การวางแผนการผลิตและการดำเนินงานเพื่อลดต้นทุนค่าไฟฟ้า
กรณีศึกษาโรงงานผลิตผลิตภัณฑ์บำรุงผม

นางสาวสุนิสา กุลพนาภินันท์

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมอุตสาหการ ภาควิชาวิศวกรรมอุตสาหการ
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2549
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

PRODUCTION AND OPERATIONS PLANNING FOR REDUCTION OF ELECTRICITY CHARGE: A CASE STUDY OF A HAIR-NOURISHING MANUFACTURING

Ms. Sunisa Kulpanapinunt

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Engineering Program in Industrial Engineering

Department of Industrial Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2006

หัวข้อวิทยานิพนธ์	การวางแผนการผลิตและการดำเนินงานเพื่อลดต้นทุนค่าไฟฟ้า กรณีศึกษาโรงงานผลิตผลิตภัณฑ์บำรุงผม
โดย	นางสาวสุนิสา กุลพนาภินันท์
สาขาวิชา	วิศวกรรมอุตสาหการ
อาจารย์ที่ปรึกษา	อาจารย์ ดร.สีรง ปรีชานนท์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.ดิเรก ลาวัณย์ศิริ)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.เหรียญ บุญดีสกุลโชค)

..... อาจารย์ที่ปรึกษา
(อาจารย์ ดร.สีรง ปรีชานนท์)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.วิภาวี ธรรมาภรณ์พิลาศ)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ปวีณา เชาวลิตวงศ์)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สุนิสา กุลพนาภินันท์: การวางแผนการผลิตและการดำเนินงานเพื่อลดต้นทุนค่าไฟฟ้า กรณีศึกษาโรงงานผลิตผลิตภัณฑ์บำรุงผม. (PRODUCTION AND OPERATIONS PLANNING FOR REDUCTION OF ELECTRICITY CHARGE: A CASE STUDY OF A HAIR-NOURISHING MANUFACTURING) อ. ที่ปรึกษา : อ.ดร.สีรง ปรีชานนท์, 94 หน้า.

งานวิจัยนี้เป็นการศึกษาวิธีการวางแผนการผลิตในขั้นตอนการผสมของโรงงานผลิตผลิตภัณฑ์บำรุงเส้นผม โดยมีวัตถุประสงค์เพื่อลดต้นทุนค่าไฟฟ้า จึงได้พัฒนาตัวแบบเชิงคณิตศาสตร์เพื่อบรรลุวัตถุประสงค์ดังกล่าว ผู้วิจัยได้ใช้ตัวแบบเชิงคณิตศาสตร์ในรูปแบบโปรแกรมเชิงจำนวนเต็มแบบผสม (Mixed Integer Programming) เพื่อช่วยในการหาคำตอบ แต่เนื่องจากการหาคำตอบด้วยวิธีดังกล่าวจะใช้เวลาในการหาคำตอบนานมากถ้าปัญหามีขนาดใหญ่ ดังนั้นจึงมีความจำเป็นต้องพัฒนาฮิวริสติกขึ้นมาเพื่อช่วยในการหาคำตอบที่ดีภายในเวลาอันเหมาะสม

ฮิวริสติกที่ได้พัฒนาขึ้นประกอบด้วย 2 ส่วนหลัก ส่วนแรกเป็นการจัดงานให้กับเครื่องจักร โดยตัวแบบเชิงคณิตศาสตร์ในรูปแบบโปรแกรมเชิงจำนวนเต็มแบบผสม ซึ่งมีวัตถุประสงค์เพื่อทำให้ค่าพลังงานไฟฟ้ารวมมีค่าน้อยที่สุด ส่วนที่สองเป็นกำหนดเวลาการผลิตของแต่ละงานบนแต่ละเครื่องจักรเพื่อให้ค่าพลังงานไฟฟ้าสูงสุดมีค่าน้อยที่สุด ในส่วนนี้ได้กำหนดกลยุทธ์กำหนดเวลาการผลิตไว้ 3 แบบ คือ แบบที่ 1 เป็นการนำเอาเครื่องจักรที่มีกำลังไฟฟ้ามากที่สุดมาเริ่มต้นกำหนดเวลาการผลิตของงานแต่ละงานก่อนแล้วตามด้วยเครื่องจักรที่มีกำลังไฟฟ้าน้อยกว่า แบบที่ 2 เป็นการนำเอาเครื่องจักรที่มีกำลังไฟฟ้าน้อยที่สุดมาเริ่มต้นกำหนดเวลาการผลิตของงานแต่ละงานก่อนแล้วตามด้วยเครื่องจักรที่มีกำลังไฟฟ้ามากกว่า และแบบที่ 3 เป็นการนำเอาเครื่องจักรที่มีกำลังไฟฟ้าแบบสุ่มมาเริ่มต้นกำหนดเวลาการผลิตของงานแต่ละงาน

ผลการทดสอบความสามารถในการหาคำตอบของฮิวริวติกที่นำเสนอ สำหรับปัญหาทดสอบที่มีจำนวนงาน 5 งาน 10 งาน 15 งาน 20 งาน และ 30 งาน พบว่าการหาคำตอบของกลยุทธ์แบบที่ 1 ทำให้ต้นทุนค่าไฟฟ้าต่ำที่สุดในปัญหาทดสอบดังกล่าว โดยมีเปอร์เซ็นต์ความผิดพลาดเฉลี่ยไม่เกิน 3% และใช้เวลาในการหาคำตอบไม่เกิน 5 วินาที สรุปได้ว่าฮิวริสติกที่นำเสนอมีประสิทธิภาพอยู่ในเกณฑ์ดีสามารถนำไปใช้แก้ปัญหาจริงในอุตสาหกรรมได้

ภาควิชา.....วิศวกรรมอุตสาหการ.....ลายมือชื่อนิสิต.....สุนิสา กุลพนาภินันท์.....

สาขาวิชา.....วิศวกรรมอุตสาหการ.....ลายมือชื่ออาจารย์ที่ปรึกษา.....

ปีการศึกษา.....2549.....

4770510021 : MAJOR INDUSTRIAL ENGINEERING

KEY WORD: Electricity Charge / Planning

SUNISA KULPANAPINUNT : PRODUCTION AND OPERATIONS PLANNING FOR REDUCTION OF ELECTRICITY CHARGE: A CASE STUDY OF A HAIR-NOURISHING MANUFACTURING. THESIS ADVISOR : SEERONK PRICHANONT, Ph.D., 94 pp.

This research studies the production planning process at the mixing stage of a hair-nourishing manufacturing plant. The objective of the study is to reduce the plant's electricity charge. The author developed a mixed-integer program for the problem. But since the computational time for real-world problems is usually high, a heuristic to solve the problem was also developed.

The heuristic that was developed consisted of two main steps. With the first step, the job-machine assignments were determined with the objective of minimizing the total energy charge. This is done through the solving of a mixed-integer program, specifically developed for this step. The second step determines the start times of the jobs on each machine. The objective of the second step is to find the starting times of the jobs that minimize the demand charge. In the second step, there are three strategies used in choosing the sequence of the machine to be processed in the heuristic. Strategy 1 chooses to process machines in a decreasing order of the machine's power, Strategy 2 chooses to process machines in an increasing order of the machine's power, while Strategy 3 chooses to process machines in a random manner.

The computational results for different sizes of problems (i.e., 5, 10, 20, and 30 jobs) show that the heuristic with Strategy 1 outperforms the other two strategies with the maximum of 3 percent deviation from the best integer solution found from the mixed-integer program. The computational time of the heuristic using Strategy 1 is at most 5 second. This shows that the heuristic can be practically used in real-world environment.

Department.....Industrial Engineering.....Student's signature Sunisa Kulpanapinunt

Field of study.....Industrial Engineering.....Advisor's signature .....

Academic year .....2006.....

กิตติกรรมประกาศ

วิทยานิพนธ์เรื่องการวางแผนการผลิตและการควบคุมการใช้พลังงานไฟฟ้าเพื่อลดค่าไฟฟ้า กรณีศึกษาโรงงานผลิตผลิตภัณฑ์บำรุงผม ได้จัดทำขึ้นเมื่อในช่วงเวลาตั้งแต่ ตุลาคม พ.ศ. 2548 จนถึง เมษายน พ.ศ.2550 เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์มหาบัณฑิต ณ ภาควิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์สำเร็จลุล่วงได้ด้วยความกรุณาจาก อ.ดร.สีรง ปรีชานนท์ ที่กรุณารับเป็นอาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งให้การดูแล ให้คำแนะนำเป็นอย่างดี และให้ข้อคิดเห็นต่างๆ ด้วยความเมตตาตลอดการดำเนินการวิจัย ขอกราบขอบพระคุณคณาจารย์ทุกท่านที่ประสิทธิ์ประสาทวิชาความรู้แก่ผู้วิจัย ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบพระคุณ ผศ.ดร.เหรียญ บุญดีสกุลโชค ที่กรุณารับเป็นประธานกรรมการสอบและให้คำแนะนำในการดำเนินการงานวิจัย ขอขอบพระคุณ ผศ.ดร.วิภาวี ธรรมาภรณ์พิลาศ และ ผศ.ดร. ปวีณา เชาวลิตวงศ์ ที่กรุณารับเป็นกรรมการสอบวิทยานิพนธ์และให้การเสนอแนะเกี่ยวกับงานวิจัยนี้

นอกจากนี้ผู้วิจัยขอขอบคุณ นิว จุ พี่แนน อุ๋ม เจน พี รวมถึงเพื่อนๆ ทุกคน และเจ้าหน้าที่ที่ภาควิชาทุกท่าน สำหรับการดูแล ให้คำแนะนำ และประสบการณ์ที่ดีในการเรียนและทำงานวิจัยที่จุฬาลงกรณ์มหาวิทยาลัยแห่งนี้ด้วยคะ

สุดท้ายที่สำคัญมากที่สุดขอกราบขอบพระคุณคุณพ่อและคุณแม่ที่ให้การสนับสนุน ดูแลเอาใจใส่ผู้วิจัยด้วยความรัก ความห่วงใย จนทำให้การวิจัยสำเร็จลุล่วงไปด้วยดี ขอบคุณทุกคนค่ะ

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

หน้า

บทคัดย่อภาษาไทย..... ง
บทคัดย่อภาษาอังกฤษ..... จ
กิตติกรรมประกาศ..... ฉ
สารบัญตาราง..... ฌ
สารบัญภาพ..... ญ

บทที่

1 บทนำ..... 1
1.1 ความเป็นมาและความสำคัญของปัญหา..... 1
1.2 การคิดค่าไฟฟ้า..... 1
1.3 โรงงานกรณีศึกษา..... 10
1.4 วัตถุประสงค์..... 11
1.5 ขอบเขต..... 11
1.6 ประโยชน์ที่คาดว่าจะได้รับ..... 11

2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง..... 12
2.1 รูปแบบของปัญหาการจัดตารางการผลิตและงานวิจัยที่เกี่ยวข้อง……..... 12
2.2 ความหมายของการจัดตารางการผลิต..... 13
2.2.1 วิธีการต่างๆ ในการจัดตารางการผลิต..... 15
2.2.2 กฎต่างๆที่เป็น ฮิวริสติก (Heuristics)..... 16
2.2.3 ข้อจำกัดในการจัดตารางการผลิต (Constrain) 17
2.2.4 การลดต้นทุนการผลิต..... 18
2.3 การจัดการพลังงาน..... 18

3 การพัฒนาวิธีการหาคำตอบ..... 20

บทที่ หน้า

3.1 ลักษณะการทำงานของโรงงานกรณีศึกษาในส่วนการผสมผลิตภัณฑ์ในอาคารหลัก....20
3.2 ตัวแบบเชิงคณิตศาสตร์ของปัญหางานวิจัย....21
3.3 ฮิวริสติกสำหรับแก้ปัญหา....24
3.3.1 ตัวแทนคำตอบของปัญหา....24
3.3.2 ฮิวริสติก....25

4 การทดสอบฮิวริสติกและการวิเคราะห์ผลงานวิจัย....42
4.1 วิธีการทดสอบฮิวริสติก....42
4.2 ผลการทดสอบฮิวริสติก....42
4.2.1 ปัญหาทดสอบที่มีงานที่ผลิต 5 งาน....42
4.2.2 ปัญหาทดสอบที่มีงานที่ผลิต 10 งาน....44
4.2.3 ปัญหาทดสอบที่มีงานที่ผลิต 15 งาน....45
4.2.4 ปัญหาทดสอบที่มีงานที่ผลิต 20 งาน....47
4.2.5 ปัญหาทดสอบที่มีงานที่ผลิต 30 งาน....48
4.2.6 การเปรียบเทียบประสิทธิภาพของฮิวริสติก....50
4.3 การวิเคราะห์ผล....53

5 สรุปผลงานวิจัย....54
5.1 สรุปผลงานวิจัย....54
5.2 ผลประโยชน์ที่ได้รับจากงานวิจัย....55
5.3 ข้อเสนอแนะสำหรับงานวิจัยในอนาคต....55

รายการอ้างอิง....56
ภาคผนวก....57
ภาคผนวก ก ตัวแบบเชิงคณิตศาสตร์บน AMPL....58
ภาคผนวก ข โปรแกรมสร้างข้อมูลนำเข้าสำหรับทดสอบฮิวริสติก....60
ภาคผนวก ค โปรแกรมทดสอบฮิวริสติก....63
ประวัติผู้เขียนวิทยานิพนธ์....94

สารบัญตาราง

หน้า

ตารางที่ 1.1 อัตราตามช่วงเวลาของการใช้ (Time of Use Rate: TOU) 5
ตารางที่ 3.1 อาร์เรย์ 2 มิติ แสดงตัวอย่างคำตอบของปัญหา..... 25
ตารางที่ 3.2 รายละเอียดของงานทั้งหมด.....32
ตารางที่ 3.3 การเลือกเครื่องจักรในการทำงาน.....33
ตารางที่ 3.4 การเรียงกำลังไฟฟ้าของเครื่องจักร..... 34
ตารางที่ 3.5 ช่วงเวลาที่เหลือจากเมื่องานได้ทำการผลิตแล้ว.....35
ตารางที่ 3.6 ช่วงเวลาสุดท้ายที่สามารถเริ่มทำงานได้..... 37
ตารางที่ 3.7 ช่วงเวลาที่เหลือติดต่อกัน..... 38
ตารางที่ 4.1 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด
สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 5 งาน..... 41
ตารางที่ 4.2 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด
สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 10งาน 42
ตารางที่ 4.3 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด
สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 15 งาน..... 44
ตารางที่ 4.4 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด
สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 20 งาน..... 45
ตารางที่ 4.5 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด
สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 30 งาน..... 47

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญภาพ

หน้า

รูปที่ 1.1 อัตราค่าไฟฟ้าตามช่วงเวลาของวัน (Time of Day Rate: TOD) 4
รูปที่ 1.2 อัตราค่าไฟฟ้าตามช่วงเวลาของการใช้ (Time of Use: TOU) 5
รูปที่ 1.3 การใช้ไฟฟ้าที่แปรเปลี่ยนตามเวลา....7
รูปที่ 1.4 กำลังไฟฟ้าเฉลี่ย (P_{av}) ในช่วงเวลา T ที่กำหนด....8
รูปที่ 1.5 แสดงค่าไฟฟ้าส่วนต่างๆ ของโรงงานอาคารการผลิตหลัก....10
รูปที่ 2.1 ตัวอย่างการแสดงผลในรูปแบบ Gantt Chart....14
รูปที่ 2.2 ตัวอย่าง Gantt Chart....14
รูปที่ 2.3 โครงสร้างของการจัดงานด้วยวิธี Dynamic Programming....15
รูปที่ 3.1 รูปแบบการจัดตารางการผลิตของโรงงาน....20
รูปที่ 3.2 แนวโน้มเวลาในการหาคำตอบของตัวแบบเชิงคณิตศาสตร์....24
รูปที่ 3.3 ฮิวริสติกสำหรับการแก้ปัญหา....26
รูปที่ 3.4 การจัดงานลงเครื่องจักรเบื้องต้น....28
รูปที่ 3.5 ขั้นตอนในการหาเวลาเริ่มต้นทำการผลิต....30
รูปที่ 3.6 งานที่ 8 เริ่มทำงาน....36
รูปที่ 3.7 งานที่ 1 เริ่มทำงาน....39
รูปที่ 3.8 ผลิตครบทุกงาน....40
รูปที่ 3.9 ตารางการทำงานเมื่อมีการปรับปรุงคำตอบ....41
รูปที่ 4.1 เปรียบเทียบค่าความผิดพลาดของฮิวริสติกแบบต่างๆ....51
รูปที่ 4.2 เปรียบเทียบผลลัพธ์การหาคำตอบฮิวริสติกทั้ง 3 แบบใน 5 งาน 10 งาน 15 งาน 20 งาน และ 30 งาน....52
รูปที่ 4.3 เวลาที่ใช้ในการค้นหาคำตอบด้วยฮิวริสติกที่นำเสนอ....53

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

พลังงานไฟฟ้ามีบทบาทสำคัญในชีวิตประจำวันของมนุษย์เป็นอย่างมาก อีกทั้งยังเป็นสิ่งสำคัญในการขับเคลื่อนธุรกิจโรงงานอุตสาหกรรมต่างๆ ให้เดินหน้าต่อไปได้ ในปัจจุบันนี้การใช้พลังงานไฟฟ้าอย่างไม่มีประสิทธิภาพจะทำให้ธุรกิจ และโรงงานอุตสาหกรรมต่างๆ ต้องแบกภาระต้นทุนด้านพลังงานสูงเกินจำเป็น

อุตสาหกรรมการผลิตนั้นมีค่าใช้จ่ายในการดำเนินการทั้งหมดที่เกิดขึ้น จะเห็นได้ว่าค่าใช้จ่ายด้านพลังงานไฟฟ้าเป็นส่วนสำคัญของอุตสาหกรรม ซึ่งการคิดค่าไฟฟ้าจะมีการคิดตามประเภทของผู้ใช้ไฟฟ้าจะทำให้มีอัตราค่าไฟฟ้าที่แตกต่างกัน ค่าไฟฟ้าที่มากหรือน้อยนั้นขึ้นอยู่กับหน่วยที่ใช้และค่าความต้องการพลังงานไฟฟ้าสูงสุดเป็นส่วนหลักของค่าไฟฟ้า ซึ่งถ้าไม่สามารถควบคุมค่าความต้องการพลังงานไฟฟ้า (Demand Charge) ให้สม่ำเสมอได้ จะทำให้เกิดค่าความต้องการพลังงานไฟฟ้าสูงสุดเกิดขึ้นได้ เนื่องจากเมื่อไม่มีการวางแผนการใช้พลังงานไฟฟ้า การเดินเครื่องจักรในกระบวนการการผลิตเป็นไปโดยไม่มีแบบแผน และแนวทางการเดินเครื่องที่เป็นกิจจะลักษณะ

การลดค่าไฟฟ้าถือว่าเป็นส่วนหนึ่งที่จะสามารถลดต้นทุนการผลิตลงได้ การลดค่าใช้จ่ายในด้านของไฟฟ้านี้ จะมีวิธีการคิดค่าไฟฟ้าหลายขั้นตอนซึ่งแสดงให้ดูในส่วนต่อไป ซึ่งจะใช้การจัดตารางการผลิตงานเข้าสู่เครื่องจักรเพื่อให้เกิดค่าความต้องการพลังงานไฟฟ้าให้มีความสม่ำเสมอและคงที่ที่สุด สามารถทำได้โดยการหลีกเลี่ยงการใช้เครื่องจักรพร้อมๆ กันในเวลาเดียวกัน

1.2 การคิดค่าไฟฟ้า

การคิดค่าไฟฟ้าของการไฟฟ้านครหลวง และส่วนภูมิภาคนั้นมีการคิดด้วยกันหลากหลายวิธี ซึ่งต้องรู้ว่าอุตสาหกรรมนั้นๆ อยู่ในประเภทของผู้ใช้ไฟฟ้า เมื่อทราบแล้วขั้นต่อไปก็ต้องทราบว่ารูปแบบอัตราค่าไฟฟ้าว่าอยู่ในรูปแบบใด และมีส่วนประกอบในการคำนวณอะไรบ้าง ซึ่งมีรายละเอียดดังนี้

1.2.1 ส่วนประกอบในการคำนวณค่าไฟฟ้า

ค่าไฟฟ้าที่การไฟฟ้านครหลวงและการไฟฟ้าส่วนภูมิภาค เรียกเก็บจากผู้ใช้ไฟฟ้าในแต่ละเดือน จะประกอบด้วยค่าไฟฟ้าหลายส่วนทั้งนี้ขึ้นอยู่กับประเภทของผู้ใช้ไฟฟ้า และปริมาณการใช้ไฟฟ้าของผู้ใช้ไฟฟ้า ซึ่งประกอบไปด้วย

ก. ค่าความต้องการพลังงานไฟฟ้า (Demand Charge) ความต้องการพลังงานไฟฟ้าที่เกิดขึ้นในแต่ละเดือนที่การไฟฟ้าฯ นำมาคิดค่าไฟฟ้ากับผู้ใช้ไฟฟ้านั้น คือ ความต้องการพลังงานไฟฟ้ามีหน่วยเป็นกิโลวัตต์ที่เป็นค่าเฉลี่ยใน 15 นาทีใดๆ ที่สูงที่สุดในช่วง On Peak ในแต่ละเดือน ค่าความต้องการพลังงานไฟฟ้ามีหน่วยเป็นบาทต่อกิโลวัตต์

ข. ค่าพลังงานไฟฟ้า (Energy Charge) เป็นค่าพลังงานไฟฟ้าที่ผู้ใช้ไฟฟ้าได้ใช้ไปในรอบเดือนนั้นๆ อัตราค่าพลังงานไฟฟ้ามีหน่วยเป็น บาทต่อกิโลวัตต์-ชั่วโมง หรือบาทต่อหน่วย

ค. ค่าบริการ (Service Charge) เป็นค่าบริการเกี่ยวกับเครื่องวัดฯ ค่าดำเนินการจดหน่วย จัดทำใบเสร็จรับเงินค่าไฟฟ้า และดำเนินการจัดเก็บไฟฟ้ามีหน่วยเป็น บาทต่อเดือน ค่าไฟฟ้า

ง. ค่าตัวประกอบการปรับอัตราค่าไฟฟ้าโดยอัตโนมัติ (Ft) เป็นค่าใช้จ่ายที่ไม่อยู่ในการควบคุมของการไฟฟ้าฯ เช่น ค่าราคาเชื้อเพลิงในการผลิตไฟฟ้าเปลี่ยนไปจากฐานที่ใช้ในการกำหนดอัตราค่าไฟฟ้า

จ. ค่าภาษีมูลค่าเพิ่ม (VAT) ส่วนประกอบของอัตราค่าไฟฟ้าส่วนนี้ ปัจจุบันเก็บอัตราร้อยละ 7

1.2.2 การกำหนดประเภทผู้ใช้ไฟฟ้า

การไฟฟ้าได้เรียบเก็บค่าไฟฟ้าจากผู้ใช้ไฟฟ้าทั้งหลายตามอัตราค่าไฟฟ้า โดยในโครงสร้างอัตราค่าไฟฟ้าได้จัดแบ่งผู้ใช้ออกเป็น 7 ประเภทคือ

ประเภทที่ 1 บ้านอยู่อาศัยสำหรับการใช้ไฟฟ้าในบ้านเรือนที่อยู่อาศัย วัด โบสถ์ของศาสนาต่างๆ ตลอดจนบริเวณที่เกี่ยวข้องโดยแบ่งเป็นบ้านอยู่อาศัยขนาดเล็กและขนาดใหญ่ใช้พลังงานไฟฟ้าไม่เกิน 150 หน่วยต่อเดือน

ประเภทที่ 2 กิจการขนาดเล็ก สำหรับการใช้ไฟฟ้าเพื่อประกอบธุรกิจ ธุรกิจรวมกับที่อยู่อาศัย อุตสาหกรรมและหน่วยงานรัฐวิสาหกิจอื่นๆ โดยมีความต้องการพลังงานไฟฟ้าเฉลี่ยใน 15 นาทีที่สูงสุดต่ำกว่า 30 กิโลวัตต์

ประเภทที่ 3 กิจการขนาดกลาง สำหรับการใช้ไฟฟ้าเพื่อประกอบธุรกิจ อุตสาหกรรม และหน่วยงานรัฐวิสาหกิจ ซึ่งมีความต้องการพลังไฟฟ้าเฉลี่ยใน 15 นาทีที่สูงสุดตั้งแต่ 30 ถึง 999 กิโลวัตต์ และมีปริมาณการใช้พลังงานไฟฟ้าเฉลี่ย 3 เดือนไม่เกิน 250,000 หน่วยต่อเดือน

ประเภทที่ 4 กิจการขนาดใหญ่ สำหรับการใช้ไฟฟ้าเพื่อประกอบธุรกิจ อุตสาหกรรม ส่วนราชการ ซึ่งมีความต้องการพลังไฟฟ้าเฉลี่ยใน 15 นาทีที่สูงสุดตั้งแต่ 1,000 กิโลวัตต์ขึ้นไป หรือมีปริมาณการใช้พลังงานไฟฟ้าเฉลี่ย 3 เดือนเกินกว่า 250,000 หน่วยต่อเดือน

ประเภทที่ 5 กิจการเฉพาะอย่าง สำหรับการใช้เพื่อประกอบกิจการโรงแรม กิจการให้เช่าอาศัย ซึ่งมีความต้องการพลังงานไฟฟ้าเฉลี่ยใน 15 นาทีสูงสุดตั้งแต่ 30 กิโลวัตต์

ประเภทที่ 6 ส่วนงานราชการตามกฎหมายว่าด้วยระเบียบการบริหารงานท้องถิ่นซึ่งมีความต้องการพลังงานไฟฟ้าเฉลี่ยใน 15 นาทีสูงสุดต่ำกว่า 1,000 กิโลวัตต์และมีปริมาณการใช้ไฟฟ้าเฉลี่ยใน 3 เดือนไม่เกิน 250,000 หน่วยต่อเดือน

ประเภทที่ 7 สูบน้ำเพื่อการเกษตร สำหรับการใช้ไฟฟ้ากับเครื่องสูบน้ำเพื่อการเกษตรของหน่วยราชการ สหกรณ์เพื่อการเกษตร

1.2.3 รูปแบบอัตราค่าไฟฟ้า

เป็นรูปแบบของการเสียค่าไฟฟ้าว่าจะมีการเสียค่าไฟฟ้าที่ใช้ไปนั้นเป็นแบบไหน ซึ่งสามารถจำแนกได้เป็น 4 แบบ

1.2.3.1 อัตราค่าไฟ 1 ส่วน (One-Part Tariff)

เป็นอัตราค่าไฟฟ้าที่รวมค่าความต้องการพลังงานไฟฟ้ากับค่าพลังงานไฟฟ้าเข้าด้วยกันอยู่ในรูปของค่าพลังงานไฟฟ้า แบ่งออกได้เป็นมี 3 แบบ ย่อยคือ

ก. อัตราคงที่ (Flat-Rate) เป็นการคิดอัตราต่อหน่วยคงที่ตลอดไม่ว่าจะใช้ไฟฟ้ามากน้อยเพียงใดก็ตาม

ข. อัตราแบบกึ่งคงที่ (Semi Flat-Rate) เป็นการคิดอัตราหน่วยคงที่ 2 ระดับ การใช้ไฟฟ้าในช่วงแรกจะคิดอัตราค่าไฟในราคาที่ต่ำกว่า แต่เมื่อใช้เกินกำหนดจะคิดในอัตราที่แพงขึ้น

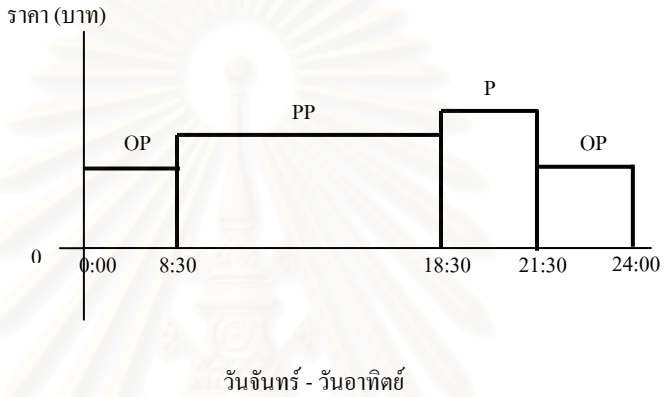
ค. อัตราแบบก้าวหน้า (Progressive Rate) เป็นการคิดอัตราต่อหน่วยสูงขึ้นเมื่อมีการใช้ไฟฟ้ามากขึ้น เพื่อให้สะท้อนถึงการใช้ไฟฟ้าอย่างมีประสิทธิภาพ

