แบบจำลองสองวัตถุประสงค์สำหรับปัญหาร่วมการจัดตารางการผลิตแบบขนาน เมื่อพิจารณาขนาดสั่งผลิตและลำดับเวลาติดตั้งเครื่องจักร

นายวิชัย ศรีสุรักษ์

มหาวิทยาลัยเทคโนโลยีสุรนารี

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต

สาขาวิชาวิศวกรรมการผลิต

มหาวิทยาลัยเทคโนโลยีสุรนารี

ปีการศึกษา 2562

A BI-OBJECTIVE MODEL OF JOINT PARALLEL MACHINE SCHEDULING WITH LOTSIZING PRODUCTION AND SEQUENCE DEPENDENT SETUPS

Wichai Srisuruk

มหาวิทยาลัยเทคโนโลยีสุรนารี

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Manufacturing Engineering

Suranaree University of Technology

Academic Year 2019

แบบจำลองสองวัตถุประสงค์สำหรับปัญหาร่วมการจัดตารางการผลิตแบบขนาน เมื่อพิจารณาขนาดสั่งผลิตและลำดับเวลาติดตั้งเครื่องจักร

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรปริญญาดุษฎีบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

(อ. ดร.ศักดิ์ดา คำจันทร์)
ประธานกรรมการ

(ผศ. ดร.กัญชลา สุดตาชาติ)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

(ผศ. ดร.ปรเมศวร์ ห่อแก้ว)
กรรมการ

(อ. ดร.สมศักดิ์ ศิวดำรงพงศ์)
กรรมการ

(อ. ดร.ธีทัต ดลวิชัย)
กรรมการ

(รศ. ร.อ. ดร.กนต์ธร ชำนิประศาสน์)
รองอธิการบดีฝ่ายวิชาการและพัฒนาความเป็นสากล

(รศ. ดร.พรศิริ จงกล)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีสุรนารี

วิชัย ศรีสุรักษ์ : แบบจำลองสองวัตถุประสงค์สำหรับปัญหาร่วมการจัดตารางการผลิตแบบขนาน เมื่อพิจารณาขนาดสั่งผลิตและลำดับเวลาติดตั้งเครื่องจักร (A BI-OBJECTIVE MODEL OF JOINT PARALLEL MACHINE SCHEDULING WITH LOTSIZING PRODUCTION AND SEQUENCE DEPENDENT SETUPS) อาจารย์ที่ปรึกษา: ผู้ช่วยศาสตราจารย์ ดร.กัญชลา สุดตาชาติ, 156 หน้า.

งานวิจัยนี้ศึกษาการจัดตารางการผลิต ภายใต้เงื่อนไขต่าง ๆ ดังนี้ เครื่องจักรมีการจัดเรียงแบบขนาน, ลำดับการผลิตมีผลต่อเวลาติดตั้งเครื่องจักร, พิจารณาขนาดการสั่งผลิตขั้นต่ำ และมีข้อจำกัดด้านการทำงานของเครื่องจักร กล่าวคืองานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ โดยวัตถุประสงค์ในการจัดตารางงานเพื่อทำให้ค่าใช้จ่ายรวม และเวลาในการปิดงานของระบบ มีค่าต่ำที่สุด โดยใช้วิธีโปรแกรมจำนวนเต็มผ่านวิธีแบบจำลองทางคณิตศาสตร์ และวิธีปัญญาเชิงคำนวน

ผลการศึกษาพบว่า การค้นคำตอบด้วยแบบจำลองปัญหาทางคณิตศาสตร์ทำงานได้ถูกต้องตามวัตถุประสงค์ที่กำหนดไว้ โดยคำตอบที่ได้ คือ จำนวนสั่งผลิตในแต่ละรอบงานแต่ละเครื่องจักร และลำดับงานในการผลิตแต่ละรอบงาน การค้นคำตอบด้วยวิธีปัญญาเชิงคำนวนได้ใช้แนวคิดของสองวิธีมาเปรียบเทียบกัน คือ การค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว และการค้นหาแบบตาบูประยุกต์ เมื่อทดสอบกับปัญหาอ้างอิงวิธีปัญญาเชิงคำนวณทั้งสองวิธีให้ผลการคำนวณแย่กว่าแบบจำลองปัญหาทางคณิตศาสตร์ 3-5% และสามารถลดเวลาการค้นคำตอบจากหนึ่งร้อยชั่วโมงเป็นหนึ่งนาที ทดสอบกับปัญหาขนาดเล็กวิธีปัญญาเชิงคำนวณทั้งสองวิธีให้คำตอบใกล้เคียงกันและใช้เวลาใกล้เคียงกันด้วย คำตอบเมื่อเทียบกับแบบจำลองปัญหาทางคณิตศาสตร์จะแย่กว่าไม่เกิน 10% ส่วนกรณีทดสอบกับปัญหาขนาดใหญ่วิธีปัญญาเชิงคำนวณทั้งสองวิธีให้คำตอบ และใช้เวลาใกล้เคียงกัน

การจัดตารางแบบสองประสงค์ ได้นำเสนอรูปแบบ ขั้นตอนการทำงาน ของวิธีปัญญาเชิงคำนวณ โดยเสนอไว้ 3 แบบ คือ การทำงานแบบเทียม, แบบขนาน และแบบอนุกรม ทั้งสามแบบให้คำตอบใกล้เคียงกัน ซึ่งเป็นผลจากการเลือกวัตถุประสงค์หลักและวัตถุประสงค์รองที่ส่งผลต่อคำตอบไปในทางเดียวกัน

สาขาวิชา วิศวกรรมการผลิต ลายมือชื่อนักศึกษา _____

ปีการศึกษา 2562 ลายมือชื่ออาจารย์ที่ปรึกษา _____

WICHAI SRISURUK : A BI-OBJECTIVE MODEL OF JOINT PARALLEL MACHINE SCHEDULING WITH LOTSIZING PRODUCTION AND SEQUENCE DEPENDENT SETUPS. THESIS ADVISOR : ASST. PROF. KANCHALA SUDTACHART, Ph.D., 156 PP.

PARALLEL MACHINE/SEQUENCE DEPENDENT SETUP/MINIMUM LOT SIZE MACHINE ELIGIBILITY/ADAPTIVE NEAREST NEIGHBOUR SEARCH/ MODIFIED TABU SEARCH

This research focuses on solving scheduling problems under various conditions, i.e., parallel machine configuration, sequence dependent setup, minimum lot size, and machine eligibility, subject to assignable job. The main objective was to devise a scheduling, whose total cost and completion time were minimized. To this end, an integer programming by using mathematical model and computational intelligent approaches were employed.

The study revealed that the solution obtained by mathematical model, which consisted of periodical per-machine production sizes and their production sequences, was valid and met the specified objective. By using computational intelligent approaches, the results obtained by an Adaptive Neighbor Search (ANS) and Modified Tabu Search (MTS) were compared. Validated against the reference problem, ANS and MTS yielded worse results than did the mathematical model by 3-5%. However, with ANS and MTS the computing time was reduced from a hundred hours to only one minute. Tested with small problems both ANS and MTS gave similar solutions and took similar amount of computing time. Compared with that obtained from a

mathematical model, their results were worse by no more than 10%. Similarly, for big problems, ANS and MTS took similar amount of computing time.

Herein, the process and computational solution of the proposed bi-objective scheduling were presented in 3 schemes, i.e., pseudo, serial, and parallel operations. All three schemes gave similar solutions. This is due to both main and second objectives led to solutions in accordance.

มหาวิทยาลัยเทคโนโลยีสุรนารี

School of Manufacturing Engineering Student's Signature _____

Academic Year 2019 Advisor's Signature _____

กิตติกรรมประกาศ

วิทยานิพนธ์นี้ดำเนินการสำเร็จลุล่วงด้วยดี เนื่องจากได้รับความช่วยเหลืออย่างดียิ่ง ทั้งด้านวิชาการและด้านการดำเนินงานวิจัยจากบุคคล และกลุ่มบุคคลต่าง ๆ ได้แก่

ผู้ช่วยศาสตราจารย์ ดร.กัญชลา สุดตาชาติ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้ให้คำปรึกษา แนะนำ และแนะแนวทางอันเป็นประโยชน์ยิ่งต่องานวิจัย รวมถึงได้ช่วยตรวจทาน และแก้ไขรายงานวิทยานิพนธ์เล่มนี้จนทำให้มีความสมบูรณ์ยิ่งขึ้น รวมทั้งเป็นกำลังใจ และเป็นแบบอย่างที่ดีในการดำเนินชีวิตหลาย ๆ ด้านให้กับผู้วิจัยเสมอมา

คณาจารย์ประจำสาขาวิชาวิศวกรรมคอมพิวเตอร์ วิศวกรรมไฟฟ้า วิศวกรรมโทรคมนาคม และวิศวกรรมการผลิต มหาวิทยาลัยเทคโนโลยีสุรนารีทุกท่าน ที่กรุณาให้คำปรึกษาและกำลังใจอย่างดีมาโดยตลอด

สุดท้ายนี้ ผู้วิจัยขอขอบคุณ ครู อาจารย์ผู้สอนทุกท่านที่ประสิทธิ์ประสาทความรู้ทางด้านต่าง ๆ ทั้งในอดีตและปัจจุบัน และขอกราบขอบพระคุณ บิดา มารดา รวมถึงญาติพี่น้องของผู้วิจัยทุกท่านที่ให้ความรัก กำลังใจ การอบรมเลี้ยงดู และให้การสนับสนุนทางด้านการศึกษาอย่างดียิ่งมาโดยตลอด จนทำให้ผู้วิจัยประสบความสำเร็จในชีวิตเรื่อยมา

วิชัย ศรีสุรักษ์

มหาวิทยาลัยเทคโนโลยีสุรนารี

สารบัญ

หน้า

บทคัดย่อ (ภาษาไทย) ..... ก
บทคัดย่อ (ภาษาอังกฤษ) ..... ข
กิตติกรรมประกาศ ..... ง
สารบัญ ..... จ
สารบัญตาราง ..... ซ
สารบัญรูป ..... ฌ
บทที่
1 บทนำ
1.1 ความเป็นมาและความสำคัญของปัญหา 1
1.2 วัตถุประสงค์ของการวิจัย 2
1.3 ขอบเขตของการวิจัย 2
1.4 ประโยชน์ที่คาดว่าจะได้รับ 2
1.5 การจัดรูปเล่มวิทยานิพนธ์ 3
2 ปริทัศน์วรรณกรรมและทฤษฎีที่เกี่ยวข้อง
2.1 บทนำ 5
2.2 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 5
2.3 ทฤษฎีที่เกี่ยวข้อง 9
2.3.1 การจัดการตารางการผลิต 9
2.3.2 แบบจำลองทางคณิตศาสตร์การจัดตารางการผลิตด้วยเครื่องจักรขนาน 16
2.3.3 การหาคำตอบที่เหมาะสม 18
2.3.4 การหาค่าที่เหมาะสมที่สุดที่มีหลายวัตถุประสงค์ 37
2.4 สรุป 34
3 การจัดตารางการผลิตโดยใช้แบบปัญหาทางคณิตศาสตร์
3.1 บทนำ 35

สารบัญ (ต่อ)

หน้า

3.2 พารามิเตอร์และตัวแปรตัดสินใจ 35
3.3 แบบปัญหาทางคณิตศาสตร์ 37
3.3.1 สมการเป้าหมาย (Objective function) 37
3.3.2 สมการเงื่อนไข (Constraints) 37
3.4 การทดสอบและผลการทดสอบแบบปัญหาทางคณิตศาสตร์ 40
3.5 สรุป 54
4 การจัดตารางการผลิตโดยวิธีปัญญาเชิงคำนวณ
4.1 บทนำ 55
4.2 การคำนวณค่าฟังก์ชันวัตถุประสงค์ 55
4.2.1 การสร้างรูปแบบการทำงานทั้งหมด 55
4.2.2 คำนวณหาจำนวนอ้างอิงในการผลิต 56
4.2.3 คำนวณหา Quantity Array 57
4.2.4 คำนวณหา Sequence Array 59
4.2.5 คำนวณหา Objective X, Y, Z 61
4.3 วิธีปัญญาเชิงคำนวณ 64
4.3.1 วิธีค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว 64
4.3.2 การค้นหาแบบตาบูประยุกต์ 65
4.4 ขนาดของปัญหาและข้อมูลที่ใช้ทดสอบ 66
4.5 การทดสอบและผลการทดสอบวิธีปัญญาเชิงคำนวณ 67
4.5.1 การทดสอบด้วยชุดข้อมูลอ้างอิง 67
4.5.2 การทดสอบด้วยชุดข้อมูลขนาดเล็ก 70
4.5.3 การทดสอบด้วยชุดข้อมูลขนาดใหญ่ 74
4.6 สรุป 80
5 การจัดตารางการผลิตแบบสองวัตถุประสงค์
5.1 บทนำ 81
5.2 รูปแบบการทดสอบ 81
5.2.1 Model-0: การจัดตารางงานแบบวัตถุประสงค์เดียว 81

สารบัญ (ต่อ)

หน้า

5.2.2 Model-1: การจัดตารางงานแบบสองวัตถุประสงค์เทียม....85
5.2.3 Model-2: การจัดตารางงานแบบสองวัตถุประสงค์แบบขนาน....89
5.2.4 Model-3: การจัดตารางงานแบบสองวัตถุประสงค์แบบอนุกรม....92
5.3 ผลการทดสอบ....95
5.4 สรุป....100
6 สรุปและข้อเสนอแนะ
6.1 สรุป....101
6.2 ข้อเสนอแนะ....101
รายการอ้างอิง....102
ภาคผนวก
ภาคผนวก ก ผลการทดสอบของโปแกรม....105
ภาคผนวก ข โปรแกรมที่พัฒนาขึ้น....120
ประวัติผู้เขียน....156

มหาวิทยาลัยเทคโนโลยีสุรนารี

สารบัญตาราง

ตารางที่ **หน้า**

2.1 ตัวอย่างข้อมูลของงานที่ต้องการเวลาปรับตั้งเครื่องจักร (หน่วย : ชั่วโมง) 15

2.2 ตัวอย่างผลรวมของเวลาปรับตั้งเครื่องจักรเมื่อมีลำดับการทำงานต่างกัน (หน่วย : ชั่วโมง) 16

3.1 ผลการทดสอบแบบจำลองคณิตศาสตร์ 53

4.1 การกำหนดค่าของตัวแปรต่าง ๆ 67

4.2 ผลการคำนวณด้วยข้อมูลปัญหาอ้างอิง 68

4.3 จำแนกตัวอย่างงานที่ใช้ทดสอบและจำนวนรูปแบบการจัดตารางงาน 70

4.4 ผลการทดสอบช-แสดงเฉพาะกรณี Test01-05J02M06P ทั้งหกชุดข้อมูล 71

4.5 ผลการทดสอบ-แสดงทุกกรณี นำเสนอกรณีละสองชุดข้อมูล 71

4.6 จำแนกตัวอย่างงานที่ใช้ทดสอบและจำนวนรูปแบบการจัดตารางงาน 74

4.7 ผลการทดสอบ-แสดงเฉพาะกรณี Test01-10J04M06P 75

4.8 ผลการทดสอบ-แสดงค่าเฉลี่ยของค่าใช้จ่ายละเวลาของแต่ละกรณีทดสอบ 76

5.1 ผลการคำนวณเพื่อหาค่าอัตราการใช้สอยทรัพยากรเฉลี่ยของการจัดตารางงาน 83

5.2 การคำนวณเพื่อหาแผนการผลิตจากค่า best.ZU 84

5.3 ผลการทดสอบ Pseudo Bi-Objective 95

5.4 ผลการทดสอบ Parallel Bi-Objective 95

5.5 ผลการทดสอบ Serial Bi-Objective 96

มหาวิทยาลัยเทคโนโลยีสุรนารี

สารบัญรูป

รูปที่ **หน้า**

2.1 เวลาว่างงานจองเครื่องจักร 10
2.2 เวลาเริ่มงาน เวลาดำเนิน เวลาเสร็จงานและเวลาไหลของงาน 10
2.3 เวลาปิดงานของระบบ 10
2.4 งาน 1 2 และ 3 เสร็จก่อนกำหนด เสร็จช้ากว่ากำหนด และเสร็จตามกำหนด ตามลำดับ 11
2.5 การสุ่มเลือกคำตอบเริ่มต้น S0 และการผลิตค่าใกล้เคียงแบบสุ่ม 27
2.6 การเปรียบเทียบค่าใกล้เคียงและการปรับค่าคำตอบเริ่มต้นใหม่ 27
2.7 ผลิตค่าใกล้เคียงแบบสุ่มรอบใหม่ 28
2.8 กลไกย้อนรอยการค้นหาและกลไกปรับรัศมีการค้นหา 29
2.9 พฤติกรรมของนกที่บินเป็นฝูง 30
3.1 ตัวแปรที่ใช้ในการคำนวณในแบบปัญหา 35
3.2 ผลการทำงานด้วยโปรแกรม LINGO 11.0 43
3.3 การวิเคราะห์คำตอบของโปรแกรมด้วยการแจงใน Excel 44
3.4 แผนภูมิแสดงการจัดตารางงาน กรณี 06J02M03P 45
3.5 แผนภูมิแสดงจำนวนงานที่ต้องการ จำนวนคงเหลือ และจำนวนที่ผลิตแต่ละเครื่องจักรแบ่งตามคาบเวลา 46
3.6 ผลการทำงานด้วยโปรแกรม LINGO 11.0 ของปัญหา 9J-2M-3P 50
3.7 แผนภูมิการจัดตารางงาน กรณี 09J02M03P 51
3.8 แผนภูมิ จำนวนงาน (ที่ต้องการ คงเหลือ และที่ผลิตแต่ละเครื่องจักร) แบ่งตามคาบเวลา กรณี 09J02M03P 52
3.9 ผลการทำงานด้วยโปรแกรม LINGO 11.0 ของปัญหา 6J-2M-6P 53
4.1 วิธีการค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว 64
4.2 การค้นหาแบบตาบูประยุกต์ (Modify Tabu Search) 65
4.3 ผลการทดสอบกับข้อมูลอ้างอิง-กรณี 06J02M03P 69
4.4 ผลการทดสอบกับข้อมูลอ้างอิง กรณี 6J2M3P, 9J2M3P, 6J2M6P 69

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.5 ค่าฟังก์ชันวัตถุประสงค์เมื่อทดสอบกับข้อมูลขนาดเล็ก 72
4.6 เปอร์เซ็นต์ผลต่างค่าฟังก์ชันวัตถุประสงค์เมื่อทดสอบกับข้อมูลขนาดเล็ก 72
4.7 เวลาในการทำงานของโปรแกรมเมื่อทดสอบกับข้อมูลขนาดเล็ก-ภาพรวม 73
4.8 เวลาในการทำงานของโปรแกรมเมื่อทดสอบกับข้อมูลขนาดเล็ก-ภาพขยาย 73
4.9 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Test01 -- 10J04M06P 77
4.10 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Test04 -- 15J04M06P 77
4.11 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Index = Test06 -- 15J06M09P 78
4.12 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Index = Test13 -- 30J06M06P 78
4.13 เปรียบเทียบค่าฟังก์ชันวัตถุประสงค์เฉลี่ยของแต่ละกรณีทดสอบ 79
4.14 เปรียบเทียบเวลาในการทำงานของโปรแกรมเฉลี่ยของแต่ละกรณีทดสอบ 79
5.1 แผนภาพขั้นตอนการคำนวณฟังก์ชันวัตถุประสงค์ 81
5.2 ตารางการผลิต กรณีทดสอบ-t0501 82
5.3 ตารางการผลิต กรณีทดสอบ-t0502 82
5.4 ตำแหน่งของแผนการผลิตที่ดีสุด best.Z, best.U และ best.ZU 85
5.5 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์ แบบสองวัตถุประสงค์เทียม 85
5.6 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective 87
5.7 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective 87
5.8 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective 88
5.9 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective 88
5.10 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์ แบบสองวัตถุประสงค์ขนาน 89

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

5.11 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective 90
5.12 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective 90
5.13 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective 91
5.14 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective 91
5.15 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์ แบบสองวัตถุประสงค์อนุกรม 92
5.16 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective 93
5.17 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective 93
5.18 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective 94
5.19 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective 94
5.20 เวลาที่ใช้ในการคำนวณ 96
5.21 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Pseudo Bi-Objective 97
5.22 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Parallel Bi-Objective 97
5.23 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Serial Bi-Objective 98
5.24 เปรียบเทียบคำตอบ z-score ของทั้งสามวิธีค้นคำตอบ 99

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันโรงงานอุตสาหกรรมส่วนใหญ่มุ่งเน้นไปที่การลดค่าใช้จ่ายในการดำเนินงาน จึงต้องมีการจัดตารางการผลิตที่เหมาะสมเพื่อทำให้ค่าใช้จ่ายจากงานที่ผลิตเสร็จก่อนและล่าช้ากว่ากำหนดเกิดขึ้นน้อยที่สุด รูปแบบการผลิตที่ใช้กันอย่างกว้างขวางในโรงงานอุตสาหกรรม คือ ระบบการผลิตของเครื่องจักรหลายเครื่องจักรที่ทำงานอย่างเดียวกันแต่ใช้เวลาดำเนินการแตกต่างกัน เรียกว่า เครื่องจักรขนานที่ไม่สัมพันธ์กัน (Unrelated Parallel Machines) เช่น การผลิตในโรงกลึง โรงเลื่อยไม้ เป็นต้น เมื่อมีการเริ่มใช้งานเครื่องจักร หน่วยซ่อมบำรุงต้องใช้เวลาในการปรับตั้งเครื่องจักรเพื่อเตรียมความพร้อม หรือเมื่อมีการเปลี่ยนอุปกรณ์บางอย่างสำหรับการทำงานใหม่ เช่น การล้างเครื่องผสมสี การล้างสารเคมี การใส่แม่พิมพ์ ซึ่งกิจกรรมดังกล่าวนี้อาจมีค่าใช้จ่ายในการดำเนินการเกิดขึ้นที่การเตรียมเครื่องจักรเพื่อใช้งาน การเตรียมความพร้อมของเครื่องจักรบางงานอาจเป็นค่าคงที่ค่าหนึ่ง (Sequence-Independent setup) แต่บางงานค่าการเตรียมขึ้นกับลำดับงานก่อนหน้านี้ (Sequence-dependent setup time) นอกจากนี้ยังมีข้อจำกัดอื่น เช่น งานบางงานไม่สามารถดำเนินการบนเครื่องจักรบางเครื่องได้ โดยมากมักเกิดกับเครื่องจักรที่มีหน้าที่เหมือนกันแต่บางเครื่องทำงานพิเศษได้ เช่น งานกัด ถ้ากัดหยาบใช้ได้ทุกเครื่อง แต่กัดละเอียดใช้ได้บางเครื่องเท่านั้น ปัญหาที่อาจจะเกิดขึ้นอีกอย่างคือเมื่อมีการจัดตารางงานไปแล้วแต่เกิดเหตุเครื่องจักรไม่สามารถทำงานได้ จึงจำเป็นต้องมีแผนการดำเนินการสำรองเมื่อเจอเหตุการณ์ดังกล่าว

งานวิจัยเรื่องนี้แบ่งออกเป็นสองส่วน ส่วนแรกจะเสนอวิธีการจัดตารางการผลิตสำหรับโรงงานอุตสาหกรรมที่มีระบบการผลิตแบบเครื่องจักรขนานที่ไม่สัมพันธ์กันโดยมีเวลาปรับตั้งเครื่องจักรขึ้นอยู่กับลำดับงาน และมีแนวคิดของระบบการผลิตแบบทันเวลาพอดี ในการจัดตารางการผลิตต้องรู้ข้อมูลของงานก่อนการวางแผนการผลิต เช่น เวลาดำเนินการปรับตั้งเครื่องจักร ค่าใช้จ่ายจากการปรับตั้งเครื่อง อัตราค่าใช้จ่ายจากงานที่ผลิตเสร็จก่อนกำหนด เป็นต้น และข้อมูลเหล่านี้เป็นเลขจำนวนเต็ม ถ้างานเริ่มดำเนินการบนเครื่องจักรใดแล้วงานนั้นต้องดำเนินการจนเสร็จโดยไม่มีการแทรกงานหรือยกเลิก งานและงานแต่ละงานมีกำหนดส่งมอบงานแตกต่างกัน (Distinct Due Date) ส่วนที่สองเป็นการนำเสนอการจัดตารางแบบสองวัตถุประสงค์เพื่อนำเสนอทางเลือกในการจัดตารางการผลิต วัตถุประสงค์แรกคือการเน้นการลดค่าใช้จ่ายในการผลิตและอีกวัตถุประสงค์คือประสิทธิภาพการใช้งานเครื่องจักร (Machine Utilization) โดยการเปรียบเทียบระหว่างเวลาเดิน

เครื่องจักรสุทธิ (Net Operation Time) ที่ได้จากจำนวนชิ้นงานที่ผลิตได้ในเวลามาตรฐานของเครื่องจักร เทียบกับเวลาเดินเครื่องจักรที่ใช้จริง (Operation Time) การทำงานในทั้งสองส่วนจะใช้วิธีปัญญาเชิงคำนวณในการทดสอบกับปัญหาที่มีขนาดใหญ่ขึ้น

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 เพื่อหาตารางการผลิตและขนาดสั่งผลิต (Lot Size) แบบที่ดีที่สุดที่ทำให้ค่าใช้จ่ายและเวลารวมของระบบต่ำสุด ซึ่งประกอบด้วยค่าใช้จ่ายจากงานที่ผลิตเสร็จก่อนกำหนด ค่าใช้จ่ายจากงานที่ผลิตและส่งมอบล่าช้า และเวลาในการปรับตั้งเครื่องจักร ด้วยวิธีตัวแบบทางคณิตศาสตร์และวิธีปัญญาเชิงคำนวณ

1.2.2 เพื่อศึกษาประสิทธิภาพของวิธีปัญญาเชิงคำนวณที่ได้พัฒนาขึ้นเปรียบเทียบกับการหาคำตอบที่ดีที่สุด (Optimal Solution) โดยใช้แบบจำลองทางคณิตศาสตร์

1.2.3 เพื่อนำเสนอการจัดตารางงานและขนาดสั่งผลิต แบบ 2 วัตถุประสงค์ เพื่อเพิ่มทางเลือกในการกำหนดขั้นตอนในกระบวนการผลิต

1.3 ขอบเขตของการวิจัย

1.3.1 เป็นการจัดตารางการผลิตและกำหนดขนาดสั่งผลิต (Lot Size)

1.3.2 ข้อกำหนดในการจัดตารางการผลิต คือ

- เครื่องจักรเป็นเครื่องจักรขนานที่ไม่สัมพันธ์กัน (Unrelated parallel machine)
- การผลิตแบบที่ต้องการเวลาในการติดตั้ง (Sequence-dependent Setup)
- มีข้อจำกัดการผลิตของเครื่องจักร (Machine Eligibility)

1.3.3 วัตถุประสงค์ในการจัดตารางการผลิต คือ

- หาตารางการผลิตและขนาดสั่งผลิต
- หาจำนวนงานที่ผลิตในแต่ละคาบเวลา ของแต่ละเครื่องจักร
- หาเวลาที่ใช้ในการผลิตรวม เวลาเริ่มของแต่ละงาน เวลาปิดของแต่ละงาน
- หาค่าใช้จ่ายที่ต่ำที่สุด

1.3.4 การแก้ปัญหาด้วยวิธีปัญญาเชิงคำนวณ (พัฒนาด้วยโปรแกรม MATLAB)

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 ทราบถึงวิธีการจัดตารางการผลิตด้วยวิธีปัญญาเชิงคำนวณ โดยเปรียบเทียบค่าใช้จ่ายรวมและเวลาปิดงาน (Makespan) กับแบบจำลองทางคณิตศาสตร์ กำหนดขอบเขตอย่างง่ายน้อย

กว่า 15% และใช้เวลาในการรันโปรแกรมปัญญาเชิงคำนวณที่สมเหตุสมผล

1.4.2 ทราบถึงขั้นตอนในการแก้ปัญหาด้วยวิธีการที่นำเสนอนี้ เพื่อเป็นแนวทางในการพัฒนาวิธีจัดตารางการผลิตสำหรับปัญหาที่มีความซับซ้อนมากขึ้น

1.4.3 ทราบถึงประสิทธิภาพของวิธีปัญญาเชิงคำนวณที่ได้พัฒนาขึ้นเปรียบเทียบกับการหาคำตอบที่ดีที่สุด (Optimal Solution) โดยใช้แบบจำลองทางคณิตศาสตร์

1.4.4 ทราบถึงแนวทางการจัดตารางงานแบบหลายวัตถุประสงค์ด้วยวิธีปัญญาเชิงคำนวณ

1.5 การจัดรูปเล่มวิทยานิพนธ์

วิทยานิพนธ์นี้ประกอบด้วย 6 บท 3 ภาคผนวก ซึ่งมีรายละเอียดโดยย่อดังนี้

บทที่ 1 เป็นบทนำซึ่งจะกล่าวถึงความเป็นมาและความสำคัญของปัญหาวัตถุประสงค์และเป้าหมายของงานวิจัยวิทยานิพนธ์ตลอดจนขอบเขตและประโยชน์ที่คาดว่าจะได้รับจากงานวิจัยนี้

บทที่ 2 กล่าวถึงการสำรวจปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง เพื่อให้ทราบถึงแนวทางและระเบียบวิธีการวิจัยที่เกี่ยวข้อง โดยผลจากการสำรวจสืบค้น จะใช้เป็นแนวทางสำหรับการประยุกต์และพัฒนาเข้ากับงานวิจัยวิทยานิพนธ์นี้ และการนำเสนอทฤษฎีต่าง ๆ ที่เกี่ยวข้องกับงานวิจัยวิทยานิพนธ์ซึ่งประกอบไปด้วยหัวข้อหลัก ๆ ได้แก่ทฤษฎีที่เกี่ยวกับการจัดการตารางการผลิต แบบจำลองทางคณิตศาสตร์ และการหาคำตอบที่เหมาะสม

บทที่ 3 นำเสนอการจัดตารางการผลิตโดยใช้แบบปัญหาทางคณิตศาสตร์ การระบุขนาดของปัญหา การจัดตารางการผลิตสำหรับปัญหาขนาดเล็ก และผลการทดสอบการทำงาน

บทที่ 4 กล่าวถึงการจัดตารางการผลิตโดยวิธีปัญญาเชิงคำนวณได้แก่ การค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว (ANN:Adaptive Nearest Neighbor Search) การค้นหาแบบตาบูประยุกต์ (MTS:Modify Tabu Search) และการใช้วิธีหาค่าเหมาะสมที่สุดแบบกลุ่มอนุภาค (PSO:Particle Swarm Optimization) สำหรับปัญหาขนาดเล็ก และปัญหาขนาดใหญ่ พร้อมผลการทดสอบ

บทที่ 5 กล่าวถึงการจัดตารางการผลิตแบบหลายวัตถุประสงค์ โดยนำเสนอสามรูปแบบคือ การจัดตารางการผลิตแบบหลายวัตถุประสงค์เทียม การทำงานแบบขนาน และการทำงานแบบอนุกรม พร้อมผลการทดสอบการทำงาน

บทที่ 6 เป็นบทสรุปและข้อเสนอแนะพร้อมงานวิจัยที่จะดำเนินการต่อ

ภาคผนวก ก. รวบผลการทดสอบของโปแกรม

ภาคผนวก ข. โปรแกรมที่พัฒนาขึ้นในวิธีแบบปัญหาทางคณิตศาสตร์ และวิธีปัญญาเชิงคำนวณของ ANN, MTS, PSO และโปรแกรมที่พัฒนาขึ้นเพื่อจัดตารางแบบหลายวัตถุประสงค์

ภาคผนวก ค. เป็นการรวบรวมผลงานที่ได้รับการเผยแพร่ของงานวิจัยวิทยานิพนธ์ในขณะดำเนินการศึกษา

บทที่ 2
ปริทัศน์วรรณกรรมและทฤษฎีที่เกี่ยวข้อง

2.1 บทนำ

วัตถุประสงค์หลักของงานวิจัยนี้คือการจัดตารางการผลิตด้วยวิธีต่าง ๆ ภายใต้เงื่อนไขข้อกำหนดที่ต่างกัน โดยใช้แบบจำลองทางคณิตศาสตร์และการประมาณค่าด้วยวิธีปัญญาเชิงคำนวณ เพื่อหาตารางการผลิตแบบที่ดีที่สุดที่สามารถลดค่าใช้จ่ายโดยรวมของระบบ ซึ่งประกอบด้วย ค่าใช้จ่ายในการผลิต ค่าใช้จ่ายในการจัดเก็บจากงานที่ผลิตเสร็จก่อนกำหนด และค่าใช้จ่ายในการปรับตั้งเครื่องจักร ดังนั้นจึงมีความจำเป็นอย่างยิ่งที่จะต้องดำเนินการสำรวจปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้องเพื่อให้ทราบถึงแนวทางการวิจัยที่เคยมีการใช้งานมาก่อน จากผลการดำเนินงานและข้อเสนอแนะต่าง ๆ จากคณะนักวิจัยตั้งแต่อดีตเป็นต้นมา โดยใช้ฐานข้อมูลที่เป็นแหล่งสะสมรายงานวิจัยและวรรณกรรมที่เกี่ยวข้องทางด้านวิทยาศาสตร์และเทคโนโลยีอันได้แก่ ฐานข้อมูลจาก IEEE IEE และ Science Direct เป็นต้น ผลการสำรวจสืบค้นงานวิจัยดังกล่าวจะใช้เป็นแนวทางสำหรับการประยุกต์และพัฒนาเข้ากับวิทยานิพนธ์นี้ และการศึกษาทฤษฎีต่าง ๆ ที่เกี่ยวข้องกับงานวิจัยมีความสำคัญมากในการดำเนินงาน เพื่อเป็นพื้นฐานความรู้และความเข้าใจ ในส่วนนี้จะกล่าวถึงทฤษฎีที่เกี่ยวข้องกับงานวิจัยนี้ ซึ่งประกอบไปด้วยหัวข้อหลัก ๆ ได้แก่ทฤษฎีที่เกี่ยวกับการจัดการตารางการผลิต แบบจำลองทางคณิตศาสตร์การจัดตารางการผลิตด้วยเครื่องจักรขนาน การหาคำตอบที่เหมาะสมและการหาค่าที่เหมาะสมที่สุดแบบหลายวัตถุประสงค์ โดยจะกล่าวถึงส่วนที่เป็นประโยชน์หรือถูกกล่าวอ้างต่อการดำเนินงานวิจัยนี้ เพื่อความกระชับและชัดเจนของเนื้อหา

2.2 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

การนำเสนอปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง สามารถสรุปทฤษฎี หลักการ และวิธีการดำเนินงานวิจัยต่าง ๆ ที่ได้ทำการศึกษาปัญหาที่เกี่ยวกับการจัดตารางการผลิตด้วยวิธีต่าง ๆ ภายใต้เงื่อนไขข้อกำหนดที่ต่างกัน ตั้งแต่อดีตจนถึงปัจจุบันสามารถสรุปได้ดังนี้

กฤษณ์พล (2544) เสนอแบบจำลองการผลิตที่เหมาะสมกับการจัดตารางการทดสอบคุณสมบัติสี โดยใช้ดัชนีชี้วัดประสิทธิภาพ คือ เวลาปิดงานของระบบ (Makespan) ความล่าช้า (Tardiness) ความล่าช้ารวม (Total Tardiness) เวลาสาย (Lateness) เวลาไหลของงาน (Flow Time) เวลาไหลของงานรวม (Total Flow Time) จำนวนงานที่สายทั้งหมด (Total Number of Late Jobs)

เปรียบเทียบการจัดตารางทดสอบระหว่างงานประจำ แบบจำลองที่เสนอ และกฎการจ่ายงานหรือกฎการจัดลำดับความสำคัญ คือ กฎการจ่ายงานโดยคำนึงถึงกำหนดการส่งมอบ กฎการจ่ายงานโดยคำนึงถึงเวลาการดำเนินการน้อยที่สุด กฎการจ่ายงานโดยคำนึงถึงเวลาการดำเนินการมากที่สุด กฎการจ่ายงานโดยคำนึงถึงงานที่เข้ามาก่อนจะดำเนินการก่อน กฎการจ่ายงานโดยคำนึงถึงเวลาหย่อนน้อยที่สุด และฮิวริสติกการย้ายคอขวดจากโปรแกรมการจัดตารางการผลิต แล้วนำมาวิเคราะห์ผล ซึ่งพบว่าดัชนีชี้วัดประสิทธิภาพของแบบจำลองที่นำเสนอนั้นสามารถลด งานที่สายทั้งหมด (Total Number of Late Jobs) ลดลงจากเดิม 47% และเวลาการไหลของงานรวม (Total Flow Time) ลดลงจากเดิม 21% ดังนั้นการจัดตารางการทดสอบคุณสมบัติสีของแบบจำลองที่นำเสนอนั้นสามารถตอบสนองต่อความเหมาะสมของบริษัท

กัญชลา (2552) เสนอการจัดลำดับการผลิตเครื่องจักรทำงานแบบขนานกัน เพื่อให้เวลาในการผลิตของระบบมีค่าต่ำสุด และมีข้อจำกัดของเครื่องจักร คือ งานบางงานไม่สามารถดำเนินการบนเครื่องจักรบางเครื่องได้ รูปแบบทางคณิตศาสตร์ได้ถูกพัฒนาขึ้นจากพื้นฐานของปัญหาการจัดเส้นทางการขนส่ง โดยใช้โปรแกรมมิงเชิงจำนวนเต็มและได้นำเสนอฮิวริสติกสำหรับปัญหาขนาดใหญ่ จากผลการทดลองพบว่า ฮิวริสติกมีประสิทธิภาพดี โดยให้คำตอบที่แตกต่างจากคำตอบที่ดีที่สุดเท่ากับ 7.6%

Alex J. Ruiz-Torres, Francisco J. Lopez and Johny C. Ho (2007) การจัดตารางการผลิตเครื่องจักรแบบขนานที่ต่างกันของทรัพยากรเพื่อลดจำนวนงานสาย โดนจัดตารางการผลิตของเครื่องจักรแบบขนาน (พนักงาน) ที่มีความเร็วในการทำงานที่ต่างกันและขึ้นอยู่กับการจัดสรรพนักงานเข้าทำงาน มีวัตถุประสงค์เพื่อลดจำนวนงานสาย ซึ่งมีการวิเคราะห์ปัญหาออกเป็น 2 กรณี ดังนี้ กรณีที่ 1 คือ การจัดลำดับงานก่อนที่จะมอบหมายงานเข้าไปในแต่ละเครื่องจักร (PMFRS) กรณีที่ 2 คือ ไม่ได้จัดลำดับงานก่อนที่จะมอบหมายงานเข้าไปในแต่ละเครื่องจักร (UMFRS) ทั้ง 2 กรณีจะทำการทดลองการจัดตารางการผลิตโดยสูตรโปรแกรมทางคอมพิวเตอร์ในการแก้ปัญหา

Dong-Won Kim, Dong-Gil Na and F Fank Chen (2003) การจัดตารางการผลิตเครื่องจักรแบบขนานที่ไม่สัมพันธ์กันโดยมีวัตถุประสงค์เพื่อลดเวลาการติดตั้งเครื่องจักรและการลดความล่าช้าถ่วงน้ำหนักรวม โดยได้ศึกษาการจัดตารางการผลิตเครื่องจักรแบบขนานที่ไม่สัมพันธ์กันซึ่งมีลักษณะของงานคล้ายคลึงกัน โดยได้พัฒนาฮิวริสติกเพื่อแก้ปัญหาการจัดตารางการผลิต 4 วิธี คือ วิธีที่ 1 Earliest weighted due date วิธีที่ 2 Shortest weight processing time วิธีที่ 3 Two-level batch scheduling วิธีที่ 4 Simulated annealing method โดยมีวัตถุประสงค์เพื่อการลดความล่าช้าถ่วงน้ำหนักรวม

Emarh B. Edis and Ceyda Oguz (2012) การจัดตารางการผลิตเครื่องจักรแบบยืดหยุ่น โดยได้ศึกษาการจัดตารางการจัดตารางการผลิตเครื่องจักรแบบยืดหยุ่นที่ไม่สัมพันธ์กัน เครื่องจักรมี

ความสามารถด้าน การผลิตที่เหมือนกัน โดยสร้างแบบจำลองทางคณิตศาสตร์ 2 วิธี คือ วิธี PMFRS และวิธี UPMFRS โดยมีเป้าหมายเพื่อลดเวลาการปิดงานของระบบ

S. Polyakovskiy and R. Hallah (2014) การจัดตารางการผลิตแบบหลายขั้นตอนโดยคำนึงถึงงานล่าช้าที่ถูกถ่วงน้ำหนักที่เร็วที่สุดของเครื่องจักรแบบขนาน ทำการศึกษาถึงปัญหาของงานแต่ละงานจะใช้เวลาในการผลิตและวันกำหนดส่งมอบงานที่ต่างกัน จึงได้มีการสร้างแบบจำลองการจัดตารางการผลิตแบบวิธี Mixed Integer Programming and Solved (MASH) ซึ่งจะเป็นวิธีการแก้ปัญหาแบบหลายขั้นตอน แบบจำลองการจัดตารางการผลิตแบบวิธี MASH จะแบ่งการจัดตารางการผลิต ระบบ G หมายถึงงานที่งานที่ถูกมอบหมายเข้าเครื่องจักรแล้ว ระบบ M จะเป็นการจัดการการจัดตารางการผลิตในขั้นตอนที่สองหลังจากจัดตารางการผลิตของระบบ I และระบบ G แล้ว ซึ่งระบบ M จะเป็นตัวประเมิน ตัวตัดสินใจและเป็นตัวขับเคลื่อนให้ระบบบรรลุเป้าหมายการออกแบบจำลองการจัดตารางการผลิตแบบวิธี MASH จะสามารถแก้ปัญหางานที่เป็นคอขวดได้

Vahid Kayvanfar, G H. M. Komaki and Amin Aalaei, M. Zandieh (2014) การจัดตารางการผลิตของเครื่องจักรแบบขนานที่ไม่สัมพันธ์กันเพื่อลดความล่าช้ารวมและเวลางานเสร็จก่อนกำหนดส่งโดยสามารถควบคุมเวลาการผลิต งานวิจัยนี้จะใช้กฎการจัดตารางการผลิตของเครื่องจักรแบบขนานด้วยวิธี Just-In-Time (JIT) โดยมีวัตถุประสงค์เพื่อลดความล่าช้ารวมและเวลางานเสร็จก่อนกำหนัดส่ง ซึ่งค่าใช้จ่ายของการผลิตขึ้นอยู่กับความเร็วและช้าของเวลาปิดงานของระบบ การจัดตารางการผลิตเครื่องจักรแต่ละเครื่องจะกำหนดให้ไม่มีการแทรกงาน และมีอัตราการผลิตที่แตกต่างกัน โดยมีการสร้างแบบจำลองการจัดตารางการผลิตแบบวิธี Mixed Integer Programming (MIP) เพื่อแก้ปัญหาดังกล่าว แต่แบบจำลองนี้สามารถแก้ปัญหาระดับเล็กเท่านั้น ถ้าต้องการที่จะแก้ปัญหาระดับปานกลางถึงปัญหาระดับใหญ่ต้องใช้วิธีการจัดตารางผลิตแบบ Hybrid เข้ามาช่วย

Zhicong Zhang, Li Zhang, Na Li, Weiping Wang, Shouyan Zhong and Kaishun Hu (2012) การจัดตารางการผลิตเพื่อลดความล่าช้าเฉลี่ยที่ถูกถ่วงน้ำหนักของเครื่องจักรแบบขนานที่ไม่สัมพันธ์กัน การจัดตารางการผลิตของเครื่องจักรแบบขนานที่ไม่สัมพันธ์กัน ด้วยวิธี Reinforcement Learning (RL) ซึ่งจะสุ่มเวลาเริ่มต้นและเวลากำหนดส่งของแต่ละงานแล้วจัดตารางการผลิตโดยใช้เงื่อนไขปัญหา RL จากนั้นทำการทดลองถึงประสิทธิภาพการจัดตารางการผลิตซึ่งจะให้ผลที่ดีกว่าการจัดตารางการผลิตด้วยวิธี WSPT WMDD ATC และ WCOVERT

C. Meszaros (1997) นำการประยุกต์ใช้วิธีจุดภายใน (Interior Point Method: IPM) ในการแก้ปัญหาคำตอบแน่นอนที่เทียบเท่ากับปัญหากำหนดการเชิงเส้นไม่แน่นอนสองขั้นตอน (Two-stage Stochastic Linear Programming) โดยทำการพิสูจน์ว่าวิธีจุดภายในกับระบบที่มีการขยาย ซึ่งให้คำตอบของปัญหาที่ดีแต่นำไปใช้ได้ยากและมีประสิทธิภาพต่ำ พร้อมทั้งทำการเสนอวิธีใหม่สำหรับการแตกแยกย่อยของระบบที่มีการขยาย ด้วยปัญหาที่มีลักษณะเฉพาะเพื่อความรวดเร็วและ

ง่ายในการแก้ปัญหากำหนดการเชิงเส้นไม่แน่นอนสองขั้นตอนที่มีขนาดใหญ่ พร้อมทั้งทำการเปรียบเทียบผลการทดลองกับวิธีฟอลวก (Folaug) ซึ่งเป็นวิธีที่ได้มีการเสนอก่อนหน้านี้

พิษณุ ศรีวิไล และบรรพต (2557) ศึกษาและนำเสนอตัวแบบฮิดเดนมาร์คอฟในการตรวจหาเครื่องจักรที่มีหัวบรรจุหลายหัวในการผลิตเครื่องดื่มสำหรับควบคุมคุณภาพกระบวนการผลิตเพื่อลดสินค้าที่มีข้อบกพร่องหรือสินค้าเสียและเปรียบเทียบตัวแบบฮิดเดนมาร์คอฟ จากสถานะของเครื่องจักรที่ซ่อนอยู่มี 2 สถานะ คือ ใช้งานได้และใช้งานไม่ได้ ความน่าจะเป็นที่หัวบรรจุเครื่องดื่มจะอยู่ในสถานะใดในแต่ละเดือนอยู่ในรูปตัวแปรสุ่ม ผลการวิจัยพบว่าการแก้ไขข้อบกพร่องของหัวบรรจุเครื่องดื่มมีโอกาสที่หัวบรรจุเครื่องดื่มนั้นจะยังคงอยู่ในสถานะใช้งานไม่ได้ มากกว่าการที่ไม่ได้แก้ไขข้อบกพร่องของหัวบรรจุนั้นอยู่ 1.455 เท่า การเพิ่มขึ้นของเวลาที่ใช้ในการแก้ไขข้อบกพร่องของหัวบรรจุเครื่องดื่ม 1 หน่วยเวลา มีโอกาสที่หัวบรรจุเครื่องดื่มนั้นจะยังคงอยู่ในสถานะใช้งานไม่ได้มากกว่าการที่ใช้เวลาแก้ไขข้อบกพร่องเดิมอยู่ 1.455 เท่า และถ้าเดือนที่ผ่านมาหัวบรรจุเครื่องดื่มอยู่ในสถานะใช้งานไม่ได้ เดือนปัจจุบันมีโอกาสที่หัวบรรจุเครื่องดื่มจะอยู่ในสถานะใช้งานไม่ได้มากกว่าการที่จะเปลี่ยนมาอยู่ในสถานะใช้งานได้อยู่ 1.455 เท่า

จารุวรรณ พีรยุทธ์ และเสรี (2556) ศึกษาและนำเสนอการพัฒนาวิธีในการแก้ปัญหากำหนดการเศษส่วนเชิงเส้นแบบหนึ่งเงื่อนไขสองขั้นตอน โดยอาศัยทฤษฎีพื้นฐานของกำหนดการเชิงเส้น กำหนดการเศษส่วนเชิงเส้น และกำหนดการเชิงเส้นสองขั้นตอน ในการพัฒนาวิธีการแก้ปัญหาให้เกิดประสิทธิภาพและประสิทธิผล ซึ่งจะทำการเปรียบเทียบผลลัพธ์ที่ได้จากวิธีเฉพาะที่พัฒนาขึ้น กับผลลัพธ์ที่ได้จากวิธีการใช้คำสั่งลินพร็อก (Linprog) ในกล่องเครื่องมือของซอฟต์แวร์แมทแลป (Matlab) โดยใช้เวลาในการคำนวณของแต่ละวิธีเป็นตัวเปรียบเทียบ เมื่อค่าของตัวแปรและเงื่อนไขของปัญหา (m, n) มีการเปลี่ยนแปลงไป

ในงานวิจัยนี้จะทำการศึกษาจะแบ่งเป็น เป็นระยะแรกการจัดตารางการผลิตภายใต้เงื่อนไขต่าง ๆ ดังนี้ (1) เป็นเครื่องจักรขนานแบบ Sequence Dependent Setup (2) มีข้อจำกัดการทำงานของเครื่องจักร และ (3) การส่งมอบงานแบ่งเป็นแต่ละคาบเวลาของการผลิต วิธีที่ใช้ในการจัดการผลิตคือ แบบจำลองทางคณิตศาสตร์และวิธีปัญญาเชิงคำนวณ

ระยะที่สองเป็นการจัดตารางงานแบบสองวัตถุประสงค์ โดยเพิ่มวัตถุประสงค์เข้าไปอีกหนึ่งข้อ คือ ต้องการให้เครื่องจักรทำงานเต็มวัตถุประสงค์ ระยะนี้เพื่อหาแนวทางการจัดตารางงานแบบหลายวัตถุประสงค์ ทั้งนี้ยังคงใช้แบบจำลองทางคณิตศาสตร์เพื่ออ้างอิงการทำงานของวิธีปัญญาเชิงคำนวณ

2.3 ทฤษฎีที่เกี่ยวข้อง

2.3.1 การจัดการตารางการผลิต

การจัดตาราง (Scheduling) หมายถึง การจัดการทรัพยากรที่มีอยู่อย่างจำกัดให้กับภารกิจภายใต้ระยะเวลาที่กำหนด เพื่อให้องค์กรบรรลุเป้าหมาย หรือวัตถุประสงค์สูงสุดที่องค์กรกำหนดไว้ คำว่าทรัพยากร หมายถึงคนหรือเครื่องจักรที่มีจำนวนจำกัด (ปารเมศ ชุติมา, 2546: 1)

2.3.1.1 คำศัพท์และความหมาย

ระบบการผลิตประกอบด้วยเครื่องจักรและงาน โดยกำหนดให้มีจำนวนงาน n งาน และมีจำนวนเครื่องจักร m เครื่อง สำหรับ i หมายถึงงาน i และ j หมายถึงเครื่องจักร j

1. เวลางานเข้าสู่ระบบ (Release Time) หมายถึง เวลาที่งานเข้าสู่ระบบและพร้อมที่จะทำงาน หรือเวลาที่เร็วที่สุดที่งานสามารถเริ่มทำได้ เวลาที่งาน i เข้าสู่ระบบเขียนแทนด้วย R_i

2. เวลาเริ่มงาน (Staring Time) หมายถึง เวลาที่งานเริ่มทำบนเครื่องจักร เวลาเริ่มงานของงาน i เขียนแทนด้วย Si

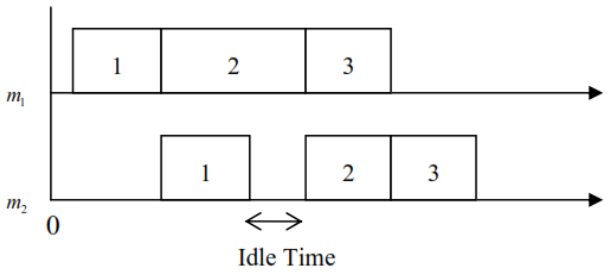
3. เวลารอคอย (Waiting Time) หมายถึง เวลาที่งานรอคอยเพื่อ ดำเนินการ อาจเรื่องมาจากงานที่อยู่ในลำดับก่อนหน้ายังดำเนินการไม่เสร็จ หรือเครื่องจักยังไม่พร้อมสำหรับการทำงานใด ๆ เวลารอคอยของงาน i เขียนแทนด้วย Wi

4. เวลาปรับตั้งเครื่องจักร (Setup Time) หมายถึง เวลาที่ใช้ปรับตั้งเครื่องจักร เพื่อเตรียมความพร้อมสำหรับการทำงาน ถ้าระบบมีการแบ่งงานออกเป็นกลุ่ม การทำงานภายในกลุ่มเดียวกันอย่างต่อเนื่องจะไม่มีการปรับตั้งเครื่องจักรส่วนการทำงานต่างกลุ่มกันอาจจะต้องมีการปรับตั้งเครื่องจักรเนื่องจากความแตกต่างในลักษณะของงานจึงต้องมีการเตรียมความพร้อมสำหรับการทำงานใหม่ ในขณะเดียวกัน จะมีค่าใช้จ่ายจากการปรับตั้งเครื่องจักร (Setup Cost) เกิดขึ้นด้วย เวลาปรับตั้งเครื่องจักรของงาน j เมื่อทำงาน i เสร็จแล้ว เขียนแทนด้วย S_{tij} เวลาปรับตั้งเครื่องจักรสามารถแบ่งได้เป็น เวลาปรับตั้งเครื่องที่ไม่ขึ้นกับลำดับงาน (Sequence independent setup time) และเวลาปรับตั้งเครื่องจักรที่ขึ้นกับลำดับงาน (Dependent Setup Time)

5. เวลาดำเนินการ (Processing Time) หมายถึง เวลาที่งานใช้ในการดำเนินการหรือทำบนเครื่องจักรใด ๆ เวลาดำเนินการของงาน i บนเครื่องจักร m เขียนแทนด้วย Pim

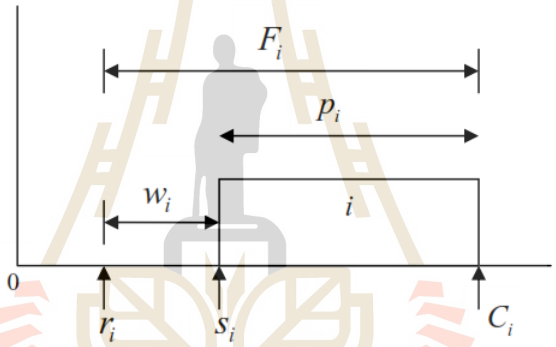
6. เวลาเสร็จงาน (Completion Time) หมายถึง เวลาที่งานเสร็จ ถ้าระบบมีเพียงเครื่องจักรเดียว เวลาเสร็จงานจะหมายถึง เวลาที่งานออกจากระบบ เขียนแทนด้วย Ci

7. เวลาว่างงาน (idle Time) หมายถึง เวลาที่เครื่องจักรหยุดทำงาน อาจเนื่องมาจากงานที่ต้องการทำยังไม่เสร็จจากขั้นตอนก่อนหน้า หรือยังไม่ถึงกำหนดเวลาเริ่มงาน



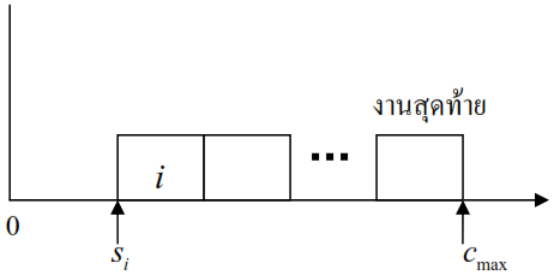
รูปที่ 2.1 เวลาว่างงานจองเครื่องจักร

8. เวลาไหลของงาน (Flow Time) หมายถึง เวลาทั้งหมดที่งานอยู่ในระบบ เวลาไหลของงาน จากรูป 2.2 เวลาไหลของงาน เขียนแทนด้วย Fi โดย Fi = Ci – Ri



รูปที่ 2.2 เวลาเริ่มงาน เวลาดำเนิน เวลาเสร็จงานและเวลาไหลของงาน

9. เวลาปิดงานของระบบ (Makespan) หมายถึง เวลาที่ระบบทำงานชิ้นสุดท้ายเสร็จ เขียนแทนด้วย Cmax



รูปที่ 2.3 เวลาปิดงานของระบบ

10. น้ำหนัก (Weigth) หมายถึง ปัจจัยที่แสดงถึงระดับสำคัญของงานเมื่อเปรียบเทียบงานอื่นในระบบ หรือจะมองเป็นสัดส่วนค่าใช้จ่ายที่เกิดขึ้นในระบบตามระดับความสำคัญของงาน

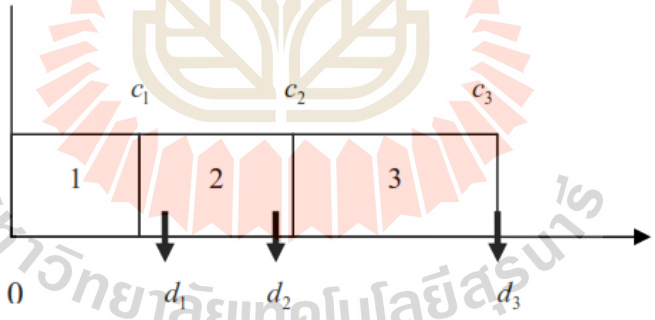
11. กำหนดส่งมอบงาน (Due Data) หมายถึง เวลาส่งมอบงานที่ตกลงไว้ว่าจะส่งมอบงานให้กับลูกค้า ซึ่งงานอาจทำเสร็จช้ากว่ากำหนดส่งมอบงานก็ได้ กำหนดส่งมอบงานของงาน i เขียนแทนด้วย di

12. เวลาสายของงาน (Lateness) เป็นเวลาที่แสดงให้เห็นว่างานทำเสร็จก่อนหรือทำเสร็จช้ากว่ากำหนดส่งมอบงานเป็นระยะเวลาเท่าไร เวลาสายของงาน i เขียนแทนด้วย Li โดยที่ $L_i = |c_i - d_i|$

13. งานเสร็จก่อนกำหนดและงานเสร็จช้ากว่ากำหนด (Earliness and Tardiness) เป็นอีกรูปแบบหนึ่งของเวลาสายของงาน เวลาที่เสร็จก่อนกำหนดและเวลาที่งานเสร็จช้ากำหนดของงาน i เขียนแทนด้วย Ei และ Ti ตามลำดับ โดยที่

Ei = max {0, di - ci}

Ti = max {0, ci - di}



รูปที่ 2.4 งาน 1 2 และ 3 เสร็จก่อนกำหนด เสร็จช้ากว่ากำหนด และเสร็จตามกำหนดตามลำดับ

จะเห็นว่าสำหรับ Li สามารถเป็นได้ทั้งค่าบวกและค่าลบ ซึ่งแตกต่างจาก Ei และ Ti ที่มีค่าเป็นศูนย์หรือเป็นบวกเท่านั้น ถ้างาน i มีกำหนดส่งมอบเป็น di การที่งานเสร็จก่อนกำหนด (Ci < di) จะทำให้เกิดค่าใช้จ่ายจากการเก็บงานนั้นเท่ากับ $\alpha_i E_i$ และงานทำเสร็จช้ากว่ากำหนด (Ci > di) จะเกิดค่าใช้จ่ายจากการเสียค่าปรับเป็น $\beta_i E_i$ เมื่อ α_i และ β_i เป็นอัตราค่าใช้จ่ายเมื่องาน i เสร็จก่อนและเสร็จช้ากว่ากำหนดตามลำดับ

2.3.1.2 การจำแนกการจัดตารางการผลิตตามลักษณะของการผลิต

1. การจัดตารางการผลิตแบบการไหลของสายงาน (Flow Shop Scheduling) ลักษณะการผลิตแบบไหลของสายงาน ประกอบด้วยเครื่องจักรหรือสถานีงานหลายสถานีงานที่ทำต่อเนื่องกันโดยลำดับขั้นตอนการทำงานของทุกงานเหมือนกัน ซึ่งหมายความว่างานเหล่านี้มีเส้นทางไหลเหมือนกัน ปัญหาการจัดตารางการผลิตแบบการไหลของสายงานประกอบด้วยเครื่องจักรที่ต่างกัน m เครื่อง และงานแต่ละงานประกอบด้วยจำนวนขั้นตอนการทำงาน n ขั้นตอน (Operation) โดยในแต่ละขั้นตอนการทำงานใช้เครื่องจักรที่แตกต่างกัน (Baker, 1974: 136)

2. การจัดตารางการผลิตแบบสั่งผลิตเป็นงาน ๆ (Job Shop Scheduling) การจัดตารางแบบสั่งผลิตเป็นงาน ๆ มีลักษณะต่างจากการจัดตารางการผลิตแบบการไหลสายงาน คือ เส้นทางการไหลของงานมีความแตกต่างกันไป การจัดตารางการผลิตแบบสั่งผลิตเป็นงาน ๆ ประกอบไปด้วยเครื่องจักรจำนวนหนึ่งและงานหลาย ๆ ประเภท โดยงานแต่ละงานประกอบด้วยขั้นตอนการทำงานหลาย ๆ ขั้นตอนซึ่งมีลำดับก่อน-หลังในการผลิตที่แน่นอน (Baker, 1974: 178)

2.3.1.3 วิธีการแก้ปัญหาการจัดตารางการผลิต (Production Scheduling Approach)

วิธีการจัดตารางการผลิตนั้นหลากหลายวิธี ตามความเหมาะสม ลักษณะของปัญหาที่ต้องการจัดการผลิต ซึ่งสามารถแบ่งแยกได้ 3 วิธีหลัก ๆ คือ

1. วิธีเชิงคณิตศาสตร์ (Mathematical) หรือวิธีการหาคำตอบที่ดีที่สุดของปัญหา (Optimal Solution) เป็นวิธีการที่หาตารางการผลิตที่ดีที่สุดได้โดยการสร้างแบบจำลองทางคณิตศาสตร์ ตัวอย่างของวิธีการแก้ปัญหาได้แก่ Integer-programming Mixed-integer programming และ Dynamic Programming เป็นต้น แต่ถ้าปัญหาที่มีความซับซ้อนสูงและปัญหาที่มีขนาดใหญ่ จะมีความยากมากหากใช้วิธีการแก้ปัญหาทางคณิตศาสตร์ ตัววัดประสิทธิภาพของการจัดตารางการผลิตด้วยวิธีเชิงคณิตศาสตร์มีดังต่อไปนี้

- เวลาไหลของงาน (Flow Time, Fj) จะหมายถึง ระยะเวลาทั้งหมดที่งานใช้เวลาอยู่ในระบบ แทนด้วย Fj = Cj - rj ซึ่งเวลาไหลของงานนี้จะเป็นตัววัดความสามารถในการตอบสนองต่องความต้องการของระบบ นอกจากนี้ยังแสดงให้เห็นถึงเวลาที่แต่ละงานต้องคอยในระบบตั้งแต่งานเข้ามาสู่ระบบจนกระทั้งงานออกจากระบบ การทำให้ค่าเวลาไหลเฉลี่ยของงาน (Mean Flow Time) มีค่าน้อยที่สุด และจะเกี่ยวข้องกับค่าผลรวมของเวลาเสร็จ (Sum of Completion Time) มีค่าน้อยที่สุด

- ปิดงานระบบ (Makespan, Cmax) เวลาปิดงานจะมีความสำคัญเมื่องานที่นำมาจัดตารางมีจำนวนจำกัด แทนด้วย Cmax = max(C_1 , C_2, ... , C_n) ซึ่งหมายถึงเวลาที่ระบบทำงานชิ้นสุดท้ายเสร็จสิ้น เวลาปิดงานมีความสัมพันธ์อย่างใกล้ชิดกับวัตถุประสงค์ด้านการทำงาน

ที่ก่อให้เกิดปริมาณผลผลิตมากที่สุดด้วย นอกจากนั้นแล้วยังทำให้เกิดการใช้งานเครื่องจักรอย่างสูง โดยเฉพาะอย่างยิ่งเครื่องจักรที่เป็นคอขวดของระบบ

- เวลาสาย (Lateness) แทนด้วย Lj = Cj – dj ถ้างานใดมีค่า Lj บวก แสดงว่างานนั้นสาย แต่ถ้างานใดมีค่า Lj ลบ แสดงว่างานนั้นทำเสร็จก่อนกำหนด และถ้างานใดมีค่า Lj เท่ากับ 0 แสดงว่างานนั้นเสร็จทันเวลาพอดี ซึ่งเวลาสายสูงสุด (Maximum Lateness) แทนด้วย Lmax = max(L1 , L2 , ... , Ln) การทำให้เวลาสายสูงสุดมีค่าน้อยที่สุด (min Lmax) คือการทำให้เวลาสายของงานที่สายมากที่สุด ในระบบมีค่าน้องที่สุดนั่นเอง
- จำนวนของงานที่สาย (Number of Tardy jobs) จะหาค่าได้จากจำนวนงานที่แสดงค่า Lj เป็นบวก
- เวลาล่าช้า (Tardiness) แทนด้วย Tj = max (Cj – dj,0) = max(Lj,0) ข้อแตกต่างระหว่างเวลาล่าช้ากับเวลาสาย คือ เวลาล่าช้าจะไม่มีทางที่จะมีค่าเป็นลบได้ น้อยที่สุดเท่ากับ 0
- เวลาล่าช้าทั้งหมดที่ถูกถ่วงน้ำหนัก (Total Weighted Tardiness) ต้องสมมุติฐานว่าแต่ละงานมีความสำคัญต่างกัน จะทำให้เราต้องให้ค่าน้ำหนักมากกับงานที่มีความสำคัญมากซึ่งเวลาล่าช้าทั้งหมดที่ถูกถ่วงน้ำหนัก
- จำนวนงานล่าช้าที่ถูกถ่วงน้ำหนัก (Weighted Number of Tardy Jobs) ในกรณีที่การล่าช้าของงานแต่ละงานมีความสำคัญไม่เท่ากัน ซึ่งจำนวนงานที่ล่าช้าทั้งหมดถูกถ่วงน้ำหนัก

