผิดพลาด ทำให้ GAs หาคำตอบผิดพลาด
2. ขีดจำกัดของจำนวนประชากร ในทางทฤษฎีแล้วมีค่าเป็นอนันต์
3. ขีดจำกัดของจำนวนเจนเนอเรชั่น ในทางทฤษฎีแล้วมีค่าเป็นอนันต์

ค่าพารามิเตอร์เหล่านี้ไม่สามารถกำหนดให้เป็นอนันต์ได้ในทางปฏิบัติเนื่องจากข้อจำกัดต่างๆของคอมพิวเตอร์

4.8 สรุปขั้นตอนการทำงานของอัลกอริทึมที่ได้รับความนิยมใช้ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์

ในหัวข้อนี้เป็นขั้นตอนการทำงานของอัลกอริทึมที่ได้รับความนิยมใช้ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ รายละเอียดต่างๆ ในอัลกอริทึมที่ได้รวบรวมมานั้นจะเป็นการสรุปในรูปแบบขั้นตอนการทำงาน ซึ่งทั้งหมด ได้แก่ MOGA, NSGA II, และ SPEA โดยแนวคิดต่างๆของอัลกอริทึมมีพื้นฐานดังนี้

ตารางที่ 4.4 อัลกอริทึมที่ได้รับความนิยมใช้ในการแก้ปัญหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์

MOEAs	MOEAs + Local Search
• Multi-objective Genetic Algorithm : MOGA (1993) • Non-dominate Sorting Genetic Algorithm II: NSGA II (2002) • Strength Pareto Evolutionary Algorithm II : SPEA II (2001)	• Multi-Objective Genetic Local Search : MOGLS (2003) • Modified MOGLS (2003) • S-MOGLS (2004)

จากตารางที่ 4.4 แสดงอัลกอริทึมที่ดีรับความนิยมในการใช้แก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ โดยสามารถจำแนกอัลกอริทึมได้เป็นสองประเภท คือประเภทแรก เอลโวลูชันนารี่แบบหลายวัตถุประสงค์ ที่ใช้การกำหนดค่าความแข็งแรง (Fitness Assignment) ด้วย

วิธีเชิงกลุ่มที่ดีที่สุดในการหาคำตอบ ในแต่ละอัลกอริทึ่มจะแต่งต่างกันในเรื่องการกำหนดค่าความแข็งแรงและความหนาแน่นของสมาชิกคำตอบ ประเภทที่สองการรวมกันของเอลโวลูชันนารี่แบบหลายวัตถุประสงค์และการประยุกต์การค้นหาเฉพาะที่ อาจเรียกอัลกอริทึ่มประเภทนี้ว่า เมมเมติกอัลกอริทึ่ม ซึ่งจากตารางข้างต้นในอัลกอริทึ่มนั้นจะแบ่งออกเป็นสองแนวคิด คือการคิดค้นวิธีการแก้ปัญหาขึ้นมาใหม่อย่าง Multi-objective Genetic Local Search ซึ่งจะใช้การกำหนดค่าความแข็งแรงให้กับสมาชิกคำตอบด้วยวิธีการรวมฟังก์ชั่นโดยอาศัยการถ่วงน้ำหนัก แต่ผลลัพธ์ยังคงให้กลุ่มคำตอบที่ดีที่สุด และนำการค้นหาเฉพาะที่มาช่วยในการปรับปรุงคำตอบ จากแนวคิดข้างต้นในงานวิจัยนี้ได้เลือกใช้อัลกอริทึ่มที่ได้รับการยอมรับว้เป็นอัลกอริทึ่มที่ได้รับความนิยมในการแก้ปัญหาการจัดตารางอย่าง NSGA II และ SPEA II มาใช้ประยุกต์ร่วมกันกับการค้นหาเฉพาะที่ รายละเอียดของวิธีการของอัลกอริทึ่มใหม่จะนำเสนอในบทที่ 6 ต่อไป

4.8.1 ขั้นตอนการทำงานของ Multi-Objective Genetic Algorithm (MOGA)

กำหนดให้ในเจเนอเรชั่น t

P_t แทนประชากรคำตอบ

ขั้นตอนที่ 1 สร้างประชากรคำตอบเบื้องต้นอย่างสุ่ม P_t จำนวน N ประชากร

ขั้นตอนที่ 2 คำนวณค่าฟังก์ชั่นวัตถุประสงค์ของประชากรคำตอบเบื้องต้น

ขั้นตอนที่ 3 กำหนดค่าความแข็งแรงให้กับสมาชิกประชากรคำตอบด้วยวิธีเชิงกลุ่มที่ดีที่สุดสำหรับอัลกอริทึ่มนี้ใช้วิธีการจัดอันดับแบบ Fonseca และ Fleming โดยค่าอันดับนี้จะเป็นค่าความแข็งแรงไม่แท้จริง หรือ Dummy Fitness Value โดยในขั้นตอนนี้จะทำให้ได้เส้นขอบเขตกลุ่มคำตอบที่ดี (Frontier) ออกมาหลายกลุ่มตามค่า Dummy Fitness

3.1 จัดสรรค่าความแข็งแรงให้กับสมาชิกประชากรคำตอบด้วยพื้นฐานของอันดับที่ได้รับ

$$f(x,t) = N - \sum_{k=1}^{r(x,t)-1} n_k - 0.5x(n_{r(x,t)} - 1) \qquad (4.27)$$

โดย n_k คือจำนวนคำตอบใยอันดับที่ k

ขั้นตอนที่ 4 คำนวณค่าการแบ่งปันความแข็งแรงให้กับคำตอบแต่ละสมาชิกประชากรคำตอบ ในที่นี้ใช้ Niche Fitness Sharing Technique ในการประมาณความหนาแน่นของคำตอบ คำตอบที่มีความหนาแน่นน้อยจะมีโอกาสได้รับการคัดเลือกเข้าสู่ Mating Pool มาก

4.1 คำนวณ Niche Count nc(x,t) ในสมชิกประชากรคำตอบในเจเนอเรชั่นปัจจุบัน

4.2 คำนวณตค่าความแบ่งปันความแข็งแรงของแต่ละสมาชิกประชากรคำตอบ

$$f'(x,t) = \frac{f(x,t)}{nc(x,t)} \tag{4.28}$$

4.3 ทำการ Normalized ค่าฟังก์ชั่นวัตถุประสงค์ด้วยค่าการแบ่งปันความแข็งแรง

$$f''(x,t) = \frac{f(x,t)n_{r(x,t)}}{\sum_{\substack{y \in P_t \\ r(y,t)=r(x,t)}} f'(x,t)} \tag{4.29}$$

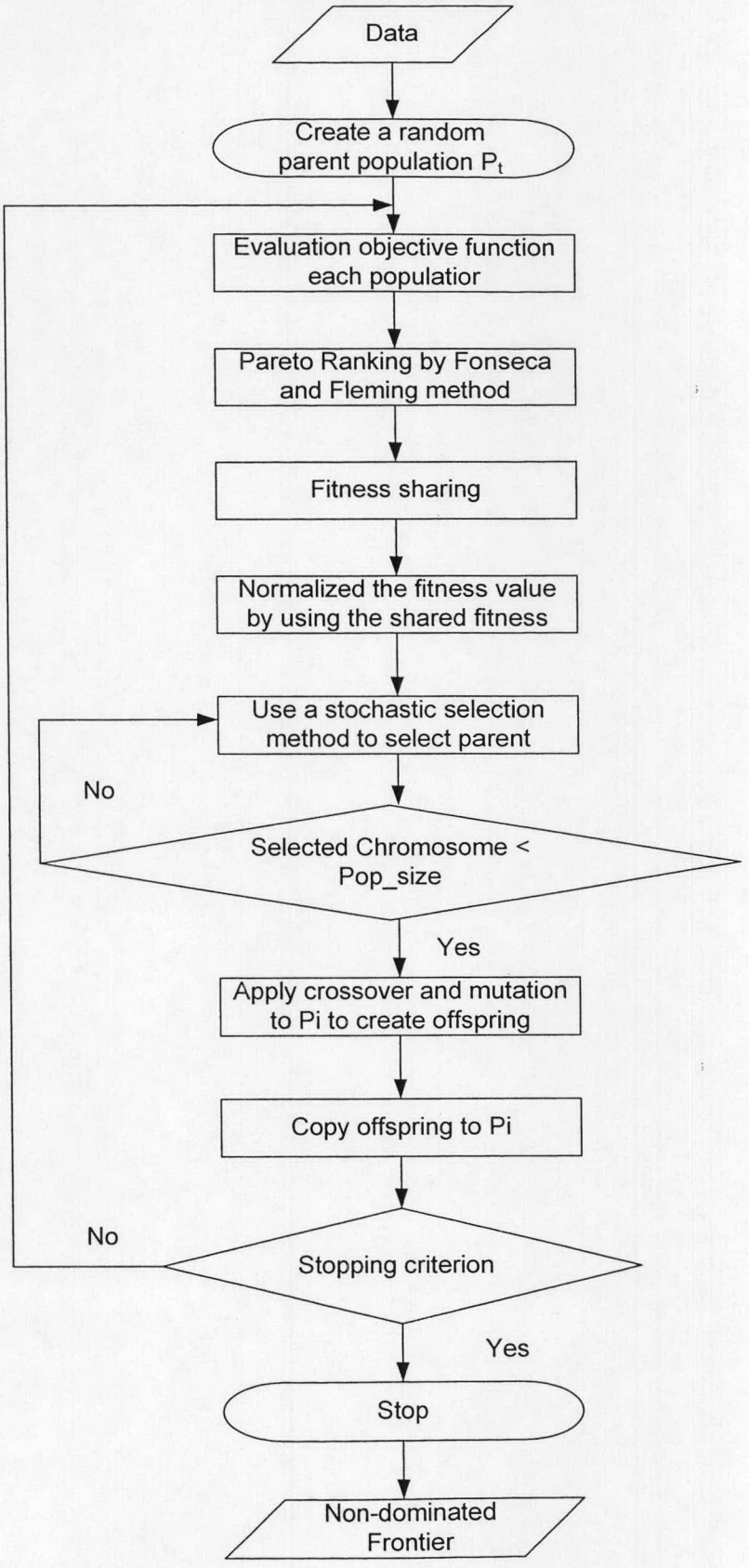
ขั้นตอนที่ 5_คัดเลือกคำตอบเข้าสู่ Mating Pool ด้วย Stochastic Method วิธีนี้จะมีหลักการเหมือนกับการคัดเลือกแบบ Roulette Wheel Selection Method ต่างกันที่หลังจากกำหนดจุดชี้ตำแหน่ง (Fixed Point) โดยการสุ่มครั้งแรกแล้ว จะทำการเลือกสมาชิกของกลุ่มประชากรด้วยการเลื่อนตัวชี้ตำแหน่งจากจุดเดิม (ตัวชี้ตำแหน่งที่ได้เป็นตัวแรก) ทีละขั้น โดยที่แต่ละขั้นนั้นจะเท่ากับ 360 องศาต่อจำนวนสมาชิกของกลุ่มประชากร จากนั้นทำการคัดเลือกสมาชิกของกลุ่มประชากรที่มีตัวชี้ตำแหน่งจนครบตามจำนวนสมาชิกของกลุ่มประชากรในหนึ่งรุ่น

ขั้นตอนที่ 6 ใช้ตัวดำเนินการพันธุกรรมอย่างครอสโอเวอร์ และมิวเทชั่นในการสร้างประชากรคำตอบใหม่ จากประชากรคำตอบ P_t จำนวน N ประชากร

ขั้นตอนที่ 7 ตรวจสอบการ Stopping Criterion ด้วยการคำนวณค่าฟังก์ชั้นไปจนครบจำนวนที่กำหนดไว้ล่วงหน้า

7.1 ถ้ายังไม่หยุดกระบวนการหาคำตอบ จะทำการคัดเลือกประชากรคำตอบไปเป็นประชากรคำตอบเริ่มต้นในเจนเนอเรชั่น $t+1$ (กลับไปขั้นตอนที่ 2)

7.2 ถ้าหยุดกระบวนการ และนำคำตอบที่ได้ในเจนเนอเรชั่นสุดท้ายเป็นคำตอบ



รูปที่ 4.14 ขั้นตอนการทำงานของ Multi-Objective Genetic Algorithm (MOGA)

4.8.2 ขั้นตอนการทำงานของ Strength Pareto Evolutionary Algorithm 2 (SPEA 2)

กำหนดให้ในเจนเนอเรชั่น t

P_t แทนประชากรคำตอบ

E_t แทนประชากรคำตอบที่ดีที่สุด (Non-dominated Solution หรือ Archive Population)

N_P แทนจำนวนประชากรคำตอบ (Population Size)

N_E แทนจำนวนของประชากรคำตอบที่ดีที่สุดที่เก็บไว้ (Archive Size)

k แทนจำนวนของประชากรคำตอบที่ดีที่สุดที่เก็บไว้

พารามิเตอร์ที่ใช้ในการคำนวณค่าความหนาแน่นของประชากรคำตอบโดย

$$k = \sqrt{N_P + N_E} \tag{4.30}$$

ขั้นตอนที่ 1 สร้างประชากรคำตอบเบื้องต้นอย่างสุ่ม P_t จำนวน N_P ประชากร และกำหนดให้ $E_t = \phi$

ขั้นตอนที่ 2 คำนวนค่าฟังก์ชันวัตถุประสงค์ของประชากรคำตอบ P_t และ E_t

ขั้นตอนที่ 3 กำหนดค่าความแข็งแรงให้กับสมาชิกประชากรคำตอบด้วยวิธีเชิงกลุ่มที่ดีที่สุดสำหรับอัลกอริทึมนี้ใช้วิธีการจัดอันดับแบบ Strength of Dominators โดยจะค่าความแข็งแรง (Fitness Value: $F(i)$) ได้มาจากผลรวมของ Raw Fitness Value: $R(i)$ และ Density Estimation: $D(i)$ ของประชากรคำตอบ P_t และ E_{t+1} ซึ่งสามารถคำนวณค่าต่างๆดังกล่าวได้ในสมการที่ (4.18) (4.16) และ (4.24) ตามลำดับ

ขั้นตอนที่ 4 พิจารณาสมาชิกประชากรคำตอบที่มี เพื่อคัดลอกคำตอบไปเก็บไว้ใน E_{t+1}

ขั้นตอนที่ 5 ตรวจสอบว่าประชากรคำตอบใน E_{t+1} มีจำนวนคำตอบเท่ากับ N_E หรือไม่

5.1 กรณี Archive มีขนาดเล็กเกินไป นั่นคือ $|E_{t+1}| > N_E$ ให้พิจารณาจำนวนคำตอบเท่ากับ $|E_{t+1}| - N_E$ ที่มีDensity มากที่สุด แล้วคัดเลือกคำตอบไปเก็บไว้ใน E_{t+1} จนครบ N_E

5.2 กรณี มีขนาดใหญ่เกินไป นั้นคือ $|E_{t+1}| < N_E$ ให้พิจารณาจำนวนคำตอบเท่ากับ $N_E - |E_{t+1}|$ ที่มี Fitness Values น้อยจาก P_t และ E_t แล้วคัดเลือกคำตอบไปเก็บไว้ใน E_{t+1} จนครบ N_E

ขั้นตอนที่ 6 ตรวจสอบการ Stopping Criterion ว่าการคำนวณค่าฟังก์ชันครบจำนวนที่กำหนดไว้ล่วงหน้าหรือไม่

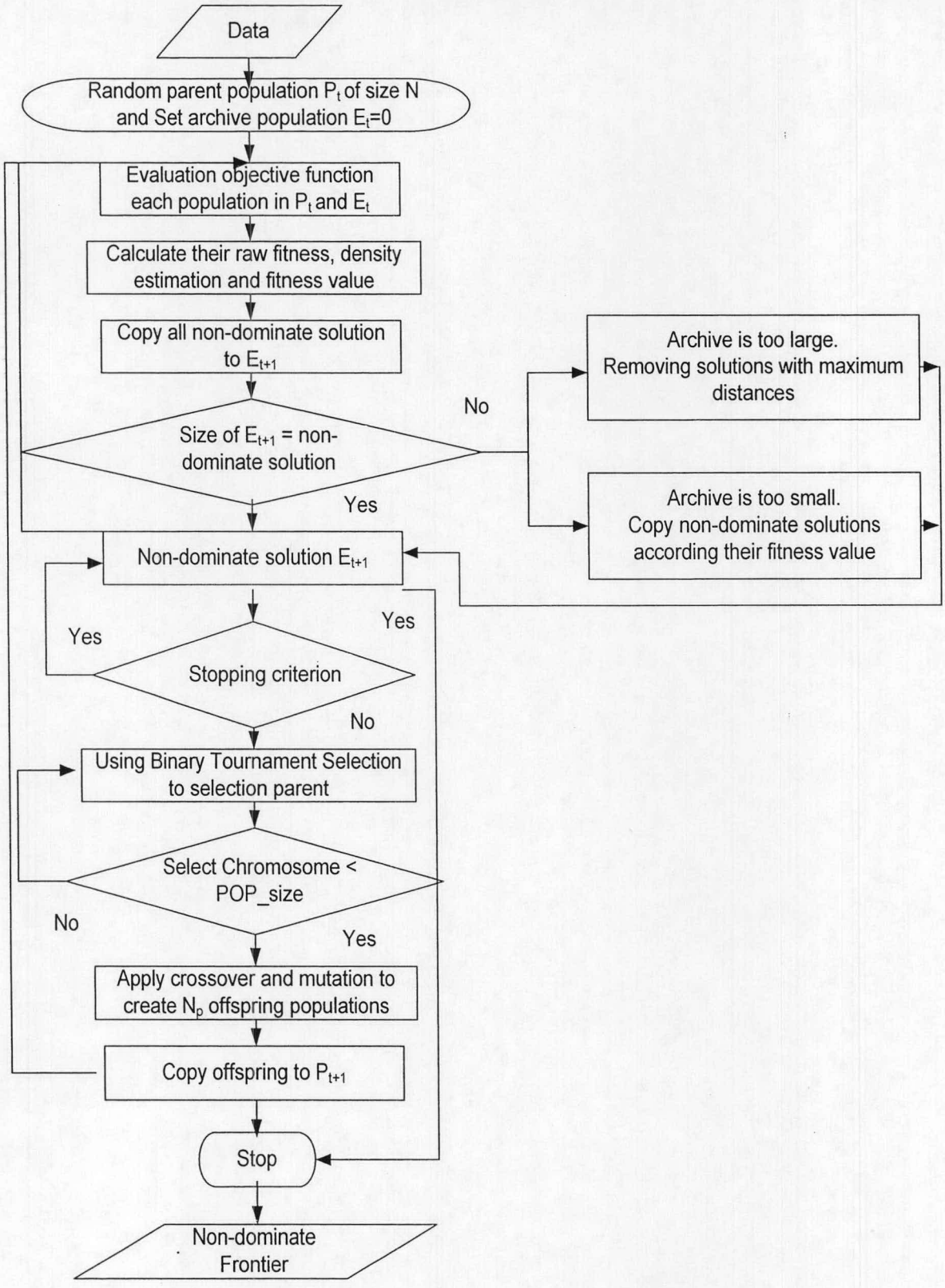
6.1 ถ้ายังไม่หยุดกระบวนหาคำตอบจะดำเนินการต่อไปในขั้นตอนที่ 7

6.2 ถ้าหยุดกระบวนการ และนำคำตอบที่ได้ใน E_{t+1} เจนเนอเรชั่นสุดท้ายเป็นคำตอบ

ขั้นตอนที่ 7 คัดเลือกคำตอบใน E_{t+1} เพื่อเป็นประชากรคำตอบรุ่นพ่อแม่ด้วยวิธี Binary Tournament Selection วิธีนี้เป็นวิธีที่ใช้การแข่งขันของประชากรคำตอบ 2 ประชากร คำตอบที่มี Fitness Valueน้อยกว่าจะได้รับเลือกเข้าสู่ Mating Pool

ขั้นตอนที่ 8 ใช้ตัวดำเนินการพันธุกรรมอย่างครอสโอเวอร์ และมิวเทชั่นในการสร้างประชากรรุ่นลูกจากประชากรคำตอบ P_t จำนวน N_P ประชากร

ขั้นตอนที่ 9 คัดลอกประชากรคำตอบรุ่นลูกที่ได้จากขั้นตอนที่ 8 ไปเป็นประชากรรุ่นพ่อแม่ในเจนเนอเรชั่น และ $t+1$ กลับไปสู่ขั้นตอนที่ 2



รูปที่ 4.15 ขั้นตอนการทำงานของ Strength Pareto Evolutionary Algorithm 2 (SPEA 2)

4.8.3 ขั้นตอนการทำงานของ Rank Density Genetic Algorithm (RDGA)

กำหนดให้ในเจนเนอเรชั่น t

P_t แทนประชากรคำตอบ

E_t แทนประชากรคำตอบที่ดีที่สุด (Non-dominated Solution หรือ Archive Population)

N_P แทนจำนวนประชากรคำตอบ (Population Size)

N_E แทนจำนวนของประชากรคำตอบที่ดีที่สุดที่เก็บไว้ (Archive Size)

ขั้นตอนที่ 1 สร้างประชากรคำตอบเบื้องต้นอย่างสุ่ม P_t จำนวน N_P ประชากร

ขั้นตอนที่ 2 คำนวณค่าฟังก์ชันวัตถุประสงค์ของประชากรคำตอบ P_t

ขั้นตอนที่ 3 กำหนดค่าความแข็งแรงให้กับสมาชิกประชากรคำตอบด้วยวิธีเชิงกลุ่มที่ดีที่สุดสำหรับอัลกอริทึมนี้ใช้วิธีการจัดอันดับแบบ Automatic Accumulated Ranking Strategy สามารถคำนวณได้ในสมการที่ (4.15) จากนั้นพิจารณาสมาชิกประชากรคำตอบที่มีค่าความาแข็งแรงเท่ากับ 1 คัดลอกคำตอบไปเก็บไว้ใน

ขั้นตอนที่ 4 ทำการแบ่งพื้นที่ฟังก์ชันวัตถุประสงค์ออกเป็นเซลล์ ซึ่งสามารถคำนวณได้ในสมการ (4.23)

ขั้นตอนที่ 5 คำนวณความหนาแน่นใน แต่ละเซลล์ให้กับแต่ละสมาชิกประชากรคำตอบ

ขั้นตอนที่ 6 ตรวจสอบว่าประชากรคำตอบใน E_{t+1} มีจำนวนคำตอบเท่ากับ N_E หรือไม่

6.1 กรณี Archive มีขนาดเล็กเกินไป นั่นคือ $|E_{t+1}| > N_E$ ให้พิจารณาจำนวนคำตอบเท่ากับ $|E_{t+1}| - N_E$ ที่มีค่าความแข็งแรงและความหนาแน่นน้อย แล้วคัดเลือกคำตอบไปเก็บไว้ใน E_{t+1} จนครบ N_E