1.2.3.2 อัตราค่าไฟฟ้า 2 ส่วน (Two-Part Tariff)

เป็นอัตราค่าไฟฟ้าที่แยกค่าความต้องการพลังไฟฟ้าและค่าพลังงานไฟฟ้าออกจากกันเพื่อให้สะท้อนถึงต้นทุนในการผลิตไฟฟ้าได้อย่างชัดเจนขึ้น อย่างไรก็ตามอัตราค่าความต้องการพลังงานไฟฟ้าและอัตราค่าพลังงานไฟฟ้ายังเป็นแบบอัตราคงที่ อัตราค่าไฟฟ้า 2 ส่วนนี้ในประกาศอัตราค่าไฟฟ้าของการไฟฟ้า จะเรียกเป็นอัตราปกติ

1.2.3.3 อัตราค่าไฟฟ้าตามช่วงเวลาของวัน (Time of Day Rate: TOD)

เป็นอัตราค่าไฟฟ้าที่แยกค่าความต้องการพลังงานไฟฟ้า และค่าพลังงานไฟฟ้าออกจากกัน โดยมีการแบ่งแยกค่าความต้องการพลังงานไฟฟ้าให้มีอัตราแตกต่างกันตามช่วงเวลาของวัน ทั้งนี้เพื่อสะท้อนถึงต้นทุนของการผลิต การส่ง และการจำหน่ายไฟฟ้า โดยช่วงเวลา 24 ชั่วโมงในแต่ละวัน จะถูกแบ่งออกเป็น 3 ช่วงเวลา ดังรูปที่ 1.1



รูปที่ 1.1 อัตราค่าไฟฟ้าตามช่วงเวลาของวัน (Time of Day Rate: TOD)

กำหนดให้ P คือ ความต้องการพลังไฟฟ้าสูงสุดในช่วง ON PEAK (กิโลวัตต์)

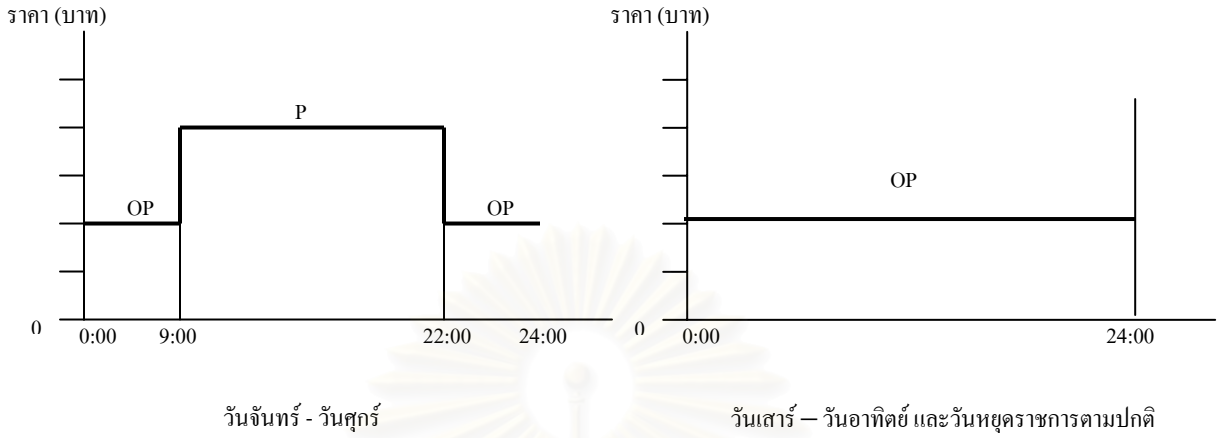
PP คือ ความต้องการพลังไฟฟ้าสูงสุดในช่วง PARTIAL PEAK (กิโลวัตต์)

OP คือ ความต้องการพลังไฟฟ้าสูงสุดในช่วง OFF PEAK (กิโลวัตต์)

สำหรับค่าพลังงานไฟฟ้าในอัตรา TOD นี้ยังเป็นแบบอัตราคงที่ คือ พลังงานไฟฟ้าทุกหน่วยมีราคาเท่ากัน ผู้ใช้ไฟฟ้าที่เสียค่าไฟฟ้าตามอัตรานี้ คือ

1.2.3.4. อัตราค่าไฟฟ้าตามช่วงเวลาของการใช้ (Time of Use: TOU)

เป็นอัตราค่าไฟฟ้าที่แยกค่าความต้องการพลังงานไฟฟ้าและค่าพลังงานไฟฟ้าออกจากกัน โดยมีการแบ่งแยกค่าพลังงานไฟฟ้าให้มีอัตราแตกต่างตามช่วงเวลาของวันและของสัปดาห์หรือตามช่วงเวลาของการใช้ ทั้งนี้เพื่อให้สะท้อนถึงต้นทุนของการผลิต การส่ง และการจำหน่ายไฟฟ้า โดยแบ่งเวลาในแต่ละสัปดาห์ออกเป็น 2 ช่วงเวลา ดังรูปที่ 1.2



รูปที่ 1.2 อัตราค่าไฟฟ้าตามช่วงเวลาของการใช้ (Time of Use: TOU)

สำหรับค่าความต้องการพลังไฟฟ้าในอัตราค่าไฟฟ้าตามช่วงเวลาของการใช้ (Time of Use: TOU) นี้จะเป็นแบบอัตราคงที่ โดยคิดค่าความต้องการพลังไฟฟ้าเฉพาะในช่วง On Peak เท่านั้น

1.2.4 อัตราค่าไฟฟ้าแบบตามช่วงเวลาของการใช้ (Time of Use Rate: TOU)

รายละเอียดที่จะกล่าวต่อไปนี้ จะเป็นรายละเอียดเกี่ยวกับการใช้ไฟฟ้าของโรงงานกรณีศึกษาซึ่งจัดอยู่ในผู้ใช้ไฟฟ้าประเภทใช้ไฟฟ้าประเภทที่ 3.2 ผู้ใช้ไฟฟ้าแบบ TOU

ตารางที่ 1.1 อัตราตามช่วงเวลาของการใช้ (Time of Use Rate: TOU)

ระดับแรงดัน	ค่าความต้องการพลังงานไฟฟ้า (บาท/กิโลวัตต์)	ค่าพลังงานไฟฟ้า (บาท/หน่วย)		ค่าบริการ (บาท/เดือน)
	Peak	Peak	Off Peak	
1.แรงดันตั้งแต่ 69 กิโลโวลท์ขึ้นไป	74.14	2.6136	1.1726	228.17
2 แรงดัน 22-33 กิโลโวลท์	132.93	2.6950	1.1914	228.17
3 แรงดันต่ำกว่า 22 กิโลโวลท์	210.00	2.8408	1.2246	228.17

Peak: วันจันทร์ - ศุกร์ 09.00 น. - 22.00 น.

Off Peak: วันจันทร์ - ศุกร์ 22.00 น. - 09.00 น. และวันเสาร์ วันอาทิตย์ วันหยุดราชการตามปกติ (ไม่รวมวันหยุดชดเชย) ทั้งวัน

อัตราขั้นต่ำ : ค่าไฟฟ้าต่ำสุดต้องไม่ต่ำกว่าร้อยละ 70 ของค่าความต้องการพลังไฟฟ้าสูงสุดในรอบ 12 เดือนที่ผ่านมาสิ้นสุดในเดือนปัจจุบัน

ตัวอย่างที่ 1 เป็นตัวอย่างที่เป็นการคำนวณค่าไฟฟ้าแบบตามช่วงเวลาของการใช้ (Time of Use Rate: TOU) โดยข้อมูลมีดังนี้

ข้อมูล	ประเภทผู้ใช้ไฟฟ้า	3.2
	แรงดันไฟฟ้าที่ใช้	22-33 กิโลโวลท์
	พลังงานไฟฟ้าสูงสุด	674 กิโลวัตต์
	พลังงานไฟฟ้าช่วง Peak	70,780 หน่วย
	พลังงานไฟฟ้าช่วง Off Peak	34,980 หน่วย
	รวมหน่วยทั้งหมด	105,760 หน่วย
	ค่าบริการรายเดือน	228.17 บาท
	ค่า Ft	0.4683

การคำนวณ

ก. ค่าพลังงานไฟฟ้าสูงสุด = พลังงานไฟฟ้าสูงสุด × ค่าความต้องการพลังงานไฟฟ้า

= 674×132.93

= 89,594.82 บาท

ข. ค่าพลังงานไฟฟ้าช่วงPeak = พลังงานไฟฟ้าช่วง Peak × ค่าพลังงานไฟฟ้า

= 70,780×2.6590

= 188,204.02 บาท

ค. ค่าพลังงานไฟฟ้าช่วง Off Peak = พลังงานไฟฟ้าช่วง Off Peak × ค่าพลังงานไฟฟ้า

= 34,980×1.1914

= 41,675.17 บาท

ง. ค่าไฟการปรับอัตราค่าไฟฟ้าอัตโนมัติ (Ft) = (พลังงานไฟฟ้าช่วง Peak + พลังงานไฟฟ้า

ช่วง Off Peak) × ค่า Ft

= (70,780 + 34980) × 0.4683

= 49,527.41 บาท

จ. รวมเงินค่าไฟฟ้าก่อนรวมภาษี = ค่าพลังไฟฟ้าสูงสุด + ค่าพลังงานไฟฟ้าช่วงPeak + ค่าพลังงานไฟฟ้าช่วง Off Peak + ค่าไฟการปรับอัตราค่าไฟฟ้าอัตโนมัติ (Ft)

= 89,594.82 + 188,204.02 + 41,675.17+ 49,527.41+ 228.17

= 369,229.59 บาท

ฉ. ภาษีมูลค่าเพิ่ม = รวมเงินค่าไฟฟ้าก่อนรวมภาษี × อัตราภาษีมูลค่าเพิ่ม

= 369,229.59×0.07

= 25,846.07 บาท

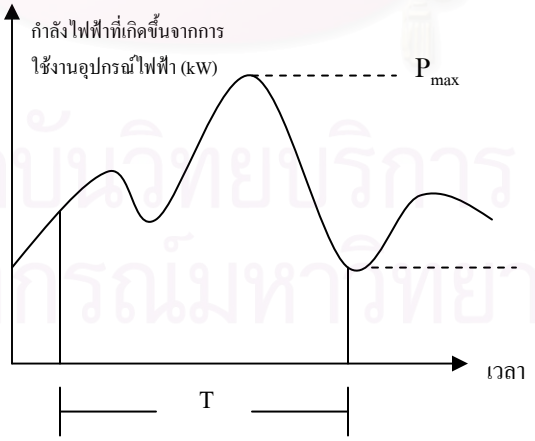
ช. รวมเงินค่าไฟฟ้าหลังรวมภาษี = รวมเงินค่าไฟฟ้าก่อนรวมภาษี + ภาษีมูลค่าเพิ่ม

= 369,229.59+25,846.07

= 395,075.66 บาท □

1.2.5 การวัดความต้องการพลังไฟฟ้าสูงสุด

ความต้องการพลังไฟฟ้าในที่นี้ หมายถึง ความต้องการพลังไฟฟ้าเฉลี่ยในช่วงเวลา 15 นาที



รูปที่ 1.3 การใช้ไฟฟ้าที่แปรเปลี่ยนตามเวลา

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

รูปที่ 1.3 แสดงลักษณะการใช้ไฟฟ้าที่แปรเปลี่ยนตามเวลา ปริมาณพลังงานไฟฟ้าที่ถูกใช้ไปทั้งหมดในช่วงเวลา T หาได้จากสมการที่ 1.1

$$E = \int_0^T P(t)\,dt \qquad (1.1)$$

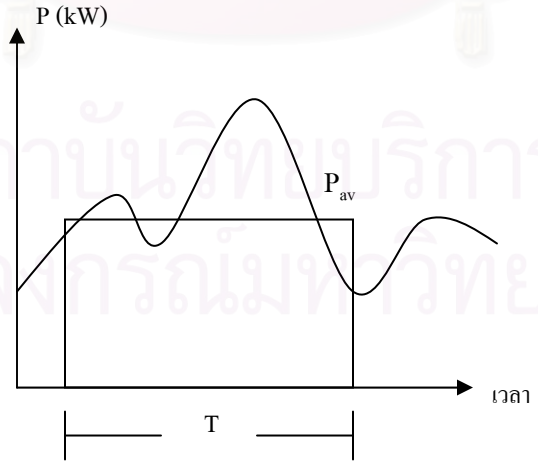
เมื่อ E คือ ปริมาณพลังงานไฟฟ้า (กิโลวัตต์-ชั่วโมง หรือหน่วย)

P(t) คือ กำลังไฟฟ้าที่เกิดขึ้น ณ เวลาใดๆ (กิโลวัตต์)

T คือ คาบเวลาที่สนใจจะหาปริมาณพลังงานที่ใช้ไป (ชั่วโมง)

เครื่องวัดหน่วยไฟฟ้าจะทำการวัดปริมาณพลังงานที่ใช้ไปในเวลา T ที่กำหนด เช่น 1 วัน, 1 สัปดาห์หรือ 1 เดือน โดยใช้หลักการวัดตามสมการที่ 1.1

ในการคิดค่าไฟฟ้าจากผู้ใช้ไฟฟ้าปริมาณมากๆ นอกจากการไฟฟ้าฯ จะคิดค่าพลังงานไฟฟ้าแล้ว การไฟฟ้าฯ ยังคิดค่าความต้องการพลังงานไฟฟ้าสูงสุด (Maximum Demand) ด้วย จากรูปที่ 1.3 หากการใช้ไฟฟ้าฯ นำค่า P_{max} ไปคิดค่าไฟฟ้าก็คงจะไม่เป็นธรรมกับผู้ใช้ไฟฟ้า เพราะ P_{max} เกิดขึ้นในช่วงเวลาสั้นๆ ในทางตรงกันข้าม หากจะนำค่า P_{min} ไปคิดค่าไฟฟ้าก็คงจะไม่เป็นธรรมกับการไฟฟ้าฯ ด้วยเช่นกัน เพราะการไฟฟ้าต้องลงทุนสร้างระบบเพื่อให้สามารถจ่ายกำลังไฟฟ้าถึงระดับ P_{max} ด้วย แต่คิดเงินจากผู้ใช้ไฟฟ้าได้นิดเดียว เพื่อความเป็นธรรมด้วยกันทั้ง 2 ฝ่าย การไฟฟ้าจึงใช้กำลังไฟฟ้าเฉลี่ยในช่วงเวลาที่กำหนด ซึ่งในที่นี้คือ 15 นาที ในการคิดเงินค่าความต้องการพลังงานไฟฟ้า



รูปที่ 1.4 กำลังไฟฟ้าเฉลี่ย (P_{av}) ในช่วงเวลา T ที่กำหนด

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

รูปที่ 1.4 แสดงค่ากำลังไฟฟ้าเฉลี่ย (P_{av}) ในช่วงเวลา T ที่กำหนด ที่จะทำให้เกิดปริมาณพลังงานไฟฟ้า E เท่ากับพื้นที่ในรูปที่ 1.3 นั้นคือ

$$P_{av} \times T = E = \int_0^T P(t)\,dt \qquad (1.2)$$

ความต้องการพลังไฟฟ้าเฉลี่ยในช่วงเวลา 15 นาที จะวัดจากปริมาณพลังงานไฟฟ้าที่ถูกใช้ไปในช่วงเวลา 15 นาที โดยใช้ความสัมพันธ์ตามสมการที่ 1.2 คือ

$$P_{av}(kW) \times \frac{1}{4}(h) = E_{in\,15\,\min}(Unit)$$

หรือ

$$P_{av} = Demand = 4 \times E_{in\,15\,\min} \qquad (1.3)$$

ดังนั้นความต้องการไฟฟ้าจึงมีค่าเท่ากับ 4 เท่า ของพลังงานไฟฟ้าที่ถูกใช้ไปในช่วงเวลาทุกๆ 15 นาที

1.2.6 แนวทางการลดค่าไฟฟ้าเบื้องต้น

สำนักคณะกรรมการนโยบายพลังงานแห่งชาติได้กล่าวไว้ว่าผู้ใช้ไฟฟ้าในประเภทที่ 3.2 จะเสียค่าความต้องการพลังไฟฟ้าในอัตราคงที่ โดยคิดค่าความต้องการพลังไฟฟ้าเฉพาะความต้องการพลังไฟฟ้าสูงสุดที่เกิดขึ้นระหว่างเวลา 9:00 น. - 22:00 น. ของวันจันทร์ถึงวันศุกร์เท่านั้น ส่วนค่าพลังงานไฟฟ้านั้น จะเสียในอัตราตามช่วงเวลาของการใช้กล่าวคือ หากใช้ไฟฟ้าในช่วงเวลา 9:00 น. - 22:00 น. วันจันทร์ถึงวันศุกร์จะเสียค่าพลังงานไฟฟ้าแพงที่สุด โดยการใช้ไฟฟ้าในช่วงกลางคืนของวันจันทร์ถึงวันศุกร์ระหว่างเวลา 22:00 น - 9:00 น. วันเสาร์ วันอาทิตย์และวันหยุดราชการตามปกติทั้งวันจะเสียค่าพลังงานไฟฟ้าถูกกว่า ดังนั้นผู้ใช้ไฟฟ้าในกลุ่มนี้จึงควรใช้แนวทางการลดค่าไฟฟ้าดังต่อไปนี้

- การเพิ่มประสิทธิภาพของอุปกรณ์โดยการเลือกใช้อุปกรณ์ที่มีประสิทธิภาพสูง และการใช้งานเพิ่มให้สูงได้โดยการเปิดใช้งานเท่าที่จำเป็น ใช้งานอย่างเหมาะสม
- การใช้ไฟฟ้าในช่วงที่ต้องเสียค่าความต้องการพลังงานไฟฟ้า จะต้องใช้ไฟฟ้าอย่างสม่ำเสมอ เพื่อไม่ให้เกิดความต้องการพลังไฟฟ้าสูงสุดสูงเกินควร

- ปรับกิจกรรมการทำงานใหม่ โดยเพิ่มกิจกรรมในช่วงกลางคืนถึงตอนเช้า (22:00 น - 9:00 น. ของวันจันทร์ถึงวันศุกร์) ให้มากขึ้นพร้อมทั้งเพิ่มกิจกรรมในวันเสาร์ วันอาทิตย์ และวันหยุดราชการตามปกติ ซึ่งมีค่าพลังงานไฟฟ้าถูกเท่ากับช่วงกลางคืน

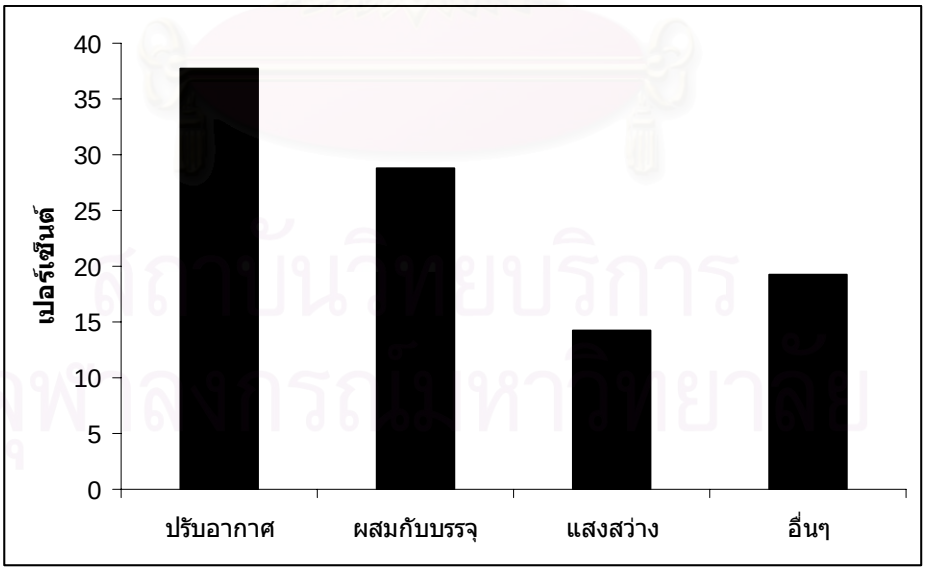
1.3 โรงงานกรณีศึกษา

โรงงานที่ทำการศึกษาเป็นโรงงานที่ทำการผลิตเครื่องสำอางที่มีผลิตภัณฑ์หลากหลายและเป็นการผลิตแบบตามงาน (Job Shop) เป็นกลุ่มใช้ไฟฟ้าประเภทที่ 3.2 ที่มีการใช้อัตราค่าไฟฟ้าแบบตามช่วงเวลาของการใช้ (Time of Use Rate: TOU) ที่มีส่วนการผลิตด้วยกันอยู่ 2 ส่วนหลักคือ

ก. ส่วนของอาคารการผลิตหลัก มีการผลิตผลิตภัณฑ์ที่ไม่มีการอัดแก๊ส เช่น แชมพู ทรีทเม้นท์ เจล โลชั่น น้ำยาดัดผม เป็นต้น

ข. ส่วนของอาคารการผลิตที่ผลิตภัณฑ์มีการอัดแก๊ส เช่น สเปรย์อัดแก๊ส มูส เป็นต้น

ซึ่งทั้งสองส่วนจะแยกการคิดค่าไฟฟ้าออกจากกัน แต่ส่วนของโรงการผลิตหลักมีสัดส่วนการใช้ไฟฟ้าที่มากกว่าโรงที่ใช้การอัดแก๊ส เนื่องจากมีปริมาณการผลิตที่มากกว่าจึงทำให้ส่วนของอาคารการผลิตหลักต้องมีการใช้ไฟฟ้าอย่างมีประสิทธิภาพเพื่อที่จะได้ทำให้ต้นทุนในการผลิตมีค่าลดลง ซึ่งมีค่าไฟฟ้าส่วนต่างๆ ของโรงงานของอาคารการผลิตหลักดังรูปที่ 1.5



รูปที่ 1.5 แสดงค่าไฟฟ้าส่วนต่างๆ ของโรงงานอาคารการผลิตหลัก

จากรูปที่ 1.5 แสดงให้เห็นว่าค่าไฟฟ้าส่วนใหญ่แล้ว จะเกิดจากส่วนปรับอากาศเป็นส่วนใหญ่ ส่วนที่รองลงมาคือ ส่วนของห้องผสมผลิตภัณฑ์กับห้องบรรจุผลิตภัณฑ์ ในส่วนของห้องบรรจุผลิตภัณฑ์นั้นมีการบรรจุตลอดเวลาการทำงานจึงเป็นส่วนที่นำมาควบคุมเวลาการผลิตไม่ได้ ต่างจากส่วนของห้องผสมผลิตภัณฑ์ที่มีการผลิตเป็นช่วงเวลาทำให้สามารถควบคุมเวลาในการผลิตได้ จึงได้มีการนำส่วนของห้องผสมผลิตภัณฑ์มาพิจารณาในการวางแผนการผลิต

ปัจจุบันนี้โรงงานมีปริมาณการใช้พลังงานไฟฟ้าที่มากสามารถลดปริมาณการใช้พลังงานไฟฟ้าได้โดยการใช้เครื่องจักรที่มีกำลังไฟฟ้าที่น้อย จำนวนเครื่องจักรที่น้อย และระยะเวลาที่น้อยด้วย ส่วนการผสมผลิตภัณฑ์ที่ไม่แน่นอนทำให้เกิดช่วงที่มีการใช้ไฟฟ้าในปริมาณที่มากจึงทำให้เกิดค่าความต้องการพลังงานไฟฟ้าสูงสุดสูงที่นำไปคิดค่าไฟฟ้า และมีช่วงที่มีการใช้ไฟฟ้าที่น้อย ทำให้ไม่เกิดความสม่ำเสมอในการใช้ไฟฟ้าเกิดขึ้น จึงทำให้ต้องมีการควบคุมการใช้ไฟฟ้าให้มีประสิทธิภาพ

1.4 วัตถุประสงค์

เพื่อพัฒนาวิธีการที่สามารถช่วยในการตัดสินใจในการวางแผนการผลิตส่วนของการผสม เพื่อลดค่าไฟฟ้าซึ่งจะทำให้ต้นทุนการผลิตลดลง

1.5 ขอบเขต

ศึกษาการวางแผนการผลิตในส่วนของอาคารการผลิตหลักในส่วนห้องผสมผลิตภัณฑ์ของโรงงานกรณีศึกษา

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1. มีวิธีการช่วยตัดสินใจในการวางแผนการผลิตเพื่อที่ทำให้ค่าความต้องการพลังงานไฟฟ้ามีความสม่ำเสมอ และคงที่
2. สามารถทำให้ค่าไฟฟ้าลดลงเป็นผลให้ต้นทุนการผลิตลดลงด้วย
3. ช่วยเพิ่มประสิทธิภาพในการใช้พลังงานไฟฟ้าได้อย่างเหมาะสม
4. ทำให้ระบบการทำงานเป็นระบบมากขึ้นจากการทำงานแบบเดิม
5. เพื่อเป็นแนวทางในการแก้ไขปรับปรุงการใช้พลังงานไฟฟ้า

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 รูปแบบของปัญหาการจัดตารางการผลิตและงานวิจัยที่เกี่ยวข้อง

ปัญหาการจัดตารางการผลิตนั้นเป็นปัญหาที่ได้รับการศึกษาอย่างมาก ซึ่งมีรูปแบบที่สำคัญของการจัดเรียงเครื่องจักรอยู่หลายลักษณะ ขึ้นอยู่กับระบบการทำงานและปัจจัยแวดล้อมต่างๆ คือ การผลิตแบบไหลเลื่อน (Flow Shop) ซึ่งเป็นกระบวนการผลิตที่ต่อเนื่องกันไปตลอด ขั้นตอนการผลิตค่อนข้างแน่นอน ผลิตภัณฑ์ที่ใช้ลักษณะการผลิตแบบนี้มักจะมีความเป็นมาตรฐานสูง มีการเปลี่ยนแปลงการผลิตค่อนข้างน้อย และอีกรูปแบบหนึ่ง คือ การผลิตแบบตามงาน (Job Shop) จะมีการจัดผังการผลิตแบบแบ่งตามหน้าที่โดยมีการจัดกลุ่มของเครื่องจักร หรือหน่วยปฏิบัติงานประเภทเดียวกันไว้ด้วยกันซึ่งจะมีความซับซ้อนอยู่มากกว่า รูปแบบส่วนใหญ่ของงาน จะมีจำนวน m การทำงาน แต่ละการทำงานจะใช้เครื่องจักร 1 เครื่อง และบางครั้งอาจจะมีการทำงานซ้ำบนเครื่องจักรเครื่องเดิมมากกว่า 1 ครั้ง

ปัญหาการจัดตารางการผลิตเป็นปัญหาที่ได้รับการศึกษาและวิจัยมาเป็นระยะเวลานาน โดยทั่วๆ ไปแล้วจะมีตัววัดสมรรถภาพต่างๆ เช่น เวลาไหลของงานน้อยที่สุด เวลาปิดงานของระบบมีค่าน้อยที่สุด จำนวนงานล่าช้าน้อยที่สุด โดยการใช้วิธีต่างๆ เช่น วิธีการแตกกิ่งและจำกัดเขต โปรแกรมมิงเชิงจำนวนเต็ม (Integer Programming) และวิธีฮิวริสติก เป็นต้น

การศึกษาวิธีการลดการใช้พลังงานไฟฟ้านั้นมีด้วยกันหลายวิธี เช่น การศึกษาความเป็นไปได้ในการประหยัดค่าไฟฟ้า โดยการเปลี่ยนวิธีคิดค่าไฟฟ้าแบบอัตราอัตราตามช่วงเวลาของการใช้ (Time of Use Rate: TOU) และปรับเปลี่ยนพฤติกรรมการทำงานเพื่อให้เกิดการใช้ไฟฟ้าอย่างคุ้มค่าและประหยัดที่สุด โดยการทำงานแบบเดิมจะเป็นการทำงานแบบ 3 กะ และใช้อัตราการใช้ไฟฟ้าแบบปกติ (จิราพร และสุรินธร, 2004)

การจัดตารางการผลิตเพื่อให้ค่าไฟฟ้าน้อยที่สุดในอุตสาหกรรมที่เป็นการผลิตแบบไหลเลื่อน (Flow Shop) โดยการจัดลำดับการผลิต และการเปลี่ยนอัตราการใช้พลังงานไฟฟ้าให้เหมาะสมกับการทำงาน เพื่อลดอัตราการใช้พลังงานไฟฟ้าโดยการจัดลำดับการผลิตใหม่ จากการเปรียบเทียบการจัดระบบตารางการทำงานใหม่เทียบกับตารางการทำงานเดิม (สุฑามาศ, 2004) และการจัดตารางการผลิตสำหรับระบบการผลิตแบบงาน (Job Shop) ทำเพื่อลดค่าความต้องการไฟฟ้าสูงสุดโดยทำการคิดกระบวนการแก้ปัญหา (Algorithm) ขึ้น 3 วิธีได้แก่ วิธีที่แรก คือ วิธีการจัดตารางการผลิตแบบสุ่ม