2. วิธีกฎเกณฑ์การจ่ายงาน (Dispatching Rules) บางครั้งอาจเรียกว่า วิธีฮิวริสติกส์ (Heuristics) วิธีนี้ออกแบบมาเพื่อหาวิธีการที่เหมาะสมและยังสามารถแก้ปัญหาที่มีความซับซ้อนได้ วิธีกฎเกณฑ์การจ่ายงานมีกฎเกณฑ์การจ่ายงานให้เลือกใช้เป็นจำนวนมาก ผลลัพธ์ที่ดีด้วยวิธีการแก้ปัญหาที่สมเหตุสมผล และใช้เวลาไม่มากนัก วิธีพื้นฐานที่นิยมใช้กันในปัจจุบัน คือ

- รับก่อนทำก่อน (First Come First Served: FCFS)
- ทำงานที่ใช้เวลาน้อยที่สุดก่อน (Shortest Processing Time: SPT)
- ทำงานที่ใช้เวลามากที่สุดก่อน (Longest Processing Time: LPT)
- ทำงานที่มีกำหนดส่งมอบเร็วที่สุดก่อน (Earliest Due Date: EDD)

ปัจจุบันมีงานวิจัยที่คิดค้นวิธีการหาการจัดตารางที่ดีที่สุดเป็นจำนวนมาก และพัฒนาขั้นตอนการหาคำตอบด้วยการสร้างปัญหา และกำหนดตัววัดประสิทธิภาพหลากหลายรูปแบบ และทำการเปรียบเทียบผลที่ได้กับวิธีการจัดตารางที่มีอยู่เดิม ซึ่งแต่ละวิธีนำไปสู่การได้ตารางที่ดีที่สุดที่สอดคล้องกับวัตถุประสงค์ที่กำหนด

3. วิธีการค้นหาใกล้เคียง (Neighborhood Search) เป็นวิธีการค้นหาคำตอบเพื่อให้ได้คำตอบที่ดีที่สุด (Optimal Solution) ซึ่งจะเป็นวิธีที่มีประสิทธิภาพและรวดเร็ว

สามารถแก้ปัญหาที่มีความซับซ้อนสูงและปัญหาที่มีขนาดใหญ่ เช่น วิธี Tabu Search, Simulated annealing, Genetics Algorithms (ยงยุทธ, 2546: 12)

2.3.1.4 รูปแบบของการจัดเรียงเครื่องจักรในระบบการผลิต

รูปแบบที่สำคัญของการจัดเรียงเครื่องจักรมีอยู่หลายลักษณะ ขึ้นอยู่กับระบบการทำงานและปัจจัยแวดล้อมต่าง ๆ ขององค์กร ซึ่งรูปแบบของการจัดเรียงเครื่องจักรที่มีความเกี่ยวข้องกับโครงการวิศวกรรมที่ได้ทำการศึกษานั้น สามารถแบ่งออกได้ดังนี้

1. การจัดตารางการผลิตเครื่องจักรเดี่ยว (Single Machine) การจัดตารางการผลิตเครื่องจักรเดี่ยว หมายถึง ระบบการผลิตที่ประกอบด้วยเครื่องจักรเพียงเครื่องเดียว ซึ่งเป็นรูปแบบที่ง่ายที่สุดในรูปแบบของการจัดเรียงเครื่องจักร นอกจากนั้นแล้วระบบนี้ยังอาจจะเป็นรูปแบบการจัดเรียงเครื่องจักรแบบที่ซับซ้อนก็ได้ เช่น ในระบบการผลิตมีหลายเครื่องจักร มีเครื่องจักรอยู่เครื่องหนึ่งที่ใช้เวลาในการผลิตมากกว่าเครื่องอื่น และมีการจัดลำดับงานที่ไม่เหมาะสมทำให้เกิดคอขวด จึงมีผลให้เลื่อนในการผลิตโดยรวมมากขึ้น ดังนั้นการจัดลำดับงานที่เหมาะสมให้กับเครื่องจักรนี้จะเป็นตัวกำหนดสมรรถนะของระบบ แนวทางนี้เป็นการลดรูปของปัญหาเริ่มต้นเดิมที่ซับซ้อนของปัจจัยการจัดตารางเครื่องจักรเดี่ยวที่ง่ายขึ้น นอกจากนี้แบบจำลองของเครื่องจักรเดี่ยวยังสามารถนำไปใช้แก้ปัญหาแบบแยกส่วน (Decompose) ได้ซึ่งในกรณีเช่นนี้ปัญหาการจัดตารางของระบบผลิตที่ซับซ้อนจะถูกแยกออกเป็นปัญหาการจัดตารางเครื่องจักรเดี่ยวย่อย ๆ จำนวนหนึ่ง

2. การจัดตารางการผลิตเครื่องจักรขนาน (Parallel Machine) การจัดตารางการผลิตเครื่องจักรขนาน หมายถึง การจัดตารางการผลิตของงานจำนวนหนึ่งที่ต้องทำบนเครื่องจักรแบบขนาน ซึ่งแต่ละงานมีการดำเนินงานเพียงขั้นตอนเดียวเท่านั้น การจัดตารางการผลิตประกอบไปด้วยการตัดสินใจหลัก ๆ อยู่ 2 ประการด้วยกัน คือ การจัดสรรงาน (Allocation) หมายถึง การจัดสรรงานจากกลุ่มงานทั้งหมดเพื่อนำไปผลิตบนเครื่องจักรใดเครื่องจักรหนึ่งจากที่มีอยู่ทั้งหมด และการจัดลำดับงาน (Sequence) หมายถึงการพิจารณาถึงลำดับก่อนก่อนหลังของงานในแต่ละเครื่องจักร (ปารเมศ ชุตมา, 2546)

- เครื่องจักรขนานที่เหมือนกัน (Identical Parallel Machines) ระบบประกอบด้วยเครื่องจักรขนานจำนวนหนึ่ง เครื่องจักรแต่ละเครื่องมีความเร็วในการทำงานเท่ากัน ถ้าเครื่องจักรทำงานอย่างเดียวกันเมื่อทำงานเข้าสู่ระบบสามารถจัดให้งานทำบนเครื่องจักรใดก็ได้

- เครื่องจักรขนานที่ไม่เหมือนกัน(Non-Identical Parallel Machines) ระบบประกอบด้วยเครื่องจักรขนานจำนวนหนึ่ง เครื่องจักรแต่ละเครื่องมีความเร็วในการทำงานไม่เท่ากัน ถ้าเครื่องจักร m มีความเร็วในการทำงานเท่ากับ vm เวลาดำเนินการของงาน i บนเครื่องจักร m เท่ากับ pi/vm

2.3.1.5 เวลาปรับตั้งเครื่องจักรที่ขึ้นกับลำดับงาน

การทำงานของเครื่องจักรจะมีเวลาปรับตั้งเครื่องจักรเพื่อเตรียมความพร้อมหรือเมื่อมีการเปลี่ยนอุปกรณ์บางอย่างสำหรับการทำงานใหม่ เช่น การล้างเครื่องผสมสี การล้างสารเคมี การใส่แม่พิมพ์ ในอุตสาหกรรมการผลิตบางประเภทไม่สามารถกำหนดให้เครื่องจักรมีเวลาปรับตั้งเครื่องเป็นอิสระจากงานที่ทำได้ ตัวอย่างที่เห็นได้ชัดคือ อุตสาหกรรมการผลิตสี ในการผสมสีแต่ละครั้งได้มีการจัดลำดับการผสมสีไว้แล้วในเครื่องผสมสีเครื่องเดียวกัน ซึ่งการเปลี่ยนจากการผสมสีหนึ่งไปผสมอีกสีหนึ่งจะต้องมีการทำความสะอาดเครื่อง ซึ่งเวลาที่ใช้ในการทำความสะอาดจะมีค่าขึ้นอยู่กับสีที่ผสมเสร็จแล้วกับสีที่ต้องการผสม (Conway et al., 1967)

- เครื่องจักรขนานไม่สัมพันธ์กัน(Unrelated Parallel Machines) ระบบประกอบด้วยเครื่องจักรขนานจำนวนหนึ่ง เวลาดำเนินของงานจะขึ้นอยู่กับความเร็วของเครื่องจักร ถ้าเครื่องจักร m ทำงาน i ด้วยความเร็วเท่ากับ vm เวลาดำเนินการของงาน i บนเครื่องจักร m เท่ากับ Pi/Vim

ตารางที่ 1 ตัวอย่างข้อมูลของงานที่ต้องการเวลาปรับตั้งเครื่องจักร (หน่วย : ชั่วโมง)

X6	**สีที่ต้องการผสมใหม่**			
	สีขาว	**สีเหลือง**	**สีแดง**	**สีน้ำเงิน**
สีขาว	0	1	2	3
สีเหลือง	6	0	1	2
สีแดง	8	6	0	1
สีน้ำเงิน	10	8	6	0

ที่มา: Conway et al.(1967)

มหาวิทยาลัยเทคโนโลยีสุรนารี

จากตารางที่ 1 ถ้าให้เริ่มต้นผสมสีขาวก่อน จะได้ว่าสามารถจัดลำดับที่เป็นไปได้ในการผสมสีได้ 6 แบบ และผลรวมของเวลาจัดตั้งเครื่องจักรเป็นดังตารางที่ 2

ถ้าเวลาปรับตั้งเครื่องจักรไม่ขึ้นกับลำดับงานจะรวมเวลาปรับตั้งเครื่องจักรให้เป็นส่วนหนึ่งของเวลาดำเนินการ แต่ถ้าเวลาปรับตั้งเครื่องจักรขึ้นกับลำดับงานจะมีผลให้เวลาดำเนินการเปลี่ยนไปทำให้ปัญหาการจัดตารางมีความซับซ้อนขึ้น เวลาที่ใช้ในการเปลี่ยนจากการทำงาน i ไปทำงาน j หรือเปลี่ยนจากการทำงาน j ไปทำงาน i อาจจะเท่ากันหรือไม่เท่ากันก็ได้

ตารางที่ 2 ตัวอย่างผลรวมของเวลาปรับตั้งเครื่องจักรเมื่อมีลำดับการทำงานต่างกัน (หน่วย: ชั่วโมง)

ลำดับการทำงาน	ผลรวมเวลาปรับตั้งเครื่อง
ขาว-เหลือง-แดง-น้ำเงิน-ขาว	13
ขาว-แดง-น้ำเงิน-เหลือง-ขาว	17
ขาว-น้ำเงิน-แดง-เหลือง-ขาว	21
ขาว-เหลือง-น้ำเงิน-แดง-ขาว	17
ขาว-แดง-เหลือง-น้ำเงิน-ขาว	20
ขาว-น้ำเงิน-เหลือง-แดง-ขาว	20

ที่มา: Conway et al. (1967)

2.3.2 แบบจำลองทางคณิตศาสตร์การจัดตารางการผลิตด้วยเครื่องจักรขนาน

ระยะแรกของงานวิจัยนี้เป็นการศึกษารูปแบบการจัดลำดับการผลิตสำหรับ งาน n งาน (0, 1, 2, 3,..., n) ซึ่งงาน 0 เป็นงานสมมุติ (Dummy Job) กำหนดขึ้นเพื่อให้สมการคณิตศาสตร์มีความสมบูรณ์เท่านั้น งานทั้ง n งาน จะถูกจัดตารางบนเครื่องจักร M(s) เครื่อง (1, 2, 3,..., M(s)) มีการตัดสินใจเลือกงานที่จะดำเนินการ และเลือกเครื่องจักรเพื่อผลิตงานนั้น ๆ การจัดเรียงเครื่องจักรเป็นแบบขนาน และมีเงื่อนไขของเครื่องจักร คือ งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ และงานทุกงานจะต้องมีเครื่องจักรอย่างน้อยหนึ่งเครื่องที่สามารถผลิตงานนั้นได้ การจัดตารางการผลิตพิจารณาเมื่อเครื่องจักรมีการเปลี่ยนการผลิตจากงาน i (i = 0, 1, 2, 3,..., n) ไปเป็นงาน j (j = 0, 1, 2, 3,..., n) จะมีเวลาติดตั้งเครื่องจักร $T(i, j) \geq 0$ และมีสมมติฐานว่างานที่ดำเนินงานเริ่มต้นบนแต่ละเครื่องจักรไม่มีเวลาติดตั้งเครื่องจักร คือ $T(0, i) = 0$ การจัดตารางการผลิตนี้เพื่อที่จะทำให้เวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด

2.3.2.1 แบบปัญหาทางคณิตศาสตร์ (Mathematical Model)

ในการสร้างรูปแบบทางคณิตศาสตร์เพื่อหาลำดับการผลิตที่เหมาะสมที่สุดในสายการผลิตที่เครื่องจักรจัดเรียงแบบขนาน กรณีลำดับการผลิตมีผลต้อเวลาติดตั้งเครื่องจักร และมีข้อจำกัดของเครื่องจักร คือ งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ เมื่อพิจารณาการจัดตารางการผลิตของงานโดยมีวัตถุประสงค์เพื่อทำให้เวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด ได้มีการประยุกต์แบบจำลองทางคณิตศาสตร์ของรูปแบบปัญหาการจัดเส้นทาง การขนส่งของ Zou Tong, Li Ning and Sun Debao (2004) โดยใช้ integer programming

2.3.2.2 สมการเป้าหมาย (Objective function)

เพื่อให้เวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด ค่าใช้จ่ายในการผลิตรวมประกอบด้วย ค่าใช้จ่ายการผลิตทุกงานทุกคาบเวลา ค่าใช้จ่ายในการเก็บรักษาทุกงานทุก

คาบเวลา และค่าใช้จ่ายในการปรับตั้งเครื่องจักรเพื่อผลิต

$$\text{Minimize: } Z = \sum_{j=1}^{n}\sum_{t=1}^{T} g_{jt} \cdot C_{jt} + \sum_{j=1}^{n}\sum_{t=1}^{T} o_{jt} \cdot H_{jt} + \sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n}\sum_{m=1}^{M}\sum_{t=1}^{T} s_{ij} \cdot X_{ijmt} \qquad (2.1)$$

เมื่อ

g_{jt} = ค่าใช้จ่ายในการผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t

C_{jt} = จำนวนชิ้นงานการผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t

o_{jt} = ค่าใช้จ่ายในการเก็บรักษาของผลิตภัณฑ์ j ในรอบการทำงาน t

H_{jt} = จำนวนชิ้นงานที่เก็บรักษาของผลิตภัณฑ์ j ในรอบการทำงาน t

s_{ij} = ค่าใช้จ่ายในการปรับตั้งเครื่องจักรในการผลิตงาน j เมื่อทำต่อจากงาน i

X_{ijmt} = 1: เมื่องาน i ถูกผลิตก่อนงาน j ทันที บนเครื่องจักร m ในรอบการทำงาน t

= 0: เมื่อเป็นอย่างอื่น

2.3.2.3 ข้อจำกัดในการจัดตารางการผลิต (Constraints)

ข้อจำกัดในการจัดตารางการผลิต คือ เงื่อนไขที่ต้องพิจารณาในการจัดตารางการผลิตสำหรับปัญหานี้มีหลายอย่างด้วยกัน เช่น

1. จำนวนงานที่ผลิตต้องเพียงพอต่อความต้องการเพื่อส่งมอบในแต่ละรอบการทำงาน
2. เวลาในการผลิตงานรวมในแต่ละรอบการทำงานต้องน้อยกว่าหรือเท่ากับ Work Hour
3. เมื่อสั่งผลิตต้องผลิตอย่างน้อยตามจำนวนขั้นต่ำ
4. จำนวนงานที่ผลิตเป็นจำนวนเต็มบวก
5. งานที่ถูกมอบหมายให้เครื่องจักรต้องสอดคล้องกับข้อจำกัดในการผลิตของเครื่องจักร

2.3.2.4 ข้อมูลที่ป้อนให้กับแบบปัญหาทางคณิตศาสตร์

ค่าของตัวแปรต่าง ๆ ที่ต้องป้อนให้กับแบบทางคณิตศาสตร์เพื่อคำนวณ

<u>Input – Parameter</u>

1. จำนวนผลิตภัณฑ์ nJobs
2. จำนวนรอบการทำงาน nPeriod
3. จำนวนเครื่องจักร nMachine

4. จำนวนชั่วโมงทำงานทั้งหมด, WorkHour

Input – Data

1. เวลาที่ใช้ในการผลิตแต่ละชิ้น Production Time P_j
2. เวลาที่ใช้ในการรอเพื่อปล่อยงานเข้าเครื่องจักร Release Time R_j
3. ความต้องการผลการผลิต ในแต่ละคาบเวลาการผลิต Demand D_{jt}
4. ข้อจำกัดการผลิตของเครื่องจักร Machine Eligibility E_{jm}
5. ค่าใช้จ่ายในการผลิตต่อชิ้น ในแต่ละคาบเวลาการผลิต g_{jt}
6. ค่าใช้จ่ายในการเก็บรักษาต่อชิ้น ในแต่ละคาบเวลาการผลิต, o_{jt}
7. ค่าใช้จ่ายในการปรับตั้งเครื่องจักร กรณีงาน I ต่อไปยังงาน J, s_{ij}

2.3.2.5 ข้อมูลที่ได้จากแบบปัหาทางคณิตศาสตร์

ค่าของตัวแปรต่าง ๆ ที่ได้จากแบบปัญหาทางคณิตศาสตร

Output – Data

1. เวลาที่ใช้ในการผลิตรวม, Complete Time C_{jt}
2. จำนวนงานที่ผลิตในแต่ละคาบเวลา N_{jmt}
3. ตารางการผลิต (Binary) X_{ijmt}

2.3.3 การหาคำตอบที่เหมาะสม

เทคนิคการหาคำตอบที่เหมาะสมสามารถแบ่งเป็น 22 ประเภท คือ การหาคำตอบที่เหมาะสมด้วยวิธีดั้งเดิม (Conventional Optimization Algorithm) และวิธีการหาคำตอบแบบการประมาณ (Approximation Optimization Algorithm)

ก. แนวคิดวิธีการหาคำตอบที่ดีที่สุด โดยอาศัยหลักทางคณิตศาสตร์ (Mathematical)

วิธีการแก้ปัญหาโดยใช้หลักการทางคณิตศาสตร์ ถูกพัฒนาขึ้นในสมัยสงครามโลกครั้งที่ 2 โดยมีวัตถุประสงค์ในการแก้ปัญหาที่มีความซับซ้อนทางทหาร วิธีการแก้ปัญหาแบบนี้ได้ถูก นำไปใช้แก้ปัญหาในด้านอื่น ๆ มากมาย ไม่ว่าจะเป็นปัญหาในด้านการจัดตาราง (Scheduling problems) ปัญหาการมอบหมายงาน (Assignment problem) รวมไปถึงปัญหาการเดินทางของพนักงานขาย (Traveling salesman problem) ด้วย ซึ่งวิธีการที่ถูกนำมาใช้ในการแก้ปัญหาเพื่อหาคำตอบสำหรับปัญหาการจัดเซลล์การผลิตนั้น มีอยู่ 4 วิธี ได้แก่ Dynamic programming, linear programming, Goal programming และ Linear and quadratic integer programming อย่างไรก็ดี วิธีการหาค่าคำตอบที่ดีที่สุดโดยอาศัยหลักการทางคณิตศาสตร์นั้นจะเหมาะสมกับปัญหาขนาดเล็กเท่านั้น เนื่องมาจากข้อจำกัดและกฎเกณฑ์ในการหาคำตอบที่ตายตัวจนเกินไป (Enumerative search) ซึ่งเมื่อนำวิธีการเหล่านี้ไปแก้ไขปัญหาที่มีขนาดใหญ่และซับซ้อนมากขึ้น ต้องใช้ระยะเวลาในการแก้ปัญหาที่นานมากขึ้นเป็นทวีคูณ

ข. แนวคิดวิธีการหาคำตอบที่ดี โดยอาศัยหลักการประมาณ (Approximation optimization algorithm)

สำหรับปัญหาการหาค่าที่ดีที่สุดโดยใช้วิธีทางคณิตศาสตร์จะเหมาะสำหรับแก้ปัญหาที่มีขนาดเล็ก ซึ่งเป็นเรื่องที่ยากมากในการที่จะหาคำตอบที่มีขนาดใหญ่ โดยเฉพาะว่าเป็นปัญหาแบบ NP-Hard problems ขณะที่ขนาดของปัญหาเพิ่มขึ้นเพียงเล็กน้อยแต่ต้องใช้เวลาในการแก้ปัญหาที่นานมากขึ้นหลายเท่าตัว ดังนั้นจึงมีวิธีการหาค่าคำตอบที่ดีที่สุดโดยอาศัยหลักของการประมาณเกิดขึ้น เพื่อจัดการกับปัญหาลักษณะเช่นนี้โดยตรง ซึ่งวิธีการในกลุ่มนี้จะมีความสามารถในการแก้ปัญหาที่มีขนาดใหญ่และมีความซับซ้อนสูง ๆ ได้

เมตาฮิวริสติก (Metaheuristics) เป็นสาขาหนึ่งของการหาค่าคำตอบที่ดีที่สุดโดยอาศัยหลักของการประมาณและประสบความสำเร็จอย่างมาก มีความรวดเร็วในการประมวลผลในการแก้ปัญหาที่มีความซับซ้อนสูง ๆ อย่างเช่น ปัญหาการเดินทางของพนักงานขาย (Traveling salesman problem) วิธีการต่าง ๆ ในกลุ่มของเมตาฮิวริสติก ก็เช่น ซิมูเลทเทดแอนนีลลิง (Simulated annealing: SA), ทาบูเสิร์ช (Taboo search: TS), นิวรอลเน็ทเวิร์ค (Neural network: NN), เจเนติคอัลกอริทึม (Genetic algorithm: GA) รวมไปถึง แอนท์โคโลนีออพติไมเซชั่น (Ant colony optimization: ACO) โดยวิธีการเหล่านี้จะได้ผลเฉลยที่ไม่ใช่ผลเฉลยดีที่สุดแต่จะได้เป็นกลุ่มของคำตอบ กระบวนการทำงานจะวนซ้ำแล้วจะหยุดทำงานเมื่อถึงเงื่อนไขตามที่กำหนดไว้ ตัวอย่างขั้นตอนวิธีเหล่านี้ เช่น ACO TS SA GA และวิธีการแบบไฮบริด (Hybrid Approaches) ซึ่งแต่ละวิธีจะมีกลยุทธ์ในการค้นหาผลเฉลยที่แตกต่างกันออกไป

2.3.3.1 วิธีการหาคำตอบแบบฮิวริสติกและวิธีการเมตาฮิวริสติก (Heuristic Optimization and Meta-Heuristic Optimization)

วิธีฮิวริสติก หมายถึง วิธีการคิดค้นขึ้นมา เพื่อใช้ในการแก้ปัญหาใดปัญหาหนึ่งโดยเฉพาะ ซึ่งไม่มีแบบแผนที่แน่นอนตายตัว โดยการสร้างฮิวริสติกนั้นมักต้องอาศัยความเข้าใจและประสบการณ์ในการแก้ไขปัญหานั้น ๆ เป็นอย่างดี ดังนั้น วิธีฮิวริสติกที่ ใช้ในการแก้ปัญหาหนึ่งอาจไม่สามารถนำไปใช้แก้ไขปัญหาอีกปัญหาหนึ่งได้ และไม่สามารถรับประกันได้ว่าจะได้คำตอบที่ดีที่สุดหรือคำตอบที่เท่ากันทุกครั้ง แต่สามารถได้คำตอบในเวลาที่รวดเร็ว หรือสามารถแก้ปัญหาที่มีความซับซ้อนจนไม่สามารถเขียนเป็นตัวแบบทางคณิตศาสตร์ได้

วิธีฮิวริสติกแบ่งออกเป็น 2 ประเภท ตามลักษณะของวิธีการในการสร้างคำตอบให้แก่

1. วิธีฮิวริสติกแบบสร้างคำตอบ วิธีนี้จะเริ่มสร้างคำตอบโดยเริ่มจากการค่อย ๆ เพิ่มลูกค้าในเส้นทางทีละรายหรือเพิ่มโหนดทีละโหนด จนประกอบกันเป็นคำตอบที่สมบูรณ์ เช่น วิธี Saving, Matching Based, Nearest Insertion, Nearest Neighbor เป็นต้น

2. วิธีฮิวริสติกแบบค้นหาคำตอบใกล้เคียง (Neighborhood Search Heuristic) เป็นวิธีสร้างคำตอบขึ้นมาคำตอบหนึ่งที่ไม่ขัดแย้งกับเงื่อนไขแล้วนำคำตอบนั้นมาทำการสลับตำแหน่งไปเรื่อย ๆ เพื่อหาคำตอบที่ดีกว่าคำตอบเดิม ตามรอบที่กำหนดที่ได้ออกแบบไว้ เช่น วิธี Cluster First-Route Second, วิธี Route First- Cluster second, Sweep, วิธี Petal เป็นต้น

ตัวอย่างของวิธีฮิวริสติกที่นิยมใช้ในงานวิจัยโลจิสติกส์ เช่น วิธี Saving วิธี Matching Based วิธี Nearest Insertion วิธี Nearest Neighbor และวิธี Local Search เป็นต้น ดังนั้นวิธีฮิวริสติกจึงถูกพัฒนาให้มีความยืดหยุ่น รวดเร็วและมีประสิทธิภาพมากยิ่งขึ้น เรียกว่า เมต้าฮิวริสติก (Meta-Heuristic)

หลักการเบื้องต้นของเมต้าฮิวริสติก

- เมต้าฮิวริสติกมีระเบียบวิธีในการค้นหาคำตอบที่ดีภายในเซตของคำตอบที่เป็นไปได้
- เมต้าฮิวริสติกมีจุดประสงค์เพื่อหาคำตอบที่ดีที่สุดหรือคำตอบที่ใกล้เคียงที่ดีที่สุด ภายในระยะเวลาที่เหมาะสม
- วิธีเมต้าฮิวริสติกอาจมีทั้งแบบซับซ้อนและไม่ซับซ้อน เช่น วิธีโลคอลเซิร์ช (Local Search) วิธีระบบมด (Ant System) วิธีทางพันธุกรรม (Genetic Algorithm) วิธีการค้นหาต้องห้าม (Tabu Search) และวิธีเลียนแบบการอบอ่อน (Simulated Annealing) เป็นต้น
- เมต้าฮิวริสติกอาจเกิดจากการรวมหลากหลายวิธีเทคนิคเพื่อค้นหาคำตอบที่ดีที่สุดภายในพื้นที่คำตอบที่เป็นไปได้
- เมต้าฮิวริสติกมีระเบียบขั้นตอนที่แน่นอนแต่สามารถปรับเปลี่ยนในรายละเอียดเมื่อนำไปใช้แต่ละปัญหา
- เมต้าฮิวริสติกบางประเภทมีการใช้ความจำชั่วคราวมากขึ้น ในการจดจำคำตอบเดิมเพื่อให้การค้นหาคำตอบที่ไม่ซ้ำที่เดิมนอกจากนี้ ยังได้เสนอการแบ่งเมต้าฮิวริสติกไว้ 6 ประเภทดังนี้

1. เมต้าฮิวริสติกที่เกิดจากแรงบันดาลใจจากธรรมชาติ ได้แก่ Ant System Genetic Algorithm, Simulated Annealing) และวิธีการเกาะกลุ่มประชากรแบบ PSO (Particle Swarm Optimization) เป็นต้น

2. เมต้าฮิวริสติกที่ไม่ได้เกิดจากแรงบันดาลใจจากธรรมชาติ ได้แก่ Tabu Search เป็นต้น

3. เมต้าฮิวริสติกแบบใช้ประชากรคือในหนึ่งรอบของการค้นหาคำตอบจะได้คำตอบมากกว่าหนึ่งคำตอบให้เลือก เช่น Ant System, Genetic Algorithm) วิธีการลอกแบบ (Memetic Algorithm), PSO (Particle Swarm Optimization) เป็นต้น

4. เมต้าฮิวริสติกแบบไม่ใช้ประชากร คือในหนึ่งรอบของการค้นหาคำตอบจะได้คำตอบออกมาเพียงหนึ่งคำตอบเท่านั้น เช่น Simulated Annealing Tabu Search) และวิธีการค้นหาในพื้นที่คำตอบที่เป็นไปได้ในแบบวนซ้ำ (Iterate Local Search) เป็นต้น

5. เมต้าฮิวริสติก แบบสมการเป้าหมายคงที่ คือในหนึ่งรอบของการคำนวณอาจมีการเปลี่ยนแปลงสมการเป้าหมาย เพื่อให้ได้คำตอบใหม่ๆ เกิดขึ้น เช่น วิธี Guided Local Search เป็นต้น

6. เมต้าฮิวริสติกแบบไม่มีการเปลี่ยนแปลงสมการเป้าหมาย เช่น Ant System, Genetic Algorithm, Simulated Annealing, Memetic Algorithm เป็นต้น

2.3.3.2 การโปรแกรมแบบฮิวริสติก

การพิจารณาหาทางเลือกที่เหมาะสมให้กับปัญหาการตัดสินใจที่ซับซ้อนจะเกี่ยวข้องกับข้อจำกัดด้านเวลาและค่าใช้จ่าย หรือบางครั้งอาจเป็นไปได้ยากในการนำมาปฏิบัติจริง เนื่องจากการจำลองเหตุการณ์ (Simulation) อาจจะต้องใช้เวลา, ค่อนข้างซับซ้อนและไม่แม่นยำ บางครั้งการใช้ฮิวริสติกอาจทำให้ได้ผลลัพธ์ที่น่าพึงพอใจได้เร็วกว่า และเสียค่าใช้จ่ายน้อยกว่า เนื่องจากเป็นวิธีที่ทำการทดลองค้นคว้าหากฎด้วยตัวเอง (ใช้ดุจพินิจ, วิจารญาณ) ในการเลือกตัดสินใจโดยไม่มีทฤษฎีใด ๆ มาช่วย โดยกฎที่ได้นี้อาจได้จากการใช้ผู้เชี่ยวชาญ หรือได้จากการใช้วิธีลองผิดลองถูก

ฮิวริสติกมักจะถูกใช้ในการแก้ปัญหาที่มีโครงสร้างไม่ดีนัก และสามารถใช้ในการหาทางแก้ปัญหาที่น่าพึงพอใจสำหรับปัญหาที่ซับซ้อนหรือปัญหาที่มีโครงสร้าง ได้เร็วกว่าและถูกกว่าการใช้อัลกอริธึม ปัญหาที่เกิดขึ้นในการใช้ฮิวริสติกก็คือ เป็นวิธีการที่ไม่เป็นที่รู้จักอย่างแพร่หลายเท่ากับการใช้ฮัลกอริธึม ดังนั้นจึงสามารถนำไปใช้กับสถานการณ์เฉพาะบางอย่างที่มีการวางเป้าหมายเอาไว้ตั้งแต่ต้นเท่านั้น อีกปัญหาหนึ่งก็คือทางแก้ปัญหาที่ได้จากการใช้ฮิวริสติกอาจไม่ใช่ทางแก้ปัญหาที่ดีที่สุดเหมือนการใช้อัลกอริธึมอื่น แต่เป็นเพียงทางแก้ปัญหาที่เป็นไปได้และ "ดีเพียงพอ" โดย คำว่า "ดีเพียงพอ" มักจะอยู่ในช่วง 90-99.9% ของทางแก้ปัญหาที่เหมาะสมจริง ๆ

วิธีการของฮิวริสติก

การคิดแบบฮิวริสติกไม่จำเป็นต้องดำเนินไปตามแนวทางการจัดการที่วางไว้ มันจะเกี่ยวข้องกับการค้นหา การเรียนรู้ การประเมินค่า และการตัดสิน โดยขบวนการในการค้นหา การเรียนรู้ และการประเมินค่านี้จะเกิดขึ้นซ้ำแล้วซ้ำเล่า เหมือนกับการสำรวจ เพื่อนำไปสู่วิธีการอีกรูปแบบหนึ่ง ความรู้จะถูกได้รับจากความสำเร็จหรือความล้มเหลวที่บางจุด ที่มีผลสะท้อน (Feed Back) กลับมา และทำการแก้ไขขบวนการค้นหานั้น ๆ ให้ดีขึ้น

ต่อไปนี้เป็นสถานการณ์ซึ่งเหมาะสมในการนำฮิวริสติกมาใช้

1. ข้อมูลเข้าไม่แน่นอน หรือมีจำกัด

2. ระบบจริง ๆ มีความซับซ้อนมากจนกระทั่งไม่สามารถใช้วิธีการหาคำตอบที่เหมาะสมที่สุด (Optimization) ได้

3. ไม่มีวิธีการหรืออัลกอริธึม ที่น่าเชื่อถือที่สามารถใช้ได้อย่างแท้จริง

4. ใช้เวลาในการคำนวณหาทางแก้ปัญหาที่เหมาะสมที่สุดมากเกินไป

5. มีความเป็นไปได้ที่จะปรับปรุงประสิทธิภาพของขบวนการในการหาทางแก้ที่เหมาะสมที่สุด (เช่น โดยการหาจุดเริ่มต้นที่ดีในการแก้ปัญหาที่มีการใช้ ฮิวริสติก) แล้วรวมการใช้ฮิวริสติกนี้เข้ากับวิธีการหาคำตอบที่เหมาะสมที่สุด

6. เป็นปัญหาที่ซับซ้อนไม่คุ้มค่ากับการใช้วิธีหาคำตอบที่เหมาะสมที่สุด หรือเป็นปัญหาที่ต้องใช้เวลามาก

7. เมื่อมีการเกี่ยวข้องกับขบวนการทางสัญลักษณ์ (Symbolic) มากกว่าทางตัวเลข (Numerical)

8. เมื่อต้องการทำการตัดสินใจอย่างรวดเร็ว โดยไม่สามารถใช้คอมพิวเตอร์เข้ามาช่วยได้ (การใช้ฮิวริสติกบางครั้ง ไม่จำเป็นต้องใช้คอมพิวเตอร์เข้ามาช่วย)

ข้อดีของฮิวริสติก

1. ง่ายในการทำความเข้าใจ และง่ายในการนำไปใช้และการอธิบาย

2. ช่วยในการอบรมคนให้มีความคิดสร้างสรรและก่อให้เกิดการสร้างฮิวริสติกกับปัญหาอื่นๆ ได้ด้วย

3. ประหยัดเวลาในการสร้างตัวแบบ

4. ลดความต้องการในด้านการเขียนโปรแกรมและความต้องการด้านแหล่งเก็บข้อมูลคอมพิวเตอร์ได้

5. ลดเวลาในการทำงานของคอมพิวเตอร์ จึงทำการตัดสินใจได้อย่างรวดเร็ว

6. ทำให้เกิดทางแก้ปัญหาได้หลายทาง

7. สามารถประยุกต์ใช้ฮิวริสติกที่มีประสิทธิภาพ เข้ากับตัวแบบซึ่งสามารถแก้ปัญหาด้านการโปรแกรมเชิงคณิตศาสตร์ได้

ปัญหาในการใช้ฮิวริสติก

1. ไม่สามารถรับประกันได้ว่าทางแก้ปัญหาที่ได้จะเป็นทางแก้ปัญหาที่ดีที่สุดหรือเหมาะสมที่สุด

2. มีข้อยกเว้นมากเกินไป ในการสร้างเป็นกฎขึ้น

3. ผลลัพธ์ที่ได้จากทางเลือกในการตัดสินใจ อาจไม่เป็นไปตามที่คาดเดาไว้

4. การขึ้นระหว่างกันของส่วนใดส่วนหนึ่งของระบบ บางครั้งสามารถมีอิทธิพลอย่างมากกับระบบทั้งระบบได้

นักวิจัยบอกว่าการใช้วิธีทั่ว ๆ ไปและการใช้ฮิวริสติกสามารถจะเกิดการผิดพลาดได้เพราะวิธีเหล่านั้นไม่มีข้อกำหนดตายตัว คือไม่มีข้อจำกัดว่าจะต้องเริ่มต้นที่จุดไหน, ลำดับขั้นตอนการทำงานต้องเป็นอย่างไร หรือไม่จำกัดว่าจะต้องสร้างตัวเลือกในการตัดสินใจหรือไม่, ไม่เจาะจงด้านข้อจำกัดของการแก้ปัญหา, ทางเลือกของเกณฑ์ที่ใช้ในการระบุขบวนการทำงาน, ระดับของค่าใช้จ่ายที่ใช้ในการหาว่าผลลัพธ์สุดท้ายที่ได้เป็นผลลัพธ์ที่ดีที่สุดจริง ๆ ผลลัพธ์เป็นพฤติกรรมที่ไม่มีจุดมุ่งหมายแน่ชัดและไม่สามารถคาดเดาได้ ผลลัพธ์อาจดีในการนำไปใช้กับระบบงานหนึ่งแต่อาจไม่ดีในการนำไปใช้กับอีกระบบงานอื่นก็ได้

2.3.3.3 วิธีการค้นหาคำตอบใกล้เคียงแบบแปรผัน

วิธีการค้นหาคำตอบใกล้เคียงแบบแปรผัน (Variable Neighborhood Search: VNS) เป็นวิธีการรูปแบบหนึ่งในสาขาเมตาฮิวริสติก ถูกนำเสนอครั้งแรกในปี 1997 โดย Pierre Hansen และ Mladenovic Nenad เป็นวิธีการที่สามารถใช้ในการหาค่าที่เหมาะสมที่สุดได้สำหรับหลายปัญหา และการออกแบบรวมถึงการประยุกต์ใช้วิธีการทางฮิวริสติกส์ สำหรับปัญหาที่มีความหลากหลาย ทางด้านตัวแปร ข้อกำหนด ปัญหาต่าง ๆ ที่ซับซ้อน การค้นหาคำตอบใกล้เคียงแบบแปรผันถูกพัฒนา จากการปรับปรุงคำตอบเฉพาะที่ กลยุทธ์ที่สำคัญของวิธีนี้คือ ทำให้คำตอบเฉพาะที่หลีกเลี่ยงไม่ให้คำตอบไปติดที่ค่าคำตอบที่ดีที่สุดในย่านคำตอบใกล้เคียง โดย VNS จะอาศัยการเปลี่ยนแปลง โครงสร้างการปรับปรุงคำตอบในย่านใกล้เคียง (Neighborhood Structure: NS1) ถ้าคำตอบใหม่ที่ค้นพบมีคุณภาพแย่กว่าคำตอบเดิมหรือว่าค่าคำตอบไปติดอยู่ที่คำตอบที่ดีที่สุดเฉพาะที่ โดยการ ค้นหาคำตอบใกล้เคียงแบบแปรผันจะทำซ้ำจนกว่าจะถึงเงื่อนไขหยุดการค้นหาหรือเกณฑ์การหยุดค้นหาตามที่ผู้วิจัยได้ออกแบบไว้ จากลักษณะของพื้นที่ในการหาค้นหาคำตอบแสดงให้เห็นว่าการเปลี่ยนแปลงโครงสร้างการปรับปรุงคำตอบย่านคำตอบใกล้เคียง สามารถเพิ่มโอกาสในการพบค่าคำตอบที่ดีมากกว่าการทำการปรับปรุงคำตอบเฉพาะที่ โดยไม่มีการเปลี่ยนแปลงโครงสร้างการปรับปรุงคำตอบย่านคำตอบใกล้เคียง การค้นหาคำตอบใกล้เคียงแบบแปรผันเป็นวิธีที่เป็นเหตุเป็นผลจึงถูกนำมาใช้แก้ปัญหาเชิงการจัดอย่างแพร่หลาย เนื่องจากที่โครงสร้างการทำงานที่สามารถทำความเข้าใจง่าย และไม่ซับซ้อนจนเกินไปอีกทั้งค่าพารามิเตอร์ที่ต้องการมีจำนวนน้อย

Hansen and Mladenovic (2001) ได้ทำการสรุปทฤษฎีของวิธีสำรวจตัวแปรข้างเคียงและการประยุกต์วิธีสำรวจตัวแปรข้างเคียงในรูปแบบต่าง ๆ เพื่อแก้ปัญหาในรูปแบบที่แตกต่างกัน กล่าวว่า VSN เป็น Search Algorithm อีกกลุ่มหนึ่งที่ถูกพัฒนาขึ้น เหมาะสำหรับปัญหาที่มีขนาดใหญ่มีแนวคิดในการพัฒนาคำตอบให้ดีขึ้นโดยพยายามที่จะออกจากช่วงคำตอบที่มีอยู่ในปัจจุบันเพื่อหา คำตอบใหม่ในช่วงอื่น ซึ่งประกอบไปด้วยขั้นตอนในการพัฒนาคำตอบ 4 ขั้นตอน คือ

1) การกำหนดกรอบการค้นหา (Neighborhood Domain)

2) การสุ่มเพื่อเลือกคำตอบที่จะทำการพัฒนา (Shaking)

3) การหาคำตอบในช่วงใกล้เคียง (Local Search)

4) เปรียบเทียบผลที่ได้ ระหว่างคำตอบเดิมและคำตอบใหม่

- หากคำตอบที่ได้มีการพัฒนาให้เลือกคำตอบใหม่ที่หาได้แล้วทำการค้นหาโดยใช้โครงสร้างการปรับปรุงคำตอบในย่านใกล้เคียง (Neighborhood Structure) เดิม
- หากคำตอบที่ได้ไม่ดีขึ้นให้เพิ่มลำดับ Neighborhood Structure ขึ้นแล้ววนซ้ำจากตอนที่ 2 จนได้ลำดับของ Neighborhood Structure เท่ากับกรอบการค้นหาที่กำหนดไว้จึงหยุดการค้นหา

ซึ่งวิธีการสำรวจคำตอบใกล้เคียงนั้นเป็นวิธีการพัฒนาคำตอบที่ดีวิธีหนึ่ง แต่ไม่มีทฤษฎีที่แน่นอนว่า การออกแบบการค้นหาตำแหน่งที่ต้องการปรับเปลี่ยนนั้นเป็นอย่างไร และวิธีการที่จะสร้างคำตอบใหม่จากตำแหน่งนั้น ๆ ดังนั้นการประยุกต์นำวิธีนี้มาทำการปรับปรุงคำตอบโดยการนำพื้นฐานการสร้างจุดที่ค้นหาคำตอบ และการปรับปรุงจึงได้ทำการออกแบบทำการสร้างและการปรับปรุงคำตอบใกล้เคียง

นักวิจัยจึงนิยมนำวิธีการค้นหาคำตอบใกล้เคียงแบบแปรผัน (Variable Neighborhood Search: VNS) มาปรับปรุงโครงสร้างการปรับปรุงคำตอบในย่านใกล้เคียง (Neighborhood Structure: NS,) ด้วยรูปแบบที่หลากหลายแตกต่างกันออกไป ทั้งนี้เพื่อเป็นการเพิ่มประสิทธิภาพใน การค้นหาคำตอบ แต่อย่างไรก็ตามการปรับปรุง NS นี้ก็ทำให้กระบวนการทำงานของวิธีค้นหา คำตอบใกล้เคียงแบบแปรผัน (VNS) มีความซับซ้อนเพิ่มขึ้นตามไปด้วย เวลาในการค้นหาคำตอบก็ จะเพิ่มขึ้น ขั้นตอนทางคอมพิวเตอร์ของวิธีค้นหาคำตอบใกล้เคียงแบบแปรผัน (VNS) ที่คิดค้นโดย Pierre Hansen และ Mladenovic Nenad ดังนี้

ขั้นตอนที่ 1 กำหนดฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 2 ทำการเลือกเซตของโครงสร้างการปรับปรุงคำตอบย่านคำตอบใกล้เคียง (Neighborhood Structures: NS(Q)); n= 1,..., nmax

ขั้นตอนที่ 3 จากนั้นกำหนดค่าคำตอบเริ่มต้น Q และเลือกเงื่อนไขของการหยุดค้นหา ทำซ้ำขั้นตอนต่อไปนี้จนกระทั่งเงื่อนไขของการหยุดค้นหาเป็นจริง

- กำหนดให้ t =1
- ดำเนินการค้นหาตามขั้นตอนต่อไปนี้จนกว่าค่า t = max

ขั้นตอนที่ 4 Shaking สร้างจุด Q' ขึ้นมาอย่างสุ่มจากโครงสร้างการปรับปรุงคำตอบย่านคำตอบใกล้เคียง ลำดับที่ n ของ Q ($Q' \in NS_n(Q)$)

ขั้นตอนที่ 5 Local Search เลือกวิธีการค้นหาเฉพาะที่เพื่อใช้ในการค้นหาค่าที่เหมาะสมเฉพาะที่ (Local optimum) Q'' โดยถือว่า Q' คือค่าเริ่มต้นของการค้นหา

ขั้นตอนที่ 6 Neighborhood Changes เปรียบเทียบผลที่ได้ ถ้าหากค่าเหมาะสมเฉพาะที่ Q" ที่ทำการค้นหามาได้ นำไปแทนลงในฟังก์ชันวัตถุประสงค์แล้วทำให้ฟังก์ชันวัตถุประสงค์มีการพัฒนาเมื่อเทียบกับการแทน Q ลงไป เช่น ต้องการหาค่า Q ที่ทำให้ f(Q) มีค่าเข้าใกล้ ศูนย์มากที่สุด

ดังนั้น ถ้าหาก f(Q'') <f(Q) จะทำการสลับค่าให้ Q ← Q" และดำเนินการ ค้นหาค่าเหมาะสมที่สุดต่อไปในย่านคำตอบย่านเดิม $NS_n(Q)$ แต่ถ้าฟังก์ชันวัตถุประสงค์ ไม่มีการพัฒนาขึ้น จะทำการเปลี่ยน โครงสร้างการปรับปรุงคำตอบย่านคำตอบใกล้เคียง ในการค้นหาคำตอบโดยทำการเพิ่มค่าของ n โดย n = n +1 และกลับไปสู่ขั้นตอนที่ 4 อีกครั้ง

จากขั้นตอนการทำงานของวิธีกระบวนการวิธีการค้นหาคำตอบใกล้เคียง แบบแปรผัน (VNS) Mladenovic et al. (2008) กล่าวว่ากระบวนการวิธีการค้นหาคำตอบใกล้เคียง แบบแปรผัน (Variable Neighborhood Search: VNS) ได้ถูกออกแบบมาเพื่อใช้ในการแก้ปัญหาได้ในหลาย ๆ รูปแบบไม่ว่าจะเป็นปัญหาที่มีลักษณะคำตอบเป็นแบบต่อเนื่องหรือไม่ต่อเนื่อง และยังสามารถนำไปใช้สำหรับการแก้ปัญหาโปรแกรมเชิงเส้น (Linear program) ปัญหาโปรแกรมไม่เชิงเส้น (Nonlinear program) ปัญหาโปรแกรมแบบผสมจำนวนจริง (Mixed integer programs) ปัญหาโปรแกรมแบบจำนวนเต็ม (Integer program) เป็นต้น และภายในกระบวนการวิธีการค้นหาคำตอบใกล้เคียงแบบแปรผัน (VNS) สิ่งที่ต้องกำหนดขึ้นมาก่อนที่จะเริ่มกระบวนการประกอบไปด้วย

(1) ฟังก์ชันวัตถุประสงค์ที่ต้องการแก้ปัญหาเพื่อหาคำตอบที่เหมาะสมที่สุด

(2) NS_n หมายถึง รูปแบบโครงสร้างคำตอบในย่านใกล้เคียง เมื่อ n คือ จำนวนรูปแบบของโครงสร้างคำตอบในย่านใกล้เคียงที่เป็นไปได้ $n = 1 \ldots n_{max}$ จะถูกสร้างขึ้นโดยผู้วิจัยเพื่อนำมาในการค้นหาค่าที่เหมาะสมที่สุด โดยรูปแบบหรือจำนวนโครงสร้างคำตอบในย่านใกล้เคียงจะขึ้นอยู่กับผู้วิจัยเป็นผู้กำหนดแต่โดยมากแล้วอาจใช้เพียงหนึ่งรูปแบบเท่านั้น ในการกำหนดโครงสร้างคำตอบในย่านใกล้เคียงสำหรับการค้นหาคำตอบ นั่นคือ Tmax = 1

(3) ค่าเริ่มต้นของการค้นหา Q ซึ่งอาจได้มาจากเลขสุ่มที่สร้างขึ้นโดยมีการแจกแจงแบบต่าง ๆ หรือสร้างมาจากวิธีการอื่นซึ่งจะมีความเหมาะสมต่างกันไปแล้วแต่สถานการณ์

เงื่อนไขของการหยุดค้นหาจะขึ้นอยู่กับผู้วิจัยเป็นผู้กำหนด อาจเป็นระยะเวลาที่มากที่สุดในการค้นหาที่ยอมรับได้หรือจำนวนรอบของการค้นหาที่มากที่สุดที่ยอมรับได้ เป็นต้น

(4) วิธีการค้นหาคำตอบเฉพาะที่ซึ่งจะนำมาใช้ในขั้นตอน Local Search ขั้นตอนนี้ จะเป็นการค้นหาคำตอบเฉพาะที่ (Local Search: LS) เป็นวิธีการหาค่าคำตอบที่ดีที่สุดโดยอาศัยหลักการประมาณวิธีหนึ่ง ที่สามารถนำมาประยุกต์ใช้กับปัญหาที่มีขนาดใหญ่ และความซับซ้อนสูงที่ปกติไม่สามารถหาคำตอบที่ดีที่สุดได้ในระยะเวลาอันสั้น ข้อดีคือมีความซับซ้อนไม่มากนัก อีกทั้ง

สามารถแก้ปัญหาขนาดใหญ่ที่ซับซ้อนได้ในเวลาไม่นาน แต่ข้อเสียคือผลลัพธ์ที่มักจะไปติดที่ค่าคำตอบที่ดีที่สุดในย่านคำตอบใกล้เคียง การปรับปรุงคำตอบเฉพาะที่นี้ดำเนินการได้หลายวิธีขึ้นอยู่กับประเภทของปัญหาที่จะแก้ไข เช่นการสลับตำแหน่ง (SWAP) การสลับสองตำแหน่ง (2-opt) การย้ายลำดับลูกค้าหนึ่งรายระหว่างเส้นทาง (One move operator) เป็นต้นกระบวนการ LS มีหลายรูปแบบ แต่จะยกตัวอย่างรูปแบบที่ถูกนำไปใช้งานในงานวิจัยนี้ คือวิธีค้นหาคำตอบแบบวนรอบซ้ำ (Iterated Local Search: ILS)

วิธีการค้นหาคำตอบเฉพาะที่แบบวนซ้ำเป็นเมตาฮิวริสติกที่พัฒนามาจากวิธีการปรับปรุงคำตอบเฉพาะที่พื้นฐาน (Basic Local Search: BLS) โดยที่แนวคิดของวิธีการปรับปรุงคำตอบเฉพาะที่พื้นฐานคือค้นหาจุดที่ดีที่สุดในพื้นที่หนึ่งที่จำกัดในพื้นที่ที่เป็นไปได้วิธีการค้นหาคำตอบเฉพาะที่แบบวนซ้ำนำข้อดีของวิธีการปรับปรุงคำตอบเฉพาะที่พื้นฐาน มาใช้คือหาคำตอบที่ดี ที่สุดในแต่ละพื้นที่หลังจากนั้นจะรบกวนคำตอบ (Perturbation) เพื่อให้ออกจากพื้นที่เดิม (Escape) แล้วค้นหาคำตอบที่ดีที่สุดในพื้นที่ใหม่จากหลักการพื้นฐานดังกล่าว การประยุกต์ใช้วิธีการค้นหา คำตอบเฉพาะที่แบบวนซ้ำจะขึ้นอยู่กับ 2 ประการหลัก คือคุณภาพของการปรับปรุงคำตอบเฉพาะที่ (Local Search) และเทคนิคการรบกวนคำตอบ (Perturbation techniques)

2.3.3.4 วิธีค้นหาแบบตาบู (Tabu Search)

การค้นหาแบบตาบู หรือ TS เป็นวิธีการ ค้นหาคำตอบที่เหมาะที่สุดในปัญหาแบบไม่ต่อเนื่องเชิงผสมผสาน อัลกอริทึมของ TS อาศัยกระบวนการค้นหาค่าใกล้เคียง (neighborhood search) และรายชื่อตาบู (Tabu list, TL) ซึ่งจะทำหน้าที่เก็บคำตอบโนอดีตคำตอบโน TL สามารถใช้ประโยชนในกรณีการค้นหาเกิดการล็อกโดย คำตอบวงแคบเฉพาะถิ่นโดยอาศัยเกณฑ์ปรารถนา (aspiration criteria, AC) กระบวนการค้นหาจะเคลื่อนที่แบบ ดีเทอร์มินิสติก ซึ่งจะถูกกำหนดไว้อย่างมีแบบแผน โดยทั่วไปจะถูกกำกับไว้ด้วยเงื่อนไขความคงอยู่ล่าสุด (recency) และเงื่อนไขความซ้ำ (frequency) นอกจากนี้ TS ยังมีกลไกที่ทำหน้าที่เพิ่มสมรรถนะการค้นหาคำตอบอีก สองกลไกด้วยกันคือ กลไกความเข้มข้น (intensification) และกลไกความหลากหลาย (diversification) อัลกอริทึม ของ TS ประกอบด้วย

1. กำหนดขอบเขตปริภูมิการค้นหา (search space), รัศมีการค้นหา (R), ตัวนับ (counter), TL, และเกณฑ์ยุติการค้นหา (termination criteria, TC)

2. สุ่มเลือกคำตอบเริ่มต้น, S0, ภายในปริภูมิการ ค้นหา โดยกำหนดให้ S0 มีค่าเป็น best_neighbor ปัจจุบัน ดังแสดงในรูปที่ 2.5

3. ผลิตค่าใกล้เคียง (neighborhood) แบบสุ่มรอบคำตอบเริ่มต้น S0 ภายในปริภูมิการค้นหารัศมี Rจำนวน N ตัว แล้ว เก็บไว้ในเซต S1 (r) ดังแสดงในรูปที่ 2.5

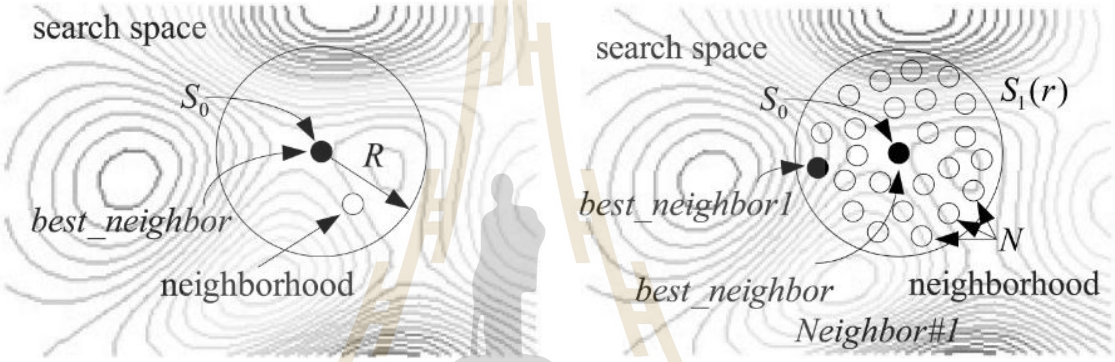
4. ประเมินค่าใกล้เคียงด้วยฟังก์ชันวัตถุประสงค์โดยกำหนดให้ S1คือค่าใกล้เคียง

ที่ทำให้ฟังก์ชันวัตถุประสงค์มีค่าน้อยที่สุด และให้ S1 มีค่าเป็น best_neighborl ดังแสดงในรูปที่ 2.5

5. ถ้า S1 <S0 ทำการเก็บ S0 ไว้ใน TL จากนั้น ปรับค่า S0 = S1 ไม่เช่นนั้นแล้วให้ทำการเก็บ S1 ไว้ใน TL แทน ดังแสดงในรูปที่ 2.7

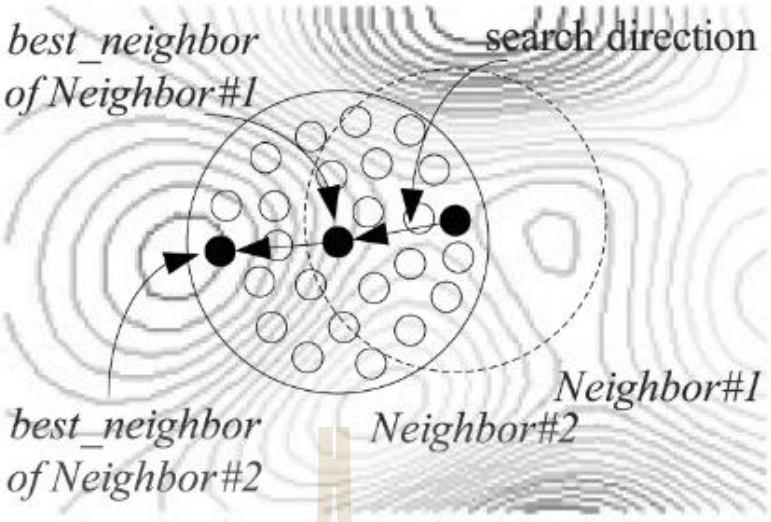
6. ตรวจสอบ TC ถ้า TC สอดคล้อง ให้ยุติการค้นหาค่า S0 คือคำตอบที่ดีที่สุด ไม่เช่นนั้นให้ปรับเพิ่มค่าตัวนับแล้ววนกลับไปยัง (3) เพื่อดำเนินการค้นหาในรอบต่อไป

อัลกอริทึมของ TS จะยุติการค้นหาเมื่อ TC สอดคล้อง โดยทั่วไปจะอาศัยเงื่อนไขเกี่ยวกับจำนวนรอบ การค้นหาที่ต้องการ หรือค่าความคลาดเคลื่อนที่ยอมรับ คำตอบที่ดีที่สุด ที่พบคือคำตอบที่เหมาะสม (optimum solution) หรือคำตอบวงกว้าง



รูปที่ 2.5 การสุ่มเลือกคำตอบเริ่มต้น S0 และการผลิตค่าใกล้เคียงแบบสุ่ม

รูปที่ 2.6 การเปรียบเทียบค่าใกล้เคียงและการปรับค่าคำตอบเริ่มต้นใหม่



รูปที่ 2.7 ผลิตค่าใกล้เคียงแบบสุ่มรอบใหม่

2.3.3.5 การค้นหาแบบตาบูเชิงปรับตัว (ATS)

การค้นหาแบบตาบูเชิงปรับตัว หรือ ATS เป็นวิธีการค้นหาคำตอบที่เหมาะที่สุดที่พัฒนามาจาก TS โดยมีกลไกเพิ่มเติมได้แก่กลไกย้อนรอยการค้นหา (back-tracking mechanism) เพื่อแก้ปัญหาการล็อกโดยคำตอบวงแคบเฉพาะถิ่นและกลไกปรับรัศมีการค้นหา (adaptive radius mechanism) เพื่อเพิ่มความเร็วให้กับกระบวนการค้นหาคำตอบอัลกอริทึมของ ATS ประกอบด้วย

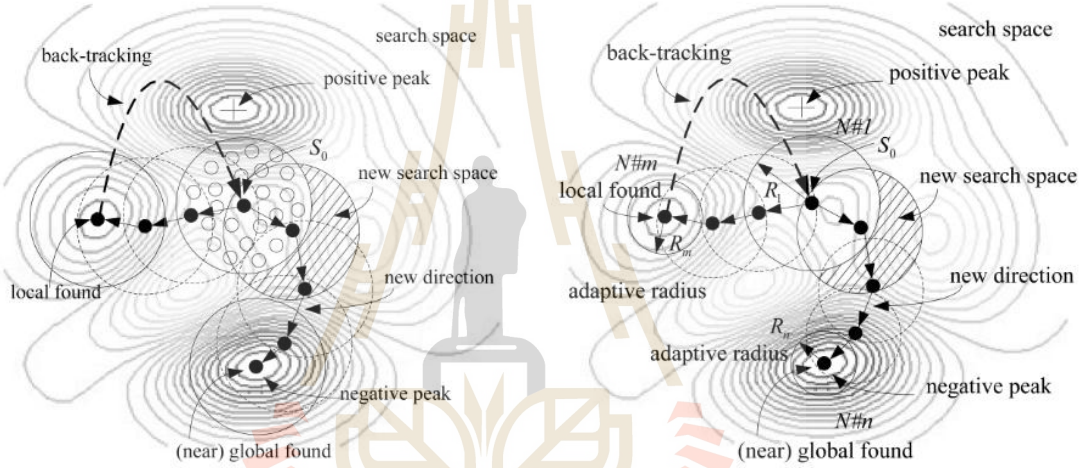
1. กำหนดขอบเขตปริภูมิการค้นหา, R, ตัวนับ, TL, และ TC
2. สุ่มเลือกคำตอบเริ่มต้น, S_0, ภายในปริภูมิการค้นหาโดยกำหนดให้ S_0 มีค่าเป็น best_neighbor ปัจจุบัน ดังแสดงในรูปที่ 2.5
3. ผลิตค่าใกล้เคียงแบบสุ่มรอบคำตอบเริ่มต้น, S_0 , ภายในปริภูมิการค้นหารัศมี R จำนวน N ตัวแล้วเก็บไว้ในเซต $S_1(r)$ ดังแสดงในรูปที่ 2.5
4. ประเมินค่าใกล้เคียงด้วยฟังก์ชันวัตถุประสงค์โดยกำหนดให้ S_1 คือค่าใกล้เคียงที่ทำให้ฟังก์ชันวัตถุประสงค์มีค่าน้อยที่สุดและให้ S_1 มีค่าเป็นbest_neighbor1 ดังแสดงในรูปที่ 3
5. ถ้า $S_1 < S_0$ ทำการเก็บ S_0 ไว้ใน TL จากนั้นปรับค่า $S_0 = S_1$ ไม่เช่นนั้นแล้วให้ทำการเก็บ S_1 ไว้ใน TL แทนดังแสดงในรูปที่ 2.7
6. เรียกใช้กลไกย่อนรอยการค้นหาเมื่อกระบวนการค้นหาเกิดการล็อกโดยคำตอบวงแคบเฉพาะถิ่นดังแสดงในรูปที่ 2.8
7. ตรวจสอบ TC ถ้า TC สอดคล้องให้ยุติการค้นหาค่า S_0 คือคำตอบที่ดี

ที่สุดไม่เช่นนั้นให้ทำขั้นตอนต่อไป

8. เรียกใช้กลไกปรับรัศมีการค้นหาเมื่อกระบวนการค้นหาเข้าใกล้คำตอบ ดังแสดงในรูปที่ 2.8

9. ปรับเพิ่มค่าตัวนับแล้ววนกลับไปยัง (3) เพื่อดำเนินกระบวนการค้นหาในรอบต่อไป

เช่นเดียวกับ TS อัลกอริทึมของ ATS จะยุติการค้นหาเมื่อ TC สอดคล้องโดยทั่วไปจะอาศัยเงื่อนไขเกี่ยวกับจำนวนรอบการค้นหาที่ต้องการหรือค่าความคลาดเคลื่อนที่ยอมรับคำตอบที่ดีที่สุดที่พบคือคำตอบที่เหมาะสมหรือคำตอบวงกว้างนั่นเองสำหรับรายละเอียดของ ATS



รูปที่ 2.8 กลไกย้อนรอยการค้นหาและกลไกปรับรัศมีการค้นหา

2.3.3.6 วิธีค้นหาค่าเหมาะที่สุดแบบกลุ่มอนุภาค (Particle Swarm Search)

การหาค่าเหมาะที่สุดแบบกลุ่มอนุภาพ (Particle Swamp Optimization) หรือ PSO เป็นเทคนิคทางสถิติที่อาศัยหลักการทำงานของกลุ่มประชากรเช่นเดียวกับอัลกอริธึมพันธุการและการหาค่าเหมาะที่สุดแบบฝูงมด การหาค่าเหมาะที่สุดแบบกลุ่ม อนุภาพนี้พัฒนาโดยอีเบอร์ฮาร์ดและเคนเนดี O' a' Kennedy, Particle Swarm Optimization 1995) ที่จำลองวิธีการหาค่าเหมาะที่สุดมาจากพฤติกรรมการเดินทางของฝูงสัตว์ โดยเฉพาะฝูงนก และฝูงปลาในการหาอาหาร

รูปที่ 2.9 พฤติกรรมของนกที่บินเป็นฝูง

เรยโนล์ด (Reynolds, 1987) และ แอบเนอร์ กับ เจอร์แนนเดอร์ (Heppner, 1990) ได้จำลองรูปแบบการบินของฝูงนก พร้อมกับศึกษาพฤติกรรมของนกที่บินเป็นฝูง เขาได้ค้นพบว่านกแต่ละตัวมีการประสานงานกันในระหว่างการบิน โดยที่นกแต่ละตัวจะพยายามรักษาความห่างระหว่างกันคือของตัวมันเอง และนกตัวที่อยู่รอบข้างมันทั้งหมดให้เหมาะสม เพื่อที่พวกมันจะได้ไม่บินชนกันเอง แม้ว่านกทั้งฝูงจะเปลี่ยนทิศทางการบินอย่างรวดเร็วก็ตาม เมื่อฝูงนกพบสิ่งกีดขวาง พวกมันก็จะกระจายกันออกไป และกลับเข้ามารวมกลุ่มกันใหม่โดยที่ไม่มีการชนกันเอง ดังนั้น PSO จึงใช้วิธีการค้นหาคำตอบด้วยการใช้อนุภาค (Particles) จำนวนมาก เคลื่อนที่ไปบนพื้นที่ที่ต้องการค้นหา (Search Space) เพื่อค้นหาคำตอบที่ดีที่สุด