6.2 กรณี มีขนาดใหญ่เกินไป นั้นคือ $|E_{t+1}| < N_E$ ให้พิจารณาจำนวนคำตอบเท่ากับ $N_E - |E_{t+1}|$ ที่มีการหาค่าความแข็งแรงด้วย VEGA เลือกประชากรคำตอบที่มีค่าความแข็งน้อยจากจาก P_t แล้วคัดเลือกคำตอบไปเก็บไว้ใน E_{t+1} จนครบ N_E

ขั้นตอนที่ 7 ตรวจสอบการ Stopping Criterion ว่าการคำนวณค่าฟังก์ชันครบจำนวนที่กำหนดไว้ล่วงหน้าหรือไม่

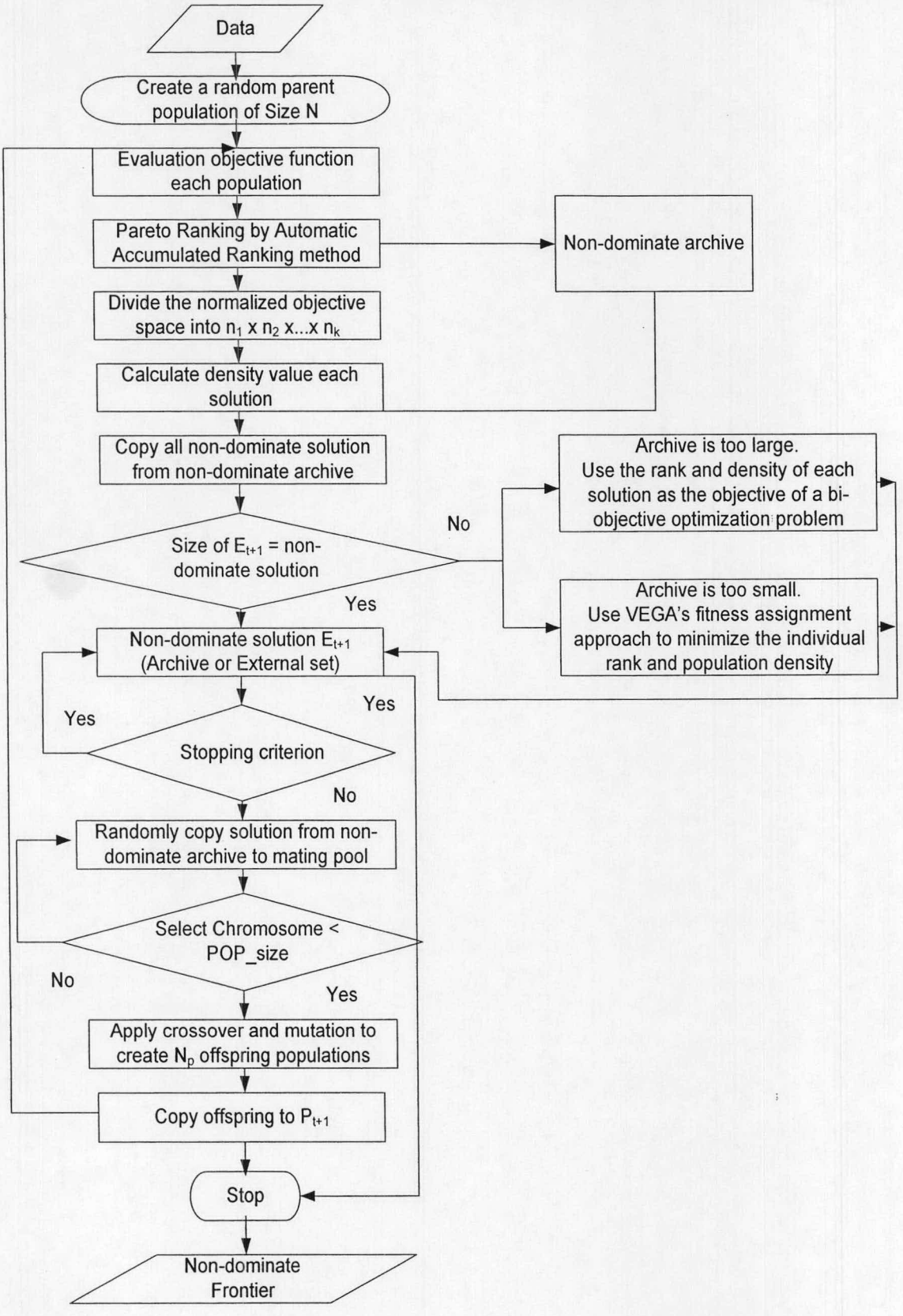
7.1 ถ้ายังไม่หยุดกระบวนหาคำตอบ จะดำเนินการต่อไปในขั้นตอนที่ 8

7.2 ถ้าหยุดกระบวนการ และนำคำตอบที่ได้ใน E_{t+1} เจนเนอเรชั่นสุดท้ายเป็นคำตอบ

ขั้นตอนที่ 8 คัดเลือกคำตอบใน E_{t+1} เพื่อเป็นประชากรคำตอบรุ่นพ่อแม่ด้วยวิธีการสุ่มประชากรคำตอบเข้าสู่ Mating Pool

ขั้นตอนที่ 9 ใช้ตัวดำเนินการพันธุกรรมอย่างครอสโอเวอร์ และมิวเทชั่นในการสร้างประชากรรุ่นลูกจากประชากรคำตอบ P_t จำนวน N_P ประชากร

ขั้นตอนที่ 10 คัดลอกประชากรคำตอบรุ่นลูกที่ได้จากขั้นตอนที่ 9 ไปเป็นประชากรรุ่นพ่อแม่ใน $t+1$ เจนเนอเรชั่น และกลับไปสู่ขั้นตอนที่ 2



รูปที่ 4.16 ขั้นตอนการทำงานของ Rank Density Genetic Algorithm (RDGA)

4.8.4 **ขั้นตอนการทำงานของ** Non-dominated sorting Genetic Algorithm II (NSGA -II)

กำหนดให้ในเจนเนอเรชั่น t

P_t แทนประชากรคำตอบ (ประชากรคำตอบรุ่นพ่อแม่)

Q_t แทนประชากรคำตอบใหม่ (ประชากรคำตอบรุ่นลูก)

R_t แทนการรวมกันของประชากรคำตอบรุ่ยพ่อแม่และประชากรคำตอบรุ่นลูก

ขั้นตอนที่ 1 สร้างประชากรคำตอบเบื่องต้นอย่างสุ่ม P_t จำนวน N ประชากร

ขั้นตอนที่ 2 คำนวณค่าฟังก์ชั่นวัตถุประสงค์ของประชากรคำตอบเบื้องต้น

ขั้นตอนที่ 3 กำหนดค่าความแข็งแรงให้กับสมาชิกคำตอบด้วยวิธีเชิงกลุ่มที่ดีที่สุด สำหรับอัลกอริทึ่มนี้ใช้วิธีการจัดอันดับแบบ Goldberg โดยค่าอันดับนี้จะเป็นค่าความแข็งแรงไม่แท้จริง หรือ Dummy Fitness Value โดยในขั้นตอนนี้จะทำให้ได้เส้นขอบเขตกลุ่มคำตอบที่ดี (Frontier) ออกมาหลายกลุ่มตามค่า Dummy Fitness

ขั้นตอนที่ 4 คัดเลือกคำตอบเข้าสู่ Mating Pool ด้วยวิธี Binary Tournament Selection วิธีนี้เป็นวิธีที่ใช้การแข่งขันของประชากรคำตอบ 2 ประชากรคำตอบ ประชากรคำตอบใดที่มีค่าความแข็งแรงที่ไม่แท้จริงน้อยกว่า จะได้รับเลือกเข้าสู่ Mating Pool และในกรณีที่พบว่าคำตอบนั้นมีค่าความแข็งแรงเท่ากัน จะพิจารณาที่ Crowding Distance ที่มีค่ามากกว่า จะได้รับการคัดเลือกเข้าสู่ Mating Pool

ขั้นตอนที่ 5 ใช้ตัวดำเนินการพันธุกรรมอย่างครอสโอเวอร์ และมิวเทชั่นในการสร้างประชากรคำตอบใหม่ จากประชากรคำตอบ P_t จำนวน N ประชากร

ขั้นตอนที่ 6 รวมประชากรคำตอบเบื้องต้น (P_t) และประชากรคำตอบใหม่ (Q_t) เป็นประชากรคำตอบใน R_t ดังนั้นจำนวนประชากรทั้งหมดเท่ากับ $2N$ และอัพเดท (Update) ในทุกเจนเนอเรชั่น

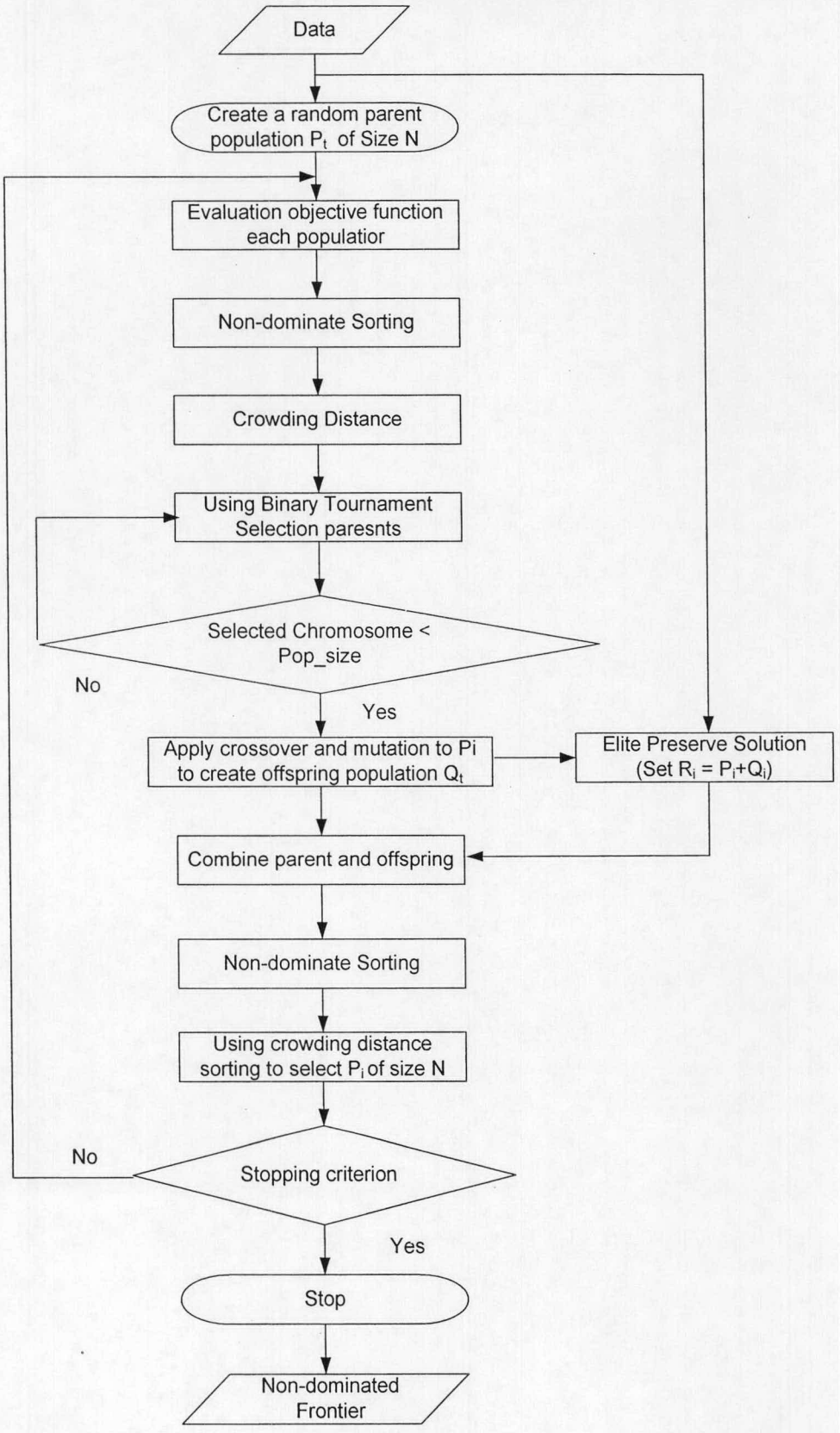
ขั้นตอนที่ 7 คำนวณค่าความแข็งแรงด้วย Non-dominated sorting และ Crowding Distance ให้กับ ประชากรคำตอบ R_t เพื่อคัดเลือกประชากรคำตอบไปสู่ประชากรคำตอบเริ่มต้น

ในเจนเนอเรชั่น $t+1$ จำนวน N ประชากรคำตอบ โดยประชากรคำตอบที่มีอันดับเป็นหนึ่ง หรืออยู่ใน Rank 1 จะได้รับโอกาสในการคัดเลือกไปเป็นคำตอบในเจนเนอเรชั่น $t+1$ อันดับแรก ส่วนอันดับอื่นๆ จะได้รับโอกาสลดหลั่นลงมาหากพบว่าจำนวนประชากรคำตอบใน Rank ที่พิจารณาอยู่มากกว่าจำนวนประชากรคำตอบที่เหลือที่จะนำไปเป็นประชากรรุ่น $t+1$ จะใช้ Crowding Distance ในการคัดเลือก ประชากรคำตอบที่มี Crowding Distance น้อยจะถูกตัดออกไปและคัดเลือกจนครบจำนวน N ประชากรคำตอบ

ขั้นตอนที่ 8 ตรวจสอบการ Stopping Criterion ว่าการคำนวณค่าฟังก์ชั่นครบจำนวนที่กำหนดไว้ล่วงหน้าหรือไม่

8.1 ถ้ายังไม่หยุดกระบวนการหาคำตอบ จะทำการคัดเลือกประชากรคำตอบไปเป็นประชากรคำตอบเริ่มต้นในเจนเนอเรชั่น $t+1$ (กลับไปขั้นตอนที่ 2)

8.2 ถ้าหยุดกระบวนการ และนำคำตอบที่ได้ในเจนเนอเรชั่นสุดท้ายเป็นคำตอบ



รูปที่ 4.17 ขั้นตอนการทำงานของ Non-dominated sorting Genetic Algorithm II (NSGA-II)

4.9 ขั้นตอนการทำงานของเจนเนติกอัลกอริทึมแบบ NSGA-II

ในส่วนนี้จะกล่าวถึงขั้นตอนการทำงานของ NSGA-II อัลกอริทึม ซึ่งจากโครงสร้างหลักของเจนแนติกอัลกอริทึม สามารถแบ่งย่อยเป็นวิธีการของ NSGA-II ได้ดังนี้

1. Data Input : รับข้อมูลนำเข้าต่าง ๆ ได้แก่ ชนิดและจำนวนผลิตภัณฑ์ แผนภาพแสดงความสัมพันธ์ในแต่ละผลิตภัณฑ์ เวลาการทำงานของแต่ละขั้นงาน
2. Representation & Initialization : นำข้อมูลนำเข้าต่าง ๆ มาสร้างคำตอบเบื้องต้นอย่างสุ่ม จำนวน *popsize* ตัว โดยผ่านกระบวนการใส่รหัสคำตอบ (Representation) และการสร้างประชากรคำตอบเบื้องต้น(Initial Population)
3. Evaluation : คำนวณหาค่าต่าง ๆ ที่ต้องการ เช่น จำนวนสถานีงาน ผลต่างสถานีงานกับความสัมพันธ์ในสถานีงาน และการกระจายภาระขั้นงานในสถานีงาน
4. Pareto Based Approach : ใช้เทคนิควิธีเชิงกลุ่มที่ดีที่สุดวิธีการจัดลำดับแบบ Goldberg หรือ Non-dominated Sorting ในการกำหนดความแข็งแรงให้กับประชากรคำตอบ ในขั้นตอนการทำงานนี้จะทำให้ประชากรคำตอบถูกแบ่งออกเป็นกลุ่ม กลุ่มที่ดีที่สุดจะมีอันดับที่ในการจัดต่ำที่สุด
5. Density Information : คำนวณค่าความหนาแน่นให้กับประชากรคำตอบ ด้วยวิธี Crowding Distance
6. Selection : คัดเลือกคำตอบที่ดีเข้าสู่ Mating Pool โดยคำตอบที่มีความแข็งแรงมาก (มีอันดับที่น้อยกว่า) และมีค่าความหนาแน่นมาก (ขจัดปัญหาการเกาะกลุ่มของคำตอบ) จะมีโอกาสในการถูกเลือกสูง
7. Crossover : ทำการจับคู่คำตอบที่อยู่ใน Mating Pool และทำการครอสโอเวอร์ด้วยความน่าจะเป็นในการครอสโอเวอร์เท่ากับ P_C
8. Mutation : ทำการมิวเทชั่นสตริงคำตอบด้วยความน่าจะเป็นในการทำมิวเทชั่นเท่ากับ P_m
9. Combination population : รวมประชากรคำตอบรุ่นพ่อแม่ที่ได้รับการปรับปรุงคำตอบด้วยฮิวริสติกแบบการค้นหาเฉพาะที่ และประชากรคำตอบรุ่นลูกที่ได้รับการปรับปรุงจากการค้นหาเฉพาะที่เช่นเดียวกัน
10. Selection next population : คัดเลือกประชากรคำตอบสำหรับเจนเนอเรชั่นถัดไปจากการรวมประชากรคำตอบในขั้นตอนที่ 11 โดยใช้หลักการ Non-dominated Sorting และ Crowding Distance ประชากรคำตอบที่มีอันดับหนึ่ง

จะมีโอกาสได้รับเลือกไปเป็นประชากรคำตอบในเจอเนอเรชั่นถัดไปสูงเป็นอันดับแรก และมีโอกาสลดหลั่นลงมาตามอันดับที่ ถ้าจำนวนประชากรคำตอบในอันดับใดมีจำนวนน้อยกว่าจำนวนประชากรคำตอบที่เหลืออยู่ จะคัดเลือกประชากรคำตอบโดยการพิจารณา Crowding Distance ที่มีค่ามาก และดำเนินการในขั้นตอนนี้จนกระทั่งครบจำนวน *popsize* ตัว

11. Strategies to maintain elitist solution in the population : เก็บกลุ่มคำตอบที่ดีที่สุดหลังจากขั้นตอนที่ 9 และ 10 ซึ่งจะทำการปรับปรุง(Update) ในทุก ๆ เจนเนอเรชั่นเพื่อเปรียบเทียบประชากรคำตอบรุ่นพ่อแม่ และประชากรคำตอบรุ่นลูก และเก็บคำตอบที่เป็น Non-dominated Solution แทนที่คำตอบที่ดีที่สุดตัวเดิมแล้วนำประชากรคำตอบนั้นไปเป็นคำตอบรุ่นพ่อแม่ในเจนเนอเรชั่นถัดไป จำนวน *popsize* ตัว
12. Stopping Criteria : ดูว่าการคำนวณนั้นครบจำนวนสูงสุดของคำตอบที่ต้องหาและค่าฟังก์ชันหรือจำนวนเจนเนอเรชั่นที่กำหนดไว้หรือไม่ ถ้าน้อยกว่าให้กลับไปทำข้อ 2-11 ใหม่ และถ้าไม่ใช่ให้ทำการในข้อที่ 13
13. Stop : หยุดกระบวนการค้นหาคำตอบ และนำคำตอบใน Strategies to maintain elitist solution in the population มาเป็นกลุ่มคำตอบที่ดีที่สุด

4.10 วิธีการของอัลกอริทึมเจนเนติกอัลกอริทึมแบบ NSGA-II

4.10.1 การใส่รหัสคำตอบ (Chromosome Representation / Coding)

ขั้นตอนแรกของเจนเนติกอัลกอริทึม คือการกำหนดรูปแบบของการใส่รหัสคำตอบ ซึ่งถือว่าเป็นขั้นตอนที่สำคัญและมีผลอย่างมากต่อขั้นตอนอื่นๆของ GAs การใส่รหัสคำตอบ คือ การเปลี่ยนคำตอบของปัญหาให้อยู่ในรูปของสตริงคำตอบ (หรือที่เรียกว่า Chromosome) วิธีการใส่รหัสคำตอบมีทั้งแบบ Binary String และ Non-binary String ในกรณีของปัญหาการจัดสมดุลของสายงานการประกอบแบบผลิตภัณฑ์ผสมลักษณะตัวยู คำตอบของปัญหาคือกลุ่มของงานที่ถูกมอบหมายให้กับสถานีทำงานสถานีต่างๆ หรือเป็นการกำหนดสิทธิในการเลือกงาน (Priority) ดังนั้น วิธีการใส่รหัสคำตอบที่ใช้จึงต้องสามารถแสดงลำดับของงานในรูปของสตริงได้ วิธีการใส่รหัสคำตอบที่ใช้จึงควรเป็นแบบ Non-binary String

ในปัญหาการจัดสมดุลของสายงานการประกอบแบบผลิตภัณฑ์ผสมแบบตัวยู งานต่างๆจะตกอยู่ภายใต้ข้อจำกัดของความสัมพันธ์ตามลำดับก่อนหลังของงาน จึงไม่สามารถกำหนดงานให้กับสถานีทำงานใดๆได้อย่างอิสระ ดังนั้นการจัดสรรงานต่าง ๆ ควรจะถูกนำมาจัดลำดับ(Sequence) โดยพิจารณาจากความสัมพันธ์ก่อนหลังของงานและทำการเลือกงานจากความสัมพันธ์ก่อนหลังจากค่าสิทธิในการเลือกงาน (Priority) เสียก่อนแล้วจึงค่อยนำลำดับงานไปเรียงใส่ให้กับสถานีทำงานต่างๆตามลำดับ จากแนวคิดดังกล่าว ค่าสิทธิในการเลือกงานนำมาใช้เป็นรูปแบบของการใส่รหัสคำตอบ สตริงคำตอบที่ได้คือค่าสิทธิในการเลือกงานของงานทั้งหมด

ลักษณะของสตริงคำตอบมีดังนี้

1) คำตอบ 1 คำตอบ แทนด้วยสตริงคำตอบ 1 ตัวที่เรียกว่า Chromosome

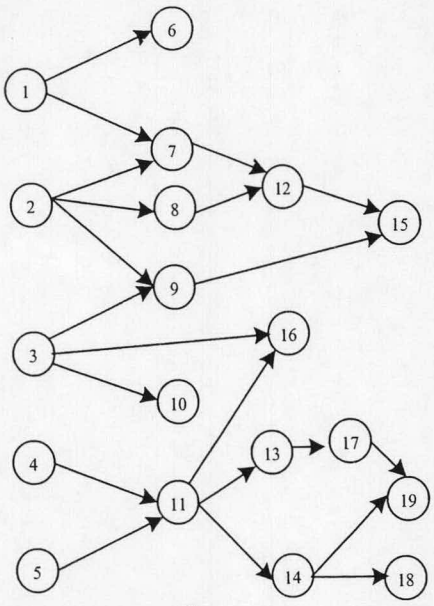
2) ใน 1 chromosome จะแบ่งเป็นหน่วยเล็กๆที่เรียกว่า bit เรียงกันอยู่ จำนวนของ bit จะเท่ากับค่าสิทธิในการเลือกขั้นงานทั้งหมดที่จะนำไปใช้ในการเลือกขั้นงานทำบนสายงานการประกอบที่พิจารณา

3) ในแต่ละ bit จะมีค่าตัวเลขตั้งแต่ 1 ถึง m บรรจุอยู่ค่าหนึ่ง ค่านี้หมายถึงหมายเลขที่ใช้แทนค่าสิทธิในการเลือกขั้นงานต่างๆ

4) ตำแหน่งของ bit หมายถึงค่าสิทธิในการเลือกขั้นงานของลำดับที่ของงานนั้นๆ

5) ตัวเลขในแต่ละ bit ต้องไม่ซ้ำกัน

ตัวอย่างเช่น ปัญหาของ Thomopoulos มีงานทั้งหมด 19 งาน จำนวนชนิดของผลิตภัณฑ์ 3 ชนิด ได้แก่ A , B และ C ซึ่งมีความสัมพันธ์ของแต่ละงานดังนี้



รูปที่ 4.18 ความสัมพันธ์ของขั้นงานในปัญหาของ Thomopoulos

สตริงคำตอบ [4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3] จะได้ว่าใน 1 Chromosome มี 19 bit หมายถึง สายงานการประกอบยูที่พิจารณามี 19 ขั้นงาน งานแรกที่จะนำไปจัดให้กับสถานีคืองานในตำแหน่งแรก จะเกิดจากการพิจารณาจากความสัมพันธ์ก่อนหลังของงานและสิทธิในการเลือกงาน ซึ่งจากรูปที่....แสดงให้มีงานที่จะถูกพิจารณาจัดสรรลงตำแหน่งทำงานข้างหน้า (Forward Work) คือ งานที่ 1 , 2 , 3 , 4 และ 5 มีค่าสิทธิในการเลือกงานจากสตริงคำตอบ (Priority) เท่ากับ 4.17.12,13 และ 1 ตามลำดับ และงานที่จะถูกพิจารณาจัดสรรลงตำแหน่งทำงานข้างหลัง(Backward Work) คือ งานที่ 6 , 10 , 15 , 16 , 18 และ19 มีค่าสิทธิในการเลือกงานจากสตริงคำตอบ (Priority) เท่ากับ 15,10,14,18,7 และ 3 ตามลำดับ เมื่อพิจารณาจากค่าสิทธิในการเลือกงานจากสตริงคำตอบพบว่าตำแหน่งที่ 16 หรือขั้นงานที่ 16 มีค่าสิทธิในการเลือกงานมากที่สุดมีค่าเท่ากับ 18 จึงกำหนดให้งานที่ 16 เป็นขั้นงานแรกที่จะนำไปจัดให้กับสถานี

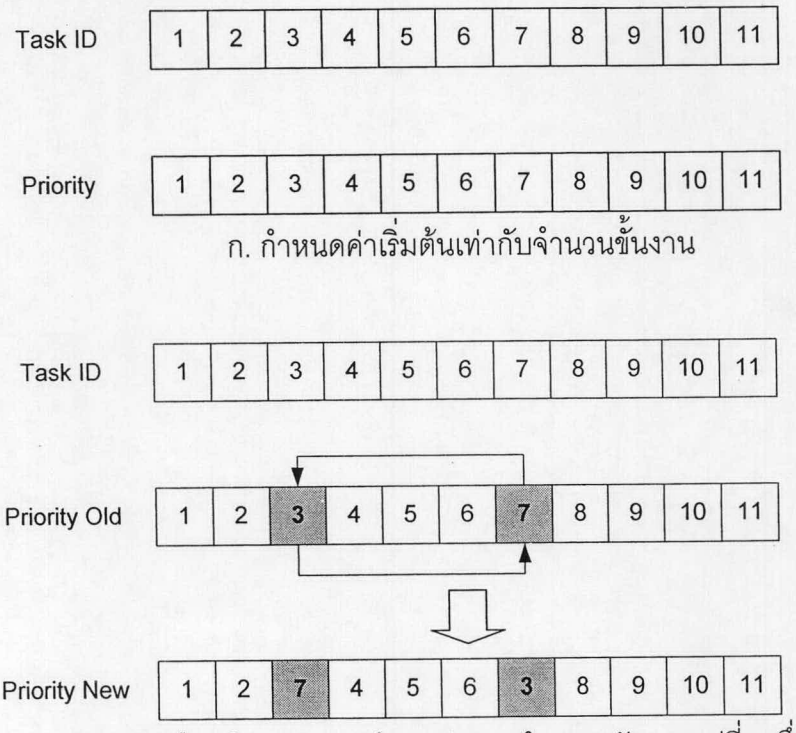
4.10.2 **การสร้างกลุ่มประชากรเบื้องต้น** (Initial Population Creating)

การสร้างกลุ่มประชากรเบื้องต้น คือ การสร้างคำตอบเบื้องต้นขึ้นมาจำนวนหนึ่งเพื่อนำไปใช้ในกระบวนการของ NSGAII โดยคำตอบ 1 คำตอบคือประชากร 1 ตัว จำนวนของประชากรที่ต้องการสร้างนั้นเป็นพารามิเตอร์ตัวหนึ่งที่ต้องมีการกำหนด ซึ่งในที่นี้กำหนดให้เท่ากับ popsize ตัว

สำหรับปัญหาการจัดสมดุลของสายการประกอบแบบผลิตภัณฑ์ผสมลักษณะตัวยูนั้นประชากร 1 ตัว หมายถึงค่าสิทธิของขั้นงานในแต่ละขั้นงานที่จะถูกเลือกลงทำบนสายงานการประกอบนั้น ดังนั้นการสร้างประชากร 1 ตัวจึงทำได้โดยการใส่ตัวเลขตั้งแต่ 1 ถึง m (m คือจำนวนงานทั้งหมด) ลงไปในแต่ละ bit ของสตริงคำตอบจนครบทุก bit และทำเช่นนี้ไปเรื่อยๆจนกว่าจะได้ประชากรทั้งหมด popsize ตัว

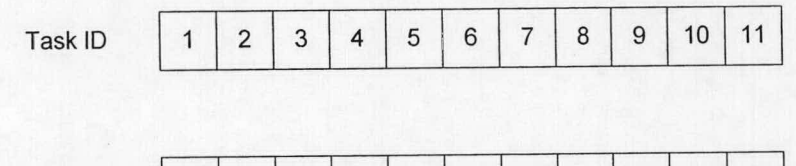
ตามหลักการของ GAs การสร้างประชากรเบื้องต้นมักใช้วิธีการแบบสุ่ม ซึ่งหมายความว่าตัวเลขที่นำมาใส่ในแต่ละ bit จะต้องถูกเลือกแบบสุ่ม แต่อย่างไรก็ตาม ในกรณีของปัญหาการจัดสมดุลของสายงานการประกอบซึ่งมีข้อจำกัดด้านความสัมพันธ์ตามลำดับก่อนหลังของงาน การสร้างประชากรแบบสุ่มอาจทำให้เกิดคำตอบที่ขัดกับข้อจำกัดดังกล่าว หรือที่เรียกว่าคำตอบที่เป็นไปไม่ได้ (Infeasible Solution) ดังนั้นในการสร้างคำตอบเบื้องต้นสำหรับปัญหาการจัดสมดุลสายการประกอบผลิตภัณฑ์ผสมที่มีลักษณะตัวยู นี้จึงต้องใช้วิธีสุ่มค่าสิทธิในการเลือกงาน(Priority) โดยจะมีขั้นตอนวิธีแบบสุ่มดังนี้

- ใส่ค่าสิทธิในการเลือกงาน (Input the priority number) โดยเริ่มแรกให้มีค่าเท่ากับ ดังรูป 4.19ก
- สุ่มตำแหน่ง 2 จุด เพื่อทำการสลับ จำนวนครั้งในการทำการสลับตำแหน่งเท่ากับจำนวนครึ่งนึงของงานหรือ $m/2$ กำหนดให้ m ขั้นงานทั้งหมด ดังรูป 4.18ข
- นำค่าที่ได้จากการสลับตำแหน่งมาใส่ในสตริง



ก. กำหนดค่าเริ่มต้นเท่ากับจำนวนขั้นงาน

ข. สุ่มเลือกตำแหน่งสองตำแหน่งและทำการสลับแลกเปลี่ยนซึ่งกัน



ค. สตริงคำตอบเบื้องต้น

รูปที่ 4.19 การสร้างประชากรเบื้องต้น

วิธีนี้จะช่วยรับประกันได้ว่าคำตอบเบื้องต้นที่สร้างขึ้นมาทั้งหมดจะเป็นคำตอบที่ไม่ขัดกับข้อจำกัดด้านความสัมพันธ์ก่อนหลังของขั้นงานดังกล่าว หรือที่เรียกว่า คำตอบที่เป็นไปได้ (Feasible Solution)

4.10.2.1 การสร้างเมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงาน (Precedence Metrix)

การสุ่มโดยพิจารณาความสัมพันธ์ตามลำดับก่อนหลังของงานร่วมด้วย จำเป็นต้องอาศัยเครื่องมือช่วยที่เรียกว่า เมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงาน (Precedence Matrix) ซึ่งแสดงตัวอย่างไว้ในตารางที่ 4.5 เมตริกซ์นี้จะช่วยบอกให้เรารู้ว่างานใดต้องทำก่อนหรือหลังงานใด อีกทั้งยังช่วยบอกว่างานนั้นๆมีงานก่อนหน้า หรืองานที่ต้องทำตามหลังอีกกี่งาน

ตารางที่ 4.5 ตัวอย่างเมตริกซ์แสดงความสัมพันธ์ตามลำดับก่อนหลังของงานในการทำงานข้างหน้า (Forward Work)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ลักษณะของตารางจะเป็นเมตริกซ์ ขนาด m x m ซึ่ง m หมายถึงจำนวนขั้นงานทั้งหมด หมายเลขของแถว (Row) หมายถึงงานที่ทำก่อน และหมายเลขของหลัก (Column) หมายถึงงานที่ต้องทำทีหลัง ตำแหน่งแถวและคอลัมน์ที่ 1 ถึง m จะประกอบไปด้วยตัวเลข 0 และ 1 ซึ่ง 0 จะหมายถึง งานที่ไม่มีความสัมพันธ์ก่อนหลังระหว่างกัน ส่วน 1 จะหมายถึง งานที่มีความสัมพันธ์ก่อนหลังระหว่างกัน เช่น จากรูปที่ 4.18 มีจำนวนขั้นงานทั้งหมด 19 ขั้นงาน สร้างเป็นเมตริกซ์ขนาด 19x19

ที่แถวที่ 1 คอลัมน์ที่ 6 มีค่าเป็น 1 หมายความว่า งานที่ 1 ต้องทำก่อนงานที่ 6

ที่แถวที่ 1 คอลัมน์ที่ 2 มีค่าเป็น 0 หมายความว่างานที่ 1 และงานที่ 2 ไม่มีความสัมพันธ์ก่อนหลังกัน

นอกจากนี้ ถ้าต้องการดูว่างานใดตามหลังงานที่เราสนใจ ก็ให้ดูที่แถวของงานนั้นและถ้าต้องการดูว่ามีงานใดต้องทำก่อนงานที่เราสนใจ ก็ให้ดูที่คอลัมน์ของงานนั้น ตัวอย่างเช่น ถ้าต้องการรู้ว่ามีงานใดตามหลังงานที่ 11 ก็ดูที่แถวที่ 11 จะได้ว่างานที่ตามหลัง คือ 13 14 และ 16 หรือถ้าต้องการรู้ว่ามีงานใดต้องทำก่อนหน้างานที่ 11 ก็ดูที่คอลัมน์ที่ 11 จะได้ว่างานก่อนหน้างานที่ 11 คืองานที่ 4 และ 5

ในกรณีเมตริกซ์แสดงความสัมพันธ์ตามลำดับก่อนหลัง ของงานในการทำงานข้างหลัง (Backward Work) จะมีลักษณะของตารางจะเป็นเมตริกซ์ ขนาด m x m ซึ่ง m หมายถึงจำนวนขั้นงานทั้งหมด หมายเลขของแถว (Row) หมายถึงงานที่ทำทีหลัง และหมายเลขของหลัก (Column) หมายถึงงานที่ต้องทำก่อน ตำแหน่งแถวและคอลัมน์ที่ 1ถึง m จะประกอบไปด้วยตัวเลข 0 และ 1 ซึ่ง 0 จะหมายถึง งานที่ไม่มีความสัมพันธ์ก่อนหลังระหว่างกัน ส่วน 1 จะหมายถึง งานที่มีความสัมพันธ์ก่อนหลังระหว่างกัน เช่น จากตารางที่ 4.6 มีจำนวนขั้นงานทั้งหมด 19 ขั้นงาน สร้างเป็นเมตริกซ์ขนาด 19x19

ตารางที่ 4.6 ตัวอย่างเมตริกซ์แสดงความสัมพันธ์ตามลำดับก่อนหลังของงานในการทำงานข้างหน้า (Backward Work)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
16	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

ที่แถวที่ 18 คอลัมน์ที่ 15 มีค่าเป็น 1 หมายความว่า งานที่ 15 ต้องทำก่อนงานที่ 18

ที่แถวที่ 18 คอลัมน์ที่ 16 มีค่าเป็น 0 หมายความว่างานที่ 18 และงานที่ 16 ไม่มีความสัมพันธ์ก่อนหลังกัน

วิธีการสร้างตารางความสัมพันธ์ตามลำดับก่อนหลังของงานของงานในการทำงานข้างหน้า (Forward Work) มีดังนี้

1. สร้างเมตริกซ์ศูนย์ขนาด m x m

2. เอาความสัมพันธ์ตามลำดับก่อนหลังของงานซึ่งเป็นข้อมูลรับเข้ามาแปลงเป็นค่า 0 หรือ 1 ที่ตำแหน่งต่างๆของเมตริกซ์ เช่น ถ้ากำหนดให้ขั้นงานที่ 6 มีงานก่อนหน้าคืองาน 2,3,4,5 ก็จะได้ว่า ในคอลัมน์ที่ 6 ให้ใส่เลข 1 ที่ แถวที่ 2, 3,4 และ 5

3. จากข้อ 1 และ 2 จะได้ตารางความสัมพันธ์ ซึ่งค่า 0-1 จะเป็นตัวบ่งชี้ถึงความสัมพันธ์ระหว่างงาน จากนั้นให้ทำการรวมค่าในแต่ละคอลัมน์ เพื่อดูว่างานต่างๆมีงานที่ต้องทำก่อนหน้าอีกกี่งาน

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

วิธีการสร้างตารางความสัมพันธ์ตามลำดับก่อนหลังของงานของงานในการทำงานข้างหลัง (Backward Work) มีดังนี้

1. สร้างเมตริกซ์ศูนย์ขนาด m x m

2. เอาความสัมพันธ์ตามลำดับก่อนหลังของงานซึ่งเป็นข้อมูลรับเข้ามาแปลงเป็นค่า 0 หรือ 1 ที่ตำแหน่งต่างๆของเมตริกซ์ เช่น ถ้ากำหนดให้ขั้นงานที่ 19 มีงานก่อนหน้าคืองาน 14 และ 17 ก็จะได้ว่า ในแถวที่ 19 ให้ใส่เลข 1 ที่ คอลัมน์ที่ 14 และ 17

3. จากข้อ 1 และ 2 จะได้ตารางความสัมพันธ์ ซึ่งค่า 0-1 จะเป็นตัวบ่งชี้ถึงความสัมพันธ์ระหว่างงาน จากนั้นให้ทำการรวมค่าในแต่ละคอลัมน์ เพื่อดูว่างานต่างๆมีงานที่ต้องทำต่อไปอีกกี่งาน

4. เมื่อได้ตารางแสดงความสัมพันธ์ตามลำดับก่อนหลังของงานแล้ว ก็สามารถสรุปได้ว่าแต่ละงานมีงานที่ต้องทำต่อไปอีกกี่งาน โดยดูจากผลรวมในแต่ละคอลัมน์ เช่น ผลรวมในแต่ละคอลัมน์เป็น 0 0 0 3 0 2 หมายความว่า งานที่ 1 2 3 5 ไม่มีงานที่ต้องทำต่อ แต่งานที่ 4 มีงานต้องทำต่อ 3 งาน และงานที่ 6 มีงานที่ต้องทำต่อ 2 งาน

หมายเหตุ การสร้างตารางความสัมพันธ์ตามลำดับก่อนหลังของงานของงานในการทำงานข้างหลัง (Backward Work) สามารถทำได้โดยการทำทรานสโพสของเมตริกซ์ (Transpose Matrix) ของตารางความสัมพันธ์ตามลำดับก่อนหลังของงานของงานในการทำงานข้างหลัง (Forward Work) จะมีค่าเท่ากัน

4.10.2.2 การสร้างคำตอบของขั้นงานจากสตริงคำตอบเบื้องต้นสำหรับปัญหา MMULB โดยใช้เมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงานทำข้างหน้าและข้างหลัง

การสร้างคำตอบของขั้นงานจากสตริงคำตอบเบื้องต้น จะใช้วิธีเลือกขั้นงานมาทำการพิจารณาจากค่าสตริงคำตอบที่มีค่าสิทธิในการเลือกงาน โดยขั้นงานที่จะมีการเลือกมาจากตารางเมตริกซ์ความสัมพันธ์ตามลำดับก่อนหลังของงานทำข้างหน้าและข้างหลัง

ขั้นตอนการสร้างสตริงคำตอบของลำดับขั้นงาน(Task Sequence) โดยละเอียด มีดังนี้

1. หางานที่ไม่มีงานก่อนหน้า (Task without Predecessor) โดยดูจากผลรวมของคอลัมน์ใน Precedence Metrix Font คอลัมน์หมายเลขใดที่ผลรวมเป็น 0 งานหมายเลขนั้นก็จะไม่มีงานก่อนหน้า และดูจากผลรวมของคอลัมน์ใน Precedence Metrix Back คอลัมน์หมายเลขใดที่ผลรวมเป็น 0 งานหมายเลขนั้นก็จะไม่มีงานใดทำงานต่อ

2. นำงานที่ได้ที่ได้จาก Precedence Metrix Font และ Precedence Metrix Back มาพิจารณาจากลำดับการเลือกงานโดยสตริงคำตอบค่าสิทธิในการเลือกงาน เช่น สตริงคำตอบค่าสิทธิในการเลือกงาน [4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3] จะได้ว่าใน 1 Chromosome มี 19 bit หมายถึง สายงานการประกอบยูที่พิจารณามี 19 ขั้นงาน งานแรกที่จะนำไปจัดให้กับสถานีคืองานในตำแหน่งแรก จะเกิดจากการพิจารณาจากความสัมพันธ์ก่อนหลังของงานและสิทธิในการเลือกงาน ซึ่งจาก Precedence Metrix Font แสดงให้เห็นว่ามีงานที่มีค่าผลรวมในคอลัมน์ที่เป็น 0 จะถูกพิจารณาจัดสรรลงตำแหน่งทำงานข้างหน้า (Forward Work) คือ งานที่ 1 , 2 , 3 , 4 และ 5 มีค่าสิทธิในการเลือกงานจากสตริงคำตอบ (Priority) เท่ากับ 4.17.12,13 และ 1 ตามลำดับ และจาก Precedence Metrix Back งานที่มีค่าผลรวมในคอลัมน์เท่ากับ 0 จะถูกพิจารณาจัดสรรลงตำแหน่งทำงานข้างหลัง (Backward Work) คือ งานที่ 6 , 10 , 15 , 16 , 18 และ19 มีค่าสิทธิในการเลือกงานจากสตริงคำตอบ (Priority) เท่ากับ 15,10,14,18,7 และ 3 ตามลำดับ เมื่อพิจารณาจากค่าสิทธิในการเลือกงานจากสตริงคำตอบพบว่าตำแหน่งที่ 16 หรือขั้นงานที่ 16 มีค่าสิทธิในการเลือกงานมากที่สุดมีค่าเท่ากับ 18 จึงกำหนดให้งานที่ 16 เป็นขั้นงานแรกที่จะนำไปจัดให้กับสถานี

3. งานหมายเลขใดที่ถูกกำหนดลงไปแล้วให้ตัดทิ้งมี 2 กรณี

- ถ้างานที่ถูกเลือกมาจาก Precedence Matrix Font ให้ตัดทิ้งโดยการเปลี่ยนตัวเลข ในแถว Precedence Matrix Font เป็น 0 ทั้งหมด และในคอลัมน์ของงานนั้นเท่ากับ 1 ทั้งหมด และทำให้คอลัมน์งานนั้นใน Precedence Matrix Back มีค่าเท่ากับ 1 ทั้งหมด
- ถ้างานที่ถูกเลือกมาจาก Precedence Matrix Backให้ตัดทิ้งโดยการเปลี่ยนตัวเลข ในแถว Precedence Matrix Back เป็น 0 ทั้งหมด และในคอลัมน์ของงานนั้นเท่ากับ 1 ทั้งหมด และทำให้คอลัมน์งานนั้นใน Precedence Matrix Font มีค่าเท่ากับ 1 ทั้งหมด

4. หาผลรวมในแต่ละคอลัมน์ใน Precedence Matrix Font และ Precedence Matrix Back ใหม่อีกครั้ง

5. ทำซ้ำข้อที่ 1 ถึง 4 เพื่อกำหนดงานลงไปใน bit ถัดๆไป จนกระทั่งงานทุกงานถูกกำหนดลงไปในสตริงคำตอบของลำดับขั้นงาน(Task Sequence) จนหมด

4.10.2.3 **จำนวนประชากรเบื้องต้น**

จากขั้นตอนการสร้างประชากรเบื้องต้นในข้อ 4.3.2.2 เป็นการสร้างประชากรเพียง 1 ตัวเท่านั้น แต่ในวิธีการของ NSGAII จำเป็นที่จะต้องมีจำนวนประชากรมากกว่า 1 ตัว เพื่อให้สามารถดำเนินการตามวิธีการของ NSGAII ในขั้นต่อๆไปได้ จำนวนประชากรเบื้องต้นจะเท่ากับจำนวนประชากรในแต่ละเจนเนอเรชั่น และเป็นพารามิเตอร์ตัวหนึ่งที่มีผลต่อประสิทธิภาพของ NSGAII การกำหนดจำนวนประชากรเบื้องต้นที่เหมาะสมจะได้กล่าวในบทต่อไป แต่ในที่นี้ให้ใช้จำนวนประชากรเท่ากับ popsize ตัว

การสร้างประชากรเบื้องต้นให้ได้ครบ popsize ตัว สามารถทำได้โดยทำตามขั้นตอนที่ 1 ถึง 5 ในข้อ 4.10.2.2 จนครบ popsize ครั้ง ในแต่ละครั้งที่เริ่มสร้างประชากรตัวใหม่ Precedence Matrix Font และ Precedence Matrix Back จะต้องถูกเปลี่ยนให้กลับสู่ Precedence Matrix Font และ Precedence Matrix Back ดั้งเดิมเหมือนในข้อ 4.10.2.1 เสียก่อน

4.10.3 การถอดรหัสคำตอบ (Decoding)

คำตอบที่ปรากฏอยู่ในประชากรหรือสตริงคำตอบลำดับงานที่สร้างขึ้นยังเป็นคำตอบที่ไม่สมบูรณ์ กล่าวคือเป็นเพียงลำดับของงานที่จะต้องนำไปจัดให้กับสถานีตามลำดับเท่านั้น ดังนั้นจึงต้องมีการนำงานตามลำดับที่ได้ในสตริงคำตอบลำดับงานไปจัดให้กับสถานีทำงานให้เรียบร้อยเสียก่อน ซึ่งเราจะเรียกขั้นตอนนี้ว่าการถอดรหัสคำตอบ (Decoding) แต่อย่างไรก็ตาม สตริงคำตอบลำดับงานที่เรามีสามารถบอกได้แต่เพียงว่างานที่อยู่ในลำดับแรกๆควรจะถูกจัดลงไปในสถานีทำงานต้นๆ และงานที่อยู่ในลำดับหลังก็ควรจะถูกจัดให้อยู่ในสถานีเดียวกันหรือสถานีหลังเท่านั้น ดังนั้นจึงสามารถนำงานในสตริงคำตอบลำดับงานมาจัดได้หลายแบบ เพื่อให้ได้การจัดที่ดีที่สุด สตริงคำตอบลำดับงานที่ได้จะต้องถูกถอดรหัสด้วยวิธีที่เหมาะสม

สำหรับปัญหา ULB ที่พิจารณา การถอดรหัสคำตอบทำได้ดังนี้

1. นำงานที่อยู่ในลำดับแรกในสตริงคำตอบไปจัดให้กับสถานีทำงานแรก

2. นำงานที่อยู่ในลำดับถัดไปจัดให้กับสถานีทำงานแรกเช่นกัน แล้วดูว่าเวลาทำงานรวมในสถานีเกินระยะเวลาทำงาน (Period of Time) ที่กำหนดให้หรือไม่ ถ้าเกินให้ตัดงานล่าสุดที่จัดให้สถานีทิ้งไป แล้วนำงานที่ตัดออกไปจัดให้กับสถานีถัดไป แต่ถ้าเวลารวมในสถานีน้อยกว่าระยะเวลาทำงาน ก็ให้เอางานที่อยู่ในลำดับต่อๆมาไปจัดให้กับสถานีนั้น จนกว่าเวลารวมของสถานีจะมากกว่าระยะเวลาทำงาน

3. เมื่อนำงานที่ตัดออกมาจากสถานีเดิม มาจัดให้กับสถานีถัดไปแล้ว ก็ให้นำงานลำดับต่อๆมาไปจัดให้กับสถานีทำงานนั้น จนกว่าเวลาทำงานรวมของสถานีจะเกินระยะเวลาทำงาน ถ้าเกินก็ให้ตัดงานนั้นออกจากสถานีเดิม และนำไปจัดให้กับสถานีใหม่ต่อไป

4. ทำตามข้อที่ 3 จนกว่างานทุกงานจะถูกจัดให้กับสถานีทำงานจนหมดจากการถอดรหัสคำตอบ สตริงที่มีลักษณะการจัดเรียงลำดับของงานที่ต่างกัน เมื่อนำมาผ่านกระบวนการถอดรหัสแล้วอาจให้คำตอบหรือผลการจัดงานที่เหมือนกันก็ได้ ดังตารางที่ 4.7

ตารางที่ 4.7 สตริงคำตอบลำดับงานในการจัดงานลงในสถานีงาน

TASK	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Priority	4	17	12	13	1	15	11	19	6	10	16	8	2	3	7	9	18	14	5
ลำดับงาน	16	2	8	6	15	12	4	3	7	10	18	9	1	19	17	14	13	11	5
เวลาทำงาน	0.1	1.2	0.5	0.2	1.5	0.5	0.4	0.4	0.6	0.2	0.5	0.4	0.1	0.4	0.5	0.2	0.1	0.3	0.2

ผลการจัดงานลงสถานีงาน จะมีงานที่ทำในสถานีงานดังนี้

สถานีงานที่ 1 คือ งาน 16 , 2, 8 และ 6

สถานีงานที่ 2 คือ งาน 15 และ 12

สถานีงานที่ 3 คือ งาน 4, 3, 7, 10

สถานีงานที่ 4 คือ งาน 18, 9, 1, 19, และ 17

สถานีงานที่ 5 คือ งาน 14, 13, 11, และ 5

การถอดรหัส ไม่เพียงแต่ให้คำตอบว่าแต่ละสถานีงานทำงานอะไรบ้าง แต่ยังให้ค่าของ จำนวนสถานีทำงานที่ต้องการ (จำนวนสถานีทำงานที่ได้จะเป็นจำนวนสถานีทำงานที่น้อยที่สุดอยู่แล้ว) และเวลาทำงานรวมในแต่ละสถานีด้วย

4.10.4 การประเมินค่า (Evaluation)

เมื่อได้สถานีงานแล้วจะทำการหาค่าฟังก์วัตถุประสงค์ ทั้งหมด 3 วัตถุประสงค์คือ จำนวนสถานีงานมีจำนวนน้อยที่สุด งานมีผลต่างความสัมพันธ์ในสถานีงานมีค่าน้อยที่สุดและความผันแปรของเวลาในสถานีงานทั้งหมดมีค่าน้อยที่สุด ดังนี้

กำหนดให้ m คือ สถานีงาน

SN_k คือ จำนวนการเชื่อมต่อการทำงานในสถานีงาน k

$S_{\max}$ คือ เวลารวมที่สูงที่สุดในสถานีงาน

S_k คือ เวลารวมในสถานีงาน k

4.11.4.1 เพื่อให้มีจำนวนสถานีทำงานน้อยที่สุด

$$f_1(X) = Minimum\ m \tag{4.31}$$

4.11.4.2 เพื่อให้งานมีผลต่างความสัมพันธ์ในสถานีงานมีค่าน้อยที่สุด

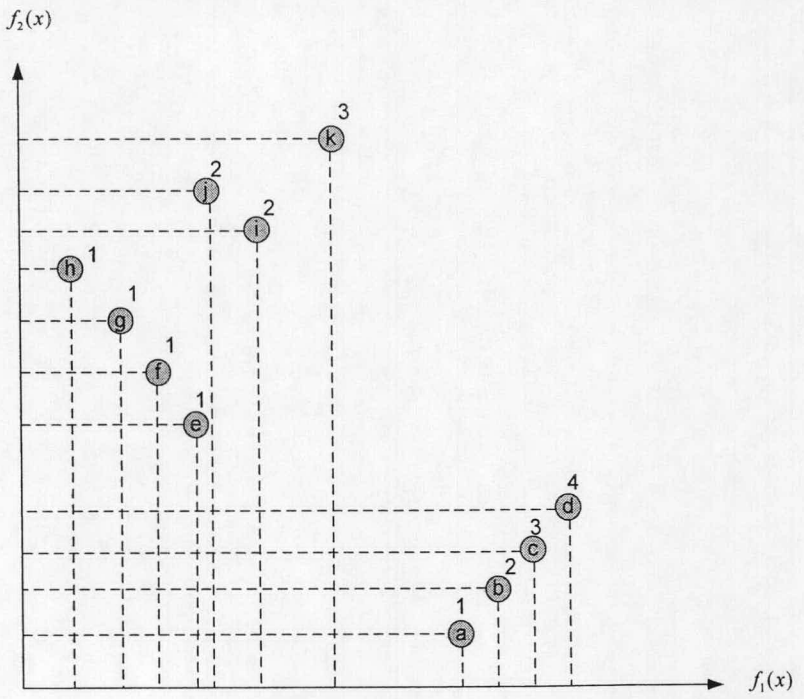
$$f_2(X) = Minimum\ \ relatedness = Minimum\ \ m - m \Big/ \sum_{k=1}^{m} SN_k \tag{4.32}$$

4.11.4.3 เพื่อหาค่าต่ำสุดของค่าการกระจายภาระงานในแต่ละสถานีงาน

$$f_3(X) = Minimum\ SI = Minimum \sqrt{\sum_{k=1}^{m} (S_{\max} - S_k)^2 \Big/ m} \tag{4.33}$$

ก่อนที่จะเข้าสู่ขั้นตอนการคัดเลือกของ NSGAII จำเป็นที่จะต้องมีการประเมินค่าประชากรแต่ละตัวเสียก่อนว่ามีความเหมาะสมมากหรือน้อยเพียงใด ความเหมาะสมนี้จะวัดจากค่าความแข็งแรงไม่แท้จริง (Dummy Fitness) ของสตริงคำตอบแต่ละตัว ตัวใดที่มีค่า Dummy Fitness น้อยก็หมายความว่ามีความเหมาะสมมากที่จะถูกมีโอกาสเลือกสูง โดยที่ค่า Dummy Fitness ดังกล่าวหมายถึง ค่าความแข็งแรงไม่แท้จริงที่เกิดจากการกำหนดอันดับของคำตอบด้วยวิธีเชิงกลุ่มที่ดีที่สุด แบบ Goldberg

การกำหนดค่าความแข็งแรงของคำตอบ ด้วยการจัดอันดับที่แบบพาเรโต ถูกนำเสนอเป็นครั้งแรกด้วย Goldberg (1989) และเรียกวิธีนี้ว่าวิธีการจัดลำดับของ Goldberg (Goldberg's Ranking) หรือ Non-dominated Sorting และเป็นเทคนิคหนึ่งในการบรรลุเป้าหมายแรกในการแก้ปัญหาการหาค่าที่เหมาะสมที่สุดที่มีหลายวัตถุประสงค์ คือการได้มาของขอบเขตกลุ่มคำตอบที่ดีที่สุด (Pareto Frontier) แนวคิดพื้นฐานของเทคนิคนี้ คือการจัดลำดับเซตของสตริงคำตอบในประชากรคำตอบทั้งหมด โดยจะพิจารณาคำตอบที่ไม่มีคำตอบใดดีกว่าเซตคำตอบนี้เป็นอันดับแรก และจัดอันดับ (Rank) เป็นอันดับที่หนึ่ง จากนั้นจะถูกตัดออกจากการพิจารณาของประชากรคำตอบทั้งหมด เซตของสตริงคำตอบที่เหลือจะถูกจัดให้เป็นอันดับต่อมา โดยที่กระบวนการหาคำตอบที่ดีที่สุดของเทคนิคนี้จะค้นหาคำตอบจนกระทั่งคำตอบในประชากรคำตอบทั้งหมดถูกจัดอันดับ แสดงได้ดังรูปที่ 4.20



รูปที่ 4.20 วิธีการจัดลำดับของ Goldberg ที่ใช้ในอัลกอริทึม NSGA-II

จากรูปที่ 4.20 จะเห็นได้ว่าสมาชิกประชากรคำตอบที่ไม่ถูกครอบงำจากคำตอบอื่น จะถูกจัดให้มีอันดับที่หนึ่ง จากนั้นจะไม่พิจารณาสมาชิกดังกล่าวชั่วคราว เพื่อกำหนดอันดับที่ ให้กับสมาชิกประชากรคำตอบที่เหลืออยู่ โดยถ้าสมาชิกประชากรคำตอบนั้นไม่ถูกครอบงำจากคำตอบอื่น จะถูกจัดอันดับที่ให้เป็นอันดับต่อมา และพิจารณาไปเรื่อย ๆ จนครบสมาชิกคำตอบทุกคำตอบ โดยเริ่มต้นจากสมาชิกตัวที่แข็งแรงที่สุด จนถึงสมาชิกตัวที่มีความอ่อนแอที่สุด

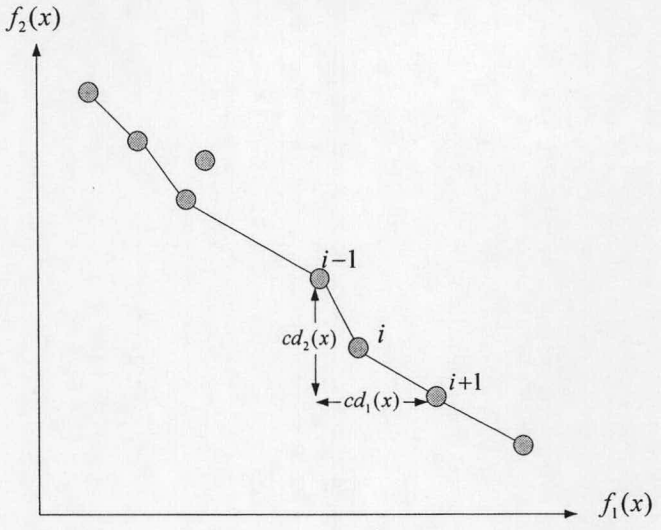
และมีการคำนวณหาค่าการแบ่งปันค่าความแข็งแรงด้วยวิธีการคำนวณแบบ Crowding Distance ซึ่งเป็นเทคนิคหนึ่งที่มีความสามารถในการทำให้ประชากรคำตอบบนขอบเขตกลุ่มคำตอบที่ดีมีลักษณะการกระจายอย่างสม่ำเสมอมากยิ่งขึ้นและนำมาใช้ในการบรรลุเป้าหมายที่สองดังกล่าวของ NSGA II โดยเทคนิคนี้จะถูกนำมาใช้คำนวณระยะทางระหว่างสมาชิกประชากรคำตอบที่อยู่ใน Front เดียวกันเท่านั้น ขั้นตอนการคำนวณ Crowding Distance แสดงได้ดังนี้

ขั้นตอนที่ 1 ถ้าสมาชิกประชากรคำตอบมีอันดับที่เท่ากันแล้วให้คำนวณขั้นตอนที่ 2 และ 3 ดังนี้

ขั้นตอนที่ 2 กำหนดให้ l แทนจำนวนประชากรคำตอบทั้งหมดใน Front ที่ $j, j = 1,..., R$ และ $x_{[i,k]}$ แทน สมาชิกประชากรคำตอบที่ i ในฟังก์ชันวัตถุประสงค์ k ที่ได้รับการเรียงลำดับฟังก์ชันวัตถุประสงค์จากน้อยไปมาก (Sort List) โดยสมาชิกประชากรคำตอบที่มีลำดับที่ 1 (ค่าฟังก์ชันวัตถุประสงค์น้อยที่สุด) และลำดับสุดท้าย (ค่าฟังก์ชันวัตถุประสงค์มากที่สุด) จะถูกกำหนดให้มี Crowding Distance เป็นค่ามาก ๆ (Infinity) นั่นคือ $cd_k(x_{[1,k]}) = \infty$ และ $cd_k(x_{[l,k]}) = \infty$ ส่วนสมาชิกประชากรคำตอบรายการเรียงลำดับที่ 2 ถึง ลำดับที่ $l-1$ จะคำนวณ Crowding Distance จาก

$$cd_k(x_{[i,k]}) = \frac{f_k(x_{[i+1,k]}) - f_k(x_{[i-1,k]})}{f_k^{\max} - f_k^{\min}} \tag{4.34}$$

ขั้นตอนที่ 3 คำนวณผลรวมของ Crowding Distance ทั้ง k ฟังก์ชันวัตถุประสงค์ จะได้ว่า $cd(x) = \sum cd_k(x)$ นั่นคือค่า Crowding Distance ของสมาชิกคำตอบนั้น ๆ โดยค่านี้จะแสดงถึงระยะห่างระหว่างจุดที่อยู่ต่อเนื่องบนคำตอบใน Front เดียวกัน ค่า Crowding Distance น้อยจะแสดงให้เห็นถึงกลุ่มคำตอบใน Front นั้นมีการเกาะกลุ่มกัน ส่วนค่า Crowding Distance มากจะแสดงให้เห็นว่ากลุ่มคำตอบใน Front นั้นมีการกระจายอย่างชัดเจน



รูปที่ 4.21 Crowding Distance ที่ใช้ในอัลกอริทึม NSGA-II

4.10.5 การคัดเลือกคำตอบ (Selection)

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

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

4.10.5.1 การสร้างวงล้อรูเล็ต

วงล้อรูเล็ต คือวงกลมที่มีพื้นที่ขนาด 1 หน่วยซึ่งพื้นที่ถูกแบ่งออกเป็นส่วนๆตามจำนวนของประชากรในแต่ละเจนเนอเรชั่น (เท่ากับ popsize ส่วน) พื้นที่แต่ละส่วนจะมีขนาดเท่ากับความน่าจะเป็นในการถูกเลือกของสตริงคำตอบแต่ละตัว วิธีการสร้างมีดังนี้

1. หาค่า Fitness รวมของสตริงคำตอบทั้งหมด popsize ตัว ดังสมการที่ (4.35)

$$F = \sum_{i=1}^{popsize} f(x_i) \tag{4.35}$$

โดยที่ $f(x_i)$ คือ ค่า Fitness ของสตริงตัวที่ i

2. หาค่าความน่าจะเป็นในการถูกคัดเลือก (Probability of Selection) ของสตริงคำตอบแต่ละคำตอบแต่ละตัว ตามสมการที่ (4.36)

$$p_i = \frac{f(x_i)}{F} \qquad i = 1, 2, ..., popsize \tag{4.36}$$

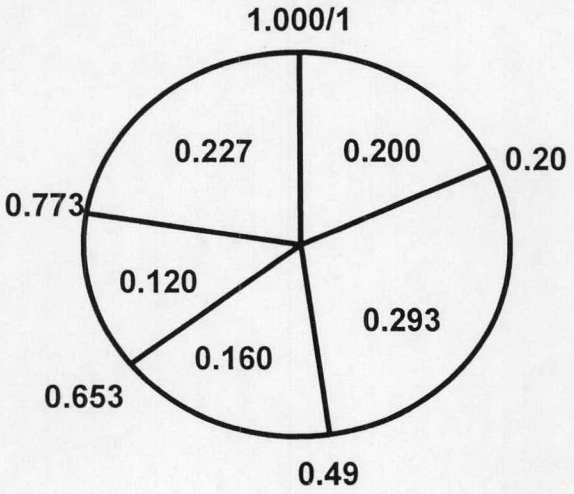
3. หาค่าความน่าจะเป็นในการถูกคัดเลือกสะสม (Cumulative Probability of Selection) ของสตริงคำตอบแต่ละตัว ตามสมการที่ (4.37)

$$q_i = \sum_{j=1}^{i} p_j \tag{4.37}$$

ตัวอย่างของวงล้อรูเล็ตแสดงได้ดังตารางที่ 4.8 และรูปที่ 4.22

ตารางที่ 4.8 ตัวอย่างตารางแสดงการสร้างวงล้อรูเล็ต

String No.	Fitness	p_i	q_i
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 4.22 วงล้อรูเล็ต

4.10.5.2 **วิธี** Tournament Selection

การคัดเลือกสตริงคำตอบโดยวิธี Roulette Wheel Selection ซึ่งใช้กันอยู่ทั่วไป จะใช้สุ่มสตริงคำตอบจากวงล้อรูเล็ต ซึ่งมีโอกาสที่จะสุ่มได้สตริงคำตอบที่มีค่า Fitness น้อยๆด้วย แต่สำหรับการคัดเลือกสตริงคำตอบโดยวิธี Tournament Selection เป็นการสุ่มสตริงคำตอบจากวงล้อรูเล็ตมา 2 ตัว แล้วนำค่า Fitness มาเปรียบเทียบกันอีกครั้งหนึ่ง สตริงคำตอบที่ถูกเลือกจึงเป็นตัวที่มีความเหมาะสมมากกว่า สำหรับขั้นตอนการเลือกมีดังนี้

1. สร้างตัวเลขสุ่ม r ซึ่งมีค่าระหว่าง 0 ถึง 1 ขึ้นมา 1 ค่า คือ r_1

2. ถ้า $r_1 < q_i$ 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 $r_2 < q_i$ ให้เลือกสตริงคำตอบตัวแรก แต่ถ้า $q_{i-1} < r_2 < q_i$ (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 มากก็จะมีพื้นที่มาก จึงมีโอกาสที่ตัวเลขสุ่มที่สร้างขึ้นมาจะตกอยู่ภายในบริเวณของสตริงคำตอบตัวนั้นมากกว่าตัวที่มีค่า Fitnessน้อย (มีพื้นที่น้อย) ทำให้สตริงคำตอบที่ถูกเลือกเข้าสู่ Mating Pool เป็นสตริงคำตอบที่มีค่า Fitness โดยเฉลี่ยสูงกว่าสตริงคำตอบเดิม

ตารางที่ 4.9 ตัวอย่างการคัดเลือกด้วยวิธี Tournament Selection

ครั้งที่	ประชากรตัวที่ 1				ประชากรตัวที่ 2				หมายเลขประชากรที่เลือก
	r_1	$r_1 < q_i$	หมายเลขประชากร	ค่า Fitness	r_2	$r_2 < q_i$	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	0.733	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

ตารางที่ 4.9 แสดงตัวอย่างการคัดเลือกด้วยวิธี Tournament Selection ซึ่งจะเห็นได้ว่าสตริงคำตอบหมายเลข 2 ซึ่งมีค่า Fitness มากที่สุด จะถูกสุ่มเลือกขึ้นมามากครั้งที่สุดในขณะที่สตริงคำตอบซึ่งมีค่า Fitness น้อยก็จะถูกสุ่มเลือกน้อยครั้งเช่นกัน ข้อสังเกตประการหนึ่งจากตัวอย่างก็คือ ในการสุ่ม ก็สุ่มได้สตริงคำตอบหมายเลข 4 ที่มีค่า Fitness น้อยที่สุดด้วย ซึ่งถ้าใช้วิธี Roulette Wheel สตริงคำตอบหมายเลข 4 นี้ก็จะมีโอกาสที่จะถูกเลือกเข้าสู่ Mating Pool และจะได้รับการดำเนินการตามกระบวนการ GAs ต่อไป แม้ว่าสตริงคำตอบตัวนี้จะมีความเหมาะสมต่ำก็ตาม แต่เมื่อใช้วิธีคัดเลือกแบบ Tournament Selection สตริงคำตอบหมายเลข 4 นี้จะต้องถูกนำไปเปรียบเทียบกับสตริงคำตอบอีกตัวก่อน ดังนั้นโอกาสที่สตริงคำตอบตัวนี้จะถูกเลือกเข้าสู่ Mating Pool ก็จะลดลง

4.10.6 การครอสโอเวอร์ (Crossover)

4.10.6.1 การจับคู่สตริงคำตอบ

จากสตริงคำตอบจำนวน popsize ตัวที่ได้มาจากกระบวนการคัดเลือกจะมีสตริงคำตอบเพียงบางส่วนเท่านั้นที่จะถูกนำมาจับคู่เพื่อเตรียมสำหรับกระบวนการครอสโอเวอร์สตริงคำตอบที่ไม่ได้ถูกนำไปจับคู่ก็จะยังคงสภาพเดิมและอยู่ใน Mating Pool (เป็นประชากร

ในเจนเนอเรชั่น) 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

4.10.6.2 **การครอสโอเวอร์**

สตริงคำตอบที่เตรียมไว้ $Nc/2$ คู่จะถูกนำมาผ่านกระบวนการครอสโอเวอร์ ซึ่งเป็นกระบวนการที่นำสตริงคำตอบที่ถูกจับคู่ไว้มาแลกเปลี่ยนส่วนซึ่งกันและกันเพื่อให้เกิดสตริงใหม่ขึ้นโดยจะเรียกสตริงคำตอบ 2 ตัวที่ถูกจับคู่นี้ว่า “สตริงคำตอบรุ่นพ่อแม่ (Parent)” และจะเรียกสตริงคำตอบ 2 ตัวที่ได้จากการครอสโอเวอร์นี้ว่า “สตริงคำตอบรุ่นลูก (Offspring)” วิธีการครอสโอเวอร์มีหลายวิธี ในที่นี้เสนอวิธีการครอสโอเวอร์ คือ

วิธี Weight mapping crossover (WMX)

วิธีการครอสโอเวอร์แบบ WMX ขั้นแรกจะทำการเลือกคู่สตริงพ่อแม่ขึ้นมาอย่างสุ่ม จากนั้นทำการเลือกตำแหน่งที่จะทำการครอสโอเวอร์อย่างสุ่มที่อยู่ในช่วง [1, m]

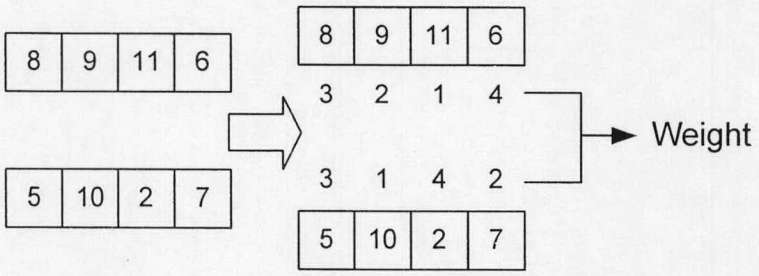
โดยที่ m คือความยาวของสตริง และยีนตัวแรกคือยีนหมายเลข 1 และยีนตัวสุดท้ายคือยีนหมายเลข m ขอบเขตของการครอสโอเวอร์อยู่ในช่วงเครื่องหมาย " | "

p1 = [7 2 10 4 5 | 8 9 11 6 | 1 3]

p2 = [3 6 1 9 8 | 5 10 2 7 | 11 4]

ในขั้นตอนต่อไปจะทำการสลับค่าระหว่างสตริงที่อยู่ในช่วง " | " นั่นคือตำแหน่งสุ่มอยู่ในช่วง [6,9] ของโครโมโซมลูกหลานทั้งสอง นำค่าที่อยู่ในช่วง " | " ทั้งพ่อแม่มากำหนดค่าน้ำหนัก (Weight) ของยีน ซึ่งยีนตำแหน่งไหนมีหมายเลขยีนที่สูงสุดจะกำหนดให้มีค่าน้ำหนัก (Weight) เท่ากับ 1 และหมายเลขถัดไปจะมีค่าน้ำหนัก (Weight) เท่ากับ 2

จะเห็นได้ว่าสตริงคำตอบที่ได้จากการสุ่มช่วงจากพ่อแม่มีค่า [8,9,11,6] และ [5,10,2,7] เมื่อทำการกำหนดค่าน้ำหนัก (Weight) ใน [8,9,11,6] ตำแหน่งยีนที่ 3 มีค่าสูงสุดเท่ากับ 11 จึงให้ค่าน้ำหนักเท่ากับ 1 ลองลงมาตำแหน่งที่ 2 มีหมายเลขเท่ากับ 9 จึงให้ค่าน้ำหนักมีค่าเท่ากับ 2 เป็นต้น ดังรูปที่ 4.23

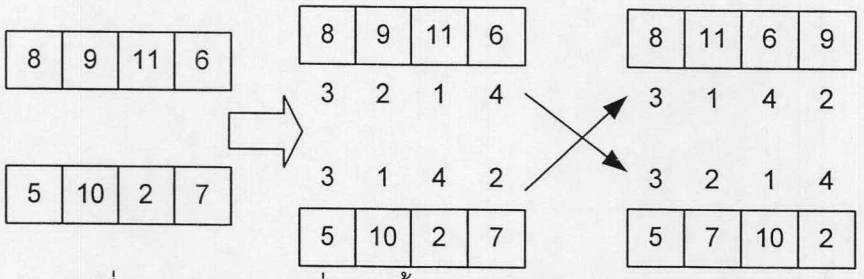


รูปที่ 4.23 การกำหนดค่าน้ำหนักให้แก่สตริงคำตอบพ่อและแม่

เมื่อได้มีการกำหนดค่าน้ำหนักเรียบร้อย จะทำการสลับคู่ของค่าน้ำหนักเพื่อทำการแลกเปลี่ยนโครโมโซมกัน โดยค่าน้ำหนักของพ่อจะไปอยู่ที่แม่ และค่าหน้ำหนักของแม่จะย้ายมาอยู่กับพ่อ หลังการสลับค่าน้ำหนักจะทำการปรับเปลี่ยนค่าในโครโมโซมให้มีค่าตรงตามค่าน้ำหนัก ซึ่งจะใช้หลักการเดิมคือ ค่าน้ำหนักที่มีค่าเท่ากับ 1 จะมีค่าหมายเลขในโครโมโซมที่สูงและค่าน้ำหนักที่ 2 จะมีค่าหมายเลขในโครโมโซมที่สูงถัดไป

จากตัวอย่างสตริงคำตอบของพ่อจะมีค่าน้ำหนักจาก [3,2,1,4] เปลี่ยนเป็น [3,1,4,2] ทำให้ค่าโครโมโซมที่อยู่ในช่วง[6,9] มีค่าที่เป็นจากเดิม[8,9,11,6]

กลายเป็น [8,11,6,4] เนื่องจากค่าน้ำหนักของ 1 หลังการสลับเปลี่ยน อยู่ในตำแหน่งที่ 2 ทำให้หมายเลยที่สูงที่สุดคือ 11 ต้องทำการย้ายจากตำแหน่งที่ 3 มาอยู่ที่ตำแหน่งที่ 2 ตามรูปที่ 4.24



รูปที่ 4.24 การแลกเปลี่ยนค่าน้ำหนักของสตริงคำตอบของพ่อแม่

ขั้นตอนสุดท้ายหลังจากทำการแปลงค่าน้ำหนักหลังการสลับเปลี่ยนแล้ว สตริงที่ได้จะทำการแทนค่ากลับในช่วงที่ทำการสุ่มมา จะได้สตริงคำตอบในรุ่นลูก

p1 = [7 2 10 4 5 | 8 9 11 6 | 1 3]

p2 = [3 6 1 9 8 | 5 10 2 7 | 11 4]

จะได้

q1 = [7 2 10 4 5 | 8 11 6 9 | 1 3]

q2 = [3 6 1 9 8 | 5 7 10 2 | 11 4]

4.10.6.3 **การซ่อมแซมคำตอบ** (Repair Method)

เนื่องจากสตริงคำตอบของงานวิจัยครั้งนี้เป็นสตริงค่าของสิทธิในการเลือกงาน ดังนั้นตัวเลขในแต่ละ bit ไม่มีความสัมพันธ์กันทำให้ไม่ผิดข้อจัดความสัมพันธ์ของขั้นงาน ทำให้ไม่ต้องมีการซ่อมแซมคำตอบ คำตอบของสตริงคำตอบสามารถนำมาใช้ในการคำนวณได้เลย

เมื่อทำการครอสโอเวอร์และซ่อมแซมคำตอบเสร็จแล้ว สตริงคำตอบรุ่นลูกที่เป็นคำตอบที่เป็นไปได้จะถูกส่งกลับเข้าสู่ Mating Pool เพื่อไปรวมกับสตริงคำตอบรุ่นพ่อแม่ที่ไม่ได้ถูกเลือกมาครอสโอเวอร์ จากนั้นก็จะเข้าสู่กระบวนการของ NSGAII ลำดับถัดไป

4.10.7 **การมิวเตชั่น** (Mutation)

คือการสลับตำแหน่งของค่าภายในสตริงคำตอบตัวเดียว วิธีการมิวเตชั่นมีหลายวิธีแต่ในกรณีของปัญหาการจัดสมดุลของสายการประกอบแบบผลิตภัณฑ์ผสม จะใช้วิธีการมิวเตชั่นแบบ Reciprocal Exchange Mutation ซึ่งสตริงที่ได้จากการมิวเตชั่นแบบนี้จะไม่ขัดกับหลักความสัมพันธ์ตามลำดับก่อนหลังของงานเนื่องจากสตริงคำตอบที่นำมาทำมิวเตชั่นคือสตริงคำตอบค่าสิทธิในการเลือกงาน (String Priority)

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

วิธี Reciprocal Exchange Mutation

เมื่อได้สตริงตัวที่จะทำการมิวเตชั่นแล้ว ให้ทำการสุ่มตำแหน่งที่จะทำการมิวเตชั่น (Mp) ขึ้นมา 2 ตำแหน่ง ซึ่งเป็นค่าระหว่าง [1, m] โดยที่ m คือความยาวของสตริงคำตอบตำแหน่งที่จะทำการมิวเตชั่น โดยเริ่มจากการทำการสุ่มตัว 2 ตัวที่ไม่ซ้ำกัน สมมติสุ่มได้เลข 6 และ 17 จากนั้นทำการสลับตำแหน่งของตัวเลขทั้งสอง

14	11	4	9	18	17	13	10	6	8	5	16	1	2	15	19	7	12	3

14	11	4	9	18	7	13	10	6	8	5	16	1	2	15	19	17	12	3

รูปที่ 4.25 วิธี Reciprocal Exchange Mutation

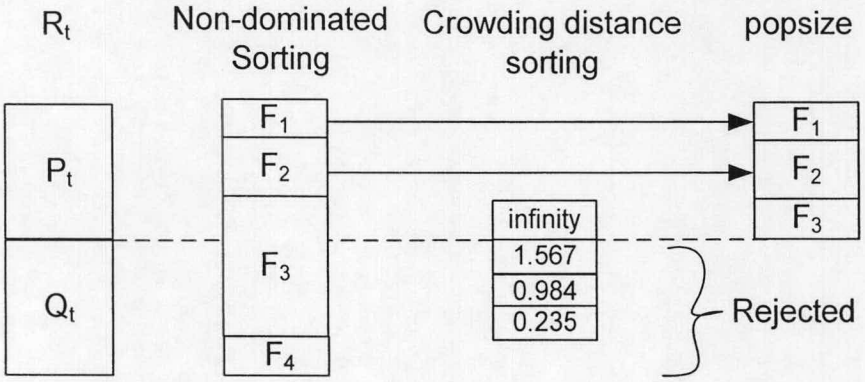
สตริงคำตอบที่ได้จากการมิวเตชั่นและสตริงคำตอบที่ไม่ได้ถูกเลือกมาทำการมิวเตชั่นจะถูกนำมารวมกัน เพื่อเตรียมเข้าสู่เจนเนอเรชั่นต่อไป

4.10.8 **เทคนิคการเก็บค่าที่ดีที่สุด** (Elite Preserve Strategy)

เทคนิคการเก็บค่าที่ดีที่สุด เป็นเทคนิคที่นำมาใช้เพื่อเก็บค่าดีที่สุดไว้ และป้องกันการสูญเสียคำตอบที่ดีไปหลังจากผ่านกระบวนการต่าง ๆ เนื่องจากสตริงคำตอบที่ได้จากการ

ค้นหาเฉพาะที่ การครอสโอเวอร์และมิวเทชั่น อาจทำให้เกิดคำตอบที่แย่กว่าคำตอบที่เคยปรากฏในเจนเนอเรชั่นที่ผ่าน ๆ มา ดังนั้นจึงต้องมีการเก็บค่าที่ดีที่สุดเอาไว้ เพื่อใช้เปรียบเทียบกับค่าที่ดีที่สุดของกลุ่มสตริงคำตอบชุดใหม่ โดยพิจารณาค่าความแข็งแรงที่ดีกว่า (เนื่องจากในปัญหาที่ทำการพิจารณาในงานวิจัยนี้เป็นการหาค่าต่ำสุดดังนั้นสตริงคำตอบที่ให้ค่าต่ำ จะเป็นสตริงคำตอบที่ดีกว่า) จะได้รับการแทนที่สตริงคำตอบที่ให้ค่าแย่และคัดเลือกคำตอบนั้นออกไป เทคนิคการเก็บค่าที่ดีที่สุดที่นำมาใช้ในอัลกอริทึม M-NSGA กับ NSGAII มีลักษณะเดียวกันคือ

กลยุทธ์ในการรักษาประชากรคำตอบ (Strategies to maintain elitist solutions in the population) เทคนิคการเก็บค่าที่ดีที่สุดนี้จะทำการเก็บค่าคำตอบที่เป็น Non-dominated Solution และในระหว่างกระบวนการค้นหาคำตอบตั้งแต่การสร้างประชากรรุ่นพ่อแม่ (P_t) การสร้างประชากรรุ่นลูกด้วยวิธีการทางพันธุกรรม (Q_t) ในแต่ละเจนเนอเรชั่น จากนั้นจะนำประชากรรุ่นพ่อแม่และรุ่นลูกมารวมกันคือสถานที่รวมคำตอบ ($R_t = P_t + Q_t$) และเก็บคำตอบที่ได้จาก Non-dominated Solution โดยให้ความสำคัญกับสตริงคำตอบที่มีอันดับต่ำและมีค่า Crowding Distance มาก และทำการปรับปรุง(Update) สตริงคำตอบใหม่ในสถานที่เก็บคำตอบด้วยการย้ายสตริงคำตอบที่ดีที่สุดตัวเดิมออกไป และเพิ่มสตริงคำตอบที่ดีที่สุดใหม่เข้ามา ดังรูปที่ 4.26



รูปที่ 4.26 วิธีการเก็บค่าที่ดีที่สุดของอัลกอริทึม NSGAII และ M-NSGAII

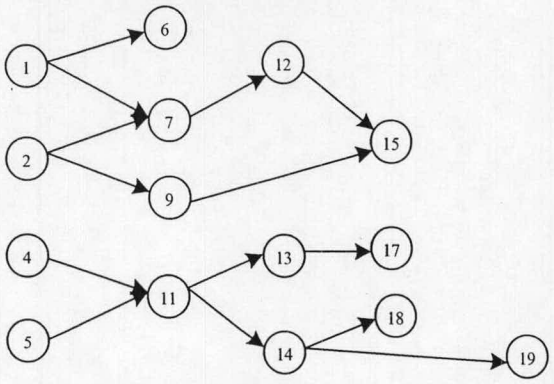
สตริงคำตอบที่ได้ภายหลังจากขั้นตอนนี้จะกลายเป็นสตริงคำตอบพ่อแม่ที่แท้จริงในเจนเนอเรชั่นต่อไป

4.11 ตัวอย่างการนำวิธี NSGAII อัลกอริทึมไปใช้ในการแก้ปัญหาสมดุลสายการประกอบลักษณะตัวยูผลิตภัณฑ์ผสม ในระบบการผลิตแบบทันเวลาพอดี

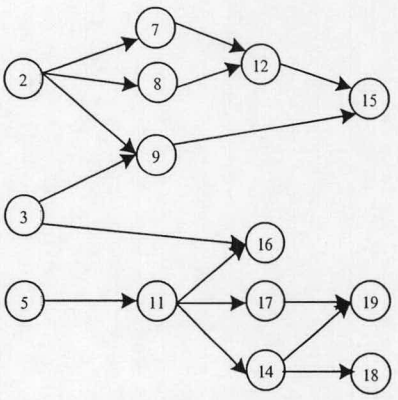
จากขั้นตอนของ NSGAII ที่ได้เสนอมาทั้งหมด สามารถนำมาทดลองใช้แก้ปัญหาตัวอย่าง ซึ่งเป็นสายการประกอบผลิตภัณฑ์ผสมของปัญหา Thomopoulos มีงานทั้งหมด 19 งาน จำนวนชนิดของผลิตภัณฑ์ 3 ชนิด ได้แก่ A, B และ C มีรอบเวลาในการทำงานในแต่ละสถานีงานเท่ากับ 2 ซึ่งมีความสัมพันธ์ของแต่ละงานดังนี้

4.11.1 การเตรียมข้อมูล (Data Input)

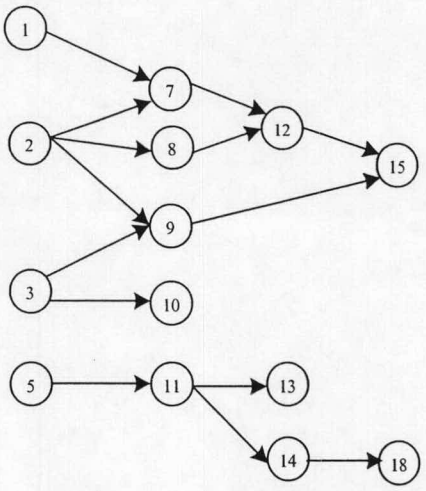
4.11.1.1 การสร้างแผนภาพความสัมพันธ์รวม (Overall Precedence Diagram) แสดงได้ดังรูปที่ 4.27



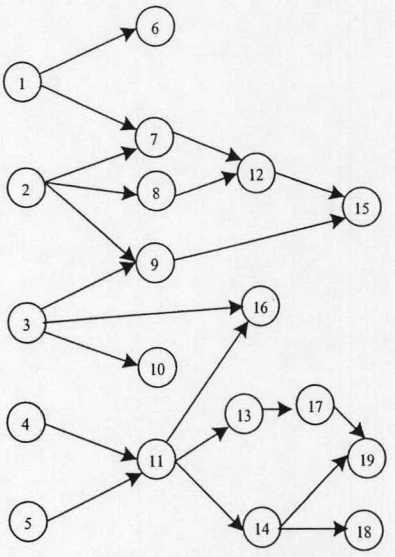
แผนภาพความสัมพันธ์ของผลิตภัณฑ์ A



แผนภาพความสัมพันธ์ของผลิตภัณฑ์ B



แผนภาพความสัมพันธ์ของผลิตภัณฑ์ C



แผนภาพความสัมพันธ์ของผลิตภัณฑ์รวม A, B และ C

รูปที่ 4.27 การสร้างแผนภาพความสัมพันธ์รวม (Overall Precedence Diagram) ของปัญหาตัวอย่างขนาด 19 ขั้นงาน

4.11.1.2 การหาเวลาทำงานเฉลี่ยในแต่ละขั้นงานซึ่งสามารถแสดงได้ดังตารางที่ 4.10

ตารางที่ 4.10 เวลาในการผลิตสินค้าชนิด A B และ C ในแต่ละขั้นงาน

Task	Model			
	A	B	C	Mean
1	0.1	0	0.1	0.0667
2	1.2	1.2	1.2	1.2
3	0	0.4	0.4	0.2667
4	0.4	0	0	0.1333
5	0.2	0.2	0.2	0.2
6	0.2	0	0	0.0667
7	0.6	0.6	0.6	0.6
8	0	0.5	0.5	0.3333
9	0.4	0.4	0.4	0.4
10	0	0	0.2	0.0667
11	0.3	0.3	0.3	0.3
12	0.5	0.5	0.5	0.5
13	0.1	0	0.1	0.0667
14	0.2	0.2	0.2	0.2
15	1.5	1	1.5	1.3333
16	0	0.1	0	0.0333
17	0.5	0.5	0	0.3333
18	0.5	0.5	0.5	0.5
19	0.4	0.4	0	0.2667

4.11.1.3 **สร้างตาราง** Precedence Matrix Font **และ** Precedence Matrix Back จาก แผนภาพความสัมพันธ์รวม รูปที่ 4.27 จะได้ดังตารางที่ 4.11 และ 4.12

ตารางที่ 4.11 ความสัมพันธ์ของขั้นงานในการทำงานข้างหน้า (Precedence Matrix Font)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ตารางที่ 4.12 ความสัมพันธ์ของขั้นงานในการทำงานข้างหลัง (Precedence Matrix Back)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
16	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

4.11.1.4 พารามิเตอร์ของ NSGAII ที่เลือกใช้คือ

- จำนวนประชากรเบื้องต้น 5 ตัว
- วิธีการครอสโอเวอร์แบบ Weight mapping crossover (WMX)
- ความน่าจะเป็นในการครอสโอเวอร์ 0.8
- ความน่าจะเป็นในการมิวเตชั่น 0.2

4.11.2 การสร้างสตริงคำตอบเบื้องต้น

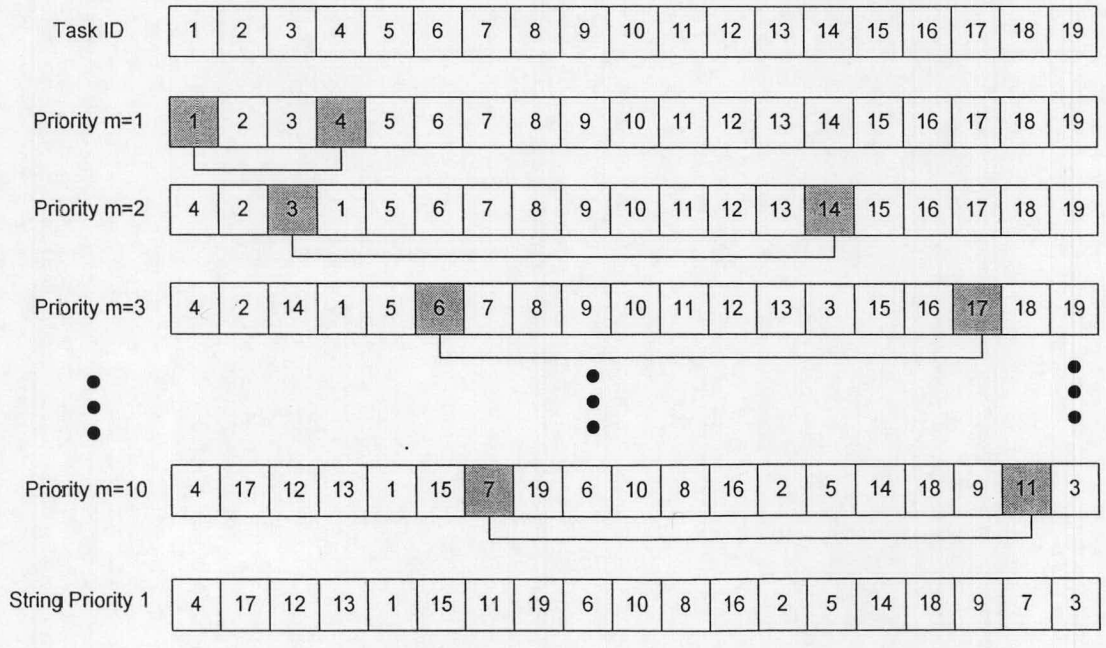
เนื่องจากงานวิจัยนี้ใช้วิธีสุ่มสตริงคำตอบโดยวิธีกำหนดค่าสิทธิในการเลือกงาน (Priority) โดยจะมีขั้นตอนวิธีแบบสุ่มดังนี้