(Random) วิธีที่สอง คือ วิธีการจัดตารางการผลิตแบบรวมเวลาการทำงานซึ่งจะทำการหาเวลารวมของงานย่อยในแต่ละงานแล้วทำการจัดตารางการผลิตโดยเรียงจากเวลารวมของงานย่อยที่มีค่าน้อยไปหาเวลารวมของงานย่อยที่มีค่ามากตามลำดับ วิธีที่สาม คือ วิธี การจัดตารางการผลิตแบบผสม จะทำการหาเวลารวมของงานย่อยในแต่ละงานแล้วทำการจัดตารางการผลิตโดยเรียงจากเวลารวมของงานย่อยที่มีค่าน้อยไปหาเวลารวมของงานย่อยที่มีค่ามากตามลำดับซึ่งจะเหมือนกับวิธีที่สอง แต่ในวิธีนี้จะมีการขยับช่วงเวลาการทำงานที่เกิดค่าความต้องการสูงสุดโดยจะเลื่อนออกไปให้ใกล้กับวันกำหนดหลังจากการเลื่อนแล้วส่งผลให้ค่าความต้องการไฟฟ้าสูงสุดลดลง การเปรียบเทียบประสิทธิภาพของทั้ง 3 วิธีจะเห็นได้ว่าวิธีที่ 3 มีค่าความต้องการไฟฟ้าสูงสุดต่ำกว่าวิธีที่ 1 และ 2 จากการทดลองทั้งหมด 17 ตัวอย่าง (จีรศักดิ์และชัชวาลย์, 2004)

ส่วนด้านการควบคุมการใช้พลังงานไฟฟ้ามีการศึกษาในอุตสาหกรรมโรงแรม กรณีศึกษาโรงแรมขนาดกลางและขนาดเล็ก สรุปได้ว่าการประหยัดพลังงานนี้ควรทำในทุกส่วนของการใช้พลังงานไฟฟ้า (กันต์ธร, 2541) มีการศึกษาวิเคราะห์การจัดการด้านพลังงานของกระบวนการผลิตปูนซีเมนต์ที่เกิดขึ้น และได้ดำเนินการปรับปรุงการจัดการด้านพลังงานทำให้มีการใช้พลังงานต่างๆ น้อยลงจากเดิม (เอกสิทธิ, 2543) และการประหยัดพลังงานในโรงงานอุตสาหกรรมปลาป่น การประหยัดพลังงานย่อมให้ผลในการเพิ่มกำไรของกิจการ และการดำเนินโปรแกรมการประหยัดพลังงานในโรงงานอุตสาหกรรมเป็นสิ่งจำเป็น และจะต้องทำเป็นกิจกรรมต่อเนื่องจึงจะได้ผลดี (ศิริกัลยา, 2531)

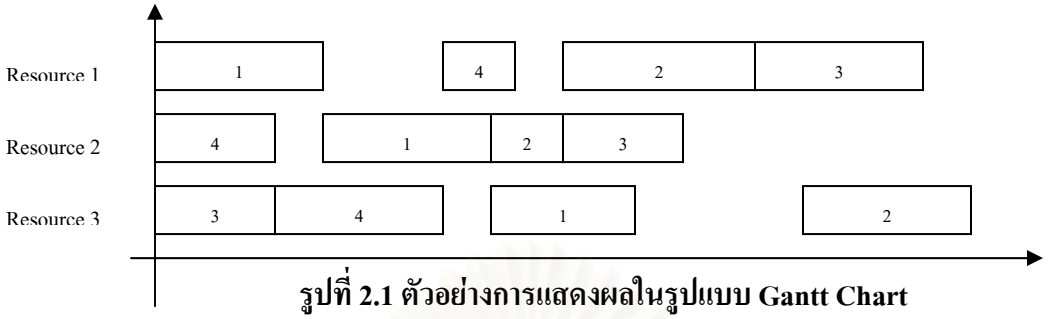
2.2 ความหมายของการจัดตารางการผลิต

มีนักวิจัยหลายท่านได้ให้คำนิยามของการจัดตารางการผลิต (Scheduling) ไว้ดังนี้

Baker (1974): การจัดตาราง เป็นการจัดสรรทรัพยากรภายในเวลาที่มีอยู่ เพื่อการดำเนินการ

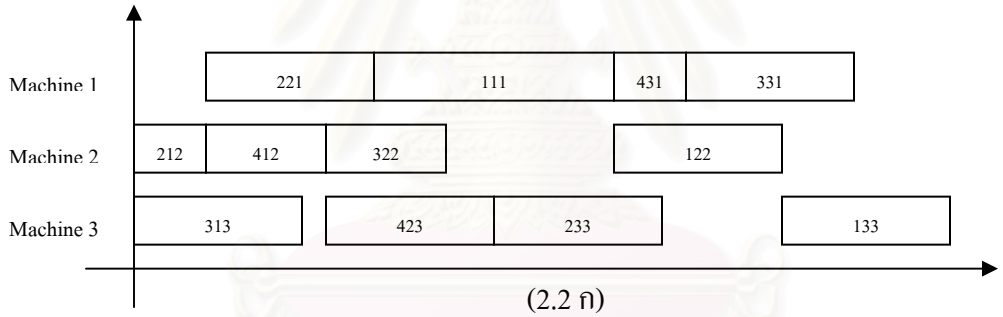
Prabhu และ Baker (1986): การจัดตาราง เป็นกระบวนการของการกำหนดเวลาเริ่มต้นและสิ้นสุดของการทำงานแต่ละงาน สำหรับเครื่องจักรแต่ละเครื่อง

โดยปกติการจัดตารางนั้นจะแสดงผลในรูปของ Gantt Chart ซึ่งเป็นการแสดงความสัมพันธ์ระหว่างการจัดสรรทรัพยากร (เช่น เครื่องจักร คน) กับเวลา ดังรูปที่ 2.1

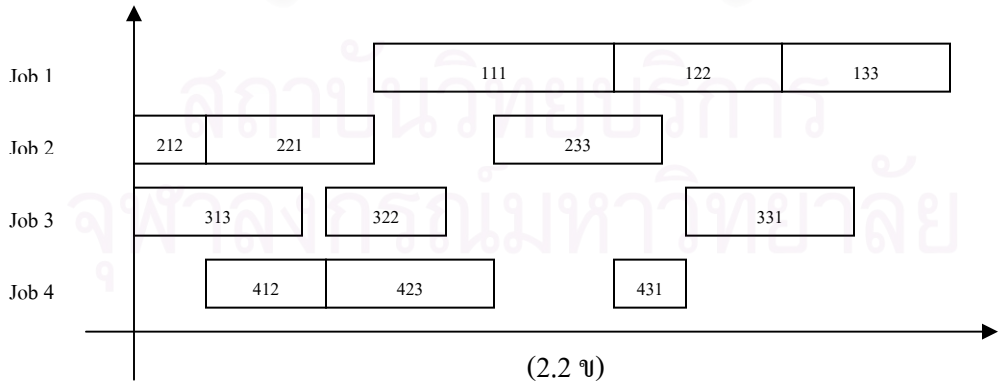


รูปที่ 2.1 ตัวอย่างการแสดงผลในรูปแบบ Gantt Chart

โดยทั่วไปแล้ว Gantt Chart จะแสดงได้ 2 แบบ ดังรูป 2.2(ก) แสดงการทำงานบนเครื่องจักรแต่ละเครื่อง และรูป 2.2(ข) แสดงการทำงานของงานแต่ละงาน โดยเครื่องจักรแต่ละเครื่องไม่สามารถทำงานพร้อมกันมากกว่า 1 การทำงาน และการปฏิบัติงานของแต่ละงานเป็นไปตามลำดับขั้นตอนอย่างไม่เหลื่อมล้ำกัน



(2.2 ก)



(2.2 ข)

รูปที่ 2.2 ตัวอย่าง Gantt Chart

(ก) การทำงานของเครื่องจักรแต่ละเครื่อง

(ข) การทำงานของงานแต่ละงาน

2.2.1 วิธีการต่างๆ ในการจัดตารางการผลิตนั้น มีหลายวิธี ดังต่อไปนี้

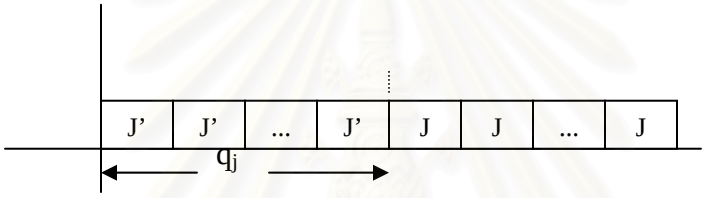
ก. การสับเปลี่ยนคู่ที่อยู่ติดกัน (ADJACENT PAIRWISE INTERCHANGE)

วิธีการนี้ จะเป็นวิธีการสลับตำแหน่งงานแบบเป็นคู่ที่ติดกัน ซึ่งสามารถให้คำตอบที่ดีที่สุด (Optimum Solution)ได้

ข. DYNAMIC PROGRAMMING

การจัดตารางการผลิตโดยวิธี Dynamic Programming จะทำให้วิธีการในการจัดลำดับการผลิตลดลงจาก n! เหลือเพียง 2^n

การจัดลำดับการผลิตโดยวิธีนี้ จะทำการแบ่งงานในระบบออกเป็น 2 กลุ่มคือ งานที่อยู่ใน J และงานที่อยู่ใน j' และให้ q_j และเป็นเวลาที่งาน J ตัวแรกจะเริ่มได้ ดังรูปที่ 2.3



รูปที่ 2.3 โครงสร้างของการจัดงานด้วยวิธี Dynamic Programming

วัตถุประสงค์ในการจัดลำดับการผลิต โดยวิธี Dynamic Programming นี้คือ ต้องการจัดงานในกลุ่มของงาน J ให้เป็นแบบ Optimum Solution

ค. วิธีการแตกกิ่ง (BRANCH AND BOUND)

วิธี Branch and Bound เป็นเทคนิคอย่างหนึ่งที่ใช้ในการแก้ปัญหาในการจัดลำดับ คือ การตัดส่วนที่ไม่มีโอกาสที่จะเป็นคำตอบออกจากการพิจารณา

วิธีการนี้ประกอบด้วย 2 ขั้นตอน คือ Branching เป็นกระบวนการแบ่งส่วนของปัญหาที่มี ขนาดใหญ่ออกเป็นปัญหาย่อย มากกว่า 2 ปัญหาย่อยขึ้นไป และ Bounding เป็นกระบวนการของการคำนวณ Lower Bound ที่ดีที่สุดของปัญหาย่อยนั้น ประสิทธิผลจะขึ้นอยู่กับ Lower Bound ที่ดี ซึ่งจะทำให้ผลที่ได้จะดีที่สุด

ง. วิธีการทางฮิวริสติก (HEURISTICS METHOD)

วิธีการทางฮิวริสติกเป็นการนำกฎต่างๆ มาใช้ในการหาผลลัพธ์ที่น่าพอใจของปัญหา ซึ่งวิธีที่ทำให้ผลลัพธ์เป็นที่น่าพอใจนั้น ไม่สามารถรับรองได้ว่าเป็นผลลัพธ์ที่ดีที่สุด ซึ่งวิธีการนี้ สามารถหาผลลัพธ์ของปัญหาที่มีขนาดใหญ่โดยไม่ต้องคำนวณมากนัก

วิธีการทางฮิวริสติกสามารถแก้ปัญหาได้ทั้งปัญหาขนาดเล็ก ขนาดกลาง และขนาดใหญ่ ซึ่งอาจนำคอมพิวเตอร์เข้ามาช่วยในการคำนวณ และเป็นเทคนิคที่เหมาะสมที่สุดที่จะใช้ในโรงงาน เพราะเป็นวิธีที่เชื่อถือได้ว่าจะให้ผลลัพธ์ที่ดีที่สุด เหมาะสมที่สุดที่เป็นไปได้ ไม่ว่าปัญหาที่ต้องการแก้ไขจะมีขนาดใหญ่เท่าใด

2.2.2 กฎต่างๆที่เป็น ฮิวริสติก (Heuristics)

2.2.2.1 PRIORITY DISPATCHING RULES

เป็นการใช้หลักลำดับความสำคัญเป็นเกณฑ์ในการจัดตารางการผลิตอันได้แก่

ก. ทำงานที่ใช้เวลาน้อยที่สุดก่อน (Shortest Processing Time, SPT) คืองานใดที่ใช้เวลาการทำงานน้อยที่สุด จะได้รับการจัดเข้าเป็นอันดับแรก และจะจัดงานเรียงลำดับจากงานที่มีเวลาการทำงานน้อยไปมาก

ข. EDD (Early Due Date) เลือกงานที่กำหนดส่งมอบเร็วสุดมาทำก่อน

ค. First Arrival at the Shop First Served (FASFS) เป็นการถือเอาเวลาในการเข้ามาของงานใน Shop มาเป็นเวลาในการจัดตาราง กรณีที่ Lead Time ของการผลิตนาน จะใช้วิธีนี้

ง. รับก่อนทำก่อน(First Come – First Serve, FCFS) หรือ มาก่อนทำก่อน (First In First Out, FIFO) กล่าวคือ งานที่เข้ามาที่หน่วยงานหรือเครื่องจักร จะเข้าแถวคอยรับบริการตามลำดับก่อนหรือหลังของการมาถึงของงาน

จ. MWKR (Most Work Remaining) การเลือกการทำงาน ที่มีปริมาณงานที่ยังไม่เสร็จมากที่สุดก่อน

ฉ. LWKR (Least Work Remaining) การเลือกการทำงาน ที่มีปริมาณงานที่ยังไม่เสร็จน้อยที่สุดก่อน

ช. MOPNR (Most Operations Remaining) การเลือกงาน ที่มีจำนวนขั้นตอนของงานที่ยังไม่เสร็จมากที่สุดก่อน

ซ. AWINQ (Anticipated Work in Next Queue) เป็นการให้ความสำคัญของงานโดยขึ้นอยู่กับงานในขั้นตอนการทำงานถัดไป

ฌ. FOFO (First Off First On) เป็นการให้ลำดับของสำคัญกับงานที่กำลังจะเสร็จมากที่สุดก่อน เครื่องจักรจะรอจนกว่าวานนั้นจะมาถึงใน Queue

ฏ. TWORK (Total Work) เป็นการนำงานที่มีเวลาปฏิบัติงานรวมทั้งหมดน้อยที่สุด มาทำก่อน

ฐ. MST (Minimum Slack Time) เป็นการเลือกงาน ที่มีเวลาน้อย (เวลาหลังทำงานเสร็จจนถึงกำหนดส่งงาน) น้อยที่สุดมาทำก่อน

ฑ. RANDOM (Random) เป็นการเลือกงานแบบสุ่ม

จากกฎการจัดลำดับงานข้างต้นอาจแบ่งได้ คือ กฎการจัดลำดับงานโดยขึ้นอยู่กับข้อมูลของงานที่อยู่ใน Queue ของเครื่องจักรนั้นๆ หรือ ขึ้นอยู่กับข้อมูลทั้งระบบ Local Priority เป็นการกำหนดลำดับงานโดยดูจากข้อมูลของงานที่อยู่ใน Queue ของเครื่องจักรนั้นๆ เท่านั้น อันได้แก่ SPT, LWKR, MWKR และGlobal Priority เป็นการกำหนดลำดับงานโดยดูจากข้อมูลของงานจากเครื่องจักรตัวอื่นๆเพิ่มเติมจากข้อมูลของงานใน Queue นั้นๆ อันได้แก่ AWINQ, FOFO

กฎการจัดลำดับงานสัมพันธ์กับการเปลี่ยนแปลงตามเวลา Static Priority การกำหนดลำดับงานไม่เปลี่ยนแปลงตามเวลา อันได้แก่ FASFS, TWORK, EDD และDynamic Priority การกำหนดลำดับงานจะเปลี่ยนแปลงตามเวลา อันได้แก่ FCFS, MST

2.2.2.2 SAMPLING PROCEDURES วิธีการนี้จะทำการแก้ปัญหาในการสร้าง Schedule อย่างสุ่ม จำนวนตัวอย่างจากการสุ่มที่มากกว่า จะได้คำตอบที่เข้าใกล้ความเหมาะสมมากกว่าจำนวนตัวอย่างน้อยกว่า

2.2.2.3 PROBABILITIES DISPATCHING PROCEDURES เป็นการนำความน่าจะเป็นมาใช้ในการหาผลลัพธ์ที่ดีที่สุด

2.2.3 ข้อจำกัดในการจัดตารางการผลิต (Constrain)

ข้อจำกัดในการจัดตารางการผลิต คือเงื่อนไขที่ต้องพิจารณาในการจัดตารางการผลิต ประกอบด้วย

2.2.3.1. ข้อจำกัดของทรัพยากร (Capacity Constrain)

ทรัพยากรสามารถทำงานได้ถึงค่าจำกัดค่าหนึ่งเท่านั้น โดยทั่วไปในการผลิตจะมีทรัพยากรบางอย่างที่สามารถทดแทนกันได้ ดังนั้นในการจัดตารางการผลิตหากมีทรัพยากรบางตัวไม่ว่าง สามารถนำทรัพยากรตัวอื่นที่ว่างอยู่และมีคุณสมบัติสามารถทดแทนกันได้มาทำงานแทน จะทำให้การจัดตารางการผลิตมีประสิทธิภาพมากขึ้น

2.2.3.2. ลำดับการดำเนินงาน (Precedence Constrain)

งานแต่ละงานนั้น มีลำดับของขั้นตอนการทำงานอยู่ ดังนั้นในการจัดตารางการผลิต การทำงานขั้นตอนแรกต้องถูกกระทำก่อนการทำงานถัดไป โดยไม่สามารถจัดข้ามขั้นตอนได้

2.2.4 การลดต้นทุนการผลิต

2.2.4.1 การลดต้นทุนการผลิตด้วยการวางแผนการผลิต

การลดต้นทุนการผลิตด้วยการวางแผนการผลิต ทำได้โดยการวางแผนการผลิตในส่วนที่เกี่ยวข้องกับงานประจำวันที่ทำอยู่ ได้แก่ งานจัดทำตารางการผลิต การวางแผนจัดลำดับการผลิต วางแผนด้านตารางเวลา วางแผนด้านวัสดุและการสั่งงาน เป็นต้น แผนการผลิตที่ให้ผลดีจะต้องพิจารณาร่วมกันทั้งการขาย การผลิต และการเก็บสต๊อก ซึ่งเกี่ยวข้องกับแผนการขาย รูปแบบการผลิต การวางผังเครื่องจักร เป็นต้น แผนการผลิตที่จัดทำขึ้นตะต้องมีประโยชน์สูงสุด ในขณะเดียวกันการทำงานระหว่างการผลิตจะต้องราบรื่นไม่มีการสูญเสียในด้านปริมาณการผลิต และเวลาส่งมอบงาน

2.2.4.2 การลดต้นทุนการผลิตด้วยการควบคุมการทำงานของอุปกรณ์เครื่องจักร

อุปกรณ์เครื่องจักรที่ใช้ในการผลิตนั้น จะต้องมีความสามารถในการทำงานได้อย่างเหมาะสม การเพิ่มอัตราการทำงานของอุปกรณ์เครื่องจักรให้สูงขึ้น สามารถทำได้ดังนี้

ก. การปรับปรุงแผนการผลิตให้มีการถอดเปลี่ยนเครื่องมือให้น้อยลง

ข. การลดหรือขจัดการชำรุดและการขัดข้องของอุปกรณ์เครื่องจักรให้ลดลงหรือหมดไป

ค. การปรับปรุงให้การใช้เวลาในการซ่อมบำรุงแต่ละครั้งลดน้อยลง

ง. การควบคุมคุณภาพการผลิตอย่างเข้มงวด เพื่อขจัดการเกิดของเสีย

จ. การปรับปรุงการบริหาร และควบคุมในสถานประกอบการ เพื่อลดเวลาการรองานในแต่ละขั้นตอนให้น้อยลง

2.3 การจัดการพลังงาน

คือ ความพยายามในการใช้พลังงานในจำนวนน้อยที่สุดเพื่อให้ได้ประโยชน์สูงสุดโดยไม่ทำให้กิจกรรมการผลิตต่ำลงและไม่ลดคุณภาพของผลิตภัณฑ์ การทำให้ต้นทุนการผลิตของผลิตภัณฑ์ในส่วน

ของพลังงานลดน้อยลง การใช้พลังงานตามความจำเป็น และในขณะเดียวกันก็ลดการสูญเสียที่ไม่จำเป็นต่างๆ เพื่อให้ประสิทธิภาพในการใช้พลังงานสูงขึ้น และการเลือกใช้พลังงานให้เหมาะสมทางเทคนิคและเศรษฐศาสตร์ และความต่อเนื่องในการจัดหา ซึ่งมีแนวทางในการพิจารณาจัดการพลังงานประกอบด้วย

- การเลือกใช้พลังงานที่เหมาะสม ว่าชนิดของพลังงานนั้นจะต้องพิจารณาจากคุณสมบัติทั้งทางด้านกายภาพและทางด้านเศรษฐกิจโดยพิจารณาในแง่ของประสิทธิภาพรวมที่จะได้ นอกจากนี้ยังอาจต้องพิจารณาถึงผลกระทบในระยะยาวอื่นๆ ด้วย
- การป้องกันการสูญเสียพลังงานและการใช้พลังงานอย่างมีประสิทธิภาพ พลังงานไฟฟ้านั้นมีที่ใช้งานต่างๆ อย่างกว้างขวาง เช่น ใช้ในการขับเคลื่อนมอเตอร์ ใช้ในการให้ความร้อน ให้แสงสว่าง และใช้ในงานควบคุม เป็นต้น การศึกษาสภาพการใช้งานและหาทางลดการสูญเสียในรูปต่างๆ เช่น การเดินเครื่องตัวเปล่าของมอเตอร์ ความร้อนรั่ว ลมรั่วหรือน้ำรั่ว นับว่าเป็นสิ่งจำเป็นอย่างยิ่งในการใช้พลังงานให้ประสิทธิภาพสูงขึ้น
- การใช้ประโยชน์พลังงานที่ยังไม่ได้ใช้ให้เป็นประโยชน์

โดยทั่วไปแนวทางการจัดการพลังงานจะประกอบด้วยขั้นตอนดังต่อไปนี้

ก. ค้นหาปริมาณการใช้และปริมาณสูญเสียของพลังงาน ขั้นตอนนี้เป็นขั้นตอนเริ่มแรกซึ่งสามารถแบ่งออกเป็นขั้นตอนย่อยๆ ได้ดังนี้

- ศึกษาชนิดและปริมาณพลังงานที่ใช้ระบบต่างๆ ของโรงงานอย่างละเอียดและพลังงานที่เข้าไปในระบบต่างๆ นั้น มีการกระจายการใช้ให้เกิดประโยชน์ หรือมีการสูญเสียมากน้อยเพียงใด
- สร้างและวิเคราะห์สมดุลพลังงานในแต่ละขั้นตอนผลิตอย่างละเอียดถี่ถ้วน ซึ่งสามารถเขียนเป็นสมการอัตราการไหลพลังงานเข้า-ออก ในแต่ละขั้นตอนการผลิต

ข. ดำเนินการจัดการพลังงานโดยวิธีการต่างๆ จากการศึกษาการใช้พลังงาน ตามข้อ ก. เป็นผลทำให้ทราบถึงรายละเอียดต่างๆ ซึ่งสามารถกำหนดวิธีการต่างๆ ในการจัดการพลังงานได้โดยจะต้องเสียค่าใช้จ่ายในการลงทุนเริ่มต้น

ค. ติดตามผลที่ได้จากการดำเนินการจัดการพลังงาน การติดตามผลนี้จะทำให้รู้ถึงส่วนเปลี่ยนแปลงของปริมาณพลังงานที่ใช้ และสามารถวางแผนระบบการบำรุงรักษาเครื่องมือเครื่องจักรต่างๆ ตลอดจนสามารถทราบถึงประสิทธิภาพของเครื่องมือเครื่องจักรนั้นๆ ว่าอยู่ในระดับใด

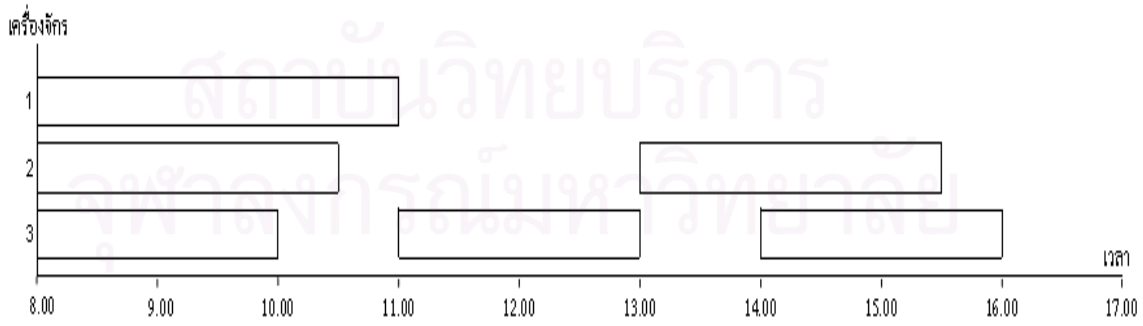
บทที่ 3

การพัฒนาวิธีการหาคำตอบ

ในบทนี้ผู้วิจัยได้นำเสนอการพัฒนาวิธีการหาคำตอบของปัญหา โดยผู้วิจัยจะนำเสนอแบบจำลองทางคณิตศาสตร์ของปัญหางานวิจัยก่อน ซึ่งการหาคำตอบที่เหมาะสมที่สุดจากแบบจำลองทางคณิตศาสตร์นั้นใช้เวลาค่อนข้างนานในการหาคำตอบ และสามารถประยุกต์ใช้ได้กับปัญหาขนาดเล็กเท่านั้น ผู้วิจัยจึงได้พัฒนาฮิวริสติกสำหรับแก้ปัญหาขึ้น ซึ่งสามารถหาคำตอบได้ใกล้เคียงกับคำตอบที่เหมาะสมที่สุดและใช้เวลาในการหาคำตอบรวดเร็วกว่าการหาคำตอบจากแบบจำลองทางคณิตศาสตร์

3.1 ลักษณะการทำงานของโรงงานกรณีศึกษาในส่วนการผสมผลิตภัณฑ์ในอาคารหลัก

งานที่จะทำการผลิตนั้นจะมีการจัดลงบนเครื่องจักรที่แต่ละงานนั้นมีการกำหนดไว้แล้วว่าต้องทำบนเครื่องจักรนี้ แล้วจากนั้นก็จะเริ่มทำการผลิตงานทั้งหมดนั้นเลยตั้งแต่เวลา 8.00 น. แล้วถ้าเครื่องจักรตัวนั้นมีการทำงาน 2 งานก็จะเริ่มทำงานอีกงานเวลา 13.00 น. แต่ถ้างานบนเครื่องจักรนั้นมี 3 งาน ก็จะมีการทำงานที่ 2 เวลา 11.00 น. และงานที่ 3 เวลา 14.00 น. ถ้าเครื่องจักรมีกำลังไฟฟ้าที่แตกต่างกันงานที่ทำงานก็จะมีเวลาการทำงานเท่ากัน และเมื่องานแต่ละงานได้ทำเสร็จแล้วก็จะมีเวลาทำความสะอาด 15 นาทีเสมอ ดังรูปที่ 3.1



รูปที่ 3.1 รูปแบบการจัดตารางการผลิตของโรงงาน

จากรูปที่ 3.1 จะเห็นได้ว่าช่วงเวลา 8.00 น. ถึง 10.00 น. มีการใช้กำลังไฟฟ้ามากที่สุด ซึ่งโดยทั่วไปแล้วการไฟฟ้าฯ จะคิดค่าความต้องการพลังงานไฟฟ้าที่เวลาเฉลี่ย 15 นาที ที่สูงสุดมาคิดค่าความต้องการพลังงานไฟฟ้า ซึ่งลักษณะของงานที่นำมาพิจารณาเป็นแบบตามงาน (Job Shop) คือแต่ละงานจะมีเส้นทางหรือสายการผลิตที่แตกต่างกัน ไม่จำเป็นต้องผ่านเครื่องจักรทุกตัวในโรงงาน หรือมีลำดับการผ่านเครื่องจักรเหมือนกัน