คำจำกัดความ และค่าตัวแปร พารามิเตอร์ต่าง ๆ ที่จะปรากฏใน PSO

- Particle คือสมาชิก 1 ตัวที่อยู่ในกลุ่มประชากร ผมขอให้นึกถึงฝูงนกที่มี Particle (นก) มารวมกันจนกลายเป็นฝูงนก โดย Particle แต่ละตัวจะมีความเร็ว (Velocity) และตำแหน่ง (Position) ซึ่ง Particle จะมีการรับรู้ตำแหน่งของมัน ณ ปัจจุบัน และรู้คำตอบของตำแหน่งนั้น ๆ บน Search Space ซึ่งในตำแหน่งที่ดีที่สุดนั้นเราจะเรียกมันว่า Personal Best Position
- กลุ่มประชากร คือเซตของ Particle ที่มี n ตัว ซึ่งมีตั้งแต่ 1 ถึง n หรืออาจจะเรียกว่า ฝูง (Swarm)
- Position ของ Particle ตัวที่ i ที่ทำการวนซ้ำครั้งที่ t ถูกเขียนแทนด้วย Xi(t) โดยตำแหน่งดังกล่าวประกอบด้วย D มิติ คือ Xi(t) = (xi1(t),…, xid(t),…,xiD(t)) โดยที่ xid(t) คือค่าของตำแหน่ง

ของมิติที่ d ของตัว Particle ตัวที่ i แต่ละตำแหน่ง Xi(t) สามารถแปลงเป็นคำตอบ (Solution) ของปัญหาทางคณิตศาสตร์ โดยค่านอกตำแหน่งมีถูกจำกัดในขอบเขต [Xmin, Xmax]

- ค่าความเหมะสม (Fitness Value) คือ f(Xi(t)) คือค่าของคำตอบที่แปลงมาจากตำแหน่ง Xi(t) โดยจะถูกเรียกว่าค่าความเหมาะสมของตำแหน่ง Xi(t)
- ความเร็ว (Velocity) คือ Vi(t) แทนค่าความเร็วของตัว Particle ตัวที่ i ที่ทำการวนซ้ำครั้งที่ t คือถูกเขียนแทนด้วยค่าเวคเตอร์ที่มีมิติ D มิติ คือ Vi(t) = (vi1(t),…, vid(t),…, viD(t)) โดย vid(t) คือค่าของความเร็วที่มิติที่ d ของตัวพาร์ทิเคิลตัวที่ i ที่การวนซ้ำครั้งที่ t และ Vi(t + 1) คืออัตราเร็วที่ตัว Particle ตัวที่ i จะเคลื่อนที่จากตำแหน่ง Xi(t) ไปตำแหน่ง Xi(t + 1)
- ความเร็วสูงสุด (Maximum Velocity) คือ Vmax คือขีดจำกัดของความเร็ว โดยแต่ละ vid(t) ไม่สามารถมีค่าออกนอกช่วง [-Vmax, Vmax] หรืออาจพูดง่าย ๆ ว่านกแต่ละตัวต้องมีความเร็วสูงสุด และต้องไม่ต่ำไปไม่งั้นจะเคลื่อนที่ไม่ได้
- น้ำหนักแรงเฉื่อย (Inertia Weight) คือ w(t) ซึ่งเป็นพารามิเตอร์ตัวที่ใช้ในการควบคุมผลกระทบของความเร็วของการทำงานก่อนหน้า ซึ่งตัวนี้จะทำให้ Particle ของเราทั้งหมดได้รับการกำหนดค่าลงไปด้วย
- Personal Best Position คือ Pi คือตำแหน่งที่ถูกพบโดยตัว Particle ตัวที่ i ที่มีค่าความเหมาะสมที่ดีที่สุด
- Global Best Position คือ Pg เป็นสัญลักษณ์ที่ใช้แทนตำแหน่งที่ดีสุด คือตำแหน่งที่ดีที่สุดที่ถูกพบโดยฝูง
- Maximum Iteration คือ T ใช้เขียนแทนค่าการวนซ้ำสูงสุด โดยจะหยุดการวนเมื่อจำนวนการวนเท่ากับค่าสูงสุด

มหาวิทยาลัยเทคโนโลยีสุรนารี

ทำงานของอัลกอริธึมนี้เป็นรหัสเทียม (Pseudo code) จะได้ดังนี้

```
For อนุภาคแต่ละตัว
Do
        กำหนดค่าเริ่มต้นชุดหนึ่งให้กับอนุภาคทุกตัว
End Do
While not (จำนวนรอบสูงสุด หรือบรรลุวัตถุประสงค์ของการค้นหา)
For อนุภาคแต่ละตัว
Do
    คำนวณค่า ฟิตเนส
    If ค่า ฟิตเนส นี้ดีกว่าค่า pbest Then ให้ค่า ฟิตเนส นี้แทนที่ค่า pbest End Do
เลือกอนุภาคที่มีค่า ฟิตเนส ดีสุดของทั้งกลุ่มมาเปรียบเทียบกับ gbest If ค่าที่ได้ดีกว่า gbest Then
ให้ค่าที่ได้เป็นค่า gbest ใหม่ For อนุภาคแต่ละตัว
Do..
        คำนวณค่าความเร็วใหม่ของอนุภาคตามสมการ
        v[] = v[] + C1*rand()*(pbest[]-present[]) + c2*rand()*(gbest[]-
        present[])
        ปรับปรุงค่าตำแหน่งใหม่ของอนุภาคตามสมการ
        present[] = p resent[] + v[]
End Do
End while
```

สรุปขั้นตอนการทำงานและสูตรที่ใช้คำนวณให้ดังนี้

1. สร้างอนุภาคโดยสุ่มตำแหน่งเริ่มต้นให้กับอนุภาคและกำหนดค่า T = 1 จากนั้นกำหนดความเร็วและตำแหน่งให้กับ Particle

2. คำนวณหาค่าความเหมาะสม ถ้าหากว่าค่าความเหมาะสมที่ได้มีค่ามากกว่าค่าเดิม (Personal Best Position) ให้เปลี่ยนไปใช้ค่าที่มากกว่า

3. เลือกค่าความเหมาะสมที่ดีที่สุดจากอนุภาคทุกตัวแล้วใช้เป็นค่าที่ดีที่สุด (Global Best Position)

4. ในทุก ๆ รอบให้กำหนดตำแหน่งและความเร็วใหม่

5. เคลื่อนที่อนุภาคไปยังตำแหน่งใหม่ ทำวนซ้ำจนได้คำตอบหรือครบจำนวนรอบที่กำหนด

2.3.4 การหาค่าที่เหมาะสมที่สุดที่มีหลายวัตถุประสงค์

การหาค่าที่เหมาะสมที่สุด (Optimization) คือกระบวนการหาคำตอบที่ดีที่สุดของปัญหา โดยคำตอบนั้นจะต้องไม่ละเมิดต่อข้อจำกัดหรือเงื่อนไขที่มีอยู่ ปัญหาการหาค่าที่เหมาะสมหลายวัตถุประสงค์ (Multi-Objective Optimization Problem) โดยในการหาค่าที่เหมาะสมที่สุด จะประกอบไปด้วยเวกเตอร์ตัวแปรตัดสินใจ (Vector of Decision Variables), ข้อจำกัด (Constraints) และฟังก์ชันวัตถุประสงค์(Objective Functions)

ฟังก์ชันวัตถุประสงค์ คือเกณฑ์ที่เป็นตัวกำหนดว่าเป้าหมายในการหาค่าที่เหมาะสมเป็นไปในลักษณะใด โดยส่วนใหญ่แล้วจะมีเป้าหมายเพื่อหาค่าที่มากที่สุด (Maximization) หรือน้อยที่สุด (Minimization) การแก้ปัญหาการหาค่าที่เหมาะสมที่สุดที่มีหลายวัตถุประสงค์ คือการหาของคำตอบภายในพื้นที่คำตอบที่เป็นไปได้ (Feasible Region) ที่ทำให้ฟังก์ชันวัตถุประสงค์ทุกฟังก์ชันที่กำหนดมีค่าที่ดีที่สุดพร้อมๆกันไปซึ่งเป้าหมายของเหล่าฟังก์ชันวัตถุประสงค์อาจเป็นรูปแบบที่ทุกฟังก์ชันต้องการค่าน้อยที่สุดหรือมากที่สุด

เนื่องจากปัญหาการจัดสมดุลสายการประกอบนั้นเป็นปัญหาแบบ NP-Hard ที่การหาเซตคำตอบที่ดีที่สุดนั้นเป็นไปได้ยากมาก หรืออาจไม่สามารถหาเซตคำตอบที่ดีที่สุดได้เลย จึงทำให้เกิดอัลกอริทึมต่างๆขึ้นมาเพื่อมีเป้าหมายที่จะหาเซตคำตอบที่ดีที่สุดหรือมีค่าที่ใกล้เคียงที่สุด ซึ่งปกติแล้วการประมาณเซตคำตอบที่ดีจะเกี่ยวข้องกับสองกระบวนการ ได้แก่ การกำหนดค่าความแข็งแรง (Fitness Assignment) ให้กับคำตอบ และการกระจายประชากรคำตอบ(Population Diversify) เพื่อให้ในท้ายที่สุดจะได้เซตคำตอบที่ดีซึ่งมีลักษณะของขอบเขตใกล้เคียงกับขอบเขตของกลุ่มคำตอบที่แท้จริง และควรเป็นเซ็ตของกลุ่มคำตอบที่แท้จริง คำตอบมีการกระจายตัวแบบยูนิฟอร์ม (Uniform Distribution) ไม่เกาะอยู่บนบริเวณใดบริเวณหนึ่งควรจับสเปกตรัมของขอบเขตกลุ่มคำตอบที่แท้จริงทั้งหมดได้ กล่าวคือ สามารถสืบค้นถึงคำตอบที่อยู่ปลายสุดของกลุ่มคำตอบในพื้นที่ฟังก์ชันวัตถุประสงค์ทั้งหมดได้

การกำหนดค่าความแข็งแรง (Fitness Assignment) นั้นมีวัตถุประสงค์เพื่อจัดอันดับคำตอบที่ดีที่สุดในกลุ่มคำตอบที่กำลังพิจารณา โดยวิธีการกำหนดค่าความแข็งแรงนั้นมีอยู่ เช่น วิธีการโกล์โปรแกรมมิ่ง (Goal Programming Approach), วิธีการประเมินผลเวกเตอร์(Vector Evaluation Approach), วิธีการรวมถ่วงน้ำหนัก(Weighted Sum Approach) และรวมไปถึงวิธีเชิงกลุ่มที่ดีที่สุด (Pareto-Based Approach) โดยวิธีเชิงกลุ่มที่ดีสุดนี้สามารถแบ่งออกได้ในหลายแนวทาง อาทิเช่น วิธีการการจัดอันดับของ Goldberg (หรือมีอีกชื่อ Non-Dominated Sorting), วิธีการจัดอันดับของ Fonseca และ Fleming, วิธีการจัดอันดับแบบ Accumulate Ranking Density Strategy (AARS) และวิธีการจัดอันดับแบบ Strength of Dominators

2.4 สรุป

จากการทบทวนปริทัศน์วรรณกรรม และงานวิจัยที่เกี่ยวข้องกับการจัดตารางการผลิต จากฐานข้อมูล IEEE Science Direct IEE และอื่น ๆ ซึ่งทำให้ทราบถึงแนว ทางการวิจัยที่เกี่ยวข้อง ระเบียบวิธีที่ผู้วิจัยอื่น ๆ ได้นำมาใช้ผลการดำเนินงาน ข้อเสนอแนะต่าง ๆ จากคณะนักวิจัยตั้งแต่อดีตจนถึงปัจจุบันสามารถพิจารณาการจัดตารางการผลิตด้วยวิธีต่าง ๆ ภายใต้เงื่อนไขข้อกำหนดที่ต่างกัน โดยใช้แบบจำลองทางคณิตศาสตร์และการประมาณค่าด้วยวิธีปัญญาเชิงคำนวณ เพื่อหาตารางการผลิตแบบที่ดีที่สุดที่สามารถลดค่าใช้จ่ายโดยรวมของระบบ ซึ่งประกอบด้วยค่าใช้จ่ายจากการผลิต ค่าใช้จ่ายในการเก็บงานที่ผลิตเสร็จก่อนกำหนด และค่าใช้จ่ายในการปรับตั้งเครื่องจักร

และในบทนี้ยังได้นำเสนอทฤษฎีและหลักการต่าง ๆ ที่เกี่ยวข้องกับงานวิจัยซึ่งได้แก่ทฤษฎีที่เกี่ยวกับการจัดการตารางการผลิต แบบจำลองทางคณิตศาสตร์การจัดตารางการผลิตด้วยเครื่องจักรขนาน การหาคำตอบที่เหมาะสมด้วยวิธีปัญญาเชิงคำนวณ และการหาค่าที่เหมาะสมที่สุดแบบหลายวัตถุประสงค์

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 3
การจัดตารางการผลิตโดยใช้แบบปัญหาทางคณิตศาสตร์

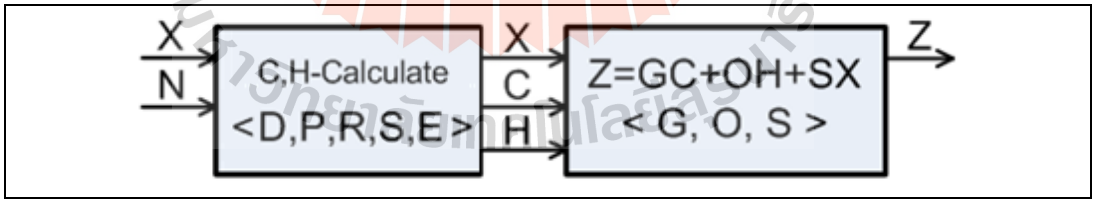
3.1 บทนำ

การใช้ในวิธีสร้างรูปแบบทางคณิตศาสตร์เพื่อหาลำดับการผลิตที่เหมาะสมที่สุดในสายการผลิตเป็นที่ทราบกันว่า วิธีนี้ใช้กำลังในการคำนวณค่อนข้างมาก แต่คำตอบที่ได้เป็นผลลัพธ์ที่ดีที่สุด

รูปแบบทางคณิตศาสตร์ที่จะสร้างขึ้นเพื่อหาตารางการผลิตนี้ จะอยู่ภายใต้เงื่อนไขต่าง ๆ ดังนี้ เครื่องจักรมีการจัดเรียงแบบขนาน, ลำดับการผลิตมีผลต่อเวลาติดตั้งเครื่องจักร, พิจารณาขนาดการสั่งผลิตขั้นต่ำ และมีข้อจำกัดของเครื่องจักร คือ งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ โดยวัตถุประสงค์เพื่อทำให้ค่าใช้จ่าย และเวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด โดยใช้ integer programming

ผลการทำงานจะแสดงในรูปตารางลำดับการทำงานตามรอบเวลางาน และตารางลำดับงานของเครื่องจักรแต่ละตัว

3.2 พารามิเตอร์และตัวแปรตัดสินใจ



รูปที่ 3.1 ตัวแปรที่ใช้ในการคำนวณในแบบปัญหา

การคำนวณในแบบปัญหาแต่ละแบบการผลิตจะเริ่มจากการมี ตารางลำดับงาน(ตัวแปร X) และจำนวนงานที่ผลิตรวมทุกรอบงาน (ตัวแปร N) จากนั้นคำนวณว่าเมื่อผลิตตามจำนวนและลำดับแต่ละรอบงานจะทำให้เกิดการผลิตกี่ชิ้นงาน (ตัวแปร C) มีชิ้นงานต้องเก็บจำนวนเท่าใด (ตัวแปร H) ขั้นสุดท้ายคือการคำนวณเวลาในการผลิตรวมของระบบ (ตัวแปร C)

การสร้างรูปแบบทางคณิตศาสตร์เพื่อหาลำดับการผลิตที่เหมาะสมที่สุดในสายการผลิตที่เครื่องจักรจัดเรียงแบบขนาน เมื่อพิจารณาเงื่อนไขต่างดังนี้

(1) การผลิตทำโดยการใช้เครื่องจักรที่มีการจัดเรียงแบบขนาน

(2) ลำดับการผลิตมีผลต่อเวลาติดตั้งเครื่องจักรการผลิต

(3) พิจารณาการผลิตแบบขนาดสั่งผลิต

(4) ข้อจำกัดของเครื่องจักร คือ งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้

เมื่อพิจารณาการจัดตารางการผลิตของงานโดยมีวัตถุประสงค์เพื่อทำให้ ค่าใช้จ่ายและเวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด โดยใช้ integer programming

คำอธิบายความหมายของพารามิเตอร์และตัวแปรตัดสินใจที่ใช้ในการสร้างรูปแบบทางคณิตศาสตร์

Parameters

n	=	จำนวนผลิตภัณฑ์
MC	=	จำนวนเครื่องจักร
P	=	จำนวนรอบการทำงาน
M	=	เซตของเครื่องจักร = {1, 2, 3,..., MC}
i, j	=	ตัวชี้ (index) ของผลิตภัณฑ์
m	=	ตัวชี้ (index) ของเครื่องจักร
t	=	ตัวชี้ (index) ของคาบเวลาการผลิต
P_{jm}	=	เวลาที่ใช้ในการผลิต (Production) ต่อหน่วยของผลิตภัณฑ์ j บนเครื่องจักร m ใด ๆ
R_j	=	เวลาที่ใช้ในการปล่อยงาน (Release) ต่อหน่วยของผลิตภัณฑ์ j บนเครื่องจักร m ใด ๆ
	=	เวลาที่ใช้ในการรอผลิตต่อหน่วยของผลิตภัณฑ์ j บนเครื่องจักร m ใด ๆ
s_{ij}	=	ค่าใช้จ่ายในการปรับตั้งเครื่องจักรในการผลิตงาน j เมื่อทำต่อจากงาน i
d_{jt}	=	ความต้องการในการผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t
g_{jt}	=	ค่าใช้จ่ายในการผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t
C_{jt}	=	เวลาที่ใช้การผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t
o_{jt}	=	ค่าใช้จ่ายในการเก็บรักษาของผลิตภัณฑ์ j ในรอบการทำงาน t
H_{jt}	=	จำนวนชิ้นงานที่เก็บรักษาของผลิตภัณฑ์ j ในรอบการทำงาน t
N_{jmt}	=	จำนวนชิ้นงานที่ผลิตชิ้นงาน j บนเครื่องจักร m ในคาบเวลา t
BigM	=	จำนวนจริงที่มีค่ามาก ๆ และมีค่าเป็นบวก (BigM)

Decision variables

H_{jt}	=	จำนวนชิ้นงานที่เก็บรักษาของผลิตภัณฑ์ j ในรอบการทำงาน t
N_{jmt}	=	จำนวนชิ้นงานที่ผลิตชิ้นงาน j บนเครื่องจักร m ในคาบเวลา t

C_{jt} = เวลาที่ใช้การผลิตของผลิตภัณฑ์ j ในรอบการทำงาน t

Binary variables

E_{im} = 1: เมื่องาน i สามารถผลิตบนเครื่องจักร m ได้, = 0: เมื่อเป็นอย่างอื่น

E_{jm} = 1: เมื่องาน j สามารถผลิตบนเครื่องจักร m ได้, = 0: เมื่อเป็นอย่างอื่น

X_{ijmt} = 1: เมื่องาน i ถูกผลิตก่อนงาน j ทันที บนเครื่องจักร m ในรอบการทำงาน t

= 0: เมื่อเป็นอย่างอื่น

Y_{jmt} = 1: เมื่องาน i ถูกผลิตก่อนงาน j ทันที บนเครื่องจักร m ในรอบการทำงาน t

= 0: เมื่อเป็นอย่างอื่น

Z_{jmt} = 1: เมื่องาน i ถูกผลิตก่อนงาน j ทันที บนเครื่องจักร m ในรอบการทำงาน t

= 0: เมื่อเป็นอย่างอื่น

3.3 แบบปัญหาทางคณิตศาสตร์

3.3.1 สมการเป้าหมาย (Objective function)

$$\text{Minimize: } Z = \sum_{j=1}^{n}\sum_{t=1}^{T} g_{jt} \cdot C_{jt} + \sum_{j=1}^{n}\sum_{t=1}^{T} o_{jt} \cdot H_{jt} + \sum_{i=1}^{n}\sum_{j=1,\, j\neq i}^{n}\sum_{m=1}^{M}\sum_{t=1}^{T} s_{ij} \cdot X_{ijmt} \qquad (1)$$

เพื่อให้ค่าใช้จ่าย และเวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด

3.3.2 สมการเงื่อนไข (Constraints)

$$H_{ij} = H_{j,t-1} + \sum_{m=1}^{M} N_{jmt} - d_{jt} \qquad \forall j,t \quad j \neq 1 \qquad (2)$$

จำนวนสินค้าที่เก็บ คือ จำนวนงานที่เก็บในรอบการผลิตก่อนก่อน รวมกับ จำนวนงานที่ผลิตบนเครื่องจักรในรอบผลิตปัจจุบัน หักด้วย จำนวนความต้องการ

$$\sum_{i=1,i\neq j}^{n} X_{ijmt} = 1 - Y_{jmt} \qquad \forall j,m,t \quad j \neq 1 \qquad (3)$$

$$N_{jmt} \leq BigM \cdot (1 - Y_{jmt}) \qquad \forall j,m,t \quad j \neq 1 \qquad (4)$$

$$\sum_{m=1}^{MC} N_{jmt} \geq LowerLotSize \qquad \forall j,t \quad j \neq 1 \qquad (5)$$

ผลรวมของจำนวนชิ้นงาน j ทุกเครื่องจักร ทุกช่วงเวลา ต้อง มากกว่าหรือเท่ากับ Lower lot size

$$\sum_{m=1}^{M} Y_{jmt} = 1 \qquad \forall j,t \quad j \neq 1 \qquad (6)$$

$$Z_{jmt} + Y_{jmt} = 1 \qquad \forall j,m,t \quad j \neq 1 \qquad (7)$$

…

$$C_{jt} - C_{it} + BigM \cdot (1 - (X_{ijmt} \cdot E_{im} \cdot E_{jm})) \geq P_{jm} N_{jmt} + S_{ij} \cdot (X_{ijmt} \cdot E_{im} \cdot E_{jm}) \qquad \forall i,j,m,t \quad i \neq j, j \neq 1 \qquad (8)$$

เวลาแล้วเสร็จของงาน j ใด ๆ จะเท่ากับเวลาแล้วเสร็จของงาน i รวมกับเวลาดำเนินการของงาน j รวมกับเวลาติดตั้งเครื่องจักรของงาน j เมื่องาน j ผลิตตามหลังงาน i บนเครื่องจักรเครื่องเดียวกันและมีข้อแม้ว่าทั้งงาน i และงาน j สามารถผลิตบนเครื่องจักรเครื่องนี้ได้

$$C_{jt} - P_{jm} \cdot N_{jmt} - (S_{ij} \cdot (X_{ijmt} \cdot E_{im} \cdot E_{jm})) \geq R_j \qquad \forall i,j,m,t \quad i \neq j, j \neq 1 \qquad (9)$$

เวลาแล้วเสร็จของงานเมื่อหักด้วยเวลาผลิต และติดตั้งต้องมากกว่าเวลาในการปล่อยงาน

$$C_{jt} \geq WorkHour \qquad \forall j,t \qquad (10)$$

เวลาในการผลิตงาน j ในคาบเวลา t ต้องน้อยกว่าหรือเท่ากับ Work Hour

$$\sum_{i=1}^{n} \sum_{m=1}^{M} X_{ijmt} \cdot E_{im} \cdot E_{jm} = 1 \qquad \forall j,t \quad j \geq 2 \qquad (11)$$

$$\sum_{j=1}^{n}\sum_{m=1}^{M} X_{ijmt} \cdot E_{im} \cdot E_{jm} = 1 \qquad \forall j,t \quad j \geq 2 \qquad (12)$$

งานที่ตามหลังงาน i ใด ๆ จะมีเพียงหนึ่งงานเท่านั้นและทั้งงาน i และหนึ่งงานนั้นจะต้องเลือกผลิตบนเครื่องจักรเพียงเครื่องเดียว

งานที่ผลิตก้อนงาน j ใด ๆ จะมีเพียงหนึ่งงานเท่านั้นและทั้งงาน j และหนึ่งงานนั้นจะต้องเลือกผลิตบนเครื่องจักรเพียงเครื่องเดียว

$$\sum_{i=1}^{n} X_{ijmt} - \sum_{w=1}^{n} X_{jwmt} = 0 \qquad \forall j,m,t \quad i \neq j, j \neq w \qquad (13)$$

สำหรับแต่ละเครื่องจักรถ้ามีงานหนึ่งงานใด ถูกเลือกผลิตก้อนงาน i แล้ว งานนั้นจะไม่สามารถ ถูกเลือกเพื่อผลิตเป็นงานในลำดับถัดจากงาน i ได้อีก

$$X_{jjmt} = 0 \qquad \forall j,m,t \qquad (14)$$

ไม่อนุญาตให้มีกรณีของงาน j ผลิตต่อจากงาน j ได้ ในทุกงาน ทุกช่วงเวลาและทุกเครื่องจักร

$$\sum_{j=2}^{n} X_{1jmt} \cdot E_{im} \cdot E_{jm} = 1 \qquad \forall m,t \quad j \geq 2 \qquad (15)$$

$$\sum_{i=2}^{n} X_{i1mt} \cdot E_{im} \cdot E_{jm} = 1 \qquad \forall m,t \quad j \geq 2 \qquad (16)$$

ทุกเครื่องจักรจะต้องมีอย่างน้อยหนึ่งงานเป็นงานเริ่มต้น

ทุกเครื่องจักรจะต้องมีอย่างน้อยหนึ่งงานเป็นงานที่ผลิตเป็นลำดับสุดท้าย

$$\sum_{m=1}^{M} (X_{ijmt} + X_{jimt}) \cdot E_{im} \cdot E_{jm} \leq 1 \qquad \forall i,j,t \quad i \neq j \qquad (17)$$

งานทุกงานจะต้องได้รับเลือกเพื่อผลิตบนเครื่องจักรเพียงเครื่องเดียวเท่านั้น

มหาวิทยาลัยเทคโนโลยีสุรนารี

$$\sum_{j=1}^{n}\sum_{m=1}^{MC} X_{1jmt} \cdot E_{jm} = MC \qquad \forall t \quad j \neq 1 \qquad (18)$$

$$\sum_{j=1}^{n}\sum_{m=1}^{MC} X_{1jmt} \cdot E_{jm} = MC \qquad \forall t \quad j \neq 1 \qquad (19)$$

ทุกงานจะต้องมีอย่างน้อยหนึ่งเครื่องจักรเป็นเครื่องจักรเริ่มต้น

ทุกงานจะต้องมีอย่างน้อยหนึ่งเครื่องจักรเป็นเครื่องจักรที่ผลิตเป็นลำดับสุดท้าย

จำนวนชิ้นงานที่ผลิต, จำนวนชิ้นงานที่เก็บในคลังสินค้า และจำนวนชิ้นงานที่ผลิตเป็นจำนวนเต็มบวก

$$X_{ijmt}, Y_{jmt}, Z_{jmt} \in \{0,1\} \qquad (20)$$

ค่า X, Y และ Z มีค่าเป็น 0 และ 1 เท่านั้น

$$C_{jt}, H_{it}, N_{jmt} \in \{0,1,...\} \qquad (21)$$

3.4 การทดสอบและผลการทดสอบแบบปัญหาทางคณิตศาสตร์

3.4.1 ค่าของพารามิเตอร์และตัวแปรต่าง ๆ ที่ต้องป้อนให้กับแบบปัญหาทางคณิตศาสตร์เพื่อคำนวณ

Input – Parameter

1. จำนวนผลิตภัณฑ์, nJobs
2. จำนวนรอบการทำงาน, nPeriod
3. จำนวนเครื่องจักร, nMachine
4. จำนวนชั่วโมงทำงานทั้งหมด, WorkHour
5. จำนวนผลิตภัณฑ์ขั้นต่ำเมื่อผลิต, LowerLotSize

Input - Data

1. เวลาที่ใช้ในการผลิตแต่ละชิ้น, Production Time, P_{jm}
2. เวลาที่ใช้ในการรอเพื่อปล่อยงานเข้าเครื่องจักร, Release Time, R_j
3. ความต้องการผลการผลิต ในแต่ละรอบเวลาการผลิต, Demand, D_{jt}
4. ข้อจำกัดการผลิตของเครื่องจักร, Machine Eligibility, E_{jm}

มหาวิทยาลัยเทคโนโลยีสุรนารี

5. ค่าใช้จ่ายในการผลิตต่อชิ้น ในแต่ละรอบเวลาการผลิต, g_{jt}
6. ค่าใช้จ่ายในการเก็บรักษาต่อชิ้น ในแต่ละรอบเวลาการผลิต, o_{jt}
7. ค่าใช้จ่ายในการปรับตั้งเครื่องจักร กรณีงาน i ต่อไปยังงาน j, s_{ij}

Output - Data

ค่าของตัวแปรต่าง ๆ ที่ได้จากแบบปัญหาทางคณิตศาสตร์

1. ตารางการผลิต (Binary Data), X_{ijmt}
2. จำนวนงานที่ผลิตในแต่ละคาบเวลา, N_{jmt}
3. เวลาที่ใช้ในการผลิตรวม, Complete Time, C_{jt}
4. จำนวนงานที่เก็บรักษาของผลิตภัณฑ์ ในแต่ละคาบเวลา, H_{jt}

3.4.2 การทดสอบแบบปัญหากรณีที่ 1 – 6J-2M-3P

ก. ข้อมูลที่ป้อนให้แบบปัญหา

Arry_P = 0 1 2 3 4 5;

Arry_R = 0 2 5 10 10 5;

Arry_D =

0	0	0
10	20	10
10	10	30
20	25	20
40	20	20
20	10	20

;

Arry_O =

0	0	0	
2	3	4	
2	3	4	
3	4	5	
3	4	5	
3	4	5;	

Arry_G =

0	0	0
8	12	18
8	12	18
10	15	20
10	15	20
10	15	20;

Arry_E =!Machine M1 M2;

1	1
1	0
1	1
0	1
1	0
1	1;

Arry_S =

0	0	0	0	0	0
0	0	12	25	8	4
0	5	0	13	23	20
0	11	23	0	9	12
0	17	19	26	0	9
0	4	24	29	23	0;

มหาวิทยาลัยเทคโนโลยีสุรนารี

ข. ข้อมูลที่ได้จากแบบปัญหา

% เวลาในการผลิตรวมในรอบที่ 1, 2, 3

Arry_C =

0	0	0
54	10	10
25	127	15
70	179	43
342	38	38
282	30	72

% จำนวนงานที่เก็บในรอบที่ 1, 2, 3

Arry_H =

0	0	0
14	2	0
0	25	0
0	15	0
30	15	0
20	15	0

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 1

Arry_N(:,:,1) =

0	0
24	0
10	0
0	20
70	0
0	40

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 2

Arry_N(:,:,2) =

0	0
8	0
35	0
0	40
5	0
0	5

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 3

Arry_N(:,:,3) =

0	0
8	0
0	5
0	5
5	0
5	0

% ตารางผลิตของ M1 ในรอบ 1

Arry_X(:,:,1,1) =

0	0	1	0	0	0
0	0	0	0	1	0
0	1	0	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0

% ตารางผลิตของ M2 ในรอบ 1

Arry_X(:,:,2,1) =

0	0	0	1	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	0	0
1	0	0	0	0	0

% ตารางผลิตของ M1 ในรอบ 2

Arry_X(:,:,1,2) =

0	1	0	0	0	0
0	0	0	0	1	0
1	0	0	0	0	0
0	0	0	0	0	0
0	0	1	0	0	0
0	0	0	0	0	0

% ตารางผลิตของ M2 ในรอบ 2

Arry_X(:,:,2,2) =

0	0	0	0	0	1
0	0	0	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0
0	0	0	0	0	0
0	0	0	1	0	0

% ตารางผลิตของ M1 ในรอบ 3

Arry_X(:,:,1,3) =

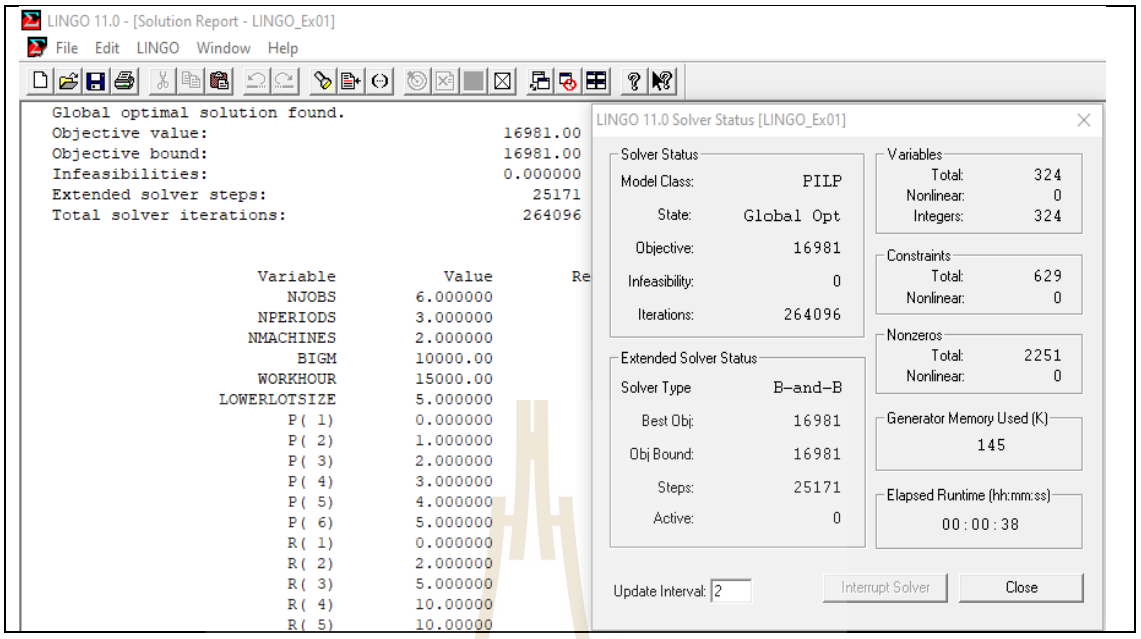
0	1	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	1
1	0	0	0	0	0

% ตารางผลิตของ M2 ในรอบ 3

Arry_X(:,:,2,3) =

0	0	1	0	0	0
0	0	0	0	0	0
0	0	0	1	0	0
1	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

มหาวิทยาลัยเทคโนโลยีสุรนารี



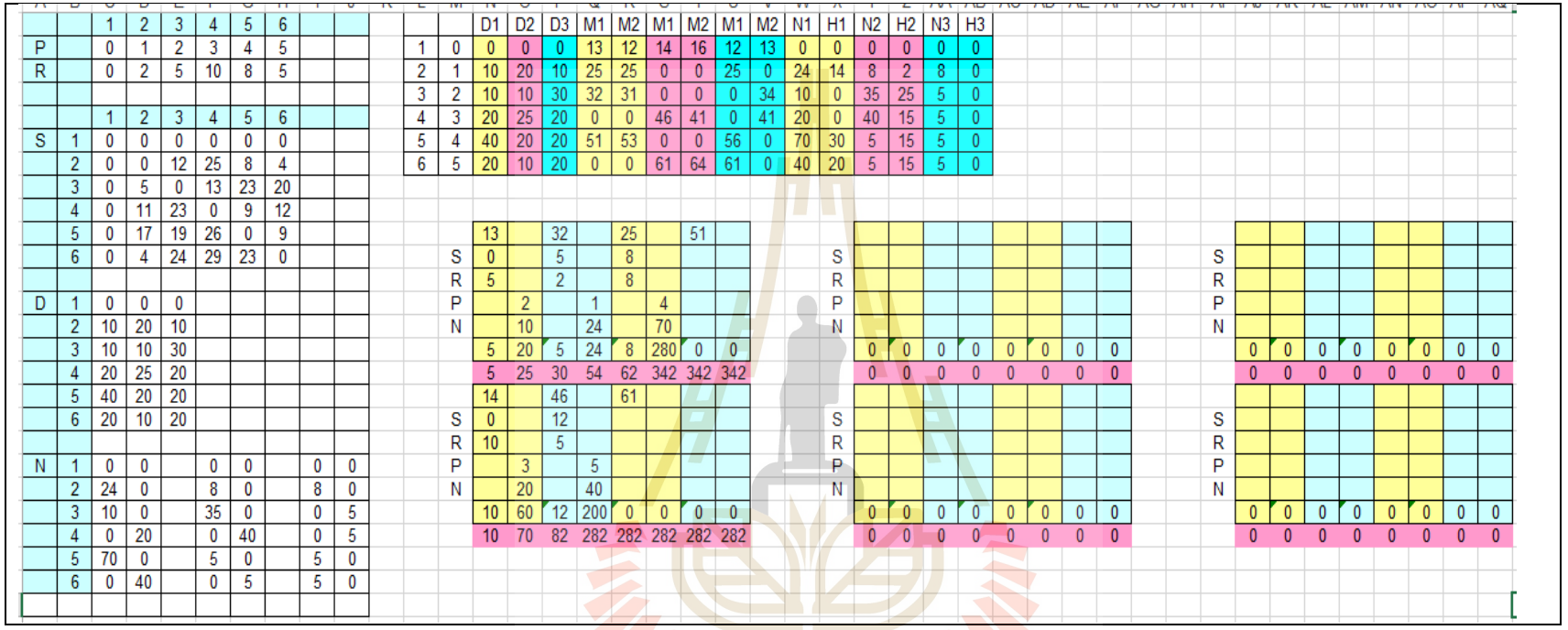
รูปที่ 3.2 ผลการทำงานด้วยโปรแกรม LINGO 11.0

ผลการคำนวณ เมื่อทำงานด้วยโปรแกรม LINGO 11.0 บนคอมพิวเตอร์ระบบปฏิบัติการ Window10-64 บิต, หน่วยประมวลผล Intel I5–6200U @2.30GHz, หน่วยความจำแรม 4 GB ใช้เวลาทำงาน 38 วินาที

ให้คำตอบเป็น Global Optimal Solution = 16981, จำนวน Solver Step = 14199 และจำนวนรอบทำงานของโปรแกรม = 305705

ค. ตารางการผลิต

มหาวิทยาลัยเทคโนโลยีสุรนารี



		1	2	3	4	5	6		
P		0	1	2	3	4	5		
R		0	2	5	10	8	5		
		1	2	3	4	5	6		
S	1	0	0	0	0	0	0		
	2	0	0	12	25	8	4		
	3	0	5	0	13	23	20		
	4	0	11	23	0	9	12		
	5	0	17	19	26	0	9		
	6	0	4	24	29	23	0		
D	1	0	0	0					
	2	10	20	10					
	3	10	10	30					
	4	20	25	20					
	5	40	20	20					
	6	20	10	20					
N	1	0	0		0	0		0	0
	2	24	0		8	0		8	0
	3	10	0		35	0		0	5
	4	0	20		0	40		0	5
	5	70	0		5	0		5	0
	6	0	40		0	5		5	0

		D1	D2	D3	M1	M2	M1	M2	M1	M2	N1	H1	N2	H2	N3	H3
1	0	0	0	0	13	12	14	16	12	13	0	0	0	0	0	0
2	1	10	20	10	25	25	0	0	25	0	24	14	8	2	8	0
3	2	10	10	30	32	31	0	0	0	34	10	0	35	25	5	0
4	3	20	25	20	0	0	46	41	0	41	20	0	40	15	5	0
5	4	40	20	20	51	53	0	0	56	0	70	30	5	15	5	0
6	5	20	10	20	0	0	61	64	61	0	40	20	5	15	5	0

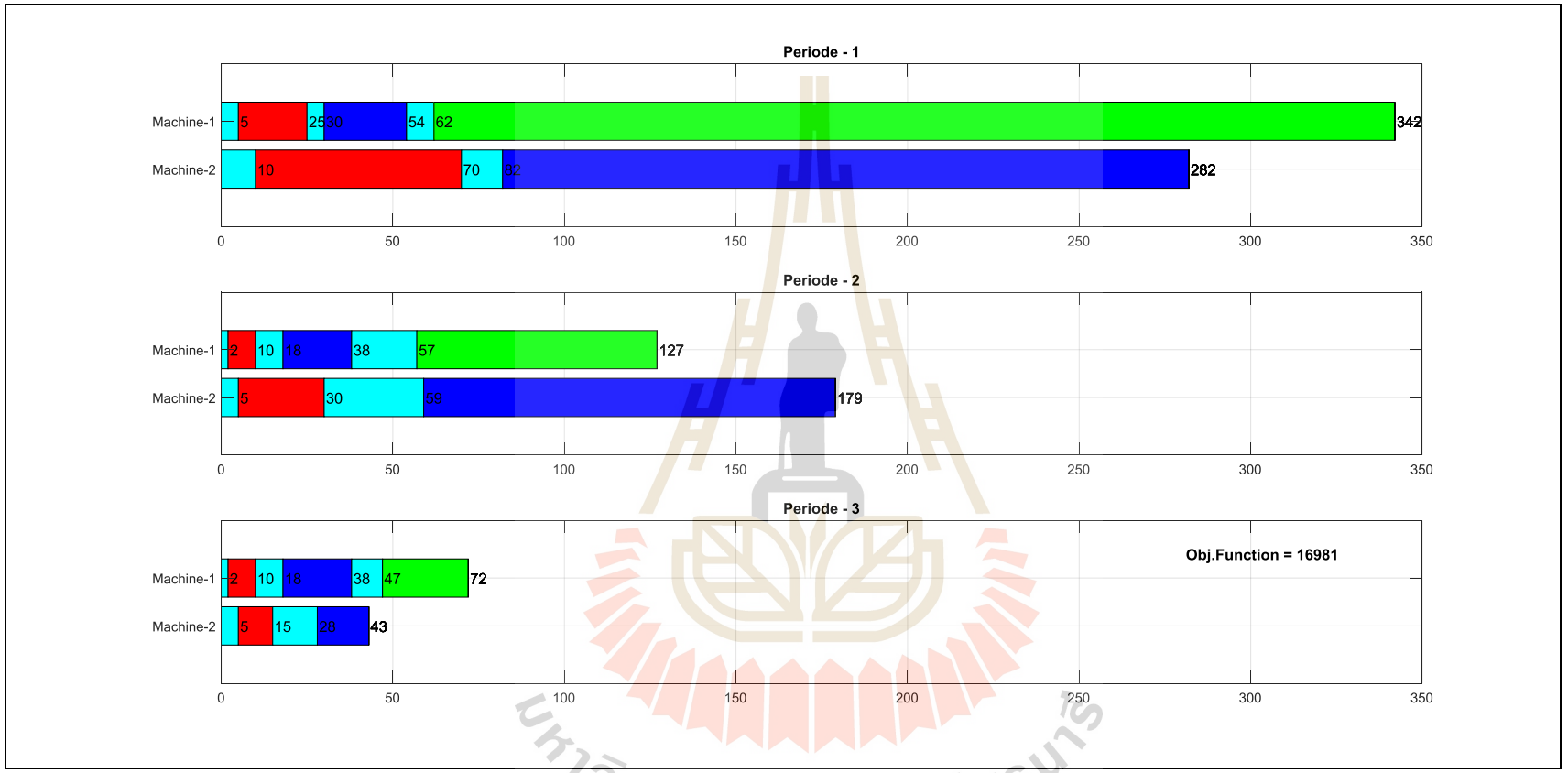
	13		32		25		51		
S	0		5		8				
R	5		2		8				
P		2		1		4			
N		10		24		70			
	5	20	5	24	8	280	0	0	
	5	25	30	54	62	342	342	342	
	14		46		61				
S	0		12						
R	10		5						
P		3		5					
N		20		40					
	10	60	12	200	0	0	0	0	
	10	70	82	282	282	282	282	282	

S									
R									
P									
N									
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
S									
R									
P									
N									
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

S									
R									
P									
N									
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
S									
R									
P									
N									
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

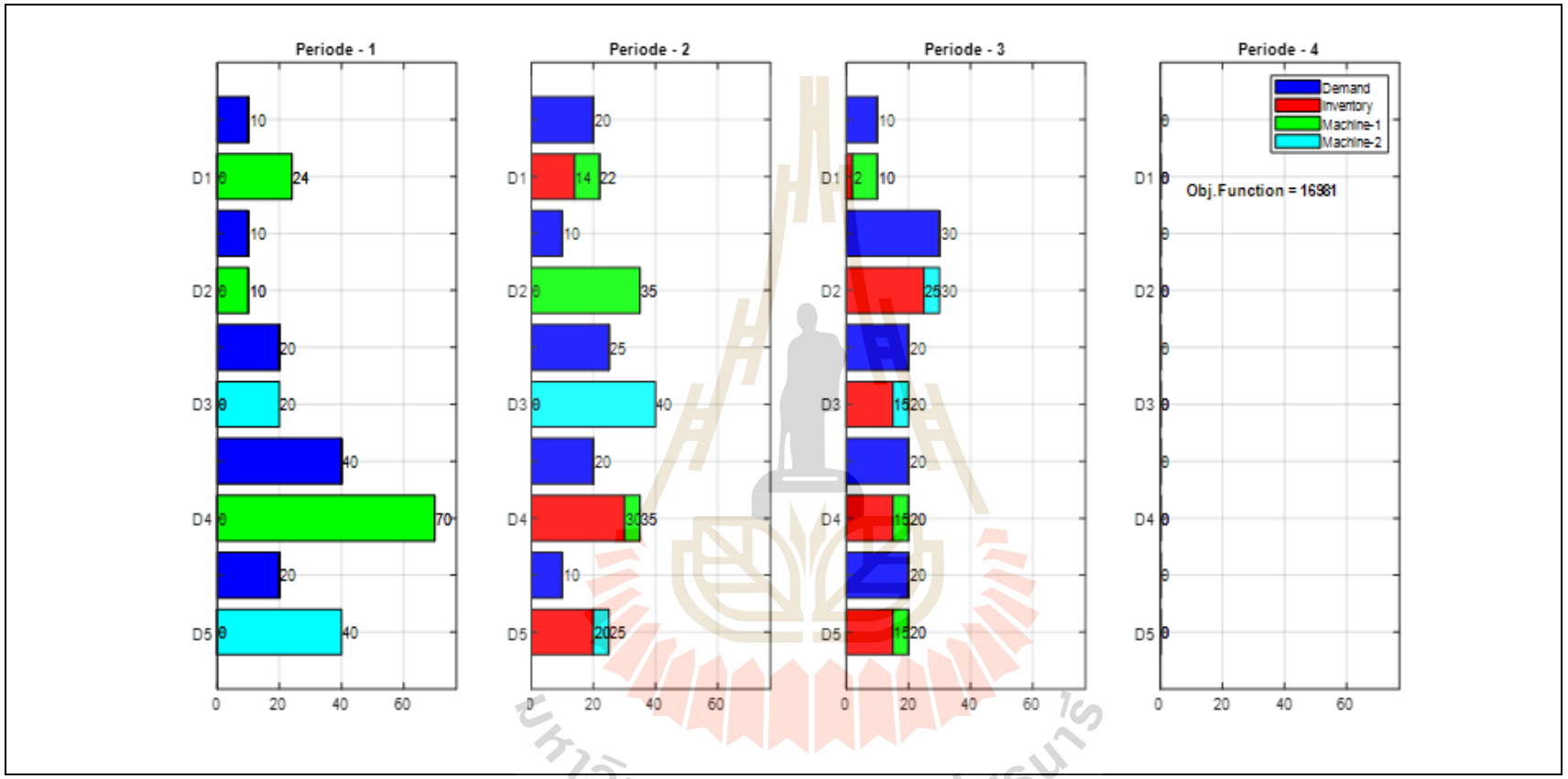
รูปที่ 3.3 การวิเคราะห์คำตอบของโปรแกรมด้วยการแจงใน Excel

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 3.4 แผนภูมิแสดงการจัดตารางงาน กรณี 06J02M03P

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 3.5 แผนภูมิแสดงจำนวนงานที่ต้องการ จำนวนคงเหลือ และจำนวนที่ผลิตแต่ละเครื่องจักรแบ่งตามคาบเวลา

มหาวิทยาลัยเทคโนโลยีสุรนารี

3.4.3 การทดสอบแบบปัญหากรณีที่ 2 – 9J-2M-3P

ก. ข้อมูลที่ป้อนให้แบบปัญหา

Arry_P = 0 1 2 3 4 5 6 7 8;

Arry_R = 0 2 5 10 10 5 0 10 0;

Arry_D =

0	0	0
10	20	10
10	10	30
20	25	20
40	20	20
20	10	20
10	20	20
30	35	30
40	30	30;

Arry_O =

0	0	0
2	3	4
2	3	4
3	4	5
3	4	5
3	4	5
2	3	4
3	4	5
2	3	4;

Arry_G =

0	0	0
8	12	18
8	12	18
10	15	20
10	15	20
10	15	20
12	18	24
12	18	24
15	20	25;

Arry_E =!Machine M1 M2;

1	1
1	0
1	1
0	1
1	0
0	1
0	1
1	0
1	0;

Arry_S =

0	0	0	0	0	00	0	0	
0	0	12	25	8	42	21	5	
0	5	0	13	23	20	24	21	21
0	11	23	0	9	12	14	4	1
0	17	19	26	0	926	29	26	
0	4	24	29	23	05	18	17	
0	24	3	13	28	70	18	12	
0	2	19	16	3	76	0	19	
0	23	12	8	24	11	17	17	0;

มหาวิทยาลัยเทคโนโลยีสุรนารี

ข. ข้อมูลที่ได้จากแบบปัญหา

% เวลาในการผลิตรวมในรอบที่ 1, 2, 3

Arry_C =

0	0	0
28	8	10
145	15	45
323	43	73
422	217	89
535	80	110
60	295	30
259	666	66
1000	53	323

% จำนวนงานที่เก็บในรอบที่ 1, 2, 3

Arry_H =

0	0	0
16	2	0
30	25	0
35	15	0
0	15	0
20	15	0
0	15	0
0	25	0
29	4	0

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 1

Arry_N(:,:,1) =

0	0
26	0
0	40
0	55
40	0
0	40
0	10
30	0
69	0

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 2

Arry_N(:,:,2) =

0	0
6	0
0	5
0	5
35	0
0	5
0	35
60	0
5	0

% จำนวนชิ้นงานที่ผลิตของ M1,M2 ในรอบ 3

Arry_N(:,:,3) =

0	0
8	0
0	5
0	5
5	0
0	5
0	5
5	0
26	0

% ตารางผลิตของ M1 ในรอบ 1

Arry_X(:,:,1,1) =

0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0

% ตารางผลิตของ M2 ในรอบ 1

Arry_X(:,:,2,1) =

0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

มหาวิทยาลัยเทคโนโลยีสุรนารี

% ตารางผลิตของ M1 ในรอบ 2

Arry_X(:,:,1,2) =

0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0

% ตารางผลิตของ M2 ในรอบ 2

Arry_X(:,:,2,2) =

0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

% ตารางผลิตของ M1 ในรอบ 3

Arry_X(:,:,1,3) =

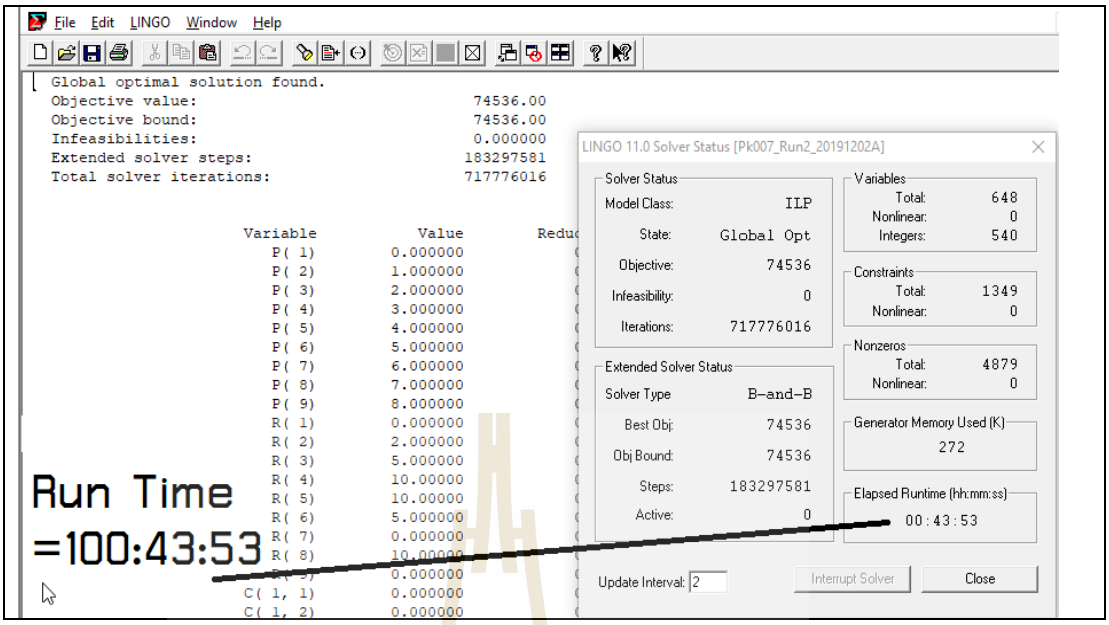
0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0

% ตารางผลิตของ M2 ในรอบ 3

Arry_X(:,:,2,3) =

0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

มหาวิทยาลัยเทคโนโลยีสุรนารี



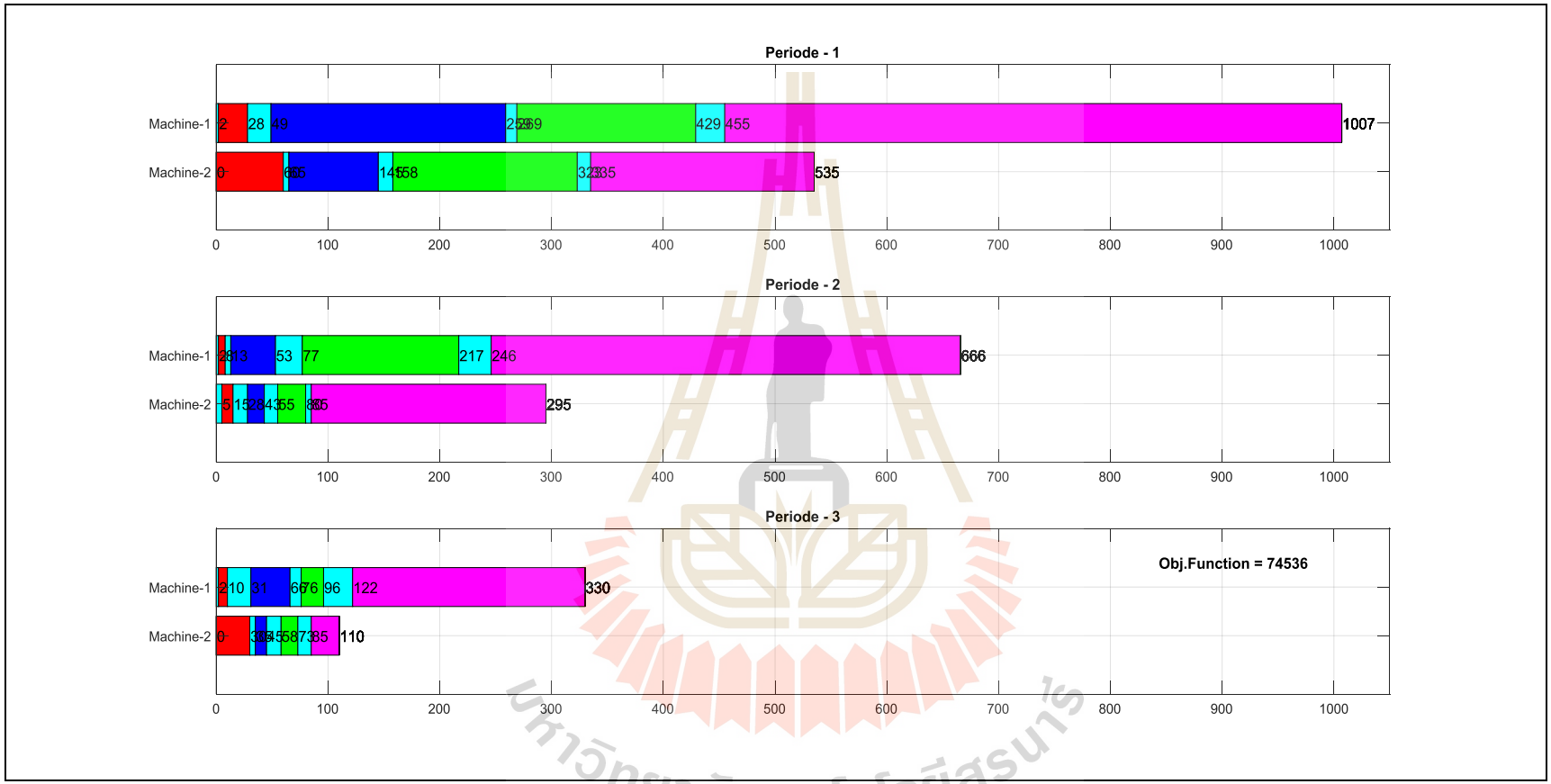
รูปที่ 3.6 ผลการทำงานด้วยโปรแกรม LINGO 11.0 ของปัญหา 9J-2M-3P

ผลการคำนวณของปัญหานี้ เมื่อทำงานด้วยโปรแกรม LINGO 11.0 บนคอมพิวเตอร์ระบบปฏิบัติการ Window10-64 บิต, หน่วยประมวลผล Intel I5-6200U @2.30GHz, หน่วยความจำแรม 4 GB ใช้เวลาทำงาน(HH:MM:SS) = 100:43:53 (4วัน-4ชม-43นาที–53วินาที)

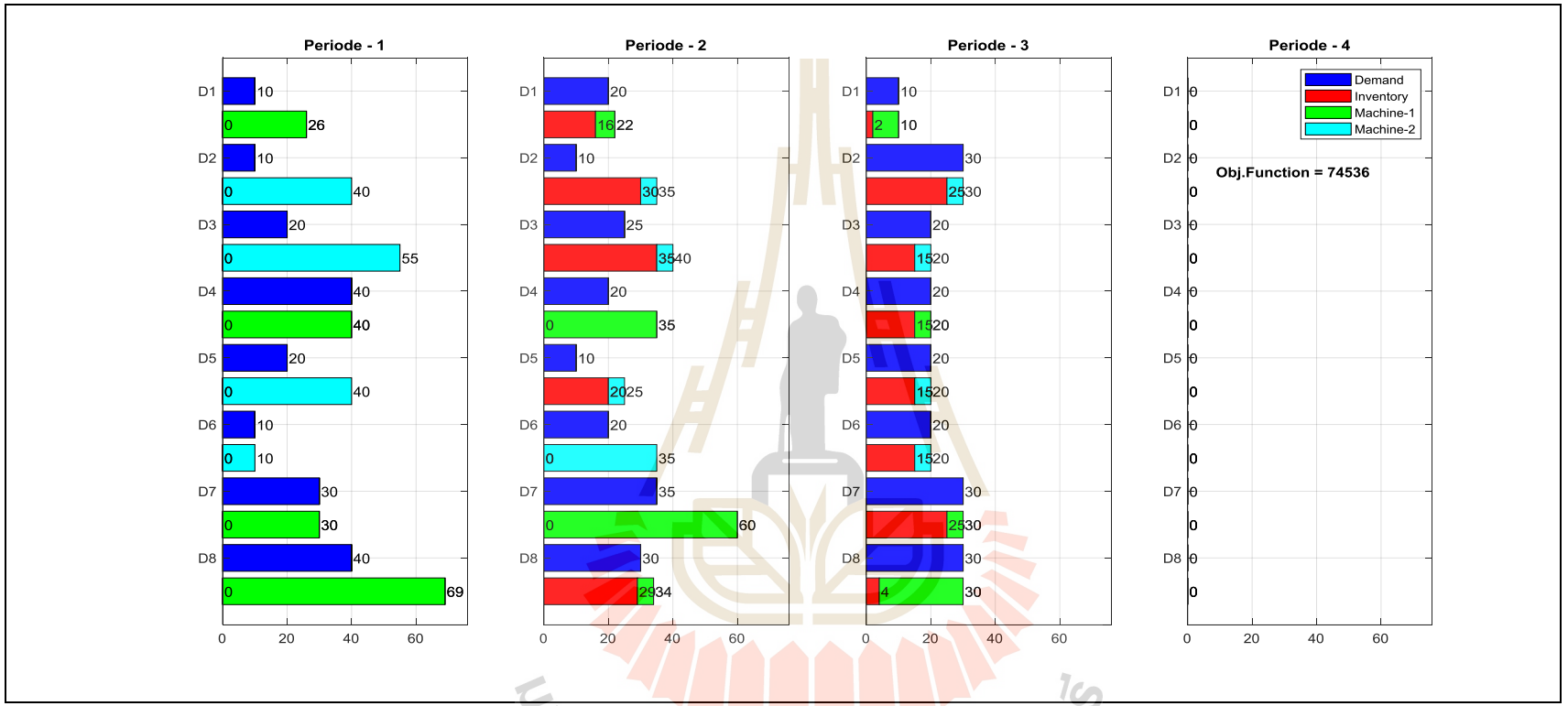
ให้คำตอบเป็น Global Optimal Solution = 74536, จำนวน Solver Step = 183,297,581 และจำนวนรอบทำงานของโปรแกรม = 717,116,016

ค. ตารางการผลิต

มหาวิทยาลัยเทคโนโลยีสุรนารี

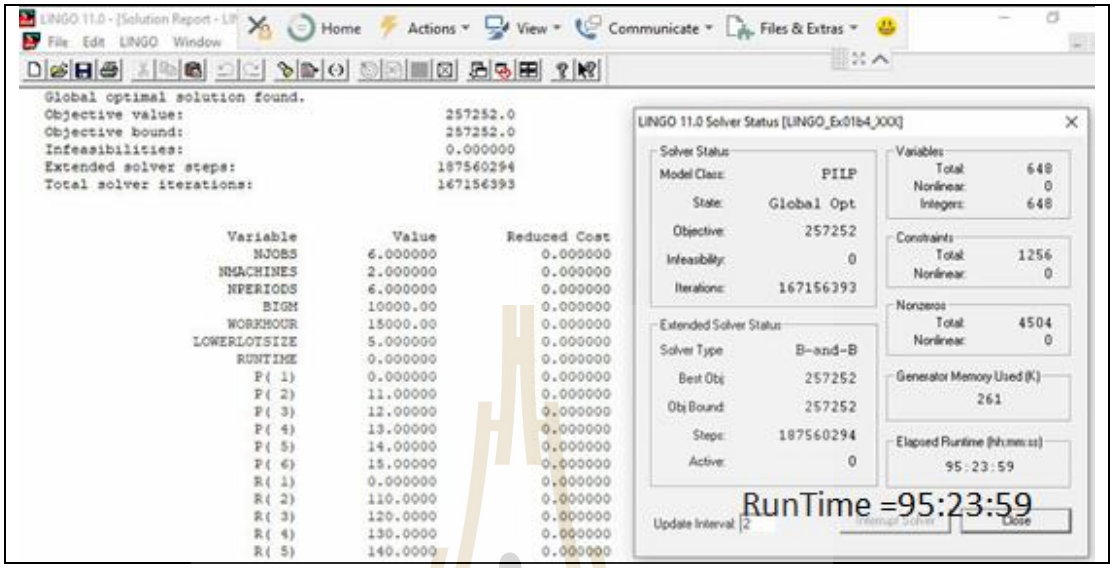


รูปที่ 3.7 แผนภูมิการจัดตารางงาน กรณี 09J02M03P



รูปที่ 3.8 แผนภูมิ จำนวนงาน (ที่ต้องการ คงเหลือ และที่ผลิตแต่ละเครื่องจักร) แบ่งตามคาบเวลา กรณี 09J02M03P

3.4.4 การทดสอบแบบปัญหากรณีที่ 3-6J-2M-6P



รูปที่ 3.9 ผลการทำงานด้วยโปรแกรม LINGO 11.0 ของปัญหา 6J-2M-6P

ผลการคำนวณของปัญหานี้ เมื่อทำงานด้วยโปรแกรม LINGO 11.0 บนคอมพิวเตอร์ระบบปฏิบัติการ Window10-64 บิต, หน่วยประมวลผล Intel I5-6200U @/2.30GHz, หน่วยความจำแรม 4 GB ใช้เวลาทำงาน(HH:MM:SS) = 96:23:59 (4วัน-0ชม-23นาที–59วินาที)

ให้คำตอบเป็น Global Optimal Solution = 257251, จำนวน Solver Step = 187,560,294 และ จำนวนรอบทำงานของโปรแกรม = 167,156,393

ตารางที่ 3.1 ผลการทดสอบแบบจำลองคณิตศาสตร์

ลำดับ	กรณีทดสอบ	ค่าฟังก์ชันวัตถุประสงค์	เวลาที่ใช้ในการค้นคำตอบ
1	6J2M3P	16981	00:00:38
2	6J2M3P	16981	00:00:31
3	9J2M3P	74536	100:43:53
4	9J2M3P	74536	101:23:12
5	6J2M6P	257251	95:23:59
6	6J2M6P	257251	94:05:57