- ใส่ค่าสิทธิในการเลือกงาน (Input the priority number) โดยเริ่มแรกให้มีค่าเท่ากับ-ขั้นงาน

Task ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Priority	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

รูปที่ 4.28 การสร้างสตริงคำตอบเริ่มต้น

- สุ่มตำแหน่ง 2 จุด เพื่อทำการสลับ จำนวนครั้งในการทำการสลับตำแหน่งเท่ากับจำนวนครึ่งนึงของงานหรือ $m/2 = 19/2 \approx 10$ กำหนดให้ m ขั้นงานทั้งหมด



รูปที่ 4.29 สตริงคำตอบค่าสิทธิในการเลือกขั้นงาน

จะได้สตริงคำตอบค่าสิทธิในการเลือกขั้นงานเท่ากับ 5 สตริงคำตอบ ดังนี้

String Priority 1 = [4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]

String Priority 2 = [10 7 14 13 2 6 3 1 8 9 18 4 11 15 5 12 16 19 17]

String Priority 3 = [19 11 4 14 13 5 12 10 1 9 6 18 8 7 16 2 3 15 17]

String Priority 4 = [14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]

String Priority 5 = [16 14 718 1 15 5 10 13 11 6 19 9 17 3 4 8 12 2]

4.11.3 การถอดรหัสคำตอบ

จากสตริงคำตอบเบื้องต้นทั้ง 5 ตัว ยังไม่สามารถนำไปจัดลงสถานีงานได้ จึงต้องทำการแปลงสตริงคำตอบเป็นลำดับงานให้เรียบร้อยเสียก่อน ซึ่งขั้นตอนการแปลงค่าสตริงคำตอบเป็นลำดับงานดังนี้

- พิจารณาว่ามีขั้นงานใดที่สามารถเลือกทำได้ลงในตำแหน่งข้างหน้างาน (Forward Work) จะดูจากตารางแสดงความสัมพันธ์ของขั้นงานในการทำงานข้างหน้า โดยหาผลรวมของคอลัมน์ใดมีค่าเท่ากับ 0 แสดงว่าขั้นงานนั้นสามารถถูกเลือกลงตำแหน่งของลำดับงานได้โดยไม่ผิดข้อจำกัดความสัมพันธ์ของขั้นงาน
- พิจารณาว่ามีขั้นงานใดที่สามารถเลือกทำได้ลงในตำแหน่งข้างหลังงาน (Backward Work) จะดูจากตารางแสดงความสัมพันธ์ของขั้นงานในการทำงานข้างหลัง โดยหาผลรวมของคอลัมน์ใดมีค่าเท่ากับ 0 แสดงว่าขั้นงานนั้นสามารถถูกเลือกลงตำแหน่งของลำดับงานได้โดยไม่ผิดข้อจำกัดความสัมพันธ์ของขั้นงาน
- พิจารณาค่าสิทธิในการเลือกงานจากสตริงคำตอบตามตำแหน่งของงานที่ถูกเลือกลงในตำแหน่งข้างหน้างาน (Forward Work) และถูกเลือกลงในตำแหน่งข้างหลังงาน (Backward Work) ซึ่งขั้นงานที่สามารถถูกเลือกลงตำแหน่งข้างหน้าและข้างหลังงานใดมีค่ามากสุดจะถูกเลือกลงในลำดับขั้นงานก่อน
- ถ้างานที่ถูกเลือกมาจากความสัมพันธ์ของขั้นงานในการทำงานข้างหน้า (Precedence Matrix Font) ให้ตัดทิ้งโดยการเปลี่ยนตัวเลข ในแถว Precedence Matrix Font เป็น 0 ทั้งหมด และในคอลัมน์ของงานนั้นเท่ากับ 1 ทั้งหมด และทำให้คอลัมน์งานนั้นใน Precedence Matrix Back มีค่าเท่ากับ 1 ทั้งหมด

- ถ้างานที่ถูกเลือกมาจากความสัมพันธ์ของขั้นงานในการทำงานข้างหลัง (Precedence Matrix Back)ให้ตัดทิ้งโดยการเปลี่ยนตัวเลข ในแถว Precedence Matrix Back เป็น 0 ทั้งหมด และในคอลัมน์ของงานนั้นเท่ากับ 1 ทั้งหมด และทำให้คอลัมน์งานนั้นใน Precedence Matrix Font มีค่าเท่ากับ 1 ทั้งหมด

ซึ่งเมื่อพิจารณาจากสตริงคำตอบที่ 1 ทำการถอดรหัสหาลำดับขั้นงานในการทำงานเพื่อนำไปจัดลงสถานีงานได้ จะสามารถพิจารณาลำดับงานได้ ดังตารางที่ 4.13

ตารางที่ 4.13 การคัดเลือกลำดับขั้นงานในสตริงคำตอบที่ 1

No.	TASK(Font)	TASK(back)	Task Sequence
1	1,2,3,4,5	6,10,15,16,18,19	16
2	1,2,3,4,5	6,10,15,18,19	2
3	1,3,4,5,8	6,10,15,18,19	8
4	1,3,4,5	6,10,15,18,19	6
5	1,3,4,5	10,15,18,19	15
6	1,3,4,5	9,10,12,18,19	12
7	1,3,4,5	7,9,10,18,19	4
8	1,3,5	7,9,10,18,19	3
9	1,5,9,10	7,9,10,18,19	7
10	1,5,9,10	1,9,10,18,19	10
11	1,5,9	1,9,18,19	18
12	1,5,9	1,9,19	9
13	1,5	1,19	1
14	5	19	19
15	5	14,17	17
16	5	13,14	14
17	5	13	13
18	5	11	11
19	5	5	5

เมื่อทำการแปลงค่าสตริงคำตอบทั้งหมด 5 ตัว จะได้ลำดับงานของสตริงคำตอบทั้ง 5 ตัว จะมีค่าดังนี้

Task Sequence 1 = [16 2 8 6 15 12 4 3 7 10 18 9 1 19 17 14 13 11 5]

Task Sequence 2 = [18 19 17 14 3 4 16 13 11 1 10 2 9 6 15 12 7 5 8]

Task Sequence 3 = [1 19 15 12 18 4 5 7 2 8 10 14 11 13 6 3 17 16 9]

Task Sequence 4 = [16 5 6 15 1 18 2 7 8 4 12 9 11 13 17 3 19 14 10]

Task Sequence 5 = [4 1 6 2 18 10 8 3 9 7 12 16 15 19 14 17 13 11 5]

จากลำดับงานที่ 1 จะทำการจัดสรรลงสถานีงาน เพื่อคำนวณหาค่าฟังก์ชันวัตถุประสงค์ โดยมีรอบเวลา (Cycle time) ในการทำงานเท่ากับ 2 ซึ่งลำดับงานที่ 1 สามารถจัดสรรลงสถานีงานได้ดังตารางที่ 4.14

ตารางที่ 4.14 การจัดสรรงานลงสถานีงาน

งาน	เวลาขั้นงาน	เวลารวมสถานีงานเมื่องานถูกจัดสรร	เวลาที่เหลือในสถานีงาน	สถานีงานที่
16	0.033	0.033	1.967	1
2	1.200	1.233	0.767	1
8	0.333	1.566	0.434	1
6	0.067	1.633	0.367	1
15	1.333	**2.933**	**เกิน**	
15	1.333	1.333	0.667	2
12	0.500	1.833	0.167	2
4	0.133	1.966	0.034	2
3	0.267	**2.233**	**เกิน**	
3	0.267	0.267	1.733	3
7	0.600	0.867	1.133	3
10	0.067	0.934	1.066	3
18	0.500	1.434	0.566	3
9	0.400	1.834	0.166	3

ตารางที่ 4.14 การจัดสรรงานลงสถานีงาน (ต่อ)

งาน	เวลาขั้นงาน	เวลารวมสถานีงานเมื่องานถูกจัดสรร	เวลาที่เหลือในสถานีงาน	สถานีงานที่
1	0.066	1.900	0.100	3
19	0.267	**2.167**	**เกิน**	
19	0.267	0.267	1.733	4
17	0.333	0.600	1.400	4
14	0.200	0.800	1.200	4
13	0.067	0.867	1.133	4
11	0.300	1.167	0.833	4
5	0.200	1.367	0.633	4

จากสตริงคำตอบที่ 1 จะได้

String Priority 1 = [4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]

Task Sequence 1 = [16 2 8 6 15 12 4 3 7 10 18 9 1 19 17 14 13 11 5]

มีจำนวนสถานีงานทั้งหมด 4 สถานีได้แก่

สถานีงานที่ 1 มีงาน 16, 2, 8 และ 6 เวลารวมในสถานีงานเท่ากับ 1.633

สถานีงานที่ 2 มีงาน 15, 12 และ 4 เวลารวมในสถานีงานเท่ากับ 1.966

สถานีงานที่ 3 มีงาน 3, 7, 10, 18, 9, และ 1 เวลารวมในสถานีงานเท่ากับ 1.900

สถานีงานที่ 4 มีงาน 19, 17, 14, 13, 11 และ 5 เวลารวมในสถานีงานเท่ากับ 1.367

เมื่อได้งานในแต่ละสถานีงานเรียบร้อย จะทำการคำนวณหาค่าตามฟังก์ชันวัตถุประสงค์ ในงานวิจัยนี้จะทำการหาวัตถุประสงค์ทั้งหมด 3 วัตถุประสงค์ คือ จำนวนสถานีงานมีจำนวนน้อยที่สุด งานมีผลต่างความสัมพันธ์ในสถานีงานมีค่าน้อยที่สุดและความผันแปรของเวลาในสถานีงานทั้งหมดมีค่าน้อยที่สุด ดังนี้

กำหนดให้ m คือ สถานีงาน

SN_k คือ จำนวนการเชื่อมต่อการทำงานในสถานีงาน k

$S_{\max}$ คือ เวลารวมที่สูงที่สุดในสถานีงาน

S_k คือ เวลารวมในสถานีงาน k

1 เพื่อให้มีจำนวนสถานีทำงานน้อยที่สุด

$$f_1(X) = Minimum\ m \tag{4.37}$$

2 เพื่อให้งานมีผลต่างความสัมพันธ์ในสถานีงานมีค่าน้อยที่สุด

$$f_2(X) = Minimum\ \ relatedness = Minimum\ \ m - m \Big/ \sum_{k=1}^{m} SN_k \tag{4.38}$$

3 เพื่อหาค่าต่ำสุดของค่าการกระจายภาระงานในแต่ละสถานีงาน

$$f_3(X) = Minimum\ SI = Minimum \sqrt{\sum_{k=1}^{m} (S_{\max} - S_k)^2 \Big/ m} \tag{4.39}$$

สตริงคำตอบที่ 1 จะทำให้มีค่าวัตถุประสงค์จำนวนสถานีงานเท่ากับ 4 สถานีงาน

$$f_1(x) = m = 4$$

ในส่วนการหาค่าวัตถุประสงค์ที่ 2 คือ ผลต่างความสัมพันธ์ในสถานีงานมีค่ามีวิธีการคำนวณดังนี้

สถานีงานที่ 1 มีเครือข่ายงานที่เชื่อมต่อกันในสถานีคือ 2-8, 6, 16 มีค่าเท่ากับ 3

สถานีงานที่ 2 มีเครือข่ายงานที่เชื่อมต่อกันในสถานีคือ 12-15, 4 มีค่าเท่ากับ 2

สถานีงานที่ 3 มีเครือข่ายงานที่เชื่อมต่อกันในสถานีคือ 3-9-10, 1-7, 18 มีค่าเท่ากับ 3

สถานีงานที่ 4 มีเครือข่ายงานที่เชื่อมต่อกันในสถานีคือ5-11-13-14-17-19 มีค่าเท่ากับ 1

ดังนั้น

$$f_2(x) = WR = m - \frac{m}{\sum_{k=1}^{m} SN_k}$$

$$f_2(x) = WR = 4 - \frac{4}{3+2+3+1}$$

$$f_2(x) = WR = 3.556$$

วัตถุประสงค์ที่ 3 เพื่อหาค่าการกระจายภาระงานในแต่ละสถานีงาน มีวิธีการคำนวณดังนี้

$$f_3(x) = SI = \sqrt{\frac{\sum_{k=1}^{m} (S_{\max} - S_k)^2}{m}}$$

$$f_3(x) = SI = \sqrt{\frac{(1.966-1.633)^2 + (1.966-1.966)^2 + (1.966-1.9)^2 + (1.966-1.367)^2}{4}}$$

$$f_3(x) = SI = \sqrt{\frac{0.474046}{4}} = 0.3442$$

จากสตริงคำตอบที่ 1 จะได้

1. String Priority 1 = [4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]

2. *Task Sequence* 1 = [16 2 8 6 15 12 4 3 7 10 18 9 1 19 17 14 13 11 5]

3.จำนวนสถานีงานทั้งหมด 4 สถานีได้แก่

- สถานีงานที่ 1 มีงาน 16, 2, 8 และ 6 เวลารวมในสถานีงานเท่ากับ 1.633
- สถานีงานที่ 2 มีงาน 15, 12 และ 4 เวลารวมในสถานีงานเท่ากับ 1.966
- สถานีงานที่ 3 มีงาน 3, 7, 10, 18, 9, และ 1 เวลารวมในสถานีงานเท่ากับ 1.900
- สถานีงานที่ 4 มีงาน 19, 17, 14, 13, 11 และ 5 เวลารวมในสถานีงานเท่ากับ 1.367

4.ผลต่างความสัมพันธ์ในสถานีงานมีค่า 3.556

5.ค่าการกระจายภาระงานในสถานีงานมีค่าเท่ากับ 0.3442

จากสตริงคำตอบทั้งหมด 5 ตัว ได้ค่าจากการคำนวณวัตถุประสงค์ ในตารางที่ 4.15

ตารางที่ 4.15 ค่าจากการคำนวณวัตถุประสงค์ทั้ง 3 วัตถุประสงค์

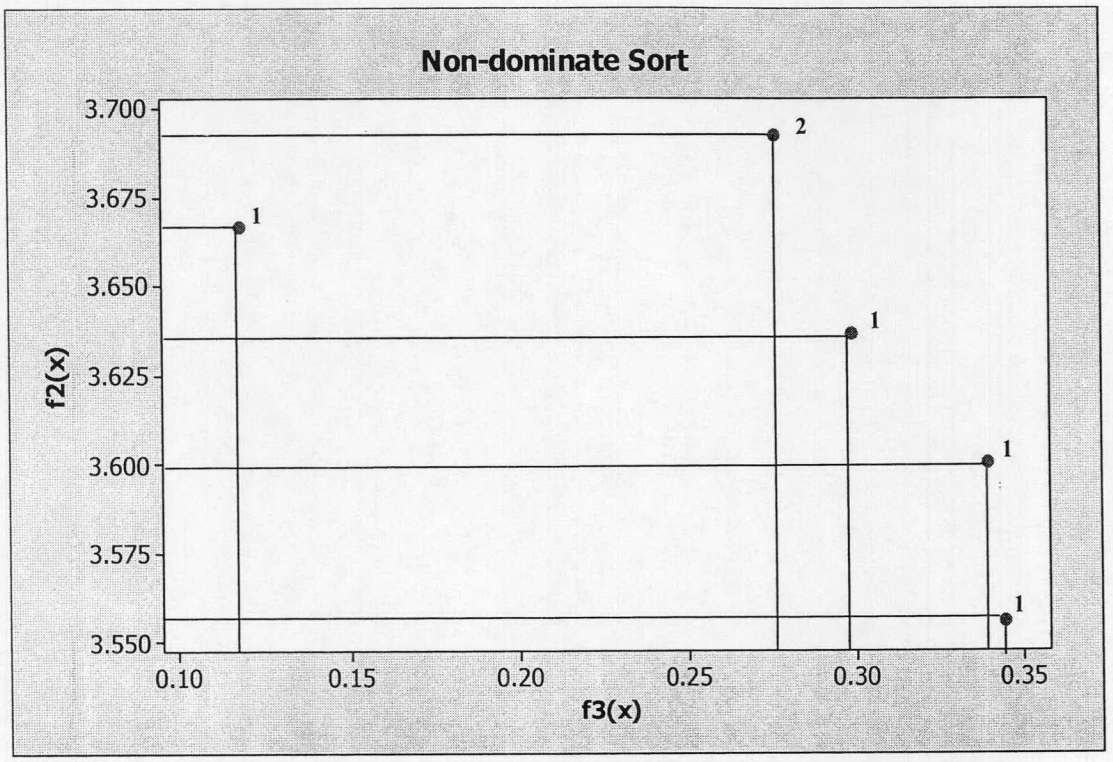
สตริงคำตอบที่	จำนวนสถานีงาน	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน
1	4	3.55556	0.344207
2	4	3.66667	0.117734
3	4	3.69231	0.276406
4	4	3.63636	0.299056
5	4	3.60000	0.339191

<u>หมายเหตุ</u> วัตถุประสงค์ที่ 1 หรือจำนวนสถานีงาน สามารถคำนวณได้จากค่าในวัตถุประสงค์ที่ 2 หรือผลต่างความสัมพันธ์ในสถานีงาน โดยการปัดเศษขึ้นเป็นจำนวนจริง ตัวอย่างเช่น 3.25 แสดงว่ามีจำนวนสถานีงานเท่ากับ 4 สถานีงาน

4.11.4 การประเมินค่า

ในการกำหนดค่าความแข็งแรง (Fitness Value) ให้แก่สตริงคำตอบ จะใช้วิธีการจัดอันดับแบบ Goldberg โดยค่าอันดับที่ได้นี้จะเป็นค่าความแข็งแรงไม่แท้จริง (Dummy Fitness

Value) โดยขั้นตอนนี้จะได้เส้นขอบเขตกลุ่มคำตอบที่ดี (Frontier) ออกมาหลายกลุ่มตามค่า Dummy Fitness เนื่องจากค่าจำนวนสถานีงานของสตริงคำตอบเท่ากันหมดจึงไม่ทำการพิจารณาวัตถุประสงค์นี้ จะได้ค่าดังรูปที่ 4.30 และตารางที่ 4.16



รูปที่ 4.30 ค่า Dummy Fitness วิธีการจัดอันดับแบบ Goldberg

ตารางที่ 4.16 ค่าความแข็งแรงไม่แท้จริง (Dummy Fitness)

สตริงคำตอบที่	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน	Dummy Fitness
1	3.55556	0.344207	1
2	3.66667	0.117734	1
3	3.69231	0.276406	2
4	3.63636	0.299056	1
5	3.60000	0.339191	1

การคำนวณหาค่าความหนาแน่นด้วยวิธี Crowding Distance

การคำนวณจะทำหาค่าสูงสุดและต่ำสุดของค่าวัตถุประสงค์ทั้ง 2 วัตถุประสงค์ ซึ่ง

กำหนดให้ $f_2^{\max}, f_2^{\min}$ คือ ค่าฟังก์ชันวัตถุประสงค์ที่ 2 ที่มีค่าสูงสุดสุดและต่ำสุด ตามลำดับ

$f_3^{\max}, f_3^{\min}$ คือ ค่าฟังก์ชันวัตถุประสงค์ที่ 3 ที่มีค่าสูงสุดสุดและต่ำสุด ตามลำดับ

ในการคำนวณหาค่า Crowding Distance จะทำการพิจารณาที่ละ Front ดังนั้นในที่นี้จะทำการพิจารณาที่ Front ที่ 1 ก่อน

จากตารางที่ ทำการหาค่าฟังก์ชันวัตถุประสงค์ที่ 2 และ 3 ที่มีค่าสูงสุดและต่ำที่สุด จะมีค่าดังนี้ $f_2^{\max} = 3{,}69231$, $f_2^{\min} = 3.55556$, $f_3^{\max} = 0.344207$ และ $f_3^{\min} = 0.117734$ จากนั้นจะทำการเรียงค่าที่อยู่ใน Front ที่ 1 โดยเรียงค่าวัตถุประสงค์ที่ 2 จากน้อยไปหามากได้ดังตารางที่ 4.17

ตารางที่ 4.17 การเรียงลำดับค่าฟังก์ชันวัตถุประสงค์ที่ 2 ใน Front ที่ 1

สตริงคำตอบที่	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน	Dummy Fitness	i
1	3.55556	0.344207	1	1
5	3.60000	0.339191	1	2
4	3.63636	0.299056	1	3
2	3.66667	0.117734	1	4

จากตารางที่ 4.17 สมาชิกคำตอบที่มีลำดับที่ 1 ($i = 1$) หรือมีค่าฟังก์ชันวัตถุประสงค์น้อยที่สุด และลำดับสุดท้าย ($i = 4$) หรือมีค่าฟังก์ชันวัตถุประสงค์มากที่สุด คำตอบสองคำตอบนี้จะถือว่ามีค่า Crowding Distance เท่ากับอนันต์ (Infinity)

ส่วนลำดับที่เหลือจะทำการคำนวณหา Crowding Distance ซึ่งในที่นี้คือลำดับที่ 2 และ 3 โดยคำนวณที่ลำดับที่ 2 ได้ค่าดังนี้

$$cd_1\left(x_{[2,2]}\right) = \left|\frac{f_2\left(x_{[2+1,2]}\right) - f_2\left(x_{[2-1,2]}\right)}{f_2^{\max} - f_2^{\min}}\right|$$

$$cd_1\left(x_{[2,2]}\right) = \left|\frac{3.63636 - 3.55556}{3.66667 - 3.55556}\right| = \left|\frac{0.0808}{0.11111}\right| = 0.72721$$

และ

$$cd_2\left(x_{[2,3]}\right)=\left|\frac{f_3\left(x_{[2+1,3]}\right)-f_3\left(x_{[2-1,3]}\right)}{f_3^{\max}-f_3^{\min}}\right|$$

$$cd_2\left(x_{[2,3]}\right)=\left|\frac{0.299056-0.344207}{0.344207-0.114434}\right|=\left|\frac{-0.045151}{0.229773}\right|=0.19650$$

ลำดับที่ 2 จะมีค่า Crowding Distance เท่ากับ $0.72721+0.19650=0.9237$

คำนวณค่า Crowding Distance ที่ลำดับที่ 3 ได้ค่าดังนี้

$$cd_1\left(x_{[3,2]}\right)=\left|\frac{f_2\left(x_{[3+1,2]}\right)-f_2\left(x_{[3-1,2]}\right)}{f_2^{\max}-f_2^{\min}}\right|$$

$$cd_1\left(x_{[3,2]}\right)=\left|\frac{3.66667-3.60000}{3.66667-3.55556}\right|=\left|\frac{0.06667}{0.11111}\right|=0.60000$$

และ

$$cd_2\left(x_{[3,3]}\right)=\left|\frac{f_3\left(x_{[3+1,3]}\right)-f_3\left(x_{[3-1,3]}\right)}{f_3^{\max}-f_3^{\min}}\right|$$

$$cd_2\left(x_{[3,3]}\right)=\left|\frac{0.117734-0.339191}{0.344207-0.114434}\right|=\left|\frac{-0.22145}{0.229773}\right|=0.96377$$

ลำดับที่ 3 จะมีค่า Crowding Distance เท่ากับ $0.60000+0.96377=1.5637$

ในส่วนค่า Crowding Distance ของ Front ที่ 2 มีค่าเดียวจึงกำหนดให้มีค่าเป็น อนันต์ (infinity) ได้เลย ดังนั้นค่า Crowding Distance ของสตริงคำตอบทั้ง 5 ตัว มีค่าดังตารางที่ 4.18

ตารางที่ 4.18 ค่า Crowding Distance ของสตริงคำตอบ

สตริงคำตอบที่	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน	Dummy Fitness	Crowding Distance
1	3.55556	0.344207	1	infinity
2	3.66667	0.117734	1	infinity
3	3.69231	0.276406	2	infinity
4	3.63636	0.299056	1	1.5637
5	3.60000	0.339191	1	0.9237

4.11.5 การคัดเลือกสตริงคำตอบ

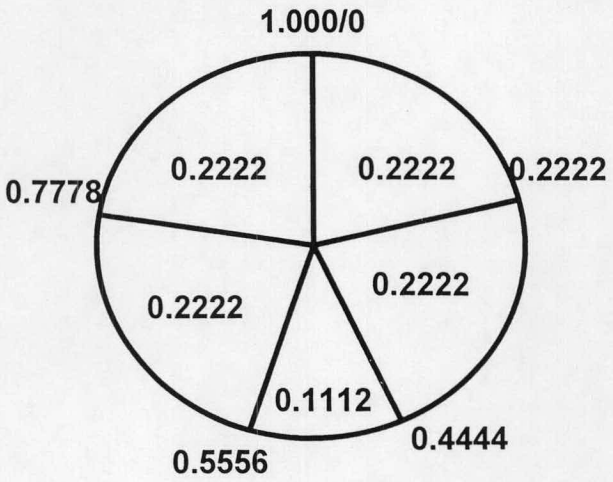
คัดเลือกคำตอบรุ่นพ่อแม่จากการใช้ binary tournament selection จากการหาค่า Fitness Value ที่ได้จากการหา Non-dominated Sorting โดยทำการสลับค่าให้ค่าจากค่า Dummy Fitness จาก ค่าน้อยเป็นค่ามากและคำนวณหาค่า pi และ qi ซึ่งค่า qi คือค่า pi สะสม จากสตริงคำตอบที่ 1 มีค่า Dummy Fitness เท่ากับ 1 เปลี่นนเป็นค่าเท่ากับ 2 และทำการหาค่า $p_1 = 2/9 = 0.2222$ ดังรูปที่ 4.31 และ ตารางที่ 4.19-4.20

ตารางที่ 4.19 การแปลงค่าความแข็งแรงไม่แท้จริง (Dummy Fitness)

String No	WR	SI	Fitness Value	แปลง Fitness	Crowding Distance
1	3.55556	0.344207	1	2	infinity
2	3.66667	0.117734	1	2	infinity
3	3.69231	0.276406	2	1	infinity
4	3.63636	0.299056	1	2	1.5637
5	3.60000	0.339191	1	2	0.9237

ตารางที่ 4.20 การสร้างวงล้อรูเล็ตของปัญหาตัวอย่าง 19 ชิ้นงาน

String No	แปลง Fitness	pi	qi
1	2	0.2222	0.2222
2	2	0.2222	0.4444
3	1	0.1112	0.5556
4	2	0.2222	0.7778
5	2	0.2222	1
รวม	9	1	



รูปที่ 4.31 วงล้อรูเล็ตของปัญหาตัวอย่าง 19 ขั้นงาน

สุ่มเลือกสตริงคำตอบ 2 ตัวจากวงล้อรูเล็ตแล้วนำมาเปรียบเทียบกัน เพื่อคัดเลือกสตริงคำตอบที่มีค่า Fitness มากกว่าเข้าสู่ Mating Pool ผลการคัดเลือกจะได้สตริงทั้ง 5 ตัวคือ สตริงหมายเลข 1 2 4 4 1 แสดงได้ดังตารางที่ 4.21 ซึ่งจะกลายเป็นสตริงหมายเลข 1-5 ตามลำดับเข้าสู่ขั้นตอนต่อไป

ตารางที่ 4.21 วิธี Binary Tournament Selection สำหรับการคัดเลือกสตริงคำตอบ

No.	Population 1				Population 2				N0_String
	r_1	$q_i > r_1$	String	Fitness	r_2	$q_i > r_2$	String	Fitness	Selected
1	0.0707	0.2222	1	2	0.0119	0.2222	1	2	1
2	0.2272	0.4444	2	2	0.5163	0.5556	3	1	2
3	0.4582	0.5556	3	1	0.7032	0.7778	4	2	4
4	0.5825	0.7778	4	2	0.5092	0.5556	3	1	4
5	0.0743	0.2222	1	2	0.9874	1	5	2	1*

*หมายเหตุ สตริงคำตอบที่ 1 และ 5 มีค่า Fitness ที่มีการแปลงค่าเท่ากัน มีค่าเท่ากับ 2 ดังนั้นจะพิจารณาเลือกสตริงที่มีค่า Crowding Distance ในที่นี้คือสตริงคำตอบที่ 1 มีค่าเป็นอนันต์ (Infinity)

4.11.6 การครอสโอเวอร์

ทำการสุ่มเลือกสตริงคำตอบเพื่อทำการครอสโอเวอร์ โดยพิจารณาจากสตริงคำตอบที่มีค่าสุ่ม r น้อยกว่าค่า Pc ซึ่งในที่นี้กำหนดให้ $Pc = 0.8$ ดังนั้นสตริงที่จะถูกครอสโอเวอร์ จึงจะมีประมาณ 80% ของสตริงคำตอบทั้งหมด หรือเท่ากับ $0.8 \times 5 = 4$ ตัว การสุ่มเลือกสตริงคำตอบแสดงได้ดังตารางที่ 4.22

ตารางที่ 4.22 สตริงค่าสิทธิในการเลือกงานที่ถูกเลือกทำการครอสโอเวอร์

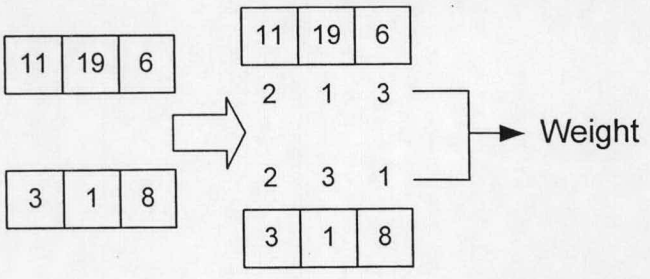
String No.	String Priority	r_i	$r_i < 0.8$
1	[4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]	0.489	Selected
2	[10 7 14 13 2 6 3 1 8 9 18 4 11 15 5 12 16 19 17]	0.557	Selected
3	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]	0.821	-
4	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]	0.702	Selected
5	[4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]	0.957	-

เนื่องจากสตริงที่ถูกสุ่มเลือกไปครอสโอเวอร์มีเพียง 3 ตัวคือสตริงหมายเลข 1 2 4 ซึ่งไม่สามารถจับคู่ได้จึงต้องทำการลดหรือเพิ่มสตริงคำตอบโดยสุ่มเลข 0 หรือ 1 ในที่นี้ให้สุ่มได้เลข 1 ซึ่งหมายความว่าจะต้องเพิ่มสตริงคำตอบเข้าไปอีกหนึ่งตัว โดยเลือกจากสตริงคำตอบที่เหลือสมมติเลือกได้สตริงหมายเลข 5 ดังนั้นจะได้สตริงตอบที่จะนำไปครอสโอเวอร์ คือสตริงหมายเลข 1 2 4 5 ซึ่งสามารถจับคู่ได้เป็น 1-2 หรือ 4-5

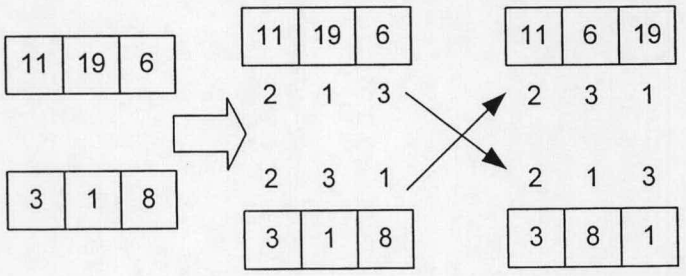
นำสตริงคู่แรกไปครอสโอเวอร์ด้วยวิธี Weight mapping crossover (WMX) โดยสุ่มเลือกตำแหน่งการครอสโอเวอร์ได้ที่ตำแหน่ง 7 และ 9

$$\text{parent } 1 = [4\ 17\ 12\ 13\ 1\ 15\ |11\ 19\ 6|\ 10\ 8\ 16\ 2\ 5\ 14\ 18\ 9\ 7\ 3]$$
$$\text{parent } 2 = [10\ 7\ 14\ 13\ 2\ 6\ |3\ 1\ 8|\ 9\ 18\ 4\ 11\ 15\ 5\ 12\ 16\ 19\ 17]$$

ทำการกำหนดค่าน้ำหนักในช่วง[7,9]ให้แก่สตริงพ่อแม่(Parent) ได้ดังรูปที่ 4.32



ก) กำหนดค่าน้ำหนักให้แก่สตริงพ่อแม่คู่ที่ 1และ 2



ข) การสลับค่าน้ำหนักที่กำหนดและทำการเปลี่ยน
ค่าภายในโครโมโซมในสตริงคำตอบที่ 1,2

รูปที่ 4.32 การครอสโอเวอร์วิธี Weight mapping crossover (WMX)
สตริงคำตอบคู่ที่ 1,2

จะได้สตริงคำตอบในรุ่นลูก(Offspring) หลังจากการทำครอสโอเวอร์วิธี Weight mapping crossover (WMX) คือ

$$\text{offspring } 1 = [4\ 17\ 12\ 13\ 1\ 15\ |11\ 6\ 19|\ 10\ 8\ 16\ 2\ 5\ 14\ 18\ 9\ 7\ 3]$$

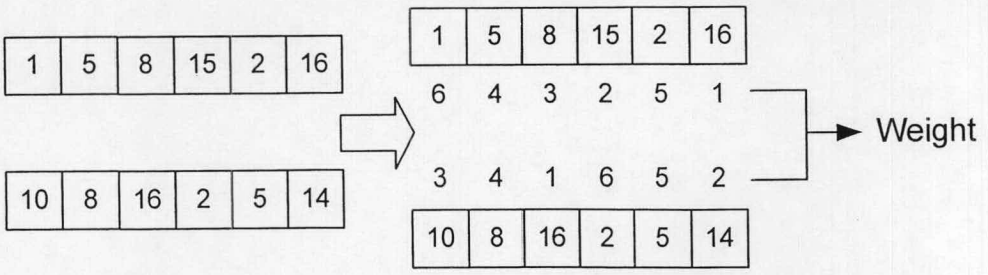
$$\text{offspring } 2 = [10\ 7\ 14\ 13\ 2\ 6\ |3\ 8\ 1|\ 9\ 18\ 4\ 11\ 15\ 5\ 12\ 16\ 19\ 17]$$

สตริงที่ 4 และ 5ไปครอสโอเวอร์ด้วยวิธี Weight mapping crossover (WMX) โดยสุ่มเลือกตำแหน่งการครอสโอเวอร์ได้ที่ตำแหน่ง 10 และ 15

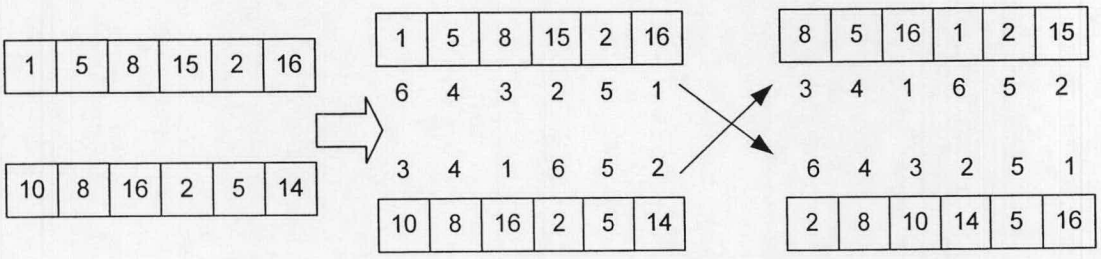
$$\text{parent } 3 = [14\ 11\ 4\ 9\ 18\ 17\ 13\ 10\ 6|1\ 5\ 8\ 15\ 2\ 16|19\ 7\ 12\ 3]$$

$$\text{parent } 4 = [4\ 17\ 12\ 13\ 1\ 15\ 11\ 19\ 6|10\ 8\ 16\ 2\ 5\ 14|18\ 9\ 7\ 3]$$

ทำการกำหนดค่าน้ำหนักในช่วง[7,9]ให้แก่สตริงพ่อแม่(Parent) ได้ดังรูปที่ 4.33



ก) กำหนดค่าน้ำหนักให้แก่สตริงพ่อแม่ที่ 3และ 4



ข) การสลับค่าน้ำหนักที่กำหนดและทำการเปลี่ยน
ค่าภายในโครโมโซมในสตริงคำตอบที่ 3,4

รูปที่ 4.33 การครอสโอเวอร์วิธี Weight mapping crossover (WMX)
สตริงคำตอบคู่ที่ 3,4

จะได้สตริงคำตอบในรุ่นลูก(Offspring) หลังจากการทำครอสโอเวอร์วิธี Weight mapping crossover (WMX) คือ

offspring 3 = [14 11 4 9 18 17 13 10 6|8 5 16 1 2 15|19 7 12 3]

offspring 4 = [4 17 12 13 1 15 11 19 6|2 8 10 14 5 16|18 9 7 3]

เนื่องจากสตริงคำตอบที่ได้จากหลังการทำครอสโอเวอร์เป็นสตริงคำตอบค่าสิทธิในการเลือกงานทำให้ไม่ต้องทำการซ่อมแซมคำตอบ สตริงที่ได้จากการทำครอสโอเวอร์จะถูกนำไปทำการมิวเตชั่น

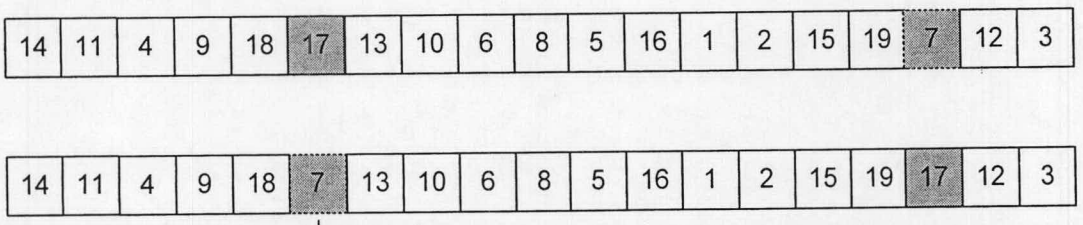
4.11.7 การมิวเตชั่น

ในที่นี้กำหนดให้ $Pm = 0.2$ ซึ่งทำให้สามารถคาดเดาได้ว่าน่าจะมีสตริงคำตอบ 20% หรือ $0.2 \times 5 = 1$ ตัว ที่จะถูกมิวเตชั่น สตริงตัวนี้จะได้มาจากการสุ่มค่า r ให้กับสตริงแต่ละตัว แล้วถ้าตัวใดที่ r น้อยกว่า Pm ก็จะถูกนำไปมิวเตชั่น ดังตารางที่ 4.23

ตารางที่ 4.23 ผลการคัดเลือกสตริงคำตอบเพื่อทำการมิวเตชั่น

String No.	String Priority	r_i	$r_i < 0.2$
1	[4 17 12 13 1 15 11 6 19 10 8 16 2 5 14 18 9 7 3]	0.489	-
2	[10 7 14 13 2 6 3 8 1 9 18 4 11 15 5 12 16 19 17]	0.557	-
3	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]	0.821	-
4	[14 11 4 9 18 17 13 10 6 8 5 16 1 2 15 19 7 12 3]	0.052	Selected
5	[4 17 12 13 1 15 11 19 6 2 8 10 14 5 16 18 9 7 3]	0.957	-

สตริงคำตอบตัวที่ 4 เป็นสตริงคำตอบที่ถูกเลือกให้ทำการมิวเตชั่น โดยใช้วิธี Reciprocal Exchange Mutation เป็นการสลับตำแหน่งของตัวเลข 2 ตัวภายในสตริงคำตอบ โดยเริ่มจากการทำการสุ่มตัว 2 ตัวที่ไม่ซ้ำกัน สมมติสุ่มได้เลข 6 และ 17 จากนั้นทำการสลับตำแหน่งของตัวเลขทั้งสอง



รูปที่ 4.34 วิธี Reciprocal Exchange Mutation

จะได้สตริงคำตอบในรุ่นลูกทั้งหมด 5 ตัว หลังการทำมิวเตชั่นเพื่อนำไปรวมกับสตริงคำตอบเริ่มต้น เพื่อทำการเก็บค่าที่ดีที่สุดของสตริงคำตอบไว้

ตารางที่ 4.24 สตริงคำตอบหลังการทำการมิวเตชั่น

String No.	String Priority
1	[4 17 12 13 1 15 11 6 19 10 8 16 2 5 14 18 9 7 3]
2	[10 7 14 13 2 6 3 8 1 9 18 4 11 15 5 12 16 19 17]
3	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]
4	[14 11 4 9 18 7 13 10 6 8 5 16 1 2 15 19 17 12 3]
5	[4 17 12 13 1 15 11 19 6 2 8 10 14 5 16 18 9 7 3]

4.11.8 เทคนิคการเก็บค่าที่ดีที่สุด

จากสตริงคำตอบเริ่มต้น (P) และสตริงคำตอบรุ่นลูก (Q) จะนำมาทำการรวมกันและทำการเก็บค่าที่ดีที่สุดเท่ากับจำนวนสตริงคำตอบเริ่มต้น ในที่นี้จะทำการเก็บสตริงคำตอบไว้จำนวนเท่ากับ 5 ตัว

ตารางที่ 4.25 สตริงคำตอบพ่อแม่รวมกับสตริงคำตอบรุ่นลูก

ลักษณะสตริงคำตอบ	String No.	String Priority
สตริงคำตอบเริ่มต้น (P)	1	[4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]
	2	[10 7 14 13 2 6 3 1 8 9 18 4 11 15 5 12 16 19 17]
	3	[19 11 4 14 13 5 12 10 1 9 6 18 8 7 16 2 3 15 17]
	4	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]
	5	[16 14 718 1 15 5 10 13 11 6 19 9 17 3 4 8 12 2]
สตริงคำตอบรุ่นลูก (Q)	6	[4 17 12 13 1 15 11 6 19 10 8 16 2 5 14 18 9 7 3]
	7	[10 7 14 13 2 6 3 8 1 9 18 4 11 15 5 12 16 19 17]
	8	[14 11 4 9 18 17 13 10 6 1 5 8 15 2 16 19 7 12 3]
	9	[14 11 4 9 18 7 13 10 6 8 5 16 1 2 15 19 17 12 3]
	10	[4 17 12 13 1 15 11 19 6 2 8 10 14 5 16 18 9 7 3]

สตริงคำตอบที่ได้ทำการรวมกันแล้ว จะนำไปคำนวณหาค่าวัตถุประสงค์ทั้ง 3 วัตถุประสงค์จึงต้องการแปลงค่าสตริงคำตอบทั้ง 10 คำตอบ ให้เป็นลำดับของขั้นงานให้เรียบร้อยก่อน ได้ค่าดังตารางที่ 4.26

ตารางที่ 4.26 ลำดับขั้นงานของสตริงคำตอบที่ทำการรวมกัน

ลักษณะสตริงคำตอบ	String No.	Task Sequence
สตริงคำตอบเริ่มต้น (P)	1	[16 2 8 6 15 12 4 3 7 10 18 9 1 19 17 14 13 11 5]
	2	[18 19 17 14 3 4 16 13 11 1 10 2 9 6 15 12 7 5 8]
	3	[1 19 15 12 18 4 5 7 2 8 10 14 11 13 6 3 17 16 9]
	4	[16 5 6 15 1 18 2 7 8 4 12 9 11 13 17 3 19 14 10]
	5	[4 1 6 2 18 10 8 3 9 7 12 16 15 19 14 17 13 11 5]
สตริงคำตอบรุ่นลูก (Q)	6	[16 2 6 15 9 12 4 3 7 10 18 8 1 19 17 14 13 11 5]
	7	[18 19 17 14 3 4 16 13 11 1 10 2 8 6 15 12 7 5 9]
	8	[16 5 6 15 1 18 2 7 8 4 12 9 11 13 17 3 19 14 10]
	9	[16 5 15 12 1 7 18 2 8 4 10 6 9 11 3 19 17 14 13]
	10	[16 2 8 15 6 4 3 12 7 18 9 1 19 17 13 14 11 10 5]

จากลำดับขั้นงานของสตริงคำตอบที่ทำการรวมกัน จะนำมาคำนวณหาค่าฟังก์ชันวัตถุประสงค์ทั้ง 3 วัตถุประสงค์ ซึ่งจะได้ค่าดังตารางที่ 4.27

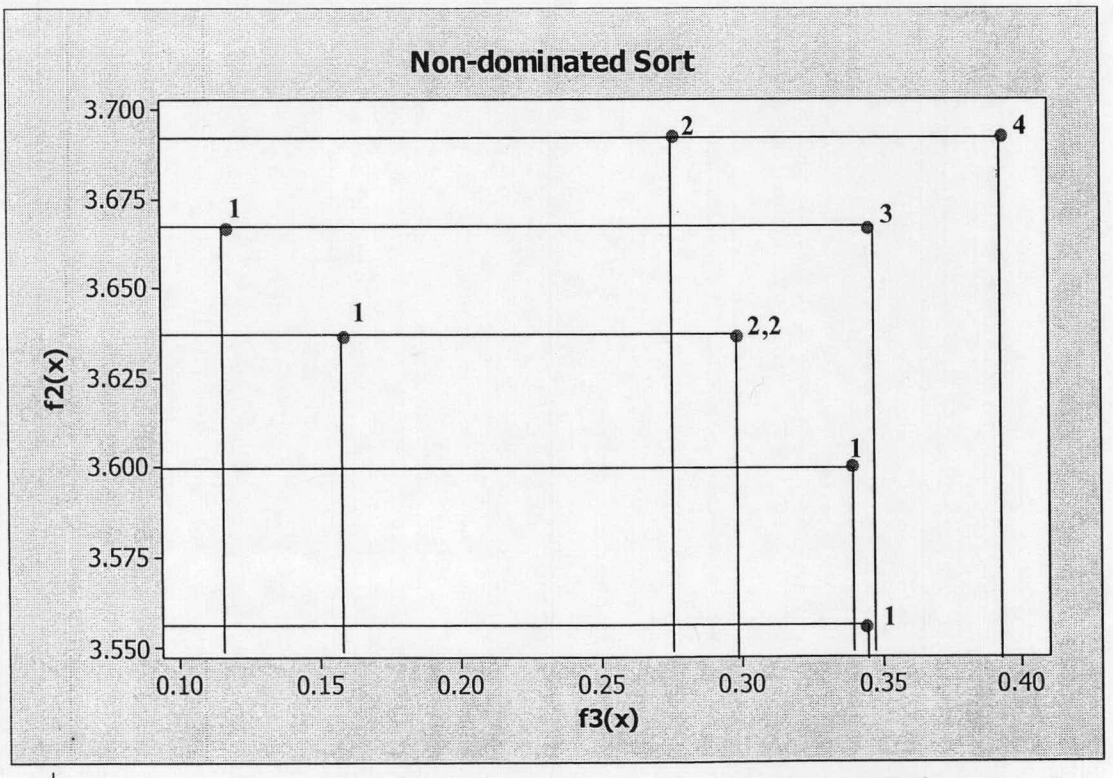
ตารางที่ 4.27 ค่าฟังก์ชันวัตถุประสงค์ของสตริงคำตอบที่ทำการรวมกัน

สตริงคำตอบที่	จำนวนสถานีงาน	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน
1	4	3.5556	0.3442
2	4	3.6667	0.1177
3	4	3.6923	0.2764
4	4	3.6364	0.2991
5	4	3.6000	0.3392

ตารางที่ 4.27 ค่าฟังก์ชันวัตถุประสงค์ของสตริงคำตอบที่ทำการรวมกัน (ต่อ)

สตริงคำตอบที่	จำนวนสถานีงาน	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน
6	5	4.5833	0.6107
7	4	3.6923	0.3939
8	4	3.6364	0.2991
9	4	3.6364	0.1588
10	4	3.6667	0.3455

ใช้วิธีการจัดอันดับแบบ Goldberg เพื่อกำหนดค่าความแข็งแรงไม่แท้จริง (Dummy Fitness Value) เนื่องจากมีจุดนึงที่มีค่าสูงสุดทำให้ไม่เห็นถึงความแตกต่างของจึงทำการตัดค่าของสตริงคำตอบ แล้วทำการกำหนดให้เป็นค่าความแข็งแรงไม่แท้จริงจุดสุดท้าย และทำการคำนวณ Crowding Distance ได้ดังรูป 4.35 และตารางที่ 4.28



รูปที่ 4.35 กำหนดค่าความแข็งแรงไม่แท้จริง (Dummy Fitness Value) ของสตริงคำตอบรวมกัน

ตารางที่ 4.28 ค่าความแข็งแรงไม่แท้จริง (Dummy Fitness Value) และค่า Crowding Distance

สตริงคำตอบที่	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน	Dummy Fitness	Crowding Distance
1	3.5556	0.3442	1	Infinity
2	3.6667	0.1177	1	Infinity
3	3.6923	0.2764	2	Infinity
4	3.6364	0.2991	2	Infinity
5	3.6	0.3392	1	1.5458
6	4.5833	0.6107	5	Infinity
7	3.6923	0.3939	4	Infinity
8	3.6364	0.2991	2	Infinity
9	3.6364	0.1588	1	1.5783
10	3.6667	0.3455	3	Infinity

ทำการเรียงค่าจากน้อยไปมากของค่า Dummy Fitness และภายใน Front ทำการเรียงค่าจามมากไปน้อยของค่า Crowding Distance ได้ดังตารางที่ 4.29

ตารางที่ 4.29 เรียงค่าจากน้อยไปมากของค่า Dummy Fitness และเรียงค่าจากมากไปน้อยของค่า Crowding Distance

สตริงคำตอบที่	ผลต่างความสัมพันธ์ในสถานีงาน	กระจายภาระงานในสถานีงาน	Dummy Fitness	Crowding Distance
1	3.5556	0.3442	1	Infinity
2	3.6667	0.1177	1	Infinity
9	3.6364	0.1588	1	1.5783
5	3.6	0.3392	1	1.5458
4	3.6364	0.2991	2	Infinity
3	3.6923	0.2764	2	Infinity
8	3.6364	0.2991	2	Infinity
10	3.6667	0.3455	3	Infinity
7	3.6923	0.3939	4	Infinity
6	4.5833	0.6107	5	Infinity

เมื่อคำการจัดเรียงเสร็จแล้ว จะทำการคัดเลือกสตริงคำตอบเพื่อทำการเก็บค่าที่ทีที่สุดของสตริงคำตอบ จะทำการพิจารณาทีละ Front จากน้อยไปมาก ในที่นี้สตริงคำตอบที่อยู่ใน Front ที่ 1 มีจำนวนสตริงคำตอบไม่เกินสตริงคำตอบที่ทำการจัดเก็บ $(popsize = 5)$ ดังนั้นจึงทำการเลือกจัดเก็บทั้งหมด เมื่อพิจารณาจาก Front ที่ 2 พบว่ามีจำนวนมากกว่าที่จะทำการนำไปเก็บจึงต้องทำการเลือกสตริงคำตอบที่มีค่า Crowding Distance มากที่สุดใน Front นั้น ซึ่งในที่นี้พบว่ามีค่าเท่ากันหมด จึงต้องทำการสุ่มเลือกสตริงคำตอบมาหนึ่งตัว สมมติสุ่มได้สตริงคำตอบตัวที่ 3 ดังนั้นจะได้สตริงคำตอบที่ดีที่สุด ทำการเก็บไว้ดำเนินขบวนการ NSGAII ในรอบถัดไป ดังตารางที่ 4.30

ตารางที่ 4.30 สตริงคำตอบรุ่นลูกที่จะถูกพัฒนาไปเป็นสตริงคำตอบเริ่มต้นในรอบถัดไป

สตริงคำตอบที่	String Priority
1	[4 17 12 13 1 15 11 19 6 10 8 16 2 5 14 18 9 7 3]
2	[10 7 14 13 2 6 3 1 8 9 18 4 11 15 5 12 16 19 17]
9	[14 11 4 9 18 7 13 10 6 8 5 16 1 2 15 19 17 12 3]
5	[16 14 718 1 15 5 10 13 11 6 19 9 17 3 4 8 12 2]
3	[19 11 4 14 13 5 12 10 1 9 6 18 8 7 16 2 3 15 17]

4.12 การวัดสมรรถนะของกลุ่มคำตอบที่ดีที่สุด

หนึ่งในความแตกต่างของการแก้ปัญหาด้วย Multi-objective GAs กับ GAs ก็คือคำตอบที่ได้จากการค้นหา เนื่องจากผลลัพธ์ที่ได้จากการแก้ปัญหาด้วย GAs จะได้คำตอบเพียงคำตอบเดียว ส่วน Multi-objective GAs คือขอบเขตกลุ่มคำตอบที่ดีที่สุด ดังนั้นขอบเขตกลุ่มคำตอบที่ได้นั้นจะยอมรับได้ว่าเป็นขอบเขตกลุ่มคำตอบที่ดีก็ต่อเมื่อสามารถวัดสมรรถนะของของคำตอบว่าใกล้เคียงกับขอบเขตกลุ่มคำตอบที่ดีที่สุดที่แท้จริง (True Pareto Optimal) หรือ (Reference Pareto Optimul)ขอบเขตกลุ่มคำตอบอ้างอิง ดังนั้นการวัดสมรรนะของกลุ่มคำตอบที่ดีที่สุด (Performance Measure) เป็นการวัดคุณภาพของคำตอบที่หาได้จากอัลกอริทึ่มในอัลกอรึทึ่มหนึ่งและสามารถเปรียบเทียบคุณภาพคำตอบที่หาได้จากหลายอัลกอริทึมด้วยตัววัดสมรรถนะในด้านต่าง ๆ เช่น การลู่เข้าสู่กลุ่มคำตอบที่แท้จริง (Convergence Measure) การกระจ่ายของกลุ่มคำตอบที่แท้จริง (Spread Measure) เป็นต้น Ishibuchi, Yoshida และMurata (2003) ในงานวิจัยนี้ ได้ใช้การวัดสมรรถนะของกลุ่มคำตอบที่ดีที่สุดบนพื้นฐานการเปรียบเทียบระยะทางจากสมาชิกกลุ่มคำตอบที่แท้จริงกับกลุ่มคำตอบทุกคำตอบที่หาได้ (Obtained Pareto Optimal) โดยในที่นี้ได้ทำการประมาณขอบเขตกลุ่มคำตอบที่แท้จริงจาการใช้กลุ่มคำตอบที่ดีสุดได้จากทุกอัลกอริทึมที่ใช้ทดลองในงานวิจัยมารวมกันและใช้หลักการ Pareto Dominance ในการสร้างกลุ่มคำตอบที่แท้จริง โดยคณะวิจัยได้กล่าวว่าในการวัดสมรรถนะกลุ่มคำตอบที่ดีที่สุดนี้จำเป็นต้องใช้ตัววัดสองตัวเพื่อให้บรรลุเป้าหมายของการหาคำตอบในลักษณะเชิงกลุ่ม ในที่นี้ได้ใช้ตัววัดสรรถนะที่เรียกว่าตัววัด $D1_R$ และ R_{NDS} ดังสมการที่(4.40) และ (4.42) ดังนี้

ตัววัดที่หนึ่ง : ตัววัดสมรรถนะการลู่เข้าสู่กลุ่มคำตอบที่แท้จริง

$$D1_R(S_j) = \frac{1}{|S^*|}\sum_{y \in s^*} \min\{d_{xy} \mid x \in S_j\} \tag{4.40}$$

โดยที่ S_j คือเซตคำตอบที่หาได้ตัวที่ j เมื่อ

S^* คือเซตคำตอบที่แท้จริง

$|S^*|$ คือจำนวนคำตอบที่แท้จริง

d_{xy} เป็นระยะทางระหว่างคำตอบที่หาได้ x กับคำตอบที่แท้จริง y ที่ได้รับการ Normalized แล้ว

$$d_{xy} = \sqrt{\sum_{i=1}^{k} \left(\frac{f_i(x) - f_i(y)}{f_i^{\max} - f_i^{\max}}\right)^2} \tag{4.41}$$

โดยที่ $f_i(x)$ เป็นค่าฟังก์ชันวัตถุประสงค์ที่ i ของคำตอบที่หาได้

$f_i(y)$ เป็นค่าฟังก์ชันวัตถุประสงค์ที่ i ของคำตอบที่แท้จริง

เมื่อ $i = 1,2,...,k$ (ในที่นี้กำหนดให้ $k = 2$)

เมื่อได้ค่าที่น้อยที่สุดในแต่ละเซตคำตอบที่หาได้กับเซตคำตอบที่แท้จริงแล้ว จะหารด้วยจำนวนคำตอบทั้งหมดในเซตคำตอบที่แท้จริง ถ้าค่าตัววัดสมรรถนะนี้มีค่าน้อยเข้าใกล้ศูนย์ จะถือว่ากลุ่มคำตอบที่ได้เป็นกลุ่มคำตอบที่ได้เป็นกลุ่มคำตอบที่มีคุณภาพใกล้เคียงกับกลุ่มคำตอบที่ดีที่สุดที่แท้จริง

ตัววัดที่สอง : อัตราส่วนของคำตอบที่ไม่คำตอบใดข่มคำตอบนี้ หรืออัตราส่วนของจำนวนกลุ่มคำตอบที่หาได้เทียบเท่ากับกลุ่มคำตอบที่แท้จริง

$$R_{NDS}(S_j) = \frac{|S_j - \{x \in S_j \mid \exists y \in : | y \prec x\}|}{|S_j|} \tag{4.42}$$

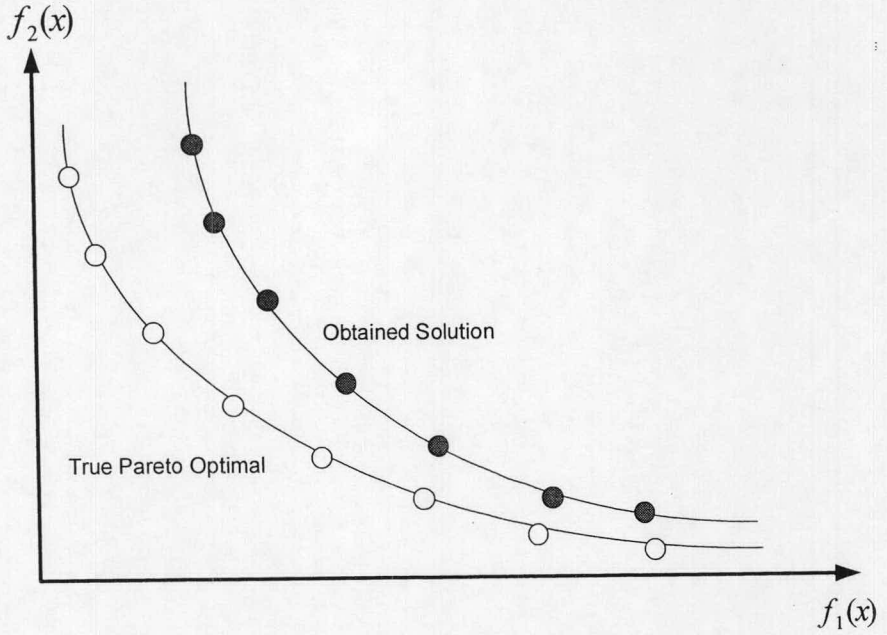
โดยที่ S_j คือ เซตคำตอบที่ j เมื่อ $j = 1,2,...,J$

S คือ การร่วมกันของ j เซตคำตอบ $S = S_1 \cup S_2 ... \cup S_J$

x คือ เซตคำตอบที่หาได้

y คือ เซตคำตอบที่แท้จริง

จากสมการที่ (4.42) สัญลักษณ์ หมายถึง $y \prec x$ คำตอบ x ถูกข่มด้วยคำตอบ y และตัวเศษของสมการนั้น หมายถึงถ้าคำตอบ x แย่กว่าคำตอบอื่นๆใน y จะนำคำตอบนี้ออกไปในเซตคำตอบ S_j นั่นคือตัววัดนี้จะวัดอัตราส่วนของคำตอบใน S_j ที่ไม่ถูกข่มจากคำตอบอื่นใน S ซึ่งถ้าค่าอัตราส่วนในแต่ละเซตคำตอบมีค่าสูง แสดงว่ามีคำตอบที่หาได้ x ในเซต S_j ถูกข่มด้วยคำตอบที่แท้จริง y น้อย และถือว่ามีคุณภาพดีใกล้เคียงกับกลุ่มคำตอบที่แท้จริง



รูปที่ 4.36 การคำนวณระยะทางจากกลุ่มคำตอบที่หาได้และกลุ่มคำตอบที่แท้จริง

Deb, Pratap, Agarwal และ Meyarivan (2002) ในงานวิจัยนี้ได้ใช้ตัววัดสมรรถนะกลุ่มคำตอบที่ดีที่สุดสองวิธีเช่นกัน และในการบรรลุเป้าหมายของการค้นหากลุ่มคำตอบนั้น ไม่สามารถใช้ตัววัดสมรรถนะเพียงหนึ่งตัวได้ ในที่นี้ได้วัดสมรรถนะกลุ่มคำตอบในด้านการเข้าสู่กลุ่มคำตอบที่แท้จริง และการรักษาความหลากหลายให้กับกลุ่มคำตอบที่ดีที่สุด โดยในงานวิจัยนี้ได้กำหนดตัววัดสมรรถนะกลุ่มคำตอบที่เกิดขึ้นจาก NSGA II

ตัววัดที่หนึ่ง : ตัววัสมรรถนะการลู่เข้าสู่กลุ่มคำตอบที่แท้จริง

มีขั้นตอนดังนี้

1. คำนวณ Euclidean Distance ที่น้อยที่สุดของแต่ละจุดในกลุ่มคำตอบที่ดีที่สุดแท้จริงกับกลุ่มคำอบที่หาได้

2. คำนวณระยะทางเฉลี่ยที่หาได้ ด้วยการหารจำนวนคำตอบที่ประมาณกลุ่มคำตอบที่แท้จริง

จะเห็นได้ว่าตัววัดสมรรถนะนี้มีลักษณะเช่นเดียวกับตัววัดสมรรถนะ $D1_R$ ที่เป็นการหาการลู่เข้าของคำตอบที่หาได้กับคำตอบลที่แท้จริง ถ้าตัววัดสมรรถนะนี้มีค่าน้อยที่เข้าใกล้ศูนย์ แสดงว่ากลุ่มคำตอบที่หาได้ ลู่เข้าสู่กลุ่มคำตอบที่แท้จริง

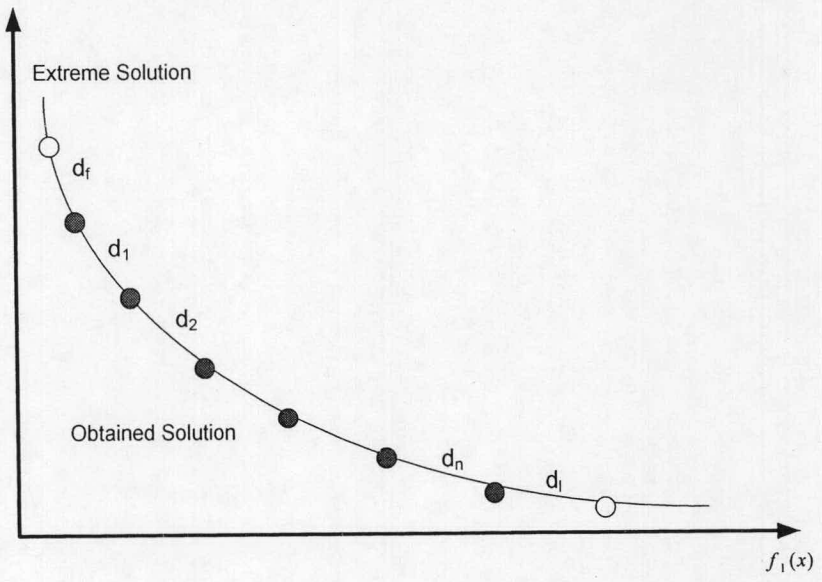
ตัววัดที่สอง : การกระจายของกลุ่มคำตอบที่หาได้

มีขั้นตอนดังนี้

1. คำนวณ Euclidean distance d_i ของคำตอบที่อยู่ต่อเนื่องกัน นาเซตคำตอบที่ดีที่สุดที่หามาได้ เมื่อ $i = 1,2,...,N-1$ และ N คือจำนวนคำตอบที่หาได้ และกำหนดให้ d_f และ d_l เป็นระยะห่างของคำตอบกปลายสุดของขอบเขตกลุ่มคำตอบที่หาได้

2. คำนวณค่าเฉลี่ยของระยะทางระหว่างคำตอบที่อยู่ติดกัน ($\bar{d}$) และแทนค่าระยะทางที่กำในสมการ (4.43)

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \bar{d} \right|}{d_f + d_l + (N-1)\bar{d}} \tag{4.43}$$



รูปที่ 4.37 การคำนวณหาค่าการกระจายของกลุ่มคำตอบที่หาได้

และถ้าตัววัดสมรรถนะนี้มีค่าน้อย แสดงว่ากลุ่มคำตอบที่หาได้มีการกระจายในลักษณะสม่ำเสมอ นั่นคือมีคำตอบกระจายอย่างสม่ำเสมอตลอดเส้นขอบเขตกลุ่มคำตอบที่ดีที่สุดและจัดได้ว่ามีลักษณะการกระจายที่ดี

4.13 สรุปท้ายบท

เนื้อหาที่ได้กล่าวไปในบทนี้เป็นทฤษฎีการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ ซึ่งเป็นพื้นฐานในการนำวิธีต่างๆ มาใช้ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ โดยความแตกต่างของปัญหาที่พิจารณาฟังก์ชันวัตถุประสงค์เดียว และหลายฟังก์ชันวัตถุประสงค์ จะมีความแตกต่างในด้านผลลัพธ์ของคำตอบ นอกจากนี้การกำหนดค่าความแข็งแรง (Fitness Assignment)) ด้วยการใช้วิธีการรวมฟังก์ชันโดยอาศัยน้ำหนัก (Weighted Sum Approach) มาใช้ในการแก้ปัญหาการหาค่าเหมาะสมที่สุดที่มีหลายวัตถุประสงค์ ยังถือว่าไม่ตอบสนองกับรูปแบบปัญหาดังกล่าวมากนัก ส่วนการใช้วิธีเชิงกลุ่มที่ดีที่สุด (Pareto-Based Approach) จะทำให้ผลลัพธ์ที่ได้ตอบสนองกับผู้ตัดสินใจมากขึ้น ทำให้มีการเลือกผลลัพธ์ตามลักษณะสถานกาณ์ต่าง ๆ ได้ เนื่องจากคำตอบที่ได้มีลักษณะเป็นกลุ่มคำตอบที่หลากหลาย แม้ว่าการแก้ปัญหาดังกล่าวจะเป็นวิธีทางฮิวริสติกแต่ด้วยการคำนวณที่ไม่ยุ่งยาก และไม่ซับซ้อนประกอบกับใช้เวลาในการคำนวณไม่มากนัก จึงทำให้วิธีดังกล่าวได้รับความนิยมจากนักวิชาการ วิธีอัลกอริทึมที่เป็นที่นิยมอย่างนึงในปัจจุบัน คือ อัลกอริทึมเจนเนติกอัลกอริทึมแบบ NSGA-II เป็นอัลกอริทึมที่มีการอาศัยทฤษฎีในการถ่ายทอดลักษณะต่างๆทางกรรมพันธ์ไปสู่ยังลูกหลาน คำตอบที่ได้จะมีความหลากหลายและเมื่อนำมาประยุกต์ใช้ในการแก้ไขปัญหาการจัดสมดุลสายการประกอบจะช่วยให้ได้คำตอบที่ดีและประหยัดเวลาในคำนวณหาคำตอบที่เหมาะสม