3.2 ตัวแบบเชิงคณิตศาสตร์ของปัญหางานวิจัย

ตัวแบบเชิงคณิตศาสตร์ของปัญหางานวิจัยนี้ ถูกจำลองให้อยู่ในรูปของการโปรแกรมเชิงจำนวนเต็มแบบผสม (Mixed Integer Programming) ดังนี้

เซต (Sets)

$J = \{1,2,\ldots,\hat{j}\}$ คือเซตของงาน โดยที่ $\hat{j}$ คือจำนวนงานที่ทำงาน

$M = \{1,2,\ldots,\hat{m}\}$ คือเซตของเครื่องจักร โดยที่ $\hat{m}$ คือจำนวนเครื่องจักรที่ทำงาน

$P = \{1,2,\ldots,\hat{p}\}$ คือช่วงเวลาการทำงาน โดยที่ $\hat{p}$ คือจำนวนช่วงเวลา

ตัวแปร (Vtariable)

$$a_{jm} = \begin{cases} 1 & \text{ถ้างาน j ทำงานบนเครื่องจักร m} \\ 0 & \text{อื่นๆ} \end{cases}$$

$$X_{jm}^{p} = \begin{cases} 1 & \text{ถ้างาน j ทำงานบนเครื่องจักร m เริ่มที่ช่วงเวลา p} \\ 0 & \text{อื่นๆ} \end{cases}$$

$$Y_{jm}^{p} = \begin{cases} 1 & \text{ถ้างาน j ทำงานบนเครื่องจักร m ที่ช่วงเวลา p} \\ 0 & \text{อื่นๆ} \end{cases}$$

$$Z_{jm}^{p} = \begin{cases} 1 & \text{ถ้างาน j ทำงานบนเครื่องจักร m มีการทำความสะอาดที่ช่วงเวลา p} \\ 0 & \text{อื่นๆ} \end{cases}$$

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

EC^p = กำลังไฟฟ้าของช่วงเวลา $p \in P$

$Peak$ = ค่ากำลังไฟฟ้าสูงสุด

พารามิเตอร์ (Parameters)

K_1 = ค่า ft = 0.4683 (ขณะทำงานวิจัย)

K_2 = ค่าความต้องการพลังงานไฟฟ้าสูงสุด = 132.93

d_j = เวลาการทำงานของงาน $j \in J$

EP_m = กำลังไฟฟ้าของเครื่องจักร $m \in M$

W_{jm} = $\begin{cases} 1 & \text{ถ้างาน j สามารถทำงานบนเครื่องจักร m ได้} \\ 0 & \text{อื่นๆ} \end{cases}$

แบบจำลอง (Model)

$$\text{Minimize Cost} = (K_1 * \sum_{j \in J} \sum_{m \in M} (d_j a_{jm} EP_m)) + (K_2 * \text{Peak}) \quad (1)$$

วัตถุประสงค์

$$\sum_{j \in J} (d_j + 1) \cdot (a_{jm}) \leq \hat{p}, \qquad \forall m \in M \quad (2)$$

$$a_{jm} \leq W_{jm}, \qquad \forall j \in J, \forall m \in M \quad (3)$$

$$\sum_{m \in M} a_{jm} = 1, \qquad \forall j \in J \quad (4)$$

$$\sum_{p \in \{1,\ldots,\hat{p}-d_j\}} X_{jm}^p = a_{jm}, \qquad \forall j \in J, \forall m \in M \quad (5)$$

$$\sum_{q \in \{p,\ldots,(p+d_j-1)\}} Y_{jm}^q \geq d_j \cdot X_{jm}^p, \qquad \forall j \in J, \forall m \in M, \forall p \in \{1,\ldots,(\hat{p}-d_j)\} \quad (6)$$

$$\sum_{q \notin \{p,\ldots,(p+d_j-1)\}} Y_{jm}^q \leq (\hat{p}-d_j).(1-X_{jm}^p), \qquad \forall j \in J, \forall m \in M, \forall p \in \{1,\ldots,(\hat{p}-d_j)\} \quad (7)$$

$$\sum_{j \in J} \sum_{m \in M} Y_{jm}^p \leq 1, \qquad \forall p \in P \quad (8)$$

$$Z_{jm}^{p+d_j} = X_{jm}^p, \qquad \forall j \in J, \forall m \in M, \forall p \in \{1,\ldots,(\hat{p}-d_j)\} \quad (9)$$

$$\sum_{j \in J} \sum_{m \in M} Y_{jm}^p \leq (1 - Z_{jm}^p) \cdot \hat{j}, \qquad \forall j \in J, \forall m \in M, \forall p \in \{(d_j+1),\ldots,\hat{p}\} \quad (10)$$

$$EC^{p} = \sum_{j\in J}\sum_{m\in M} EP_{m} \cdot Y_{jm}^{p}, \qquad \forall p \in P \tag{11}$$

$$Peak \geq EC^{p}, \qquad \forall p \in P \tag{12}$$

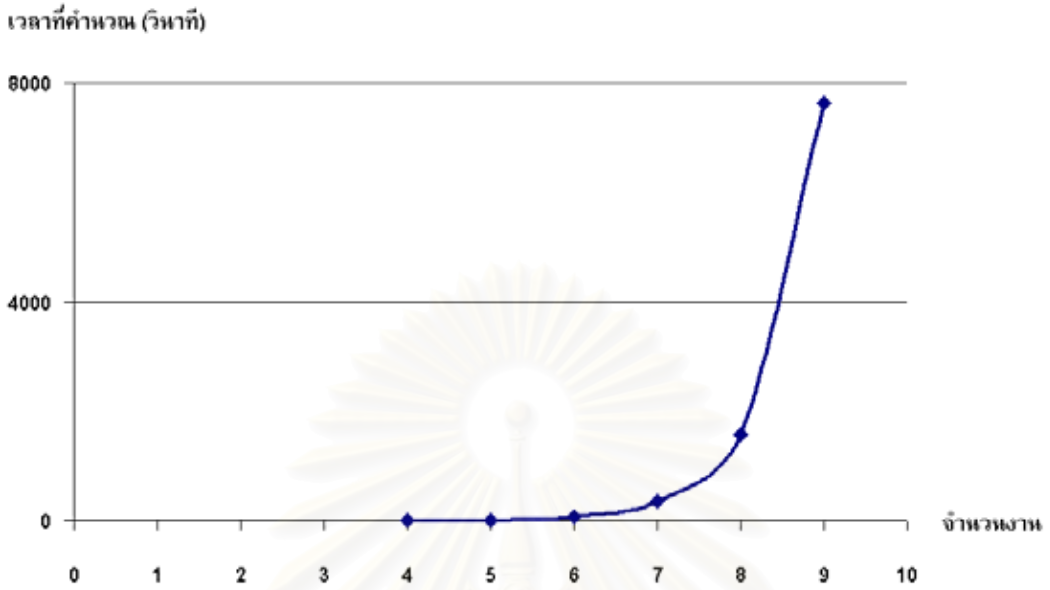
$$a_{jm},\ X_{jm}^{p},\ Y_{jm}^{p},\ Z_{jm}^{p},\ W_{jm} \in \{0,1\} \tag{13}$$

$$EC^{p}, Peak \quad \text{interger} \tag{14}$$

ฟังก์ชั่นวัตถุประสงค์ (1) มีวัตถุประสงค์เพื่อทำให้ค่าไฟฟ้าน้อยที่สุด เงื่อนไขบังคับ (2)-(12) ใช้สำหรับบังคับให้มีการจัดตารางการผลิตโดยที่เงื่อนไขบังคับ (2) ใช้สำหรับบังคับว่างานที่ทำบนเครื่องจักรใดๆ แล้วจะไม่เกินเวลาทำการผลิต เงื่อนไขบังคับ (3) ใช้สำหรับบังคับว่าถ้างานใดๆ สามารถทำงานบนเครื่องจักรใดๆ ได้นั้นงานนั้นสามารถที่จะทำงานหรือไม่ทำงานบนเครื่องจักรนั้นก็ได้ เงื่อนไขบังคับ (4) ใช้สำหรับบังคับว่างานสามารถทำงานใดๆ ได้บนเครื่องจักรเครื่องเดียว เงื่อนไขบังคับ (5) ใช้สำหรับบังคับให้ว่าเมื่อมีการเลือกเครื่องจักรที่จะทำแล้วต้องมีการเริ่มต้นการทำงานเกิดขึ้น เงื่อนไขบังคับ (6) ใช้สำหรับบังคับว่าเมื่องานเริ่มมีการทำงานแล้วต้องมีเวลาในการปฏิบัติงานตามมาด้วย เงื่อนไขบังคับ (7) ใช้สำหรับบังคับว่าก่อนที่จะมีการเริ่มการทำงานนั้นต้องไม่มีการทำงานที่ทำอยู่ก่อนแล้ว เงื่อนไขบังคับ (8) ใช้สำหรับบังคับว่าเมื่อมีการทำงานเกิดขึ้นแล้วจะเท่ากับ 1 หรือไม่มีการทำงานเกิดขึ้น เงื่อนไขบังคับ (9) ใช้สำหรับบังคับว่าเมื่อมีการเริ่มต้นทำงานแล้วนั้นจะต้องมีการทำความสะอาดเครื่องด้วยเช่นกัน เงื่อนไขบังคับ (10) ใช้สำหรับบังคับว่าเมื่อมีการทำความสะอาดแล้วนั้นจะไม่มีการทำงานของเครื่องจักร เงื่อนไขบังคับ (11) ใช้สำหรับบังคับว่าเมื่อมีการทำงานแล้วจะต้องมีการใช้พลังงานไฟฟ้าแล้วทำให้เกิดค่าพลังงานไฟฟ้าขึ้นมาด้วย เงื่อนไขบังคับ (12) ใช้สำหรับบังคับว่าค่ากำลังไฟฟ้าสูงสุดนั้นเกิดจากค่าพลังงานไฟฟ้า

เนื่องจากที่ใช้ตัวแบบเชิงคณิตศาสตร์นี้ก็เพราะว่าสามารถทำให้ค่าไฟฟ้ามีค่าลดลงได้ โดยที่เครื่องจักรที่มีกำลังไฟฟ้าต่างกันแต่เวลาการทำงานบนเครื่องจักรจะมีเวลาการทำงานที่เท่ากัน แต่จะใช้เวลาในการหาคำตอบมีแนวโน้มมากขึ้นเรื่อยๆ ตามจำนวนงานที่เพิ่มขึ้นดังรูปที่ 3.2

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 3.2 แนวโน้มเวลาในการหาคำตอบของตัวแบบเชิงคณิตศาสตร์

จากรูปที่ 3.2 ทำให้ทราบว่าเมื่อมีจำนวนงานในการผลิตเพิ่มขึ้นเวลาในการหาคำตอบจากตัวแบบเชิงคณิตศาสตร์มีเวลาเพิ่มขึ้นตามด้วย จึงทำให้เมื่อจะทำการผลิตที่มีงานจำนวนมากเวลาในการหาคำตอบก็จะนาน ดังนั้นฮิวริสติกจึงถูกนำมาใช้สำหรับแก้ปัญหาที่มีขนาดใหญ่

3.3 ฮิวริสติกสำหรับแก้ปัญหา

การแก้ปัญหาผ่านแบบจำลองทางคณิตศาสตร์นั้น ใช้เวลานานจึงสามารถนำประยุกต์ใช้ได้สำหรับปัญหาที่มีขนาดเล็กที่มีขนาดจำนวนงาน 5งาน ดังนั้นฮิวริสติกจึงถูกนำมาใช้สำหรับแก้ปัญหาที่มีขนาดใหญ่ที่มีขนาดจำนวนงาน 10 งาน 15 งาน 20 งาน และ 30 งาน ดังที่แสดงในบทที่ 4 ซึ่งใช้เวลาคำนวณน้อยกว่ามาก ฮิวริสติกที่ผู้วิจัยนำเสนอสำหรับแก้ปัญหามีรายละเอียดดังนี้

3.3.1 ตัวแทนคำตอบของปัญหา

คำตอบของปัญหาวิจัยคือการจัดเรียงลำดับการทำงาน ซึ่งจะถูกแทนด้วยอาร์เรย์ 2 มิติ ระหว่างช่วงเวลากับเครื่องจักร ระบุถึงลำดับของการทำงานโดยเครื่องจักรแต่ละเครื่อง ลักษณะของคำตอบเป็นดังนี้

Schedule[*m*][*p*] = *J*

โดยที่ m คือหมายเลขเครื่องจักร $m = 1, 2, \ldots, \hat{m}$

p คือช่วงเวลา $p = 1, 2, \ldots, \hat{p}$

J คืองาน

$J = 0$ หมายถึงไม่มีการทำงาน

$J = x$ หมายถึงงาน x มีการทำงาน

ตัวอย่างคำตอบของปัญหาเป็นดังตารางที่ 3.1

ตารางที่ 3.1 อาร์เรย์ 2 มิติ แสดงตัวอย่างคำตอบของปัญหา

m \ *p*	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	1	1	0	0
2	2	2	2	2	0	0

จากรูปที่ 3.2 แสดงลักษณะคำตอบของปัญหา โดยเครื่องจักรที่ 1 เริ่มต้นทำงานที่ช่วงเวลาที่ 2 แล้วทำงานไปถึงช่วงเวลาที่ 4 แล้วหลังจากทำงานเสร็จแล้วทุกครั้งก็จะต้องมีการทำความสะอาดของเครื่องจักรก่อนที่จะสามารถทำงานต่อไปได้ซึ่งในเครื่องจักรที่ 1 นั้นก็จะอยู่ในช่วงเวลาที่ 5 และในช่วงเวลาที่ 1 กับช่วงเวลาที่ 6 ไม่มีการทำงานเกิดขึ้น สำหรับเครื่องจักรที่ 2 เริ่มต้นทำงานที่ช่วงเวลาที่ 1 แล้วทำงานไปถึงช่วงเวลาที่ 4 แล้วหลังจากนั้นก็เป็นการทำความสะอาดในช่วงเวลาที่ 5 และในช่วงเวลาที่ 6 ไม่มีการทำงานเกิดขึ้น

3.3.2 ฮิวริสติก

ฮิวริสติกที่นำเสนอนั้น เป็นฮิวริสติกที่ทำการจัดงานใดๆ ลงบนเครื่องจักรใดๆ ที่เวลาใดแล้วทำให้ค่ากำลังไฟฟ้าสูงสุดมีค่าต่ำที่สุดเพื่อเป็นการลดต้นทุนการผลิต โดยที่เวลาที่งานแต่ละงานที่ทำการผลิตนั้นต้องมีการซ้อนทับกันน้อยที่สุด ซึ่งมีรายละเอียดดังรูปที่ 3.3

```
Heuristic Minimize ElectricCost(H) {
  Assign( J,M )
  Sort(H,M)
  M′ ← InitialSchedule(J,M)
    While M′ ≠ ∅ {
        m ← Pop(M′)
        foreach(j in J_m,) Schedule(j)
        M′ ← M′-{m}
    }
    S ← Solution(J,M)
    Improve(S)
}
```

รูปที่ 3.3 ฮิวริสติกสำหรับการแก้ปัญหา

นำงานที่จะทำการผลิตทั้งหมดมาดูว่างานเหล่านั้นจะทำการผลิตที่เครื่องจักรใดใน M โดยงาน J ที่จะทำการผลิตนั้นจะเลือกเครื่องจักรมาทำการผลิตตามฟังก์ชั่น Assign(J,M) แล้วหลังจากงานที่แต่ละงานได้ทำการเลือกเครื่องจักรได้แล้วนั้นก็จะนำเครื่องจักรที่ได้รับการเลือกมาเรียงลำดับตามกำลังไฟฟ้าของแต่ละเครื่องตามฟังก์ชั่น Sort(H,M) แล้วทำจัดงานเบื้องต้นตามฟังก์ชั่น InitialSchedule(J,M) ซึ่งถ้าจัดงานเบื้องต้นไปแล้วยังมีเครื่องจักรที่ยังไม่ได้ทำการผลิตอยู่อีกก็นำเครื่องจักรที่ยังไม่ได้มีการทำงานมาทำงาน โดยนำเครื่องจักรที่ได้เรียงไว้ตามกำลังไฟฟ้าตามฟังก์ชั่น Pop(M') มาเริ่มต้นทำการหาช่วงเวลาที่จะทำการผลิต ซึ่งงานที่ทำอยู่ในเครื่องจักรแต่ละเครื่องนั้น (J_m) จะเริ่มทำงานที่ช่วงเวลาใดนั้นตามฟังก์ชั่น Foreach($j \ in \ J_m$,) โดยการจัดให้งานทำงานได้ตามฟังก์ชั่น Schedule(j)

จากนั้นจะเข้าสู่กระบวนการหาคำตอบโดยทำการวนซ้ำโดยที่จะหยุดก็ต่อเมื่อไม่มีเครื่องจักรที่จะทำการผลิตแล้วก็จะทำการเก็บคำตอบตามฟังก์ชั่น Solution(J,M) หลังจากนั้นก็มีการปรับปรุงคำตอบตามฟังก์ชั่น Improve(S)

สำหรับรายละเอียดแต่ละส่วนของฮิวริสติก มีรายละเอียดดังนี้

3.3.2.1 ฟังก์ชั่น Assign(J,M)

ฟังก์ชั่นนี้จะเป็นฟังก์ชั่นที่งานแต่ละงานจะเลือกเครื่องจักรที่จะทำงานด้วยว่าแต่ละงานสามารถทำการผลิตได้ตามตัวแบบเชิงคณิตศาสตร์ดังนี้

$$\text{Minimize TotalElectricPower} = \sum_{m \in M} d_j a_{jm} EP_m \quad (1)$$

วัตถุประสงค์

$$\sum_{j \in J} \left(d_j + 1\right) \cdot \left(a_{jm}\right) \leq \hat{p}, \quad \forall m \in M \quad (2)$$

$$a_{jm} \leq W_{jm}, \quad \forall j \in J, \forall m \in M \quad (3)$$

$$\sum_{m \in M} a_{jm} = 1, \quad \forall j \in J \quad (4)$$

ฟังก์ชั่นวัตถุประสงค์ (1) มีวัตถุประสงค์เพื่อทำให้ค่ากำลังไฟฟ้ารวมมีค่าต่ำที่สุด เงื่อนไขบังคับ (2)-(4) ใช้สำหรับบังคับให้มีการจัดตารางการผลิตโดยที่เงื่อนไขบังคับ (2) ใช้สำหรับบังคับว่างานที่ทำบนเครื่องจักรใดๆ แล้วจะไม่เกินเวลาทำการผลิต เงื่อนไขบังคับ (3) ใช้สำหรับบังคับว่าถ้างานใดๆ สามารถทำงานบนเครื่องจักรใดๆ ได้นั้นงานนั้นสามารถที่จะทำงานหรือไม่ทำงานบนเครื่องจักรนั้นก็ได้ เงื่อนไขบังคับ (4) ใช้สำหรับบังคับว่างานสามารถทำงานใดๆ ได้บนเครื่องจักรเครื่องเดียว

โดยถ้างานได้เครื่องจักรที่ทำงานแล้วตามวัตถุประสงค์นี้ก็จะทำให้ค่าพลังงานไฟฟ้ามีค่าน้อยที่สุด ซึ่งส่วนของค่าพลังงานไฟฟ้านี้เป็นส่วนหนึ่งของค่าไฟฟ้าจึงทำให้สามารถลดค่าไฟฟ้าได้ถ้าค่าพลังงานไฟฟ้ามีค่าน้อยที่สุด และจะทำให้แน่ใจได้ว่าการหาคำตอบในขั้นตอนต่อไปมีความเป็นไปได้

3.3.2.2 ฟังก์ชั่น Sort(*H,M*)

ฟังก์ชั่นนี้เป็นฟังก์ชั่นที่นำเครื่องจักรที่ได้เลือกว่าจะทำการผลิตมาจัดลำดับกำลังไฟฟ้าของเครื่องจักรที่จะนำมาหาเวลาการผลิตของงานโดยมีด้วยกัน 3 รูปแบบ คือ

3.3.2.2.1 แบบที่ 1 การนำเอาเครื่องจักรที่มีกำลังไฟฟ้ามากที่สุดมาเริ่มต้นหาเวลาการผลิตของงานก่อนแล้วตามด้วยเครื่องจักรที่มีกำลังไฟฟ้าน้อยกว่า

3.3.2.2.2 แบบที่ 2 การนำเอาเครื่องจักรที่มีกำลังไฟฟ้าน้อยที่สุดมาเริ่มต้นหาเวลาการผลิตของงานก่อนแล้วตามด้วยเครื่องจักรที่มีกำลังไฟฟ้ามากกว่า

3.3.2.2.3 แบบที่ 3 การนำเอาเครื่องจักรที่มีกำลังไฟฟ้าแบบสุ่มมาเริ่มต้นหาเวลาการผลิตของงาน

เนื่องจากที่ทำการจัดตารางการผลิตเป็น 3 รูปแบบ เพื่อดูว่ากำลังไฟฟ้าของเครื่องจักรมีผลต่อการทำให้ค่าพลังงานไฟฟ้าสูงสุดมีค่าน้อยหรือเปล่า

3.3.2.3 ฟังก์ชั่น InitialSchedule(*J,M*)

การจัดงานลงเครื่องจักรเบื้องต้นนั้นเป็นการทำให้ไม่ต้องเสียเวลาในการที่จะดูว่างานแต่ละงานจะทำการผลิตที่เวลาใด เพราะงานที่ได้ทำการจัดตามนี้นั้นสามารถเริ่มเวลาการผลิตได้ที่เวลาเริ่มต้นทำการผลิตในแต่ละครั้งได้เลย ซึ่งขั้นตอนในการจัดนั้นเป็นไปตามรูปที่ 3.4

Procedure InitialSchedule(*J,M*)
Begin
$L \leftarrow \min_{j \in J} \{d_j\}$
$M' = M$
For each $m \in M$
RM(m) $\leftarrow$ Calculate_RemainingTime(m)
If RM(m) < L **then**
Scheduling (j,m,b)
$M' \leftarrow M' - \{m\}$
End if
End for
Return M'
End

รูปที่ 3.4 การจัดงานลงเครื่องจักรเบื้องต้น

งานที่จะเริ่มต้นทำการผลิตตามรูปที่ 3.4 ได้เลยนั้น เนื่องจากว่าจะไม่มีงานใดๆ ที่จะทำการผลิตก่อนหรือหลังจากงานที่ได้ทำการจัดลงเครื่องจักรเบื้องต้นแล้วจะทำให้ไม่เกิดการซ้อนทับของงานแต่ละงานเกิดขึ้น จึงทำให้งานที่จัดไปเบื้องต้นนั้นไม่ว่าจะเริ่มทำการผลิตที่เวลาใดก็ตามก็จะมีผลเหมือนกันกับที่งานเริ่มต้นทำการผลิตเมื่อเริ่มเวลาการผลิตก็คือเกิดการซ้อนทับของงาน และลำดับการผลิตของงานที่อยู่บน

เครื่องจักรเดียวกันนั้นจะเอางานใดมาเริ่มทำงานก่อนก็ได้ซึ่งจะไม่มีผลต่อค่ากำลังไฟฟ้าเนื่องจากว่างานแต่ละงานที่อยู่บนเครื่องจักรเดียวกันใช้กำลังไฟฟ้าที่มีค่าเท่ากัน

จากรูปที่ 3.4 เป็นการจัดงานลงเครื่องจักรเบื้องต้น โดยการนำเครื่องจักรที่ได้ทำการจัดเรียงตามกำลังไฟฟ้าแล้วมาทำการดูว่างานบนเครื่องจักรเหล่านั้นเมื่อรวมกันแล้วมีช่วงเวลาที่เหลือที่ไม่ได้ทำงานมีค่าเป็น RM(m) น้อยกว่า $\min_{j\in J}\{d_j\}$ หรือเปล่า ซึ่งค่าของ RM(m) มีค่าตามสมการที่ 3.1

$$\hat{p} - \sum_{j\in J_m}(d_j + 1) \;=\; RM(m) \qquad ,\forall m \in M \tag{3.1}$$

งานที่มีค่าของ RM(m) น้อยกว่า $\min_{j\in J}\{d_j\}$ นั้นสามารถนำงานมาทำการผลิตบนเครื่องจักรนั้นโดยเริ่มทำงานได้เมื่อเริ่มเวลาทำการผลิตได้โดยไม่ต้องคำนึงถึงว่าค่ากำลังไฟฟ้าจะเป็นเท่าไร

3.3.2.4 ฟังก์ชั่น Pop(M')

ฟังก์ชั่นนี้เป็นฟังก์ชั่นที่นำเอาเครื่องจักรที่ได้ทำการเรียงตามแบบต่างๆ มาทำดูว่างานที่ทำบนเครื่องจักรตัวที่เลือกมาทำนั้นสามารถเริ่มทำการผลิตได้ที่ช่วงเวลาใดโดยช่วงเวลาสุดท้ายที่งานสามารถจะทำการผลิตได้นั้นมีค่าเป็น RT ซึ่งค่าของ RT มีค่าตามสมการที่ 3.2

$$\hat{p} - \sum_{j\in J_m}(d_j + 1) - 1 \;=\; RT \qquad ,\forall m \in M \tag{3.2}$$

3.3.2.5 ฟังก์ชั่น Foreach(j *in* J_m,)

ฟังก์ชั่นนี้เป็นฟังก์ชั่นที่แสดงรายละเอียดในการดูว่างานที่จะทำการผลิตในแต่ละเครื่องจักรนั้น (J_m) ควรจะเริ่มทำการผลิตที่เวลาใดตามรูปที่ 3.5