3.5 สรุป

การสร้างแบบปัญหาทางคณิตศาสตร์เพื่อหาลำดับการผลิตที่เหมาะสมที่สุดในสายการผลิตภายใต้เงื่อนไขต่างๆ ดังนี้ (1) เครื่องจักรมีการจัดเรียงแบบขนาน, (2) ลำดับการผลิตมีผลต่อเวลาติดตั้งเครื่องจักร (3) พิจารณาขนาดผลิต และ (4) มีข้อจำกัดของเครื่องจักรที่งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ โดยวัตถุประสงค์เพื่อทำให้ค่าใช้จ่าย และเวลาในการผลิตรวมของระบบมีค่าต่ำที่สุด แบบปัญหาทางคณิตศาสตร์นี้จะถูกใช้อ้างอิงกับวิธีอื่น แต่ทุกวิธีของแบบปัญหาจะมีข้อมูล แบ่งเป็น

ข้อมูล อินพุต 7 ตัว ได้แก่

(1) เวลาที่ใช้ในการผลิตแต่ละชิ้น P_{jm}

(2) เวลาที่ใช้ในการรอเพื่อปล่อยงานเข้าเครื่องจักร R_j

(3) ความต้องการผลการผลิต ในแต่ละรอบเวลาการผลิต, Demand, D_{jt}

(4) ข้อจำกัดการผลิตของเครื่องจักร, Machine Eligibility, E_{jm}

(5) ค่าใช้จ่ายในการผลิตต่อชิ้น ในแต่ละรอบเวลาการผลิต, g_{jt}

(6) ค่าใช้จ่ายในการเก็บรักษาต่อชิ้น ในแต่ละรอบเวลาการผลิต, o_{jt}

(7) ค่าใช้จ่ายในการปรับตั้งเครื่องจักร กรณีงาน i ต่อไปยังงาน j, s_{ij}

ข้อมูลเอาต์พุต 4 ตัว ได้แก่

(1) ตารางการผลิต (Binary Data), X_{ijmt}

(2) จำนวนงานที่ผลิตในแต่ละคาบเวลา, N_{jmt}

(3) เวลาที่ใช้ในการผลิตรวม, Complete Time, C_{jt}

(4) จำนวนงานที่เก็บรักษาของผลิตภัณฑผลิตในแต่ละคาบเวลา, H_{jt}

เมื่อให้โปรแกรมทำงาน และนำผลของปัญหาขนาดเล็กมาวิเคราะห์ใน Excel พบว่าคำตอบของแบบปัญหาทางคณิตศาสตร์ทำงานได้ถูกต้อง จากนั้นได้สร้างแผนภูมิแสดงจำนวนชิ้นงานที่ต้องการ จำนวนชิ้นงานคงเหลือ และจำนวนชิ้นงานที่ผลิตแต่ละเครื่องจักรแบ่งตามคาบเวลา และแผนภูมิแสดงลำดับงานของเครื่องจักรแบ่งตามคาบเวลา

ในบทนี้มีกรณีทดสอบเพียงสามแบบปัญหาเท่านั้น ปัญหาแรกเป็นปัญหาขนาดเล็กจำนวนงานจริง 5 งาน ใช้เวลาคำนวณประมาณ 1 นาที, ปัญหาที่สองเป็นปัญหาขนาดกลาง โดยเพิ่มจำนวนงานจริงเป็น 8 งาน ใช้เวลาคำนวณประมาณ 4 วัน 4 ชั่วโมง, ปัญหาที่สามเป็นปัญหาขนาดกลางโดยเพิ่มจำนวนรอบการทำงานเป็น 6 รอบ ใช้เวลาคำนวณประมาณ 4 วัน 1 ชั่วโมง ซึ่งปัญหาที่สอง และสามใช้เวลาค่อนข้างนานจึงจำเป็นต้องหาแบบปัญหาวิธีอื่นมาแทน จำนวนกรณีทดสอบในบทถัดไปจะมีมากขึ้นเพื่อเทียบกับแบบปัญหาที่สร้างจากปัญญาเชิงคำนวณแบบต่าง ๆ

บทที่ 4
การจัดตารางการผลิต โดยวิธีปัญญาเชิงคำนวณ

4.1 บทนำ

การแก้ปัญหาเพื่อหาผลเฉลยของแบบจำลองการจัดตารางการผลิต สามารถแก้ปัญหาได้ทั้ง 2 รูปแบบ คือ วิธีการหาผลเฉลยที่ดีที่สุด และวิธีฮิวริสติค โดยปกติแบบจำลองการจัดตารางการผลิตจะมีขนาดใหญ่ และมีตัวแปรในการตัดสินใจจำนวนมาก ทำให้การใช้วิธีการหาผลเฉลยที่ดีที่สุดใช้เวลาในการคำนวณมาก อย่างไรก็ดี ได้มีการพัฒนาวิธีฮิวริสติค ให้สามารถค้นหาคำตอบที่รวดเร็วขึ้น และสามารถให้ค่าคำตอบใกล้เคียงกับผลเฉลยที่ดีที่สุด

เนื้อหาส่วนนี้กล่าวถึงวิธีฮิวริสติค ที่ใช้ปัญญาเชิงคำนวณเพื่อจัดตารางการผลิต ประกอบด้วย ขั้นตอนการคำนวณฟังก์ชันวัตถุประสงค์, การแบ่งขนาดของปัญหา, การกำหนดข้อมูลทดสอบตามขนาดของปัญหา กรณีปัญหาขนาดเล็กจะทำการเปรียบเทียบ 3 วิธี คือ (1) แบบปัญหาทางคณิตศาสตร์, (2) การค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว (ANS: Adaptive Nearest Neighbor Search) และ (3) การค้นหาแบบตาบูประยุกต์ (MTS: Modify Tabu Search) ส่วนกรณีปัญหาขนาดใหญ่จะทำการเปรียบเทียบเพียง 2 วิธี โดยไม่มีแบบปัญหาทางคณิตศาสตร์

4.2 การคำนวณค่าฟังก์ชันวัตถุประสงค์

การคำนวณค่าฟังก์ชันวัตถุประสงค์เป็นส่วนหลักของทุกวิธีการคำนวณ ส่วนนี้เป็นการอธิบายขั้นตอนการคำนวณ โดยยกตัวอย่างกรณี 05J02M06P ซึ่งเป็นปัญหาขนาดเล็กเพื่อให้เห็นขั้นตอนการทำงานของโปรแกรม

4.2.1 การสร้างรูปแบบการทำงานทั้งหมด

- จากจำนวนงาน = 5 → 1 Dummy Job + 4 Real Jobs
- งาน 1 เป็น Dummy Jobs นั่นคือ ทุกตารางงานต้องเริ่มจากงาน 1 และจบที่งาน 1
- จำนวนงานจริง คือ 4 สามารถจัดรูปแบบงานได้ 4! = 4x3x2x1 = 24 แบบ
- สร้างข้อมูล testAllSeq ได้ดังนี้

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
J1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
J2	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4	5	5	5	5	5	5
J3	3	3	4	4	5	5	2	2	4	4	5	5	2	2	3	3	5	5	2	2	3	3	4	4
J4	4	5	3	5	3	4	4	5	2	5	2	4	3	5	2	5	2	3	3	4	2	4	2	3
J5	5	4	5	3	4	3	5	4	5	2	4	2	5	3	5	2	3	2	4	3	4	2	3	2

- หากเลือก Index 10 และมีลำดับงานเป็น 1→3→4→5→2→1
- กรณี Index 10 ลำดับงานอาจเปลี่ยนหากจำนวนที่ผลิตเป็น 0 เช่น งาน 4 จำนวนที่ผลิตเป็น 0 ทำให้ลำดับงานเปลี่ยนเป็น 1→3→5→2→1

4.2.2 คำนวณหาจำนวนอ้างอิงในการผลิต

- ทฤษฎี EOQ หรือ economic order quantity หมายถึง ปริมาณการสั่งซื้อที่ประหยัด โดยการสั่งซื้อสินค้าในแต่ละครั้งจะสั่งในปริมาณหรือจำนวนที่ทำให้ค่าใช้จ่ายรวมต่ำที่สุด ซึ่งค่าใช้จ่ายรวมนั้นเกิดจากค่าใช้จ่ายในการสั่งซื้อ (Ordering Cost) และค่าใช้จ่ายในการจัดเก็บสินค้า (Carrying Cost) (ค่าใช้จ่ายสองตัวนี้จะแปรผกผันกัน)
- สูตร EOQ = sqrt (2 x D x O/C)
  - D = ความต้องการสินค้าใน 1 ปี
  - = ค่าใช้จ่ายในการสั่งซื้อแต่ละครั้ง
  - C = ค่าใช้จ่ายในการจัดเก็บสินค้าต่อหน่วยต่อปี

  * sqrt -> ฟังก์ชันรากที่สอง
- ปรับสูตร EOQ เป็น

  ceil (sqrt(2x sum(Arry.DA)xsum(Arry.SA)/sum(Arry.OA)))
  - Arry.DA(Average) = ความต้องการสินค้าเฉลี่ยทุกรอบผลิต
  - Arry.SA(Average) = ค่าใช้จ่ายในติดตั้งเครื่องจักรเฉลี่ยทุกลำดับงาน
  - Arry.OA(Average) = ค่าใช้จ่ายในการจัดเก็บสินค้าเฉลี่ยทุกรอบผลิต

  * ceil -> ฟังก์ชันจะมีการปัดเศษขึ้นเสมอ
- ผลการคำนวณ

มหาวิทยาลัยเทคโนโลยีสุรนารี

Arry.D	Arry.G	∑G	Arry.O	∑O	EOQ
0 0 0 0 0 0	0 0 0 0 0 0	0	0 0 0 0 0 0	0	0
15 31 31 13 22 10	16 28 12 36 23 30	145	12 50 25 17 40 20	164	15
30 14 11 26 22 38	47 20 41 27 43 36	214	45 19 15 39 33 47	198	18
37 39 35 20 15 30	42 40 37 26 42 48	235	35 36 11 25 17 19	143	25
26 26 33 26 17 38	33 19 39 43 44 28	206	24 34 27 44 49 25	203	19

4.2.3 คำนวณหา Quantity Array

- ฟังก์ชั่นนี้มีหน้าที่คำนวณหา Quantity Array หรือจำนวนที่ผลิตในแต่ละรอบการผลิต แต่ละงาน แต่ละเครื่องจักร
- อินพุท: Arry.D, Arry.E, Arry.P, EOQ, LowerLotsize
- เอาต์พุทขั้นต้น: จำนวนชิ้นงานที่ผลิตแต่ละผลิตภันฑ์ ในแต่ละรอบทำงาน
- เอาต์พุทขั้นสุดท้าย :จำนวนชิ้นงานที่ผลิตแต่ละผลิตภันฑ์ ในแต่ละรอบทำงานของแต่ละเครื่องจักร
- การทำงานเพื่อหาเอาต์พุทขั้นต้น: จำนวนชิ้นงานที่ผลิตแต่ละผลิตภันฑ์ ในแต่ละรอบทำงาน
  - ค่า EOQ เป็นเพียงตัวเลขจำนวนอ้างอิงในการผลิตเท่านั้น
  - จำนวนการสั่งผลิตต้อง เป็นไปตามเงื่อนไขดังนี้

    (1) เมื่อรวมกับจำนวนที่เก็บในรอบการผลิตก่อนนี้ต้องไม่ต่ำกว่าความต้องการในการส่งมอบ

    (2) จำนวนผลิตต้องมากกว่าค่า EOQ

    (3) จำนวนผลิตต้องมากกว่าค่า LowerLotSize
  - การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

Hold(j) = 0	กำหนดค่าเริ่มต้นจำนวนสินค้าค้างส่ง = 0
All_D(j) = ∑ D(j,p) All p	คำนวณจำนวนที่ต้องการทั้งหมด รวมทุกรอบการผลิต
For p=1 : nPeriod	วนรอบทุกรอบการผลิต
For j=1 : nJobs	วนรอบทุกชนิดสินค้าที่ผลิต
tmpD = D(j,p) – Hold(j)	จำนวนผลิต = จำนวนที่ต้องการ – ที่ผลิตไว้ก่อนนี้
tmpD = max(EOQ(j) , tmpD)	จำนวนผลิต = ต้องผลิตมากกว่า EOQ
tmpD = max(LowerLotSize, tmpD)	จำนวนผลิต = ต้องผลิตมากกว่า LowerLotSize
	จำนวนผลิต = ต้องผลิตไม่เกินความต้องการ

Hold(j) = 0	กำหนดค่าเริ่มต้นจำนวนสินค้าค้างส่ง = 0
All_D(j) = $\sum$ D(j,p) All p	คำนวณจำนวนที่ต้องการทั้งหมด รวมทุกรอบการผลิต
For p=1 : nPeriod	วนรอบทุกรอบการผลิต
For j=1 : nJobs	วนรอบทุกชนิดสินค้าที่ผลิต
tmpD = D(j,p) – Hold(j)	จำนวนผลิต = จำนวนที่ต้องการ – ที่ผลิตไว้ก่อนนี้
tmpD = max(EOQ(j) , tmpD)	จำนวนผลิต = ต้องผลิตมากกว่า EOQ
tmpD = max(LowerLotSize, tmpD)	จำนวนผลิต = ต้องผลิตมากกว่า LowerLotSize
tmpD = min(All_D(j),tmpD)	จำนวนผลิต = ต้องผลิตไม่เกินความต้องการทั้งหมด
tempX(j,p) = tmpD	บันทึกจำนวนที่ต้องผลิต
All_D(j) = All_D(j) – tmpD	ปรับปรุงจำนวนที่ต้องการทั้งหมด
Hold(j) = tmpD - D(j,p)	ปรับปรุงจำนวนสินค้าค้างส่ง

EOQ	LowerLotSize	Arry.D						$\sum$ D	Arry.tempX						$\sum$ tempX
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	1	15	31	31	13	22	10	122	15	31	31	15	20	10	122
18	1	30	14	11	26	22	38	141	30	18	18	18	19	38	141
25	1	37	39	35	20	15	30	176	37	39	35	25	25	15	176
19	1	26	26	33	26	17	38	166	26	26	33	26	19	36	166

- การทำงานเพื่อหาเอาต์พุทขั้นสุดท้าย :จำนวนชิ้นงานที่ผลิต แต่ละผลิตภัณฑ์ ใน แต่ละรอบทำงาน ของแต่ละเครื่องจักร
  - ขั้นตอนนี้เป็นการแบ่งจำนวนผลิตให้กับเครื่องจักรแต่ละเครื่อง
  - หากงานที่ผลิตสามารถใช้เครื่องจักรผลิตได้มากกว่า 1 เครื่อง จะมอบหมายงานให้กับเครื่องจักรที่มีเวลาผลิตรวม ณ ปัจจุบัน น้อยสุด
  - การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

tmp1Qtt(j) = Arry.E(j) * Arry.tempX(j,1)	กำหนดงานให้ทุกงานเครื่องจักรที่ผลิตได้ในรอบการผลิตที่ 1
For p=2:nPeriod	วนรอบทุกรอบการผลิตเริ่มตั้งแต่รอบที่ 2
tmp1Qtt(j) = Arry.E(j) * Arry.tempX(j,p)	กำหนดงานทุกงานให้ทุกเครื่องจักรที่ผลิตได้ในรอบการผลิตที่ p
tmpQtt(j,:) = horzcat(TmpQtt(j,:), tmp1Qtt(j))	แทรกงานเข้าในตารางแจกแจงงาน
tRMachine(j,m) = 0	กำหนดเวลาทำงานรวมของแต่ละเครื่องจักร = 0
For p=1:nPeriod	วนรอบทุกรอบการผลิตเริ่มตั้งแต่รอบที่ 1
For j=1:nJobs	วนรอบทุกชิ้นงานตั้งแต่งานที่ 1
tTMachine(j,x) = tmpQtt(j,x)	เลือกชุดข้อมูลทดสอบของรอบผลิตและชิ้นงาน
if(sum(Arry.E(j)>1)	ตรวจสอบหากเครื่องจักรทำงานได้มากกว่าหนึ่งเครื่อง
mask tTMachine(j,x)	กำหนดให้เครื่องจักรที่ชั่วโมงทำงานน้อยสุดรับงาน
tTMachine(j,x) *= ArrayP(j)	คำนวณชั่วโมงการทำงานของเครื่องจักร
tRMachine (j,m) += tTMachine(j,x)	ปรับปรุงชั่วโมงการทำงานของแต่ละเครื่องจักรรวม

Arry.tempE		Arry.tempX						testQty											
E1	E2	P1	P2	P3	P4	P5	P6	P1(M1	M2)	P2(M1	M2)	P3(M1	M2)	P4(M1	M2)	P5(M1	M2)	P6(M1	M2)
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	15	31	31	15	20	10	15	0	31	0	31	0	15	0	20	0	10	0
0	1	30	18	18	18	19	38	0	30	0	18	0	18	0	18	0	19	0	38
1	0	37	39	35	25	25	15	37	0	39	0	35	0	25	0	25	0	15	0
1	1	26	26	33	26	19	36	0	26	0	26	0	33	0	26	0	19	0	36

4.2.4 คำนวณหา Sequence Array

- ฟังก์ชั่นนี้มีหน้าที่คำนวณหาลำดับการผลิตในแต่ละรอบการผลิต แต่ละงานแต่ละเครื่องจักร โดยจะตัดลำดับงานที่จำนวนผลิตเป็นศูนย์ออกแล้วขยับลำดับขึ้นมา
- อินพุท: testAllSeq, testIndex, testQtt
- เอาท์พุต: testSeq
- จาก testAllSeq, testIndex, testQtt

มหาวิทยาลัยเทคโนโลยีสุรนารี

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
J1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
J2	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4	5	5	5	5	5	5
J3	3	3	4	4	5	5	2	2	4	4	5	5	2	2	3	3	5	5	2	2	3	3	4	4
J4	4	5	3	5	3	4	4	5	2	5	2	4	3	5	2	5	2	3	3	4	2	4	2	3
J5	5	4	5	3	4	3	5	4	5	2	4	2	5	3	5	2	3	2	4	3	4	2	3	2

- งาน P1M1 เลือก Index 15 ซึ่งมีลำดับงานเป็น 1→4→3→2→5
- งาน P1M1 มีการผลิตเฉพาะงาน 2 กับ 4 ทำให้ลำดับงานเปลี่ยนเป็น 1→4→2
- การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

for kLoop = 1:nLoop	วนรอบทุกรอบการผลิต
kIndexSave = 2;	เริ่มบันทึกที่งาน 2
for jJobs = 2:nJobs	วนรอบทุกงานเริ่มจากงาน 2
indexJobTest = testAllSeq(jJobs,testIndex(kLoop));	อ่านค่างานที่ต้องการทดสอบ
if testQtt(indexJobTest,kLoop)~= 0	หากจำนวนที่ผลิตของงานที่ต้องการทดสอบไม่เป็นศูนย์
tempSeqOut(kIndexSave) = indexJobTest;	บันทึกงานในตารางงาน
kIndexSave = kIndexSave + 1;	เพิ่มตัวชี้ตารางงาน

testIndex											
P1(M1 M2)	P2(M1 M2)		P3(M1 M2)		P4(M1 M2)		P5(M1 M2)		P6(M1 M2)		
15 3	3	7	16	11	4	17	5	16	21	13	
testQty											
P1(M1 M2)	P2(M1 M2)		P3(M1 M2)		P4(M1 M2)		P5(M1 M2)		P6(M1 M2)		
0	0	0	0	0	0	0	0	0	0	0	0
15	0	31	0	31	0	15	0	20	0	10	0
0	30	0	18	0	18	0	18	0	19	0	38
37	0	39	0	35	0	25	0	25	0	15	0
0	26	0	26	0	33	0	26	0	19	0	36
testSeq											
P1(M1 M2)	P2(M1 M2)		P3(M1 M2)		P4(M1 M2)		P5(M1 M2)		P6(M1 M2)		
1	1	1	1	1	1	1	1	1	1	1	1
4	3	2	3	4	3	2	5	2	3	2	3
2	5	4	5	2	5	4	3	4	5	4	5
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

มหาวิทยาลัยเทคโนโลยีสุรนารี

4.2.5 คำนวณหา Objective X, Y, Z

- ฟังก์ชั่นนี้มีหน้าที่คำนวณหาค่าฟังก์ชันวัตถุประสงค์ X, Y, Y โดย Objective_X มาจากจำนวนที่ผลิต, Objective_Y มาจากจำนวนเก็บ และ Objective_Z มาจากการติดตั้งเครื่องจักร
- อินพุท: testSeq, testQtt, FactData
- เอาท์พุต: ObjX, ObjY, ObjZ
- จาก testSeq, testQtt

testQty											
P1(M1 M2)		P2(M1 M2)		P3(M1 M2)		P4(M1 M2)		P5(M1 M2)		P6(M1 M2)	
0	0	0	0	0	0	0	0	0	0	0	0
15	0	31	0	31	0	15	0	20	0	10	0
0	30	0	18	0	18	0	18	0	19	0	38
37	0	39	0	35	0	25	0	25	0	15	0
0	26	0	26	0	33	0	26	0	19	0	36
testSeq											
P1(M1 M2)		P2(M1 M2)		P3(M1 M2)		P4(M1 M2)		P5(M1 M2)		P6(M1 M2)	
1	1	1	1	1	1	1	1	1	1	1	1
4	3	2	3	4	3	2	5	2	3	2	3
2	5	4	5	2	5	4	3	4	5	4	5
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

4.2.5.1 <u>คำนวณหา Objective X</u>

- Objective_X = Arry.G * Pcs → โดย Pcs คือ จำนวนที่ผลิตแต่ละรอบการผลิต
- ข้อมูลที่ต้องใช้จาก Factory Data คือ
  - ค่าใช้จ่ายในการผลิต - Arry.G
  - เวลาในการผลิต - Arry.P
  - เวลาในการปล่อยงาน - Arry.R
  - เวลาในการติดตั้งเครื่องจักร - Arry.S
- การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

tmpArryC = zeros(nJobs,nPeriod);
for kPrd=1:nPeriod
for jMach=1:nMachine
xFrom = 1;
CTotal = 0;
for iJobs=2:nJobs
xNext = testSeq(iJobs,kReadIndex);
if ~((xNext == 0)||(xNext == 1))
CRelease = max(CTotal,
FactData.ArrayR(xNext));
CSetup =
FactData.ArrayS(xFrom,xNext);
CRunTime =
testQtt(xNext,kReadIndex)*FactData.ArrayP(xNext);
CTotal = CRelease + CSetup +
CRunTime;
tmpArryC(xNext,kPrd) = CTotal;
ObjX = sum(sum(FactData.ArrayG.*tmpArryC));
	สร้างตัวแปร การผลิตแต่ละรอบการผลิต
	วนรอบ ทุกรอบการผลิต
	วนรอบทุกเครื่องจักรที่ทำการผลิต
	เริ่มต้นที่เครื่องจักรหมายเลข 1
	ให้จำนวนชิ้นงานเป็น 0
	เริ่มคำนวณจากงาน 2 จนครบทุกงาน
	อ่านลำดับงานถัดไป
	หากงานยังไม่จบ
	ตรวจสอบค่าเวลาปล่อยงาน
	อ่านค่าเวลาติดตั้ง
	คำนวณเวลาในการผลิต
	รายการนี้
	คำนวณเวลาในการผลิตรวม
	บันทึกค่าเวลาผลิต
	คำนวณค่า Objective_X จาก FactData.ArrayG *
	เวลาในการผลิต

4.2.5.2 <u>คำนวณหา Objective Y</u>

- Objective_Y = Arry.O * Hold → โดย Hold คือ จำนวนที่เก็บรักษาแต่ละรอบการผลิต
- ข้อมูลที่ต้องใช้จาก Factory Data คือ Arry.D
- การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

tHold = zeros(nJobs,nPeriod);
tHBf0 = zeros(nJobs,1);
tPrd1 = zeros(nJobs,1);
For p = 1:nPeriod
indexF = (iPeriod-1) *nMachine + 1;
indexT = indexF + nMachine - 1;
tPrd1 = sum(testQtt(:,indexF:indexT),2);
tHBf0 = tPrd1+tHBf0 -
	สร้างตัวแปร จำนวนสินค้าเก็บรักษา
	สร้างตัวแปร จำนวนสินค้าเก็บรักษาก่อนรอบผลิตนี้
	สร้างตัวแปร จำนวนที่ผลิตในรอบนี้
	วนรอบการทำงานทุกรอบการผลิต
	คำนวณ คอลัมน์เริ่ม ของทุกเครื่องจักรที่ทำงาน
	คำนวณ คอลัมน์สุดท้าย ของทุกเครื่องจักรที่ทำงาน
	คำนวณ จำนวนผลิตในรอบการผลิต
	คำนวณ จำนวนที่เก็บ จากจำนวนผลิตบวกจำนวนเก็บ

FactData.ArrayD(:,iPeriod); tHold(:,iPeriod) = tHBf0; ObjY = sum(sum(FactData.ArrayO.*tHold))	ก่อนนี้ลบจำนวนส่งมอบ บันทึกจำนวนสินค้าเก็บรักษา คำนวณค่า Objective_Y จาก FactData.ArrayO * จำนวนสินค้าเก็บรักษา

testQty

P1(M1 M2)	P2(M1 M2)	P3(M1 M2)	P4(M1 M2)	P5(M1 M2)	P6(M1 M2)
0 0	0 0	0 0	0 0	0 0	0 0
15 0	31 0	31 0	15 0	20 0	10 0
0 30	0 18	0 18	0 18	0 19	0 38
37 0	39 0	35 0	25 0	25 0	15 0
0 26	0 26	0 33	0 26	0 19	0 36

Arry.D

P1	P2	P3	P4	P5	P6
0	0	0	0	0	0
15	31	31	13	22	10
30	14	11	26	22	38
37	39	35	20	15	30
26	26	33	26	17	38

Arry.Hold

P1	P2	P3	P4	P5	P6
0	0	0	0	0	0
0	0	0	2	0	0
0	4	11	3	0	0
0	0	0	5	15	0
0	0	0	0	2	0

4.2.5.3 <u>คำนวณหา Objective Z</u>

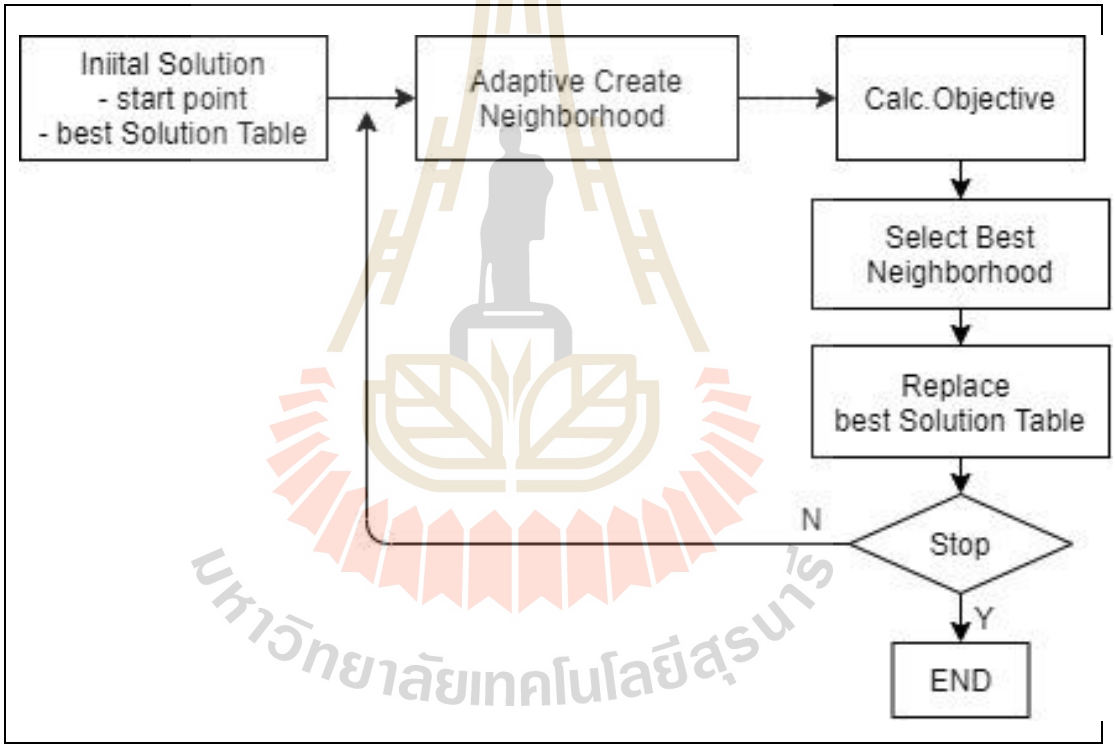
- Objective_S = Arry.S * Seq → โดย Seq คือ ลำดับการผลิตละรอบการผลิต แต่ละเครื่องจักร
- ข้อมูลที่ต้องใช้จาก Factory Data คือ Arry.S
- การทำงานตามคำสั่งเทียม และผลการทำงานตามตาราง

tX = zeros(nJobs,nJobs,nMachine,nPeriod);
for iPeriod=1:nPeriod
for iMachine=1:nMachine
iFrom = testSeq(1,kReadIndex);
for kLoop=2:nJobs
iNext = testSeq(kLoop,kReadIndex);
if(iNext~=0)
	สร้างตัวแปร ลำดับในการผลิตแต่ละเครื่องจักร แต่ละรอบงาน
	วนรอบการทำงานทุกรอบการผลิต
	วนรอบทุกเครื่องจักรที่ทำการผลิต
	เริ่มที่งานลำดับที่ 1
	วนรอบงานตั้งแต่งานลำดับที่ 2
	อ่านหมายเลขงานถัดไป

tX(iFrom,iNext,iMachine,iPeriod) = 1; iFrom = iNext;	หากเป็นไม่ใช่งานหมายเลข 0 บันทึกงานงานนี้ให้ปฏิบัติกำหนดงานถัดไป
ObjZ = sum(sum(sum(sum(FactData.ArrayS.*tX))));	คำนวณค่า Objective_Z จาก FactData.ArrayS * ลำดับงานที่ทำ

4.3 วิธีปัญญาเชิงคำนวณ

4.3.1 วิธีการค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว (Adaptive Neighborhood Search)



รูปที่ 4.1 วิธีการค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว

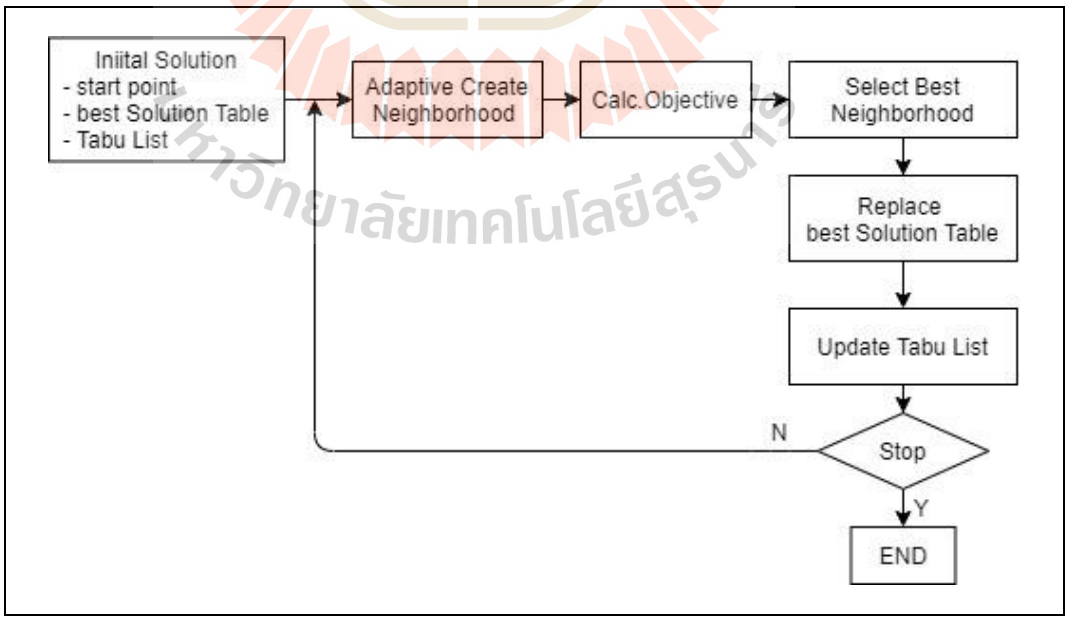
ขั้นตอนการทำงานของโปรแกรม

1. กำหนดจุดเริ่มต้นการค้นหาคำตอบ และสร้างตารางคำตอบดีเยี่ยม
2. กำหนดเพื่อนบ้านที่ใกล้เคียงแบบปรับตัวได้
   - แต่ละรอบจะมีเพื่อนบ้านจากสองตัวแปร คือ จำนวนผลิต ลำดับการผลิต

- ตรวจสอบจำนวนครั้งที่คำตอบซ้ำที่เดิม หากจำนวนเกินค่าที่กำหนด จะปรับขนาดการเลื่อนลำดับการผลิต และขนาดการผลิต
- ลำดับการผลิต$_n$ = {-4, -2, -1, 0, 1, 2, 4} .x ขนาดการเลื่อน .+ ลำดับการผลิต$_b$
- จำนวนที่ผลิต$_n$ = {-4, -2, -1, 0, 1, 2, 4} .x ขนาดการผลิต .+ จำนวนที่ผลิต$_b$

3. คำนวณฟังก์ชันวัตถุประสงค์กับเพื่อนบ้านทุกตัว
4. เลือกเพื่อนบ้านที่ให้คำตอบดีที่สุด
5. แทนค่าในตารางคำตอบดีเยี่ยม
   - ตารางคำตอบดีเยี่ยมจะเก็บคำตอบที่ดีสุดไว้จำนวนหนึ่ง
   - หากคำตอบใหม่ ดีกว่าค่าที่แย่สุดของตารางให้แทนค่า
   - จัดเรียงตารางคำตอบดีเยี่ยมใหม่
   - เลือกค่าทีดีสุดของตารางให้เป็นจุดเริ่มค้นหารอบถัดไป
6. ตรวจสอบเงื่อนไขการหยุดโปรแกรม
   - จำนวนรอบทำงานครบรอบที่กำหนด ให้หยุดโปรแกรม
   - คำตอบซ้ำกันเกินจำนวนที่กำหนด ให้หยุดโปรแกรม
   - คำตอบเข้าใกล้ค่าที่กำหนด ให้หยุดโปรแกรม

4.3.2 การค้นหาแบบตาบูประยุกต์ (Modify Tabu Search)



รูปที่ 4.2 การค้นหาแบบตาบูประยุกต์ (Modify Tabu Search)

ขั้นตอนการทำงานของโปรแกรม

1. กำหนดจุดเริ่มต้นการค้นหาคำตอบ, สร้างตารางคำตอบดีเยี่ยม, สร้างตารางตาบู
2. กำหนดเพื่อนบ้านที่ต้องการค้นหาแบบปรับตัวได้
   - แต่ละรอบจะมีเพื่อนบ้านจากสองตัวแปร คือ จำนวนผลิต ลำดับการผลิต
   - ตรวจสอบจำนวนครั้งที่คำตอบซ้ำที่เดิม หากจำนวนเกินค่าที่กำหนด จะปรับรัศมีการเลื่อนลำดับการผลิต และรัศมีขนาดการผลิต
   - ลำดับการผลิต$_n$ = {-4, -2, -1, 0, 1, 2, 4} .x รัศมีขนาดการเลื่อน .+ ลำดับการผลิต$_b$
   - จำนวนที่ผลิต$_n$ = {-4, -2, -1, 0, 1, 2, 4} .x รัศมีขนาดการผลิต .+ จำนวนที่ผลิต$_b$
3. คำนวณฟังก์ชันวัตถุประสงค์กับเพื่อนบ้านทุกตัว
4. เลือกเพื่อนบ้านที่ให้คำตอบดีที่สุด
5. แทนค่าในตารางคำตอบดีเยี่ยม
   - ตารางคำตอบดีเยี่ยมจะเก็บคำตอบที่ดีสุดไว้จำนวนหนึ่ง
   - หากคำตอบใหม่ ดีกว่าค่าที่แย่สุดของตารางให้แทนค่า
   - จัดเรียงตารางคำตอบดีเยี่ยมใหม่
   - เลือกค่าทีดีสุดของตารางให้เป็นจุดเริ่มค้นหารอบถัดไป
6. แก้ไข้ปรับปรุง ตารางตาบู
7. ตรวจสอบเงื่อนไขการหยุดโปรแกรม
   - จำนวนรอบทำงานครบรอบที่กำหนด ให้หยุดโปรแกรม
   - คำตอบซ้ำกันเกินจำนวนที่กำหนด ให้หยุดโปรแกรม
   - คำตอบเข้าใกล้ค่าที่กำหนด ให้หยุดโปรแกรม

4.4 ขนาดของปัญหาและข้อมูลที่ใช้ทดสอบ

ข้อมูลที่ใช้ในการทดสอบ สำหรับแบบจำลองต่างๆ แบ่งเป็น 3 กลุ่ม คือ (1) กลุ่มข้อมูลชุดเดี่ยวที่ใช้อ้างอิงในการพัฒนาแบบจำลอง, (2) กลุ่มข้อมูลขนาดเล็กหลายชุดเพื่อทดสอบแบบจำลอง และ (3) กลุ่มข้อมูลขนาดใหญ่หลายชุดเพื่อทดสอบแบบจำลอง

ข้อมูลที่ใช้ทดสอบประกอบด้วย

- พารามิเตอร์ ได้แก่ จำนวนงานที่ทำ-nJobs, จำนวนรอบการผลิต-nPeriod, จำนวนเครื่องจักร-nMachine, จำนวนชั่วโมงการทำงานรวม-WorkHour, ขนาดในการสั่งผลิต-LotSize
- ตัวแปร ได้แก่ เวลาที่ใช้ในการผลิตแต่ละชิ้น-Production Time, เวลาที่ใช้ในการรอเพื่อปล่อยงานเข้าเครื่องจักร-Release Time, ความต้องการผลการผลิต ในแต่ละรอบเวลาการผลิต-

Demand, ข้อจำกัดการผลิตของเครื่องจักร-Machine Eligibility, ค่าใช้จ่ายในการผลิตต่อชิ้น ในแต่ละคาบเวลาการผลิต, ค่าใช้จ่ายในการเก็บรักษาต่อชิ้น ในแต่ละรอบเวลาการผลิต, ค่าใช้จ่ายในการปรับตั้งเครื่องจักร กรณีงาน I ต่อไปยังงาน J }

อ้างอิงจาก Ref: Resource-constrained unrelated parallel machine scheduling.pdf-https://linkinghub.elsevier.com/retrieve/pii/S0360835216301619 และการทดสอบการทำงานของแบบจำลองคณิตศาสตร์ แบ่งขนาดปัญหาและขอบเขตตัวแปร ดังตาราง

ตาราง 4.1 การกำหนดค่าของตัวแปรต่าง ๆ

	Reference Model	Small Problem	Big Problem
n-Jobs	{6, 9}	{5, 6, 7, 8}	{10, 15, 20, 25, 30}
n-Machines	{2}	{2, 3}	{4, 6, 8, 10}
n-Periods	{3, 6}	{6, 9, 12}	{6, 9}
P(J) Processing Times	fix	U(10, 20)	U(20, 50)
R(J) Release Time	Fix	U(5, 30)	U(1, 200)
S(I,J) Setup Times	Fix	U(5, 20)	U(4, 10)
D(J,P) Demand	Fix	U(10, 40)	U(10, 200)
E(J,M) Machine Eligibility	Fix	U(0, 1)	U(0, 1)
G(J,P) Production Course	Fix	U(10, 50)	U(50, 100)
O(J,P) Hold course	Fix	U(10, 50)	U(50, 100)

4.5 การทดสอบและผลการทดสอบ

4.5.1 การทดสอบด้วยชุดข้อมูลอ้างอิง

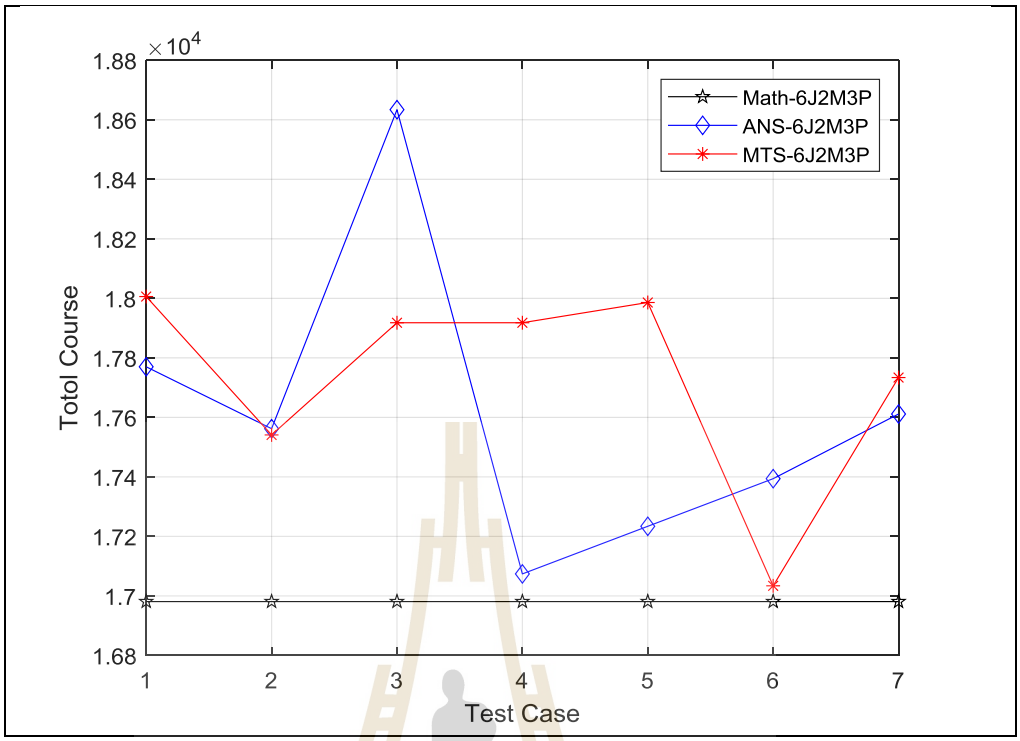
- ชุดข้อมูลอ้างอิง คือ ชุดที่ใช้ทดสอบแบบปัญหาทางคณิตศาสตร์(ในบทที่ 3)
- เพื่อทดสอบแบบปัญหาด้วยข้อมูลชุดเดียวกัน
- ทดสอบปัญญาเชิงคำนวณที่พัฒนาขึ้นด้วยชุดข้อมูลอ้างอิง
- เปรียบเทียบผลการทำงานของโปรแกรม ด้านเวลาที่ใช้ในการค้นหาคำตอบของปัญญาเชิงคำนวณ และ ค่าใช้จ่ายและเวลารวมของตารางการผลิต
- ตาราง 4.2 แสดงให้เห็นว่าค่าใช้จ่ายจากการทำงานด้วย ANS, MTS มีมากกว่า Math Model 3-5%
- ตาราง 4.2 เวลาในการค้นคำตอบของ ANS และ MTS ทำที่ประมาณ 40 วินาที โดยเฉลี่ย แต่วิธี Math ใช้เวลาประมาณ 4 วัน

- รูปที่ 4.3 ผลการทดสอบกับข้อมูลอ้างอิง-กรณี 06J02M03P การทำงานของ ANS และ MTS ให้คำตอบใกล้เคียงกัน แต่ไม่ชัดเจนว่าวิธีใดทำงานได้ดีกว่า อาจมาจากขนาดของปัญหายังไม่ใหญ่พอจะแยกความสามารถของทั้งสองวิธี
- รูปที่ 4.4 เปรียบเทียบการทำงานของ ANS, MTS และ Math ผลการค้นหาคำตอบมีค่าใกล้เคียงกัน

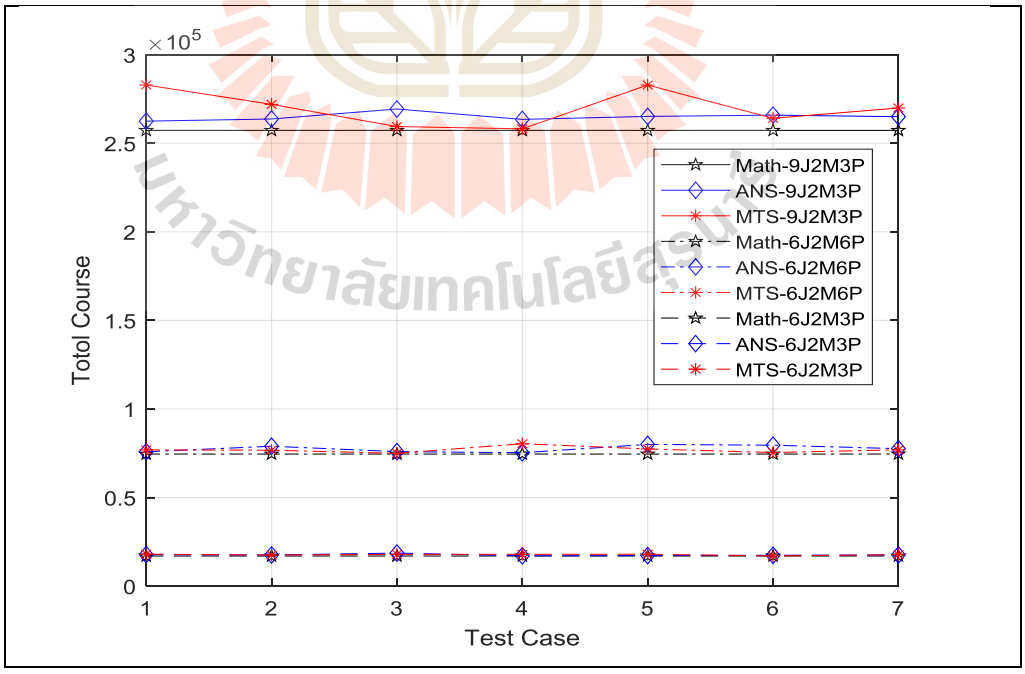
ตาราง 4.2 ผลการคำนวณด้วยข้อมูลปัญหาอ้างอิง

			Objective Value			Run Time		
Test	**J-M-P**	**Run**	**Math**	**ANS**	**MTS**	**Math**	**ANS**	**MTS**
1	6J-2M-3P	Run-1	16981	17770	18006	0d-00:00:38	33.5	33.5
		Run-2	16981	17562	17541	0d-00:00:31	30.5	34.2
		Run-3	16981	18634	17918	0d-00:00:29	29.5	32.3
		Run-4	16981	17074	17918	0d-00:00:32	34.5	31.9
		Run-5	16981	17234	17986	0d-00:00:28	33.5	32.0
		Run-6	16981	17394	17034	0d-00:00:40	36.5	32.0
		Average	16981.00	17611.33	17733.83	0d-00:00:33	33.00	32.65
	Diff. from Math		##	**-630.33**	**-752.83**	##	##	##
	Diff. from Math (%)		##	**-3.71**	**-4.43**	##	##	##
2	9J-2M-3P	Run-1	74536	75834	76878	4d-04:43:53	35.1	33.5
		Run-2	74536	78945	76696	4d-05:23:12	34.0	132.0
		Run-3	74536	75912	74952	4d-06:23:56	34.0	41.0
		Run-4	74536	75289	80352	4d-04:52:12	34.0	35.9
		Run-5	74536	79981	77408	4d-05:14:42	41.0	32.2
		Run-6	74536	79531	75395	4d-04:06:26	41.0	32.1
		Average	74536.00	77582.00	76946.83	4d-05:14:03	36.52	51.12
	Diff. from Math		##	**-3046.00**	**-2410.83**	##	##	##
	Diff. from Math (%)		##	**-4.09**	**-3.23**	##	##	##
3	6J-2M-6P	Run-1	257251	262467	282864	3d-23:23:59	38.5	36.7
		Run-2	257251	263675	271934	3d-22:05:57	37.4	36.5
		Run-3	257251	269307	259421	3d-22:08:18	38.1	37.2
		Run-4	257251	263458	258110	3d-21:40:40	37.2	37.2
		Run-5	257251	265133	282864	3d-22:07:33	39.2	36.5
		Run-6	257251	265858	264076	3d-23:50:41	37.3	36.7
		Average	257251.00	264983.00	269878.17	3d-22:24:33	37.95	36.80
	Diff. from Math		##	**-7732.00**	**-12627.17**	##	##	##
	Diff. from Math (%)		##	**-3.01**	**-4.91**	##	##	##

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 4.3 ผลการทดสอบกับข้อมูลอ้างอิง-กรณี 06J02M03P



รูปที่ 4.4 ผลการทดสอบกับข้อมูลอ้างอิง กรณี 6J2M3P, 9J2M3P, 6J2M6P

4.5.2 การทดสอบด้วยชุดข้อมูลขนาดเล็ก

- ข้อมูลขนาดเล็กเป็นข้อมูลที่ทดสอบด้วยแบบปัญหาทางคณิตศาสตร์ แล้วใช้เวลาต่ำกว่า 100 ชั่วโมง
- เป็นชุดข้อมูลที่สร้างขึ้นและมีขนาดของตัวแปรตามตาราง 4.3
- รูปแบบการจัดตารางงาน(Run Case) ของแต่ละรอบงานแต่ละเครื่องจักรจะมีค่าเป็นแฟคตอเรียลของจำนวนงาน(J!) เมื่อรวมแต่ละรอบงานแต่ละเครื่องจักรเข้าไป จะได้ว่า Run Case = J! ^ (M x P)
- เมื่อนำรูปแบบงานมาคิดขนาดสั่งผลิตให้ครบตามจำนวนที่ต้องการ จำนวนแบบการคำนวณจะมากขึ้น
- ทำการทดสอบ 10 กรณี แต่ละกรณีสร้างข้อมูลทดสอบ 6 ชุดที่แตกต่างกัน การแบ่งกรณีการทำงานและผลการทดสอบปรากฏในภาคผนวก ก.2 และนำเสนอบางส่วนของการทดสอบตามตาราง 4.4-กรณีที่หนึ่งทุกชุดทดสอบ และตาราง 4.5-ทุกกรณีชุดทดสอบที่หนึ่งและสอง
- ทดสอบทั้งสามวิธี Math, ANS, MTS ได้ผลการทำงานตามรูปที่ 4.5 ทั้งสามวิธีคำตอบด้านเวลาในการผลิตรวมของระบบไปในทางเดียวกัน และรูปที่ 4.6 คำตอบของ ANS, MTS ให้ผลการคำนวณที่มากกว่า Math ประมาณ 1-10%
- รูปที่ 4.7 ในกรณีของรูปแบบการจัดตารางงานสูง เวลาในการทำงานของ ANS และ MTS ต่ำกว่า Math มาก แต่เมื่อเปรียบเทียบ ANS กับ MTS ในรูปที่ 4.8 การทำงานของ ANS ใช้เวลาน้อยกว่าโดยเฉลี่ย

ตาราง 4.3 จำแนกตัวอย่างงานที่ใช้ทดสอบและจำนวนรูปแบบการจัดตารางงาน

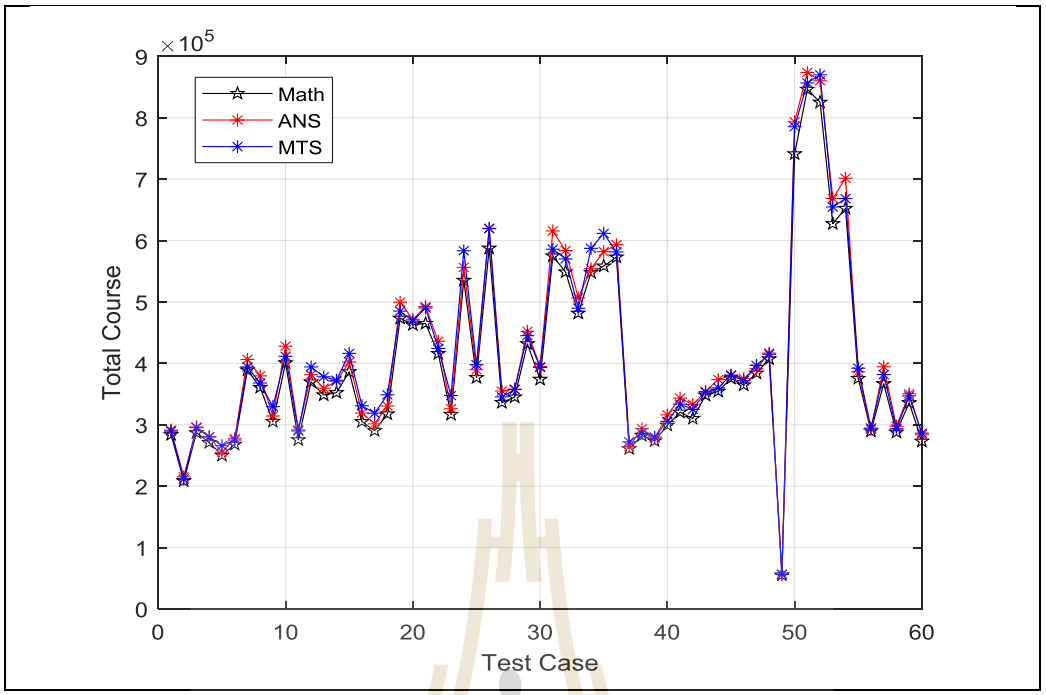
Test	J-M-P	nJob	nMachine	nPeriod	J!	M x P	Run Case
1	05J-02M-06P	5	2	6	120	12	8.9161 E+024
2	05J-02M-09P	5	2	9	120	18	2.66233 E+037
3	06J-02M-06P	6	2	6	720	12	1.94084 E+034
4	06J-02M-09P	6	2	9	720	18	2.70386 E+051
5	07J-02M-06P	7	2	6	5040	12	2.68637 E+044
6	07J-02M-09P	7	2	9	5040	18	4.40301 E+066
7	07J-03M-09P	7	3	9	5040	27	9.239 E+099
8	07J-03M-12P	7	3	12	5040	36	1.9386 E+133
9	08J-02M-09P	8	2	9	40320	18	7.93176 E+082
10	08J-03M-09P	8	3	9	40320	27	2.2339 E+124

ตาราง 4.4 ผลการทดสอบ-แสดงเฉพาะกรณี Test01-05J02M06P ทั้งหกชุดข้อมูล

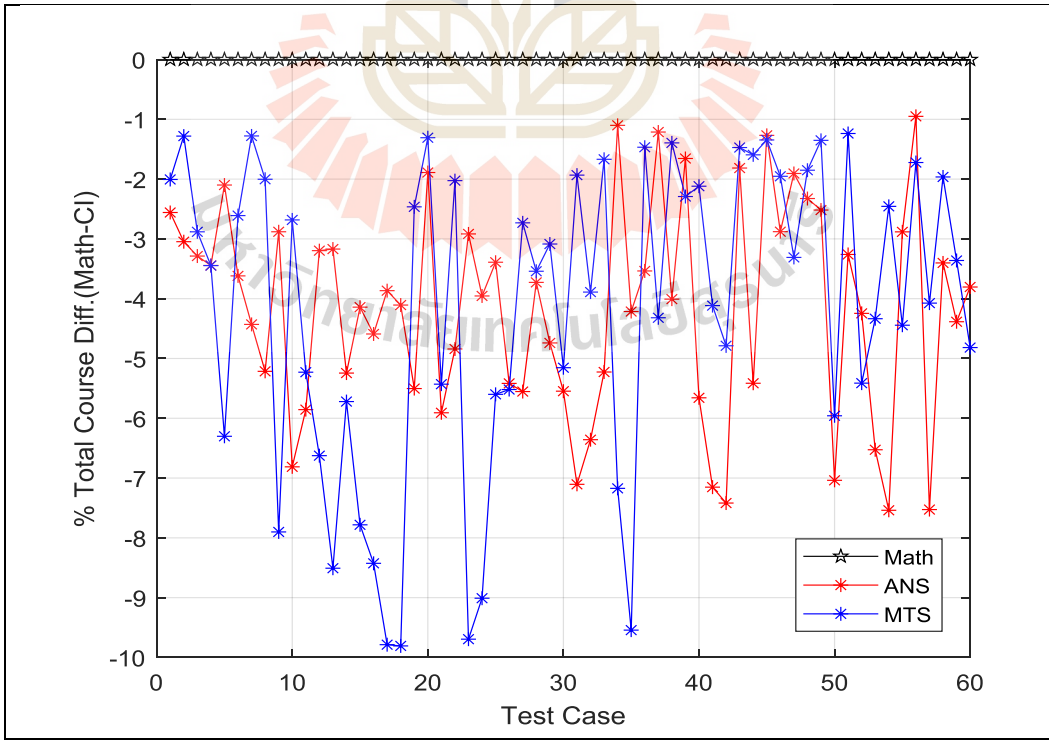
Item	Case	Objective Value			Diff. from Math (%)		Run Time(s)		
		Math	ANS	MTS	ANS	MTS	Math	ANS	MTS
1	test01_05J02M06P_A	283961	291226	289657	-2.56	-2.01	45	218	501
	test01_05J02M06P_B	208709	215068	211382	-3.05	-1.28	1	344	565
	test01_05J02M06P_C	287020	296454	295290	-3.29	-2.88	1	155	339
	test01_05J02M06P_D	270952	280285	280290	-3.44	-3.45	56	331	538
	test01_05J02M06P_E	250109	255359	265870	-2.10	-6.30	19	244	1264
	test01_05J02M06P_F	268139	277842	275138	-3.62	-2.61	1	295	135

ตาราง 4.5 ผลการทดสอบ-แสดงทุกกรณี นำเสนอกรณีละสองชุดข้อมูล

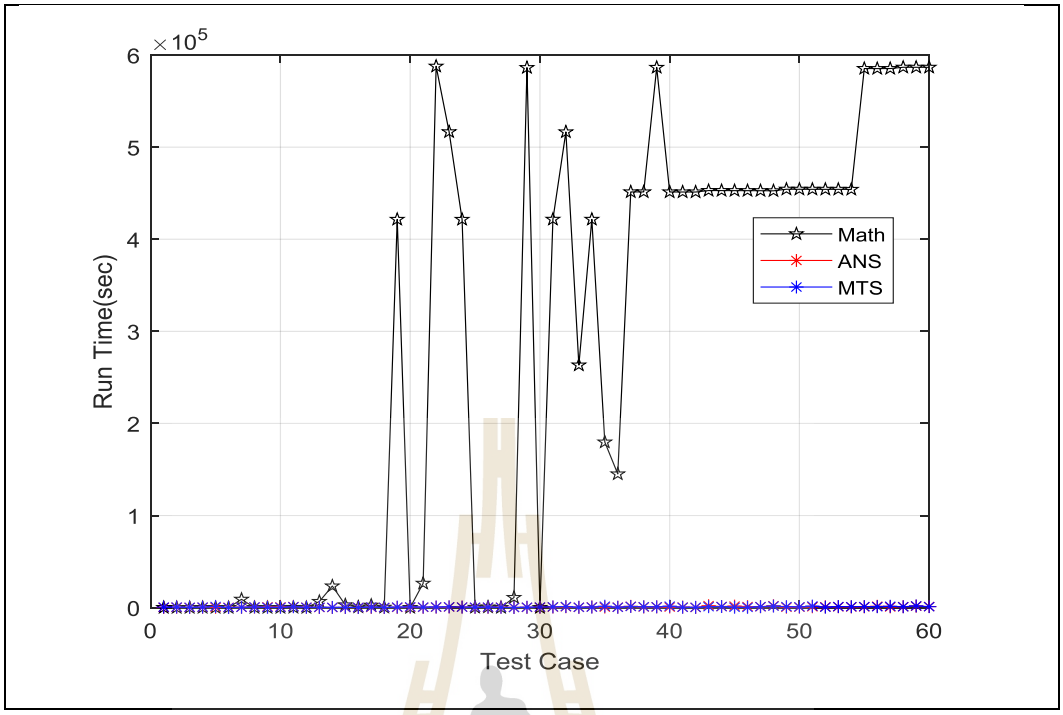
Item	Case	Objective Value			Diff. from Math (%)		Run Time(s)		
		Math	ANS	MTS	ANS	MTS	Math	ANS	MTS
1	test01_05J02M06P_A	283961	291226	289657	-2.56	-2.01	45	218	501
	test01_05J02M06P_B	208709	215068	211382	-3.05	-1.28	1	344	565
2	test02_05J02M09P_A	389196	406438	394170	-4.43	-1.28	9043	350	405
	test02_05J02M09P_B	360780	379595	368005	-5.22	-2	1	579	703
3	test03_06J02M06P_A	348475	359522	378123	-3.17	-8.51	6662	132	365
	test03_06J02M06P_B	352289	370764	372442	-5.24	-5.72	23433	157	275
4	test04_06J02M09P_A	473486	499542	485151	-5.5	-2.46	421550	282	561
	test04_06J02M09P_B	462935	471683	468989	-1.89	-1.31	60	384	825
5	test05_07J02M06P_A	376758	389532	397856	-3.39	-5.6	282	178	429
	test05_07J02M06P_B	587573	619400	620003	-5.42	-5.52	601	273	553
6	test06_07J02M09P_A	574836	615677	585949	-7.1	-1.93	421646	610	803
	test06_07J02M09P_B	548836	583732	570183	-6.36	-3.89	516520	905	1186
7	test07_07J03M09P_A	260944	264108	272214	-1.21	-4.32	451383	1430	1266
	test07_07J03M09P_B	282378	293695	286315	-4.01	-1.39	451356	676	931
8	test08_07J03M12P_A	348051	354368	353174	-1.81	-1.47	452857	2173	995
	test08_07J03M12P_B	354469	373661	360120	-5.41	-1.59	452835	1149	1155
9	test09_08J02M09P_A	55335	56728	56083	-2.52	-1.35	454278	460	1056
	test09_08J02M09P_B	741255	793424	785422	-7.04	-5.96	454249	545	809
10	test10_08J03M09P_A	375425	386246	392108	-2.88	-4.44	585367	847	1110
	test10_08J03M09P_B	289617	292369	294606	-0.95	-1.72	585286	1469	970



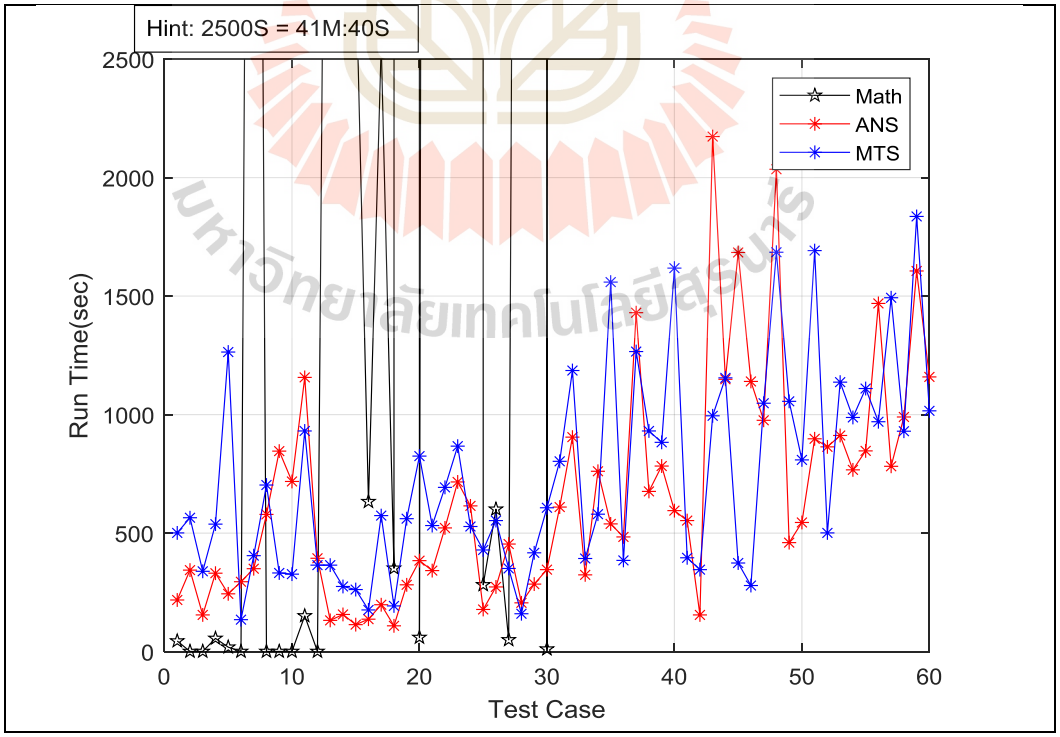
รูปที่ 4.5 ค่าฟังก์ชันวัตถุประสงค์เมื่อทดสอบกับข้อมูลขนาดเล็ก



รูปที่ 4.6 เปอร์เซ็นต์ผลต่างค่าฟังก์ชันวัตถุประสงค์เมื่อทดสอบกับข้อมูลขนาดเล็ก



รูปที่ 4.7 เวลาในการทำงานของโปรแกรม เมื่อทดสอบกับข้อมูลขนาดเล็ก-ภาพรวม



รูปที่ 4.8 เวลาในการทำงานของโปรแกรม เมื่อทดสอบกับข้อมูลขนาดเล็ก-ภาพขยาย

4.5.3 การทดสอบด้วยชุดข้อมูลขนาดใหญ่

- ข้อมูลขนาดใหญ่เป็นข้อมูลที่ทดสอบด้วยแบบปัญหาทางคณิตศาสตร์ แล้วใช้เวลามากกว่า 100 ชั่วโมง
- เป็นชุดข้อมูลที่สร้างขึ้นและมีขนาดของตัวแปรตามตาราง 4.6 โดย ทำการทดสอบ 15 กรณี แต่ละกรณีสร้างข้อมูลทดสอบอย่างต่ำ 4 ชุดที่แตกต่างกัน คำนวณรูปแบบการทำงานได้ตั้งแต่ 2.72E+157 ถึง 2.65253E+32^60 รูปแบบ
- ข้อมูลทั้งหมด ทดสอบด้วย ANS และ MTS ผลการทำงานทั้งหมดแสดงในภาคผนวก ก.3 และนำเสนอผลการทำงาน บางกรณีในตาราง 4.7 และ 4.8
- เปรียบเทียบค่าใช้จ่ายในการผลิตรวมของระบบในแต่ละกรณีทดสอบรูปที่ 4.9, 4.10, 4.11 และ 4.12 แสดงโดยการเรียงลำดับค่าใช้จ่ายของแต่ละกรณีทดสอบย่อย เห็นว่า คำตอบของ ANS ทำได้ต่ำกว่า MTS เล็กน้อย
- เปรียบเทียบค่าเฉลี่ยของแต่ละกรณีทดสอบในรูปที่ 4.13 จะเห็นว่าใกล้เคียงกัน
- เปรียบเทียบเวลาโปรแกรมทำงานเฉลี่ยของแต่ละกรณีทดสอบในรูปที่ 4.14 จะเห็นว่าใกล้เคียงกัน

ตาราง 4.6 จำแนกตัวอย่างงานที่ใช้ทดสอบและจำนวนรูปแบบการจัดตารางงาน

Test	Name Case	J-M-P	nJob	nMachine	nPeriod	J!	M x P	Run Case
1	T21, T41, T61	10J-04M-06P	10	4	6	3628800	24	2.72E+157
2	T22, T42, T62	10J-04M-09P	10	4	9	3628800	36	1.42E+236
3	T23, T43, T63	10J-06M-09P	10	6	9	3628800	54	Inf
4	T24, T44, T64	15J-04M-06P	15	4	6	1.30767E+12	24	6.25E+290
5	T25, T45, T65	15J-04M-09P	15	4	9	1.30767E+12	36	Inf
6	T26, T46, T66	15J-06M-09P	15	6	9	1.30767E+12	54	Inf
7	T27, T47, T67	20J-04M-06P	20	4	6	2.43290E+18	24	Inf
8	T28, T48, T68	20J-04M-09P	20	4	9	2.43290E+18	36	Inf
9	T29, T49, T69	20J-06M-09P	20	6	9	2.43290E+18	54	Inf
10	T30, T50, T70	25J-06M-06P	25	6	6	1.55112E+25	36	Inf
11	T31, T51, T71	25J-08M-06P	25	8	6	1.55112E+25	48	Inf
12	T32, T52, T72	25J-10M-06P	25	10	6	1.55112E+25	60	Inf
13	T33, T53, T73	30J-06M-06P	30	6	6	2.65253E+32	36	Inf
14	T34, T54, T74	30J-08M-06P	30	8	6	2.65253E+32	48	Inf
15	T35, T55, T75	30J-10M-06P	30	10	6	2.65253E+32	60	Inf

ตาราง 4.7 ผลการทดสอบ-แสดงเฉพาะกรณี Test01-10J04M06P

Item	File Name	Case	Mode	Objt	Time	Iter
1	RDTest61a_06J04M10P_20201014_2148.log	Test61a	ANN	142950	2117	7800
2	RDTest61a_06J04M10P_20201014_2340.log	Test61a	ANN	142950	2084	7800
3	RDTest61a_06J04M10P_20201015_0232.log	Test61a	ANN	142950	2036	7800
4	RDTest61a_06J04M10P_20201014_2032.log	Test61a	ANN	150833	2182	8100
5	RDTest61a_06J04M10P_20201014_2302.log	Test61a	ANN	150833	2163	8100
6	RDTest61a_06J04M10P_20201015_0057.log	Test61a	ANN	150833	2209	8100
7	RDTest61a_06J04M10P_20201015_0151.log	Test61a	ANN	150833	2124	8100
8	RDTest61a_06J04M10P_20201015_0535.log	Test61a	ANN	150833	2167	8100
9	RDTest41a_06J04M10P_20201014_2147.log	Test41a	ANN	165756	2036	7600
10	RDTest41a_06J04M10P_20201015_0057.log	Test41a	ANN	165756	2004	7600
11	RDTest41a_06J04M10P_20201015_0252.log	Test41a	ANN	182624	1319	5000
12	RDTest41a_06J04M10P_20201014_2019.log	Test41a	ANN	191852	1416	5200
13	RDTest41a_06J04M10P_20201014_2301.log	Test41a	ANN	191852	1367	5200
14	RDTest41a_06J04M10P_20201015_0536.log	Test41a	ANN	191852	1394	5200
15	RDTest21a_06J04M10P_20201015_0059.log	Test21a	ANN	237633	1142	4300
16	RDTest21a_06J04M10P_20201014_2147.log	Test21a	ANN	252108	1591	6000
17	RDTest21a_06J04M10P_20201014_1938.log	Test21a	ANN	254924	1130	4200
18	RDTest21a_06J04M10P_20201015_0539.log	Test21a	ANN	254924	1131	4200
19	RDTest21c_06J04M10P_20201015_0239.log	Test21c	ANN	258889	2270	8500
20	RDTest21c_06J04M10P_20201014_2351.log	Test21c	ANN	261594	977	3700
21	RDTest21c_06J04M10P_20201014_2111.log	Test21c	ANN	265310	1171	4400
22	RDTest21c_06J04M10P_20201015_0714.log	Test21c	ANN	265310	1176	4400
23	RDTest21d_06J04M10P_20201014_2031.log	Test21d	ANN	286347	968	3600
24	RDTest21d_06J04M10P_20201015_0633.log	Test21d	ANN	286347	974	3600
25	RDTest21d_06J04M10P_20201015_0140.log	Test21d	ANN	290878	1370	5100
26	RDTest21d_06J04M10P_20201014_2245.log	Test21d	ANN	296948	2382	9000
27	MTSTest61a_06J04M10P_20201014_2224.log	Test61a	MTS	146934	1290	4600
28	MTSTest61a_06J04M10P_20201015_0015.log	Test61a	MTS	146934	1289	4600
29	MTSTest61a_06J04M10P_20201015_0306.log	Test61a	MTS	146934	1255	4600
30	MTSTest41a_06J04M10P_20201014_2043.log	Test41a	MTS	172424	2336	8300
31	MTSTest41a_06J04M10P_20201014_2324.log	Test41a	MTS	172424	2351	8300
32	MTSTest41a_06J04M10P_20201015_0559.log	Test41a	MTS	172424	2369	8300
33	MTSTest41a_06J04M10P_20201014_2221.log	Test41a	MTS	182618	1596	5700

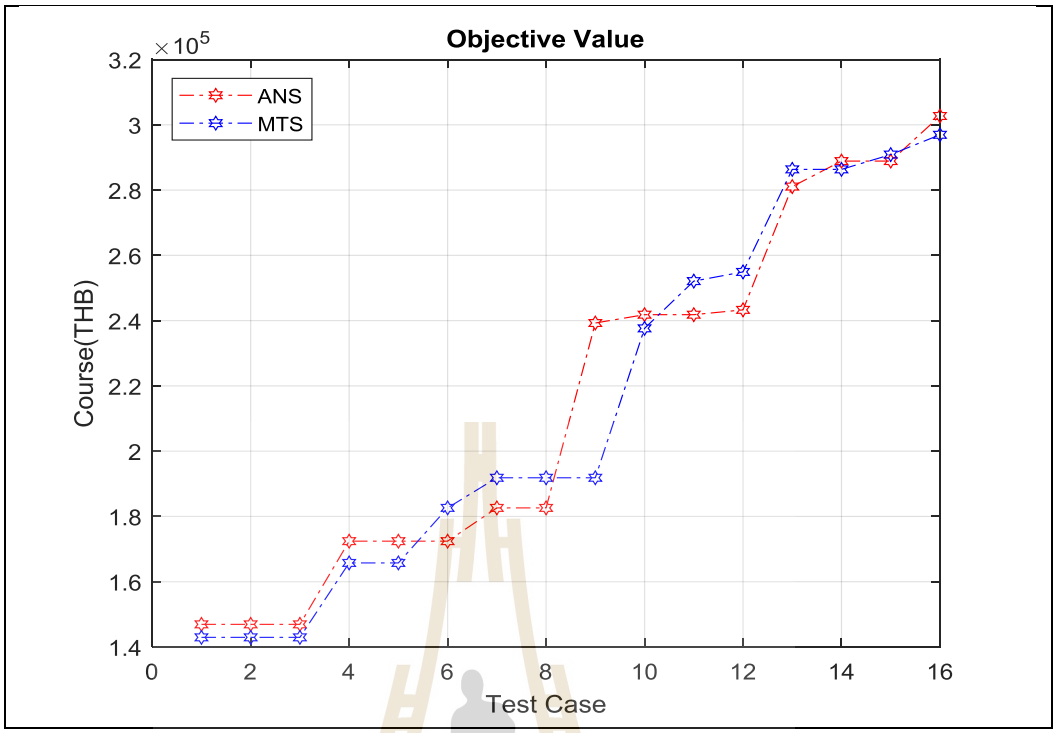
ตาราง 4.7 ผลการทดสอบ- แสดงเฉพาะกรณี Test01-10J04M06P (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
34	MTSTest41a_06J04M10P_20201015_0131.log	Test41a	MTS	182618	1596	5700
35	MTSTest21a_06J04M10P_20201015_0118.log	Test21a	MTS	239270	1335	4700
36	MTSTest21a_06J04M10P_20201014_1957.log	Test21a	MTS	241824	2031	7200
37	MTSTest21a_06J04M10P_20201015_0558.log	Test21a	MTS	241824	2082	7200
38	MTSTest21a_06J04M10P_20201014_2214.log	Test21a	MTS	243298	1839	6500
39	MTSTest21d_06J04M10P_20201014_2324.log	Test21d	MTS	281077	1595	5600
40	MTSTest21d_06J04M10P_20201014_2047.log	Test21d	MTS	288884	1451	5100
41	MTSTest21d_06J04M10P_20201015_0649.log	Test21d	MTS	288884	1486	5100
42	MTSTest21d_06J04M10P_20201015_0203.log	Test21d	MTS	302708	2181	7700

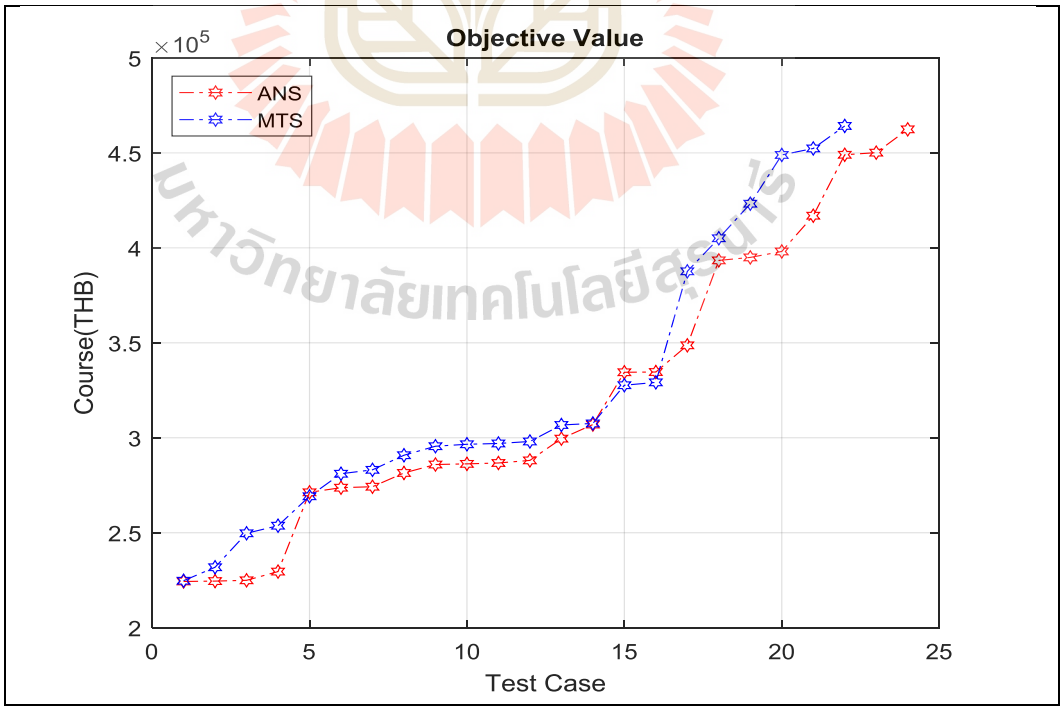
ตาราง 4.8 ผลการทดสอบ-แสดงค่าเฉลี่ยของค่าใช้จ่ายละเวลาของแต่ละกรณีทดสอบ

Test	J-M-P	ANS.Objt	ANS.Time	ANS.Iter	MTS.Objt	MTS.Time	MTS.Iter
1	10J-04M-06P	210,919.96	1,755.13	6,200.00	215,692.44	1,650.00	6,180.77
2	10J-04M-09P	474,947.76	2,813.55	7,227.27	450,610.09	2,225.38	6,095.24
3	10J-06M-09P	322,511.29	6,001.55	10,000.00	323,809.23	6,846.58	11,983.33
4	15J-04M-06P	306,949.86	4,275.93	7,128.57	307,044.36	4,099.64	7,178.57
5	15J-04M-09P	577,377.77	4,748.75	5,850.00	537,797.13	4,776.62	5,984.62
6	15J-06M-09P	634,944.40	15,759.45	12,390.91	593,477.36	13,733.60	11,270.00
7	20J-04M-06P	558,912.00	9,133.00	9,185.71	433,725.14	5,370.71	5,628.57
8	20J-04M-09P	901,687.67	6,391.00	4,500.00	962,129.00	8,513.83	6,225.00
9	20J-06M-09P	679,152.67	26,081.25	11,150.00	701,678.50	35,267.00	15,783.33
10	25J-06M-06P	394,504.31	13,770.75	5,850.00	464,974.13	15,590.46	6,930.77
11	25J-08M-06P	280,945.78	15,566.00	4,785.71	284,411.57	21,142.33	6,766.67
12	25J-10M-06P	230,016.57	27,311.00	6,100.00	261,628.00	29,114.57	6,828.57
13	30J-06M-06P	561,991.80	17,834.00	5,311.11	523,465.11	16,671.00	5,180.00
14	30J-08M-06P	368,534.60	25,111.33	5,233.33	319,330.67	26,738.40	5,800.00
15	30J-10M-06P	340,394.75	35,668.00	5,733.33	247,793.67	29,730.00	5,025.00

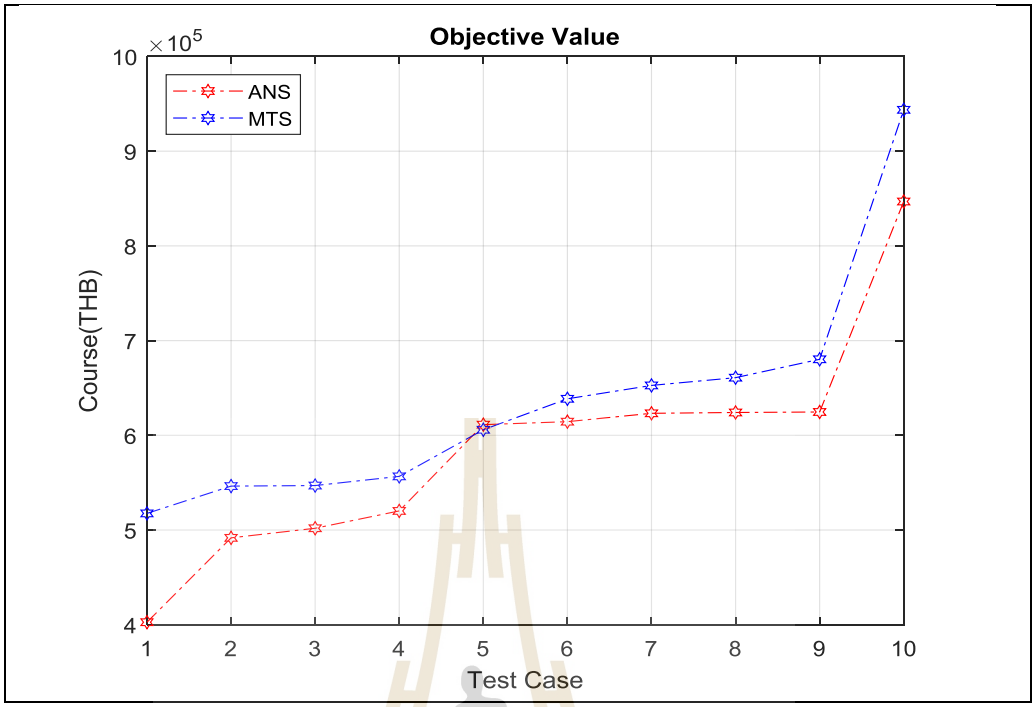
มหาวิทยาลัยเทคโนโลยีสุรนารี



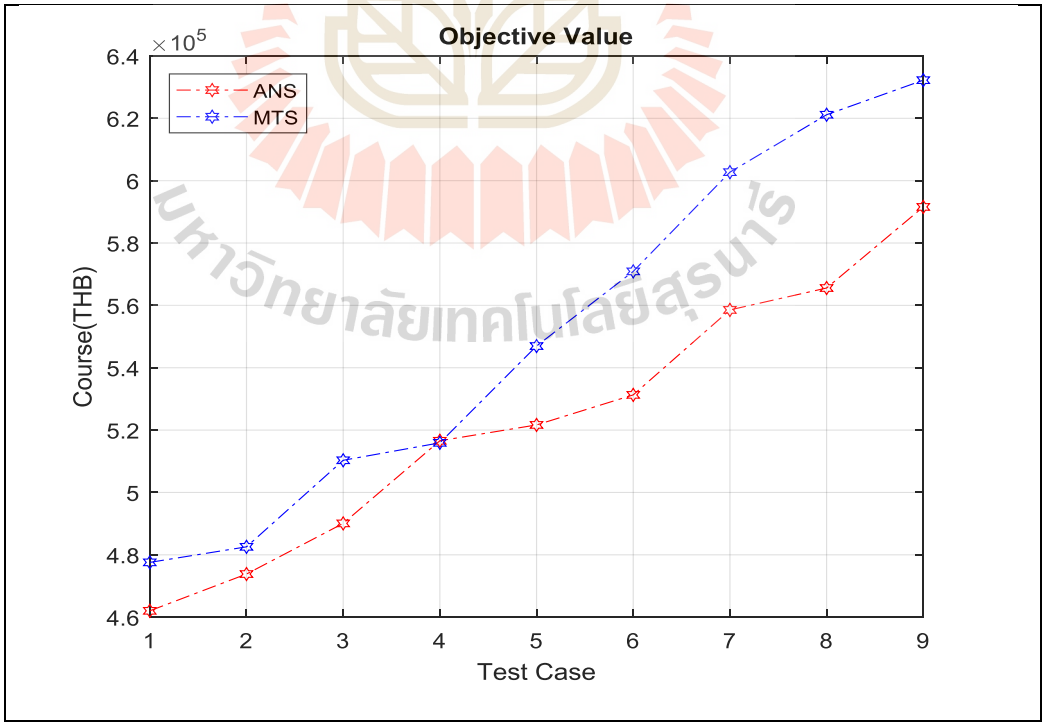
รูปที่ 4.9 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Test01 -- 10J04M06P



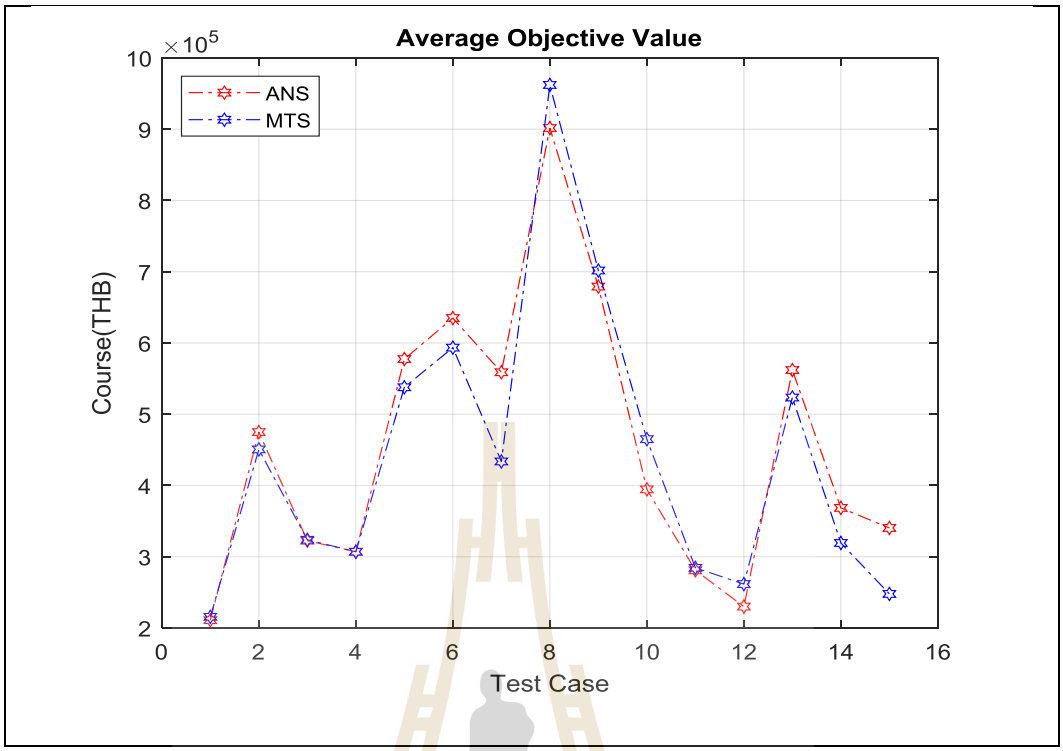
รูปที่ 4.10 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Test04 -- 15J04M06P



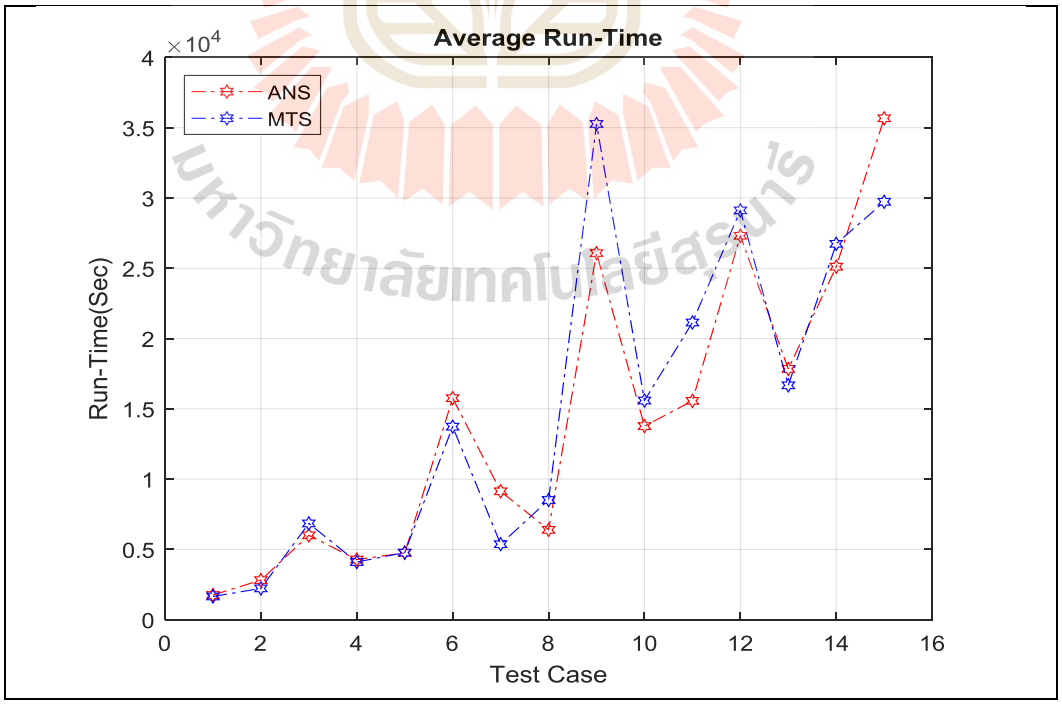
รูปที่ 4.11 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Index = Test06 – 15J06M09P



รูปที่ 4.12 ค่าฟังก์ชันวัตถุประสงค์ กรณีงาน Index = Test13 -- 30J06M06P



รูปที่ 4.13 เปรียบเทียบค่าฟังก์ชันวัตถุประสงค์เฉลี่ยของแต่ละกรณีทดสอบ



รูปที่ 4.14 เปรียบเทียบเวลาในการทำงานของโปรแกรมเฉลี่ยของแต่ละกรณีทดสอบ

4.6 สรุป

การแก้ปัญหาเพื่อหาผลเฉลยของแบบจำลองการจัดตารางการผลิตด้วยวิธีฮิวริสติคที่ใช้ปัญญาเชิงคำนวณ ในเนื้อหาส่วนนี้อธิบายถึง

1. ขั้นตอนการคำนวณฟังก์ชันวัตถุประสงค์ เริ่มตั้งแต่แนวคิดการพัฒนาโปรแกรมที่แบ่งการค้นคำตอบเป็นสองส่วน คือ รูปแบบลำดับของการจัดตารางงาน และจำนวนขนาดสั่งผลิตที่ใช้นำไปสู่ขั้นตอนการคำนวณเพื่อหาค่าฟังก์ชันวัตถุประสงค์ซึ่งมีสามส่วนคือ ค่าการผลิต ค่าเก็บรักษา และค่าติดตั้งเครื่องจักร ส่วนขั้นตอนการค้นหาคำตอบจะขึ้นกับปัญญาเชิงคำนวณแต่ละวิธีที่เลือกใช้

2. การแบ่งขนาดของปัญหา ได้กล่าวถึง ปัญหาอ้างอิงเพื่อทดสอบแบบจำลอง ปัญหาขนาดเล็ก และปัญหาขนาดใหญ่ รวมถึงการกำหนดข้อมูลทดสอบตามขนาดของปัญหา

3. การทำงานของปัญญาเชิงคำนวณ แบบการค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว (ANS: Adaptive Nearest Neighbor Search) และ การค้นหาแบบตาบูประยุกต์(MTS: Modify Tabu Search)

4. ผลการทดสอบในปัญหาอ้างอิง ANS และ MTS ให้คำตอบมากกว่า Math ประมาณ 3-5% แต่ใช้เวลาในการค้นคำตอบจาก 100 ชั่วโมง เหลือเพียง 60 วินาทีโดยประมาณ

5. ผลการทดสอบในปัญหาขนาดเล็ก ANS และ MTS ให้คำตอบมากกว่า Math ไม่เกิน 10% และคำตอบของ ANS กับ MTS เกาะกลุ่มใกล้เคียงกันแต่ของ ANS ดีกว่าเล็กน้อยทั้งในแง่ของคำตอบและเวลาในกการทำงาน

6. ผลการทดสอบในปัญหาขนาดใหญ่ ANS และ MTS ให้คำตอบเกาะกลุ่มใกล้เคียงกันแต่ของ ANS ดีกว่าเล็กน้อยทั้งในแง่ของคำตอบและเวลาในกการทำงานของโปรแกรม

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 5
การจัดตารางการผลิตแบบหลายวัตถุประสงค์

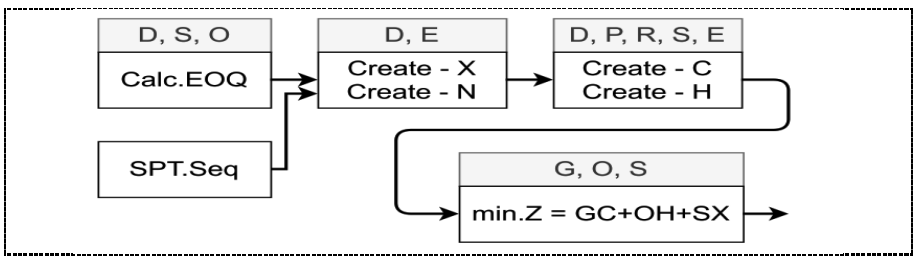
5.1 บทนำ

บทนี้นำเสนอการจัดตารางการผลิตแบบหลายวัตถุประสงค์ เพื่อให้สามารถเลือกแผนการผลิตที่เหมาะสมในการทำงาน การทดสอบทำโดยกำหนดให้มีฟังก์ชันวัตถุประสงค์สองชุด ชุดแรกเพื่อเป็นการลดเวลาและค่าใช้จ่ายในการผลิตรวมของระบบ ชุดที่สองเพื่อให้เครื่องจักรขนานมีอัตราการใช้สอยทรัพยากร (Utilization) สูงสุด

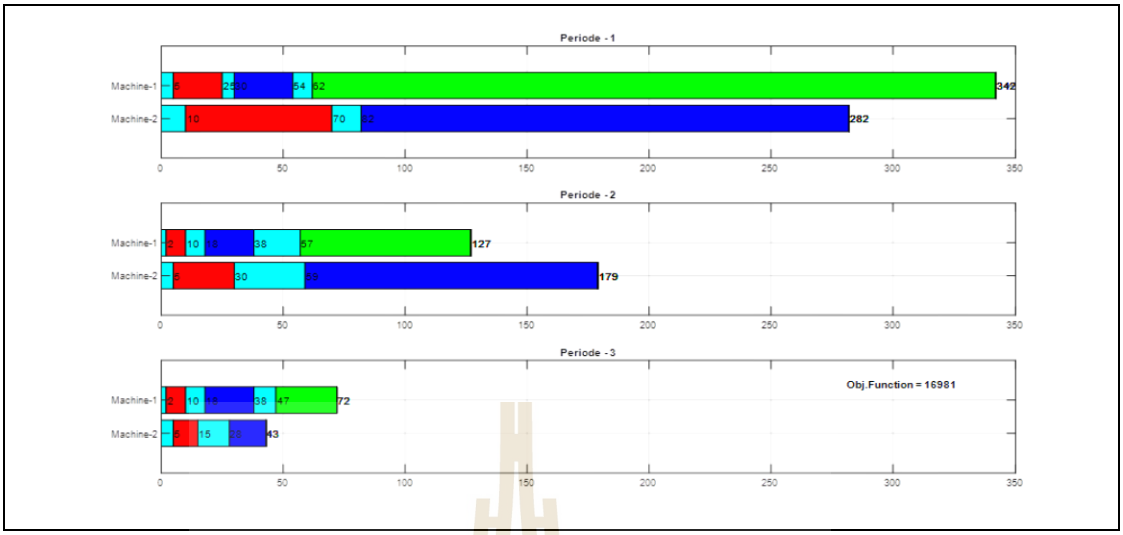
การออกแบบการทำงานของโปรแกรมทำเป็น 3 แบบ คือ แบบที่หนึ่ง เรียกว่าการจัดตารางงานแบบสองวัตถุประสงค์เทียม โปรแกรมจะทำการค้นหาคำตอบโดยใช้ฟังก์ชันวัตถุประสงค์ชุดที่หนึ่งจนเสร็จแล้วเก็บคำตอบดีที่สุดไว้จำนวนหนึ่งจากนั้นนำคำตอบที่ได้หาภาระงานของเครื่องจักรแล้วเลือกค่าที่เหมาะสมเป็นแผนการผลิต, แบบที่สอง การจัดตารางงานแบบสองวัตถุประสงค์แบบขนาน โปรแกรมแต่ละรอบการค้นหาคำตอบจะหาค่าฟังก์ชันวัตถุประสงค์ทั้งสองชุด แล้วเก็บคำตอบดีที่สุดของแต่ละชุดไว้จำนวนหนึ่ง จากนั้นก็แล้วเลือกค่าที่เหมาะสมเป็นแผนการผลิต, แบบที่สาม การจัดตารางงานแบบสองวัตถุประสงค์แบบอนุกรม โปรแกรมแต่ละรอบการค้นหาคำตอบจะหาค่าฟังก์ชันวัตถุประสงค์ของชุดแรก หากอยู่ในเงื่อนไขที่ยอมรับได้ชุดคำตอบจะส่งไปคำนวณหาค่าฟังก์ชันวัตถุประสงค์ของชุดที่สอง หากอยู่ในเงื่อนไขที่ยอมรับได้ชุดคำตอบนี้จะถูกเก็บไว้เป็นชุดคำตอบที่ดี เมื่อรอบการทำงานสิ้นสุดจะเลือกชุดคำตอบที่เหมาะสมเป็นแผนการผลิต

5.2 รูปแบบการทดสอบ

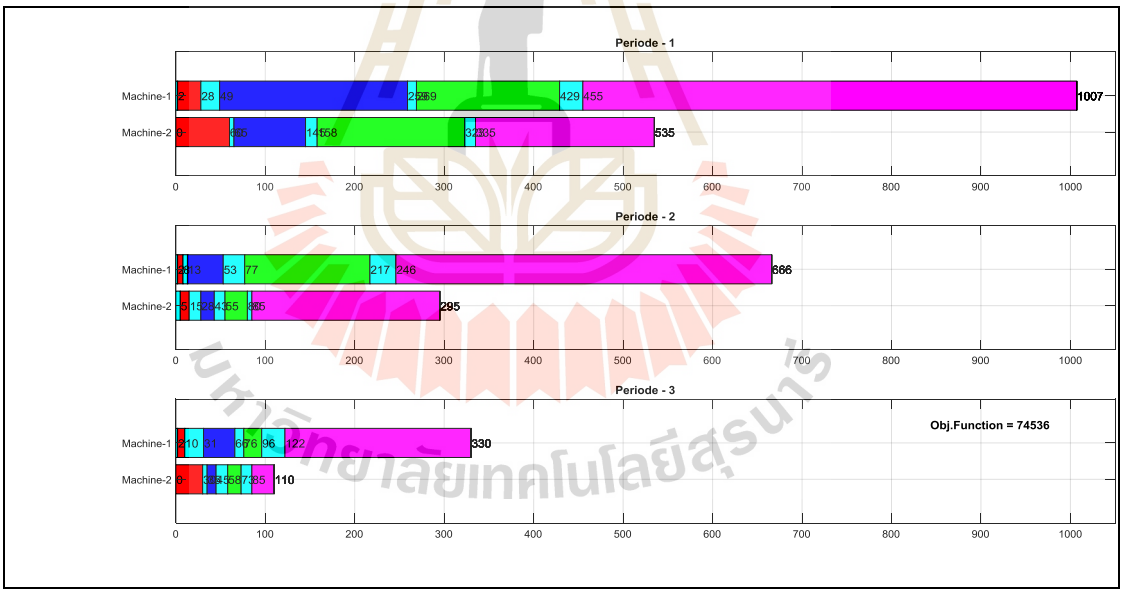
5.2.1 Model-0: การจัดตารางงานแบบวัตถุประสงค์เดียว



รูปที่ 5.1 แผนภาพขั้นตอนการคำนวณฟังก์ชันวัตถุประสงค์



รูปที่ 5.2 ตารางการผลิต กรณีทดสอบ-t0501



รูปที่ 5.3 ตารางการผลิต กรณีทดสอบ-t0502

จากการทำงานในบทที่ 3 ผลการจัดตารางงาน เห็นว่าเครื่องจักรทำงานไม่เต็มชั่วโมงการทำงาน ซึ่งเป็นการทำงานที่ไม่เต็มประโยชน์ของเครื่องจักร จึงมีแนวคิดว่าหากนำค่านี้มาพิจารณาเทียบกับการลดค่าใช้จ่ายในการผลิตจะทำให้การตัดสินใจเลือกแผนการผลิตทำได้เหมาะสมมากขึ้น

อัตราการใช้สอยทรัพยากร(Utilization) คือ ร้อยละของจำนวนเวลาที่เครื่องจักรทำงานได้จริง ยกตัวอย่าง เช่น ในการทำงาน 1 รอบงานคือ 8 ชั่วโมง มีทำงานจริง 7 ชั่วโมง เวลาพัก 1 ชั่วโมง จะมีค่าอัตราการใช้สอยทรัพยากรเท่ากับ 87.5%

ในการวิจัยกำหนดใช้เวลาปิดงานสูงสุดของแต่ละเครื่องจักรคำนวณเทียบกับชั่วโมงการทำงาน จากนั้นนำแผนการทำงานของเครื่องจักรทั้งหมดทุกรอบการทำงาน ทุกเครื่องจักรมาเฉลี่ยเพื่อเป็นตัวแทนของการจัดตารางงาน

จากผลการทำงานในบทที่ 3 การจัดตารางงานเป็นไปดัง รูปที่ 5.2 และ รูปที่ 5.3 เมื่อคำนวณค่าอัตราการใช้สอยทรัพยากร ผลการคำนวณได้ตามตารางที่ 1 โดย แผนงาน t-0501 ให้ค่า 49.76% และแผนงาน t-0502 ให้ค่า 44.59% หากทั้งสองแบบเป็นการจัดตารางงานเดียวกัน ค่าที่สูงกว่าหมายความว่าเครื่องจักรมีการใช้งานเต็มกำลังมากกว่า หากพิจารณาเฉพาะการใช้สอยทรัพยากรจึงควรเลือกแผน t-0501

ตารางที่ 5.1 ผลการคำนวณเพื่อหาค่าอัตราการใช้สอยทรัพยากรเฉลี่ยของการจัดตารางงาน

Test Case	Period	Machine	Makespan	WorkHour	Utilization	Average
t-0501	P1	M1	342	350	97.71	49.76
		M2	282	350	80.57	
	P2	M1	127	350	36.29	
		M2	179	350	51.14	
	P3	M1	72	350	20.57	
		M2	43	350	12.29	
t-0502	P1	M1	1007	1100	91.55	44.59
		M2	535	1100	48.64	
	P2	M1	666	1100	60.55	
		M2	295	1100	26.82	
	P3	M1	330	1100	30.00	
		M2	110	1100	10.00	

ในการทดสอบกำหนดตัวแปร Z เป็นค่าใช้จ่ายในการผลิต และ U เป็นค่าอัตราการใช้สอยทรัพยากร เมื่อตัดสินใจเลือกแผนการผลิตจะกำหนดไว้ 3 กรณี ได้แก่ (1) min.Z-ค่าใช้จ่ายต่ำสุด, (2) max.U-อัตราการใช้สอยทรัพยากรสูงสุด และ (3) mean.ZU-ค่าที่อยู่กลางระหว่าง min.Z กับ max.U การหาค่า mean.ZU ทำโดยการใช้คะแนนมาตรฐาน (z-score) ของชุดคำตอบ Z และ U ตามขั้นตอน ดังนี้

1. เลือกแผนการผลิตที่ดีสุดจำนวนหนึ่ง โดยเลือกจาก Z หรือ U
2. เรียกข้อมูลชุดนี้ว่า Z_List และ U_List ตามลำดับ
3. คำนวณหาขนาดของคะแนนมาตรฐาน (z-score) ของ Z_List และ U_List จาก

$$Z_{z-score} = \left| \frac{Z_i - \overline{Z}}{\sigma_z} \right| , \; U_{z-score} = \left| \frac{U_i - \overline{U}}{\sigma_u} \right|$$

4. คำนวณหาค่า $ZU_{z\text{-score}}$ ด้วยการนำค่า $Z_{z\text{-score}}$ และ $U_{z\text{-score}}$ มาเฉลี่ยเท่า ๆ กัน

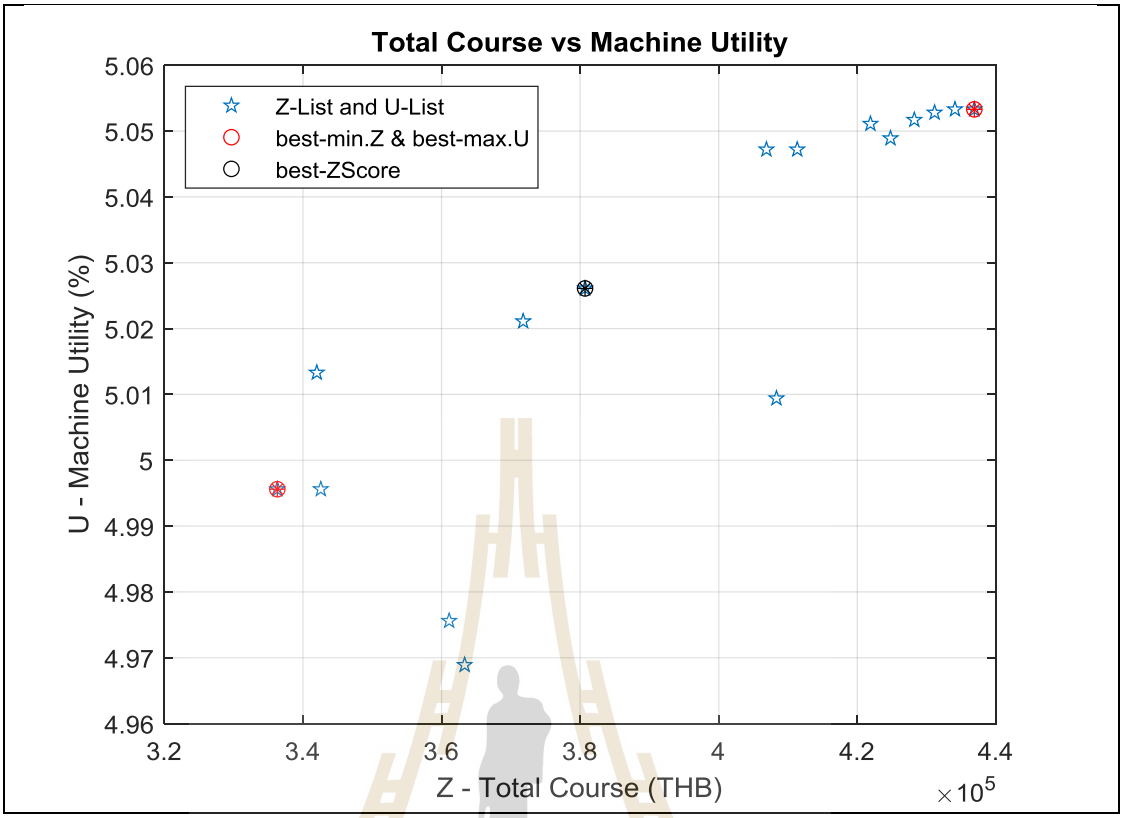
$$ZU_{z-score} = (0.5)Z_{z-score} + (0.5)U_{z-score}$$

5. เลือกแผนการผลิตที่ทำให้ค่า $ZU_{z\text{-score}}$ น้อยที่สุด แผนการผลิตนี้จะเป็นค่ากลางระหว่าง min.Z และ max.U เรียกแผนการผลิตนี้ว่า mean.ZU
6. คำตอบจากเซ็ตคำตอบสามค่าจาก Z_List และ U_List ได้แก่
   (1) best.Z หรือ min.Z คือ ค่าต่ำสุดของค่าใช้จ่ายในการผลิตรวม
   (2) best.U หรือ max.U คือ ค่าสูงสุดอัตราการใช้สอยทรัพยากร

ตารางที่ 5.2 การคำนวณเพื่อหาแผนการผลิตจากค่า best.ZU

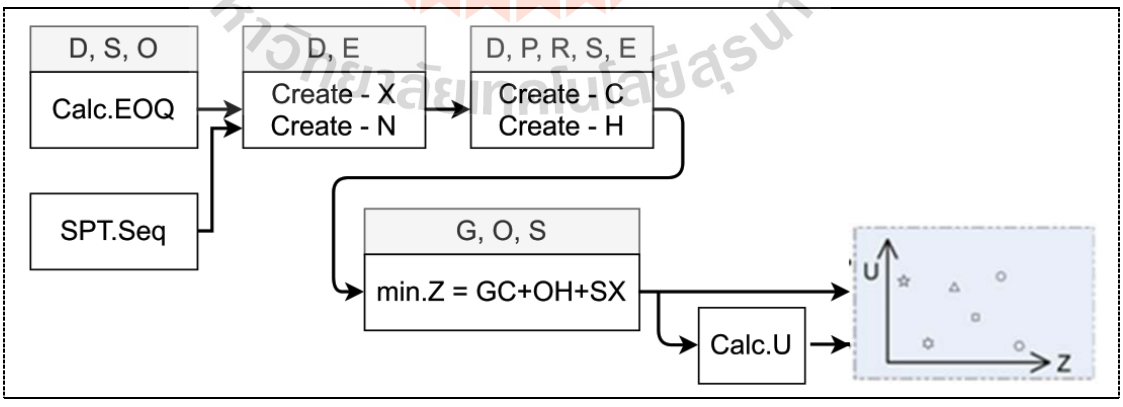
Item	Z	U	Z-zscore	U-zscore	ZU-zscore	Note
1	336302	4.9956	1.3487	1.3449	1.3468	min.Z > best.Z
2	341996	5.0133	1.1958	0.5631	0.8794	
3	342577	4.9956	1.1802	1.3449	1.2625	
4	371784	5.0211	0.3955	0.2186	0.3071	
5	380712	5.0261	0.1556	0.0022	0.0789	min.ZU > best.ZU
6	408336	5.0094	0.5865	0.7354	0.6609	
7	411333	5.0472	0.6670	0.9341	0.8006	
8	436890	5.0533	1.3536	1.2035	1.2786	max.U > best.U
9	428229	5.0517	1.1210	1.1329	1.1269	
10	406887	5.0472	0.5476	0.9341	0.7409	
Max		**5.0533**				
Min	**336302**				**0.0789**	
Mean	**386504.6**	**5.02605**				
Stdev	**37222.03**	**0.02264**				

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 5.4 ตำแหน่งของแผนการผลิตที่ดีสุด best.Z, best.U และ best.ZU

5.2.2 Model-1: การจัดตารางงานแบบสองวัตถุประสงค์เทียม



รูปที่ 5.5 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์ แบบสองวัตถุประสงค์เทียม

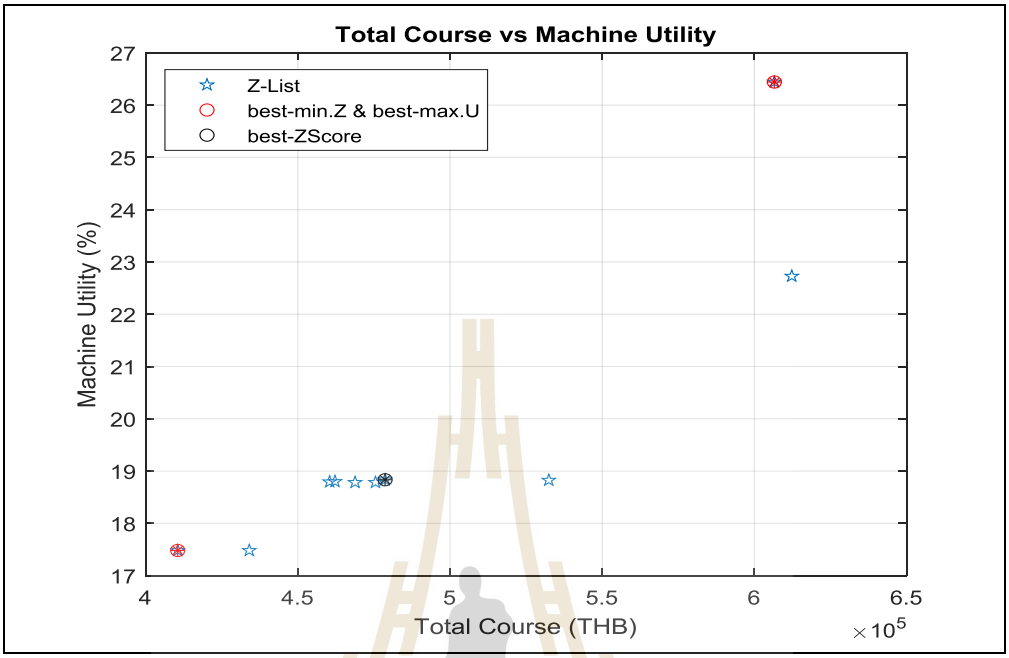
ขั้นตอนการทำงาน เงื่อนไข และการทดสอบ

1. ค้นหาเซ็ตคำตอบด้วยฟังก์ชันวัตถุประสงค์หนึ่ง { ค่า min.Z } จนแล้วเสร็จและทำการเก็บชุดคำตอบไว้จำนวนหนึ่ง ในการทดสอบเก็บไว้จำนวน 10 ชุด เรียกชุดคำตอบนี้ว่า Z_List
2. จาก Z_List คำนวณฟังก์ชันวัตถุประสงค์สอง { ค่า Utilization } จะได้ค่า U_List
3. คำนวณหา ขนาดของคะแนนมาตรฐาน (z-score) ของ Z_List และ U_List
4. คำนวณหาค่า $ZU_{z\text{-score}}$
5. เลือกคำตอบจากเซ็ตคำตอบสามค่าจาก Z_List และ U_List
   (1) best.Z แผนการผลิตที่ให้ค่าใช้จ่ายในการผลิตรวมต่ำสุด
   (2) best.U แผนการผลิตที่ให้ค่าอัตราการใช้สอยทรัพยากรสูงสุด
   (3) best.ZU แผนการผลิตที่อยู่ระหว่างค่าใช้จ่ายต่ำและใช้ทรัพยากรเหมาะสม

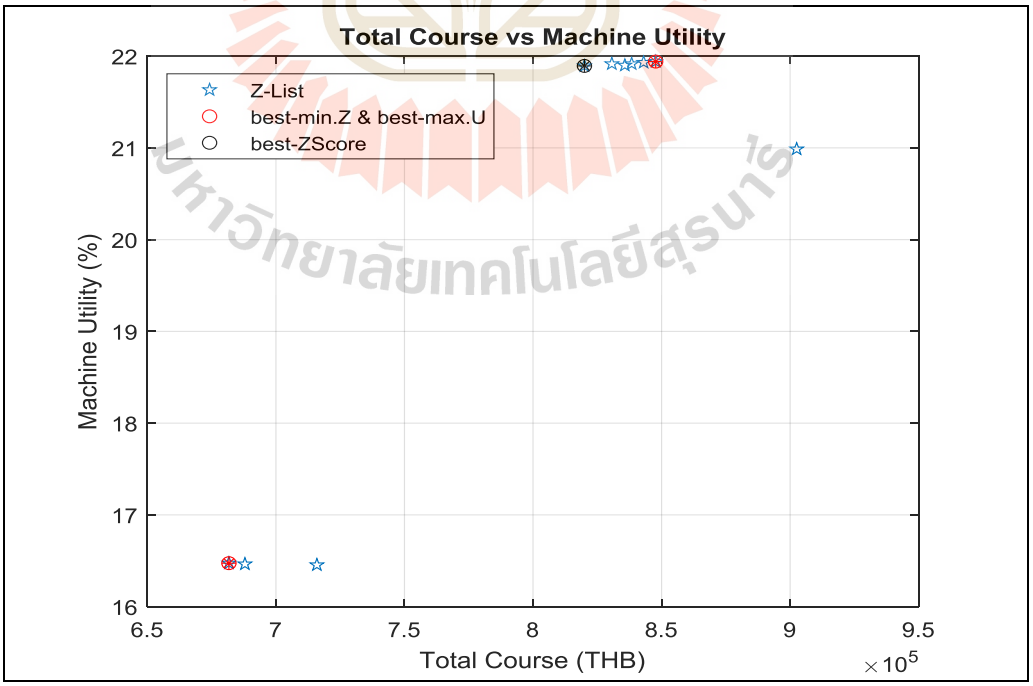
- ใช้วิธี ANS ในการค้นหาคำตอบ
- ทดสอบกับข้อมูล 12 ชุด ประกอบด้วย
- Test01 - 06J02M03 - Reference Problem Data
- Test02 - 06J02M06P - Reference Problem Data
- Test03,04,05,06 - 05J02M06P - Small Problem Data > จำนวน 4 ชุดข้อมูล {A,B,C,D}
- Test07,08,09,10 - 06J02M09P - Small Problem Data > จำนวน 4 ชุดข้อมูล {A,B,C,D}
- Test11 - 07J03M12P - Small Problem Data
- Test12 - 08J03M09P - Small Problem Data
- แสดงกราฟผลการทำงาน เปรียบเทียบค่าเวลาที่ใช้คำนวณ และผลการคำนวณ

มหาวิทยาลัยเทคโนโลยีสุรนารี

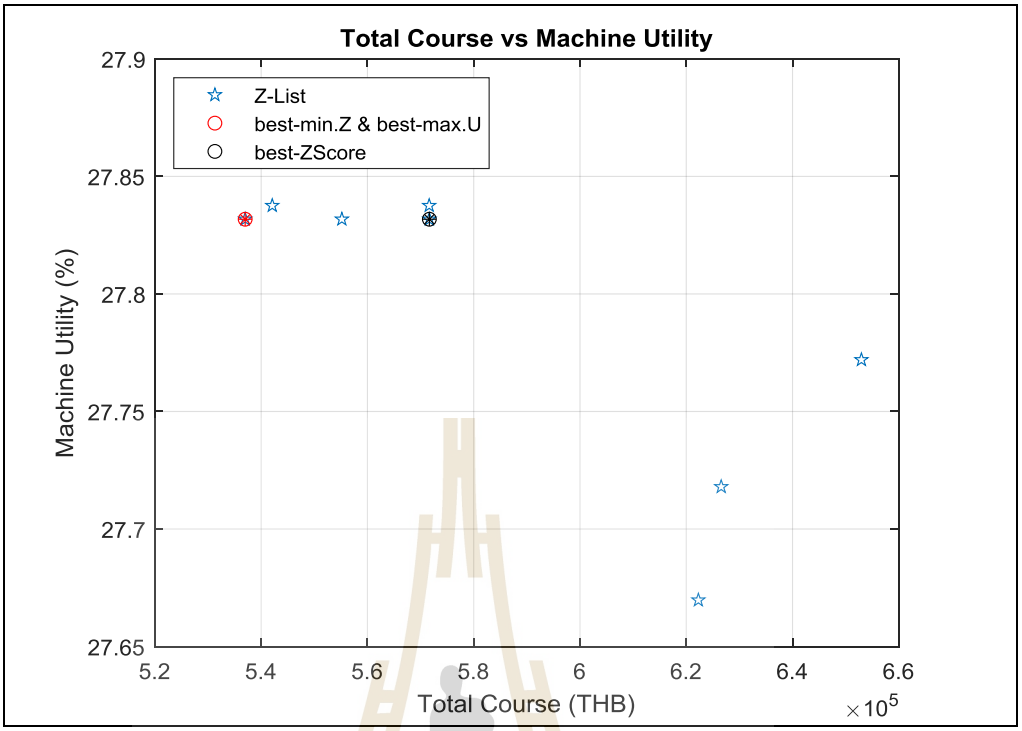
กราฟผลการทดสอบ



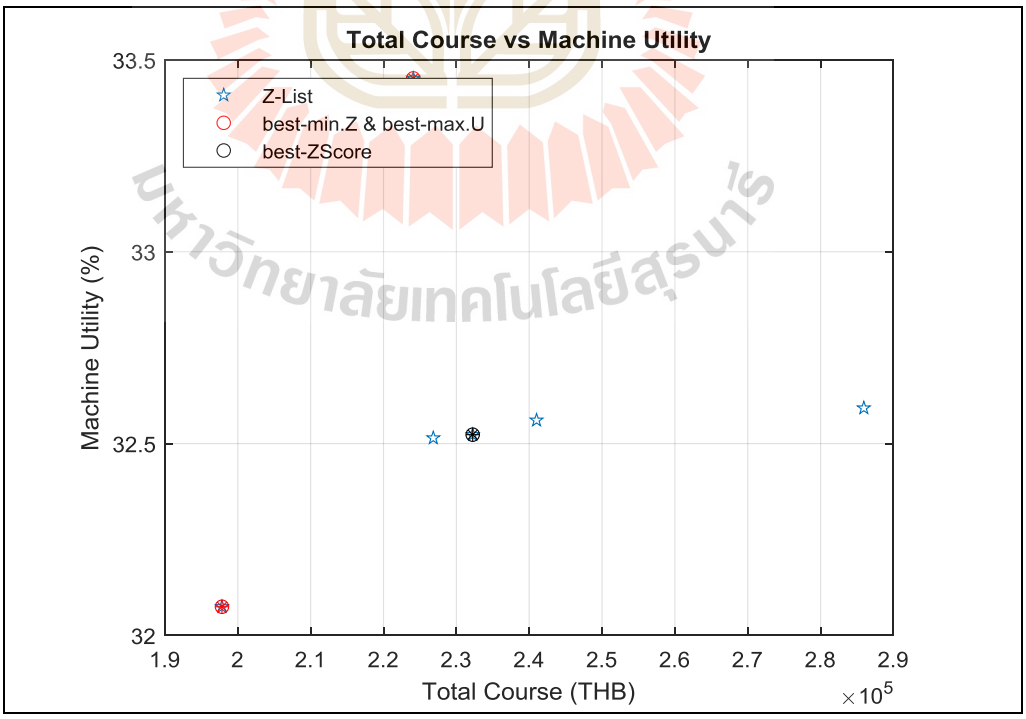
รูปที่ 5.6 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective



รูปที่ 5.7 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective

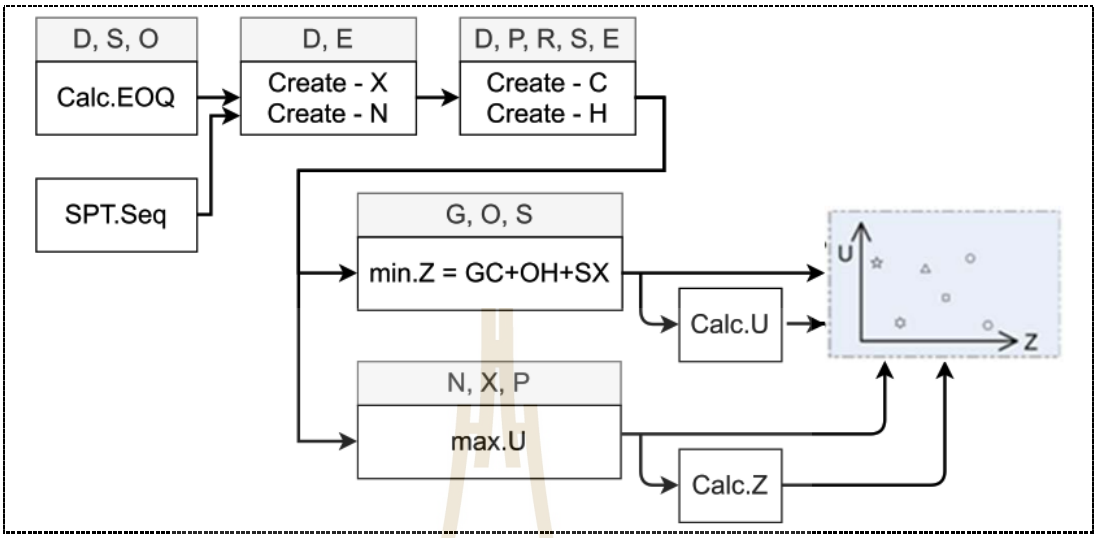


รูปที่ 5.8 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective



รูปที่ 5.9 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Pseudo Bi-Objective

5.2.3 Model-2: การจัดตารางงานแบบสองวัตถุประสงค์แบบขนาน

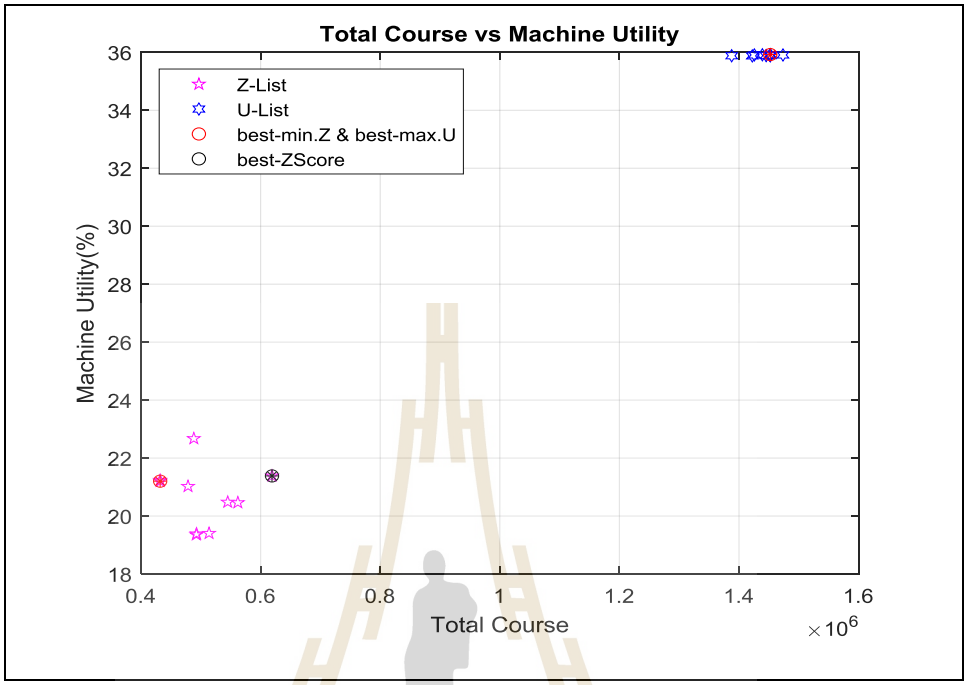


รูปที่ 5.10 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์แบบสองวัตถุประสงค์ขนาน

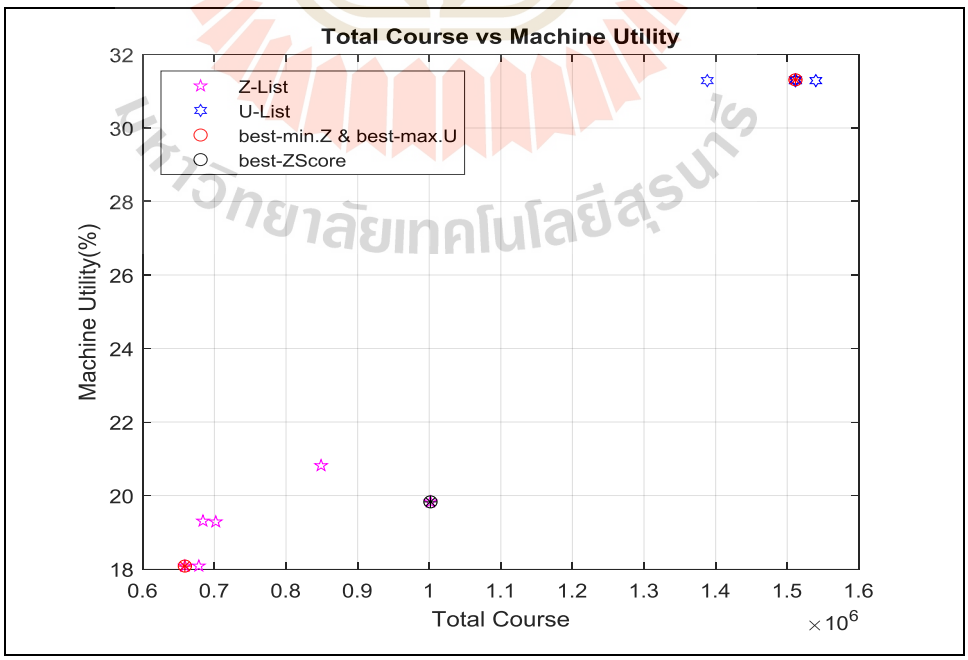
ขั้นตอนการทำงาน เงื่อนไขและการทดสอบ

1. คำนวณฟังก์ชัน Utilization และ min.Z โดยเลือกคำตอบ
   a. หาก U ติด TOP 10 ของ U-List ให้เพิ่มไว้ในลิสต์คำตอบ หรือ
   b. หาก Z ติด TOP 10 ของ Z-List ก็ให้เพิ่มไว้ในลิสต์คำตอบ เช่นกัน
   c. ใช้ U-List และ Z-List ในการเลือกแผนการผลิต
2. คำนวณหา ขนาดของคะแนนมาตรฐาน (z-score) ของ Z_List และ U_List
3. คำนวณหาค่า $ZU_{z\text{-score}}$
4. เลือกคำตอบจากเซ็ตคำตอบสามค่าจาก Z_List และ U_List
   (1) best.Z แผนการผลิตที่ให้ค่าใช้จ่ายในการผลิตรวมต่ำสุด
   (2) best.U แผนการผลิตที่ให้ค่าอัตราการใช้สอยทรัพยากรสูงสุด
   (3) best.ZU แผนการผลิตที่อยู่ระหว่างค่าใช้จ่ายต่ำและใช้ทรัพยากรเหมาะสม
5. ใช้วิธี ANS ในการค้นหาคำตอบ
6. ทดสอบการทำงานกับข้อมูลชุดเดิม และกำหนดจุดเริ่มต้นการค้นคำตอบเป็นจุดเดียวกัน
7. แสดงกราฟผลการทำงาน
8. เปรียบเทียบค่าเวลาที่ใช้คำนวณ และผลการคำนวณ