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
Procedure StartPeriod(j in J_m,)
  Begin
     For each j in J_m
        A_j ← CalculateStartingPeriod(j)
        B_j ← LowestElectricPower(A_j)
        C_j ← MaxConsecutiveEmpty(B_j)
        N_l ← Count(B_j)
        N_e ← Count(C_j)
        If (N_l = 1) then
        b ← Extract (B_j)
        Scheduling (j,m,b)
        Else if N_l > 1 then
           if N_e = 1 then
              b ← Extract(C_j)
              Scheduling (j,m,b)
           Else if N_e > 1 then
              b ← Extract(C_j)
              Scheduling (j,m,b)
           End if
        End if
     End for
  End
```

รูปที่ 3.5 ขั้นตอนในการหาเวลาเริ่มต้นทำการผลิต

ฟังก์ชั่น CalculateStartingPeriod(j) เป็นฟังก์ชั่นการคำนวณหาว่าช่วงเวลาใดสามารถเป็นช่วงเวลาที่งานเริ่มต้นทำการผลิตบนเครื่องจักรได้โดยที่ช่วงเวลาที่เริ่มต้นการผลิตนั้นต้องอยู่ในช่วงเวลาที่ทำการผลิตได้หลังจากนั้นก็จะมาทำการดูว่าช่วงเวลาที่งานสามารถเริ่มต้นทำการผลิตได้นั้นเป็นช่วงเวลาที่ทำให้ค่ากำลังไฟฟ้ามีค่าต่ำที่สุดด้วยหรือเปล่าตามฟังก์ชั่น LowestElectricPower(A_j) จากนั้นก็ทำการดูในช่วงเวลาใดที่เริ่มต้นทำการผลิตที่ทำให้ค่ากำลังไฟฟ้าต่ำที่สุดแล้วมีช่วงเวลาที่เหลือที่ไม่ได้ทำงานติดต่อกันมากที่สุดด้วยนั้นเป็นไปตามฟังก์ชั่น MaxConsecutiveEmpty(B_j) จากที่ได้ทำการดูไปแล้วว่ามีช่วงเวลาที่เริ่มต้นทำการผลิตที่ทำให้มีค่ากำลังไฟฟ้ามีค่าต่ำที่สุด(B_j) ก็จะต้องทำการนับว่ามีทั้งหมดกี่ช่วงเวลาเป็นไปตามฟังก์ชั่น Count (B_j) และทำการดูว่าช่วงเวลาที่มีการเริ่มต้นทำการผลิตที่ทำให้มีค่ากำลังไฟฟ้าต่ำที่สุดแล้วมีช่วงเวลาที่

เหลือไม่ได้ทำงานติดต่อกันมากที่สุด (C_j) มาทำการนับจำนวนว่ามีทั้งหมดกี่ช่วงเวลา เป็นไปตามฟังก์ชั่น Count(C_j)

จากที่ได้มีการนับจำนวนช่วงเวลาที่เริ่มต้นทำการผลิตที่ทำให้มีค่ากำลังไฟฟ้าต่ำที่สุด(N_l)ถ้าจำนวนช่วงเวลานั้นมีค่าเพียงค่าเดียวก็จะเป็นไปตามฟังก์ชั่น Extract(B_j) ทำการจัดให้งานมีการทำงานบนเครื่องจักรที่ช่วงเวลา b ตามฟังก์ชั่น Scheduling (j,m,b) ซึ่งถ้าจำนวนของช่วงเวลา N_l มีค่ามากกว่าหนึ่งค่าแล้วให้ดูว่าจำนวนช่วงเวลาที่เริ่มต้นทำการผลิตที่ทำให้มีค่ากำลังไฟฟ้าต่ำที่สุดแล้วมีช่วงเวลาที่เหลือไม่ได้ทำงานติดต่อกันมากที่สุด(N_e) ถ้าจำนวนช่วงเวลาที่เริ่มต้นทำการผลิตเพียงค่าเดียวก็จะนำเอาช่วงเวลานั้นเป็นช่วงเวลาที่งานเริ่มต้นทำการผลิตเป็นไปตามฟังก์ชั่น Extract(C_j) ก็จะทำการจัดให้งานมีการทำงานบนเครื่องจักรที่ช่วงเวลา b ตามฟังก์ชั่น Scheduling (j,m,b) แต่ถ้าจำนวนช่วงเวลา N_e มีค่ามากกว่าหนึ่งค่าแล้วให้ทำการดูว่าช่วงเวลาที่เริ่มต้นทำการผลิตที่ทำให้มีค่ากำลังไฟฟ้าต่ำที่สุดแล้วมีช่วงเวลาที่เหลือไม่ได้ทำงานติดต่อกันมากที่สุดมาทำการดูว่ามีผลรวมของกำลังไฟฟ้าในแต่ละช่วงเวลาที่งานใดๆ มีการทำการผลิตขึ้นว่ามีค่าน้อยที่สุดซึ่งเป็นไปตามฟังก์ชั่น Extract(C_j) ก็จะทำการจัดให้งานมีการทำงานบนเครื่องจักรที่ช่วงเวลา b ตามฟังก์ชั่น Scheduling (j,m,b)ได้ แต่ถ้ามีผลรวมของกำลังไฟฟ้าในแต่ละช่วงเวลาที่งานใดๆ เท่ากันก็จะเลือกที่จะเริ่มต้นทำการผลิตที่ช่วงเวลาที่เร็วที่สุดในการทำการผลิตงาน

3.3.2.6 ฟังก์ชั่น Solution(J,M)

ฟังก์ชั่นนี้เป็นฟังก์ชั่นการเก็บคำตอบที่ได้จากการทำตามฟังก์ชั่นต่างๆ ไว้เป็นคำตอบ S

3.3.2.7 ฟังก์ชั่น Improve(S)

ฟังก์ชั่นนี้เป็นฟังก์ชั่นที่ปรับปรุงคำตอบโดยทำการสลับงานที่อยู่ติดกันและดูว่าค่ากำลังไฟฟ้าสูงสุดเกิดที่ช่วงเวลาใดแล้วทำการนำงานนั้นไปเริ่มต้นการผลิตที่ช่วงเวลาใหม่โดยจะทำการสร้างคำตอบข้างเคียงจากคำตอบปัจจุบัน S ซึ่งคำตอบข้างเคียงที่สร้างขึ้นมาทั้งหมดนั้นต้องเป็นคำตอบที่เป็นไปได้ (Feasible Solution) จากนั้นจะทำการเลือกคำตอบที่ดีที่สุดจากคำตอบข้างเคียงที่สร้างขึ้นมาทั้งหมด ขึ้นมาเป็นคำตอบปัจจุบันตัวต่อไป

ฟังก์ชั่นต่างๆ สามารถอธิบายได้ชัดเจนยิ่งขึ้นดังตัวอย่างที่ 2

ตัวอย่างที่ 2

มีงานทั้งหมด 10 งาน ช่วงเวลาการทำงานทั้งหมด 36 ช่วงเวลา ซึ่ง 1 ชั่วโมงแบ่งเป็น 4 ช่วงเวลา โดยการทำงานเริ่มต้นที่เวลา 8.00 น.ถึง 17.00 น. งานแต่ละงานเป็นอิสระต่อกัน รายละเอียดของงานแต่ละงานแสดงดังตารางที่ 3.2

ตารางที่ 3.2 รายละเอียดของงานทั้งหมด

งาน	เครื่องจักรที่สามารถทำงานได้	กำลังไฟฟ้า (kW)	ช่วงเวลาที่ทำการผลิต
1	5	49.02	16
2	2	41.67	14
	10	45.10	
3	7	1.98	16
4	3	25.50	16
	4	27.46	
	9	27.46	
5	6	31.87	12
6	3	25.50	8
	4	27.46	
	9	27.46	
7	6	31.87	14
8	2	41.67	32
	10	45.10	
9	7	1.98	8
10	1	7.37	8

1. เลือกเครื่องจักรที่งานจะทำงานได้โดยการเลือกเครื่องจักรตามฟังก์ชั่น Assign (*J,M*) ดังตารางที่ 3.3 แล้วจะทำให้มีค่าพลังงานไฟฟ้ารวม 1089.5 kWh

ตารางที่ 3.3 การเลือกเครื่องจักรในการทำงาน

งาน	เครื่องจักรที่สามารถทำงานได้	เครื่องจักรที่เลือกทำงาน
1	5	5
2	2	2
	10	
3	7	7
4	3	3
	4	
	9	
5	6	6
6	3	3
	4	
	9	
7	6	6
8	2	10
	10	
9	7	7
10	1	1

2. นำงานทั้งหมดมาเรียงกำลังไฟฟ้าของเครื่องจักรจากกำลังไฟฟ้ามากกว่าไปหากำลังไฟฟ้าน้อยกว่าซึ่งเป็นไปตามฟังก์ชั่น Sort(*H*,*M*) ตามตารางที่ 3.4

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 3.4 การเรียงกำลังไฟฟ้าของเครื่องจักร

งาน	เครื่องจักร	กำลังไฟฟ้า (kW)
1	5	49.02
8	10	45.10
2	2	41.67
5	6	31.87
7	6	31.87
4	3	25.50
6	3	25.50
10	1	7.37
3	7	1.98
9	7	1.98

3. เริ่มทำการจัดงานเบื้องต้นตามฟังก์ชั่น InitialSchedule(*J,M*)โดยทำการคำนวณหาช่วงเวลาที่เหลือจากการที่งานได้ทำการผลิตไปแล้วซึ่งเป็นไปตามฟังก์ชั่น Calculated_RemainningTime(*m*) นั้นมีช่วงเวลาเหลือเท่าไรตามตารางที่ 3.5

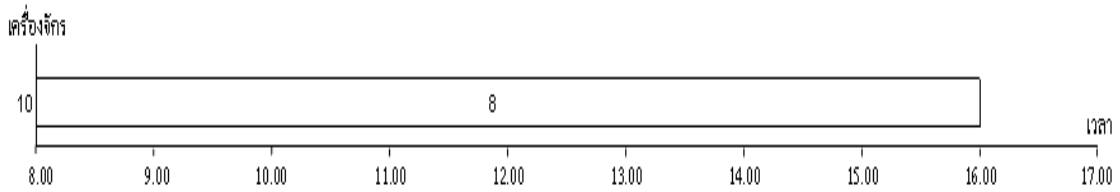
สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 3.5 ช่วงเวลาที่เหลือจากเมื่องานได้ทำการผลิตแล้ว

งาน	เครื่องจักร	ช่วงเวลาการผลิต	ช่วงเวลาการผลิตที่เหลือเมื่องานได้มีการผลิต
1	5	16	(36-(16+1)) = 19>8
8	10	32	(36-(32+1)) = 3<8
2	2	14	(36-(14+1)) = 21>8
5	6	12	(36-(12+1)-(14+1))= 8=8
7	6	14	
4	3	16	(36-(16+1)-(8+1))= 13>8
6	3	8	
10	1	8	(36-(8+1)) = 27>8
3	7	16	(36-(16+1)-(8+1))= 13>8
9	7	8	

งานที่ 8 มีช่วงเวลาที่เหลือจากที่งานได้มีการผลิตไปแล้วมีค่าน้อยกว่าช่วงเวลาที่น้อยที่สุดของช่วงเวลาการผลิตของแต่ละงาน ซึ่งงานอื่นๆ ไม่ได้เป็นไปตามสมการที่ 3.1 ดังนั้นนำงานที่ 8 มาทำการผลิตเมื่อเริ่มเวลาการผลิตได้เลย โดยการจัดงานที่ 8 เครื่องจักรที่ 10 เริ่มทำงานที่ช่วงเวลาที่ 1 ดังรูปที่ 3.6

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)
1	45.10	**10**	45.10	**19**	45.10	**28**	45.10
2	45.10	**11**	45.10	**20**	45.10	**29**	45.10
3	45.10	**12**	45.10	**21**	45.10	**30**	45.10
4	45.10	**13**	45.10	**22**	45.10	**31**	45.10
5	45.10	**14**	45.10	**23**	45.10	**32**	45.10
6	45.10	**15**	45.10	**24**	45.10	**33**	0
7	45.10	**16**	45.10	**25**	45.10	**34**	0
8	45.10	**17**	45.10	**26**	45.10	**35**	0
9	45.10	**18**	45.10	**27**	45.10	**36**	0

รูปที่ 3.6 งานที่ 8 เริ่มทำงาน

4. นำเครื่องจักรที่เหลือมาทำการจัดตารางโดยการเริ่มนำงานที่ได้เรียงไว้ดังตารางที่ 3.4 ซึ่งเครื่องจักรที่นำมาจัดการผลิตต่อไปคือเครื่องจักรที่ 5 ซึ่งเป็นไปตามฟังก์ชั่น Pop(M') แล้วทำการดูว่างานที่ทำการผลิตนั้นสามารถทำงานได้เป็นช่วงเวลาสุดท้ายที่เริ่มทำการผลิตได้ตามสมการที่ 3.2 ที่เป็นไปตามฟังก์ชั่น CalculateStartingPeriod(j) ดังตารางที่ 3.6

จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 3.6 ช่วงเวลาสุดท้ายที่สามารถเริ่มทำงานได้

งาน	เครื่องจักร	ช่วงเวลาการผลิต	ช่วงเวลาสุดท้ายที่เริ่มทำการผลิตได้
1	5	16	(36-(16+1)-1) = 20
2	2	14	(36-(14+1)-1) = 22
5	6	12	(36-(12+1)-(14+1)-1) = 9
7	6	14	
4	3	16	(36-(16+1)-(8+1)-1) = 11
6	3	8	
10	1	8	(36-(8+1)-1) = 28
3	7	16	(36-(16+1)-(8+1)-1) = 11
9	7	8	

5. งานที่ 1 บนเครื่องจักรที่ 5 มีช่วงเวลาสุดท้ายที่สามารถเริ่มทำการผลิตได้คือ ช่วงเวลาที่ 20 ดังนั้นก็จะมาดูว่างานที่ 1 เริ่มทำการผลิตได้แล้วทำให้ค่ากำลังไฟฟ้าต่ำที่สุดเป็นไปตามฟังก์ชั่น LowestElectricPower(A_j) ซึ่งดูได้ดังรูปที่ 3.6 ซึ่งไม่ว่างานที่ 1 จะเริ่มทำงานช่วงเวลาที่ 1 ถึงช่วงเวลาที่ 20 กำลังไฟฟ้าต่ำที่สุด คือ 45.10+49.02 = 94.12 kW เท่ากันทุกช่วงเวลา หลังจากนั้นก็จะมาดูว่าช่วงเวลาที่เหลือติดต่อกันของแต่ละช่วงเวลาจากที่งานได้ทำไปแล้วนั้นดังตารางที่ 3.7

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

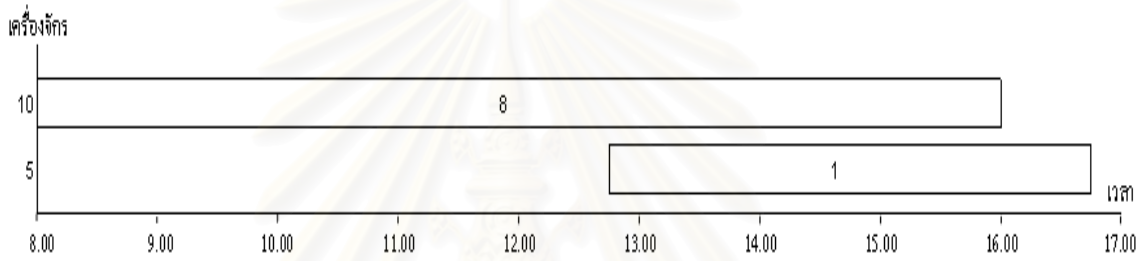
ตารางที่ 3.7 ช่วงเวลาที่เหลือติดต่อกัน

ช่วงเวลาที่เริ่มทำงาน	ช่วงเวลาที่เหลือติดต่อกันของแต่ละช่วงเวลา
1	19
2	18,1
3	17,2
4	16,3
5	15,4
6	14,5
7	13,6
8	12,7
9	11,8
10	10,9
11	9,10
12	8,11
13	7,12
14	6,13
15	5,14
16	4,15
17	3,16
18	2,17
19	1,18
20	19

จากตารางที่ 3.7 ช่วงเวลาที่เหลือติดต่อกันของแต่ละช่วงเวลาที่มากที่สุดที่เป็นไปตามฟังก์ชั่น MaxConsecutiveEmpty(B_j) คือการเริ่มผลิตที่ช่วงเวลาที่ 1 และช่วงเวลาที่ 20

6. จากตารางที่ 3.6 งานที่ 1 มีตำแหน่งที่เริ่มทำการผลิตแล้วทำให้ค่ากำลังไฟฟ้าต่ำที่สุดที่เป็นไปตามฟังก์ชั่น Count(B_j) มีด้วยกัน 20 ช่วงเวลา จึงทำให้ต้องไปดูต่อไปว่าจะเริ่มทำงานได้ที่ช่วงเวลาใดใน

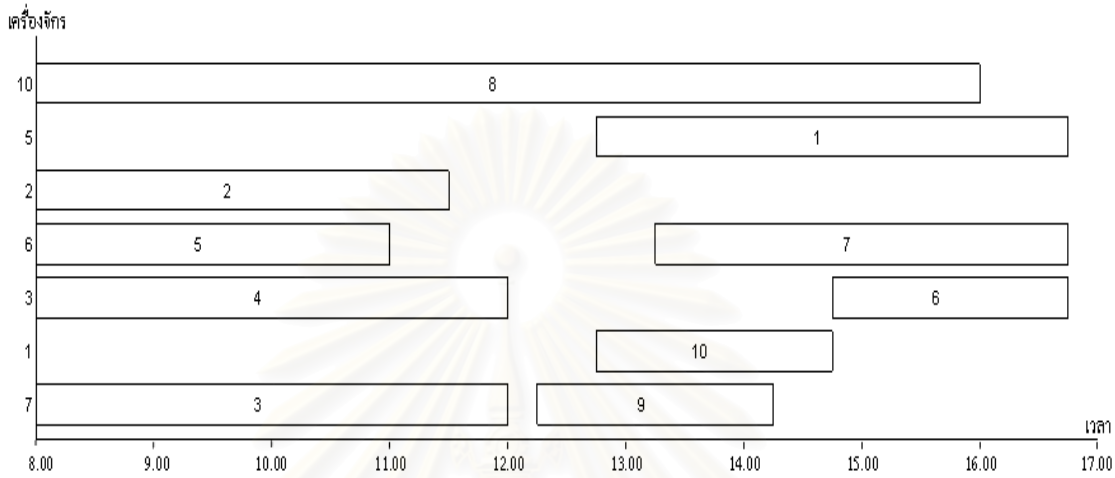
20 ช่วงเวลาที่ทำให้ค่ากำลังไฟฟ้าต่ำที่สุดและทำให้ช่วงเวลาที่เหลือติดต่อกันของแต่ละช่วงเวลามีค่ามากที่สุด ถ้ามีเพียง 1 ช่วงเวลา ก็จะเลือกช่วงเวลาดังกล่าวมาทำการผลิตได้เลย แต่ถ้ามีหลายช่วงเวลาก็จะไปดูว่าเมื่องานได้เริ่มทำการผลิตแล้วทำให้ค่าผลรวมกำลังไฟฟ้าของแต่ละช่วงเวลาที่มีการผลิตมีค่าน้อยกว่ากัน แต่ในที่นี้มี 2 ช่วงเวลาจึงทำให้ต้องดูว่าช่วงเวลาดังกล่าวนั้นช่วงเวลาใดมีค่าผลรวมกำลังไฟฟ้าของแต่ละช่วงเวลาน้อยกว่ากัน ซึ่งถ้าเริ่มการผลิตที่ช่วงเวลาที่ 1 มีผลรวมกำลังไฟฟ้าเท่ากับ 1,505.92 kW แต่ถ้าเริ่มการผลิตที่ช่วงเวลาที่ 20 มีผลรวมกำลังไฟฟ้าเท่ากับ 1370.62 kW ดังนั้นจึงเริ่มการผลิตที่ช่วงเวลาที่ 20 ดังรูปที่ 3.7 งานที่ 1 เครื่องจักรที่ 5 เริ่มทำงานที่ช่วงเวลาที่ 20



ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)
1	45.10	**10**	45.10	**19**	45.10	**28**	94.12
2	45.10	**11**	45.10	**20**	94.12	**29**	94.12
3	45.10	**12**	45.10	**21**	94.12	**30**	94.12
4	45.10	**13**	45.10	**22**	94.12	**31**	94.12
5	45.10	**14**	45.10	**23**	94.12	**32**	94.12
6	45.10	**15**	45.10	**24**	94.12	**33**	49.02
7	45.10	**16**	45.10	**25**	94.12	**34**	49.02
8	45.10	**17**	45.10	**26**	94.12	**35**	49.02
9	45.10	**18**	45.10	**27**	94.12	**36**	0

รูปที่ 3.7 งานที่ 1 เริ่มทำงาน

หลังจากนี้งานต่อไปก็ทำตามขั้นตอนเหมือนกับงานที่แล้วไปเรื่อยๆ จนกระทั่งครบทุกงานแล้ว จะได้ผลดังรูปที่ 3.8



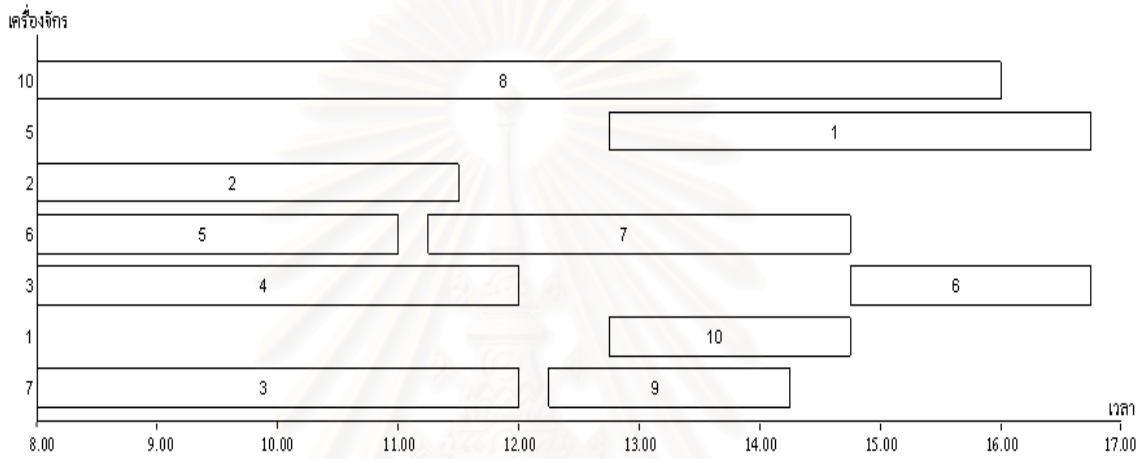
ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)	ช่วงเวลา	กำลังไฟฟ้า (kW)
1	146.12	**10**	146.12	**19**	103.47	**28**	**151.49**
2	146.12	**11**	146.12	**20**	103.47	**29**	**151.49**
3	146.12	**12**	146.12	**21**	135.34	**30**	**151.49**
4	146.12	**13**	114.25	**22**	135.34	**31**	**151.49**
5	146.12	**14**	114.25	**23**	135.34	**32**	**151.49**
6	146.12	**15**	72.58	**24**	135.34	**33**	106.39
7	146.12	**16**	72.58	**25**	135.34	**34**	106.39
8	146.12	**17**	45.10	**26**	133.36	**35**	106.39
9	146.12	**18**	47.08	**27**	133.36	**36**	0

รูปที่ 3.8 ผลิตครบทุกงาน

เมื่องานทุกงานได้ทำการผลิตเสร็จแล้วมีค่ากำลังไฟฟ้าสูงสุด 151.49 kW

จากนั้นก็นำเอาคำตอบที่ได้มาทำการปรับปรุงคำตอบโดยการทำให้ค่ากำลังไฟฟ้าสูงสุดมีค่าน้อยลง โดยการดูว่าช่วงเวลาที่ทำให้เกิดค่ากำลังไฟฟ้าสูงสุดอยู่ที่ช่วงเวลาใด ซึ่งในปัญหานี้อยู่ในช่วงเวลาที่ 28-32 แล้วก็จะพิจารณาว่ามีงานที่ 1 งานที่ 6 งานที่ 7 และงานที่ 8 ที่อยู่ในช่วงเวลาดังกล่าว

สามารถไปเริ่มต้นการผลิตใหม่ที่ทำให้คำตอบเป็นไปได้แล้วทำให้ค่ากำลังไฟฟ้าสูงสุดมีค่าน้อยลง ถ้าน้อยลงก็จะนำงานนั้นมาทำการเริ่มต้นการผลิตที่เวลาใหม่ได้ แต่ถ้าไม่งานดังกล่าวก็จะทำการเริ่มต้นที่เวลาการผลิตเดิม ซึ่งในที่นี้งานที่สามารถเริ่มต้นการผลิตใหม่แล้วทำให้มีคำตอบที่เป็นไปได้และมีค่ากำลังไฟฟ้าน้อยลงจากค่ากำลังไฟฟ้าเดิม คือ งานที่ 7 ได้ทำการเลื่อนการผลิตมาที่ช่วงเวลาที่ 14 ดังรูปที่ 3.9 ทำให้ค่ากำลังไฟฟ้าสูงสุดมีค่าเท่ากับ 146.12 kW



รูปที่ 3.9 ตารางการทำงานเมื่อมีการปรับปรุงคำตอบ

ค่าไฟฟ้ารวมมีค่า (1089.5*0.4683)+(132.93*146.12) เท่ากับ 19,933.94 บาท □

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 4

การทดสอบฮิวริสติกและการวิเคราะห์ผลงานวิจัย

การประเมินผลการจัดตารางการผลิตที่ได้จากการหาคำตอบด้วยฮิวริสติกสำหรับปัญหาการจัดตารางการผลิตที่ทำให้ค่ากำลังไฟฟ้าสูงสุดมีค่าน้อยที่สุด จะแสดงให้เห็นถึงประสิทธิภาพในการค้นหาคำตอบที่เหมาะสม นอกจากนี้การวิเคราะห์และประเมินผลงานวิจัยยังทำให้ทราบถึงข้อบกพร่องของฮิวริสติกเพื่อนำไปปรับปรุงต่อไป ซึ่งแนวทางการวิเคราะห์และประเมินงานวิจัยมีดังนี้

4.1 วิธีการทดสอบฮิวริสติก

เนื่องจากการค้นหาคำตอบด้วยฮิวริสติกนั้น ไม่รับประกันว่าคำตอบที่ได้จะเป็นคำตอบที่เหมาะสมที่สุด ดังนั้นการทดสอบประสิทธิภาพของฮิวริสติกจึงควรเปรียบเทียบกับคำตอบที่เหมาะสมที่สุดภายใต้ปัญหาเดียวกัน

สำหรับการทดสอบประสิทธิภาพของฮิวริสติกในด้านการหาคำตอบนั้น ผู้วิจัยได้ทำการสร้างปัญหาจำนวน 5 ขนาด ได้แก่ งานที่มีการผลิตจำนวน 5 งาน 10 งาน 15 งาน 20 งาน และ 30 งาน โดยที่งานแต่ละงานสามารถมีเครื่องจักรที่ทำการผลิตได้ตั้งแต่ 1 เครื่องจักร งานแต่ละงานเป็นอิสระต่อกัน งานที่ทำบนเครื่องจักรเดียวกันเป็นงานประเภทเดียวกันแต่มีเวลาต่างกันเนื่องจากมีการปริมาณการผลิตไม่เท่ากัน แล้วทำการหาคำตอบที่เหมาะสมที่สุดโดยสร้างตัวแบบเชิงคณิตศาสตร์ และหาคำตอบโดยใช้โปรแกรม CPLEX 8.0 จากนั้น จึงทำการหาคำตอบของปัญหาทดสอบเหล่านี้อีกครั้งหนึ่งโดยการใช้ฮิวริสติกที่นำเสนอ และทำการเปรียบเทียบคำตอบ และเวลาที่ใช้ในการคำนวณระหว่างตัวแบบเชิงคณิตศาสตร์กับฮิวริสติกแบบที่ 1 แบบที่ 2 และแบบที่ 3

4.2 ผลการทดสอบฮิวริสติก

4.2.1 ปัญหาทดสอบที่มีงานที่ผลิต 5 งาน

เบื้องต้นได้ทำการสร้างปัญหาที่มีงานที่ผลิต 5 งาน จากนั้นทำการหาค่าที่เหมาะสมที่สุดของปัญหาทดสอบโดยทำการแปลงปัญหาทดสอบให้อยู่ในตัวแบบเชิงคณิตศาสตร์ ซึ่งใช้ระยะเวลาในการค้นหาคำตอบที่เหมาะสมที่สุด 97.68 วินาทีโดยเฉลี่ย จากนั้นจึงทำการหาคำตอบของปัญหาทดสอบโดยใช้ฮิวริสติกเพื่อเปรียบเทียบคำตอบและเวลาที่ใช้ในการคำนวณดังตารางที่ 4.1

ตารางที่ 4.1 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 5 งาน

ปัญหาทดสอบ	คำตอบที่เหมาะสมที่สุด		คำตอบโดยวิธีฮิวริสติก				ค่าความผิดพลาด (%)		
	ค่าไฟฟ้า (บาท)	เวลา (วินาที)	ค่าไฟฟ้า(บาท)			เวลา (วินาที)	ค่าไฟฟ้า (บาท)		
			แบบที่ 1	แบบที่ 2	แบบที่ 3		แบบที่ 1	แบบที่ 2	แบบที่ 3
P5-1	6961.08	9.08	6961.08	6961.08	6961.08	0.22	0	0	0
P5-2	12085.80	3.73	12085.80	12085.80	12085.80	0.22	0	0	0
P5-3	12179.28	416.53	12179.28	12179.28	12179.28	0.22	0	0	0
P5-4	8280.31	5.23	8280.31	8280.31	8280.31	0.22	0	0	0
P5-5	9029.48	1.41	9029.48	9029.48	9029.48	0.22	0	0	0
P5-6	12631.34	21.38	12631.34	12631.34	12631.34	0.22	0	0	0
P5-7	9772.20	7.09	9772.20	9772.20	9772.20	0.22	0	0	0
P5-8	10080.71	40.88	10080.71	10080.71	10080.71	0.22	0	0	0
P5-9	7830.26	152.03	7830.26	7830.26	7830.26	0.22	0	0	0
P5-10	11016.14	319.42	11016.14	11016.14	11016.14	0.22	0	0	0
ค่าเฉลี่ย	**-**	**97.68**	**-**	**-**	**-**	**0.22**	**0.0**	**0.0**	**0.0**

จากการพิจารณาตารางที่ 4.1 เห็นได้ว่ากำลังไฟฟ้าที่ใช้ในการผลิตที่ได้จากฮิวริสติกที่นำเสนอและกำลังไฟฟ้าที่ใช้ในการผลิตที่เหมาะสมที่สุดสำหรับปัญหาทดสอบ พบว่าฮิวริสติกแบบที่ 1 ฮิวริสติกแบบที่ 2 และฮิวริสติกแบบที่ 3 ไม่มีความแตกต่าง เมื่อพิจารณาความแตกต่างด้านเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอและวิธีการหาคำตอบที่เหมาะสมที่สุด ซึ่งทำการทดสอบบนเครื่องคอมพิวเตอร์รุ่น Pentium IV 3.0 GHz หน่วยความจำ 512 Mb พบว่าค่าเฉลี่ยของเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอคือ 0.22 วินาที ในขณะที่ค่าเฉลี่ยของเวลาที่ในการหาคำตอบที่เหมาะสมที่สุดคือ 97.68 วินาที

4.2.2 ปัญหาทดสอบที่มีงานที่ผลิต 10 งาน

เบื้องต้นได้ทำการสร้างปัญหาที่มีงานที่ผลิต 10 งาน จากนั้นทำการหาค่าที่เหมาะสมที่สุดของปัญหาทดสอบโดยทำการแปลงปัญหาทดสอบให้อยู่ในตัวแบบเชิงคณิตศาสตร์ ซึ่งได้กำหนดระยะเวลาในการค้นหาคำตอบที่เหมาะสมที่สุดประมาณ 7,200 วินาทีโดยเฉลี่ย เนื่องจากเป็นปัญหาที่มีขนาดใหญ่ ถ้าทำการแก้ปัญหาด้วยตัวแบบเชิงคณิตศาสตร์อาจจะหาคำตอบได้ซึ่งทำให้ใช้เวลานาน หรือไม่ก็หาคำตอบไม่ได้ จากนั้นจึงทำการหาคำตอบของปัญหาทดสอบโดยใช้ฮิวริสติกเพื่อเปรียบเทียบคำตอบและเวลาที่ใช้ในการคำนวณดังตารางที่ 4.2

ตารางที่ 4.2 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุดสำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 10 งาน

ปัญหาทดสอบ	คำตอบที่เหมาะสมที่สุด		คำตอบโดยวิธีฮิวริสติก				ค่าความผิดพลาด (%)		
	ค่าไฟฟ้า (บาท)	Gaps (%)	ค่าไฟฟ้า(บาท)			เวลา (วินาที)	ค่าไฟฟ้า (บาท)		
			แบบที่ 1	แบบที่ 2	แบบที่ 3		แบบที่ 1	แบบที่ 2	แบบที่ 3
P10-1	21179.81	8.70	21435.67	26547.24	26547.24	1.98	1.21	5.82	1.21
P10-2	24900.46	41.67	25631.58	29462.38	29462.38	1.98	2.94	13.61	13.61
P10-3	17816.87	14.06	19832.31	28903.42	28903.42	1.65	11.31	11.31	11.31
P10-4	22661.44	14.49	22661.44	31464.32	31464.32	1.70	0.00	2.31	0.00
P10-5	18997.83	26.26	19521.58	28149.21	28149.21	1.98	2.76	2.76	2.76
P10-6	20889.43	39.89	20878.42	27883.61	27883.61	1.59	-0.05	18.98	18.98
P10-7	21045.85	14.89	21501.79	26377.15	26377.15	1.59	2.17	2.17	2.17
P10-8	27043.67	46.15	27043.67	26704.97	26704.97	1.65	0.00	0.00	0.00
P10-9	19621.77	26.92	20274.01	26577.42	26186.61	1.59	3.32	14.60	14.60
P10-10	21278.23	25.00	21278.23	17013.71	17013.71	1.59	0.00	0.00	0.00
ค่าเฉลี่ย	**-**	**25.08**	**-**	**-**	**-**	**1.73**	**2.37**	**7.16**	**6.46**