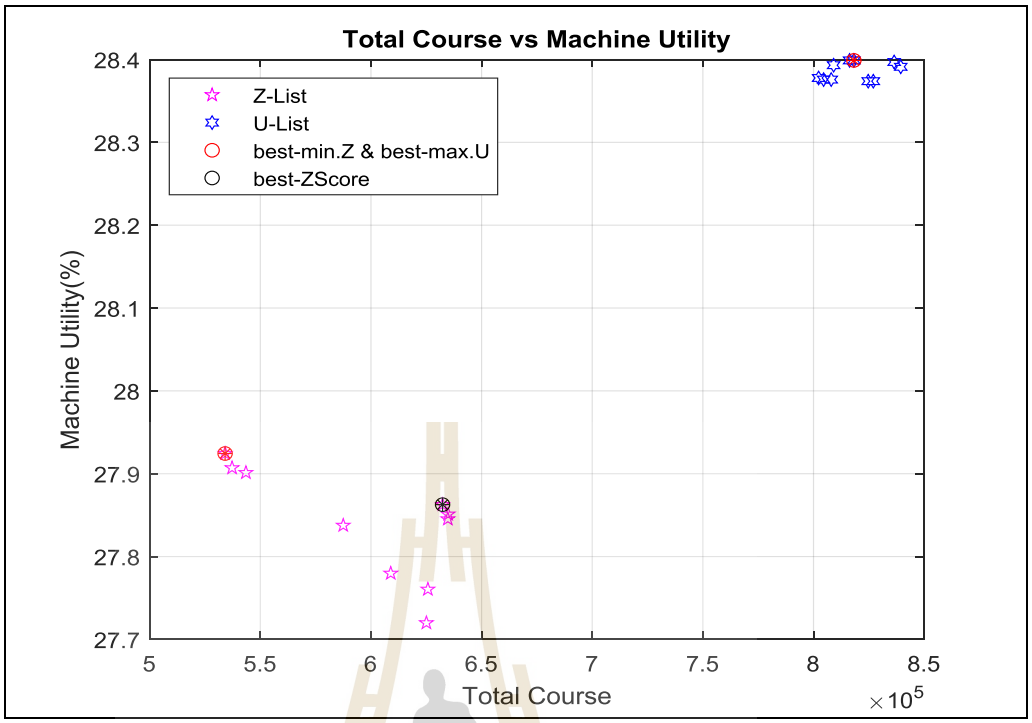
กราฟผลการทดสอบ



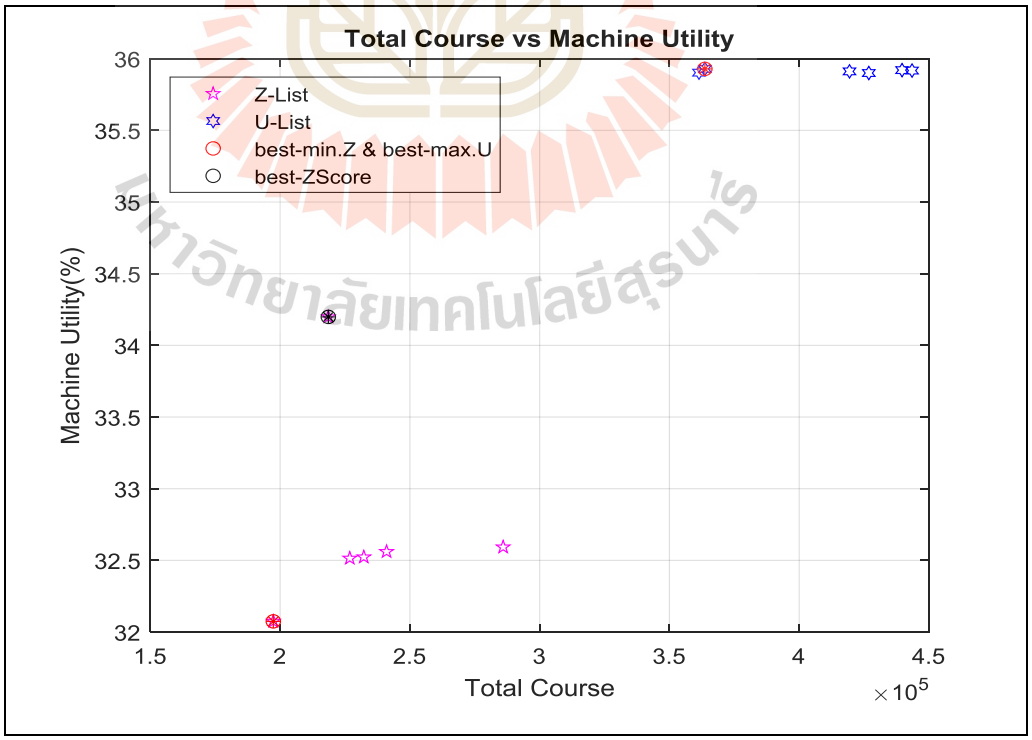
รูปที่ 5.11 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective



รูปที่ 5.12 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective

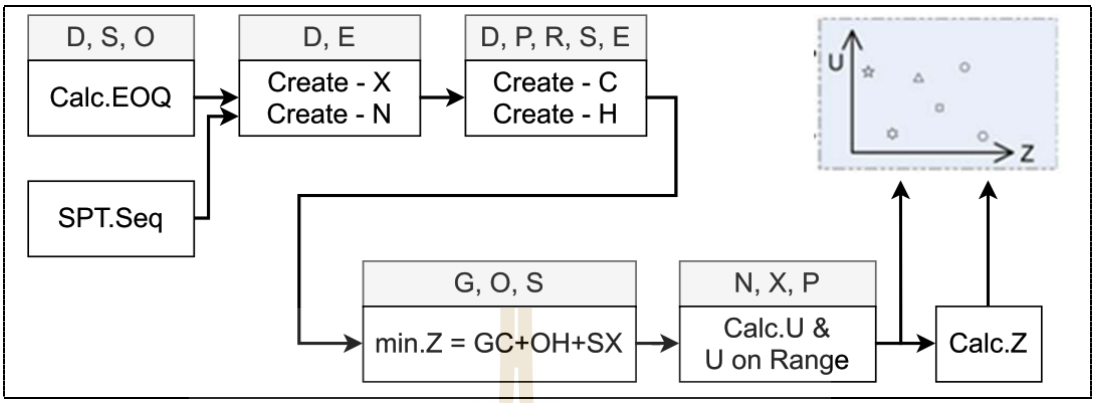


รูปที่ 5.13 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective



รูปที่ 5.14 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Parallel Bi-Objective

5.2.4 Model-3: การจัดตารางงานแบบสองวัตถุประสงค์แบบอนุกรม

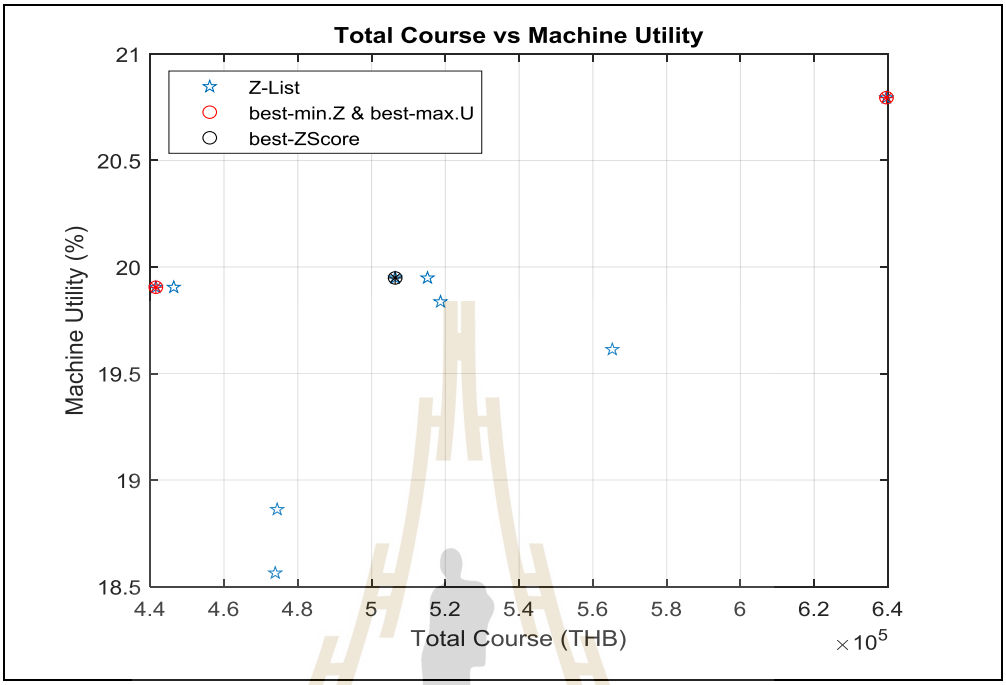


รูปที่ 5.15 แผนภาพขั้นตอนการคำนวณ เพื่อหาค่าของฟังก์ชันวัตถุประสงค์ แบบสองวัตถุประสงค์อนุกรม

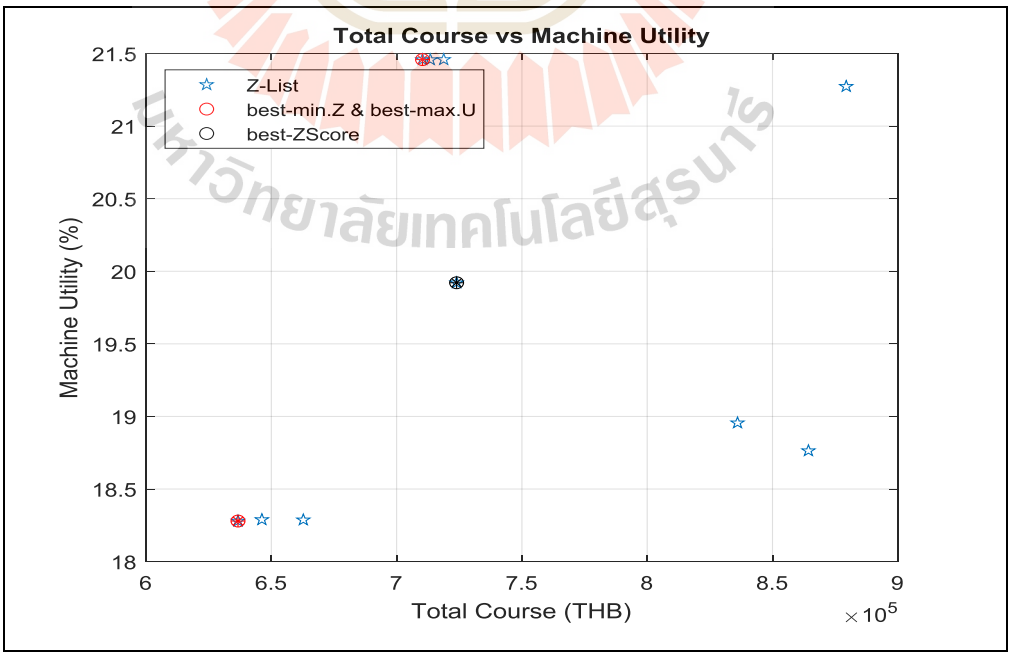
เงื่อนไขและขั้นตอนการทดสอบ

1. กำหนดค่าที่ยอมรับได้ของวัตถุประสงค์สอง
2. มีการคำนวณฟังก์ชัน min.Z หากคำตอบดีกว่าค่าแย่สุดใน Zs-List
   a. ให้คำนวณ ค่าคำตอบของค่าวัตถุประสงค์สอง
   b. หากอยู่ในค่าที่ยอมรับได้ให้บันทึกใน ZU-List
3. จาก Zs_List คำนวณฟังก์ชันวัตถุประสงค์สอง { ค่า Utilization } จะได้ค่า Us_List
4. คำนวณหา ขนาดของคะแนนมาตรฐาน (z-score) ของ Zs_List และ Us_List
5. คำนวณหาค่า $ZU_{z\text{-score}}$
6. เลือกคำตอบจากเซ็ตคำตอบสามค่าจาก Z_List และ U_List
   (1) best.Z แผนการผลิตที่ให้ค่าใช้จ่ายในการผลิตรวมต่ำสุด
   (2) best.U แผนการผลิตที่ให้ค่าอัตราการใช้สอยทรัพยากรสูงสุด
   (3) best.ZU แผนการผลิตที่อยู่ระหว่างค่าใช้จ่ายต่ำและใช้ทรัพยากรเหมาะสม
7. ใช้วิธี ANS ในการค้นหาคำตอบ
8. ทดสอบการทำงานกับข้อมูลชุดเดิม และกำหนดจุดเริ่มต้นการค้นคำตอบเป็นจุดเดียวกัน
9. แสดงกราฟผลการทำงาน
10. เปรียบเทียบค่าเวลาที่ใช้คำนวณ และผลการคำนวณ

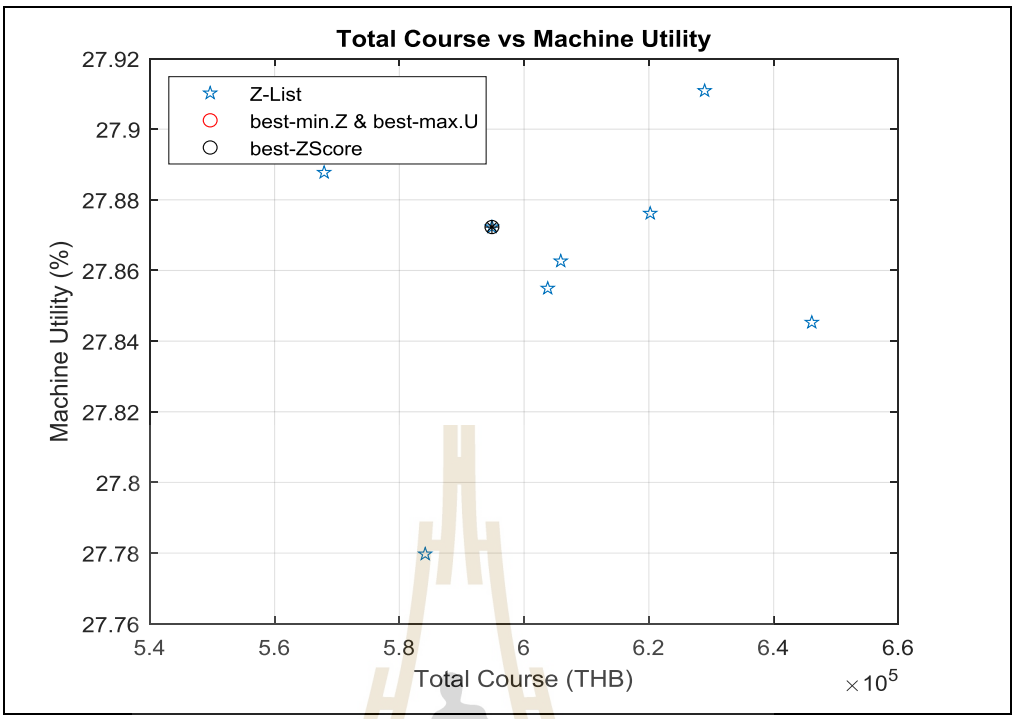
กราฟผลการทดสอบ



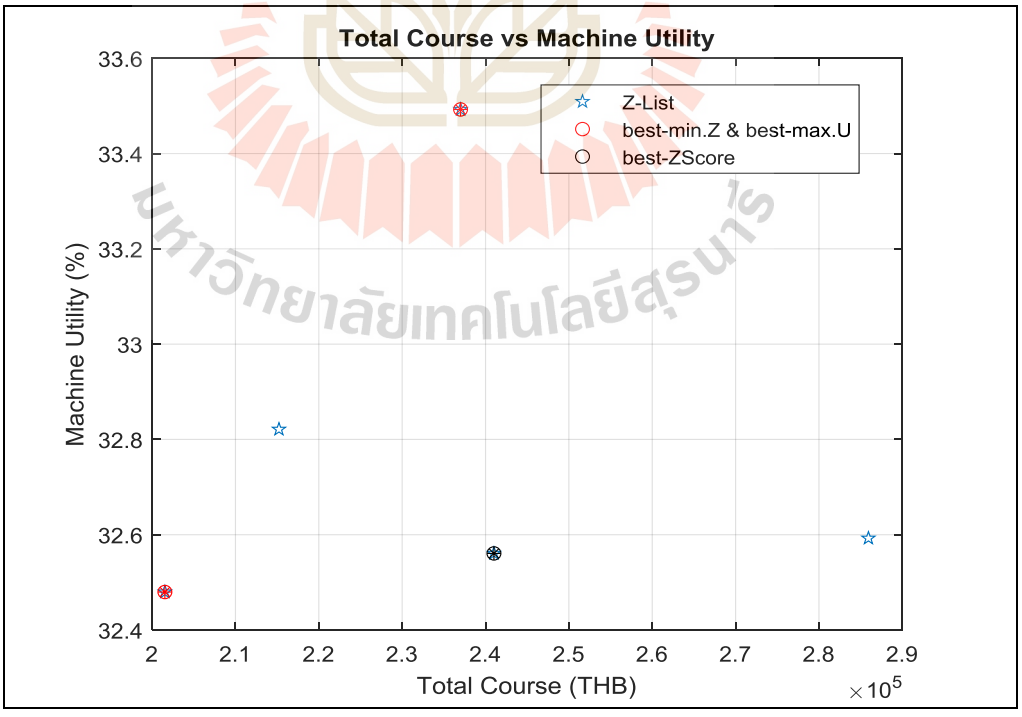
รูปที่ 5.16 ผลการทดสอบงาน Test12-08J03M09P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective



รูปที่ 5.17 ผลการทดสอบงาน Test11-07J03M12P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective



รูปที่ 5.18 ผลการทดสอบงาน Test10-06J02M09P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective



รูปที่ 5.19 ผลการทดสอบงาน Test02-06J02M06P เมื่อหาคำตอบด้วยวิธี Series Bi-Objective

5.3 ผลการทดสอบ

ตารางที่ 5.3 ผลการทดสอบ Pseudo Bi-Objective

Item	Case	Time(s)	iTeration	bestZ.Z	bestZ.U	bestU.Z	bestU.U	zscore.Z	zscore.U
1	Test-01	79	3900	13918	5.527	17454	5.932	15707	5.671
2	Test-02	155	2600	197825	32.075	224087	33.452	232242	32.523
3	Test-03	219	3900	320474	26.030	320474	26.030	337546	26.001
4	Test-04	253	4600	248686	21.730	259082	21.753	288851	21.522
5	Test-05	155	2800	343518	33.504	343518	33.504	362367	33.368
6	Test-06	198	3600	332802	26.421	344191	26.817	394300	26.649
7	Test-07	356	2900	650897	29.016	663750	29.020	674227	28.956
8	Test-08	378	3100	703393	33.542	758969	33.561	757912	33.493
9	Test-09	512	4500	641629	35.598	743343	35.912	666615	35.729
10	Test-10	570	4700	536989	27.832	548007	27.882	571650	27.832
11	Test-11	1021	2900	681715	16.474	847715	21.941	819998	21.893
12	Test-12	940	4000	410532	17.481	606558	26.442	478666	18.834

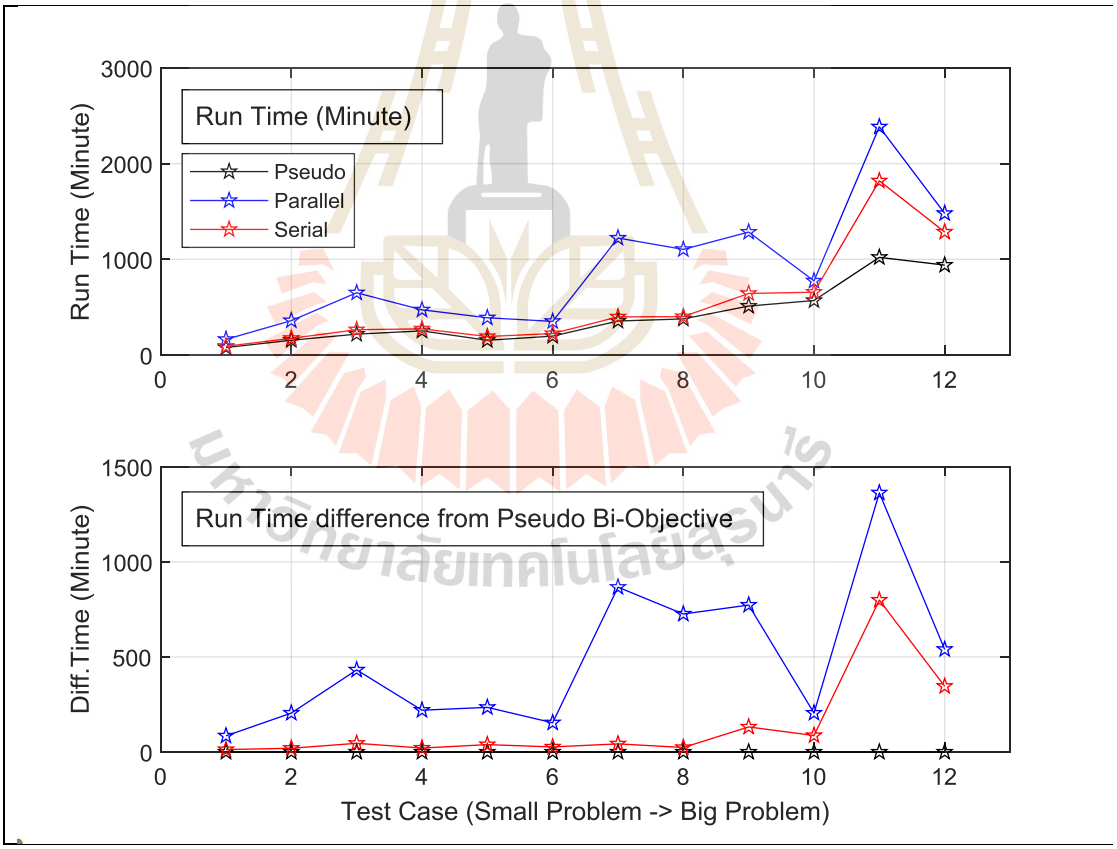
ตารางที่ 5.4 ผลการทดสอบ Parallel Bi-Objective

Item	Case	Time(s)	iTeration	bestZ.Z	bestZ.U	bestU.Z	bestU.U	zscore.Z	zscore.U
1	Test-01	164	3400	13666	5.642	26387	6.539	16990	5.804
2	Test-02	360	2600	197369	32.075	363660	35.929	218571	34.198
3	Test-03	651	5000	319753	26.010	394570	26.372	366738	26.027
4	Test-04	473	3700	253637	21.554	370607	22.075	334794	21.803
5	Test-05	390	3000	346026	33.368	448149	33.843	407532	33.802
6	Test-06	352	2700	331252	26.455	470017	27.083	412340	26.256
7	Test-07	1223	4400	621555	28.964	809395	29.348	770306	29.334
8	Test-08	1104	4000	689307	33.482	932027	34.157	797522	33.785
9	Test-09	1285	5000	618921	35.581	937506	36.555	729740	36.071
10	Test-10	775	2800	534089	27.924	818252	28.399	632367	27.863
11	Test-11	2384	3000	658700	18.081	1511478	31.311	1001678	19.827
12	Test-12	1480	2800	431981	21.202	1452250	35.922	619032	21.384

มหาวิทยาลัยเทคโนโลยีสุรนารี

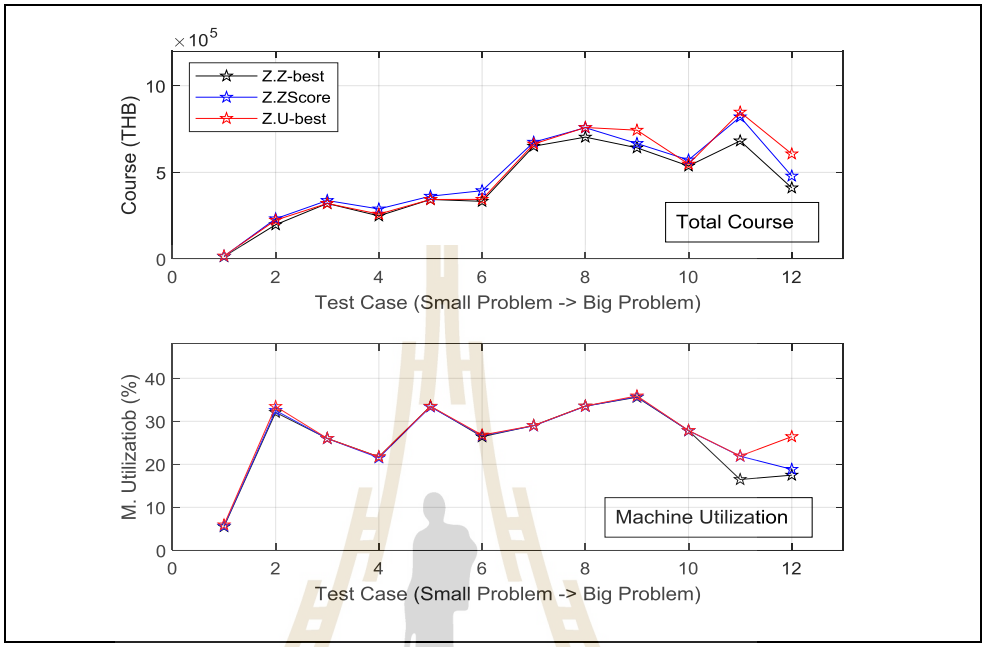
ตารางที่ 5.5 ผลการทดสอบ Serial Bi-Objective

Item	Case	LowerU	Time(s)	iTeration	bestZ.Z	bestZ.U	bestU.Z	bestU.U	zscore.Z	zscore.U
1	Test-01	5.6	92	4100	13672	5.613	14989	5.851	15138	5.712
2	Test-02	30	175	2600	201534	32.480	237007	33.492	241010	32.561
3	Test-03	25	265	4200	322751	26.030	370817	26.273	343923	26.010
4	Test-04	20	274	4400	249884	21.736	252402	21.759	268810	21.554
5	Test-05	30	194	3100	344761	33.521	370388	33.605	374095	33.568
6	Test-06	25	225	2800	330984	26.505	359521	26.742	387325	26.505
7	Test-07	25	399	2900	633264	28.835	746941	29.331	695419	28.881
8	Test-08	30	402	2900	694500	33.407	883188	34.066	736583	33.476
9	Test-09	30	644	5000	641427	35.559	750637	36.119	651342	35.779
10	Test-10	25	657	3300	548158	27.911	548158	27.911	594844	27.872
11	Test-11	18	1820	4600	696695	18.279	710320	21.458	723937	19.921
12	Test-12	18	1286	5000	441523	19.905	639568	20.795	506397	19.949

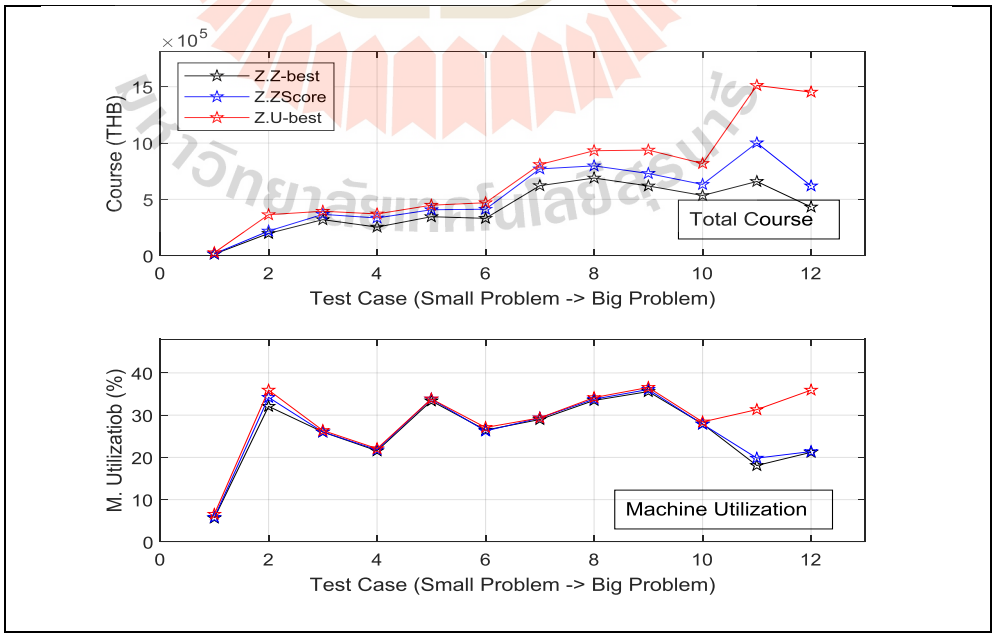


รูปที่ 5.20 เวลาที่ใช้ในการคำนวณ

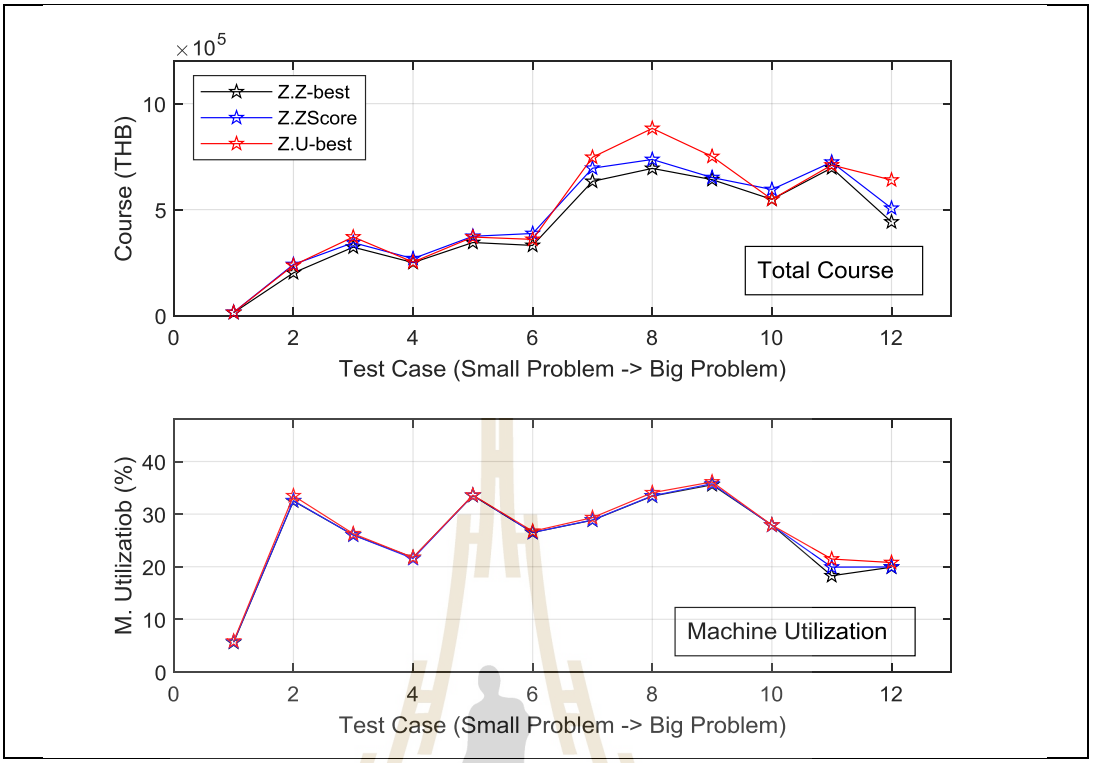
จากรูปที่ 5.20 เวลาในการทำงานของการค้นหาแบบขนานและแบบอนุกรมใช้เวลามากกว่าแบบเทียม โดยค้นหาแบบขนานใช้เวลามากสุด เนื่องมาจากจำนวนงานในรอบการทำงานมีมากกว่า



รูปที่ 5.21 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Pseudo Bi-Objective



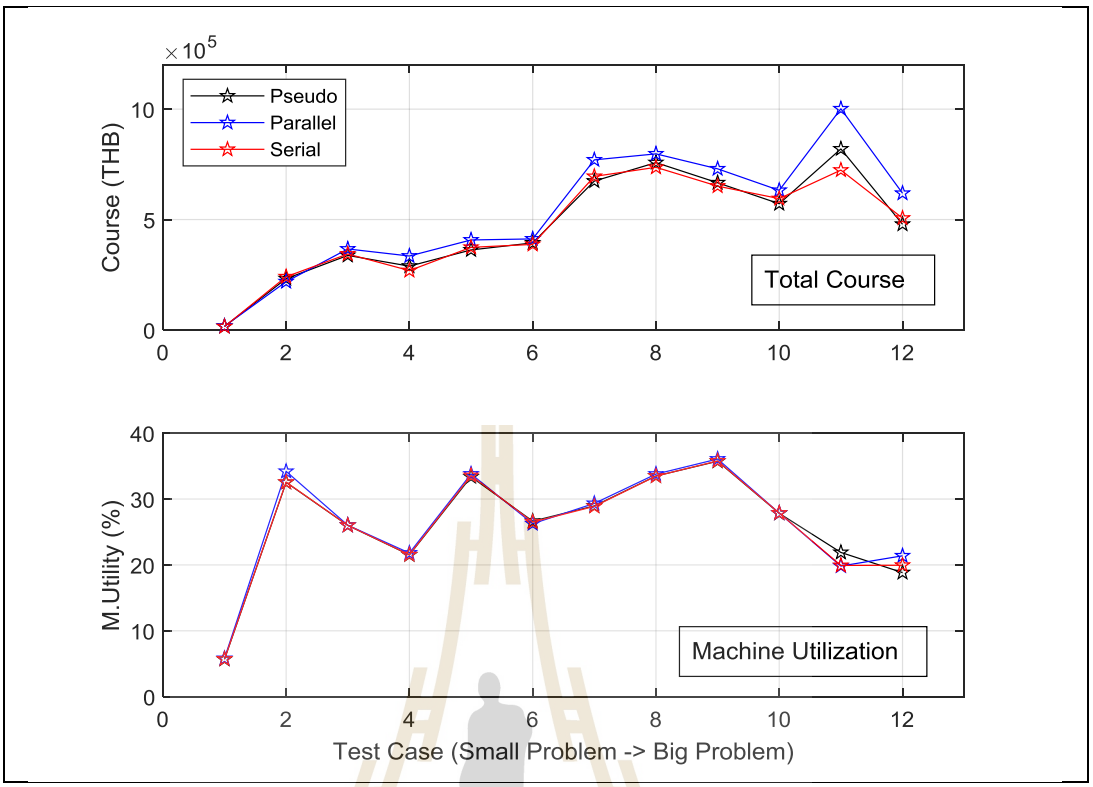
รูปที่ 5.22 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Parallel Bi-Objective



รูปที่ 5.23 ผลการคำนวณค่า Z-best, U-best, ZU-ZScore กรณี Serial Bi-Objective

รูปที่ 5.21-5.23 แสดงผลการคำนวณค่า Z และ ค่า U แต่ละรูปจะมีข้อมูลจะมาจาก 3 ชุดคำตอบ คือ คำตอบมาจาก Z-best, U-best และ ZScore ผลการทดสอบเห็นได้ว่า

- ในทุกรูปจุดทดสอบจุดที่หนึ่งให้คำตอบที่ได้จาก Z-best หรือ U-best ใกล้เคียงกันมากเนื่องจากขนาดของปัญหาเล็กทำให้คำตอบของปัญหามีจำกัด
- ในทุกรูปจุดทดสอบที่มีขนาดปัญหาใหญ่ขึ้นคำตอบที่ได้จาก Z-best หรือ U-best ให้คำตอบห่างกันมากขึ้น
- คำตอบที่เลือกจาก z-score เป็นคำตอบที่อยู่ระหว่างค่า Z-best และ U-best ในรูปที่ 5.22 เห็นได้ชัดเจน แต่บางตำแหน่งของรูปที่ 5.23 คำตอบ z-score อยู่นอกจากขอบของ Z-best และ U-best ซึ่งเกิดขึ้นได้ ดังในรูปที่ 5.13 คำตอบ ZU-best ที่ได้จาก z-score อยู่ต่ำกว่าคำตอบของ Z-best และ U-best



รูปที่ 5.24 เปรียบเทียบคำตอบ z-score ของทั้งสามวิธีค้นคำตอบ

รูปที่ 5.24 แสดงคำตอบที่เลือกจาก z-score ของวิธีค้นหาคำตอบทั้งสามวิธี กราฟรูปบนเป็นวัตถุประสงค์แรก-ค่าใช้จ่ายในการผลิตต่ำสุด กราฟรูปล่างเป็นวัตถุประสงค์สอง-การใช้งานเครื่องจักรเต็มกำลัง ผลการทำงานสรุปได้ว่า

- วิธีการค้นคำตอบทั้งสาม ให้คำตอบใกล้เคียงกัน และมีแนวโน้มว่าการค้นแบบขนานจะให้คำตอบที่แย่กว่าเพราะต้องคำนึงถึงวัตถุประสงค์สองมากกว่าวิธีอื่น
- คำตอบที่ได้จาก Z-Best และ U-best เป็นไปในทิศทางเดียวกัน กล่าวคือ เมื่อผลการค้นคำตอบได้ค่า Z ต่ำสุดแล้วจะส่งผลให้ค่า U สูงสุดไปด้วย ด้วยเหตุนี้ส่งผลให้คำตอบที่ได้จากการทำงานแบบสองวัตถุประสงค์ไม่เห็นข้อแตกต่างมากนัก แต่อย่างไรก็ดี การนำเสนอวิธีแบบสองวัตถุประสงค์นี้ เพื่อเป็นตัวอย่างการค้นคำตอบและนำไปใช้ในงานอื่นต่อไป

5.4 สรุป

การค้นคำตอบแบบสองวัตถุประสงค์ในบทนี้ใช้เพื่อหาคำตอบของการจัดตารางงานโดยกำหนดวัตถุประสงค์แรกคือค่าใช้จ่ายในการผลิตต่ำสุดและสองการใช้งานเครื่องจักรเต็มประสิทธิภาพ การค้นคำตอบแบ่งเป็นสามวิธี ได้แก่ การทำงานแบบเทียม แบบขนานและแบบอนุกรม ทดสอบทั้งสามวิธีด้วยข้อมูลชุดเดียวกันจำนวน 12 ชุด และกำหนดจุดเริ่มต้นการค้นคำตอบจุดเดียวกัน คำตอบที่ได้จะมีสองค่าตามจำนวนวัตถุประสงค์การรวมคำตอบทั้งสองค่าทำโดยการใช้ค่าคะแนนมาตรฐานของคำตอบทั้งสองมาพิจารณาโดยให้ความสำคัญเท่ากันและเลือกคำตอบที่ให้ค่ากลางที่อยู่ระหว่างกลุ่มคำตอบทั้งหมด

ผลการทดสอบทั้งสามวิธี ประเด็นค่าคำตอบทั้งสามวิธีสามารถค้นคำตอบได้ใกล้เคียงกันในปัญหาขนาดเล็กแต่ปัญหาขนาดใหญ่คำตอบที่ได้ก็แตกต่างกันมากขึ้น ประเด็นเวลาในการค้นคำตอบการทำงานแบบขนานใช้เวลามากสุด รองลงมาคือแบบอนุกรม และสุดท้ายคือแบบเทียม สาเหตุคือจำนวนงานที่คำนวณในโปรแกรมที่มากกว่า ประเด็นสุดท้ายคือปัญหาสองวัตถุประสงค์ที่ใช้ทดสอบวัตถุประสงค์ทั้งสองเป็นไปในทางเดียวกันคือเมื่อค่าใช้จ่ายการผลิตต่ำการใช้งานเครื่องจักรก็ใช้เต็มประสิทธิภาพด้วยทำให้แบบจำลองการค้นหาทั้งสามวิธีเห็นผลไม่แต่ต่างเท่าที่ควร เพื่อทดสอบการค้นคำตอบทั้งสามวิธีอาจต้องหาวัตถุประสงค์ที่ไม่เป็นไปตามกันในการทดสอบ

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 6
สรุปและข้อเสนอแนะ

6.1 สรุป

จากผลการทำงานเพื่อจัดตารางการผลิต ภายใต้เงื่อนไขต่าง ๆ ดังนี้ เครื่องจักรมีการจัดเรียงแบบขนาน, ลำดับการผลิตมีผลต่อเวลาติดตั้งเครื่องจักร, พิจารณาขนาดการสั่งผลิตขั้นต่ำ และมีข้อจำกัดด้านการทำงานของเครื่องจักร คือ งานบางงานไม่สามารถผลิตบนเครื่องจักรบางเครื่องได้ โดยวัตถุประสงค์เพื่อทำให้ค่าใช้จ่ายและเวลาในการผลิตรวมของระบบ (Makespan) มีค่าต่ำที่สุด โดยใช้ integer programming สรุปได้เป็นข้อ ๆ ดังนี้

1. การทำงานของ Math:แบบจำลองปัญหาทางคณิตศาสตร์ ทำงานได้ถูกต้องตามวัตถุประสงค์ที่กำหนดไว้ โดยคำตอบที่ได้ คือ จำนวนสั่งผลิต และลำดับงานในการผลิต

2. แบบจำลองด้วยวิธีปัญญาเชิงคำนวนให้ผลการทำงาน ดังนี้

2.1 พัฒนาวิธีปัญญาเชิงคำนวนด้วยวิธี ANS:การค้นหาเพื่อนบ้านใกล้สุดเชิงปรับตัว และ MTS:การค้นหาแบบตาบูประยุกต์

2.2 ทดสอบกับปัญหาอ้างอิง ทั้ง ANS และ MTS ให้ผลการคำนวณแย่กว่า Math 3-5% และสามารถลดเวลาการค้นคำตอบจากหนึ่งร้อยชั่วโมงเป็นหนึ่งนาที

2.3 ทดสอบกับปัญหาขนาดเล็ก ทั้ง ANS และ MTS ให้คำตอบใกล้เคียงกัน และใช้เวลาใกล้เคียงกันด้วย คำตอบเมื่อเทียบกับ Math จะแย่กว่าไม่เกิน 10%

2.4 ทดสอบกับปัญหาขนาดใหญ่ ANS และ MTS ให้คำตอบและใช้เวลาใกล้เคียงกัน

3. การจัดตารางแบบหลายวัตถุประสงค์ ได้นำเสนอรูปแบบ ขั้นตอนการทำงาน ของวิธีปัญญาเชิงคำนวณ โดยเสนอไว้ 3 แบบ คือ การทำงานแบบเทียม, แบบขนาน และแบบอนุกรม ทั้งสามแบบให้คำตอบใกล้เคียงกัน ซึ่งเป็นผลจากการเลือกวัตถุประสงค์หลักและวัตถุประสงค์รองที่ส่งผลต่อคำตอบไปในทางเดียวกัน

6.2 ข้อเสนอแนะ

1. ยังไม่มีการทดสอบกับงานในการผลิตจริง

2. แนะนำให้ใช้โครงข่ายประสาทเทียม (Neural Network) ในการค้นหาคำตอบ ซึ่งวิธีนี้เป็นหนึ่งในวิธีการของปัญญาเชิงคำนวณ

รายการอ้างอิง

ปารเมศ ชุติมา (2555). **เทคนิคการจัดตารางการดำเนินงาน**. จุฬาลงกรณ์มหาวิทยาลัย.

อาทิตย์ ศรีแก้ว (2558). **ปัญญาเชิงคำนวณ**. มหาวิทยาลัยเทคโนโลยีสุรนารี. พิมพ์ครั้งที่ 1.

Adamopoulos GI, Pappis CP. (1998). Scheduling under a common due date on parallel unrelated machines. **Eur J Oper Res**. 105: 494-501.

Baker K. (1995). **Elements of sequencing and scheduling**. Hanover, NH:Amos Tuck School of Business Administration Dartmouth College.

Baker KR, Bertrand JW. (1983). A dynamic priority rule for sequencing against due dates. **J Oper Manage**. 3: 37-42.

Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., Weglarz, J. (1996). **Scheduling computers and manufacturing processes**. Berlin: Springer.

Blum C., Roli A. (2003). **Metaheuristics in combinatorial optimization: Overview and conceptual comparison**. ACM Comput Surv. 35(3): 268-308.

Centeno G., Armacost R. L. (1997). **Parallel machine scheduling with release time and machine eligibility restrictions**. Computers & Industrial Engineering. 33(1): 273-6.

Cheng TCE, Sin CCS. (1990). A State-of-the-art review of parallel machine scheduling research. **Eur J Oper Res**. 77: 271-92.

Chengen W .Chengbin C., Proth J-M. (1997). Heuristic approaches for n/m/F/ $\sum$Ci scheduling problems. **European Journal of Operational Research**. 96(3): 636-44.

Drexl, A., & Kimms, A. (1997). Lot sizing and scheduling-Survey and extensions. **European Journal of Operational Research**. 99: 221-235.

Fleischmann, B. (1990). The discrete lot-sizing and scheduling problem. **European Journal of Operational Research**. 44: 337-348.

Ho JC, Chang YL. (1995). Minimizing the number of tardy jobs for m-parallel machines. **Eur J Oper Res**. 84: 343-55.

Jozefowska J, Mika M, Rozycki R, Waligora G, Weglarz J. (1998). Local search meta-heuristics for discrete-continuous scheduling problems. Eur J Oper Res. 107: 354-70.

Karp RM. (1972). **Reducibility among combinatorial problems: complexity of computer computations**. New York: Plenum Press. 85-103.

Khouja M., Michalewicz Z., Wilmot M. (1998). The use of genetic algorithms to solve the economic lot size scheduling problem. **European Journal of Operational Research**. 110(3): 509-24.

Kim DW, Kim KH, Jang W, Chen FF. (2002). **Unrelated parallel machine scheduling with setup times using simulated annealing.Robotics Computer Integr Manuf**. 18: 223-31.

Lee H, Guignard M. (1996). Hybrid bounding procedure for the workload allocation problem on parallel unrelated machines with setups. **J Oper Res Soc**. 47: 1247-61.

Lee Y. H., Pinedo M. (1997). Scheduling jobs on parallel machines with sequence-dependent setup times. **European Journal of Operational Research**. 100(3): 464-74.

Liaee MM, Emmons H. (1997). Scheduling families of jobs with setup times. **Int J Prod Econom.** 51: 165-76.

Pattloch, M., Schmidt, G., & Kovalyov, M. Y. (2001). **Heuristic algorithms for lot size scheduling with application in the tobacco industry**. Computers and Industrial Engineering. 39: 235-253.

Potts CN, Kovalyov MY. (2000). Scheduling with batching: a review. **Eur J Oper Res**. 120: 228-49.

Potts CN, Van Wassenhove L. (1982). **Decomposition algorithm for the single machine total tardiness problem**. Oper Res Lett. 1:177-81.

Rajakumar S., Arunachalam V. P., Selladurai V. (2006). Workflow balancing in parallel machine scheduling with precedence constraints using genetic algorithm. **Journal of Manufacturing Technology Management**. 17(2): 239-54.

Salomon, M., Kroon, L. G., Kuik, R., & Van Wassenhove, L. N. (1991). **Some extensions of the discrete lotsizing and scheduling problem**. Management Science. 37: 801-812.

Slowinski R. (1988). **Production scheduling on parallel machines subject to staircase demands**. Eng Costs Prod Econom. 14: 11-7.

Staggemeier, A.T., Clark, A.R. (2001). **A survey of lot sizing and scheduling models**. Proceedings of the 23rd Annual Symposium of the Brazilian Operational Research Society (SOBRAPO), Brazil. 938-947.

Süer G. A., Pico F., Santiago A. (1997). **Identical machine scheduling to minimize the number of tardy jobs when lot-splitting is allowed**. Computers & Industrial Engineering. 33(1): 277-80.

Suresh V, Chaudhuri D. (1996). **Bicriteria scheduling problem for unrelated parallel machines**. Comput Ind Eng. 30: 77-82.

ภาคผนวก ก

ผลการทดสอบของโปรแกรม

ตารางภาคผนวก ก.1 ข้อมูลผลการทดสอบปัญหาอ้างอิง

Test	J-M-P	Run	Objective Value			Run Time		
			Math	ANS	MTS	Math	ANS	MTS
1	6J-2M-3P	Run-1	16981	17770	18006	0d-00:00:38	33.5	33.5
		Run-2	16981	17562	17541	0d-00:00:31	30.5	34.2
		Run-3	16981	18634	17918	0d-00:00:29	29.5	32.3
		Run-4	16981	17074	17918	0d-00:00:32	34.5	31.9
		Run-5	16981	17234	17986	0d-00:00:28	33.5	32.0
		Run-6	16981	17394	17034	0d-00:00:40	36.5	32.0
		Average	16981.00	17611.33	17733.83	0d-00:00:33	33.00	32.65
	Diff. from Math		##	**-630.33**	**-752.83**	##	##	##
	Diff. from Math (%)		##	**-3.71**	**-4.43**	##	##	##
2	9J-2M-3P	Run-1	74536	75834	76878	4d-04:43:53	35.1	33.5
		Run-2	74536	78945	76696	4d-05:23:12	34.0	132.0
		Run-3	74536	75912	74952	4d-06:23:56	34.0	41.0
		Run-4	74536	75289	80352	4d-04:52:12	34.0	35.9
		Run-5	74536	79981	77408	4d-05:14:42	41.0	32.2
		Run-6	74536	79531	75395	4d-04:06:26	41.0	32.1
		Average	74536.00	77582.00	76946.83	4d-05:14:03	36.52	51.12
	Diff. from Math		##	**-3046.00**	**-2410.83**	##	##	##
	Diff. from Math (%)		##	**-4.09**	**-3.23**	##	##	##
3	6J-2M-6P	Run-1	257251	262467	282864	3d-23:23:59	38.5	36.7
		Run-2	257251	263675	271934	3d-22:05:57	37.4	36.5
		Run-3	257251	269307	259421	3d-22:08:18	38.1	37.2
		Run-4	257251	263458	258110	3d-21:40:40	37.2	37.2
		Run-5	257251	265133	282864	3d-22:07:33	39.2	36.5
		Run-6	257251	265858	264076	3d-23:50:41	37.3	36.7
		Average	257251.00	264983.00	269878.17	3d-22:24:33	37.95	36.80
	Diff. from Math		##	**-7732.00**	**-12627.17**	##	##	##
	Diff. from Math (%)		##	**-3.01**	**-4.91**	##	##	##

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.2 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก

			Objective Value			Diff. from Math (%)		Run Time(s)		
Item	Test	Case	Math	ANS	MTS	ANS	MTS	Math	ANS	MTS
1	test01_05J02M06P	test01_05J02M06P_A	283961	291226	289657	-2.56	-2.01	45	218	501
2		test01_05J02M06P_B	208709	215068	211382	-3.05	-1.28	1	344	565
3		test01_05J02M06P_C	287020	296454	295290	-3.29	-2.88	1	155	339
4		test01_05J02M06P_D	270952	280285	280290	-3.44	-3.45	56	331	538
5		test01_05J02M06P_E	250109	255359	265870	-2.10	-6.30	19	244	1264
6		test01_05J02M06P_F	268139	277842	275138	-3.62	-2.61	1	295	135
7	test02_05J02M09P	test02_05J02M09P_A	389196	406438	394170	-4.43	-1.28	9043	350	405
8		test02_05J02M09P_B	360780	379595	368005	-5.22	-2.00	1	579	703
9		test02_05J02M09P_C	305416	314214	329547	-2.88	-7.90	1	846	332
10		test02_05J02M09P_D	400724	428023	411474	-6.81	-2.68	1	718	327
11		test02_05J02M09P_E	275503	291635	289910	-5.86	-5.23	151	1157	931
12		test02_05J02M09P_F	369694	381511	394187	-3.20	-6.63	1	394	365
13	test03_06J02M06P	test03_06J02M06P_A	348475	359522	378123	-3.17	-8.51	6662	132	365
14		test03_06J02M06P_B	352289	370764	372442	-5.24	-5.72	23433	157	275
15		test03_06J02M06P_C	386257	402254	416315	-4.14	-7.78	2921	114	262
16		test03_06J02M06P_D	305123	319129	330833	-4.59	-8.43	633	137	176
17		test03_06J02M06P_E	290335	301559	318749	-3.87	-9.79	2563	198	574
18		test03_06J02M06P_F	317600	330642	348748	-4.11	-9.81	352	109	193
19	test04_06J02M09P	test04_06J02M09P_A	473486	499542	485151	-5.50	-2.46	421550	282	561
20		test04_06J02M09P_B	462935	471683	468989	-1.89	-1.31	60	384	825
21		test04_06J02M09P_C	464974	492445	490221	-5.91	-5.43	26385	342	532
22		test04_06J02M09P_D	415873	436005	424297	-4.84	-2.03	587845	522	693
23		test04_06J02M09P_E	316814	326055	347528	-2.92	-9.69	516630	716	867
24		test04_06J02M09P_F	535176	556339	583393	-3.95	-9.01	421520	615	528
25	test05_07J02M06P	test05_07J02M06P_A	376758	389532	397856	-3.39	-5.60	282	178	429
26		test05_07J02M06P_B	587573	619400	620003	-5.42	-5.52	601	273	553
27		test05_07J02M06P_C	336221	354892	345405	-5.55	-2.73	50	454	351
28		test05_07J02M06P_D	345116	357980	357337	-3.73	-3.54	10970	206	160
29		test05_07J02M06P_E	431801	452268	445125	-4.74	-3.09	586313	285	417
30		test05_07J02M06P_F	374082	394832	393364	-5.55	-5.15	11	346	607
31	test06_07J02M09P	test06_07J02M09P_A	574836	615677	585949	-7.10	-1.93	421646	610	803
32		test06_07J02M09P_B	548836	583732	570183	-6.36	-3.89	516520	905	1186
33		test06_07J02M09P_C	481712	506887	489750	-5.23	-1.67	263374	324	393
34		test06_07J02M09P_D	548045	554077	587352	-1.10	-7.17	421598	761	580
35		test06_07J02M09P_E	558472	582010	611769	-4.21	-9.54	179733	539	1559
36		test06_07J02M09P_F	573035	593298	581441	-3.54	-1.47	145115	484	385

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.2 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

			Objective Value			Diff. from Math (%		Run Time(s)		
Item	**Test**	**Case**	**Math**	**ANS**	**MTS**	**ANS**	**MTS**	**Math**	**ANS**	**MTS**
37	test07_07J03M09P	test07_07J03M09P_A	260944	264108	272214	-1.21	-4.32	451383	1430	1266
38		test07_07J03M09P_B	282378	293695	286315	-4.01	-1.39	451356	676	931
39		test07_07J03M09P_C	273769	278300	280042	-1.66	-2.29	586548	783	883
40		test07_07J03M09P_D	299168	316094	305507	-5.66	-2.12	451319	595	1618
41		test07_07J03M09P_E	320447	343365	333636	-7.15	-4.12	451297	553	396
42		test07_07J03M09P_F	310136	333142	324984	-7.42	-4.79	451278	155	346
43	test08_07J03M12P	test08_07J03M12P_A	348051	354368	353174	-1.81	-1.47	452857	2173	995
44		test08_07J03M12P_B	354469	373661	360120	-5.41	-1.59	452835	1149	1155
45		test08_07J03M12P_C	375724	380489	380779	-1.27	-1.35	452776	1684	374
46		test08_07J03M12P_D	364737	375239	371860	-2.88	-1.95	452752	1140	279
47		test08_07J03M12P_E	384071	391391	396783	-1.91	-3.31	452707	976	1048
48		test08_07J03M12P_F	407276	416744	414816	-2.32	-1.85	452575	2035	1685
49	test09_08J02M09P	test09_08J02M09P_A	55335	56728	56083	-2.52	-1.35	454278	460	1056
50		test09_08J02M09P_B	741255	793424	785422	-7.04	-5.96	454249	545	809
51		test09_08J02M09P_C	846268	873857	856738	-3.26	-1.24	454222	898	1692
52		test09_08J02M09P_D	825243	860278	869910	-4.25	-5.41	454200	864	501
53		test09_08J02M09P_E	627486	668440	654688	-6.53	-4.34	454179	912	1137
54		test09_08J02M09P_F	652353	701534	668374	-7.54	-2.46	454104	767	988
55	test10_08J03M09P	test10_08J03M09P_A	375425	386246	392108	-2.88	-4.44	585367	847	1110
56		test10_08J03M09P_B	289617	292369	294606	-0.95	-1.72	585286	1469	970
57		test10_08J03M09P_C	366792	394400	381736	-7.53	-4.07	585333	782	1493
58		test10_08J03M09P_D	287823	297615	293476	-3.40	-1.96	586733	990	930
59		test10_08J03M09P_E	335784	350510	347070	-4.39	-3.36	586714	1606	1836
60		test10_08J03M09P_F	273215	283610	286377	-3.80	-4.82	586701	1159	1016
					Min	**-0.95**	**-1.24**			
					Max	**-7.54**	**-9.81**			

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก

Item	File Name	Case	Mode	Objt	Time	Iter
1	RDTest21a_06J04M10P_20201014_1938.log	Test21a	ANN	254924	1130	4200
2	RDTest21a_06J04M10P_20201014_2147.log	Test21a	ANN	252108	1591	6000
3	RDTest21a_06J04M10P_20201015_0059.log	Test21a	ANN	237633	1142	4300
4	RDTest21a_06J04M10P_20201015_0539.log	Test21a	ANN	254924	1131	4200
5	MTSTest21a_06J04M10P_20201014_1957.log	Test21a	MTS	241824	2031	7200
6	MTSTest21a_06J04M10P_20201014_2214.log	Test21a	MTS	243298	1839	6500
7	MTSTest21a_06J04M10P_20201015_0118.log	Test21a	MTS	239270	1335	4700
8	MTSTest21a_06J04M10P_20201015_0558.log	Test21a	MTS	241824	2082	7200
9	RDTest21c_06J04M10P_20201014_2111.log	Test21c	ANN	265310	1171	4400
10	RDTest21c_06J04M10P_20201014_2351.log	Test21c	ANN	261594	977	3700
11	RDTest21c_06J04M10P_20201015_0239.log	Test21c	ANN	258889	2270	8500
12	RDTest21c_06J04M10P_20201015_0714.log	Test21c	ANN	265310	1176	4400
13	MTSTest21c_06J04M10P_20201014_2131.log	Test21c	MTS	*	*	*
14	MTSTest21c_06J04M10P_20201015_0007.log	Test21c	MTS	*	*	*
15	MTSTest21c_06J04M10P_20201015_0317.log	Test21c	MTS	*	*	*
16	RDTest21d_06J04M10P_20201014_2031.log	Test21d	ANN	286347	968	3600
17	RDTest21d_06J04M10P_20201014_2245.log	Test21d	ANN	296948	2382	9000
18	RDTest21d_06J04M10P_20201015_0140.log	Test21d	ANN	290878	1370	5100
19	RDTest21d_06J04M10P_20201015_0633.log	Test21d	ANN	286347	974	3600
20	MTSTest21d_06J04M10P_20201014_2047.log	Test21d	MTS	288884	1451	5100
21	MTSTest21d_06J04M10P_20201014_2324.log	Test21d	MTS	281077	1595	5600
22	MTSTest21d_06J04M10P_20201015_0203.log	Test21d	MTS	302708	2181	7700
23	MTSTest21d_06J04M10P_20201015_0649.log	Test21d	MTS	288884	1486	5100
24	RDTest22a_09J04M10P_20201014_1938.log	Test22a	ANN	*	*	*
25	RDTest22a_09J04M10P_20201014_1954.log	Test22a	ANN	422598	2424	6700
26	RDTest22a_09J04M10P_20201014_2037.log	Test22a	ANN	*	*	*
27	RDTest22a_09J04M10P_20201014_2038.log	Test22a	ANN	406815	1884	5200
28	RDTest22a_09J04M10P_20201014_2333.log	Test22a	ANN	*	*	*
29	RDTest22a_09J04M10P_20201014_2334.log	Test22a	ANN	427347	2035	5600
30	RDTest22a_09J04M10P_20201015_0251.log	Test22a	ANN	402186	1934	5300
31	RDTest22a_09J04M10P_20201015_0536.log	Test22a	ANN	422598	2410	6700
32	MTSTest22a_09J04M10P_20201014_2035.log	Test22a	MTS	*	*	*
33	MTSTest22a_09J04M10P_20201014_2109.log	Test22a	MTS	389577	2774	7200
34	MTSTest22a_09J04M10P_20201015_0008.log	Test22a	MTS	389557	2714	7100
35	MTSTest22a_09J04M10P_20201015_0323.log	Test22a	MTS	*	*	*
36	MTSTest22a_09J04M10P_20201015_0617.log	Test22a	MTS	*	*	*
37	RDTest22c_09J04M10P_20201014_2259.log	Test22c	ANN	413156	1833	5000
38	RDTest22c_09J04M10P_20201015_0217.log	Test22c	ANN	*	*	*

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
39	MTSTest22c_09J04M10P_20201014_2330.log	Test22c	MTS	*	*	*
40	RDTest22d_09J04M10P_20201014_2155.log	Test22d	ANN	440339	1576	4300
41	RDTest22d_09J04M10P_20201015_0053.log	Test22d	ANN	421702	2800	7700
42	MTSTest22d_09J04M10P_20201014_2222.log	Test22d	MTS	408269	2255	5800
43	MTSTest22d_09J04M10P_20201015_0140.log	Test22d	MTS	397363	2219	5700
44	RDTest23a_09J06M10P_20201014_1941.log	Test23a	ANN	393388	16459	28400
45	MTSTest23a_09J06M10P_20201015_0015.log	Test23a	MTS	464188	4705	7600
46	RDTest23c_09J06M10P_20201015_0419.log	Test23c	ANN	286720	8450	14500
47	MTSTest23c_09J06M10P_20201015_0640.log	Test23c	MTS	295546	7225	11700
48	RDTest23d_09J06M10P_20201015_0134.log	Test23d	ANN	449012	7299	12600
49	RDTest23d_09J06M10P_20201015_0840.log	Test23d	ANN	462385	6677	11500
50	MTSTest23d_09J06M10P_20201015_0335.log	Test23d	MTS	452328	2637	4300
51	MTSTest23d_09J06M10P_20201015_1032.log	Test23d	MTS	448918	5341	8700
52	RDTest23e_09J06M10P_20201015_1201.log	Test23e	ANN	394960	8188	14100
53	MTSTest23e_09J06M10P_20201015_1417.log	Test23e	MTS	423246	4969	8100
54	RDTest23f_09J06M10P_20201015_1540.log	Test23f	ANN	306966	5166	8900
55	RDTest24a_06J04M15P_20201014_1941.log	Test24a	ANN	416922	3967	7100
56	RDTest24a_06J04M15P_20201015_0605.log	Test24a	ANN	418290	2408	4300
57	MTSTest24a_06J04M15P_20201014_2047.log	Test24a	MTS	416258	3286	5600
58	MTSTest24a_06J04M15P_20201015_0645.log	Test24a	MTS	392539	5241	8800
59	RDTest24c_06J04M15P_20201015_0119.log	Test24c	ANN	275817	5709	10200
60	RDTest24c_06J04M15P_20201015_0953.log	Test24c	ANN	276843	3289	5900
61	MTSTest24c_06J04M15P_20201015_0254.log	Test24c	MTS	260860	3505	6000
62	MTSTest24c_06J04M15P_20201015_1048.log	Test24c	MTS	285645	2740	4600
63	RDTest24d_06J04M15P_20201014_2142.log	Test24d	ANN	359698	7964	14200
64	RDTest24d_06J04M15P_20201015_0352.log	Test24d	ANN	349713	3420	6100
65	RDTest24d_06J04M15P_20201015_0813.log	Test24d	ANN	380210	2623	4700
66	RDTest24d_06J04M15P_20201015_1134.log	Test24d	ANN	368323	6366	11400
67	MTSTest24d_06J04M15P_20201014_2355.log	Test24d	MTS	362982	5044	8600
68	MTSTest24d_06J04M15P_20201015_0449.log	Test24d	MTS	373961	3045	5200
69	MTSTest24d_06J04M15P_20201015_0857.log	Test24d	MTS	377773	3342	5600
70	MTSTest24d_06J04M15P_20201015_1320.log	Test24d	MTS	350165	8670	14600
71	RDTest24e_06J04M15P_20201015_0540.log	Test24e	ANN	*	*	*
72	RDTest25a_09J04M15P_20201014_1942.log	Test25a	ANN	*	*	*
73	RDTest25a_09J04M15P_20201014_1954.log	Test25a	ANN	585775	2968	3600
74	RDTest25a_09J04M15P_20201014_2047.log	Test25a	ANN	*	*	*
75	RDTest25a_09J04M15P_20201014_2048.log	Test25a	ANN	539679	9593	11700

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
76	RDTest25a_09J04M15P_20201015_0132.log	Test25a	ANN	585775	2786	3600
77	RDTest25a_09J04M15P_20201015_0252.log	Test25a	ANN	539679	9120	11700
78	MTSTest25a_09J04M15P_20201014_2043.log	Test25a	MTS	*	*	*
79	MTSTest25a_09J04M15P_20201014_2328.log	Test25a	MTS	522377	5687	6600
80	MTSTest25a_09J04M15P_20201015_0219.log	Test25a	MTS	*	*	*
81	MTSTest25a_09J04M15P_20201015_0524.log	Test25a	MTS	522377	5352	6600
82	RDTest25d_09J04M15P_20201015_0103.log	Test25d	ANN	*	*	*
83	RDTest26a_09J06M15P_20201014_1943.log	Test26a	ANN	546951	10755	9000
84	MTSTest26a_09J06M15P_20201014_2242.log	Test26a	MTS	491748	12863	10100
85	RDTest26c_09J06M15P_20201015_1247.log	Test26c	ANN	*	*	*
86	MTSTest26c_09J06M15P_20201015_1955.log	Test26c	MTS	402644	33140	26100
87	RDTest26d_09J06M15P_20201015_0217.log	Test26d	ANN	638555	20427	16700
88	RDTest26d_09J06M15P_20201016_0507.log	Test26d	ANN	652679	10849	8900
89	MTSTest26d_09J06M15P_20201015_0757.log	Test26d	MTS	624106	17298	13500
90	RDTest27a_06J04M20P_20201014_1944.log	Test27a	ANN	480020	7661	8100
91	RDTest27a_06J04M20P_20201014_2355.log	Test27a	ANN	450946	7386	7800
92	MTSTest27a_06J04M20P_20201014_2152.log	Test27a	MTS	461499	7378	7500
93	MTSTest27a_06J04M20P_20201015_0158.log	Test27a	MTS	451367	19277	19300
94	RDTest27c_06J04M20P_20201015_0958.log	Test27c	ANN	642481	10419	11100
95	RDTest27d_06J04M20P_20201014_2355.log	Test27d	ANN	*	*	*
96	RDTest27d_06J04M20P_20201015_0719.log	Test27d	ANN	323187	4358	4600
97	MTSTest27d_06J04M20P_20201015_0832.log	Test27d	MTS	328270	5081	5100
98	RDTest28a_09J04M20P_20201014_1945.log	Test28a	ANN	*	*	*
99	RDTest28a_09J04M20P_20201014_1954.log	Test28a	ANN	1015067	9249	6900
100	RDTest28a_09J04M20P_20201014_2242.log	Test28a	ANN	1036738	6580	4900
101	RDTest28a_09J04M20P_20201015_0056.log	Test28a	ANN	*	*	*
102	RDTest28a_09J04M20P_20201015_0057.log	Test28a	ANN	*	*	*
103	RDTest28a_09J04M20P_20201015_0058.log	Test28a	ANN	1015067	9368	6900
104	RDTest28a_09J04M20P_20201015_0538.log	Test28a	ANN	1015067	9405	6900
105	MTSTest28a_09J04M20P_20201014_2228.log	Test28a	MTS	*	*	*
106	MTSTest28a_09J04M20P_20201015_0032.log	Test28a	MTS	*	*	*
107	MTSTest28a_09J04M20P_20201015_0334.log	Test28a	MTS	*	*	*
108	RDTest29a_09J06M20P_20201014_1947.log	Test29a	ANN	638413	45088	19900
109	MTSTest29a_09J06M20P_20201015_0818.log	Test29a	MTS	616603	30533	13000
110	MTSTest29d_09J06M20P_20201015_2048.log	Test29d	MTS	709881	44085	18600
111	RDTest30a_06J06M25P_20201014_1948.log	Test30a	ANN	453030	15486	6900
112	MTSTest30a_06J06M25P_20201015_0006.log	Test30a	MTS	433656	10703	4600

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
113	RDTest30c_06J06M25P_20201015_1219.log	Test30c	ANN	442003	15140	6700
114	RDTest30d_06J06M25P_20201015_0304.log	Test30d	ANN	580493	16417	7300
115	RDTest30d_06J06M25P_20201015_2023.log	Test30d	ANN	615909	16989	7500
116	MTSTest30d_06J06M25P_20201015_0738.log	Test30d	MTS	597527	16824	7100
117	MTSTest30d_06J06M25P_20201016_0106.log	Test30d	MTS	616819	11272	4800
118	RDTest30e_06J06M25P_20201016_0414.log	Test30e	ANN	331688	8967	4000
119	RDTest30f_06J06M25P_20201016_1125.log	Test30f	ANN	203461	13029	5800
120	RDTest31a_06J08M25P_20201014_1948.log	Test31a	ANN	229637	28518	9200
121	MTSTest31a_06J08M25P_20201015_0344.log	Test31a	MTS	301579	13340	4100
122	MTSTest31c_06J08M25P_20201015_1944.log	Test31c	MTS	268206	12706	3900
123	RDTest31d_06J08M25P_20201015_0726.log	Test31d	ANN	360866	18054	5800
124	RDTest31d_06J08M25P_20201015_2316.log	Test31d	ANN	378411	15306	4900
125	MTSTest31d_06J08M25P_20201015_1227.log	Test31d	MTS	392168	11714	3600
126	RDTest32a_06J10M25P_20201014_1949.log	Test32a	ANN	302998	24801	5600
127	MTSTest32a_06J10M25P_20201015_0243.log	Test32a	MTS	317084	18645	4000
128	RDTest32c_06J10M25P_20201015_2237.log	Test32c	ANN	*	*	*
129	RDTest32d_06J10M25P_20201015_0753.log	Test32d	ANN	227838	31682	7200
130	RDTest33a_06J06M30P_20201014_1951.log	Test33a	ANN	510317	17749	5500
131	MTSTest33a_06J06M30P_20201015_0047.log	Test33a	MTS	473835	26901	7900
132	RDTest33c_06J06M30P_20201015_1901.log	Test33c	ANN	546957	12607	3900
133	MTSTest33c_06J06M30P_20201015_2231.log	Test33c	MTS	531293	16367	4800
134	RDTest33d_06J06M30P_20201015_0815.log	Test33d	ANN	659587	18464	5700
135	RDTest33d_06J06M30P_20201016_0304.log	Test33d	ANN	632228	17135	5300
136	RDTest34a_06J08M30P_20201014_1953.log	Test34a	ANN	425304	46417	10000
137	RDTest34d_06J08M30P_20201016_0042.log	Test34d	ANN	360737	19952	4300
138	RDTest35a_06J10M30P_20201014_1954.log	Test35a	ANN	417405	27677	4700
139	MTSTest35a_06J10M30P_20201015_0335.log	Test35a	MTS	*	*	*
140	RDTest35d_06J10M30P_20201015_1809.log	Test35d	ANN	314896	40783	6900
141	RDTest41a_06J04M10P_20201014_2019.log	Test41a	ANN	191852	1416	5200
142	RDTest41a_06J04M10P_20201014_2147.log	Test41a	ANN	165756	2036	7600
143	RDTest41a_06J04M10P_20201014_2300.log	Test41a	ANN	*	*	*
144	RDTest41a_06J04M10P_20201014_2301.log	Test41a	ANN	191852	1367	5200
145	RDTest41a_06J04M10P_20201015_0057.log	Test41a	ANN	165756	2004	7600
146	RDTest41a_06J04M10P_20201015_0252.log	Test41a	ANN	182624	1319	5000
147	RDTest41a_06J04M10P_20201015_0536.log	Test41a	ANN	191852	1394	5200
148	MTSTest41a_06J04M10P_20201014_2043.log	Test41a	MTS	172424	2336	8300
149	MTSTest41a_06J04M10P_20201014_2221.log	Test41a	MTS	182618	1596	5700
150	MTSTest41a_06J04M10P_20201014_2324.log	Test41a	MTS	172424	2351	8300

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
151	MTSTest41a_06J04M10P_20201015_0131.log	Test41a	MTS	182618	1596	5700
152	MTSTest41a_06J04M10P_20201015_0314.log	Test41a	MTS	*	*	*
153	MTSTest41a_06J04M10P_20201015_0559.log	Test41a	MTS	172424	2369	8300
154	RDTest41d_06J04M10P_20201014_2122.log	Test41d	ANN	*	*	*
155	RDTest41d_06J04M10P_20201014_2248.log	Test41d	ANN	*	*	*
156	RDTest41d_06J04M10P_20201015_0003.log	Test41d	ANN	*	*	*
157	RDTest41d_06J04M10P_20201015_0157.log	Test41d	ANN	*	*	*
158	RDTest42a_09J04M10P_20201014_2020.log	Test42a	ANN	532184	2580	7000
159	RDTest42a_09J04M10P_20201014_2147.log	Test42a	ANN	552510	2238	6100
160	RDTest42a_09J04M10P_20201014_2243.log	Test42a	ANN	566903	2659	7300
161	RDTest42a_09J04M10P_20201015_0057.log	Test42a	ANN	564159	1384	3800
162	RDTest42a_09J04M10P_20201015_0132.log	Test42a	ANN	532184	2532	7000
163	RDTest42a_09J04M10P_20201015_0253.log	Test42a	ANN	552510	2219	6100
164	RDTest42a_09J04M10P_20201015_0538.log	Test42a	ANN	532184	2534	7000
165	MTSTest42a_09J04M10P_20201014_2103.log	Test42a	MTS	*	*	*
169	MTSTest42a_09J04M10P_20201015_0215.log	Test42a	MTS	*	*	*
170	MTSTest42a_09J04M10P_20201015_0330.log	Test42a	MTS	*	*	*
171	RDTest42d_09J04M10P_20201014_2352.log	Test42d	ANN	*	*	*
172	RDTest43a_09J06M10P_20201014_2021.log	Test43a	ANN	450124	7118	12600
173	MTSTest43a_09J06M10P_20201014_2220.log	Test43a	MTS	387627	11924	19800
174	RDTest43c_09J06M10P_20201015_0352.log	Test43c	ANN	398092	6806	11900
175	MTSTest43c_09J06M10P_20201015_0545.log	Test43c	MTS	405040	6752	11200
176	RDTest43d_09J06M10P_20201015_0139.log	Test43d	ANN	224536	3928	6900
177	RDTest43d_09J06M10P_20201015_0738.log	Test43d	ANN	224345	9958	17400
178	MTSTest43d_09J06M10P_20201015_0244.log	Test43d	MTS	231822	4072	6800
179	MTSTest43d_09J06M10P_20201015_1024.log	Test43d	MTS	224740	6655	11100
180	RDTest43e_09J06M10P_20201015_1215.log	Test43e	ANN	271280	3922	6900
181	MTSTest43e_09J06M10P_20201015_1320.log	Test43e	MTS	290808	4548	7600
182	RDTest43f_09J06M10P_20201015_1436.log	Test43f	ANN	416930	9537	16600
183	RDTest44a_06J04M15P_20201014_2022.log	Test44a	ANN	183549	3748	6400
184	RDTest44a_06J04M15P_20201015_0058.log	Test44a	ANN	215756	3008	5100
185	RDTest44a_06J04M15P_20201015_0606.log	Test44a	ANN	182993	3136	5300
186	MTSTest44a_06J04M15P_20201014_2124.log	Test44a	MTS	198851	3585	5800
187	MTSTest44a_06J04M15P_20201015_0148.log	Test44a	MTS	196352	3169	5400
188	MTSTest44a_06J04M15P_20201015_0659.log	Test44a	MTS	187940	3993	6700
189	RDTest44c_06J04M15P_20201015_0000.log	Test44c	ANN	*	*	*
190	RDTest44c_06J04M15P_20201015_0541.log	Test44c	ANN	*	*	*
191	RDTest44d_06J04M15P_20201014_2224.log	Test44d	ANN	276815	3533	6000

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
192	RDTest44d_06J04M15P_20201015_0240.log	Test44d	ANN	303337	2429	4100
193	RDTest44d_06J04M15P_20201015_0805.log	Test44d	ANN	289032	5795	9700
194	MTSTest44d_06J04M15P_20201014_2323.log	Test44d	MTS	292571	2230	3600
195	MTSTest44d_06J04M15P_20201015_0321.log	Test44d	MTS	304092	8417	13500
196	MTSTest44d_06J04M15P_20201015_0942.log	Test44d	MTS	298632	3596	5800
197	RDTest45a_09J04M15P_20201014_2023.log	Test45a	ANN	533318	5300	6700
198	RDTest45a_09J04M15P_20201014_2301.log	Test45a	ANN	529685	3011	3800
199	RDTest45a_09J04M15P_20201015_0102.log	Test45a	ANN	490882	3298	4200
200	RDTest45a_09J04M15P_20201015_0543.log	Test45a	ANN	533318	5539	6700
201	MTSTest45a_09J04M15P_20201014_2151.log	Test45a	MTS	505841	3681	4900
202	MTSTest45a_09J04M15P_20201014_2351.log	Test45a	MTS	526006	3223	3900
203	MTSTest45a_09J04M15P_20201015_0157.log	Test45a	MTS	483469	4807	6200
204	MTSTest45a_09J04M15P_20201015_0715.log	Test45a	MTS	505841	3837	4900
205	RDTest45d_09J04M15P_20201014_2253.log	Test45d	ANN	*	*	*
206	RDTest45d_09J04M15P_20201015_0045.log	Test45d	ANN	*	*	*
207	RDTest45d_09J04M15P_20201015_0317.log	Test45d	ANN	*	*	*
208	RDTest46a_09J06M15P_20201014_2023.log	Test46a	ANN	943705	12795	10300
209	MTSTest46a_09J06M15P_20201014_2357.log	Test46a	MTS	846868	14465	11200
210	RDTest46c_09J06M15P_20201015_0933.log	Test46c	ANN	660866	15526	12500
211	MTSTest46c_09J06M15P_20201015_1352.log	Test46c	MTS	614406	9215	7100
212	RDTest46d_09J06M15P_20201015_0358.log	Test46d	ANN	517541	7257	5900
213	MTSTest46d_09J06M15P_20201015_0559.log	Test46d	MTS	501856	12863	10000
214	MTSTest46d_09J06M15P_20201015_2202.log	Test46d	MTS	520168	14038	10900
215	RDTest46e_09J06M15P_20201016_0156.log	Test46e	ANN	*	*	*
216	RDTest47a_06J04M20P_20201014_2024.log	Test47a	ANN	380938	4783	5000
217	RDTest47a_06J04M20P_20201015_0541.log	Test47a	ANN	380938	4706	5000
218	MTSTest47a_06J04M20P_20201014_2144.log	Test47a	MTS	404656	7805	7800
219	MTSTest47a_06J04M20P_20201015_0700.log	Test47a	MTS	404656	7703	7800
220	RDTest47c_06J04M20P_20201015_0401.log	Test47c	ANN	*	*	*
221	RDTest47d_06J04M20P_20201014_2354.log	Test47d	ANN	479193	6436	6700
222	RDTest47d_06J04M20P_20201015_0908.log	Test47d	ANN	479193	6385	6700
223	MTSTest47d_06J04M20P_20201015_0141.log	Test47d	MTS	492814	8387	8400
224	MTSTest47d_06J04M20P_20201015_1054.log	Test47d	MTS	492814	8300	8400
225	RDTest48a_09J04M20P_20201014_2025.log	Test48a	ANN	777352	10812	7800
226	RDTest48a_09J04M20P_20201014_2325.log	Test48a	ANN	*	*	*
227	RDTest48a_09J04M20P_20201014_2336.log	Test48a	ANN	*	*	*
228	RDTest48a_09J04M20P_20201014_2337.log	Test48a	ANN	851389	5718	4100
229	RDTest48a_09J04M20P_20201015_0135.log	Test48a	ANN	777352	10639	7800

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
230	RDTest48a_09J04M20P_20201015_0541.log	Test48a	ANN	777352	10874	7800
231	MTSTest48a_09J04M20P_20201014_2325.log	Test48a	MTS	*	*	*
232	MTSTest48a_09J04M20P_20201015_0112.log	Test48a	MTS	*	*	*
233	MTSTest48a_09J04M20P_20201015_0432.log	Test48a	MTS	*	*	*
234	RDTest49a_09J06M20P_20201014_2026.log	Test49a	ANN	765383	51332	22800
235	MTSTest49a_09J06M20P_20201015_1041.log	Test49a	MTS	*	*	*
236	RDTest49d_09J06M20P_20201015_1901.log	Test49d	ANN	781021	36918	16500
237	MTSTest49d_09J06M20P_20201016_0517.log	Test49d	MTS	821879	10630	4600
238	RDTest50a_06J06M25P_20201014_2025.log	Test50a	ANN	493294	11950	5300
239	MTSTest50a_06J06M25P_20201014_2345.log	Test50a	MTS	524049	8675	3700
240	MTSTest50c_06J06M25P_20201015_1846.log	Test50c	MTS	545936	14688	6200
241	RDTest50d_06J06M25P_20201015_0209.log	Test50d	ANN	307070	15920	7100
242	RDTest50d_06J06M25P_20201015_2250.log	Test50d	ANN	289492	10795	4800
243	MTSTest50d_06J06M25P_20201015_0634.log	Test50d	MTS	305995	26934	11500
244	MTSTest50d_06J06M25P_20201016_0150.log	Test50d	MTS	*	*	*
245	RDTest50e_06J06M25P_20201016_0925.log	Test50e	ANN	*	*	*
246	RDTest51a_06J08M25P_20201014_2027.log	Test51a	ANN	233742	17986	5800
247	MTSTest51a_06J08M25P_20201015_0127.log	Test51a	MTS	268326	16818	5200
248	MTSTest51c_06J08M25P_20201015_1841.log	Test51c	MTS	283576	13667	4200
249	RDTest51d_06J08M25P_20201015_0607.log	Test51d	ANN	298695	11542	3700
250	RDTest51d_06J08M25P_20201015_2228.log	Test51d	ANN	266167	19707	6300
251	MTSTest51d_06J08M25P_20201015_0920.log	Test51d	MTS	236045	18562	5700
252	RDTest52a_06J10M25P_20201014_2028.log	Test52a	ANN	201219	20134	4800
253	MTSTest52a_06J10M25P_20201015_0204.log	Test52a	MTS	206172	35977	8200
254	RDTest52c_06J10M25P_20201015_2350.log	Test52c	ANN	158228	26069	6200
255	RDTest52d_06J10M25P_20201015_1203.log	Test52d	ANN	212436	15634	3700
256	RDTest53a_06J06M30P_20201014_2029.log	Test53a	ANN	477578	24443	7600
257	MTSTest53a_06J06M30P_20201015_0316.log	Test53a	MTS	462044	14805	4400
258	RDTest53c_06J06M30P_20201015_1738.log	Test53c	ANN	602723	12115	3800
259	MTSTest53c_06J06M30P_20201015_2100.log	Test53c	MTS	558545	13857	4200
260	RDTest53d_06J06M30P_20201015_0723.log	Test53d	ANN	621145	22859	7100
261	RDTest53d_06J06M30P_20201016_0051.log	Test53d	ANN	*	*	*
262	MTSTest53d_06J06M30P_20201016_0928.log	Test53d	MTS	591578	14803	4500
263	RDTest54a_06J08M30P_20201014_2029.log	Test54a	ANN	295713	20499	4500
264	MTSTest54a_06J08M30P_20201015_0211.log	Test54a	MTS	218322	19197	4000
265	RDTest54d_06J08M30P_20201015_0731.log	Test54d	ANN	*	*	*
266	MTSTest54d_06J08M30P_20201015_2127.log	Test54d	MTS	344209	32383	6800
267	RDTest55a_06J10M30P_20201014_2030.log	Test55a	ANN	370373	28376	4700

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
268	MTSTest55a_06J10M30P_20201015_0423.log	Test55a	MTS	351440	23380	3700
269	MTSTest55d_06J10M30P_20201015_1818.log	Test55d	MTS	154176	37751	6000
270	RDTest61a_06J04M10P_20201014_2030.log	Test61a	ANN	*	*	*
271	RDTest61a_06J04M10P_20201014_2032.log	Test61a	ANN	150833	2182	8100
272	RDTest61a_06J04M10P_20201014_2148.log	Test61a	ANN	142950	2117	7800
273	RDTest61a_06J04M10P_20201014_2301.log	Test61a	ANN	*	*	*
274	RDTest61a_06J04M10P_20201014_2302.log	Test61a	ANN	150833	2163	8100
275	RDTest61a_06J04M10P_20201014_2340.log	Test61a	ANN	142950	2084	7800
276	RDTest61a_06J04M10P_20201015_0055.log	Test61a	ANN	*	*	*
277	RDTest61a_06J04M10P_20201015_0057.log	Test61a	ANN	150833	2209	8100
278	RDTest61a_06J04M10P_20201015_0148.log	Test61a	ANN	*	*	*
279	RDTest61a_06J04M10P_20201015_0149.log	Test61a	ANN	*	*	*
280	RDTest61a_06J04M10P_20201015_0151.log	Test61a	ANN	150833	2124	8100
281	RDTest61a_06J04M10P_20201015_0232.log	Test61a	ANN	142950	2036	7800
282	RDTest61a_06J04M10P_20201015_0535.log	Test61a	ANN	150833	2167	8100
283	MTSTest61a_06J04M10P_20201014_2108.log	Test61a	MTS	*	*	*
284	MTSTest61a_06J04M10P_20201014_2224.log	Test61a	MTS	146934	1290	4600
285	MTSTest61a_06J04M10P_20201014_2338.log	Test61a	MTS	*	*	*
286	MTSTest61a_06J04M10P_20201015_0015.log	Test61a	MTS	146934	1289	4600
287	MTSTest61a_06J04M10P_20201015_0134.log	Test61a	MTS	*	*	*
288	MTSTest61a_06J04M10P_20201015_0226.log	Test61a	MTS	*	*	*
289	MTSTest61a_06J04M10P_20201015_0306.log	Test61a	MTS	146934	1255	4600
290	RDTest61d_06J04M10P_20201014_2245.log	Test61d	ANN	*	*	*
291	RDTest61d_06J04M10P_20201015_0036.log	Test61d	ANN	*	*	*
292	RDTest61d_06J04M10P_20201015_0327.log	Test61d	ANN	*	*	*
293	RDTest62a_09J04M10P_20201014_2030.log	Test62a	ANN	466159	2240	6100
294	RDTest62a_09J04M10P_20201014_2240.log	Test62a	ANN	468485	1502	4100
295	RDTest62a_09J04M10P_20201014_2339.log	Test62a	ANN	463020	1843	5000
296	RDTest62a_09J04M10P_20201015_0130.log	Test62a	ANN	453074	2481	6700
297	RDTest62a_09J04M10P_20201015_0251.log	Test62a	ANN	467631	3399	9200
298	RDTest62a_09J04M10P_20201015_0534.log	Test62a	ANN	466159	2226	6100
299	MTSTest62a_09J04M10P_20201014_2107.log	Test62a	MTS	469348	4681	11900
300	MTSTest62a_09J04M10P_20201014_2305.log	Test62a	MTS	495670	1876	4800
301	MTSTest62a_09J04M10P_20201015_0010.log	Test62a	MTS	449236	3683	9400
302	MTSTest62a_09J04M10P_20201015_0211.log	Test62a	MTS	449291	1673	4300
303	MTSTest62a_09J04M10P_20201015_0347.log	Test62a	MTS	467760	2969	7600
304	MTSTest62a_09J04M10P_20201015_0611.log	Test62a	MTS	469348	4642	11900
305	RDTest62d_09J04M10P_20201014_2225.log	Test62d	ANN	*	*	*

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
306	RDTest62d_09J04M10P_20201014_2336.log	Test62d	ANN	*	*	*
307	RDTest62d_09J04M10P_20201015_0111.log	Test62d	ANN	*	*	*
308	RDTest62d_09J04M10P_20201015_0239.log	Test62d	ANN	*	*	*
309	RDTest62d_09J04M10P_20201015_0437.log	Test62d	ANN	*	*	*
310	RDTest63a_09J06M10P_20201014_2031.log	Test63a	ANN	229547	9220	16300
311	RDTest63a_09J06M10P_20201014_2152.log	Test63a	ANN	224951	5615	10000
312	MTSTest63a_09J06M10P_20201014_2305.log	Test63a	MTS	253711	2808	4700
313	MTSTest63a_09J06M10P_20201014_2326.log	Test63a	MTS	249706	8095	13700
314	RDTest63c_09J06M10P_20201015_0152.log	Test63c	ANN	299553	3449	6100
315	RDTest63c_09J06M10P_20201015_0435.log	Test63c	ANN	334621	5611	9900
316	MTSTest63c_09J06M10P_20201015_0250.log	Test63c	MTS	298041	9458	15900
317	MTSTest63c_09J06M10P_20201015_0608.log	Test63c	MTS	306735	9083	15200
318	RDTest63d_09J06M10P_20201014_2352.log	Test63d	ANN	274234	3239	5700
319	RDTest63d_09J06M10P_20201015_0141.log	Test63d	ANN	273724	8322	14800
320	RDTest63d_09J06M10P_20201015_0527.log	Test63d	ANN	281471	2989	5300
321	RDTest63d_09J06M10P_20201015_0840.log	Test63d	ANN	286268	3459	6100
322	MTSTest63d_09J06M10P_20201015_0046.log	Test63d	MTS	283128	3989	6700
323	MTSTest63d_09J06M10P_20201015_0359.log	Test63d	MTS	296580	2114	3600
324	MTSTest63d_09J06M10P_20201015_0617.log	Test63d	MTS	281064	3870	6500
325	MTSTest63d_09J06M10P_20201015_0937.log	Test63d	MTS	269191	5962	10000
326	RDTest63e_09J06M10P_20201015_0722.log	Test63e	ANN	334490	6388	11300
327	RDTest63e_09J06M10P_20201015_1117.log	Test63e	ANN	348673	4119	7300
328	MTSTest63e_09J06M10P_20201015_0908.log	Test63e	MTS	329172	8779	14800
329	MTSTest63e_09J06M10P_20201015_1225.log	Test63e	MTS	307549	7214	12100
330	RDTest63f_09J06M10P_20201015_1135.log	Test63f	ANN	288084	12858	22700
331	RDTest63f_09J06M10P_20201015_1426.log	Test63f	ANN	285917	5541	9800
332	MTSTest63f_09J06M10P_20201015_1509.log	Test63f	MTS	296999	7527	12700
333	MTSTest63f_09J06M10P_20201015_1558.log	Test63f	MTS	327664	4307	7200
334	RDTest65a_09J04M15P_20201014_2036.log	Test65a	ANN	687765	4554	5700
335	RDTest65a_09J04M15P_20201014_2154.log	Test65a	ANN	690902	5142	6400
336	RDTest65a_09J04M15P_20201015_0544.log	Test65a	ANN	687765	4498	5700
337	MTSTest65a_09J04M15P_20201014_2152.log	Test65a	MTS	*	*	*
338	MTSTest65a_09J04M15P_20201014_2319.log	Test65a	MTS	663996	2991	3600
339	MTSTest65a_09J04M15P_20201015_0659.log	Test65a	MTS	*	*	*
340	RDTest65c_09J04M15P_20201015_0321.log	Test65c	ANN	571267	3216	4100
341	MTSTest65c_09J04M15P_20201015_0414.log	Test65c	MTS	*	*	*
342	RDTest65d_09J04M15P_20201015_0009.log	Test65d	ANN	530101	3071	3900

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
343	MTSTest65d_09J04M15P_20201015_0100.log	Test65d	MTS	572470	8412	10100
344	RDTest66a_09J06M15P_20201014_2033.log	Test66a	ANN	556636	32203	26600
345	MTSTest66a_09J06M15P_20201015_0530.log	Test66a	MTS	623236	23338	18500
346	MTSTest66c_09J06M15P_20201015_1915.log	Test66c	MTS	611259	10778	8600
347	RDTest66d_09J06M15P_20201015_1159.log	Test66d	ANN	605863	8861	7300
348	RDTest66d_09J06M15P_20201015_2215.log	Test66d	ANN	680274	8690	7200
349	MTSTest66d_09J06M15P_20201015_1427.log	Test66d	MTS	667329	6531	5200
350	MTSTest66d_09J06M15P_20201016_0040.log	Test66d	MTS	624631	18825	15100
351	RDTest66e_09J06M15P_20201016_0553.log	Test66e	ANN	546374	9973	8300
352	RDTest67a_06J04M20P_20201014_2035.log	Test67a	ANN	*	*	*
353	RDTest67a_06J04M20P_20201014_2042.log	Test67a	ANN	716052	3985	4000
354	RDTest67a_06J04M20P_20201014_2150.log	Test67a	ANN	627612	3724	3800
355	RDTest67a_06J04M20P_20201014_2303.log	Test67a	ANN	716052	3889	4000
356	RDTest67a_06J04M20P_20201015_0102.log	Test67a	ANN	*	*	*
357	RDTest67a_06J04M20P_20201015_0103.log	Test67a	ANN	716052	3817	4000
358	RDTest67a_06J04M20P_20201015_0252.log	Test67a	ANN	716052	3831	4000
359	RDTest67a_06J04M20P_20201015_0543.log	Test67a	ANN	*	*	*
360	RDTest67a_06J04M20P_20201015_0544.log	Test67a	ANN	716052	3810	4000
361	MTSTest67a_06J04M20P_20201014_2148.log	Test67a	MTS	*	*	*
362	MTSTest67a_06J04M20P_20201014_2252.log	Test67a	MTS	*	*	*
363	MTSTest67a_06J04M20P_20201015_0008.log	Test67a	MTS	*	*	*
364	MTSTest67a_06J04M20P_20201015_0206.log	Test67a	MTS	*	*	*
365	MTSTest67a_06J04M20P_20201015_0356.log	Test67a	MTS	*	*	*
366	RDTest68a_09J04M20P_20201014_2034.log	Test68a	ANN	892187	7481	5500
367	RDTest68a_09J04M20P_20201014_2244.log	Test68a	ANN	878307	7001	5100
368	RDTest68a_09J04M20P_20201015_0252.log	Test68a	ANN	892187	7407	5500
369	RDTest68a_09J04M20P_20201015_0543.log	Test68a	ANN	892187	7632	5500
370	MTSTest68a_09J04M20P_20201014_2239.log	Test68a	MTS	*	*	*
371	MTSTest68a_09J04M20P_20201015_0040.log	Test68a	MTS	962129	6391	4500
372	MTSTest68a_09J04M20P_20201015_0455.log	Test68a	MTS	*	*	*
373	RDTest68d_09J04M20P_20201015_0227.log	Test68d	ANN	*	*	*
374	RDTest69a_09J06M20P_20201014_2037.log	Test69a	ANN	702755	20683	9400
375	MTSTest69a_09J06M20P_20201015_0222.log	Test69a	MTS	658351	19077	8400
376	RDTest69c_09J06M20P_20201015_2003.log	Test69c	ANN	695535	38607	17500
377	RDTest69d_09J06M20P_20201015_0740.log	Test69d	ANN	491809	18974	8600
378	MTSTest69d_09J06M20P_20201015_1256.log	Test69d	MTS	*	*	*
379	RDTest70a_06J06M25P_20201014_2036.log	Test70a	ANN	248939	12625	5600
380	MTSTest70a_06J06M25P_20201015_0007.log	Test70a	MTS	281182	8492	3600

ตารางภาคผนวก ก.3 ข้อมูลผลการทดสอบปัญหาขนาดเล็ก (ต่อ)

Item	File Name	Case	Mode	Objt	Time	Iter
381	RDTest70c_06J06M25P_20201015_1205.log	Test70c	ANN	431308	12533	5600
382	RDTest70d_06J06M25P_20201015_0228.log	Test70d	ANN	357912	21987	9800
383	RDTest70d_06J06M25P_20201015_1921.log	Test70d	ANN	373957	30838	13700
384	MTSTest70d_06J06M25P_20201015_0835.log	Test70d	MTS	414629	12578	5300
385	MTSTest70d_06J06M25P_20201016_0355.log	Test70d	MTS	*	*	*
386	RDTest71a_06J08M25P_20201014_2036.log	Test71a	ANN	224684	28040	9100
387	MTSTest71a_06J08M25P_20201015_0423.log	Test71a	MTS	240981	22155	6800
388	RDTest71c_06J08M25P_20201015_2138.log	Test71c	ANN	213569	28602	9000
389	RDTest71d_06J08M25P_20201015_1033.log	Test71d	ANN	322741	22526	7100
390	RDTest72a_06J10M25P_20201014_2035.log	Test72a	ANN	251826	45113	10700
391	RDTest72d_06J10M25P_20201015_1955.log	Test72d	ANN	255571	40369	9600
392	RDTest73a_06J06M30P_20201014_2037.log	Test73a	ANN	515923	12178	3800
393	MTSTest73a_06J06M30P_20201015_0000.log	Test73a	MTS	521661	20131	6000
394	MTSTest73c_06J06M30P_20201015_1948.log	Test73c	MTS	516581	17895	5300
395	RDTest73d_06J06M30P_20201015_0535.log	Test73d	ANN	570898	17642	5500
396	RDTest73d_06J06M30P_20201016_0046.log	Test73d	ANN	482562	11518	3600
397	MTSTest73d_06J06M30P_20201015_1029.log	Test73d	MTS	490078	21947	6600
398	MTSTest73d_06J06M30P_20201016_0358.log	Test73d	MTS	565571	13800	4100
399	RDTest74a_06J08M30P_20201014_2038.log	Test74a	ANN	417335	23877	5200
400	MTSTest74a_06J08M30P_20201015_0316.log	Test74a	MTS	395461	23754	4900
401	RDTest74c_06J08M30P_20201016_0008.log	Test74c	ANN	*	*	*
402	RDTest74d_06J08M30P_20201015_0952.log	Test74d	ANN	343584	22947	5000
403	RDTest75a_06J10M30P_20201014_2039.log	Test75a	ANN	258905	22084	3800
404	MTSTest75a_06J10M30P_20201015_0247.log	Test75a	MTS	237765	45873	7500

มหาวิทยาลัยเทคโนโลยีสุรนารี

ภาคผนวก ข

โปรแกรมที่พัฒนาขึ้น

มหาวิทยาลัยเทคโนโลยีสุรนารี

ภาคผนวก ข.1 LINGO Math Model >> Case-A – 5J2M3P (6J2M3P)

```
! Reference to file >> ;
! From File NewEdition_Completed_TWT with Lot-sizing_THREEperiod_XX.lg4 ;
! Model_A -- 6J2M5P;

MODEL:

DATA:
        nJobs = 6;
        nPeriods = 4;
        nMachines = 2;
        BigM = 10000;
        WorkHour = 15000;
        LowerLotSize = 5;
ENDDATA

SETS:
        JOB/1..nJobs/: P, R;
        MC/1..nMachines/;
        PED/1..nPeriods/;
        FT( JOB, PED): C, H;
        CXC( JOB, JOB, MC, PED): X;
        DXD( JOB, MC, PED): Y, Z;
        NUM( JOB, MC, PED): N;
        LINK( JOB, JOB): S;
        HOL( JOB, PED): O;
        PROD( JOB, PED): G;
        WorkLoad( JOB, MC): E;
        DEMAND ( JOB, PED): D;
ENDSETS

DATA:

P =     0       11      12      13      14      15;
R =     0       110     120     130     140     120;

S =     0       0       0       0       0       0
        0       0       5       12      12      13
        0       5       0       10      15      12
        0       11      11      0       23      9
        0       14      14      19      0       11
        0       1       3       12      9       0;

G =     0       0       0       0
        8       12      18      22
        8       12      18      22
        9       12      22      23
        10      15      20      25
        15      25      35      45;

O =     0       0       0       0
        2       3       4       7
        2       3       4       8
        2       5       4       8
        3       4       5       8
        3       4       5       2;

D =     0       0       0       0
        10      30      30      20
        10      10      30      30
        20      25      30      40
        25      35      35      45
        25      15      40      30;

E =     !Machine M1     M2;
                1       1
                0       1
                1       1
                1       1
                1       0
                1       0 ;
ENDDATA

!=========================================================================
=====;
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

```
        MIN =   @Sum( JOB( J) : @SUM( PED( T) : G( J, T) * C( J, T) ) ) +
                @Sum( JOB( J) : @SUM( PED( T) : O( J, T) * H( J, T) ) ) +
                @Sum(  MC( M) : @SUM( JOB( I) : @SUM( JOB( J): @SUM( PED( T)| I #NE# J #AND#
J#NE#1 : S( I, J) * X( I, J, M, T) ) ) ) );

        @FOR( JOB( J) :
                H( J, 1) = @Sum( MC( M): N( J, M, 1)) - D( J, 1) ;
        );

        @FOR( JOB( J) : @FOR( PED( T) | J#NE#1 #AND# T#GE#2:
                H( J, T) = H( J, T-1) + @Sum( MC( M): N( J, M, T)) - D( J, T) ;
        ); );

        @FOR( MC( M) : @FOR( JOB( J): @FOR( PED( T) | J#NE#1 :
                @Sum( JOB( I) | I#NE#J : X( I, J, M, T) ) = 1 - Y( J, M, T) ;
        ); ); );


        @FOR( MC( M) : @FOR( JOB( J) : @FOR( PED( T)| J#NE#1:
                N( J, M, T) <= BigM * (1 - Y( J, M, T) ) ;
        ); ); );

        @FOR( JOB( J) : @FOR( PED( T)| J#NE#1:
                @Sum( MC( M): N( J, M, T)) >= LowerLotSize ;
        ); );

        @FOR( JOB( J) : @FOR( PED( T)| J#NE#1:
                @Sum( MC( M): Y( J, M, T)) = 1 ;
        ); );

        @FOR( MC( M) : @FOR( JOB( J) : @FOR( PED( T)| J#NE#1:
                Z( J, M, T) + Y( J, M, T) = 1 ;
        ); ); );

        @FOR( MC( M) : @FOR( JOB( I) : @FOR( JOB( J) : @FOR( PED( T)| I #NE# J #AND# J#NE#1:

                C( J, T) - C( I, T) + ( BigM * ( 1 - ( X( I, J, M, T)* E( I, M) * E( J, M) ) ) )
                        >= P( J)* N( J, M, T) + S( I, J) *( X( I, J, M, T)* E( I, M) * E( J, M) );
        ); ); ); );

        @FOR( MC( M) : @FOR( JOB( I) : @FOR( JOB( J) : @FOR( PED( T)| I #NE# J #AND# J#NE#1:

                C( J, T) - P( J)* N( J, M, T) - ( S( I, J) * X( I, J, M, T)* E( I, M) * E( J, M) ) >= R( J) ;
        ); ); ); );

        @FOR( JOB( J) : @FOR( PED( T):
                C( J, T) <= WorkHour ;
        ); );

        @FOR( JOB( J) : @FOR( PED( T)| J #GE# 2 :
                @SUM( JOB( I) : @SUM( MC( M) : X( I, J, M, T) * E( I, M) * E( J, M) ) ) = 1;
        ); );

        @FOR( JOB( I) : @FOR( PED( T)| I #GE# 2 :
                @SUM( JOB( J) : @SUM( MC( M) : X( I, J, M, T) * E( I, M) * E( J, M) ) ) = 1;
        ); );

        @FOR( MC( M) : @FOR( JOB( J) : @FOR( PED( T):
                @SUM( JOB( I) : X( I, J, M, T)) - @SUM( JOB( W) : X( J, W, M, T)) = 0 ;
        ); ); );

        @FOR( MC( M) : @FOR( JOB( J): @FOR( PED( T):
                X( J, J, M, T) = 0 ;
        ); ); );

        @FOR( MC( M) : @FOR( PED( T) :
                @SUM( JOB( J) | J#NE#1 : X( 1, J, M, T) * E( J, M) ) = 1;
        ); );

        @FOR( MC( M) : @FOR( PED( T) :
                @SUM( JOB( I) | I#NE#1 : X( I, 1, M, T) * E( I, M) ) = 1;
        ); );

        @FOR( JOB( I) : @FOR( JOB( J) | J#NE#I: @FOR( PED( T) :
                @SUM( MC( M) : ( X( I, J, M, T) + X(J, I, M, T) )* E( I, M)* E( J, M) ) <= 1;
        ); ); );
```

```
                @SUM( MC( M) : ( X( I, J, M, T) + X(J, I, M, T) )* E( I, M)* E( J, M) ) <= 1;
        ); ); );

        @FOR( PED( T) :
                @SUM( JOB( J) | J#NE#1 : @SUM( MC( M) : X( 1, J, M, T)* E( J, M) ) ) = nMachines ;
        );

        @FOR( PED( T) :
                @SUM( JOB( I) | I#NE#1 : @SUM( MC( M) : X( I, 1, M, T)* E( I, M) ) ) = nMachines ;
        );

        @FOR( CXC: @BIN( X));
        @FOR( DXD: @BIN( Y));
        @FOR( DXD: @BIN( Z));
        @FOR( NUM: @GIN( N));
        @FOR( FT: @GIN( C));
        @FOR( FT: @GIN( H));

END
```

ภาคผนวก ข.2 MATLAB-Generate Test Case

1/2-ไฟล์ MATLAB “B21_GenerrateLG4_Code.m”

```
% ================================================================
% B21_GenerrateLG4_Code.m
% Update Gen E-Matrix 20201014
% ================================================================
clc, clear all

LoopGenCode4File('61', 10, 4, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('62', 10, 4, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('63', 10, 6, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('64', 15, 4, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('65', 15, 4, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('66', 15, 6, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('67', 20, 4, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('68', 20, 4, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('69', 20, 6, 9); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('70', 25, 6, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('71', 25, 8, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('72', 25, 10,6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('73', 30, 6, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('74', 30, 8, 6); % Name,nJobs,mMachine,tPeriod
LoopGenCode4File('75', 30, 10,6); % Name,nJobs,mMachine,tPeriod

% LoopGenCode4File('00', 7, 10, 3); % Name,nJobs,mMachine,tPeriod

%================================================================
%================================================================

%-------------------------------------------------------
function LoopGenCode4File(test,nJobs,mMachine,tPeriod)
    Gen_CodeLINGO_Run(test+"a",nJobs,mMachine,tPeriod);
    Gen_CodeLINGO_Run(test+"b",nJobs,mMachine,tPeriod);
    Gen_CodeLINGO_Run(test+"c",nJobs,mMachine,tPeriod);
    Gen_CodeLINGO_Run(test+"d",nJobs,mMachine,tPeriod);
    Gen_CodeLINGO_Run(test+"e",nJobs,mMachine,tPeriod);
    Gen_CodeLINGO_Run(test+"f",nJobs,mMachine,tPeriod);
end

%-------------------------------------------------------
function Gen_CodeLINGO_Run(test,nJobs,mMachine,tPeriod)
% -------- Paper Reference ----------
%P(J)   Processing Times    U(10, 20)
%R(J)   Release Time        U(5, 30)
%D(J,P) Demand              U(10, 40)
```

```
%S(I,J) Setup Times         U(5, 20)
%E(J,M) Machine Eligibility U(0, 1)

% -------- My Jobs ----------
nJobs = nJobs-1;
ModelData.nJobs = nJobs+1;
ModelData.mMachine = mMachine;
ModelData.tPeriod = tPeriod;
ModelData.P = [0 randi([10 20],1,nJobs)];
ModelData.R = [0 randi([5 30],1,nJobs)];
ModelData.D = vertcat(zeros(1,tPeriod), randi([10 40],nJobs,tPeriod));
ModelData.G = vertcat(zeros(1,tPeriod), randi([10 50],nJobs,tPeriod));
ModelData.O = vertcat(zeros(1,tPeriod), randi([10 50],nJobs,tPeriod));
ModelData.BigM = 100000;
ModelData.WorkHour = 15000;
ModelData.LowerLotSize = 1;

S = ones(nJobs) - eye(nJobs);
S = horzcat(zeros(nJobs,1), S);
S = vertcat(zeros(1,nJobs+1), S);
S = randi([5 20],nJobs+1,nJobs+1) .* S;
ModelData.S = S;

minJobs_PerMach = 0;
minMach_PerJobs = 0;
while(~((minJobs_PerMach > 2) && (minMach_PerJobs > 2)))
    E = vertcat(ones(1,mMachine), randi([0 1], nJobs, mMachine));
    Sum_AllRow = sum(E,1);
    minMach_PerJobs = min(Sum_AllRow);
    Sum_AllCol = sum(E,2);
    minJobs_PerMach = min(Sum_AllCol);
end
ModelData.E = E;

nameDataSave  =
sprintf('MLTest%s_%02dJ%02dM%02dP.mat',test,nJobs+1,mMachine,tPeriod);
nameLINGOSave =
sprintf('LGTest%s_%02dJ%02dM%02dP.lg4',test,nJobs+1,mMachine,tPeriod);
save (nameDataSave, 'ModelData');
clearvars -except nameDataSave nameLINGOSave;
load(nameDataSave)
Generate_Save_LINGO(ModelData,nameLINGOSave)
end

%-----------------------------------------------------------
function Code = MatrixDisplay(nameX,X,nRow)
    Code = nameX;
    for i=1:nRow
        Code = Code + sprintf("\t");
        Code = Code + sprintf(" %3d",X(i,:));
        if(i==nRow)
            Code = Code + sprintf(";\n\n");
        else
            Code = Code + newline;
        end
    end
end

%-----------------------------------------------------------
function Generate_Save_LINGO(ModelData,nameLINGOSave)
    %fprintf(nameLINGOSave);
    fileSave = fopen(nameLINGOSave,'wt');

    Code = sprintf("\nMODEL:\n\n");
    Code = Code + sprintf("DATA:\n");
    Code = Code + sprintf(" nJobs = %d;\n", ModelData.nJobs);
    Code = Code + sprintf(" nMachines = %d;\n", ModelData.mMachine);
    Code = Code + sprintf(" nPeriods = %d;\n", ModelData.tPeriod);
    Code = Code + sprintf(" BigM = %ld;\n", ModelData.BigM);
    Code = Code + sprintf(" WorkHour = %d;\n", ModelData.WorkHour);
    Code = Code + sprintf(" LowerLotSize = %d;\n", ModelData.LowerLotSize);
```

```
    Code = Code + sprintf(" RunTime = 0;\n");
    Code = Code + sprintf("ENDDATA\n\n");
    fprintf(fileSave, Code);
    disp(Code);

    Code = sprintf("SETS:\n");
    Code = Code + sprintf(" JOB/1..nJobs/: P, R;\n");
    Code = Code + sprintf(" MC/1..nMachines/;\n");
    Code = Code + sprintf(" PED/1..nPeriods/;\n");
    Code = Code + sprintf(" FT( JOB, PED): C, H;\n");
    Code = Code + sprintf(" CXC( JOB, JOB, MC, PED): X;\n");
    Code = Code + sprintf(" DXD( JOB, MC, PED): Y, Z;\n");
    Code = Code + sprintf(" NUM( JOB, MC, PED): N;\n");
    Code = Code + sprintf(" LINK( JOB, JOB): S;\n");
    Code = Code + sprintf(" HOL( JOB, PED): O;\n");
    Code = Code + sprintf(" PROD( JOB, PED): G;\n");
    Code = Code + sprintf(" WorkLoad( JOB, MC): E;\n");
    Code = Code + sprintf(" DEMAND ( JOB, PED): D;\n");
    Code = Code + sprintf("ENDSETS\n\n");
    fprintf(fileSave, Code);
    disp(Code);

    Code = sprintf("DATA:\n");
    Code = Code + sprintf("P =");
    Code = Code + sprintf(" %3d",ModelData.P);
    Code = Code + sprintf(";\n");
    Code = Code + sprintf("R =");
    Code = Code + sprintf(" %3d",ModelData.R);
    Code = Code + sprintf(";\n\n");
    fprintf(fileSave, Code);
    disp(Code);

    Code = MatrixDisplay('D =', ModelData.D, ModelData.nJobs);
    Code = Code + MatrixDisplay('G =', ModelData.G, ModelData.nJobs);
    Code = Code + MatrixDisplay('O =', ModelData.O, ModelData.nJobs);
    Code = Code + MatrixDisplay('S =', ModelData.S, ModelData.nJobs);
    Code = Code + MatrixDisplay('E =', ModelData.E, ModelData.nJobs);
    Code = Code + sprintf("ENDDATA\n\n");
    fprintf(fileSave, Code);
    disp(Code);
    fclose(fileSave);

    cmdRUN = ['copy ',nameLINGOSave,'+B21_LINGO_Code.txt ',nameLINGOSave];
    system(cmdRUN);

end
```

2/2-ไฟล์ข้อมูลประกอบการสร้างโค้ดโปรแกรม “B21_LINGO_Code.txt”

```
!=========================================================================;

        MIN =   @Sum( JOB( J) : @SUM( PED( T) : G( J, T) * C( J, T) ) ) +
                @Sum( JOB( J) : @SUM( PED( T) : O( J, T) * H( J, T) ) ) +
                @Sum(  MC( M) : @SUM( JOB( I) : @SUM( JOB( J): @SUM( PED( T)| I #NE# J #AND# J#NE#1 : S( I, J) * X(
I, J, M, T) ) ) ) );

        @FOR( JOB( J) :
                H( J, 1) = @Sum( MC( M): N( J, M, 1)) - D( J, 1) ;
        );

        @FOR( JOB( J) : @FOR( PED( T) | J#NE#1 #AND# T#GE#2:
                H( J, T) = H( J, T-1) + @Sum( MC( M): N( J, M, T)) - D( J, T) ;
        ); );

        @FOR( MC( M) : @FOR( JOB( J): @FOR( PED( T) | J#NE#1 :
                @Sum( JOB( I) | I#NE#J : X( I, J, M, T) ) = 1 - Y( J, M, T) ;
```

```
        ); );

        @FOR( MC( M) : @FOR( PED( T) :
                @SUM( JOB( I) | I#NE#1 : X( I, 1, M, T) * E( I, M) ) = 1;
        ); );

        @FOR( JOB( I) : @FOR( JOB( J) | J#NE#I: @FOR( PED( T) :
                @SUM( MC( M) : ( X( I, J, M, T) + X(J, I, M, T) )* E( I, M)* E( J, M) ) <= 1;
        ); ); );

        @FOR( PED( T) :
                @SUM( JOB( J) | J#NE#1 : @SUM( MC( M) : X( 1, J, M, T)* E( J, M) ) ) = nMachines ;
        );

        @FOR( PED( T) :
                @SUM( JOB( I) | I#NE#1 : @SUM( MC( M) : X( I, 1, M, T)* E( I, M) ) ) = nMachines ;
        );

        @FOR( CXC: @BIN( X));
        @FOR( DXD: @BIN( Y));
        @FOR( DXD: @BIN( Z));
        @FOR( NUM: @GIN( N));
        @FOR( FT: @GIN( C));
        @FOR( FT: @GIN( H));

END
```

ภาคผนวก ข.3 MATLAB-Function

1/2-ไฟล์ MATLAB – “FncPrepare10_GenerateSEED.m”

```
% function prepair
function testAllSeq = FncPrepare10_GenerateSEED(nJobs)
    nJobs = nJobs - 1;
    nFullRun = factorial(nJobs);
    nNowRun = 0;
    while nFullRun ~= nNowRun
        for k1 = 1:10*nFullRun
            caseRun(k1,:) = randperm(nJobs,nJobs);
            if(mod(k1,1000) == 0) fprintf("Line = %d\n",k1);, end
        end
        caseRun = unique(caseRun, 'rows');
        nNowRun = size(caseRun,1);
        %fprintf("Data = %d, nFact > %d, RunCase > %d\n",nJobs, nFullRun, nNowRun)
    end
    runSequence = caseRun + 1;
    SeqX = runSequence';
    Seq1 = ones(1,size(runSequence,1));
    testAllSeq = vertcat(Seq1,SeqX);
```

2/5-ไฟล์ MATLAB – “FncPrepare20_ExtractResultLGR.m”

```
%=================================================================
function resultModel = FncPrepare20_ExtractResultLGR(fileName)
    filetext = fileread(fileName);
    resultModel.BigM        = GetData(filetext,'BIGM');
    resultModel.nJobs       = GetData(filetext,'NJOBS');
    resultModel.nMachines   = GetData(filetext,'NMACHINES');
    resultModel.nPeriods    = GetData(filetext,'NPERIODS');
    resultModel.WorkHour    = GetData(filetext,'WORKHOUR');
    resultModel.LowerLotSize = GetData(filetext,'LOWERLOTSIZE');

    resultModel.ObjValue    = GetData(filetext,'Objective value:');
    resultModel.SolverStep  = GetData(filetext,'solver steps:');
    resultModel.SolverIter  = GetData(filetext,'solver iterations:');

    nJobs = resultModel.nJobs;
    nPeriod = resultModel.nPeriods;
    nMachines = resultModel.nMachines;

    %P(J)   Processing Times
    %R(J)   Release Time
    %D(J,P) Demand
    %G(J,P) Production Course
    %O(J,P) Hold course
    %S(I,J) Setup Times
    %E(J,M) Machine Eligibility
    resultModel.ArrayP = Get1DArray(nJobs,filetext,'P( ');
    resultModel.ArrayR = Get1DArray(nJobs,filetext,'R( ');
    resultModel.ArrayD = Get2DArray(nJobs,nPeriod,filetext,'D( ');
    resultModel.ArrayG = Get2DArray(nJobs,nPeriod,filetext,'G( ');
    resultModel.ArrayO = Get2DArray(nJobs,nPeriod,filetext,'O( ');
    resultModel.ArrayS = Get2DArray(nJobs,nJobs,filetext,'S( ');
    resultModel.ArrayE = Get2DArray(nJobs,nMachines,filetext,'E( ');

    %C(J,P)
    %H(J,P) Hold Quntity
    %N(I,J) Product Quntity
    %X(I,J,M,P) Sequence Process
    resultModel.ArrayC = Get2DArray(nJobs,nPeriod,filetext,'C( ');
    resultModel.ArrayH = Get2DArray(nJobs,nPeriod,filetext,'H( ');
    resultModel.ArrayN = Get3DArray(nJobs,nMachines,nPeriod,filetext,'N( ');
    resultModel.ArrayX = Get4DArray(nJobs,nJobs,nMachines,nPeriod,filetext,'X( ');

    resultModel.RunTime.Str = GetStrTime(filetext);
    resultModel.RunTime.HH  = str2num(resultModel.RunTime.Str(1:2));
    resultModel.RunTime.MM  = str2num(resultModel.RunTime.Str(4:5));
    resultModel.RunTime.SS  = str2num(resultModel.RunTime.Str(7:8));

end

function Result = GetData(filetext,RefString)
    StrPost = strfind(filetext,RefString);
    StrFrm = StrPost + size(RefString,2);
    StrEnd = strfind(filetext,newline);
    E_Result = StrEnd - StrPost(1);
    E_Result (E_Result <=0) = nan;
    StrEnd = StrPost + min(E_Result) - 1;
    Result = filetext(StrFrm:StrEnd);
    Result = str2num(Result);
    Result = Result(1);
end

function Result = Get1DArray(cLoop,filetext,RefString)
    Result = zeros(1, cLoop);
    for i=1:cLoop
        strRef = sprintf('%d)',i);
        strRef = horzcat(RefString, strRef);
        Result(i) = GetData(filetext,strRef);
    end
end
```

```
function Result = Get2DArray(jLoop,pLoop,filetext,RefString)
    Result = zeros(jLoop, pLoop);
    for j=1:jLoop
        for p=1:pLoop
            strRefJ = sprintf('%d, ',j);
            strRefP = sprintf('%d)',p);
            strRef = horzcat(RefString, strRefJ, strRefP);
            Result(j,p) = GetData(filetext,strRef);
        end
    end
end

function Result = Get3DArray(jLoop,mLoop,pLoop,filetext,RefString)
    Result = zeros(jLoop, mLoop, pLoop);
    for j=1:jLoop
        for m=1:mLoop
            for p=1:pLoop
                strRefJ = sprintf('%d, ',j);
                strRefM = sprintf('%d, ',m);
                strRefP = sprintf('%d)',p);
                strRef = horzcat(RefString, strRefJ, strRefM, strRefP);
                Result(j,m,p) = GetData(filetext,strRef);
            end
        end
    end
end

function Result = Get4DArray(jLoop,iLoop,mLoop,pLoop,filetext,RefString)
    Result = zeros(jLoop,iLoop,mLoop,pLoop);
    for j=1:jLoop
        for i=1:iLoop
            for m=1:mLoop
                for p=1:pLoop
                    strRefJ = sprintf('%d, ',j);
                    strRefI = sprintf('%d, ',i);
                    strRefM = sprintf('%d, ',m);
                    strRefP = sprintf('%d)',p);
                    strRef = horzcat(RefString, strRefJ, strRefI, strRefM, strRefP);
                    Result(j,i,m,p) = GetData(filetext,strRef);
                end
            end
        end
    end
end

function Result = GetStrTime(filetext)
    RefString = 'RUNTIME';
    StrPost = strfind(filetext,RefString);
    StrFrm = StrPost + size(RefString,2);
    StrEnd = StrFrm + 18;
    Result = filetext(StrFrm:StrEnd);
    Result = strrep(Result,' ','');
end
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

3/5-ไฟล์ MATLAB – “FncPrepare30_ReadDataMATLAB.m”

```
function resultModel = FncPrepare30_ReadDataMATLAB(fileName)
    load(fileName);

    resultModel.nJobs        = ModelData.nJobs;
    resultModel.nMachines    = ModelData.mMachine;
    resultModel.nPeriods     = ModelData.tPeriod;

    resultModel.BigM         = ModelData.BigM;
    resultModel.WorkHour     = ModelData.WorkHour;
    resultModel.LowerLotSize = ModelData.LowerLotSize;

    resultModel.ArrayP = ModelData.P;
    resultModel.ArrayR = ModelData.R;
    resultModel.ArrayD = ModelData.D;
    resultModel.ArrayG = ModelData.G;
    resultModel.ArrayO = ModelData.O;
    resultModel.ArrayS = ModelData.S;
    resultModel.ArrayE = ModelData.E;
 end
```

4/5-ไฟล์ MATLAB – “FncRun00_RunSeq_SeedQtt_2_ObjectiveValue.m”

```
function [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(testAllSeq,
FactData, tmpRunIndex, tmpRunQSeed)
    testQtt = FncRun0020_EOQ_2_testQtt(FactData, tmpRunQSeed);
    testSeq = FncRun0030_IndexSeq_2_testSeq(testAllSeq, tmpRunIndex, testQtt);

%========== EOQ to testQuantity ========================================
    function testQtt = FncRun0020_EOQ_2_testQtt(FactData, tmpEOQ)
        nJobs = FactData.nJobs;
        nMachine = FactData.nMachines;
        nPeriod  = FactData.nPeriods;

        testQtt = zeros(nJobs, nPeriod*nMachine);

        runEOQ     = tmpEOQ';
        runRemain  = sum(FactData.ArrayD,2);
        runDemand  = FactData.ArrayD(:,1);
        runProduce = min(max(runDemand,runEOQ),runRemain);
        runHoldPrd = runProduce - runDemand;
        XX = horzcat(runRemain,runDemand,runProduce,runHoldPrd);
        tX = runProduce;
        for pPrd = 2:nPeriod
            runRemain = runRemain - runProduce;
            runDemand  = FactData.ArrayD(:,pPrd);
            runDemand = runDemand - runHoldPrd;
            runDemand(runDemand < 0) = 0; % if negative setto 0
            runProduce = min(max(runDemand,runEOQ),runRemain);
            runHoldPrd = runProduce - runDemand;
            XX = horzcat(XX,runRemain,runDemand,runProduce,runHoldPrd);
            tX = horzcat(tX,runProduce);
        end
        % XX, tX, sum(tX,2)

        testQtt = FactData.ArrayE .* tX(:,1);
        for jPrd = 2:nPeriod
            testQtt = horzcat(testQtt, FactData.ArrayE.*tX(:,jPrd));
        end
        % testQtt, FactData.ArrayE, sum(tX,2), sum(testQtt,2)

        ESumm  = sum(FactData.ArrayE,2);
        ECount = ones(1,nMachine);
        for iPerd = 1:nPeriod
            for jJobs = 2:nJobs
```

```
                ETest(1,1:nMachine) = testQtt(jJobs,(iPerd-1)*2+(1:nMachine));
                if ESumm(jJobs)>1
                    tF = find(ECount==min(ECount),1);
                    maskETest = zeros(1,nMachine);
                    maskETest(1,tF) = 1;
                    ETest = ETest .* maskETest;
                    testQtt(jJobs,(iPerd-1)*2+(1:nMachine)) = ETest;
                end
                ECount = ECount + ETest .* FactData.ArrayP(1,jJobs);
            end
        end
        % testQtt, sum(testQtt,2), sum(tX,2)
    end

%========== testQuantity to Index Sequence(Matrix.X) ====================
    function testSeq = FncRun0030_IndexSeq_2_testSeq(testAllSeq, testIndex, testQtt)
        [nJobs, nLoop] = size(testQtt);
        testSeq = zeros(nJobs,nLoop);
        tempFrm = vertcat([1], zeros(nJobs-1,1));

        for kLoop = 1:nLoop
            tempSeqOut  = tempFrm;
            kIndexSave = 2;
            for jJobs = 2:nJobs
                indexJobTest = testAllSeq(jJobs,testIndex(kLoop));
                if testQtt(indexJobTest,kLoop)~= 0
                    tempSeqOut(kIndexSave) = indexJobTest;
                    kIndexSave = kIndexSave + 1;
                end
            end
            testSeq(:,kLoop) = tempSeqOut;
        end
    end

end
```

5/5 ไฟล์ MATLAB – “FncRun10_SEED_2_ObjValue.m”

```
% function run
function [ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData)
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    ObjX = Calc_ObjectiveX(testSeq, testQtt, FactData);
    ObjY = Calc_ObjectiveY(testSeq, testQtt, FactData);
    ObjZ = Calc_ObjectiveZ(testSeq, testQtt, FactData);

    %-------------------------------------------------
    function ObjY = Calc_ObjectiveY(~, testQtt, FactData)
        % ObjY = sum(sum(O.*H));
        tHold = zeros(nJobs,nPeriod);
        tHBf0 = zeros(nJobs,1);
         for iPeriod = 1 : nPeriod
            indexF = (iPeriod-1) * nMachine + 1;
            indexT = indexF + nMachine - 1;
            tPrd1 = sum(testQtt(:,indexF:indexT),2);
            tHBf0 = tPrd1 + tHBf0 - FactData.ArrayD(:,iPeriod);
            tHold(:,iPeriod) = tHBf0;
        end
        ObjY = sum(sum(FactData.ArrayO.*tHold));
    end

    %-------------------------------------------------
    function ObjZ = Calc_ObjectiveZ(testSeq, ~, FactData)
        % ObjZ = sum(sum(sum(sum(S.*X))));
        kReadIndex = 0;
        tX = zeros(nJobs,nJobs,nMachine,nPeriod);
```

```
        for iPeriod=1:nPeriod
            for iMachine=1:nMachine
                kReadIndex = kReadIndex+1;
                iFrom = testSeq(1,kReadIndex);
                for kLoop=2:nJobs
                    iNext = testSeq(kLoop,kReadIndex);
                    if(iNext~=0)
                        tX(iFrom,iNext,iMachine,iPeriod) = 1;
                        iFrom = iNext;
                    end
                end
            end
        end
        ObjZ = sum(sum(sum(sum(FactData.ArrayS.*tX))));
    end

    %-------------------------------------------------
    function ObjX = Calc_ObjectiveX(testSeq, testQtt, FactData)
        % ObjX = sum(sum(FactData.ArrayG.*FactData.ArrayC))
        kReadIndex = 0;
        tmpArryC = zeros(nJobs,nPeriod);
        for kPrd=1:nPeriod
            for jMach=1:nMachine
                xFrom = 1;
                CTotal = 0;
                kReadIndex = kReadIndex + 1;
                for iJobs=2:nJobs
                    xNext = testSeq(iJobs,kReadIndex);
                    if ~((xNext == 0)||(xNext == 1))
                        CRelease = max(CTotal, FactData.ArrayR(xNext));
                        CSetup   = FactData.ArrayS(xFrom,xNext);
                        CRunTime = testQtt(xNext,kReadIndex)*FactData.ArrayP(xNext);
                        CTotal = CRelease + CSetup + CRunTime;
                        tmpArryC(xNext,kPrd) = CTotal;
                        xFrom = xNext;
                    end
                end
            end
        end
        % tmpArryC
        % FactData.ArrayC - tmpArryC
        ObjX = sum(sum(FactData.ArrayG.*tmpArryC));
    end


end
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

ภาคผนวก ข.4 MATLAB ANS Code

1/2-ไฟล์ MATLAB – “A20y_ANS_PureRun.m”

```
%%
%-------------------------------------------------------------------------
% clc, clear all, close all
function A20y_ANS_PureRun(testCaseName,zRunLoop)

tic, clc, close all force;
clearvars -except testCaseName zRunLoop;

% testCaseName = "Test21a_06J04M10P";
% testCaseName = "Test21b_06J04M10P";
% testCaseName = "Test21c_06J04M10P";
% testCaseName = "Test21d_06J04M10P";

%-----------------------------------------
fileName = sprintf('ML%s.mat',testCaseName);
RefFName = sprintf('LG%s.lgr',testCaseName);
logFileName = sprintf('RD%s_%s.log', testCaseName,datestr(now,'yyyymmdd_HHMM'));
datFileName = replace(logFileName,".log",".mat");
diary(logFileName);

%%
%-------------------------------------------------------------------------
fprintf("\n######  Step1/5: Run-1of2 > ANN Prepair Data ############\n");
FactData = FncPrepare30_ReadDataMATLAB(fileName);
FactData.nSaveAns = 10;

testAllSeqName = sprintf('DataRef%02dJ_testAllSeq.mat',FactData.nJobs);
fprintf(">> Test Case = "), disp(testCaseName);
fprintf(">> File Data = "), disp(testAllSeqName);
fprintf(">> log File  = "), disp(logFileName);
fprintf(">> data File = "), disp(datFileName);
fprintf(">> Ref File Name = "), disp(RefFName);
fprintf(">> Run File Name = "), disp(fileName);
if ~(isfile(testAllSeqName))
    testAllSeq = FncPrepare10_GenerateSEED(FactData.nJobs);
    save(testAllSeqName, 'testAllSeq');
end
load(testAllSeqName);

%%
%-------------------------------------------------------------------------
fprintf("\n######  Step2/5: run model ################\n");
StopCondition.LoopTestRun   = zRunLoop;
StopCondition.nRepeatAnswer = 35;
StopCondition.LoopPctError  = 1.0; % 12.0;
ObjMX  = 50; % 275865;
ObjMY  = 25; % 7928;
ObjMZ  = 25; % 168;
StopCondition.LoopTarget = ObjMX + ObjMY + ObjMZ;

fprintf(">> max.LoopRun  = %d\n",StopCondition.LoopTestRun);
fprintf(">> max.nRepeat  = %d\n",StopCondition.nRepeatAnswer);
fprintf(">> min.Target   = %d\n",StopCondition.LoopTarget);
fprintf(">> %%.TargetDiff = %f\n",StopCondition.LoopPctError);
ResultData = A21a_ANS_Subr(testAllSeq, FactData, StopCondition);
disp('>> Result >> '), disp(ResultData.Answer);

%%
fprintf("\n######  Step3/5: display resul ##########\n");
tmpBestObjt = ResultData.Answer(1,:);
tmpRunQSeed = ResultData.SeedQtty(1,:);
tmpRunIndex = ResultData.Sequence(1,:);
[testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
    testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
%Dbg> testSeq
```

```
%%
%Dbg> testQtt

[ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData);
ObjectiveFnc = ObjX + ObjY + ObjZ;
fprintf(">> MATLAB >> Calc = %d",tmpBestObjt);
fprintf("\n  >>> best Seed Qtty = "), fprintf("%d,", tmpRunQSeed);
fprintf("\n  >>> best Run Index = "), fprintf("%d,", tmpRunIndex);
fprintf("\n  >>> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n");

ObjMAll = ObjMX + ObjMY + ObjMZ;
fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
fprintf("\n");


%%
fprintf("\n######  Step4/5: compare Math Model #####\t");
if isfile(RefFName)
    FactData = FncPrepare20_ExtractResultLGR(RefFName);
    ObjMX = sum(sum(FactData.ArrayG.*FactData.ArrayC));
    ObjMY = sum(sum(FactData.ArrayO.*FactData.ArrayH));
    ObjMZ = sum(sum(sum(sum(FactData.ArrayS.*FactData.ArrayX))));
    ObjMAll = ObjMX + ObjMY + ObjMZ;
    fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
    fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
    fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
    fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
    fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
    fprintf("\n");
else
    fprintf("\n  ->> Warning: file does not exist");
    fprintf("\n  ->>          ""%s""\n", RefFName);
end

%% 319653 321751
fprintf("\n######  Step5/5: run time ###############\n");
fprintf(">> "),toc
diary off
save(datFileName)
end
```

2/2-ไฟล์ MATLAB – “A21a_ANS_Subr.m”

```
function globalAnswer = A21a_ANS_Subr(testAllSeq, FactData, StopCondition)
    %% ---- initial parameter data ----
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    nRunCase = size(testAllSeq,2);  % Maximun of run case
    nBestAns = FactData.nSaveAns;   % number of best answer store

    globalAnswer.Sequence = zeros(nBestAns, nMachine*nPeriod);
    globalAnswer.SeedQtty = zeros(nBestAns, nJobs);
    globalAnswer.Answer   = ones(nBestAns, 1)*9999999;

    %% ---- initial best sequence ----
    [startModelSeq, startModelQtt] = Initial_ANS_Seed(FactData);
    globalAnswer.Sequence(1,:) = startModelSeq;
    globalAnswer.SeedQtty(1,:) = startModelQtt;
    %Dbg> disp(globalAnswer.Sequence(1,:));
    %Dbg> disp(globalAnswer.SeedQtty(1,:));

     %% ---- ready start -----
```

```
    pctRefD = 0;
    globalAnswer.SeqMoveStep    = 1;   % Move Sqeuence 1 --> -1,0,+1
    globalAnswer.PctQttStepMove = 20;  % Percent Move Qtt 0-100% of EOQ

    %% ---- strat progress bar ----
    H_WaitBar = waitbar(0,'Initializing waitbar...');
    oldBestAnswer = 9999999;
    nRepeatCounter = 0;
    for iLoopRun = 1:StopCondition.LoopTestRun
        perc = 100*iLoopRun/StopCondition.LoopTestRun;
        if( perc >= pctRefD )
            waitbar(perc/100,H_WaitBar,sprintf('%3.1f%% along...',perc))
            pctRefD = pctRefD + 2.33;
        end

        %-----------------------------------------------
        if(mod(iLoopRun,100) == 0)
            %------------------
            xError = globalAnswer.Answer(1) - StopCondition.LoopTarget;
            xError = xError * 100 / StopCondition.LoopTarget;
            %------------------
            if(oldBestAnswer==LocalAnswer.BestAnswer)
                nRepeatCounter = nRepeatCounter + 1;
            else
                oldBestAnswer =  LocalAnswer.BestAnswer;
                nRepeatCounter = 0;
            end
            %------------------
            fprintf("  ->> %6d(%6d), %3d(%3d), %8.4f(%8.4f)\n"...
               ,iLoopRun,StopCondition.LoopTestRun...
               ,nRepeatCounter,StopCondition.nRepeatAnswer...
               ,xError,StopCondition.LoopPctError);
            %-----------------------------------------------
            if(nRepeatCounter >= StopCondition.nRepeatAnswer)
                fprintf("Break Repeat Condition = %3d(%3d)\n"...
                    ,nRepeatCounter,StopCondition.nRepeatAnswer);
                break; % for iLoopRun
            end
            %-----------------------------------------------
            if (xError <= StopCondition.LoopPctError)
                fprintf("Break Diff Error = %8.4f(%8.4f)\n"...
                    ,xError,StopCondition.LoopPctError);
                break; % for iLoopRun
            end
        end

        testData.bestSequence  = globalAnswer.Sequence(1,:);
        testData.bestSeedQtty  = globalAnswer.SeedQtty(1,:);
        testData.SeqMoveStep   = globalAnswer.SeqMoveStep;
        testData.QttMoveStep   =
round(globalAnswer.PctQttStepMove/100*mean(testData.bestSeedQtty));
        testData.maxSeqRunCase = size(testAllSeq,2);      % Maximun of sequence run
case
        testData.maxQttRunCase = sum(FactData.ArrayD,2)'; % Maximun of Qtty run case
is All D > Transpose
        % disp(testData.bestSeedQtty)

        testCase = GenTestCase(testData);
        % disp("testCase.testAnsw = "), disp(testCase.testAnsw);

        testCase = CalcObjectiveValue(testCase);
        % disp("testCase.testAnsw = "), disp(testCase.testAnsw);

        ObjValueMin = min(min(testCase.testAnsw));
        % disp("ObjValueMin = "), disp(ObjValueMin);

        [MinRow,MinCol] = find(testCase.testAnsw == ObjValueMin);
        % disp("position-A = "), disp(MinRow), disp(MinCol);

        MinRow = MinRow(1);
        MinCol = MinCol(1);
```

```
        % disp("position-1 = "), disp(MinRow), disp(MinCol);

        LocalAnswer.BestSequence = testCase.testCase(MinRow,:);
        LocalAnswer.BestSeedQtty = testCase.testQtt(MinCol,:);
        LocalAnswer.BestAnswer   = testCase.testAnsw(MinRow,MinCol);
        % disp("LocalAnswer.BestSequence = "), disp(LocalAnswer.BestSequence);
        % disp("LocalAnswer.BestSeedQtty = "), disp(LocalAnswer.BestSeedQtty);
        % disp("LocalAnswer.BestAnswer = "),   disp(LocalAnswer.BestAnswer);

        globalAnswer = ReplaceLocal2AnwsArray(globalAnswer, LocalAnswer);
    end
    close(H_WaitBar)
%=========================================================================
%=========================================================================

 %% ====== Force Start ========================================
function [startModelSeq, startModelQtt] = Initial_ANS_Seed(FactData)
    meanArryD =  sum(FactData.ArrayD,2) ./ nPeriod ;  % row summ
    meanArryO =  sum(FactData.ArrayO,2) ./ nPeriod ;  % row summ
    meanArryS =  sum(FactData.ArrayS,1)' ./ nPeriod ; % col summ and transpose
    Model_EOQ = ceil(meanArryD).*2;
    Model_EOQ(isnan(Model_EOQ)) = 0;
    %Dbg> Model_EOQ              % j-Row,single-Col
    %Dbg> FactData.ArrayD

    startModelQtt = Model_EOQ';     % Transpose
    startModelSeq = randi(nRunCase, nMachine*nPeriod,1);
    startModelSeq = startModelSeq';
    % disp(startModelSeq);
    % disp(startModelQtt);

    % save ('XX.mat', 'startModelQtt', 'startModelSeq');
    % load ('XX.mat', 'startModelQtt', 'startModelSeq');
    % disp(startModelSeq);
    % disp(startModelQtt);
end

 %% ====== Replace new answer and sort result ====================
    function globalAnswer = ReplaceLocal2AnwsArray(globalAnswer, LocalAnswer)
        l_Sequence = LocalAnswer.BestSequence;
        l_SeedQtty = LocalAnswer.BestSeedQtty;
        l_Answer   = LocalAnswer.BestAnswer;
        if(globalAnswer.Answer(1,1) == l_Answer)
            globalAnswer.QttyMoveStep = globalAnswer.QttyMoveStep + 5;
            if(globalAnswer.QttyMoveStep > 100)
                globalAnswer.SeqMoveStep  = globalAnswer.SeqMoveStep + 2;
                globalAnswer.QttyMoveStep = 7;
            end
        else
            globalAnswer.QttyMoveStep = 7;
            globalAnswer.SeqMoveStep = 10;
            for iSort=1:size(globalAnswer.Sequence,1)
                g_Answer = globalAnswer.Answer(iSort,1);
                if(l_Answer < g_Answer)
                    tmp_Sequence = globalAnswer.Sequence(iSort,:);
                    tmp_SeedQtty = globalAnswer.SeedQtty(iSort,:);
                    tmp_Answer   = globalAnswer.Answer(iSort,1);
                    globalAnswer.Sequence(iSort,:) = l_Sequence;
                    globalAnswer.SeedQtty(iSort,:) = l_SeedQtty;
                    globalAnswer.Answer(iSort,1)   = l_Answer;
                    l_Sequence = tmp_Sequence;
                    l_SeedQtty = tmp_SeedQtty;
                    l_Answer   = tmp_Answer;
                end
            end
        end
    end

 %% ====== Calculate Objective Value from Data ====================
   function testData = CalcObjectiveValue(testData)
       for iRowSeq = 1:  size(testData.testAnsw,1)      % row is test case sequence
```

```
            for jColQty = 1:  size(testData.testAnsw,2)  % col is test case qtty
                tmpRunIndex = testData.testCase(iRowSeq,:);
                tmpRunQSeed = testData.testQtt(jColQty,:);
                if(sum(tmpRunQSeed) == 0)
                    testData.testAnsw(iRowSeq,jColQty) = 9999999;
                else
                    [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
                        testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
                    [ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt,
FactData);
                    ObjectiveFnc = ObjX + ObjY + ObjZ;
                    testData.testAnsw(iRowSeq,jColQty) = ObjectiveFnc;
                    if ((ObjX <= 0)||(ObjY <= 0)||(ObjZ <= 0)||(ObjectiveFnc <= 0))
                         testData.testAnsw(iRowSeq,jColQty) = 9999999;
                    end
                end
              end
          end
     end

%% ====== Generate test Case from seed and step move =============
     function rtnData = GenTestCase(testData)
         %-----------------------------------------------
         tmpTestCase = testData.bestSequence;
         maxSeqData  = testData.maxSeqRunCase;
         moveStepSeq = testData.SeqMoveStep;
         for ix = 1: size(testData.bestSequence,2)
             testCaseInc = testData.bestSequence; % load all frame
             testCaseDec = testData.bestSequence; % load all frame
             testCaseInc(ix) =
StepWithRange(1,maxSeqData,testCaseInc(ix),moveStepSeq);    % change 1 position
             testCaseDec(ix) =
StepWithRange(1,maxSeqData,testCaseDec(ix),moveStepSeq*-1); % change 1 position
             tmpTestCase = vertcat(tmpTestCase, testCaseInc, testCaseDec);
         end
         rtnData.testCase = tmpTestCase;
         % disp(rtnData.testCase)       % test case on parent loop
         %-----------------------------------------------

         frameTest = [-7, -5,-3,-1, 0, 1, 3, 5, 7];
         ansRow = size(tmpTestCase,1);
         ansCol = size(frameTest,2);
         rtnData.testAnsw = zeros(ansRow,ansCol);
         % disp(rtnData.testAnsw)       % test amswer on parent loop
         %-----------------------------------------------

         ansRow = size(frameTest,2);
         ansCol = size(testData.bestSeedQtty,2);
         tempTestQtty = zeros(ansRow,ansCol);

         RefTestQtty  = testData.bestSeedQtty;
         maxQttyData  = testData.maxQttRunCase;
         moveStepQtty = testData.QttMoveStep;

         % one for random
         tmpQtty = RefTestQtty;
         for xi=2:ansCol, tmpQtty(xi) = randi(maxQttyData(xi)); end
         tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
         tempTestQtty(1,:) = tmpQtty;
         % two for random
         tmpQtty = RefTestQtty;
         for xi=2:ansCol, tmpQtty(xi) = randi(maxQttyData(xi)); end
         tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
         tempTestQtty(2,:) = tmpQtty;
         % etc for gen nornal
         for iRow = 3:ansRow
             tmpQtty = RefTestQtty + frameTest(iRow)*moveStepQtty;
             tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
             tempTestQtty(iRow,:) = tmpQtty;
         end
```

```
            tempTestQtty(iRow,:) = tmpQtty;
        end

        rtnData.testQtt = tempTestQtty;
        %disp(rtnData.testQtt);
        %pause

        %-------------------------------------------
    end

%% ====== Update in Range [Min-Max] Ring Counter =================
    function result = StepWithRange(Min,Max,From,Step)
        dTemp = From - Min ;
        mTemp = Max - Min + 1 ;
        dTemp = dTemp + Step ;
        dTemp = mod(dTemp, mTemp);
        result = dTemp + Min ;
    end
    function iData = RangeCutQtty(Min,Max,iData)
        for i=1:size(iData,2)
            dataTemp = iData(i);
            dataTemp(dataTemp > Max(i)) = Max(i);
            dataTemp(dataTemp < Min(i)) = Min(i);
            iData(i) = dataTemp;
        end
    end

end
```

ภาคผนวก ข.5 MATLAB MTS Code

1/2-ไฟล์ MATLAB – “A20x_MTS_PureRun.m”

```
%%
%-------------------------------------------------------------------------
% clc, clear all, close all
function A20x_MTS_PureRun(testCaseName, zRunLoop)

tic, clc, close all force;
clearvars -except testCaseName zRunLoop;


% testCaseName = "Test21a_06J04M10P";
% testCaseName = "Test21b_06J04M10P";
% testCaseName = "Test21c_06J04M10P";
% testCaseName = "Test21d_06J04M10P";

%-------------------------------------------
fileName = sprintf('ML%s.mat',testCaseName);
RefFName = sprintf('LG%s.lgr',testCaseName);
logFileName = sprintf('MTS%s_%s.log', testCaseName,datestr(now,'yyyymmdd_HHMM'));
datFileName = replace(logFileName,".log",".mat");
diary(logFileName);

%%
%-------------------------------------------------------------------------
fprintf("\n######  Step1/5: Run-2of2 > MTS Prepair Data ############\n");
FactData = FncPrepare30_ReadDataMATLAB(fileName);
FactData.nSaveAns = 10;
FactData.nTabuList = 100;

testAllSeqName = sprintf('DataRef%02dJ_testAllSeq.mat',FactData.nJobs);
```

```
fprintf(">> Test Case = "), disp(testCaseName);
fprintf(">> File Data = "), disp(testAllSeqName);
fprintf(">> log File  = "), disp(logFileName);
fprintf(">> data File = "), disp(datFileName);
fprintf(">> Ref File Name = "), disp(RefFName);
fprintf(">> Run File Name = "), disp(fileName);
if ~(isfile(testAllSeqName))
    testAllSeq = FncPrepare10_GenerateSEED(FactData.nJobs);
    save(testAllSeqName, 'testAllSeq');
end
load(testAllSeqName);

%%
%------------------------------------------------------------------------
fprintf("\n######  Step2/5: run model ###############\n");
StopCondition.LoopTestRun   = zRunLoop;
StopCondition.nRepeatAnswer = 35;
StopCondition.LoopPctError  = 1.0; % 12.0;
ObjMX  = 50; % 275865;
ObjMY  = 25; % 7928;
ObjMZ  = 25; % 168;
StopCondition.LoopTarget = ObjMX + ObjMY + ObjMZ;
fprintf(">> max.LoopRun  = %d\n",StopCondition.LoopTestRun);
fprintf(">> max.nRepeat  = %d\n",StopCondition.nRepeatAnswer);
fprintf(">> min.Target   = %d\n",StopCondition.LoopTarget);
fprintf(">> %%.TargetDiff = %f\n",StopCondition.LoopPctError);

ResultData = A21a_MTS_Subr(testAllSeq, FactData, StopCondition);
disp('>> Result >> '), disp(ResultData.Answer);

%%
fprintf("\n######  Step3/5: display resul ##########\n");
tmpBestObjt = ResultData.Answer(1,:);
tmpRunQSeed = ResultData.SeedQtty(1,:);
tmpRunIndex = ResultData.Sequence(1,:);
[testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
    testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
%Dbg> testSeq
%Dbg> testQtt

[ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData);
ObjectiveFnc = ObjX + ObjY + ObjZ;
fprintf(">> MATLAB >> Calc = %d",tmpBestObjt);
fprintf("\n  >>> best Seed Qtty = "), fprintf("%d,", tmpRunQSeed);
fprintf("\n  >>> best Run Index = "), fprintf("%d,", tmpRunIndex);
fprintf("\n  >>> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n");

ObjMAll = ObjMX + ObjMY + ObjMZ;
fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
fprintf("\n");


%%
fprintf("\n######  Step4/5: compare Math Model #####\t");
if isfile(RefFName)
    FactData = FncPrepare20_ExtractResultLGR(RefFName);
    ObjMX = sum(sum(FactData.ArrayG.*FactData.ArrayC));
    ObjMY = sum(sum(FactData.ArrayO.*FactData.ArrayH));
    ObjMZ = sum(sum(sum(sum(FactData.ArrayS.*FactData.ArrayX))));
    ObjMAll = ObjMX + ObjMY + ObjMZ;
    fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
    fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
    fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
    fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
    fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
    fprintf("\n");
else
```

```
%%
    fprintf("\n  ->> Warning: file does not exist");
    fprintf("\n  ->>          ""%s""\n", RefFName);
end

%% 319653 321751
fprintf("\n######  Step5/5: run time ###############\n");
fprintf(">> "),toc
diary off
save(datFileName)

end
```

2/2-ไฟล์ MATLAB – “A21a_MTS_Subr.m”

```
function globalAnswer = A21a_MTS_Subr(testAllSeq, FactData, StopCondition)
    %% ---- initial parameter data ----
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    nRunCase = size(testAllSeq,2);  % Maximun of run case
    nBestAns = FactData.nSaveAns;   % number of best answer store

    globalAnswer.Sequence = zeros(nBestAns, nMachine*nPeriod);
    globalAnswer.SeedQtty = zeros(nBestAns, nJobs);
    globalAnswer.Answer   = ones(nBestAns, 1)*9999999;

    % Empty TabuList
    runTabuList = zeros(FactData.nTabuList, nMachine*nPeriod+nJobs);

    %% ---- initial best sequence ----
    [startModelSeq, startModelQtt] = Initial_MTS_Seed(FactData);
    globalAnswer.Sequence(1,:) = startModelSeq;
    globalAnswer.SeedQtty(1,:) = startModelQtt;
    %Dbg> disp(globalAnswer.Sequence(1,:));
    %Dbg> disp(globalAnswer.SeedQtty(1,:));

     %% ---- ready start -----
     pctRefD = 0;
     globalAnswer.SeqMoveStep    = 1;   % Move Sqeuence 1 --> -1,0,+1
     globalAnswer.PctQttStepMove = 20;  % Percent Move Qtt 0-100% of EOQ

     %% ---- strat progress bar ----
     H_WaitBar = waitbar(0,'Initializing waitbar...');
     oldBestAnswer = 9999999;
     nRepeatCounter = 0;
     for iLoopRun = 1:StopCondition.LoopTestRun
         perc = 100*iLoopRun/StopCondition.LoopTestRun;
         if( perc >= pctRefD )
             waitbar(perc/100,H_WaitBar,sprintf('%3.1f%% along...',perc))
             pctRefD = pctRefD + 2.33;
         end

         %-----------------------------------------------
         if(mod(iLoopRun,100) == 0)
             %------------------
             xError = globalAnswer.Answer(1) - StopCondition.LoopTarget;
             xError = xError * 100 / StopCondition.LoopTarget;
             %------------------
             if(oldBestAnswer==LocalAnswer.BestAnswer)
                 nRepeatCounter = nRepeatCounter + 1;
             else
                 oldBestAnswer =  LocalAnswer.BestAnswer;
                 nRepeatCounter = 0;
             end
             %------------------
```

```
function globalAnswer = A21a_MTS_Subr(testAllSeq, FactData, StopCondition)
    %% ---- initial parameter data ----
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    nRunCase = size(testAllSeq,2);  % Maximun of run case
    nBestAns = FactData.nSaveAns;   % number of best answer store

    globalAnswer.Sequence = zeros(nBestAns, nMachine*nPeriod);
    globalAnswer.SeedQtty = zeros(nBestAns, nJobs);
    globalAnswer.Answer   = ones(nBestAns, 1)*9999999;

    % Empty TabuList
    runTabuList = zeros(FactData.nTabuList, nMachine*nPeriod+nJobs);

    %% ---- initial best sequence ----
    [startModelSeq, startModelQtt] = Initial_MTS_Seed(FactData);
    globalAnswer.Sequence(1,:) = startModelSeq;
    globalAnswer.SeedQtty(1,:) = startModelQtt;
    %Dbg> disp(globalAnswer.Sequence(1,:));
    %Dbg> disp(globalAnswer.SeedQtty(1,:));

     %% ---- ready start -----
     pctRefD = 0;
     globalAnswer.SeqMoveStep    = 1;   % Move Sqeuence 1 --> -1,0,+1
     globalAnswer.PctQttStepMove = 20;  % Percent Move Qtt 0-100% of EOQ

     %% ---- strat progress bar ----
     H_WaitBar = waitbar(0,'Initializing waitbar...');
     oldBestAnswer = 9999999;
     nRepeatCounter = 0;
     for iLoopRun = 1:StopCondition.LoopTestRun
         perc = 100*iLoopRun/StopCondition.LoopTestRun;
         if( perc >= pctRefD )
             waitbar(perc/100,H_WaitBar,sprintf('%3.1f%% along...',perc))
             pctRefD = pctRefD + 2.33;
         end

         %-----------------------------------------------
         if(mod(iLoopRun,100) == 0)
             %------------------
             xError = globalAnswer.Answer(1) - StopCondition.LoopTarget;
             xError = xError * 100 / StopCondition.LoopTarget;
             %------------------
             if(oldBestAnswer==LocalAnswer.BestAnswer)
                 nRepeatCounter = nRepeatCounter + 1;
             else
                 oldBestAnswer =  LocalAnswer.BestAnswer;
                 nRepeatCounter = 0;
             end
             %------------------
             fprintf("  ->> %6d(%6d), %3d(%3d), %8.4f(%8.4f)\n"...
                ,iLoopRun,StopCondition.LoopTestRun...
                ,nRepeatCounter,StopCondition.nRepeatAnswer...
                ,xError,StopCondition.LoopPctError);
             %-----------------------------------------------
             if(nRepeatCounter >= StopCondition.nRepeatAnswer)
                 fprintf("Break Repeat Condition = %3d(%3d)\n"...
                     ,nRepeatCounter,StopCondition.nRepeatAnswer);
                 break; % for iLoopRun
             end
             %-----------------------------------------------
             if (xError <= StopCondition.LoopPctError)
                 fprintf("Break Diff Error = %8.4f(%8.4f)\n"...
                     ,xError,StopCondition.LoopPctError);
                 break; % for iLoopRun
             end
         end

         testData.bestSequence  = globalAnswer.Sequence(1,:);
         testData.bestSeedQtty  = globalAnswer.SeedQtty(1,:);
```

```
        testData.SeqMoveStep   = globalAnswer.SeqMoveStep;
        testData.QttMoveStep   =
round(globalAnswer.PctQttStepMove/100*mean(testData.bestSeedQtty));
        testData.maxSeqRunCase = size(testAllSeq,2);      % Maximun of sequence run
case
        testData.maxQttRunCase = sum(FactData.ArrayD,2)'; % Maximun of Qtty run case
is All D > Transpose
        % disp(testData.bestSeedQtty)

        testCase = GenTestCase(testData);
        % disp("testCase.testAnsw = "), disp(testCase.testAnsw);

        testCase = CalcObjectiveValue(testCase);
        % disp("testCase.testAnsw = "), disp(testCase.testAnsw);

        ObjValueMin = min(min(testCase.testAnsw));
        % disp("ObjValueMin = "), disp(ObjValueMin);

        [MinRow,MinCol] = find(testCase.testAnsw == ObjValueMin);
        % disp("position-A = "), disp(MinRow), disp(MinCol);

        MinRow = MinRow(1);
        MinCol = MinCol(1);
        % disp("position-1 = "), disp(MinRow), disp(MinCol);

        LocalAnswer.BestSequence = testCase.testCase(MinRow,:);
        LocalAnswer.BestSeedQtty = testCase.testQtt(MinCol,:);
        LocalAnswer.BestAnswer   = testCase.testAnsw(MinRow,MinCol);
        % disp("LocalAnswer.BestSequence = "), disp(LocalAnswer.BestSequence);
        % disp("LocalAnswer.BestSeedQtty = "), disp(LocalAnswer.BestSeedQtty);
        % disp("LocalAnswer.BestAnswer = "),   disp(LocalAnswer.BestAnswer);

        globalAnswer = ReplaceLocal2AnwsArray(globalAnswer, LocalAnswer);
    end
    close(H_WaitBar)
%=========================================================================
%=========================================================================

 %% ====== Force Start ==========================================
function [startModelSeq, startModelQtt] = Initial_MTS_Seed(FactData)
    meanArryD =  sum(FactData.ArrayD,2) ./ nPeriod ;  % row summ
    meanArryO =  sum(FactData.ArrayO,2) ./ nPeriod ;  % row summ
    meanArryS =  sum(FactData.ArrayS,1)' ./ nPeriod ; % col summ and transpose
    % Model_EOQ = ceil(sqrt(meanArryS .* meanArryD .* 2 ./ meanArryO));
    Model_EOQ = ceil(meanArryD).*2;
    Model_EOQ(isnan(Model_EOQ)) = 0;
    %Dbg> Model_EOQ              % j-Row,single-Col
    %Dbg> FactData.ArrayD


    startModelQtt = Model_EOQ';     % Transpose
    startModelSeq = randi(nRunCase, nMachine*nPeriod,1);
    startModelSeq = startModelSeq';
    % disp(startModelSeq);
    % disp(startModelQtt);

    % save ('XX.mat', 'startModelQtt', 'startModelSeq');
    % load ('XX.mat', 'startModelQtt', 'startModelSeq');
    % disp(startModelSeq);
    % disp(startModelQtt);

end

 %% ====== Replace new answer and sort result ====================
    function globalAnswer = ReplaceLocal2AnwsArray(globalAnswer, LocalAnswer)
        l_Sequence = LocalAnswer.BestSequence;
        l_SeedQtty = LocalAnswer.BestSeedQtty;
        l_Answer   = LocalAnswer.BestAnswer;
        if(globalAnswer.Answer(1,1) == l_Answer)
            globalAnswer.QttyMoveStep = globalAnswer.QttyMoveStep + 5;
            if(globalAnswer.QttyMoveStep > 100)
```

```
        testData.SeqMoveStep   = globalAnswer.SeqMoveStep;

                globalAnswer.SeqMoveStep  = globalAnswer.SeqMoveStep + 2;
                globalAnswer.QttyMoveStep = 7;
            end
        else
            globalAnswer.QttyMoveStep = 7;
            globalAnswer.SeqMoveStep = 10;
            for iSort=1:size(globalAnswer.Sequence,1)
                g_Answer = globalAnswer.Answer(iSort,1);
                if(l_Answer < g_Answer)
                    tmp_Sequence = globalAnswer.Sequence(iSort,:);
                    tmp_SeedQtty = globalAnswer.SeedQtty(iSort,:);
                    tmp_Answer   = globalAnswer.Answer(iSort,1);
                    globalAnswer.Sequence(iSort,:) = l_Sequence;
                    globalAnswer.SeedQtty(iSort,:) = l_SeedQtty;
                    globalAnswer.Answer(iSort,1)   = l_Answer;
                    l_Sequence = tmp_Sequence;
                    l_SeedQtty = tmp_SeedQtty;
                    l_Answer   = tmp_Answer;
                end
            end
        end
    end

 %% ====== Calculate Objective Value from Data ====================
    function testData = CalcObjectiveValue(testData)
        for iRowSeq = 1:  size(testData.testAnsw,1)      % row is test case sequence
            for jColQty = 1:  size(testData.testAnsw,2)  % col is test case qtty
                tmpRunIndex = testData.testCase(iRowSeq,:);
                tmpRunQSeed = testData.testQtt(jColQty,:);
                tmpRunData = horzcat(tmpRunQSeed, tmpRunIndex);  % Tabu list Check
                [isInTabuList, runTabuList] = UpdateTabu(runTabuList, tmpRunData);
                if isInTabuList
                    testData.testAnsw(iRowSeq,jColQty) = 9999999;
                elseif(sum(tmpRunQSeed) == 0)
                    testData.testAnsw(iRowSeq,jColQty) = 9999999;
                else
                    [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
                        testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
                    [ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt,
FactData);
                    ObjectiveFnc = ObjX + ObjY + ObjZ;
                    testData.testAnsw(iRowSeq,jColQty) = ObjectiveFnc;
                    if ((ObjX <= 0)||(ObjY <= 0)||(ObjZ <= 0)||(ObjectiveFnc <= 0))
                        testData.testAnsw(iRowSeq,jColQty) = 9999999;
                    end
                end
            end
        end
    end

%% ====== Generate test Case from seed and step move =============
    function rtnData = GenTestCase(testData)
        %-----------------------------------------------
        tmpTestCase = testData.bestSequence;
        maxSeqData  = testData.maxSeqRunCase;
        moveStepSeq = testData.SeqMoveStep;
        for ix = 1: size(testData.bestSequence,2)
            testCaseInc = testData.bestSequence; % load all frame
            testCaseDec = testData.bestSequence; % load all frame
            testCaseInc(ix) =
StepWithRange(1,maxSeqData,testCaseInc(ix),moveStepSeq);    % change 1 position
            testCaseDec(ix) =
StepWithRange(1,maxSeqData,testCaseDec(ix),moveStepSeq*-1); % change 1 position
            tmpTestCase = vertcat(tmpTestCase, testCaseInc, testCaseDec);
        end
        rtnData.testCase = tmpTestCase;
        % disp(rtnData.testCase)       % test case on parent loop
        %-----------------------------------------------
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

```
        frameTest = [-7,-5,-3,-1, 0, 1, 3, 5, 7];
        ansRow = size(tmpTestCase,1);
        ansCol = size(frameTest,2);
        rtnData.testAnsw = zeros(ansRow,ansCol);
        % disp(rtnData.testAnsw)       % test amswer on parent loop
        %-------------------------------------------
        ansRow = size(frameTest,2);
        ansCol = size(testData.bestSeedQtty,2);
        tempTestQtty = zeros(ansRow,ansCol);

        RefTestQtty  = testData.bestSeedQtty;
        maxQttyData  = testData.maxQttRunCase;
        moveStepQtty = testData.QttMoveStep;

        % one for random
        tmpQtty = RefTestQtty;
        for xi=2:ansCol, tmpQtty(xi) = randi(maxQttyData(xi)); end
        tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
        tempTestQtty(1,:) = tmpQtty;
        % two for random
        tmpQtty = RefTestQtty;
        for xi=2:ansCol, tmpQtty(xi) = randi(maxQttyData(xi)); end
        tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
        tempTestQtty(2,:) = tmpQtty;
        % etc for gen nornal
        for iRow = 3:ansRow
            tmpQtty = RefTestQtty + frameTest(iRow)*moveStepQtty;
            tmpQtty = RangeCutQtty(maxQttyData.*0,maxQttyData,tmpQtty);
            tempTestQtty(iRow,:) = tmpQtty;
        end
        rtnData.testQtt = tempTestQtty;
        %disp(rtnData.testQtt);
        %pause

        %-------------------------------------------
    end
%% ====== Update in Range [Min-Max] Ring Counter ================
    function result = StepWithRange(Min,Max,From,Step)
        dTemp = From - Min ;
        mTemp = Max - Min + 1 ;
        dTemp = dTemp + Step ;
        dTemp = mod(dTemp, mTemp);
        result = dTemp + Min ;
    end
    function iData = RangeCutQtty(Min,Max,iData)
        for i=1:size(iData,2)
            dataTemp = iData(i);
            dataTemp(dataTemp > Max(i)) = Max(i);
            dataTemp(dataTemp < Min(i)) = Min(i);
            iData(i) = dataTemp;
        end
    end

%% ===== Update Tabu List ========================================
   function [isInTabuList, runTabuList] = UpdateTabu(runTabuList, testData)
       isInTabuList = 0; % defualt = no
       nArrayTabu = size(runTabuList,1);
       for nCheck = 1:nArrayTabu
           if isequal(runTabuList(nCheck,:),testData)
               isInTabuList = 1;
           end
       end
       if isInTabuList == 0
           for nCheck = 1:nArrayTabu-1
               runTabuList(nCheck,:) = runTabuList(nCheck+1,:);
           end
           runTabuList(nArrayTabu,:) = testData;
       end
   end

end
```

ภาคผนวก ข.6 MATLAB-BiObj-Pseudo

1/2-ไฟล์ MATLAB – “A20y_ANS_PureRun2Pseudo.m”

```
%%
%--------------------------------------------------------------------------
tic, clc, clear all, close all
% testCaseName = "t01R_A06J02M03P";
% testCaseName = "t02R_A06J02M06P";
testCaseName = "t03S_A05J02M06P";
% testCaseName = "t04S_B05J02M06P";
% testCaseName = "t05S_C05J02M06P";
% testCaseName = "t06S_D05J02M06P";
% testCaseName = "t07S_A06J02M09P";
% testCaseName = "t08S_B06J02M09P";
% testCaseName = "t09S_C06J02M09P";
% testCaseName = "t10S_D06J02M09P";
% testCaseName = "t11S_A07J03M12P"; % Plot Graph-1/2
% testCaseName = "t12S_A08J03M09P"; % Plot Graph-2/2

%-------------------------------------------
fileName = sprintf('%s.mat',testCaseName);
RefFName = sprintf('%s.lgr',testCaseName);
logFileName = sprintf('zANS%s_%s.log', testCaseName,datestr(now,'yyyymmdd_HHMM'));
datFileName = replace(logFileName,".log",".mat");
% diary(logFileName);

%%
%--------------------------------------------------------------------------
fprintf("\n####  Multiple Objective - 1/3.Psudo #####\n");
fprintf("\n######  Step1/6: prepair data ############\n");
% FactData = FncPrepare31_ReadDataLG4(fileName); % t01, t02
FactData = FncPrepare30_ReadDataMATLAB(fileName);
FactData.nSaveAns = 10;
FactData.WorkHour = 8*6*60; % 8Hour 6Day 60Min > Min

testAllSeqName = sprintf('DataRef%02dJ_testAllSeq.mat',FactData.nJobs);
fprintf(">> Test Case = "), disp(testCaseName);
fprintf(">> File Data = "), disp(testAllSeqName);
fprintf(">> log File  = "), disp(logFileName);
fprintf(">> data File = "), disp(datFileName);
fprintf(">> Ref File Name = "), disp(RefFName);
fprintf(">> Run File Name = "), disp(fileName);
if ~(isfile(testAllSeqName))
    testAllSeq = FncPrepare10_GenerateSEED(FactData.nJobs);
    save(testAllSeqName, 'testAllSeq');
end
load(testAllSeqName);

%%
%--------------------------------------------------------------------------
fprintf("\n######  Step2/6: run model ################\n");
StopCondition.LoopTestRun   = 1000;
StopCondition.nRepeatAnswer = 25;
StopCondition.LoopPctError  = 1.0; % 12.0;
ObjMX  = 50; % 275865;
ObjMY  = 25; % 7928;
ObjMZ  = 25; % 168;
StopCondition.LoopTarget = ObjMX + ObjMY + ObjMZ;

fprintf(">> max.LoopRun  = %d\n",StopCondition.LoopTestRun);
fprintf(">> max.nRepeat  = %d\n",StopCondition.nRepeatAnswer);
fprintf(">> min.Target   = %d\n",StopCondition.LopTarget);
fprintf(">> %%.TargetDiff = %f\n",StopCondition.LoopPctError);

% ResultData = A21a_ANS_Subr(testAllSeq, FactData, StopCondition);
% save ('z_abcdef.mat');
load ('z_abcdef.mat');
disp('>> Result >> '), disp(ResultData.Answer);
```

```
%%
fprintf("\n######  Step3/6: display resul ##########\n");
tmpBestObjt = ResultData.Answer(1,:);
tmpRunQSeed = ResultData.SeedQtty(1,:);
tmpRunIndex = ResultData.Sequence(1,:);
[testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
    testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
%Dbg> testSeq
%Dbg> testQtt

[ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData);
ObjectiveFnc = ObjX + ObjY + ObjZ;
fprintf(">> MATLAB >> Calc = %d",tmpBestObjt);
fprintf("\n  >>> best Seed Qtty = "), fprintf("%d,", tmpRunQSeed);
fprintf("\n  >>> best Run Index = "), fprintf("%d,", tmpRunIndex);
fprintf("\n  >>> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n");

ObjMAll = ObjMX + ObjMY + ObjMZ;
fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
fprintf("\n");


%%
fprintf("\n######  Step4/6: compare Math Model #####\t");
if isfile(RefFName)
    FactDataLG = FncPrepare20_ExtractResultLGR(RefFName);
    ObjMX = sum(sum(FactDataLG.ArrayG.*FactDataLG.ArrayC));
    ObjMY = sum(sum(FactDataLG.ArrayO.*FactDataLG.ArrayH));
    ObjMZ = sum(sum(sum(sum(FactDataLG.ArrayS.*FactDataLG.ArrayX))));
    ObjMAll = ObjMX + ObjMY + ObjMZ;
    fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
    fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
    fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
    fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
    fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
    fprintf("\n");
else
    fprintf("\n  ->> Warning: file does not exist");
    fprintf("\n  ->>          ""%s""\n", RefFName);
end

%% 319653 321751
fprintf("\n######  Step5/6: run time ###############\n");
diary off
save(datFileName)
fprintf(">> "),toc

%% 319653 321751
fprintf("\n######  Step6/6: plot Data ################\n");
FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData)
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

2/2-ไฟล์ MATLAB – “FncRun90_PlotMutiObj.m”

```
function PlotResult = FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData)
    XX = zeros(1,FactData.nSaveAns);    XX(XX == 0) = NaN;
    YY = zeros(1,FactData.nSaveAns);    YY(YY == 0) = NaN;
    for jj=1:FactData.nSaveAns
        if (ResultData.Answer(jj) ~= 9999999)
            tmpRunIndex = ResultData.Sequence(jj,:);
            tmpRunQSeed = ResultData.SeedQtty(jj,:);
            [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
                testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
            % [ObjX, ObjY, ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt,
FactData);
            [~, xxUtilty] = Calc_ArrayC(testSeq, testQtt, FactData);
            XX(jj) = ResultData.Answer(jj);
            YY(jj) = xxUtilty;
        end
    end
    figure(1);
    hold off;
    Graph_HH = plot(XX,YY,'p');
    hold on
    grid on;
    title('Total Course vs Machine Utility');
    xlabel('Total Course (THB)');
    ylabel('Machine Utility (%)');

     %%
    %--- Create ArrayZZ, ArrayUU
    ArrayZZ = XX(1);
    ArrayUU = YY(1);
    for iloop=2 : FactData.nSaveAns
        if ~isnan(XX(iloop))
            ArrayZZ = vertcat(ArrayZZ,XX(iloop));
            ArrayUU = vertcat(ArrayUU,YY(iloop));
        end
    end
    %%
    %--- Create ArrayZZ, ArrayUU
    [ zUBest, zZBest ] = Calc_ArrayZZ_Min(ArrayZZ, ArrayUU);
    [ uUBest, uZBest ] = Calc_ArrayUU_Max(ArrayZZ, ArrayUU);
    fprintf('best_ZZ(min)     >> (%d,%f)\n',zZBest, zUBest);
    fprintf('best_UU(max)     >> (%d,%f)\n',uZBest, uUBest);
    plot([zZBest,uZBest],[zUBest,uUBest],'ro');

    [ UBest, ZBest ] = Calc_ArrayUZ_ZScore(ArrayZZ, ArrayUU);
    fprintf('best_UZ(Z-Score) >> (%d,%f)\n',ZBest, UBest);
    plot(ZBest,UBest,'ko');

    plot([zZBest,uZBest],[zUBest,uUBest],'r*');
    plot(ZBest,UBest,'k*');

    legend('Z-List','best-min.Z & best-max.U','best-ZScore','Location','northwest');
    set(Graph_HH, 'ButtonDownFcn', @getPoint);

    %%
    %##########################################################################
    function [UBest, ZBest] = Calc_ArrayUU_Max(Array_ZZ, Array_UU)
        postArray = find(Array_UU == max(Array_UU));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
    end    %%

    %%
    function [UBest, ZBest] = Calc_ArrayZZ_Min(Array_ZZ, Array_UU)
        postArray = find(Array_ZZ == min(Array_ZZ));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
    end

%%
```

```
    function getPoint(varargin)
              currentPoint = get(gca, 'CurrentPoint');
              fprintf('Hit Point! Coordinates: %f, %f \n', ...
                       currentPoint(1), currentPoint(3));
    end

    %%
    function [tmpArryC, mUtility] = Calc_ArrayC(testSeq, testQtt, FactData)
        fullTime = FactData.WorkHour;
        nJobs = FactData.nJobs;
        nMachine = FactData.nMachines;
        nPeriod  = FactData.nPeriods;
        kReadIndex = 0;
        % ---- Calc ArryC
        tmpArryC = zeros(nJobs,nMachine,nPeriod);
        for kPrd=1:nPeriod
            for jMach=1:nMachine
                xFrom = 1;
                CTotal = 0;
                kReadIndex = kReadIndex + 1;
                for iJobs=2:nJobs
                    xNext = testSeq(iJobs,kReadIndex);
                    if ~((xNext == 0)||(xNext == 1))
                        CRelease = max(CTotal, FactData.ArrayR(xNext));
                        CSetup   = FactData.ArrayS(xFrom,xNext);
                        CRunTime = testQtt(xNext,kReadIndex)*FactData.ArrayP(xNext);
                        CTotal = CRelease + CSetup + CRunTime;
                        tmpArryC(xNext,jMach,kPrd) = CTotal;
                        xFrom = xNext;
                    end
                end
            end
        end

        % ---- Calc U
        ZZ = [];
        for kPrd=1:nPeriod
           ZZ = horzcat(ZZ, max(tmpArryC(:,:,kPrd)));
        end
        ZZ = ZZ *100 / fullTime;
        mUtility = sum(ZZ)/size(ZZ,2);
    end

end
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

ภาคผนวก ข.7 MATLAB-BiObj-Parallel

1/3-ไฟล์ MATLAB – “A20y_ANS_Parallel.m”

```
%%
%-------------------------------------------------------------------------
tic, clc, clear all, close all
% testCaseName = "t01R_A06J02M03P";
% testCaseName = "t02R_A06J02M06P";
% testCaseName = "t03S_A05J02M06P";
% testCaseName = "t04S_B05J02M06P";
% testCaseName = "t05S_C05J02M06P";
% testCaseName = "t06S_D05J02M06P";
% testCaseName = "t07S_A06J02M09P";
% testCaseName = "t08S_B06J02M09P";
% testCaseName = "t09S_C06J02M09P";
% testCaseName = "t10S_D06J02M09P";
% testCaseName = "t11S_A07J03M12P"; % Plot Graph-1/2
testCaseName = "t12S_A08J03M09P"; % Plot Graph-2/2

%-------------------------------------------
fileName = sprintf('%s.mat',testCaseName);
RefFName = sprintf('%s.lgr',testCaseName);
logFileName = sprintf('zANS%s_%s.log', testCaseName,datestr(now,'yyyymmdd_HHMM'));
datFileName = replace(logFileName,".log",".mat");
% diary(logFileName);

%%
%-------------------------------------------------------------------------
fprintf("\n####  Multiple Objective - 1/3.Psudo #####\n");
fprintf("\n######  Step1/6: prepair data ############\n");
% FactData = FncPrepare31_ReadDataLG4(fileName); % t01, t02
FactData = FncPrepare30_ReadDataMATLAB(fileName);
FactData.nSaveAns = 10;
FactData.WorkHour = 6*8*60; % 6Day 8Hour 60Min -> Min

testAllSeqName = sprintf('DataRef%02dJ_testAllSeq.mat',FactData.nJobs);
fprintf(">> Test Case = "), disp(testCaseName);
fprintf(">> File Data = "), disp(testAllSeqName);
fprintf(">> log File  = "), disp(logFileName);
fprintf(">> data File = "), disp(datFileName);
fprintf(">> Ref File Name = "), disp(RefFName);
fprintf(">> Run File Name = "), disp(fileName);
if ~(isfile(testAllSeqName))
    testAllSeq = FncPrepare10_GenerateSEED(FactData.nJobs);
    save(testAllSeqName, 'testAllSeq');
end
load(testAllSeqName);

%%
%-------------------------------------------------------------------------
fprintf("\n######  Step2/6: run model ################\n");
StopCondition.LoopTestRun   = 100;
StopCondition.nRepeatAnswer = 25;
StopCondition.LoopPctError  = 1.0; % 12.0;
ObjMX  = 50; % 275865;
ObjMY  = 25; % 7928;
ObjMZ  = 25; % 168;
StopCondition.LoopTarget = ObjMX + ObjMY + ObjMZ;

fprintf(">> max.LoopRun  = %d\n",StopCondition.LoopTestRun);
fprintf(">> max.nRepeat  = %d\n",StopCondition.nRepeatAnswer);
fprintf(">> min.Target   = %d\n",StopCondition.LoopTarget);
fprintf(">> %%.TargetDiff = %f\n",StopCondition.LoopPctError);

ResultData = A21a_ParANS_Subr(testAllSeq, FactData, StopCondition);
save ('z_abcdef.mat');
load ('z_abcdef.mat');
disp('>> Result Major >> '), disp(ResultData.OneAnswer);
```

```
disp('>> Result Minor U >> '), disp(ResultData.TwoAnswerU);
disp('>> Result Minor Z >> '), disp(ResultData.TwoAnswerZ);

%%
fprintf("\n######  Step3/6: display resul ##########\n");
tmpBestObjt = ResultData.OneAnswer(1,:);
tmpRunQSeed = ResultData.SeedQtty(1,:);
tmpRunIndex = ResultData.Sequence(1,:);
[testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
    testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
%Dbg> testSeq
%Dbg> testQtt

[ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData);
ObjectiveFnc = ObjX + ObjY + ObjZ;
fprintf(">> MATLAB >> Calc = %d",tmpBestObjt);
fprintf("\n  >>> best Seed Qtty = "), fprintf("%d,", tmpRunQSeed);
fprintf("\n  >>> best Run Index = "), fprintf("%d,", tmpRunIndex);
fprintf("\n  >>> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n");

ObjMAll = ObjMX + ObjMY + ObjMZ;
fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
fprintf("\n");


%%
fprintf("\n######  Step4/6: compare Math Model #####\t");
if isfile(RefFName)
    FactDataLG = FncPrepare20_ExtractResultLGR(RefFName);
    ObjMX = sum(sum(FactDataLG.ArrayG.*FactDataLG.ArrayC));
    ObjMY = sum(sum(FactDataLG.ArrayO.*FactDataLG.ArrayH));
    ObjMZ = sum(sum(sum(sum(FactDataLG.ArrayS.*FactDataLG.ArrayX))));
    ObjMAll = ObjMX + ObjMY + ObjMZ;
    fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
    fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
    fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
    fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
    fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
    fprintf("\n");
else
    fprintf("\n  ->> Warning: file does not exist");
    fprintf("\n  ->>          ""%s""\n", RefFName);
end

%% 319653 321751
fprintf("\n######  Step5/6: run time ###############\n");
fprintf(">> "),toc

%% 319653 321751
fprintf("\n######  Step6/6: plot Data ###############\n");
FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData);
diary off
% save(datFileName)
```

2/3-ไฟล์ MATLAB – “FncRun90_PlotMutiObj.m”

```
function FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData)
    XX = zeros(1,FactData.nSaveAns);    XX(XX == 0) = NaN;
    YY = zeros(1,FactData.nSaveAns);    YY(YY == 0) = NaN;
    HH = zeros(1,FactData.nSaveAns);    HH(HH == 0) = NaN;
    KK = zeros(1,FactData.nSaveAns);    KK(KK == 0) = NaN;
    for jj=1:FactData.nSaveAns
        if (ResultData.OneAnswer(jj) ~= 9999999)
            tmpRunIndex = ResultData.Sequence(jj,:);
            tmpRunQSeed = ResultData.SeedQtty(jj,:);
            [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
                testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
            % [ObjX, ObjY, ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt,
FactData);
            [~, xxUtilty] = FncRun91_Calculate_ArrayC(testSeq, testQtt, FactData);
            XX(jj) = ResultData.OneAnswer(jj);
            YY(jj) = xxUtilty;
            HH(jj) = ResultData.TwoAnswerZ(jj);
            KK(jj) = ResultData.TwoAnswerU(jj);
        end
    end
    figure(1);
    hold off;
    Graph_HH = plot(XX,YY,'mp',HH,KK,'bh');
    hold on;
    grid on;
    title('Total Course vs Machine Utility');
    xlabel('Total Course');
    ylabel('Machine Utility(%)');

    %%
    %--- Create ArrayZZ, ArrayUU
    ArrayZZ = XX(1);
    ArrayUU = YY(1);
    for iloop=2 : FactData.nSaveAns
        if ~isnan(XX(iloop))
            ArrayZZ = vertcat(ArrayZZ,XX(iloop));
            ArrayUU = vertcat(ArrayUU,YY(iloop));
        end
        if ~isnan(HH(iloop))
            ArrayZZ = vertcat(ArrayZZ,HH(iloop));
            ArrayUU = vertcat(ArrayUU,KK(iloop));
        end
    end

    %%
    %--- Create ArrayZZ, ArrayUU
    [ zUBest, zZBest ] = Calc_ArrayZZ_Min(ArrayZZ, ArrayUU);
    [ uUBest, uZBest ] = Calc_ArrayUU_Max(ArrayZZ, ArrayUU);
    fprintf('best_ZZ(min)     >> (%d,%f)\n',zZBest, zUBest);
    fprintf('best_UU(max)     >> (%d,%f)\n',uZBest, uUBest);
    plot([zZBest,uZBest],[zUBest,uUBest],'ro');

    [ UBest, ZBest ] = Calc_ArrayUZ_ZScore(ArrayZZ, ArrayUU);
    fprintf('best_UZ(Z-Score) >> (%d,%f)\n',ZBest, UBest);
    plot(ZBest,UBest,'ko');

    plot([zZBest,uZBest],[zUBest,uUBest],'r*');
    plot(ZBest,UBest,'k*');

    legend({'Z-List','U-List','best-min.Z & best-max.U','best-ZScore'},...
        'Location','northwest');
    set(Graph_HH, 'ButtonDownFcn', @getPoint);

    %%
    %#########################################################################
    function [UBest, ZBest] = Calc_ArrayUU_Max(Array_ZZ, Array_UU)
        postArray = find(Array_UU == max(Array_UU));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
```

```
    end

%%
    function [UBest, ZBest] = Calc_ArrayUZ_ZScore(Array_ZZ, Array_UU)
        UZ_ZScore = abs(zscore(Array_ZZ))*(0.5) + abs(zscore(Array_UU))*(0.5);
        postArray = find(UZ_ZScore == min(UZ_ZScore));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
    end
%%
    function getPoint(varargin)
              currentPoint = get(gca, 'CurrentPoint');
              fprintf('Hit Point! Coordinates: %f, %f \n', ...
                       currentPoint(1), currentPoint(3));
    end

end
```

3/3-ไฟล์ MATLAB – “FncRun91_Calculate_ArrayC.m”

```
function [tmpArryC, mUtility] = FncRun91_Calculate_ArrayC(testSeq, testQtt, FactData)
    fullTime = FactData.WorkHour;
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    kReadIndex = 0;
    % ---- Calc ArryC
    tmpArryC = zeros(nJobs,nMachine,nPeriod);
    for kPrd=1:nPeriod
        for jMach=1:nMachine
            xFrom = 1;
            CTotal = 0;
            kReadIndex = kReadIndex + 1;
            for iJobs=2:nJobs
                xNext = testSeq(iJobs,kReadIndex);
                if ~((xNext == 0)||(xNext == 1))
                    CRelease = max(CTotal, FactData.ArrayR(xNext));
                    CSetup   = FactData.ArrayS(xFrom,xNext);
                    CRunTime = testQtt(xNext,kReadIndex)*FactData.ArrayP(xNext);
                    CTotal = CRelease + CSetup + CRunTime;
                    tmpArryC(xNext,jMach,kPrd) = CTotal;
                    xFrom = xNext;
                end
            end
        end
    end

    % ---- Calc U
    ZZ = [];
    for kPrd=1:nPeriod
        ZZ = horzcat(ZZ, max(tmpArryC(:,:,kPrd)));
    end
    ZZ = ZZ *100 / fullTime;
    mUtility = sum(ZZ)/size(ZZ,2);
end
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

ภาคผนวก ข.8 MATLAB-BiObj-Serial

1/3-ไฟล์ MATLAB – “A30z_ANS_Serial.m”

```
%%
%-------------------------------------------------------------------------
tic, clc, clear all, close all
% testCaseName = "t01R_A06J02M03P";
% testCaseName = "t02R_A06J02M06P";
testCaseName = "t03S_A05J02M06P";
% testCaseName = "t04S_B05J02M06P";
% testCaseName = "t05S_C05J02M06P";
% testCaseName = "t06S_D05J02M06P";
% testCaseName = "t07S_A06J02M09P";
% testCaseName = "t08S_B06J02M09P";
% testCaseName = "t09S_C06J02M09P";
% testCaseName = "t10S_D06J02M09P";
% testCaseName = "t11S_A07J03M12P"; % Plot Graph-1/2
% testCaseName = "t12S_A08J03M09P"; % Plot Graph-2/2

%-----------------------------------------
fileName = sprintf('%s.mat',testCaseName);
RefFName = sprintf('%s.lgr',testCaseName);
logFileName = sprintf('zANS%s_%s.log', testCaseName,datestr(now,'yyyymmdd_HHMM'));
datFileName = replace(logFileName,".log",".mat");
% diary(logFileName);

%%
%-------------------------------------------------------------------------
fprintf("\n####  Multiple Objective - 1/3.Psudo #####\n");
fprintf("\n######  Step1/6: prepair data ############\n");
% FactData = FncPrepare31_ReadDataLG4(fileName); % t01, t02
FactData = FncPrepare30_ReadDataMATLAB(fileName);
FactData.lowerU = 25;
FactData.nSaveAns = 10;
FactData.WorkHour = 8*6*60; % 8Hour 6Day 60Min > Min

testAllSeqName = sprintf('DataRef%02dJ_testAllSeq.mat',FactData.nJobs);
fprintf(">> Test Case = "), disp(testCaseName);
fprintf(">> File Data = "), disp(testAllSeqName);
fprintf(">> log File  = "), disp(logFileName);
fprintf(">> data File = "), disp(datFileName);
fprintf(">> Ref File Name = "), disp(RefFName);
fprintf(">> Run File Name = "), disp(fileName);
if ~(isfile(testAllSeqName))
    testAllSeq = FncPrepare10_GenerateSEED(FactData.nJobs);
    save(testAllSeqName, 'testAllSeq');
end
load(testAllSeqName);

%%
%-------------------------------------------------------------------------
fprintf("\n######  Step2/6: run model ################\n");
StopCondition.LoopTestRun   = 100;
StopCondition.nRepeatAnswer = 25;
StopCondition.LoopPctError  = 1.0; % 12.0;
ObjMX  = 50; % 275865;
ObjMY  = 25; % 7928;
ObjMZ  = 25; % 168;
StopCondition.LoopTarget = ObjMX + ObjMY + ObjMZ;

fprintf(">> max.LoopRun  = %d\n",StopCondition.LoopTestRun);
fprintf(">> max.nRepeat  = %d\n",StopCondition.nRepeatAnswer);
fprintf(">> min.Target   = %d\n",StopCondition.LoopTarget);
fprintf(">> %%.TargetDiff = %f\n",StopCondition.LoopPctError);

ResultData = A31a_SrlANS_Subr(testAllSeq, FactData, StopCondition);
save ('z_abcdef.mat');
load ('z_abcdef.mat');
```

```
disp('>> Result >> '), disp(ResultData.Answer);

%%
fprintf("\n######  Step3/6: display resul ##########\n");
tmpBestObjt = ResultData.Answer(1,:);
tmpRunQSeed = ResultData.SeedQtty(1,:);
tmpRunIndex = ResultData.Sequence(1,:);
[testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
    testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
%Dbg> testSeq
%Dbg> testQtt

[ObjX,ObjY,ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt, FactData);
ObjectiveFnc = ObjX + ObjY + ObjZ;
fprintf(">> MATLAB >> Calc = %d",tmpBestObjt);
fprintf("\n  >>> best Seed Qtty = "), fprintf("%d,", tmpRunQSeed);
fprintf("\n  >>> best Run Index = "), fprintf("%d,", tmpRunIndex);
fprintf("\n  >>> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n");

ObjMAll = ObjMX + ObjMY + ObjMZ;
fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
fprintf("\n");

%%
fprintf("\n######  Step4/6: compare Math Model #####\t");
if isfile(RefFName)
    FactDataLG = FncPrepare20_ExtractResultLGR(RefFName);
    ObjMX = sum(sum(FactDataLG.ArrayG.*FactDataLG.ArrayC));
    ObjMY = sum(sum(FactDataLG.ArrayO.*FactDataLG.ArrayH));
    ObjMZ = sum(sum(sum(sum(FactDataLG.ArrayS.*FactDataLG.ArrayX))));
    ObjMAll = ObjMX + ObjMY + ObjMZ;
    fprintf("\n  ->> ObjMath > %d + %d + %d = %d",ObjMX, ObjMY, ObjMZ, ObjMAll);
    fprintf("\n  ->> ObjHeur > %d + %d + %d = %d",ObjX, ObjY, ObjZ, ObjectiveFnc);
    fprintf("\n  ->>    Diff > %d + %d + %d",ObjMX-ObjX, ObjMY-ObjY, ObjMZ-ObjZ);
    fprintf("\n  ->>    Diff > = %d",(ObjMAll - ObjectiveFnc));
    fprintf("\n  ->>    Diff > = %8.4f%%",(ObjMAll - ObjectiveFnc)*100/ObjMAll);
    fprintf("\n");
else
    fprintf("\n  ->> Warning: file does not exist");
    fprintf("\n  ->>          ""%s""\n", RefFName);
end

%% 319653 321751
fprintf("\n######  Step5/6: run time ###############\n");
diary off
% save(datFileName)
fprintf(">> "),toc

%% 319653 321751
fprintf("\n######  Step6/6: plot Data ###############\n");
FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData)
```

มหาวิทยาลัยเทคโนโลยีสุรนารี

2/3-ไฟล์ MATLAB – “FncRun90_PlotMutiObj.m”

```
function FncRun90_PlotMutiObj(testAllSeq,FactData,ResultData)
    XX = zeros(1,FactData.nSaveAns);    XX(XX == 0) = NaN;
    YY = zeros(1,FactData.nSaveAns);    YY(YY == 0) = NaN;
    HH = zeros(1,FactData.nSaveAns);    HH(HH == 0) = NaN;
    KK = zeros(1,FactData.nSaveAns);    KK(KK == 0) = NaN;
    for jj=1:FactData.nSaveAns
        if (ResultData.OneAnswer(jj) ~= 9999999)
            tmpRunIndex = ResultData.Sequence(jj,:);
            tmpRunQSeed = ResultData.SeedQtty(jj,:);
            [testSeq, testQtt] = FncRun00_RunSeq_SeedQtt_2_ObjectiveValue(...
                testAllSeq, FactData, tmpRunIndex, tmpRunQSeed);
            % [ObjX, ObjY, ObjZ] = FncRun10_SEED_2_ObjValue(testSeq, testQtt,
FactData);
            [~, xxUtilty] = FncRun91_Calculate_ArrayC(testSeq, testQtt, FactData);
            XX(jj) = ResultData.OneAnswer(jj);
            YY(jj) = xxUtilty;
            HH(jj) = ResultData.TwoAnswerZ(jj);
            KK(jj) = ResultData.TwoAnswerU(jj);
        end
    end
    figure(1);
    hold off;
    Graph_HH = plot(XX,YY,'mp',HH,KK,'bh');
    hold on;
    grid on;
    title('Total Course vs Machine Utility');
    xlabel('Total Course');
    ylabel('Machine Utility(%)');

    %%
    %--- Create ArrayZZ, ArrayUU
    ArrayZZ = XX(1);
    ArrayUU = YY(1);
    for iloop=2 : FactData.nSaveAns
        if ~isnan(XX(iloop))
            ArrayZZ = vertcat(ArrayZZ,XX(iloop));
            ArrayUU = vertcat(ArrayUU,YY(iloop));
        end
        if ~isnan(HH(iloop))
            ArrayZZ = vertcat(ArrayZZ,HH(iloop));
            ArrayUU = vertcat(ArrayUU,KK(iloop));
        end
    end

    %%
    %--- Create ArrayZZ, ArrayUU
    [ zUBest, zZBest ] = Calc_ArrayZZ_Min(ArrayZZ, ArrayUU);
    [ uUBest, uZBest ] = Calc_ArrayUU_Max(ArrayZZ, ArrayUU);
    fprintf('best_ZZ(min)     >> (%d,%f)\n',zZBest, zUBest);
    fprintf('best_UU(max)     >> (%d,%f)\n',uZBest, uUBest);
    plot([zZBest,uZBest],[zUBest,uUBest],'ro');

    [ UBest, ZBest ] = Calc_ArrayUZ_ZScore(ArrayZZ, ArrayUU);
    fprintf('best_UZ(Z-Score) >> (%d,%f)\n',ZBest, UBest);
    plot(ZBest,UBest,'ko');

    plot([zZBest,uZBest],[zUBest,uUBest],'r*');
    plot(ZBest,UBest,'k*');

    legend({'Z-List','U-List','best-min.Z & best-max.U','best-ZScore'},...
        'Location','northwest');
    set(Graph_HH, 'ButtonDownFcn', @getPoint);

    %%
    %##########################################################################
    function [UBest, ZBest] = Calc_ArrayUU_Max(Array_ZZ, Array_UU)
        postArray = find(Array_UU == max(Array_UU));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
```

```
    end    %%

    %%
    function [UBest, ZBest] = Calc_ArrayZZ_Min(Array_ZZ, Array_UU)
        postArray = find(Array_ZZ == min(Array_ZZ));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
    end

%%
    function [UBest, ZBest] = Calc_ArrayUZ_ZScore(Array_ZZ, Array_UU)
        UZ_ZScore = abs(zscore(Array_ZZ))*(0.5) + abs(zscore(Array_UU))*(0.5);
        postArray = find(UZ_ZScore == min(UZ_ZScore));
        UBest = Array_UU(postArray(1,1));
        ZBest = Array_ZZ(postArray(1,1));
    end
%%
    function getPoint(varargin)
              currentPoint = get(gca, 'CurrentPoint');
              fprintf('Hit Point! Coordinates: %f, %f \n', ...
                       currentPoint(1), currentPoint(3));
    end

end
```

3/3-ไฟล์ MATLAB – “FncRun91_CalcArrayC.m”

```
function [tmpArryC, mUtility] = FncRun91_Calculate_ArrayC(testSeq, testQtt, FactData)
    fullTime = FactData.WorkHour;
    nJobs = FactData.nJobs;
    nMachine = FactData.nMachines;
    nPeriod  = FactData.nPeriods;
    kReadIndex = 0;
    % ---- Calc ArryC
    tmpArryC = zeros(nJobs,nMachine,nPeriod);
    for kPrd=1:nPeriod
        for jMach=1:nMachine
            xFrom = 1;
            CTotal = 0;
            kReadIndex = kReadIndex + 1;
            for iJobs=2:nJobs
                xNext = testSeq(iJobs,kReadIndex);
                if ~((xNext == 0)||(xNext == 1))
                    CRelease = max(CTotal, FactData.ArrayR(xNext));
                    CSetup   = FactData.ArrayS(xFrom,xNext);
                    CRunTime = testQtt(xNext,kReadIndex)*FactData.ArrayP(xNext);
                    CTotal = CRelease + CSetup + CRunTime;
                    tmpArryC(xNext,jMach,kPrd) = CTotal;
                    xFrom = xNext;
                end
            end
        end
    end

    % ---- Calc U
    ZZ = [];
    for kPrd=1:nPeriod
        ZZ = horzcat(ZZ, max(tmpArryC(:,:,kPrd)));
    end
    ZZ = ZZ *100 / fullTime;
    mUtility = sum(ZZ)/size(ZZ,2);
end
```

ประวัติผู้เขียน

นายวิชัย ศรีสุรักษ์ เกิดเมื่อวันที่ 21 มีนาคม พ.ศ. 2517 ที่ตำบลรังแร้ง อำเภออุทุมพรพิสัย จังหวัดศรีสะเกษ ปัจจุบันอาศัยอยู่ที่ บ้านเลขที่ 123 หมู่ที่ 12 ตำบลธงชัยเหนือ อำเภอปักธงชัย จังหวัดนครราชสีมา สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลาย จากโรงเรียนสตรีศิริเกศ อำเภอเมือง จังหวัดศรีสะเกษ สำเร็จการศึกษาระดับอนุปริญญาวิทยาศาสตร์ (สาขาเคมีปฏิบัติ) จากมหาวิทยาลัยราชภัฏสุรินทร์ จังหวัดสุรินทร์ เมื่อ พ.ศ. 2536 สำเร็จการศึกษาระดับปริญญาตรีวิศวกรรมศาสตรบัณฑิต (วิศวกรรมโทรคมนาคม) จากมหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2541 และสำเร็จการศึกษาระดับปริญญาโท วิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมไฟฟ้า) จากมหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2550 จากนั้นศึกษาต่อ ขณะศึกษาต่อในระดับปริญญาเอก วิศวกรรมศาสตรดุษฎีบัณฑิต (วิศวกรรมการผลิต) จากมหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2557

มหาวิทยาลัยเทคโนโลยีสุรนารี