จากตารางที่ 4.2 จะเห็นว่าคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์ที่เวลา 2 ชั่วโมงมีช่องว่างของคำตอบเฉลี่ย 25.08% จากขอบเขตล่าง ส่วนกำลังไฟฟ้าที่ใช้ในการผลิตที่ได้จากฮิวริสติกที่นำเสนอและกำลังไฟฟ้าที่ใช้ในการผลิตที่เหมาะสมที่สุดสำหรับปัญหาทดสอบ พบว่าฮิวริสติกแบบที่ 1 มีความ

แตกต่างโดยเฉลี่ย 2.37 % ฮิวริสติกแบบที่ 2 มีความแตกต่างโดยเฉลี่ย 7.16 % และฮิวริสติกแบบที่ 3 มีความแตกต่างโดยเฉลี่ย 6.46 % เมื่อพิจารณาความแตกต่างด้านเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอและวิธีการหาคำตอบที่เหมาะสมที่สุด ซึ่งทำการทดสอบบนเครื่องคอมพิวเตอร์รุ่น Pentium IV 3.0 GHz หน่วยความจำ 512 Mb พบว่าค่าเฉลี่ยของเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอคือ 1.73 วินาที

4.2.3 ปัญหาทดสอบที่มีงานที่ผลิต 15 งาน

เบื้องต้นได้ทำการสร้างปัญหาที่มีงานที่ผลิต 15 งาน จากนั้นทำการหาค่าที่เหมาะสมที่สุดของปัญหาทดสอบโดยทำการแปลงปัญหาทดสอบให้อยู่ในตัวแบบเชิงคณิตศาสตร์ ซึ่งใช้ระยะเวลาในการค้นหาคำตอบที่เหมาะสมที่สุดประมาณ 7,200 วินาทีโดยเฉลี่ย เนื่องจากเป็นปัญหาที่มีขนาดใหญ่ถ้าทำการแก้ปัญหาด้วยตัวแบบเชิงคณิตศาสตร์อาจจะหาคำตอบได้ซึ่งทำให้ใช้เวลานาน หรือไม่ก็หาคำตอบไม่ได้ จากนั้นจึงทำการหาคำตอบของปัญหาทดสอบโดยใช้ฮิวริสติกเพื่อเปรียบเทียบคำตอบดังตารางที่ 4.3

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 4.3 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 15 งาน

ปัญหาทดสอบ	คำตอบที่เหมาะสมที่สุด		คำตอบโดยวิธีฮิวริสติก				ค่าความผิดพลาด (%)		
	ค่าไฟฟ้า (บาท)	Gaps (%)	ค่าไฟฟ้า(บาท)			เวลา (วินาที)	ค่าไฟฟ้า (บาท)		
			แบบที่ 1	แบบที่ 2	แบบที่ 3		แบบที่ 1	แบบที่ 2	แบบที่ 3
P15-1	25639.93	8.33	26547.24	26547.24	26547.24	2.69	3.54	3.54	3.54
P15-2	27518.86	11.27	28876.16	29462.38	29462.38	2.25	4.93	7.06	7.06
P15-3	27861.25	6.45	28903.42	28903.42	28903.42	2.53	3.74	3.74	3.74
P15-4	29395.59	32.89	30291.88	31464.32	31464.32	2.20	3.05	7.04	7.04
P15-5	26591.56	16.47	28149.21	28149.21	28149.21	2.20	5.86	5.86	5.86
P15-6	26517.23	12.50	27883.61	27883.61	27883.61	2.20	5.15	5.15	5.15
P15-7	25006.65	24.00	25047.85	26377.15	26377.15	2.31	0.16	5.48	5.48
P15-8	25700.71	27.59	25573.74	26704.97	26704.97	3.24	-0.49	3.91	3.91
P15-9	25206.92	18.75	26186.61	26577.42	26186.61	2.86	3.89	5.44	3.89
P15-10	16359.69	10.94	17013.71	17013.71	17013.71	2.69	4.00	4.00	4.00
ค่าเฉลี่ย	**-**	**16.92**	**-**	**-**	**-**	**2.52**	**3.38**	**5.12**	**4.97**

จากตารางที่ 4.3 จะเห็นว่าคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์ที่เวลา 2 ชั่วโมงมีช่องว่างของคำตอบเฉลี่ย 16.92% จากขอบเขตล่าง ส่วนกำลังไฟฟ้าที่ใช้ในการผลิตที่ได้จากฮิวริสติกที่นำเสนอและกำลังไฟฟ้าที่ใช้ในการผลิตที่เหมาะสมที่สุดสำหรับปัญหาทดสอบ พบว่าฮิวริสติกแบบที่ 1 มีความแตกต่างโดยเฉลี่ย 2.52 % ฮิวริสติกแบบที่ 2 มีความแตกต่างโดยเฉลี่ย 5.12 % และฮิวริสติกแบบที่ 3 มีความแตกต่างโดยเฉลี่ย 4.97 % เมื่อพิจารณาความแตกต่างด้านเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอและวิธีการหาคำตอบที่เหมาะสมที่สุด ซึ่งทำการทดสอบบนเครื่องคอมพิวเตอร์รุ่น Pentium IV 3.0 GHz หน่วยความจำ 512 Mb พบว่าเวลาเฉลี่ยที่ใช้ในการหาคำตอบของฮิวริสติกคือ 2.52 วินาที

4.2.4 ปัญหาทดสอบที่มีงานที่ผลิต 20 งาน

เบื้องต้นได้ทำการสร้างปัญหาที่มีงานที่ผลิต 20 งาน จากนั้นทำการหาค่าที่เหมาะสมที่สุดของปัญหาทดสอบโดยทำการแปลงปัญหาทดสอบให้อยู่ในตัวแบบเชิงคณิตศาสตร์ ซึ่งใช้ระยะเวลาในการค้นหาคำตอบที่เหมาะสมที่สุดประมาณ 7,200 วินาทีโดยเฉลี่ย เนื่องจากเป็นปัญหาที่มีขนาดใหญ่ถ้าทำการแก้ปัญหาด้วยตัวแบบเชิงคณิตศาสตร์อาจจะหาคำตอบได้ซึ่งทำให้ใช้เวลานาน หรือไม่ก็หาคำตอบไม่ได้ จากนั้นจึงทำการหาคำตอบของปัญหาทดสอบโดยใช้ฮิวริสติกเพื่อเปรียบเทียบคำตอบดังตารางที่ 4.4

ตารางที่ 4.4 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุดสำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 20 งาน

ปัญหาทดสอบ	คำตอบที่เหมาะสมที่สุด		คำตอบโดยวิธีฮิวริสติก				ค่าความผิดพลาด (%)		
	ค่าไฟฟ้า (บาท)	Gaps (%)	ค่าไฟฟ้า(บาท)			เวลา (วินาที)	กำลังไฟฟ้า (บาท)		
			แบบที่ 1	แบบที่ 2	แบบที่ 3		แบบที่ 1	แบบที่ 2	แบบที่ 3
P20-1	40786.72	32.97	43258.21	43258.21	43258.21	3.30	6.06	6.06	6.06
P20-2	32538.39	2.48	33058.52	33058.52	33058.52	3.30	1.60	1.60	1.60
P20-3	37987.13	8.86	39598.09	41489.68	39598.09	2.91	4.24	9.22	4.24
P20-4	35371.76	26.43	37386.84	40643.62	40643.62	3.63	5.70	14.90	14.90
P20-5	35971.77	16.47	36882.75	40530.35	37601.90	3.13	2.53	12.67	4.53
P20-6	31299.29	14.77	31858.99	32773.54	32773.54	2.91	1.79	4.71	4.71
P20-7	31772.75	61.54	31734.20	32776.37	33430.38	2.91	-0.12	3.16	5.22
P20-8	31593.85	10.96	31699.34	31699.34	31699.34	3.08	0.33	0.33	0.33
P20-9	29874.04	78.54	31372.16	32216.26	32216.26	3.08	5.01	7.84	7.84
P20-10	30507.57	65.78	31422.13	33180.79	32334.03	3.63	3.00	8.76	5.99
ค่าเฉลี่ย	**-**	**31.88**	**-**	**-**	**-**	**3.19**	**3.01**	**6.93**	**5.54**

จากตารางที่ 4.4 จะเห็นว่าคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์ที่เวลา 2 ชั่วโมงมีช่องว่างของคำตอบเฉลี่ย 31.88% จากขอบเขตล่าง และช่องว่างของคำตอบระหว่างคำตอบจากตัวแบบเชิงคณิตศาสตร์กับฮิวริสติกมีค่าที่ติดลบเนื่องจากคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์นั้นมีคำตอบที่ดี

ที่สุดที่เวลา 2 ชั่วโมง และกำลังไฟฟ้าที่ใช้ในการผลิตที่ได้จากฮิวริสติกที่นำเสนอและกำลังไฟฟ้าที่ใช้ในการผลิตที่เหมาะสมที่สุดสำหรับปัญหาทดสอบ พบว่าฮิวริสติกแบบที่ 1 มีความแตกต่างโดยเฉลี่ย 3.01% ฮิวริสติกแบบที่ 2 มีความแตกต่างโดยเฉลี่ย 6.93% และฮิวริสติกแบบที่ 3 มีความแตกต่างโดยเฉลี่ย 5.54% เมื่อพิจารณาความแตกต่างด้านเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอและวิธีการหาคำตอบที่เหมาะสมที่สุด ซึ่งทำการทดสอบบนเครื่องคอมพิวเตอร์รุ่น Pentium IV 3.0 GHz หน่วยความจำ 512 Mb พบว่าค่าเฉลี่ยของเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอคือ 3.19 วินาที

4.2.5 ปัญหาทดสอบที่มีงานที่ผลิต 30 งาน

เบื้องต้นได้ทำการสร้างปัญหาที่มีงานที่ผลิต 30 งาน จากนั้นทำการหาค่าที่เหมาะสมที่สุดของปัญหาทดสอบโดยทำการแปลงปัญหาทดสอบให้อยู่ในตัวแบบเชิงคณิตศาสตร์ ซึ่งใช้ระยะเวลาในการค้นหาคำตอบที่เหมาะสมที่สุดประมาณ 7,200 วินาทีโดยเฉลี่ย เนื่องจากเป็นปัญหาที่มีขนาดใหญ่ถ้าทำการแก้ปัญหาด้วยตัวแบบเชิงคณิตศาสตร์อาจจะหาคำตอบได้ซึ่งทำให้ใช้เวลานาน หรือไม่ก็หาคำตอบไม่ได้ จากนั้นจึงทำการหาคำตอบของปัญหาทดสอบโดยใช้ฮิวริสติกเพื่อเปรียบเทียบคำตอบดังตารางที่ 4.5

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 4.5 การเปรียบเทียบผลคำตอบของฮิวริสติกที่นำเสนอและคำตอบที่เหมาะสมที่สุด สำหรับปัญหาทดสอบที่มีงานที่ต้องการผลิต 30 งาน

ปัญหาทดสอบ	คำตอบที่เหมาะสมที่สุด		คำตอบโดยวิธีฮิวริสติก				ค่าความผิดพลาด (%)		
	ค่าไฟฟ้า (บาท)	Gaps (%)	ค่าไฟฟ้า(บาท)			เวลา (วินาที)	ค่าไฟฟ้า (บาท)		
			แบบที่ 1	แบบที่ 2	แบบที่ 3		แบบที่ 1	แบบที่ 2	แบบที่ 3
P30-1	48061.57	30.83	45984.28	46635.64	45984.28	5.27	-4.32	-2.97	-4.32
P30-2	47184.09	15.32	47247.19	47247.19	49792.79	4.23	0.13	0.13	5.53
P30-3	53728.19	46.34	53313.52	57547.34	53313.52	4.23	-0.77	7.11	-0.77
P30-4	55339.24	33.94	57325.81	57325.81	57325.81	4.18	3.59	3.59	3.59
P30-5	56022.92	25.23	56264.33	56003.79	57830.24	4.51	0.43	-0.03	3.23
P30-6	56506.89	100.0	59701.20	59701.20	59701.20	4.12	5.65	5.65	5.65
P30-7	53287.64	100.0	53482.97	54720.55	53482.97	4.12	0.37	2.69	0.37
P30-8	58359.93	50.00	55281.11	55348.91	55739.72	3.85	-5.28	-5.16	-4.49
P30-9	57596.56	100.0	58446.83	58644.90	58902.78	4.23	1.48	1.82	2.27
P30-10	47977.10	60.00	49149.54	49996.30	49996.30	4.51	2.44	4.21	4.21
ค่าเฉลี่ย	**-**	**56.17**	**-**	**-**	**-**	**4.32**	**0.37**	**1.70**	**1.53**

จากตารางที่ 4.5 จะเห็นว่าคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์ที่เวลา 2 ชั่วโมงมีช่องว่างของคำตอบเฉลี่ย 56.17% จากขอบเขตล่าง และช่องว่างของคำตอบระหว่างคำตอบจากตัวแบบเชิงคณิตศาสตร์กับฮิวริสติกมีค่าที่ติดลบเนื่องจากคำตอบที่ได้จากตัวแบบเชิงคณิตศาสตร์นั้นมีคำตอบที่ดีที่สุดที่เวลา 2 ชั่วโมง และกำลังไฟฟ้าที่ใช้ในการผลิตที่ได้จากฮิวริสติกที่นำเสนอและกำลังไฟฟ้าที่ใช้ในการผลิตที่เหมาะสมที่สุดสำหรับปัญหาทดสอบ พบว่าฮิวริสติกแบบที่ 1 มีความแตกต่างโดยเฉลี่ย 0.37 % ฮิวริสติกแบบที่ 2 มีความแตกต่างโดยเฉลี่ย 1.70 % และฮิวริสติกแบบที่ 3 มีความแตกต่างโดยเฉลี่ย 1.53 % เมื่อพิจารณาความแตกต่างด้านเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอและวิธีการหาคำตอบที่เหมาะสมที่สุด ซึ่งทำการทดสอบบนเครื่องคอมพิวเตอร์รุ่น Pentium IV 3.0 GHz หน่วยความจำ 512 Mb พบว่าค่าเฉลี่ยของเวลาที่ใช้ในการหาคำตอบของฮิวริสติกที่นำเสนอคือ 4.32 วินาที

4.2.6 การเปรียบเทียบประสิทธิภาพของฮิวริสติก

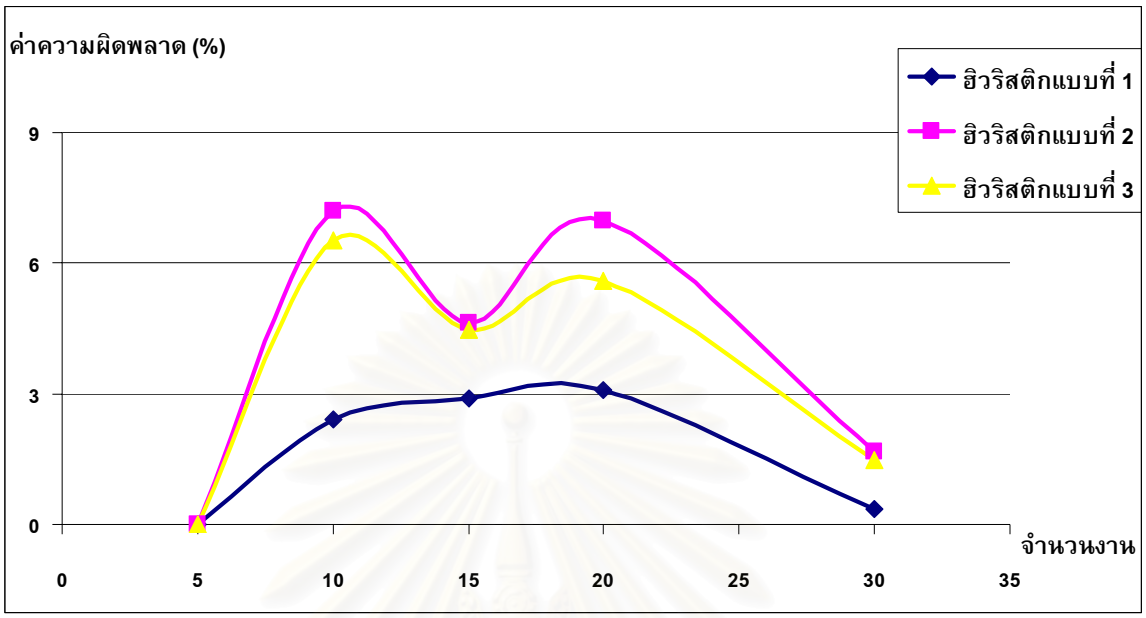
การหาคำตอบที่ดีที่สุดโดยใช้รูปแบบทางคณิตศาสตร์นั้น สามารถแก้ปัญหาได้รวดเร็วในปัญหาที่มีขนาดเล็กซึ่งการหาคำตอบที่เหมาะสมที่สุดด้วยสมการทางคณิตศาสตร์โดยใช้โปรแกรม CPLEX 8.0 พบว่าสามารถหาคำตอบได้เฉพาะปัญหาขนาด 5 งาน ส่วนขนาดจำนวนงาน 10งาน 15งาน และ 30งาน ได้มีการจำกัดเวลาที่ทำการรันตัวแบบเชิงคณิตศาสตร์เป็นเวลา 2 ชั่วโมง ผลการเปรียบเทียบคำตอบที่เหมาะสมที่สุดและคำตอบจากฮิวริสติกว่ามีความผิดพลาดมากน้อยเพียงใด สามารถแสดงได้ดังรูปที่ 4.1 ในการประเมินประสิทธิภาพของฮิวริสติกที่ได้พัฒนาขึ้นนั้น ได้ทำการเปรียบเทียบคำตอบที่ได้จากฮิวริสติกที่ได้พัฒนาขึ้นและคำตอบที่ได้จากการสร้างรูปตัวแบบเชิงคณิตศาสตร์ ได้กำหนดตัวอย่างของปัญหาที่มีขนาดแตกต่างกัน 5 ขนาด จำนวน 50 ตัวอย่าง

จากการเปรียบเทียบคำตอบที่ได้จากฮิวริสติกทั้ง 3 วิธีกับคำตอบที่เหมาะสมที่สุด จาก 50 ตัวอย่าง ดังที่ได้กล่าวมาแล้ว พบว่า

- ฮิวริสติกแบบที่ 1 ให้คำตอบเท่ากับค่าคำตอบที่เหมาะสมที่สุดจำนวน 29 ตัวอย่าง ซึ่งคิดเป็น 58 %
- ฮิวริสติกแบบที่ 2 ให้คำตอบเท่ากับค่าคำตอบที่เหมาะสมที่สุดจำนวน 20 ตัวอย่าง ซึ่งคิดเป็น 40 %
- ฮิวริสติกแบบที่ 3 ให้คำตอบเท่ากับค่าคำตอบที่เหมาะสมที่สุดจำนวน 24 ตัวอย่าง ซึ่งคิดเป็น 48 %

ฮิวริสติกแต่ละแบบก็จะมีค่าความผิดพลาดเฉลี่ยแตกต่างกันไปดังรูปที่ 4.1

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

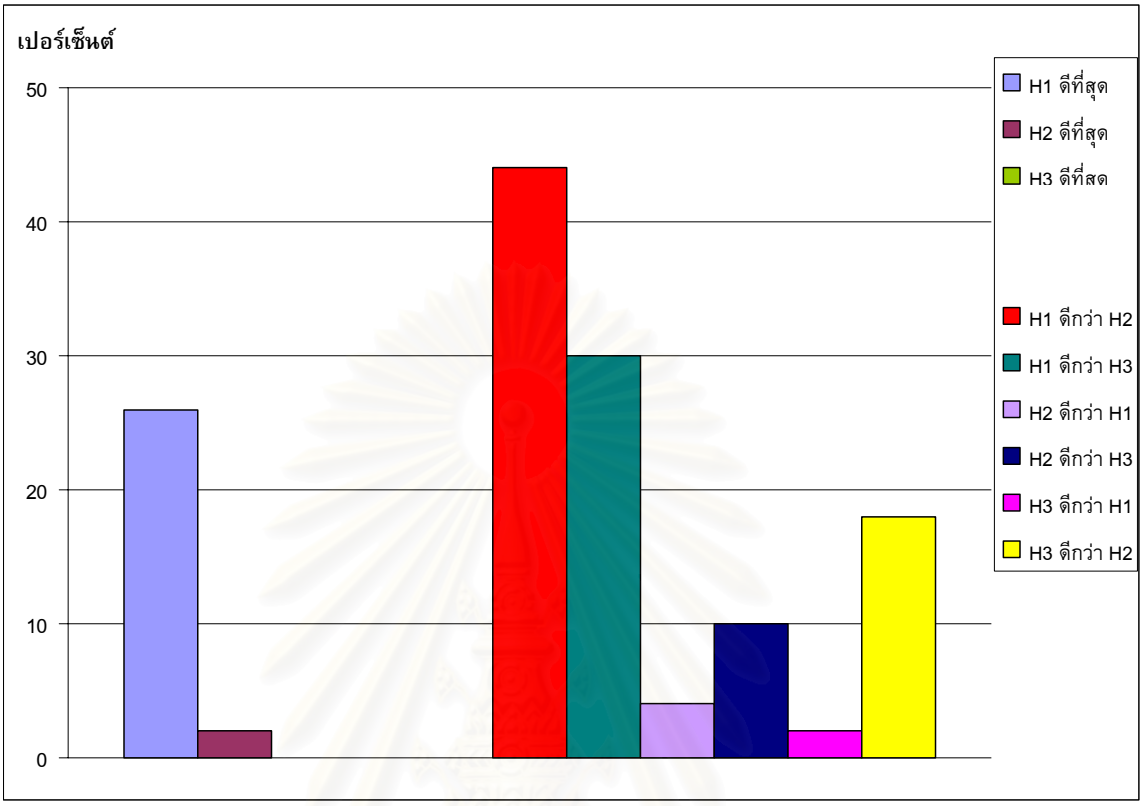


รูปที่ 4.1 เปรียบเทียบค่าความผิดพลาดของฮิวริสติกแบบต่างๆ

จะเห็นได้ว่าฮิวริสติกแบบที่ 1 มีประสิทธิภาพในการหาคำตอบได้ตรงกับค่าที่ได้จากตัวแบบเชิงคณิตศาสตร์ดีกว่าฮิวริสติกแบบที่ 2 และฮิวริสติกแบบที่ 3

4.2.6.1 การเปรียบเทียบผลลัพธ์จากการหาคำตอบฮิวริสติกทั้ง 3 แบบ ในแต่ละขนาดปัญหา ดังรูปที่ 4.2

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 4.2 เปรียบเทียบผลลัพธ์การหาคำตอบฮิวริสติกทั้ง 3 แบบใน 5 งาน 10 งาน 15 งาน 20 งาน และ 30 งาน

หมายเหตุ สัญลักษณ์ที่ใช้ในรูปที่ 4.2 แสดงความหมายได้ดังนี้

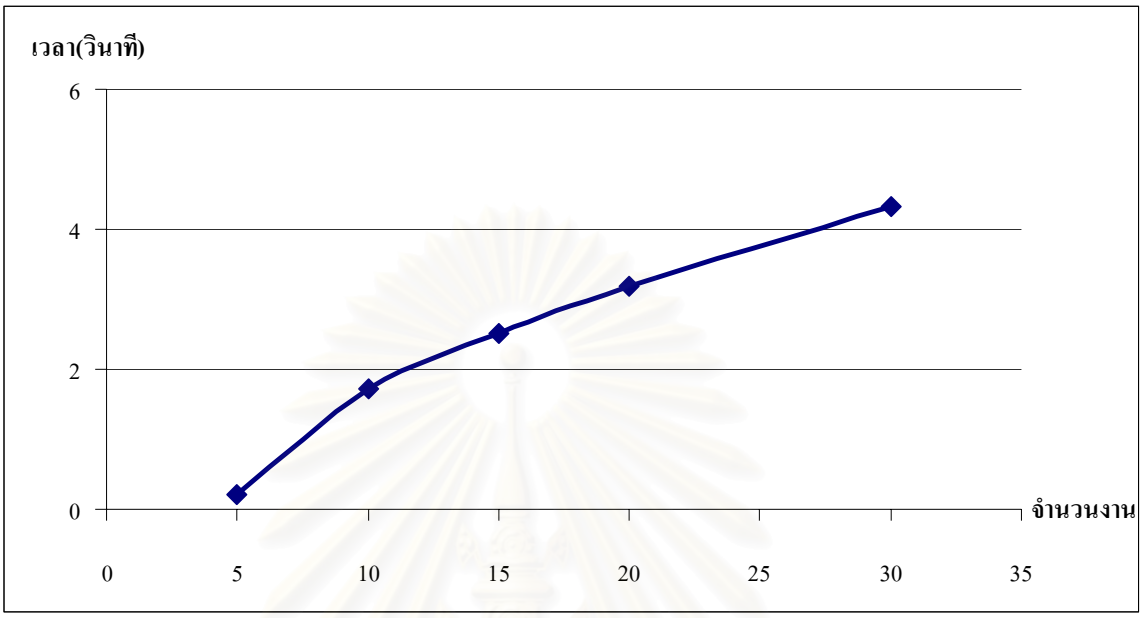
H1 คือ ฮิวริสติกแบบที่ 1

H2 คือ ฮิวริสติกแบบที่ 2

H3 คือ ฮิวริสติกแบบที่ 3

จากรูปที่ 4.2 ดูได้จากเปอร์เซ็นต์ของคำตอบที่ฮิวริสติกแบบที่ 1 มีค่าดีที่สุดสูงกว่าฮิวริสติกแบบที่ 2 และฮิวริสติกแบบที่ 3 และเมื่อนำฮิวริสติกแบบที่ 1 ไปเปรียบเทียบกับฮิวริสติกแบบที่ 2 และแบบที่ 3 ก็จะเห็นได้ว่าคำตอบของฮิวริสติกแบบที่ 1 มีเปอร์เซ็นต์ของคำตอบมีค่าที่สูงกว่า จึงสามารถสรุปได้ฮิวริสติกแบบที่ 1 ดีกว่าฮิวริสติกแบบที่ 2 และฮิวริสติกแบบที่ 3

4.2.6.2 การเปรียบเทียบแนวโน้มของเวลาที่ใช้การหาคำตอบฮิวริสติกทั้ง 3 แบบ แสดงดังรูปที่ 4.3



รูปที่ 4.3 เวลาที่ใช้ในการค้นหาคำตอบด้วยฮิวริสติกที่นำเสนอ

จากรูปที่ 4.3 จะเห็นว่าแนวโน้มของเวลาที่ใช้ในการค้นหาคำตอบของฮิวริสติกเมื่อจำนวนงานที่ต้องทำการผลิตมากขึ้น มีการเปลี่ยนแปลงไม่มากนัก โดยระยะเวลาในการค้นหาคำตอบสำหรับปัญหาที่มีจำนวนงานเพิ่มขึ้นจะใช้ระยะเวลาเพิ่มขึ้น

4.3 การวิเคราะห์ผล

จากผลการทดสอบฮิวริสติกสำหรับปัญหาการจัดตารางการผลิตให้มีค่าความต้องการพลังงานไฟฟ้าสูงสุดให้มีค่าน้อยที่สุดนั้น โดยใช้ปัญหาการจัดตารางการผลิตของโรงงานผลิตผลิตภัณฑ์บำรุงผม พบว่าฮิวริสติกแบบที่ 1 สามารถให้คำตอบได้ดีกว่าฮิวริสติกแบบที่ 2 และฮิวริสติกแบบที่ 3 โดยที่ดีกว่าเพราะว่าฮิวริสติกแบบที่ 1 มีการนำเครื่องจักรที่มีกำลังไฟฟ้ามากที่สุดมาทำการผลิตก่อนทำให้เครื่องจักรมีโอกาสเลือกช่วงเวลาในการทำงานได้มากกว่าว่าจะทำที่ช่วงเวลาใดจึงทำให้ค่ากำลังไฟฟ้าสูงสุดที่ได้มีค่าน้อยที่สุดเมื่อเทียบกับฮิวริสติกแบบที่ 2 กับฮิวริสติกแบบที่ 3

บทที่ 5

สรุปผลงานวิจัย

ผลการพัฒนาและทดสอบประสิทธิภาพการทำงานของวิธีการแก้ปัญหาแบบฮิวริสติกสำหรับปัญหาการจัดตารางการผลิตเพื่อให้ค่าไฟฟ้ามีค่าน้อยที่สุดและการดำเนินการเพื่อลดการใช้พลังงานไฟฟ้าที่นำเสนอในวิทยานิพนธ์นี้ นำมาสู่การสรุปผลการวิจัย ประโยชน์ที่ได้รับจากงานวิจัย รวมถึงข้อเสนอแนะสำหรับงานวิจัยในอนาคตดังต่อไปนี้

5.1 สรุปผลงานวิจัย

ในงานวิจัยนี้ได้ศึกษาและวิเคราะห์รูปแบบปัญหาการจัดตารางการผลิตเพื่อให้ค่าไฟฟ้ามีค่าน้อยที่สุด โดยศึกษาจากการจัดตารางการผลิตในโรงงานตัวอย่าง มีวัตถุประสงค์เพื่อหาเวลาการผลิตของงานแต่ละงาน รูปแบบปัญหาของงานวิจัยนี้มีความซับซ้อนของการคำนวณในระดับ NP-Hard ในเบื้องต้นได้ทำการศึกษาทฤษฎีและงานวิจัยที่เกี่ยวข้องกับปัญหาการจัดตารางการผลิต ซึ่งได้รับความสนใจศึกษาแยกย่อยออกไปอย่างหลากหลายตามข้อจำกัดเฉพาะของระบบงาน สำหรับวิธีแก้ไขปัญหานั้น สามารถแบ่งออกได้เป็น 2 ประเภท คือ วิธีการหาคำตอบที่ดีที่สุด และวิธีการค้นหาคำตอบโดยใช้ฮิวริสติก

สำหรับงานวิจัยนี้จะทำการวิเคราะห์และศึกษารูปแบบปัญหาภายใต้ระบบที่มีความแน่นอน นั่นคือทราบข้อมูลทั้งหมดก่อนที่จะทำการจัดตารางการผลิตและข้อมูลเหล่านี้ไม่มีการเปลี่ยนแปลง จากนั้นจึงทำการสร้างตัวแบบเชิงคณิตศาสตร์ของปัญหา โดยใช้การโปรแกรมเชิงจำนวนเต็มแบบผสม เพื่อใช้สำหรับหาคำตอบที่เหมาะสมที่สุด ซึ่งสามารถประยุกต์ใช้กับปัญหาขนาดเล็กเท่านั้น และได้มีการออกแบบขนาดปัญหาด้วยจำนวนงาน และปรับเพิ่มจำนวนงานพบว่าเมื่อกำหนดจำนวนงานเพิ่มขึ้น การหาคำตอบด้วยสมการทางคณิตศาสตร์จะมีความสามารถในการหาคำตอบได้ยากขึ้นซึ่งหมายถึงว่ามีการใช้เวลาในการหาคำตอบมากขึ้น และผลลัพธ์ที่ได้จากการหาคำตอบจากสมการทางคณิตศาสตร์สามารถหาคำตอบได้ที่ขนาดปัญหา 5 งาน ส่วนปัญหา 10 งาน 15 งาน 20 งาน และ30 งานนั้นได้มีการจำกัดเวลาในการหาคำตอบเป็น 2 ชั่วโมง

จากผลการทดสอบประสิทธิภาพการทำงานของฮิวริสติกด้วยปัญหาทดสอบ เพื่อทดสอบประสิทธิภาพในการหาคำตอบ พบว่าฮิวริสติกที่นำเสนอให้คำตอบที่เหมาะสมที่สุดคือกลยุทธ์

กำหนดเวลาการผลิตแบบที่ 1 การนำเอาเครื่องจักรที่มีกำลังไฟฟ้ามากที่สุดมาเริ่มต้นหาเวลาการผลิตของงานก่อนแล้วตามด้วยเครื่องจักรที่มีกำลังไฟฟ้าน้อยกว่า และใช้เวลาในการคำนวณน้อยมากเมื่อเปรียบเทียบกับวิธีการหาคำตอบที่เหมาะสมที่สุด ดังนั้นจากผลการทดสอบประสิทธิภาพของฮิวริสติกทั้งในด้านคุณภาพของการค้นหาคำตอบและเวลาที่ใช้ในการค้นหาคำตอบ สรุปได้ว่าฮิวริสติกที่นำเสนอมีประสิทธิภาพอยู่ในเกณฑ์ดี มีความเหมาะสมสำหรับใช้แก้ปัญหาจริงในอุตสาหกรรม

5.2 ประโยชน์ที่ได้รับจากงานวิจัย

ฮิวริสติกที่นำเสนอสามารถคำนวณหาเวลาที่งานสามารถทำการผลิตได้ โดยงานที่ทำการผลิตตามเวลาที่คำนวณได้สามารถทำให้ค่าไฟฟ้าน้อยที่สุด และใช้เวลาในการคำนวณหาคำตอบได้อย่างรวดเร็วเมื่อเปรียบเทียบกับการค้นหาคำตอบที่เหมาะสมที่สุด

5.3 ข้อเสนอแนะสำหรับงานวิจัยในอนาคต

1) ในการหาคำตอบจากสมการทางคณิตศาสตร์ ใช้เทคนิคการแบ่งย่อยเวลาเพื่อให้สามารถหาคำตอบได้ ดังนั้นสำหรับการวิเคราะห์ปัญหาที่ให้ความสนใจในข้อจำกัดอื่นๆ นอกเหนือจากนี้ต้องมีการพัฒนาเพิ่มเติม ซึ่งการจำลองปัญหาดังกล่าวช่วยให้สามารถตัดสินใจได้เพียงระดับหนึ่งเท่านั้น เมื่อปัญหามีขนาดใหญ่จะเกิดความยุ่งยากในการสร้างสมการทางคณิตศาสตร์ จึงต้องมีการพัฒนาฮิวริสติกขึ้นเพื่อให้สามารถหาคำตอบได้ใกล้เคียงกับคำตอบที่เหมาะสม

2) สามารถนำแนวคิดการจัดลำดับการทำงานไปประยุกต์ใช้ในอุตสาหกรรมอื่นๆ ได้ซึ่งงานวิจัยนี้จะช่วยลดค่าใช้จ่ายด้านค่าไฟฟ้าได้อย่างมาก

3) สามารถนำไปพัฒนากับงานที่เมื่อมีการเลือกที่จะทำการผลิตที่เครื่องจักรที่มีกำลังไฟฟ้าที่แตกต่างกันแล้วจะต้องมีเวลาการผลิตที่แตกต่างกันได้

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

เอกสารอ้างอิง

ภาษาไทย

กันต์ธร เก่งพล.การควบคุมการใช้พลังงานไฟฟ้าในโรงแรม, วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2541

จิราพร ตั้งใจทวีทรัพย์และ สุรินธร ทองคง . การศึกษาความเป็นไปได้ในการประหยัดค่าไฟฟ้าโดยการเปลี่ยนวิธีคิดค่าไฟฟ้าเป็นแบบ TOU และปรับเปลี่ยนการทำงานเพื่อให้เกิดการใช้ไฟฟ้าอย่างคุ้มค่าและประหยัดที่สุด, กิจกรรมฝึกงานเทคโนโลยีสะอาด ประจำปี 2547, ขอนแก่น, 2547.

จีรศักดิ์ ศรีสุขและ ชัชวาล ศรีหงอก . การจัดตารางการผลิตเพื่อให้ peak demand charge ในการคิดค่าไฟฟ้าแบบปกติต่ำที่สุด, ขอนแก่น, 2546.

ปารเมศ ชุติมา. เทคนิคการจัดตารางการดำเนินงาน. กรุงเทพมหานคร : สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย, 2546.

ศิริกัลยา สุวจิตตานนท์.วิทยาศาสตร์และเทคโนโลยี 3,3(ก.ย.- ธ.ค. 2531), 47-57

สำนักงานคณะกรรมการนโยบายพลังงานแห่งชาติ. การจัดระบบการจัดการพลังงาน, กรุงเทพฯ, 2545.

สุฑามาศ มนตรีบริรักษ์ . การลดการใช้พลังงานไฟฟ้าโดยการจัดลำดับการผลิตที่มีประสิทธิภาพ, วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาอุตสาหการ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น, 2547

อัตราค่าไฟฟ้าของการไฟฟ้าภูมิภาค. การไฟฟ้าส่วนภูมิภาค, 2543

เอกสิทธิ สุวรรณศรี.การปรับปรุงการจัดการด้านพลังงานในกระบวนการผลิตปูนซิเมนต์, วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2543

ภาษาอังกฤษ

Baker.K.R. Introduction to Sequencing and Scheduling. Duke University: John Wiley & Sons, 1974

Prabhu, V., and Baker, M. Industrial engineering techniques for improving operations. London : McGraw-Hill, 1986

ภาคผนวก

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ก

ตัวแบบเชิงคณิตศาสตร์บน AMPL

Minimize Cost

```
### set Index ###
set Job;
set Machine;
set Period;

### set Variables ###
var a{Job, Machine}, binary;
var StartingPeriod{Job, Machine, Period}, binary;
var WorkedPeriod{Job, Machine, Period}, binary;
var CleanedPeriod{Job, Machine, Period}, binary;
var EnergyCharge{Period}, >=0;
var Peak, >=0;

### set Parameters ###
param Horizon;
param NoJobs;
param duration{Job};
param ElectricPower{Machine};
param JobworkonMachine{Job,Machine};

### Model ###

# Obj Function #
minimize MinPeak: (0.4683*(sum{j in Job, m in Machine}
(duration[j]*a[j,m])*ElectricPower[m]) + (132.93*Peak));

# Constraint #
s.t. Constraint_00{m in Machine}:
       sum{j in Job} (duration[j]+1)*a[j,m] <= Horizon;

s.t. Constraint_0{j in Job, m in Machine}:
       a[j,m] <= JobworkonMachine[j,m];

s.t. Constraint_1{j in Job}:
       sum{m in Machine} a[j,m] = 1;

s.t. Constraint_2{j in Job, m in Machine}:
       sum{p in 1..Horizon-duration[j]} StartingPeriod[j,m,p] = a[j,m];

### Worked Period ###
s.t. Constraint_3{j in Job, m in Machine, p in 1..Horizon-duration[j]}:
       sum{q in p..p+duration[j]-1} WorkedPeriod[j,m,q] >=
duration[j]*StartingPeriod[j,m,p];

s.t. Constraint_4{j in Job, m in Machine, p in 1..Horizon-duration[j]}:
       sum{q in (Period diff p..p+duration[j]-1)} WorkedPeriod[j,m,q] <=
(Horizon-duration[j])*(1-StartingPeriod[j,m,p]);

s.t. Constraint_5{p in Period, m in Machine}:
       sum{j in Job} WorkedPeriod[j,m,p] <= 1;

### Clean ###
```

```
s.t. Constraint_6{j in Job, m in Machine, p in 1..Horizon-duration[j]}:
       CleanedPeriod[j,m,p+duration[j]] = StartingPeriod[j,m,p];

s.t. Constraint_7{j in Job, m in Machine, p in duration[j]+1..Horizon}:
       sum{ji in Job} WorkedPeriod[ji,m,p] <= (1 - CleanedPeriod[j,m,p])*NoJobs;

### Calculate Peak ###
s.t. Constraint_8{p in Period}:
       EnergyCharge[p] = sum{j in Job, m in Machine}
ElectricPower[m]*WorkedPeriod[j,m,p];

s.t. Constraint_9{p in Period}:

       Peak >= EnergyCharge[p];
```

Minimize ElectricPower

```
### set Index ###
set Job;
set Machine;
set Period;

### set Variables ###
var a{Job, Machine}, binary;
var G{Machine}, binary;

### set Parameters ###
param Horizon;
param NoJobs;
param duration{Job};
param ElectricPower{Machine};
param JobworkonMachine{Job,Machine};


### Model ###

# Obj Function #
minimize MinEP: sum{m in Machine} ElectricPower[m]*G[m];

# Constraint #
s.t. Constraint_00{m in Machine}:
       sum{j in Job} (duration[j]+1)*a[j,m] <= Horizon;
s.t. Constraint_0{j in Job, m in Machine}:
       a[j,m] <= JobworkonMachine[j,m];
s.t. Constraint_1{j in Job}:
       sum{m in Machine} a[j,m] = 1;
s.t. Constraint_2{j in Job, m in Machine}:
       G[m] >= a[j,m];
```

ภาคผนวก ข

โปรแกรมสร้างข้อมูลนำเข้าสำหรับทดสอบฮิวริสติก

```
/*Generate Problem */

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>


/* Global Variables */
FILE *fp;

int Jobs, Machines, Periods;

int i,j;


int n,r,lr;
int interval,randomduration;
float randomEP,rf,prob,fract = 100;


int main(int argc, char *argv[])
{
      char probname[100];

      if(argc != 2)
      {
            printf("Error in command line argument - Don't forget
indicate INPUT FILE!!!");
            exit(1);
      }

      printf("********************** Create Problem 'Peak'
******************\n\n");

      printf("Input the number of Jobs : ");
      scanf("%d", &Jobs);
      printf("Jobs = %d\n", Jobs);

      printf("Input the number of Machines : ");
      scanf("%d", &Machines);
      printf("Machines = %d\n", Machines);

      printf("Input the number of Periods : ");
      scanf("%d", &Periods);
      printf("Periods = %d\n", Periods);


      strcpy(probname, argv[1]);
```

```
	fp = fopen(probname,"w");


	printf("Please input NO. of Iteration to pickup 'Random Number
Seed'\n");
	printf("for Generate Problem : ");
	scanf("%d", &lr);

	for(i=1; i<lr; i++)
	{
		r = rand();
	}

	/* Write Title into File */
	fprintf(fp, "************************** INPUT
**************************\n");

	/* Write Title into File */
	fprintf(fp, "Problem		%s\n\n", probname);

	fprintf(fp, "Jobs		%d\n", Jobs);
	fprintf(fp, "Machines		%d\n", Machines);
	fprintf(fp, "Periods		%d\n\n", Periods);


	/* Write Parameters duration */
	fprintf(fp, "\n***durationofJobs***\n");
	for(i=1; i<=Jobs; i++)
	{
		fprintf(fp, "J%d", i);

		/* random number of durations */
		prob = (float)1/(6);  /* discrete probability of duraion =>
8,10,12,14,16,32 = 6 intervals */

		r = rand();
		rf = (r % 100)/fract; /* rf is random 0.00 - 0.99 */
		interval = floor(rf/prob) + 1; /* interval = 1,2,3,4,5,6 */

		/* set duration for each interval */
		if(interval == 1)
		{
			randomduration = 8;
		}
		else if(interval == 2)
		{
			randomduration = 10;
		}
		else if(interval == 3)
		{
			randomduration = 12;
		}
		else if(interval == 4)
		{
			randomduration = 14;
```

```
			}
			else if(interval == 5)
			{
				randomduration = 16;
			}
			else if(interval == 6)
			{
				randomduration = 32;
			}

			fprintf(fp, "%5d\n", randomduration);
		}


		/* Write Parameters Electric Power */
		fprintf(fp, "\n***ElectricPower***\n");
		for(i=1; i<=Machines; i++)
		{
			fprintf(fp, "M%d", i);

			r = rand();
			rf = (r % 100)/fract; /* rf is random 0.00 - 0.99 */

			randomEP = 1 + 99*rf; /* continuous distribution
Uniform[1,100] */

			fprintf(fp, "%8.2f\n", randomEP);
		}


		/* Write Parameters JobworkonMachine */
		fprintf(fp, "\n***JobworkonMachine***\n");
		for(i=1; i<=Jobs; i++)
		{
			for(j=1; j<=Machines; j++)
			{
				fprintf(fp, "J%d-M%d\n", i,j);
			}
		}

		printf("\n\nFinished");

		fclose(fp);
}
```

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ค

โปรแกรมทดสอบฮิวริสติก

```
/* Heuristic Peak */

#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include<time.h>

#define SET   1
#define UNSET 0

#define FEASIBLE 1
#define INFEASIBLE 0

#define DO  1
#define UNDO 0

#define DONE  1
#define UNDONE 0

#define START  1
#define STOP 0


FILE *fpr, *fpo;
char str[100],datafile[50];


/* Clock */
clock_t start, finish;
double runtime;

/* Read Data */
int Jobs,Machines,Periods,Horizon;
int *UsedPeriod;
int *MachineofJob;
int *durationofJob;
float *ElectricPowerofMachine;

float MinPeak, Peak;
float EnergyCharge[50];

/* Variables for Best Solution */
int **BestAllocatedPeriod, **BestWorkedPeriod, **BestScheduledPeriod;
float BestMinPeak;


int **AllocatedPeriod, **WorkedPeriod, **ScheduledPeriod; /* Period m,p
*/
int StartingPeriod[50];
```

```
int FirstPeriod[50];

/* Variables for Neighbor Solution */
int   JOB[50],FP[50];
int BestJOB1,BestFP1,BestJOB2,BestFP2;

int **TempPeriod;

int fsb = FEASIBLE;

/* variables in function Order_ElectricPower and Scheduling */
int NOofStartingPeriodofjob[100];
int MC[50],TempMC;
float EP[50],TempEP;
int SDJOB[50];
int Period,duration;


/* Parameter : function improveSolution */
int MaxIT = 20;

/****************************** Declare Prototypes
*****************************/
void ReadData();


void SetInitialSchedule();
void Order_ElectricPower_Increasing();
void Order_ElectricPower_Decreasing();
void Order_ElectricPower_Random();
void Scheduling();


void SetWorkedPeriod(int j, int StartingPeriod);
void SetAllocatedPeriod(int j, int StartingPeriod);
void SetScheduledPeriod(int j, int StartingPeriod);
void SetTempPeriod(int j, int StartingPeriod); /* same workedperiod */

void UnSetWorkedPeriod(int j, int StartingPeriod);
void UnSetAllocatedPeriod(int j, int StartingPeriod);
void UnSetScheduledPeriod(int j, int StartingPeriod);
void UnSetTempPeriod(int j, int StartingPeriod); /* same workedperiod */

void PrintSchedule();
void FPrintSchedule();
float CalculatePeak(int **TP);

void CopyPeriod(int **Period1, int **Period2);

void AllocateMemory();
void FreeMemory();

float SummationPeakofJob(int j, int i);

void FirstImproveSolution();
```

```
void ImproveSolution();

/************************** End Declare Prototypes
*****************************/

main(int argc, char *argv[])
{
      int InitialRoutes;
      int j;


      if(argc != 3)
      {
            printf("Error in command line argument - Please indicate
INPUT FILE and OUTPUT FILE!!!");
            exit(1);
      }


      strcpy(datafile, argv[1]);

      printf("Data File = %s\n\n", datafile);

      fpo = fopen(argv[2], "w");


      ReadData();


      start = clock();

      /************************** 1st Type
*****************************/
      fprintf(fpo, "\n\n\nOrdering Machine - Decreasing Electric
Power\n");
      SetInitialSchedule();
      Order_ElectricPower_Decreasing();

      Scheduling();
      if(fsb != INFEASIBLE)
      {
            /***Construct***/
            fprintf(fpo, "***Construct***\n");
            FPrintSchedule();
            fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
            printf("\n\n");
      }
      else
      {
            fprintf(fpo, "Infeasible!!!\n");
            FPrintSchedule();
      }

      FirstImproveSolution();
      /***FirstImprove***/
```

```
	fprintf(fpo, "\n\n***FirstImprove***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");

	ImproveSolution();
	/***Improve***/
	fprintf(fpo, "\n\n***Improve***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");



	/*************************** 2nd Type
****************************/
	fprintf(fpo, "\n\nOrdering Machine - Increasing Electric
Power\n");
	SetInitialSchedule();
	Order_ElectricPower_Increasing();

	Scheduling();
	if(fsb != INFEASIBLE)
	{
		/***Construct***/
		fprintf(fpo, "***Construct***\n");
		FPrintSchedule();
		fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
		printf("\n\n");
	}
	else
	{
		fprintf(fpo, "Infeasible!!!\n");
		FPrintSchedule();
	}

	FirstImproveSolution();
	/***FirstImprove***/
	fprintf(fpo, "\n\n***FirstImprove***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");

	ImproveSolution();
	/***Improve***/
	fprintf(fpo, "\n\n***Improve***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");


	/*************************** 3rd Type
****************************/
	fprintf(fpo, "\n\n\nOrdering Machine - Random Electric Power\n");
	SetInitialSchedule();
```

```
	Order_ElectricPower_Random();

	Scheduling();
	if(fsb != INFEASIBLE)
	{
		/***Construct***/
		fprintf(fpo, "***Construct***\n");
		FPrintSchedule();
		fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
		printf("\n\n");
	}
	else
	{
		fprintf(fpo, "Infeasible!!!\n");
		FPrintSchedule();
	}

	FirstImproveSolution();
	/***FirstImprove***/
	fprintf(fpo, "\n\n***FirstImprove***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");

	ImproveSolution();
	/***Improve***/
	fprintf(fpo, "\n\n***Improve***\n");
	FPrintSchedule();
	fprintf(fpo, "\n\nPeak = %.2f\n", BestMinPeak);
	printf("\n\n");



	finish = clock();

    runtime = (double)(finish - start) / CLOCKS_PER_SEC;
	printf("Run Time = %0.4lf sec\n\n", runtime);
	fprintf(fpo, "\n\nRun Time = %0.4lf sec\n\n", runtime);


	FreeMemory();
	fclose(fpo);
}


/*******************************************/
/* Function : ReadData()
/* - Read Data from Data File.
/*
/*******************************************/
void ReadData()
{
	int j,m;
	fpr = fopen(datafile, "r");
```

```
	/* Read first line - 3 parts*/
	fscanf(fpr, "%s", str);
	fscanf(fpr, "%s", str);
	fscanf(fpr, "%s", str);

	/* Read Problem */
	fscanf(fpr, "%s", str); /* Read Problem */
	fscanf(fpr, "%s", str); /* Read Problem Name */

	/* Read number of Jobs, Machine and Periods */
	fscanf(fpr, "%s", str); /* Read a Word 'Jobs' */
	fscanf(fpr, "%d", &Jobs);
	printf("Jobs = %d\n", Jobs);

	fscanf(fpr, "%s", str); /* Read a Word 'Machines' */
	fscanf(fpr, "%d", &Machines);
	printf("Machines = %d\n", Machines);

	fscanf(fpr, "%s", str); /* Read a Word 'Periods' */
	fscanf(fpr, "%d", &Periods);
	Horizon = Periods;
	printf("Periods = %d\n", Periods);

	AllocateMemory();

	/* Read duration of Job */
	fscanf(fpr, "%s", str); /* Read a line */
	for(j=1; j<=Jobs; j++)
	{
		fscanf(fpr, "%s", str); /* Read J */
		fscanf(fpr, "%d", &durationofJob[j]); /* Read duration */
	}

	for(j=1; j<=Jobs; j++)
	{
		printf("duration of J%d is %d\n", j, durationofJob[j]);
	}


	/* Electric Power */
	fscanf(fpr, "%s", str); /* Electric Power */
	for(m=1; m<=Machines; m++)
	{
		fscanf(fpr, "%s", str); /* Read M */
		fscanf(fpr, "%f", &ElectricPowerofMachine[m]); /* Read
Electric Power */
	}
	for(m=1; m<=Machines; m++)
	{
		printf("Electric Power of M%d is %.2f\n", m,
ElectricPowerofMachine[m]);
	}
```

```
	/* MachineofJob */
	fscanf(fpr, "%s", str); /* Read MachineofJob */
	for(j=1; j<=Jobs; j++)
	{
		fscanf(fpr, "%s", str); /* Read J */
		fscanf(fpr, "%d", &MachineofJob[j]);
	}
	for(j=1; j<=Jobs; j++)
	{
		printf("J%d is %d\n", j, MachineofJob[j]);
	}

	fclose(fpr);
}


/*********************************************/
/* Function : SetInitialSchedule()
/* - Set Initial Scheduling.
/*
/*********************************************/
void SetInitialSchedule()
{
	int m,p;

	for(m=1; m<=Machines; m++)
	{
		for(p=1; p<=Periods; p++)
		{
			WorkedPeriod[m][p] = 0;
			AllocatedPeriod[m][p] = 0;
			ScheduledPeriod[m][p] = 0;
			TempPeriod[m][p] = 0;

			BestWorkedPeriod[m][p] = 0;
			BestAllocatedPeriod[m][p] = 0;
			BestScheduledPeriod[m][p] = 0;
		}
	}

	BestMinPeak = 0;
	MinPeak = 0;
}


/*********************************************/
/* Function : Order_ElectricPower_Increasing()
/* - Order Electric power - Increasing.
/*
/*********************************************/
void Order_ElectricPower_Increasing()
{
	int i,j;
```

```
	/* Set Schedule JOB */
	for(j=1; j<=Jobs; j++)
	{
		SDJOB[j] = 0;
	}

	/* Copy Machine and Electric Power */
	for(i=1; i<=Machines; i++)
	{
		MC[i] = i;
		EP[i] = ElectricPowerofMachine[i];
	}

	for(i=1; i<=Machines; i++)
	{
		printf("Machine %d , Power = %f\n",MC[i], EP[i]);
	}

	/* Order Increasing */
	for(i=1; i<=Machines; i++)
	{
		for(j=1; j<Machines; j++)
		{
			if(EP[j] > EP[j+1])
			{
				TempEP = EP[j+1];
				EP[j+1] = EP[j];
				EP[j] = TempEP;

				TempMC = MC[j+1];
				MC[j+1] = MC[j];
				MC[j] = TempMC;
			}
		}
	}
	printf("\n\n");

	for(i=1; i<=Machines; i++)
	{
		printf("Machine %d , Power = %f\n",MC[i], EP[i]);
	}
}


/*******************************************/
/* Function : Order_ElectricPower_Decreasing()
/* - Order Electric power - Decreasing.
/*
/*******************************************/
void Order_ElectricPower_Decreasing()
{
	int i,j;

	/* Set Schedule JOB */
	for(j=1; j<=Jobs; j++)
```

```
	{
		SDJOB[j] = 0;
	}

	/* Copy Machine and Electric Power */
	for(i=1; i<=Machines; i++)
	{
		MC[i] = i;
		EP[i] = ElectricPowerofMachine[i];
	}

	for(i=1; i<=Machines; i++)
	{
		printf("Machine %d , Power = %f\n",MC[i], EP[i]);
	}

	/* Order Decreasing */
	for(i=1; i<=Machines; i++)
	{
		for(j=1; j<Machines; j++)
		{
			if(EP[j] < EP[j+1])
			{
				TempEP = EP[j+1];
				EP[j+1] = EP[j];
				EP[j] = TempEP;

				TempMC = MC[j+1];
				MC[j+1] = MC[j];
				MC[j] = TempMC;
			}
		}
	}
	printf("\n\n");

	for(i=1; i<=Machines; i++)
	{
		printf("Machine %d , Power = %f\n",MC[i], EP[i]);
	}
}


/*****************************************/
/* Function : Order_ElectricPower_Random()
/* - Order Electric power - Random.
/*
/*****************************************/
void Order_ElectricPower_Random()
{
	int i,j,m,r;
	int SAME;

	/* Set Schedule JOB */
	for(j=1; j<=Jobs; j++)
	{
```

```
	SDJOB[j] = 0;
}

/* Copy Machine and Electric Power */
for(i=1; i<=Machines; i++)
{
	MC[i] = i;
	EP[i] = ElectricPowerofMachine[i];
}

for(i=1; i<=Machines; i++)
{
	printf("Machine %d , Power = %f\n",MC[i], EP[i]);
}

/* Order Random */
i=1;
while(i <= Machines)
{
	r = rand();
	m = (r % Machines) + 1;

	if(i==1)
	{
		MC[i] = m;
		EP[i] = ElectricPowerofMachine[m];
		i++;
	}
	else if(i > 1)
	{
		SAME = 0;

		/* check SAME */
		for(j=1; j<i; j++)
		{
			if(MC[j] == m)
			{
				SAME = 1;
				break;
			}
		}

		if(SAME == 0)
		{
			MC[i] = m;
			EP[i] = ElectricPowerofMachine[m];
			i++;
		}
	}
}
printf("\n\n");


for(i=1; i<=Machines; i++)
{
```

```
            printf("Machine %d , Power = %f\n",MC[i], EP[i]);
      }
}


/*******************************************/
/* Function : Scheduling()
/* - Scheduling.
/*
/*******************************************/
void Scheduling()
{
      int i,j,m,n;
      int a,sp,rj,jb;
      float MinSumPeak,sumpk;


      int SMC[100];
      int WorkingPeriodsofMachine[50];
      int RemainingPeriodsofMachine[50];
      int MinimumPeriods;

      int SPMinPeak;
      int MaxEmptyPeriod, maxep;

      int nominpeak;

      jb=0;


      /* Set Machine = UNDO */
      for(m=1; m<=Machines; m++)
      {
            SMC[m] = UNDO;
      }

      /* First Step */
      /* find MinimumPeriods */
      for(j=1; j<=Jobs; j++)
      {
            if(j==1)
            {
                  MinimumPeriods = durationofJob[j];
            }
            else
            {
                  if(durationofJob[j] < MinimumPeriods)
                  {
                        MinimumPeriods = durationofJob[j];
                  }
            }
      }


      /* find WorkingPeriodsofMachine */
```

```
	for(m=1; m<=Machines; m++)
	{
		WorkingPeriodsofMachine[m] =
CalculateWorkingPeriodsofMachine(m);
	}

	/* find RemainingPeriodsofMachine */
	for(m=1; m<=Machines; m++)
	{
		RemainingPeriodsofMachine[m] = Horizon -
WorkingPeriodsofMachine[m];
	}

	/* Find machines which RemainingPeriodsofMachine < MinimumPeriods
*/
	for(m=1; m<=Machines; m++)
	{
		if(RemainingPeriodsofMachine[m] < MinimumPeriods)
		{
			SMC[m] = DO;
		}
	}

	/* Initial Schedule */
	for(m=1; m<=Machines; m++)
	{
		sp = 1;
		if(SMC[m] == DO)
		{
			for(j=1; j<=Jobs; j++)
			{
				if(MachineofJob[j] == m)
				{
					StartingPeriod[j] = sp;
					SetWorkedPeriod(j, StartingPeriod[j]);
					SetAllocatedPeriod(j, StartingPeriod[j]);
					SetScheduledPeriod(j, StartingPeriod[j]);
					FirstPeriod[j] = sp;

					SDJOB[j] = 1;
					sp = sp + durationofJob[j] + 1;
				}
			}
		}
	}


	/* Loop Schedule */
	for(m=1; m<=Machines; m++)
	{
		if(SMC[MC[m]] == UNDO)
		{
			for(j=1; j<=Jobs; j++)
			{
```

```
                        if(MachineofJob[j] == MC[m])
                        {
                              jb = jb+1;

                              /* Calculate Starting Periods */
                              duration = 0;
                              for(rj=1; rj<=Jobs; rj++)
                              {
                                    if(MachineofJob[rj] == MC[m] &&
SDJOB[rj] == 0)
                                    {
                                          duration = duration +
(durationofJob[rj]+1);
                                    }
                              }

                              NOofStartingPeriodofjob[j] = Horizon -
duration + 1;


                              n=1;
                              for(i=1; i<=NOofStartingPeriodofjob[j];
i++)
                              {
                                    /* Check Feasible Period */
                                    Period = FEASIBLE;
                                    for(a=i; a<=i+durationofJob[j]; a++)
                                    {

      if(AllocatedPeriod[MachineofJob[j]][a] == SET)
                                          {
                                                Period = INFEASIBLE;
                                                break;
                                          }
                                    }

                                    if(Period == FEASIBLE)
                                    {
                                          if(n==1)
                                          {


                                                CopyPeriod(WorkedPeriod,
TempPeriod);
                                                StartingPeriod[j] = i;
                                                SetTempPeriod(j,
StartingPeriod[j]);


                                                MinPeak =
CalculatePeak(TempPeriod);
                                                n = n+1;


                                                SPMinPeak = i;
                                                MaxEmptyPeriod =
CalculateMaxConsecutiveEmptyPeiod(j,i);
```

```
SummationPeakofJob(j,i);

TempPeriod);

CalculatePeak(TempPeriod);

= sp;

CalculateMaxConsecutiveEmptyPeiod(j,i);
SummationPeakofJob(j,i);

empty period */
CalculateMaxConsecutiveEmptyPeiod(j,i);

MaxEmptyPeriod)

i;
      MaxEmptyPeriod = maxep;
SummationPeakofJob(j,i);

MaxEmptyPeriod)

SummationPeakofJob(j,i);

MinSumPeak)

      SPMinPeak = i;
                                        MinSumPeak =
                                  }
                                  else if(n>1)
                                  {
                                        CopyPeriod(WorkedPeriod,
                                        sp = i;
                                        SetTempPeriod(j, sp);
                                        Peak =
                                        if(Peak < MinPeak)
                                        {
                                              MinPeak = Peak;
                                              StartingPeriod[j]
                                              SPMinPeak = i;
                                              MaxEmptyPeriod =
                                              MinSumPeak =
                                        }
                                        else if(Peak == MinPeak)
                                        {
                                              /* Calculate max
                                              maxep =
                                              if(maxep >
                                              {
                                                    SPMinPeak =
                                                    MinSumPeak =
                                              }
                                              else if(maxep ==
                                              {
                                                    sumpk =
                                                    if(sumpk <
                                                    {
```

```
		MaxEmptyPeriod = maxep;

		MinSumPeak = SummationPeakofJob(j,i);
                                                                        }
                                                                }
                                                        }

                                                        n = n+1;
                                                }
                                        }
                                }
                                if(n>1)
                                {
                                        SetWorkedPeriod(j, SPMinPeak);
                                        SetAllocatedPeriod(j, SPMinPeak);
                                        SetScheduledPeriod(j, SPMinPeak);
                                        FirstPeriod[j] = SPMinPeak;
                                        Peak = CalculatePeak(WorkedPeriod);
                                        SDJOB[j] = 1;
                                }
                                else if(n==1)
                                {
                                        printf("infeasible!!\n");
                                        fsb = INFEASIBLE;
                                        break;
                                }
                        }
                }
        }

        if(fsb != INFEASIBLE)
        {
                PrintSchedule();

                /* Keep Best */
                CopyPeriod(WorkedPeriod, BestWorkedPeriod);
                CopyPeriod(AllocatedPeriod, BestAllocatedPeriod);
                CopyPeriod(ScheduledPeriod, BestScheduledPeriod);

                printf("\n\n");
                MinPeak = CalculatePeak(WorkedPeriod);
                printf("minpeak = %.2f\n", MinPeak);
                BestMinPeak = MinPeak;
        }
        else
        {
                printf("\nInfeasible!!\n");
        }
}


/*******************************************/
/* Function : FirstImproveSolution()
```

```
/* -
/*
/*********************************************/
void FirstImproveSolution()
{
	float EPS = pow(10,-4);

	int JobatMinPeak[50];
	int i,j,p,m,rp;
	int OK;


	/* Set Job at MinPeak */
	for(i=1; i<=Jobs; i++)
	{
		JobatMinPeak[i] = 0;
	}

	/* Set Job at Peak */
	for(p=1; p<=Periods; p++)
	{
		if(MinPeak-EnergyCharge[p] <= EPS)
		{
			for(m=1; m<=Machines; m++)
			{
				if(ScheduledPeriod[m][p]>0)
				{
					JobatMinPeak[ScheduledPeriod[m][p]] = SET;
				}
			}
		}
	}

	/* print */
	for(j=1; j<=Jobs; j++)
	{
		printf("JobatMinPeak[%d] = %d\n", j, JobatMinPeak[j]);
	}

	for(j=1; j<=Jobs; j++)
	{
		if(JobatMinPeak[j] == SET)
		{

			//printf("JOB = %d\n", j);
			for(p=1; p<=Periods-durationofJob[j]; p++)
			{
				UnSetAllocatedPeriod(j, FirstPeriod[j]);

				OK = SET;
				/* loop check empty period */
				for(rp=p; rp<=p+durationofJob[j]; rp++)
				{
					if(AllocatedPeriod[MachineofJob[j]][rp] ==
SET)
```

```
                        {
                              OK = UNSET;
                              break;
                        }
                  }


                  //printf("p = %d , OK = %d\n", p, OK);

                  if(OK==SET)
                  {
                        CopyPeriod(WorkedPeriod, TempPeriod);
                        UnSetTempPeriod(j, FirstPeriod[j]);
                        SetTempPeriod(j, p);


                        Peak = CalculatePeak(TempPeriod);
                        //printf("OK Peak = %.2f\n", Peak);

                        if(Peak < BestMinPeak)
                        {
                              BestMinPeak = Peak;
                              UnSetWorkedPeriod(j,
FirstPeriod[j]);
                              SetWorkedPeriod(j, p);
                              UnSetAllocatedPeriod(j,
FirstPeriod[j]);
                              SetAllocatedPeriod(j, p);
                              UnSetScheduledPeriod(j,
FirstPeriod[j]);
                              SetScheduledPeriod(j, p);

                              FirstPeriod[j] = p;

                        }
                        else
                        {
                              /*copy Alocated back*/
                              SetAllocatedPeriod(j,
FirstPeriod[j]);
                        }
                        //PrintSchedule();
                        //getch();
                  }

            }
      }
}

printf("\n\n\n*** First Improve Solution ***\n\n");
PrintSchedule();
MinPeak = CalculatePeak(WorkedPeriod);
printf("minpeak = %.2f\n", MinPeak);
```

```
	//getch();
}

/*******************************************/
/* Function : ImproveSolution()
/* -
/*
/*******************************************/
void ImproveSolution()
{
	int it=0;
	int r;
	int mc,nj;
	int i,j,p,Temp;

	int J1,SP1,J2,SP2;


	while(it < MaxIT)
	{
		it = it+1;

		/* random Machine */
		r = rand();
		r = r % Machines;
		mc = r+1;

		/* Count no.job in M/C and keep value */
		nj=0;
		for(j=1; j<=Jobs; j++)
		{
			if(MachineofJob[j] == mc)
			{
				nj = nj+1;
				JOB[nj] = j;
				FP[nj] = FirstPeriod[j];
			}
		}

		printf("*** Iteration = %d ***\n", it);
		PrintSchedule();
		printf("Swaped M/C = %d , No. of Job in M/C = %d\n", mc,
nj);

		/*print*/
		for(j=1; j<=nj; j++)
		{
			printf("JOB[%d] = %d , FP[%d] =
%d\n\n",j,JOB[j],j,FP[j]);
		}
		//getch();


		if(nj == 0)
		{
```

```
                /**/
        }
        else if(nj == 1)
        {
                /**/
        }
        else if(nj > 1)
        {
                /* Order nj increasing period */
                for(i=1; i<=nj; i++)
                {
                        for(j=1; j<nj; j++)

                {
                                if(FP[j] > FP[j+1])
                                {
                                        Temp = FP[j+1];
                                        FP[j+1] = FP[j];
                                        FP[j] = Temp;

                                        Temp = JOB[j+1];
                                        JOB[j+1] = JOB[j];
                                        JOB[j] = Temp;
                                }
                        }
                }

                /*for(i=1; i<=nj; i++)
                {
                        printf("JOB[%d] = %d , FP[%d] =
%d\n",i,JOB[i],i,FP[i]);
                }
                getch();*/

                /* Swap 2-consecutive */
                for(i=1; i<nj; i++)
                {
                        if(i==1)
                        {
                                CopyPeriod(WorkedPeriod, TempPeriod);
                                UnSetTempPeriod(JOB[i], FP[i]);
                                UnSetTempPeriod(JOB[i+1], FP[i+1]);

                                SetTempPeriod(JOB[i+1], FP[i]);
                                SetTempPeriod(JOB[i],
FP[i+1]+durationofJob[JOB[i+1]]-durationofJob[JOB[i]]);

                                Peak = CalculatePeak(TempPeriod);
                                MinPeak = Peak;
                                BestJOB1 = JOB[i];
                                BestFP1 = FP[i];
                                BestJOB2 = JOB[i+1];
                                BestFP2 = FP[i+1];

                                /* First NB */
```

```
                              printf("Job1 = %d, FP1 = %d  ;  Job2 = %d,
FP2 = %d\n", JOB[i], FP[i], JOB[i+1], FP[i+1]);

                        }
                        else if(i>1)
                        {
                              CopyPeriod(WorkedPeriod, TempPeriod);
                              UnSetTempPeriod(JOB[i], FP[i]);
                              UnSetTempPeriod(JOB[i+1], FP[i+1]);

                              SetTempPeriod(JOB[i+1], FP[i]);
                              SetTempPeriod(JOB[i],
FP[i+1]+durationofJob[JOB[i+1]]-durationofJob[JOB[i]]);

                              Peak = CalculatePeak(TempPeriod);

                              /* Keep BestNeighbour */
                              if(Peak < MinPeak)
                              {
                                    MinPeak = Peak;
                                    BestJOB1 = JOB[i];
                                    BestFP1 = FP[i];
                                    BestJOB2 = JOB[i+1];
                                    BestFP2 = FP[i+1];
                              }
                        }
                  }

                  /* Copy BestNB to Current Solution */
                  UnSetWorkedPeriod(BestJOB1, BestFP1);
                  UnSetAllocatedPeriod(BestJOB1, BestFP1);
                  UnSetScheduledPeriod(BestJOB1, BestFP1);

                  UnSetWorkedPeriod(BestJOB2, BestFP2);
                  UnSetAllocatedPeriod(BestJOB2, BestFP2);
                  UnSetScheduledPeriod(BestJOB2, BestFP2);


                  SetWorkedPeriod(BestJOB2, BestFP1);
                  SetAllocatedPeriod(BestJOB2, BestFP1);
                  SetScheduledPeriod(BestJOB2, BestFP1);

                  SetWorkedPeriod(BestJOB1,
BestFP2+durationofJob[BestJOB2]-durationofJob[BestJOB1]);
                  SetAllocatedPeriod(BestJOB1,
BestFP2+durationofJob[BestJOB2]-durationofJob[BestJOB1]);
                  SetScheduledPeriod(BestJOB1,
BestFP2+durationofJob[BestJOB2]-durationofJob[BestJOB1]);

                  Temp = FirstPeriod[BestJOB2];
                  FirstPeriod[BestJOB2] = FirstPeriod[BestJOB1];
                  FirstPeriod[BestJOB1] = Temp + durationofJob[BestJOB2]
- durationofJob[BestJOB1] ;
```

```
                        /* Best NB */
                        printf("/*** Best NB ***/\n");
                        PrintSchedule();
                        //getch();

                        /* Compare BestNB and BestSoltion */
                        if(MinPeak < BestMinPeak)
                        {
                              BestMinPeak = MinPeak;
                              CopyPeriod(WorkedPeriod, BestWorkedPeriod);
                              CopyPeriod(AllocatedPeriod,
BestAllocatedPeriod);
                              CopyPeriod(ScheduledPeriod,
BestScheduledPeriod);
                        }
                  }

                  printf("BestMinPeak = %.2f\n\n\n", BestMinPeak);
                  //getch();
            }
}


/*******************************************/
/* Function : SummationPeakofJob()
/* -
/*
/*******************************************/
float SummationPeakofJob(int j, int i)
{
      float sum=0;

      int rp,rm;
      float EC=0;
      float EnergyCharge[50];

      for(rp=i; rp <= i+durationofJob[j]-1; rp++)
      {
            EnergyCharge[rp] = 0;
            for(rm=1; rm<=Machines; rm++)
            {
                  EC = TempPeriod[rm][rp]*ElectricPowerofMachine[rm];

                  EnergyCharge[rp] = EnergyCharge[rp] + EC;
            }

            sum = sum + EnergyCharge[rp];

      }
      return sum;

}


/*******************************************/
```

```
/* Function : CalculateMaxConsecutiveEmptyPeiod()
/* -
/*
/*********************************************/
int CalculateMaxConsecutiveEmptyPeiod(int j, int i)
{
	int p,n;
	int MaxCSEP,D,Count;

	MaxCSEP = 0;
	n = 0;
	D=1;
	Count = STOP;

	/* Temporary Set*/
	TempPeriod[MachineofJob[j]][i+durationofJob[j]] = SET;

	for(p=1; p<=Periods; p++)
	{
		if(TempPeriod[MachineofJob[j]][p] == UNSET)
		{
			if(n == 0)
			{
				Count = START;
			}

			if(Count == START)
			{
				n = n+1;
			}

		}
		else if(TempPeriod[MachineofJob[j]][p] == SET)
		{
			if(D == 1)
			{
				if(Count == START)
				{
					MaxCSEP = n;
					n = 0;

					D = D+1;
					Count = STOP;
				}
			}
			else if (D > 1)
			{
				if(Count == START)
				{
					if(n > MaxCSEP)
					{
						MaxCSEP = n;
						n = 0;

						D = D+1;
```

```
                                    Count = STOP;
                              }
                        }
                  }
            }

      }

      /* Last Check */
      if(TempPeriod[MachineofJob[j]][Periods] == UNSET)
      {
            if(D == 1)
            {
                  MaxCSEP = n;
            }
            else if (D > 1)
            {
                  if(n > MaxCSEP)
                  {
                        MaxCSEP = n;
                  }
            }
      }

      /* Temporary Set*/
      TempPeriod[MachineofJob[j]][i+durationofJob[j]] = UNSET;

      return MaxCSEP;
}


/*********************************************/
/* Function : CalculateWorkingPeriodsofMachine()
/* -
/*
/*********************************************/
int CalculateWorkingPeriodsofMachine(int m)
{
      int j;
      int wpm=0;

      for(j=1; j<=Jobs; j++)
      {
            if(MachineofJob[j] == m)
            {
                  wpm = wpm + durationofJob[j] + 1;
            }
      }

      return wpm;
}


/*********************************************/
/* Function : SetWorkedPeriod()
```

```
/* - Set Worked Period.
/*
/*********************************************/
void SetWorkedPeriod(int j, int StartingPeriod)
{
	int p;

	/* excluding CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
	{
		WorkedPeriod[MachineofJob[j]][p] = SET;
	}
}


/*********************************************/
/* Function : SetAllocatedPeriod()
/* - Set Allocated Period.
/*
/*********************************************/
void SetAllocatedPeriod(int j, int StartingPeriod)
{
	int p;

	/* including CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]; p++)
	{
		AllocatedPeriod[MachineofJob[j]][p] = SET;
	}
}


/*********************************************/
/* Function : SetScheduledPeriod()
/* - Set Scheduled Period.
/*
/*********************************************/
void SetScheduledPeriod(int j, int StartingPeriod)
{
	int p;

	/* excluding CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
	{
		ScheduledPeriod[MachineofJob[j]][p] = j;
	}
}


/*********************************************/
/* Function : SetTempPeriod()
/* - Set Temp Period.
/*
/*********************************************/
void SetTempPeriod(int j, int StartingPeriod)
```

```
{
	int p;

	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
	{
		TempPeriod[MachineofJob[j]][p] = SET;
	}
}


/*********************************************/
/* Function : UnSetWorkedPeriod()
/* - UnSet Worked Period.
/*
/*********************************************/
void UnSetWorkedPeriod(int j, int StartingPeriod)
{
	int p;

	/* excluding CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
	{
		WorkedPeriod[MachineofJob[j]][p] = UNSET;
	}
}


/*********************************************/
/* Function : UnSetAllocatedPeriod()
/* - Set Allocated Period.
/*
/*********************************************/
void UnSetAllocatedPeriod(int j, int StartingPeriod)
{
	int p;

	/* including CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]; p++)
	{
		AllocatedPeriod[MachineofJob[j]][p] = UNSET;
	}
}


/*********************************************/
/* Function : UnSetScheduledPeriod()
/* - Set Scheduled Period.
/*
/*********************************************/
void UnSetScheduledPeriod(int j, int StartingPeriod)
{
	int p;

	/* excluding CleanedPeriod */
	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
```

```
	{
		ScheduledPeriod[MachineofJob[j]][p] = UNSET;
	}
}


/*********************************************/
/* Function : UnSetTempPeriod()
/* - Set Temp Period.
/*
/*********************************************/
void UnSetTempPeriod(int j, int StartingPeriod)
{
	int p;

	for(p=StartingPeriod; p<=StartingPeriod+durationofJob[j]-1; p++)
	{
		TempPeriod[MachineofJob[j]][p] = UNSET;
	}
}


/*********************************************/
/* Function : PrintSchedule()
/* - Print Schedule.
/*
/*********************************************/
void PrintSchedule()
{
	int rm,rj,rp;

	/* print Worked period */
/*	printf("\nWorked Period\n");
	for(rm=1; rm<=Machines; rm++)
	{
		printf("M%d  ", rm);
		for(rp=1; rp<=Horizon; rp++)
		{
			printf("%3d",WorkedPeriod[rm][rp]);
		}
		printf("\n");
	}
	printf("\n\n");

	/* print Allocated period */
/*	printf("\nAllocated Period\n");
	for(rm=1; rm<=Machines; rm++)
	{
		printf("M%d  ", rm);
		for(rp=1; rp<=Horizon; rp++)
		{
			printf("%3d",AllocatedPeriod[rm][rp]);
		}
		printf("\n");
	}
```

```
	printf("\n\n");

	/* print schedule period */
	printf("\nSchedule Period\n");
	for(rm=1; rm<=Machines; rm++)
	{
		printf("M%d  ", rm);
		for(rp=1; rp<=Horizon; rp++)
		{
			printf("%3d",ScheduledPeriod[rm][rp]);
		}
		printf("\n");
	}
}


/*******************************************/
/* Function : FPrintSchedule()
/* - Print Schedule in file.
/*
/*******************************************/
void FPrintSchedule()
{
	int rm,rj,rp;

	/* print into file */
	/* fprint schedule period */
	fprintf(fpo, "\nSchedule Period\n");

	for(rp=1; rp<=Periods; rp++)
	{
		if(rp==1)
		{
			fprintf(fpo, "       P%d", rp);
		}
		else if(rp>1 && rp <10)
		{
			fprintf(fpo, "     P%d", rp);
		}
		else if(rp>=10)
		{
			fprintf(fpo, "    P%d", rp);
		}
	}
	fprintf(fpo, "\n");

	for(rm=1; rm<=Machines; rm++)
	{
		fprintf(fpo, "M%d", rm);
		for(rp=1; rp<=Horizon; rp++)
		{
			fprintf(fpo, "%7d",ScheduledPeriod[rm][rp]);
		}
		fprintf(fpo, "\n");
	}
```

```
	for(rp=1; rp<=Periods; rp++)
	{
		if(rp==1)
		{
			fprintf(fpo, "    %.2f", EnergyCharge[rp]);
		}
		else if(rp>1 && rp <10)
		{
			fprintf(fpo, "  %.2f", EnergyCharge[rp]);
		}
		else if(rp>=10)
		{
			fprintf(fpo, "  %.2f", EnergyCharge[rp]);
		}
	}

}


/*********************************************/
/* Function : CopyPeriod()
/* - Copy Period.
/*
/*********************************************/
void CopyPeriod(int **Period1, int **Period2)
{
	int i,j;

	for(i=1; i<=Machines; i++)
	{
		for(j=1; j<=Periods; j++)
		{
			Period2[i][j] = Period1[i][j];
		}
	}
}


/*********************************************/
/* Function : Calculate Peak()
/* - Calculate Peak.
/*
/*********************************************/
float CalculatePeak(int **TP)
{
	int rp,rm;
	float EC, Peak;


	for(rp=1; rp<=Periods; rp++)
	{
		EnergyCharge[rp] = 0;
		for(rm=1; rm<=Machines; rm++)
		{
```

```
                        EC = TP[rm][rp]*ElectricPowerofMachine[rm];

                        EnergyCharge[rp] = EnergyCharge[rp] + EC;
                }

                if(rp==1)
                {
                        Peak = EnergyCharge[rp];
                }
                else if(rp>1)
                {
                        if(EnergyCharge[rp] > Peak)
                        {
                                Peak = EnergyCharge[rp];
                        }
                }
        }

        return Peak;
}



/*********************************************/
/* Function : AllocateMemory()
/* - Allocate Memory.
/*
/*********************************************/
void AllocateMemory()
{
        int i;

        printf("%d\n", Jobs);


        MachineofJob = (int *) malloc((Jobs+1)*sizeof(int));
        durationofJob = (int *) malloc((Jobs+1)*sizeof(int));
        ElectricPowerofMachine = (float *)
malloc((Machines+1)*sizeof(float));


        BestAllocatedPeriod = (int **) malloc((Machines+1)*sizeof(int));
        for(i=0; i<=Machines; i++)
        {
                BestAllocatedPeriod[i] = (int *)
malloc((Periods+1)*sizeof(int));
        }

        BestWorkedPeriod = (int **) malloc((Machines+1)*sizeof(int));
        for(i=0; i<=Machines; i++)
        {
                BestWorkedPeriod[i] = (int *)
malloc((Periods+1)*sizeof(int));
        }
```

```
	BestScheduledPeriod = (int **) malloc((Machines+1)*sizeof(int));
	for(i=0; i<=Machines; i++)
	{
		BestScheduledPeriod[i] = (int *)
malloc((Periods+1)*sizeof(int));
	}


	AllocatedPeriod = (int **) malloc((Machines+1)*sizeof(int));
	for(i=0; i<=Machines; i++)
	{
		AllocatedPeriod[i] = (int *)
malloc((Periods+1)*sizeof(int));
	}

	WorkedPeriod = (int **) malloc((Machines+1)*sizeof(int));
	for(i=0; i<=Machines; i++)
	{
		WorkedPeriod[i] = (int *) malloc((Periods+1)*sizeof(int));
	}

	ScheduledPeriod = (int **) malloc((Machines+1)*sizeof(int));
	for(i=0; i<=Machines; i++)
	{
		ScheduledPeriod[i] = (int *)
malloc((Periods+1)*sizeof(int));
	}


	TempPeriod = (int **) malloc((Machines+1)*sizeof(int));
	for(i=0; i<=Machines; i++)
	{
		TempPeriod[i] = (int *) malloc((Periods+1)*sizeof(int));
	}



	UsedPeriod = (int *) malloc((Machines+1)*sizeof(int));
}


/*******************************************/
/* Function : FreeMemory()
/* - Free Memory.
/*
/*******************************************/
void FreeMemory()
{
	int i;

	free(MachineofJob);
	free(durationofJob);
	free(ElectricPowerofMachine);
```

```
	for(i=0; i<=Machines; i++)
	{
		free(BestAllocatedPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(BestWorkedPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(BestScheduledPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(AllocatedPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(WorkedPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(ScheduledPeriod[i]);
	}

	for(i=0; i<=Machines; i++)
	{
		free(TempPeriod[i]);
	}


	free(UsedPeriod);
}
```

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ประวัติผู้เขียนวิทยานิพนธ์

นางสาวสุนิสา กุลพนาภินันท์ เกิดเมื่อวันที่ 26 มกราคม พ.ศ.2524 บ้านเกิดอยู่ที่อำเภอละแม จังหวัดชุมพร สำเร็จการการศึกษาระดับปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมอาหาร คณะวิศวกรรมศาสตร์ จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง เมื่อพ.ศ.2546 จากนั้นในปี พ.ศ. 2547 ได้เข้าศึกษาต่อระดับบัณฑิตศึกษา วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย