การจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิง โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

UNIT COMMITMENT WITH MULTIPLE FUELS CONSIDERATION BY LAGRANGAIN RELAXATION COMBINED WITH DIFFERENTIAL EVOLUTION ALGORITHM

นางสาววนัชพร ชมปรารภ

นางสาวสุทัตตา สันตยากร

โครงงานวิศวกรรมนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ

ปีการศึกษา 2559

การจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิง โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

UNIT COMMITMENT WITH MULTIPLE FUELS CONSIDERATION BY LAGRANGAIN RELAXATION COMBINED WITH DIFFERENTIAL EVOLUTION ALGORITHM

นางสาววนัชพร ชมปรารภ
นางสาวสุทัตตา สันตยากร

โครงงานวิศวกรรมนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2559

โครงงานวิศวกรรม

เรื่อง

การจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิง โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

ของ

นางสาววนัชพร ชมปรารภ

นางสาวสุทัตตา สันตยากร

ได้รับอนุมัติจากคณะวิศวกรรมศาสตร์ให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.เวคิน ปิยรัตน์)

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.ปฐมทัศน์ จิระเดชะ)

.....กรรมการ

(อาจารย์ ดร.คมกฤษ ประเสริฐวงษ์)

.....กรรมการ

(อาจารย์ ดร.ธนาธิป สุ่มอิ่ม)

การจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิง โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม ปีการศึกษา 2559

โดย

นางสาววนัชพร ชมปรารภ

นางสาวสุทัตตา สันตยากร

อาจารย์ที่ปรึกษา

อาจารย์ ดร.ธนาธิป สุ่มอิ่ม

บทคัดย่อ

โครงงานวิศวกรรมนี้มีวัตถุประสงค์เพื่อแก้ไขปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม โดยสร้างเป็นหน้าต่างโปรแกรมเพื่อให้สะดวกต่อการใช้งาน และเพื่อให้ได้ประสิทธิภาพสูงสุด สำหรับปัญหาการวางแผนการเดินเครื่องกำเนิดไฟฟ้านี้ต้องพิจารณาเงื่อนไขและข้อจำกัดต่างๆ ได้แก่ กำลังไฟฟ้าสมดุล (Power Balance) กำลังการผลิตสำรอง (Spinning Reserve) ข้อจำกัดของเครื่องกำเนิดไฟฟ้า (Generator Constraints) เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Up Time) และเวลาหยุดเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Down Time) การสูญเสียในสายส่ง (Transmission Line Loss) และความหลากหลายทางเชื้อเพลิง (Multiple Fuels) สำหรับระบบไฟฟ้าในโครงงานวิศวกรรมนี้มีเครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต ระยะเวลาการวางแผนการผลิตไฟฟ้าจำนวน 24 ชั่วโมง จะถูกนำมาทดสอบโดยใช้กับโปรแกรมสำหรับการแก้ไขปัญหาการจัดการเดินเครื่องกำเนิดไฟฟ้าที่สร้างขึ้น ซึ่งคำนึงถึงต้นทุนการผลิตโดยนำผลลัพธ์ที่ได้ระหว่างค่าใช้จ่ายในการผลิตแบบเชื้อเพลิงเดียวเปรียบเทียบกับค่าใช้จ่ายในการผลิตแบบหลายเชื้อเพลิง จากผลการทดสอบ สรุปได้ว่าค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงมีค่าใช้จ่ายต่ำกว่าค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าแบบเชื้อเพลิงเดียว

คำสำคัญ: ยูนิตคอมมิตเมนท์ วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม ความหลากหลายทางเชื้อเพลิง

UNIT COMMITMENT WITH MULTIPLE FUELS CONSIDERATION BY LAGRANGAIN RELAXATION COMBINED WITH DIFFERENTIAL EVOLUTION ALGORITHM

Academic Year 2016

By

Ms. Wanatchaporn Chomprarop

Ms. Suthatta Santayakorn

Advisor

Thanathip Sum-Im,Ph.D.

Abstract

The purpose of this engineering project is a unit commitment problem solving by Lagrangian Relaxation (LR) combined with a differential evolution algorithm to meet the required power demand of load and the most effective, which such considering constraints consist of power balance, spinning reserve, generator limits, minimum up time, minimum down time, transmission line losses, and multiple fuels. Programming display with graphical user interface (GUI) is applied to use easily. The project test system is a 10-units generator, 24 hours for unit commitment problem, which takes into account production cost. Two production costs of each generator, which are total costs of multiple fuels and single fuel, are compared and shown in this thesis. From experimental results, the production cost of generator with multiple fuels consideration is cheaper than production cost of generator with single fuel consideration.

Keywords: Unit Commitment, Lagrange Relaxation combined With Differential Evolution Algorithm, Multiple Fuels

กิตติกรรมประกาศ

คณะผู้จัดทำขอกราบขอบพระคุณ ดร.ธนาธิป สุ่มอิ่ม อาจารย์ที่ปรึกษาโครงงานวิศวกรรมและกรรมการสอบโครงงานวิศวกรรม ผู้ให้คำปรึกษา คำแนะนำ ในการค้นคว้าในงานวิจัยในด้านต่างๆ ตลอดจนตรวจแก้ไขปริญญานิพนธ์จนกระทั่งถูกต้องและเสร็จสมบูรณ์

ขอกราบขอบพระคุณอาจารย์ภาควิชาวิศวกรรมไฟฟ้าทุกท่านที่ได้อบรมสั่งสอน และให้ความช่วยเหลือต่างๆ ให้กับคณะผู้จัดทำโครงงานมาโดยตลอด คณะผู้จัดทำโครงงานกราบขอบพระคุณเป็นอย่างสูง

สุดท้ายนี้ ขอกราบขอบพระคุณคุณพ่อ คุณแม่ที่ได้อบรมสั่งสอน คอยให้กำลังใจ และสนับสนุนในทุกๆ เรื่อง และขอขอบคุณทุกๆ ท่านที่มีส่วนเกี่ยวข้องกับโครงงานในครั้งนี้

คณะผู้จัดทำโครงงาน

สารบัญ

	หน้า
บทคัดย่อภาษาไทย	ก
บทคัดย่อภาษาอังกฤษ	ข
กิตติกรรมประกาศ	ค
สารบัญ	ง
สารบัญตาราง	ฉ
สารบัญรูป	ช
รายการสัญลักษณ์	ซ
ประมวลคำย่อ	ฌ
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญของโครงงาน	1
1.2 วัตถุประสงค์ของโครงงาน	2
1.3 ขอบเขตของโครงงาน	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ	3
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง	4
2.1 ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment: UC)	4
2.1.1 ฟังก์ชันวัตถุประสงค์ (Objective Function)	4
2.1.2 เงื่อนไขของระบบ	5
2.2 ปัญหาการจ่ายโหลดแบบประหยัด (Economic Dispatch Problem)	7
2.3 วิธีการแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า	10
2.3.1 Enumeration Method (EM)	10
2.3.2 Priority List (PL)	10
2.3.3 Dynamic Programming (DP)	11
2.3.4 Linear Programming (LP)	11
2.3.5 Ant Colony System (ACS)	12
2.3.6 Mixed Integer Programming (MIP)	12
2.3.7 วิธีลากรางจ์รีแล็กเซชัน (Lagrange Relaxation: LR)	13

สารบัญ(ต่อ)

	หน้า
2.3.8 วิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (Differential Evolution Algorithm)	23
2.4 การประยุกต์ใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชัน	32
2.5 ทบทวนงานวิจัยที่เกี่ยวข้องกับโครงงาน	34
บทที่ 3 ขั้นตอนและวิธีการดำเนินงาน	36
3.1 ขั้นตอนการดำเนินงานโครงการ	36
3.2 ขั้นตอนและวิธีการดำเนินงาน ออกแบบโปรแกรมคอมพิวเตอร์	37
3.3 กำหนดปัญหา Unit Commitment	39
3.4 วิธีการใช้หน้าต่างของโปรแกรม	39
3.5 แผนผังการดำเนินงานโครงการ	41
3.6 รายละเอียดของเครื่องคำนวณที่ใช้	41
บทที่ 4 ผลการทดลอง	42
4.1 ผลการทดลองในการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า	42
4.2 วิเคราะห์ผลการทดลอง	65
บทที่ 5 สรุปผลการทดลองและข้อเสนอแนะ	66
5.1 สรุปผลการทดลอง	66
5.2 ข้อเสนอแนะ	66
เอกสารอ้างอิง	67
ภาคผนวก	68
ประวัติย่อผู้ทำโครงงาน	142

สารบัญตาราง

ตารางที่ **หน้า**

2.1 แสดงค่าพารามิเตอร์ที่ใช้ในการควบคุม 28

2.2 แสดงค่าที่ได้จากการ Initialize ประชาการ P 28

2.3 แสดงการเลือกฟังก์ชั่นเป้าหมายและ random vector 29

2.4 แสดงผลการสร้าง mutant vector และกระบวนการ mutation 29

2.5 แสดงกระบวนการ crossover 30

2.6 แสดงประชากรในรุ่นที่ 2 31

3.1 ตารางแสดงระยะเวลาดำเนินงาน 41

3.2 คุณลักษณะคอมพิวเตอร์ที่ใช้ในโครงงาน 41

4.1 แสดงผลของค่าใช้จ่ายในแต่ละรอบการคำนวณสำหรับการเดินเครื่องกำเนิดไฟฟ้า จำนวน 100 รอบ 44

4.2 แสดงผลของสถานะของเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมง 49

4.3 สรุปค่ากำลังไฟฟ้า ราคาค่าใช้จ่ายสำหรับระบบที่มีการเดินเครื่องกำเนิดไฟฟ้า 10 ยูนิต และมีการพิจารณาความสูญเสียที่สายส่ง ในรอบการคำนวณที่ 88 ให้ค่าที่ดีที่สุด จากการคำนวณทั้งหมด 100 รอบแบบเชื้อเพลิงเดียว 50

4.4 แสดงราคาค่าใช้จ่ายในการผลิต ($) สำหรับการเดินเครื่องเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมงแบบเชื้อเพลิงเดียวของรอบที่ดีที่สุด 53

4.5 แสดงราคาค่าใช้จ่ายในการผลิต ($) สำหรับการเดินเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมงแบบหลายเชื้อเพลิงของรอบที่ดีที่สุด 54

4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด 55

4.7 สรุปค่ากำลังไฟฟ้า ราคาค่าใช้จ่ายสำหรับระบบที่มีการเดินเครื่องกำเนิดไฟฟ้า 10 ยูนิต และมีการพิจารณาความสูญเสียที่สายส่ง ในรอบการคำนวณที่ 88 ให้ค่าที่ดีที่สุด จากการคำนวณทั้งหมด 100 รอบแบบหลายเชื้อเพลิง 61

4.8 สรุปผลการคำนวณสำหรับการเดินเครื่องกำเนิดไฟฟ้าที่ค่าใช้จ่ายในการผลิตต่ำที่สุด 63

4.9 แสดงค่าทางสถิติในการผลิตสำหรับการเดินเครื่องกำเนิดไฟฟ้าเป็นเวลา 24 ชั่วโมง ในการคำนวณ 100 รอบ 64

4.10 ตารางเปรียบเทียบของการแก้ไขปัญหาระหว่างการพิจารณาการเลือกใช้เชื้อเพลิงเดียวกับการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด 64

สารบัญรูป

รูปที่ **หน้า**

2.1 ขั้นตอนการแก้ไขปัญหายูนิตคอมมิตเมนต์โดยใช้วิธีลากรางจ์รีแล็กเซชัน 16

2.2 แผนผังแสดงการทำงาน Differential Evolution Algorithm (DEA) 27

2.3 การแก้ปัญหายูนิตคอมมิตเมนต์โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับ วิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม 33

3.1 แผนผังแสดงขั้นตอนการทำงาน 36

3.2 แผนผังหลักการคิดโปรแกรม 37

3.3 หน้าต่างโปรแกรมสำหรับการแก้ไขปัญหา 39

4.1 แสดงหน้าต่างการทำงานของโปรแกรมการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า 43

4.2 กราฟการลู่เข้าหาคำตอบของราคาค่าใช้จ่ายในการผลิตในการแก้ปัญหาการจัดตาราง การเดินเครื่องกำเนิดไฟฟ้า 63

4.3 กราฟการลู่เข้าหาคำตอบของค่าช่องว่าง (Relative Duality Gap) ในการแก้ปัญหา การจัดตารางการเดินเครื่องกำเนิดไฟฟ้า 64

รายการสัญลักษณ์

สัญลักษณ์	คำอธิบาย	หน่วย
$F_i(P_{it})$	ฟังก์ชันต้นทุนเชื้อเพลิงของเครื่องกำเนิดไฟฟ้า i ขณะผลิตกำลังไฟฟkขนาด P ที่เวลา t	-
$GOFFT_{i(t-1)}$	ระยะเวลาที่เครื่องกำเนิดไฟฟ้า i หยุดเดินเครื่องเป็นเวลาติดต่อกัน $(t-1)$	-
$GONT_{i(t-1)}$	จำนวนชั่วโมงที่เครื่องกำเนิดไฟฟ้า i เดินเครื่องเป็นเวลาติดต่อกัน $(t-1)$	-
MDU_i	เวลาหยุดเดินเครื่องอย่างน้อยที่สุดของเครื่องกำเนิดไฟฟ้า i	-
MTU_i	เวลาเดินเครื่องอย่างน้อยที่สุดของเครื่องกำเนิดไฟฟ้า i	-
N	จำนวนเครื่องกำเนิดไฟฟ้า	-
P_{Dt}	ค่ากำลังไฟฟ้าที่โหลดต้องการ ณ เวลาที่ t	เมกกะวัตต์
P_i	กำลังไฟฟ้าของเครื่องกำเนิดไฟฟ้า i	-
P_{it}	กำลังไฟฟ้าที่ผลิตจากเครื่องกำเนิดไฟฟ้า i ณ เวลาที่ t	เมกกะวัตต์
$P_{i,\max}$	ค่ากำลังไฟฟ้าสูงสุดที่เครื่องกำเนิดไฟฟ้า i สามารถผลิตได้	เมกกะวัตต์
$P_{i,\max}$	กำลังผลิตต่ำสุดของเครื่องกำเนิดไฟฟ้า i	เมกกะวัตต์
P_{load}	กำลังไฟฟ้าที่โหลดต้องการ	เมกกะวัตต์
P_{loss}	กำลังสูญเสียในสายส่ง	เมกกะวัตต์
P_{total}	ต้นทุนรวมในการผลิตกำลังไฟฟ้า	ดอลลาร์/ชั่วโมง
R_t	ค่ากำลังไฟฟ้าสำรอง ณ เวลาที่ t	เมกกะวัตต์
ST_{it}	ต้นทุนการเริ่มเดินเครื่องกำเนิดไฟฟ้า i	ดอลลาร์
T	จำนวนชั่วโมงที่พิจารณา	-
U_{it}	สถานะของเครื่องกำเนิดไฟฟ้า i ณ เวลาที่ t เมื่อ $U_{it}=0$ แทนสถานะหยุดเดินเครื่องกำเนิดไฟฟ้า และ $U_{it}=1$ แทนสถานะเดินเครื่องกำเนิดไฟฟ้า	-

ประมวลคำย่อ

คำย่อ	คำอธิบาย
ACSA	Ant Colony Search Algorithm
CGA	Conventional Genetic Algorithm
CSC	Cold-Start Cost
DEA	Differential Evolution Algorithm
DP	Dynamic Programming
EAs	Evolutionary Algorithms
ED	Economic Dispatch
EP	Evolutionary Programming
ESs	Evolution Strategies
GAs	Genetic Algorithms
GENCOs	Generation Companies
HSC	Hot-Start Cost
IP	Integer Programming
LR	Lagrangian Relaxation
LRGA	Lagrangian Relaxation and Genetic Algorithm
LR-DEA	Lagrangian Relaxation with a Differential Evolution Algorithm
MDT	Minimum Down Time
MUT	Minimum Up Time
SaDEA	Self-Adaptive Differential Evolution Algorithm
SD	Standard Deviation
UC	Unit Commitment

บทที่1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ในปัจจุบันพลังงานไฟฟ้านับเป็นสิ่งจำเป็น เปรียบเสมือนปัจจัยที่ 5 ในการดำรงชีวิตของมนุษย์ ไม่ว่าจะเป็นการสื่อสาร การคมนาคม ระบบส่องสว่าง ด้านความร้อน ด้านพลังงาน ด้านเสียง เป็นต้น จะเห็นได้ว่ามนุษย์มีการใช้ปริมาณไฟฟ้าที่สูงและมีอัตราการใช้ปริมาณไฟฟ้าที่สูงขึ้นเรื่อยๆ เฉลี่ยปีละ 4-5% จนถึงปี พ.ศ.2573 อันเนื่องมาจากการขยายตัวของจำนวนประชากรที่เพิ่มขึ้นในปัจจุบัน และระบบเศรษฐกิจที่ขยายตัวตลอดเวลา ซึ่งปริมาณความต้องการใช้พลังงานไฟฟ้าในแต่ช่วงเวลาจะมีปริมาณที่ไม่เท่ากัน ขึ้นอยู่กับการใช้งานของแต่ละภาคส่วน หรือความต้องการของโหลด ดังนั้นในการผลิตไฟฟ้าจึงจำเป็นต้องวางแผนการผลิตกำลังไฟฟ้าให้มีปริมาณที่เพียงพอต่อความต้องการของผู้บริโภค โดยใช้ต้นทุนในการผลิตไฟฟ้าต่ำสุด และการส่งจ่ายที่ใช้ในการผลิตไฟฟ้าให้คุ้มค่าในทางเศรษฐศาสตร์ จึงเรียกกระบวนการนี้ว่า การจัดตารางการเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment)

การจัดตารางการเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment) มีจุดมุ่งหมายในการบริหารการผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลดในแต่ละช่วงเวลาการวางแผนเดินเครื่องกำเนิดไฟฟ้าล่วงหน้าเพื่อให้เพียงพอต่อความต้องการและเกิดประสิทธิภาพสูงที่สุดโดยมีเงื่อนไขที่พิจารณาในส่วนของคุณสมบัติของเครื่องกำเนิดไฟฟ้าและระบบไฟฟ้า เช่น กำลังไฟฟ้าสมดุล (Power Balance) ข้อจำกัดของเครื่องกำเนิดไฟฟ้า (Generator Constraints) เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Up Time) เวลาหยุดเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Down Time) กำลังไฟฟ้าสำรอง (Spinning Reserve) และการสูญเสียในสายส่ง (Transmission Line Losses) อีกทั้งยังมีเงื่อนไขที่มีความสำคัญต่อการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า นั่นคือความหลากหลายทางเชื้อเพลิง (Multiple Fuels) เนื่องจาก อัตราค่าเชื้อเพลิงมีค่าไม่เท่ากัน จึงทำให้เครื่องกำเนิดไฟฟ้าต้องเลือกใช้เชื้อเพลิงให้มีต้นทุนต่ำ และมีประสิทธิภาพมากที่สุด

โครงงานวิศวกรรมฉบับนี้จะนำเสนอการออกแบบการสร้างโปรแกรมเพื่อแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าที่มีความหลากหลายเชื้อเพลิง โดยวิธีลากรางจ์รีแลกเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม เพื่อให้คุ้มค่าในทางด้านเศรษฐศาสตร์และสามารถผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลดได้

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาทฤษฎีการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment)

1.2.2 เพื่อศึกษาการทำงานของวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม เพื่อนำมาประยุกต์ใช้งานทางวิศวกรรม

1.2.3 เพื่อศึกษาการทำงานของวิธีลากรางจ์รีแลกเซชันเพื่อนำมาประยุกต์ใช้งานทางวิศวกรรม

1.2.4 สามารถคำนวณและแก้ไขปัญหาการจัดตารางการผลิตไฟฟ้าของเครื่องกำเนิดไฟฟ้าได้

1.2.5 เพื่อศึกษาและออกแบบการสร้าง Graphical User Interface (GUI) ใน MATLAB ที่จะสามารถนำมาใช้ในการคำนวณเพื่อแก้ไขปัญหาและคำนวณการจัดตารางการผลิตไฟฟ้าของเครื่องกำเนิดไฟฟ้าให้ได้ตามความเหมาะสม

1.2.6 สามารถจัดตารางการเดินเครื่องกำเนิดไฟฟ้าที่มีความหลากหลายเชื้อเพลิงให้คุ้มค่าในทางด้านเศรษฐศาสตร์และสามารถผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลด

1.3 ขอบเขตของโครงงาน

ออกแบบและสร้างโปรแกรมสำหรับการแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าโดยนำเอาวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม มาประยุกต์ใช้ มีคุณสมบัติดังนี้

1.3.1 ออกแบบ และทดสอบโดยใช้โปรแกรม MATLAB

1.3.2 สร้างโปรแกรมที่สามารถแก้ไขปัญหาการจัดตารางการผลิตไฟฟ้าของโรงไฟฟ้า โดยพิจารณาตามเงื่อนไขและข้อจำกัดดังนี้

1.3.2.1 กำลังไฟฟ้าสมดุล (Power Balance)

1.3.2.2 กำลังการผลิตสำรอง (Spinning Reserve)

1.3.2.3 ข้อจำกัดเครื่องกำเนิดไฟฟ้า (Generator Constraints)

1.3.2.4 เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Up Time)

1.3.2.5 เวลาหยุดเดินเครื่องอย่างน้อยที่สุด (Minimum Down Time)

1.3.2.6 การสูญเสียในสายส่ง (Transmission Line Losses)

1.3.2.7 ความหลากหลายทางเชื้อเพลิง (Multiple Fuels)

1.3.3 ทดสอบโปรแกรมกับเครื่องกำเนิดไฟฟ้า 10 ยูนิต ระยะเวลาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า 24 ชั่วโมง

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 สามารถประยุกต์ใช้โปรแกรมคอมพิวเตอร์สำหรับการแก้ปัญหาการจัดตารางการผลิตไฟฟ้าของเครื่องกำเนิดไฟฟ้า

1.4.2 สามารถแก้ไขปัญหาการจัดตารางการเดินเครื่องไฟฟ้าของเครื่องกำเนิดไฟฟ้าให้มีความเหมาะสมด้านความคุ้มค่าทางด้านเศรษฐศาสตร์

1.4.3 ได้ศึกษาและเข้าใจการเขียน Graphical User Interface (GUI) ใน MATLAB และสามารถประยุกต์ใช้งานในงานวิศวกรรมด้านอื่นๆ ได้

1.4.4 โครงงานนี้สามารถใช้เป็นแนวทางในการแก้ปัญหาและวางแผนการผลิตกำลังไฟฟ้าได้จริง

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment : UC)

การจัดตารางเดินเครื่องเครื่องกำเนิดไฟฟ้าแบบดั้งเดิมจะเป็นการหาค่าต่ำสุดของต้นทุนการผลิตไฟฟ้า โดยจะทำการกำหนดสถานะของเครื่องกำเนิดไฟฟ้าว่าให้มีสถานะเดินเครื่องหรือหยุดเดินเครื่องกำเนิดไฟฟ้าให้มีประสิทธิภาพสูงสุดเพื่อให้เพียงพอต่อความต้องการของโหลด โดยมีเงื่อนไขและข้อจำกัดของเครื่องกำเนิดไฟฟ้า และจ่ายกำลังไฟฟ้าให้สอดคล้องกับความต้องการที่การพยากรณ์ไว้ทุกชั่วโมง โดยปัญหาการวางแผนเดินเครื่องกำเนิดไฟฟ้าจะต้องพิจารณาข้อจำกัดและเงื่อนไขต่างๆ ได้แก่ กำลังไฟฟ้าสมดุล (Power Balance) กำลังการผลิตสำรอง (Spinning Reserve) ข้อจำกัดของเครื่องกำเนิดไฟฟ้า (Generator Constraints) เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Up Time) เวลาหยุดเดินเครื่องอย่างน้อยที่สุด (Minimum Down Time) และการสูญเสียในสายส่ง (Transmission Line Losses) เป็นต้น ซึ่งการแก้ไขปัญหาแบบนี้เราเรียกว่า การแก้ปัญหาการจ่ายโหลดแบบประหยัด (Economic Dispatch: ED)

2.1.1 ฟังก์ชันวัตถุประสงค์ (Objective Function)

การแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าที่อยู่ในระบบไฟฟ้าทั้งหมด ว่าจะเดินเครื่องกำเนิดไฟฟ้าเครื่องใดเข้ากับระบบหรือปลดออกจากระบบไฟฟ้า เพื่อผลิตกำลังไฟฟ้าให้สอดคล้องกับเงื่อนไขของระบบไฟฟ้าและเครื่องกำเนิดไฟฟ้า และจะต้องคำนึงถึงต้นทุนการผลิตไฟฟ้ารวมทั้งหมดให้มีค่าต่ำที่สุด สามารถเขียนเป็นสมการได้ดังนี้

$$F_{Total\,cost} = \sum_{t=1}^{T}\sum_{i=1}^{N}[F_i(P_{it}) + (1-U_{t(t-1)})ST_i]U_{it} \qquad (2.1)$$

เมื่อ T คือ จำนวนชั่วโมงที่พิจารณา

N คือ จำนวนเครื่องกำเนิดไฟฟ้า

$F_i(P_{it})$ คือ ฟังก์ชันต้นทุนเชื้อเพลิงของเครื่องกำเนิดไฟฟ้า i ขณะผลิตกำลังไฟฟ้าขนาด P ที่เวลา t

P_{it} คือ กำลังไฟฟ้าที่ผลิตจากเครื่องกำเนิดไฟฟ้า i ณ เวลาที่ t

ST_{it} คือ ต้นทุนการเริ่มเดินเครื่องกำเนิดไฟฟ้า i

เมื่อ $U_{it} = 0$ แทนสถานะหยุดเดินเครื่องกำเนิดไฟฟ้า และ

$U_{it} = 1$ แทนสถานะเดินเครื่องกำเนิดไฟฟ้า

2.1.2 เงื่อนไขของระบบ

การจัดตารางเดินเครื่องกำเนิดไฟฟ้า คือ การทำให้ต้นทุนการผลิตไฟฟ้ารวมทั้งหมดมีค่าต่ำที่สุด และจะต้องคำนึงถึงเงื่อนไขและข้อจำกัดที่ซับซ้อนของระบบไฟฟ้าและเครื่องกำเนิดไฟฟ้าด้วย โดยจะพิจารณาเงื่อนไขดังต่อไปนี้

2.1.2.1 กำลังไฟฟ้าสมดุล (Power Balance) เงื่อนไขคือ กำลังผลิตไฟฟ้ารวมของเครื่องกำเนิดไฟฟ้าที่ต่อเข้ากับระบบไฟฟ้า ณ เวลาที่พิจารณาต้องมีค่าเท่ากับกำลังไฟฟ้าที่โหลดต้องการ ณ เวลานั้น

$$\sum_{i=1}^{N} U_{it} P_{it} = P_{Dt} + P_{loss} \qquad (2.2)$$

เมื่อ P_{Dt} คือ กำลังไฟฟ้าที่โหลดต้องการ ณ เวลาที่ t

2.1.2.2 กำลังการผลิตสำรอง (Spinning Reserve) คือ กำลังผลิตไฟฟ้าสำรองมีจุดมุ่งหมายเพื่อเป็นหลักประกันว่าเมื่อเกิดการเปลี่ยนแปลงของระบบไฟฟ้าระบบยังคงจ่ายไฟฟ้าต่อไปได้ซึ่งการเปลี่ยนแปลงมี 2 กรณี คือ

1. การปลดเครื่องกำเนิดไฟฟ้าเครื่องใดเครื่องหนึ่ง เนื่องจากสาเหตุใดก็ตามโหลดในระบบไฟฟ้าจะยังคงได้รับกำลังไฟฟ้าที่เพียงพอกับความต้องการเหมือนเดิม

2. ความต้องการของโหลดมากขึ้น เครื่องกำเนิดไฟฟ้าที่ต่ออยู่กับระบบไฟฟ้าขณะนั้นต้องสามารถเพิ่มกำลังการผลิตไฟฟ้าได้เพียงพอกับความต้องการของโหลดที่เพิ่มขึ้น

$$\sum_{i=1}^{N} U_{it} P_{i,\max} + P_{loss} \geq R_t + P_{Dt} \qquad (2.3)$$

2.1.2.3 ข้อจำกัดเครื่องกำเนิดไฟฟ้า (Generator Constraints) คือ เครื่องกำเนิดไฟฟ้าแต่ละเครื่องจะมีข้อจำกัดในการผลิตกำลังไฟฟ้าว่าเครื่องกำเนิดไฟฟ้านั้นสามารถผลิตกำลังไฟฟ้าสูงที่สุดและต่ำที่สุดได้เท่าใด ซึ่งการผลิตกำลังไฟฟ้าในแต่ละช่วงเวลานั้นต้องคำนึงถึงข้อจำกัดของการผลิตกำลังไฟฟ้าของเครื่องกำเนิดไฟฟ้าแต่ละเครื่องด้วย

$$P_{i,\min} \leq P_i \leq P_{i,\max} \qquad (2.4)$$

เมื่อ $P_{i,\min}$ คือ กำลังผลิตต่ำสุดของเครื่องกำเนิดไฟฟ้า i

P_i คือ กำลังไฟฟ้าที่ผลิตออกมาของเครื่องกำเนิดไฟฟ้า i

$P_{i,\max}$ คือ กำลังไฟฟ้าสูงสุดที่เครื่องกำเนิดไฟฟ้า i ที่สามารถผลิตได้

เครื่องกำเนิดไฟฟ้าที่เดินเครื่องเพื่อผลิตกำลังไฟฟ้านั้น จะไม่สามารถสั่งหยุดเดินเครื่องได้ทันที จะต้องใช้ระยะช่วงเวลาหนึ่ง ซึ่งปกติอาจใช้เวลาหลายชั่วโมง และระยะเวลาของแต่ละเครื่องมีค่าไม่เท่ากัน ก็คือ เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Up Time : T_i^{up}) และ เวลาหยุดเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด (Minimum Down Time : T_i^{down})

Minimum Up Time: T_i^{up}

$$U_{it} = 1 \; for \sum_{h=t-T_i^{up}}^{t-1} U_{it} < T_i^{up} \qquad (2.5)$$

Minimum Down Time: T_i^{down}

$$U_{it} = 0 \; for \sum_{h=t-T_i^{down}}^{t-1} (1-U_{it}) < T_i^{down} \tag{2.6}$$

2.1.2.4 การสูญเสียในสายส่ง (Transmission Line Losses)

ในการส่งกำลังไฟฟ้าจากโรงไฟฟ้าแต่ละแห่งไปยังโหลดผู้ใช้ไฟฟ้านั้นอาจมีระยะทางระหว่างหน่วยผลิตและโหลดที่ไม่เท่ากัน การส่งกำลังไฟฟ้าจากหน่วยผลิตซึ่งอยู่ไกลออกไปจากโหลดนั้นย่อมทำให้เกิดการสูญเสียในรูปของความร้อนในสายส่ง ดังนั้นการวิเคราะห์การจัดตารางการเดินเครื่องกำเนิดไฟฟ้า (Unit Commitment: UC) โดยพิจารณาการสูญเสียกำลังไฟฟ้าในสายส่ง เขียนได้สมการดังนี้ (Hadi Saadat. 2004)

$$P_{loss} = \sum_{i=1}^{N} \sum_{j=1}^{N} P_i B_{ij} P_j + \sum_{i=1}^{N} B_{i0} P_i + B_{00} \tag{2.7}$$

เมื่อ B คือ ค่าสัมประสิทธิ์การสูญเสีย

N คือ จำนวนเครื่องกำเนิดไฟฟ้า

2.2 ปัญหาการจ่ายโหลดแบบประหยัด (Economic Dispatch Problem)

เนื่องจากโรงไฟฟ้ามีหลายประเภท มีการใช้เชื้อเพลิงแตกต่างกัน ต้นทุนค่าเชื้อเพลิงต่างกัน และขึ้นอยู่กับประสิทธิภาพของการเดินเครื่อง จึงทำให้ต้นทุนการผลิตไฟฟ้าของเครื่องกำเนิดไฟฟ้าแต่ละตัวไม่เท่ากัน ปัญหาการจ่ายโหลดแบบประหยัด (Economic Dispatch Problem) มีวัตถุประสงค์ในการแก้ปัญหาคือ การทำให้ต้นทุนการผลิตกำลังไฟฟ้ามีค่าน้อยที่สุด และจ่ายกำลังไฟฟ้าให้กับความต้องการของโหลดมีประสิทธิภาพสูงที่สุด หรือการผลิตไฟฟ้าโดยใช้เชื้อเพลิงในการผลิตน้อยที่สุด โดยรูปแบบทางคณิตศาสตร์ของปัญหาการจ่ายโหลดอย่างประหยัดเขียนเป็นสมการได้ดังนี้

$$F_T = \sum_{i=1}^{N} F_i(P_i) \tag{2.8}$$

เมื่อ F_T คือ ต้นทุนรวมในการผลิตกำลังไฟฟ้า

F_i คือ ราคาเชื้อเพลิงของเครื่องกำเนิดไฟฟ้าที่ i

P_i คือ กำลังไฟฟ้าของเครื่องกำเนิดไฟฟ้า i

เมื่อ i แทนเครื่องกำเนิดไฟฟ้าเครื่องที่ i และ N แทนจำนวนเครื่องกำเนิดไฟฟ้าที่เชื่อมต่ออยู่ ในระบบไฟฟ้าทั้งหมด โดยที่สมการ (2.1) แทนฟังก์ชันเป้าหมาย (Objective Function) ที่ต้องการหาค่าต้นทุนในการผลิตกำลังไฟฟ้ารวมน้อยที่สุด ซึ่งจะขึ้นอยู่กับข้อจำกัดการทำงานของแต่ละเครื่อง สามารถเขียนเป็นสมการได้ดังนี้ (สร้อยพลู วรภพ. 2554)

$$\sum_{i=1}^{N} (P_i) = P_{load} + P_{loss} \tag{2.9}$$

เมื่อ P_{load} คือ กำลังไฟฟ้าที่โหลดต้องการ

P_{loss} คือ กำลังสูญเสียในสายส่ง

โดยฟังก์ชันราคาเชื้อเพลิงของเครื่องกำเนิดไฟฟ้าที่ใช้ในปัญหาการจ่ายโหลดแบบประหยัดจะอยู่ในรูปสมการกำลังสอง (Quadratic Function) ดังนี้

$$F_i(P_i) = a_i + b_i P_i + c_i P_i^2 \tag{2.10}$$

เมื่อ a_i, b_i, c_i คือ สัมประสิทธิ์ของฟังก์ชันอัตราความร้อนของเครื่องกำเนิดไฟฟ้าเครื่องที่ i

สำหรับโรงไฟฟ้าหนึ่งโรงในแต่ละโรงจะประกอบไปด้วยเครื่องกำเนิดไฟฟ้าได้หลายหน่วย โดยแต่ละหน่วยนั้นจะมีการใช้เชื้อเพลิงที่แตกต่างกันซึ่งในการวิเคราะห์การจ่ายกำลังไฟฟ้าให้โหลดอย่างประหยัดจากเครื่องกำเนิดไฟฟ้าจำนวน n เครื่อง โดยวิธีการทางคณิตศาสตร์เชิงวิเคราะห์ จะได้ราคาเชื้อเพลิงรวมของระบบเท่ากับ

$$F_T = F_1 + F_2 + \ldots + F_n = \sum_{i=1}^{n} F_i \tag{2.11}$$

ปัญหาการจ่ายโหลดอย่างประหยัดจึงเป็นปัญหาการหาค่าที่เหมาะสม(Optimization Problem) โดยมีฟังก์ชันของราคาค่าเชื้อเพลิงเป็นฟังก์ชันวัตถุประสงค์ (Objective Function) คือ

$$Minimize \;\; F_T = \sum_{i=1}^{n} F_i(P_i) \tag{2.12}$$

ในเครื่องกำเนิดไฟฟ้าบางชนิดมีการทำงานโดยใช้เชื้อเพลิงที่แตกต่างกันได้หลายชนิด (Multiple Fuels) การเปลี่ยนแปลงชนิดของเชื้อเพลิงจึงมีผลต่อฟังก์ชันราคาต้นทุน (Cost Function) จาก Quadratic Function จะเป็น Piecewise Quadratic Function แสดงสมการได้ดังนี้

$$F_i(P_i)\begin{cases} a_{i1} + b_{i1}P_{i1} + c_{i1}P_{i1}^2;\ fule1, P_i^{\min} \le P_i \le P_{i1} \\ a_{i2} + b_{i2}P_{i2} + c_{i2}P_{i2}^2;\ fule2, P_{i1}^{\min} \le P_i \le P_{i2} \\ \vdots \\ a_{ik} + b_{ik}P_{ik} + c_{ik}P_{ik}^2;\ fulek, P_{ik-1}^{\min} \le P_i \le P_i^{\max} \end{cases} \tag{2.13}$$

โดยที่ F_T คือ ราคาเชื้อเพลิงรวม ($/hr)

F_i คือ กำลังขาออกของเครื่องกำเนิดไฟฟ้า (MW)

P_d คือ กำลังไฟฟ้าที่โหลดต้องการใช้ (MW)

P_L คือ กำลังไฟฟ้าสูญเสียที่เกิดขึ้นในสายส่ง (MW)

$P_{\min}$ คือ กำลังขาออกที่น้อยที่สุดของเครื่องกำเนิดไฟฟ้าที่ i (MW)

$P_{\max}$ คือ กำลังขาออกที่มากที่สุดของหน่วยผลิตที่ i (MW)

n คือ จำนวนเครื่องกำเนิดไฟฟ้าที่ทำงาน

2.3 วิธีการแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า

การแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้านั้นมีหลากหลายวิธี เช่น Enumeration Method (EM), Priority List (PL), Dynamic Programming (DP), Linear Programming (LP), Ant Colony System (ACS), Mixed Integer Programming (MIP) และ Priority List Scheduling แต่ในโครงงานฉบับนี้จะเป็นการแก้ปัญหาด้วยวิธี Lagrange Relaxation (LR) ร่วมกับวิธี Differential Evolution Algorithm (DEA)

2.3.1 Enumeration Method (EM)

วิธีนี้เป็นวิธีการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าที่ให้ผลลัพธ์ดีที่สุดโดยสำรวจทุกทางเลือกที่เป็นไปได้ แต่จะใช้เวลามากโดยเฉพาะอย่างยิ่งเมื่อจำนวนโรงไฟฟ้ามากขึ้น ถ้าต้องการจัดตารางการเดินเครื่องล่วงหน้า NT ชั่วโมงและมีเครื่องกำเนิดไฟฟ้า NG เครื่อง จำนวนทางเลือกที่เป็นไปได้ทั้งหมดจะเท่ากับ $(2^{NG}-1)^{NT}$ ดังนั้น วิธีนี้จึงไม่เหมาะกับระบบไฟฟ้าขนาดใหญ่ จึงควรใช้เทคนิคในการหาผลลัพธ์ที่เหมาะสมที่สุดในการจัดตารางการเดินเครื่อง

2.3.2 Priority List (PL)

วิธีนี้เป็นวิธีที่เดินเครื่องกำเนิดไฟฟ้าเรียงลำดับตามต้นทุนการผลิตเต็มกำลังเฉลี่ย (Full Load Average Cost, FLAC) ดังสมการ (2.13) คำนวณจากต้นทุนการผลิตเมื่อเครื่องกำเนิดไฟฟ้าผลิตกำลังไฟฟ้าสูงสุดหารด้วยกำลังไฟฟ้าสูงสุดที่ผลิตได้ โดยให้เครื่องกำเนิดไฟฟ้าที่มีต้นทุนการผลิตเต็มกำลังเฉลี่ยต่ำสุดเดินเครื่องก่อน

$$FLAC_i = \frac{F_i(P_{i,\max})}{P_{i,\max}}, t = 1, \ldots, NT \tag{2.14}$$

วิธีเรียงตามลำดับยังเป็นวิธีที่ยังใช้อยู่ในบางประเทศเนื่องจากความเรียบง่ายไม่ซับซ้อน สำหรับการจัดตารางการเดินเครื่องเครื่องกำเนิดไฟฟ้าในประเทศไทยใช้โปรแกรมคอมพิวเตอร์คำนวณเบื้องต้นก่อนแล้วอาศัยประสบการณ์ของเจ้าหน้าที่ฝ่ายควบคุมระบบกำลังไฟฟ้า ปรับตารางเดินเครื่องโดยคำนึงถึงต้นทุนการผลิตไฟฟ้า ทำเลที่ตั้งของโรงไฟฟ้า ความเชื่อถือได้ของระบบส่งและเงื่อนไขอื่นๆได้แก่ ปัญหาด้านมลภาวะที่เกิดขึ้นจากการเดินเครื่องของเครื่องกำเนิดไฟฟ้าต่างๆ ปริมาณเชื้อเพลิงที่ใช้ปริมาณน้ำที่มีใช้ในแต่ละวัน

2.3.3 Dynamic Programming (DP)

วิธีนี้เป็นวิธีค้นหาเส้นทางที่ประกอบด้วยสถานะของโรงไฟฟ้า (เดินเครื่อง-หยุดเครื่อง) ที่ให้ต้นทุนการผลิตต่ำสุด โดยเริ่มจากชั่วโมงแรกไปจนถึงชั่วโมงสุดท้าย วิธีที่นำมาใช้กับการจัดตารางการเดินเครื่องได้อย่างประสิทธิภาพคือ วิธีไดนามิกโปรแกรมมิงแบบก้าวหน้า (Forward Dynamic Programming) โดยเริ่มสำรวจจากสถานะของโรงไฟฟ้าในชั่วโมงเริ่มต้น แล้วค่อยๆก้าวไปทีละชั่วโมงหรือทีละคาบ (อาจเป็นครึ่งชั่วโมงหรือ 1-4 ชั่วโมง) แล้วคำนวณต้นทุนการผลิตและบันทึกค่าไว้ทั้งต้นทุนและตารางเดินเครื่อง แล้วบวกสะสมต้นทุนการผลิตที่เพิ่มขึ้นตามทางเส้นทางที่ก้าวเดิน ซึ่งอาจมีหลายเส้นทาง เมื่อไปถึงปลายทางคือคาบสุดท้าย แต่ละเส้นทางจะมีต้นทุนการผลิตรวมที่แตกต่างกัน ให้เลือกเส้นทางที่มีต้นทุนการผลิตต่ำสุด แล้วให้ติดตามย้อนกลับไปต้นทาง โดยย้อนไปตามทางที่เป็นที่มาของค่าต่ำสุด วิธีนี้ใช้เวลาเพิ่มขึ้นมากเมื่อจำนวนโรงไฟฟ้า

วิธีปลดเครื่องกำเนิดไฟฟ้าส่วนเกิน (Unit Decomitment) เป็นการจัดตารางการเดินเครื่องแบบปลดเครื่องกำเนิดไฟฟ้า (สั่งให้ $'U_i^t = 0'$) ส่วนเกินออกไป โดยเริ่มต้นจัดให้เดินเครื่องกำเนิดไฟฟ้าทุกเครื่องทุกชั่วโมง แล้วค่อยพิจารณาปลดเครื่องกำเนิดไฟฟ้าส่วนเกินออกทีละเครื่อง โดยต้องสอดคล้องกับเงื่อนไขข้อจำกัดเวลาเดินเครื่อง/หยุดเดินเครื่องในสมการ (2.14)

$$U_i^t = \begin{cases} 1, & if \quad T_{i,on}^{t-1} < T_{i,up'} \\ 0, & if \ T_{i,off}^{t-1} < T_{i,down'} \\ 0 or 1, & if \quad otherwise, \end{cases} \tag{2.15}$$

2.3.4 Linear Programming (LP)

วิธีโปรแกรมเชิงเส้น (Linear Programming) ประกอบไปด้วย 2 ส่วนคือ สมการเป้าหมายหรือสมการวัตถุประสงค์ (Objective Function) และ สมการข้อจำกัด (Constraints) ซึ่งจะอยู่ในรูปสมการ

หรืออสมการ โดยที่สมการทั้งหมดเป็นสมการเชิงเส้นเมื่อเทียบกับตัวแปร คำตอบของสมการข้อจำกัดอาจจะมีได้หลายคำตอบซึ่งคำตอบเหล่านี้อยู่ภายใต้ข้อจำกัดที่กำหนด อย่างไรก็ตามสมการเป้าหมายจะเป็นตัววัดผลระหว่างคำตอบทั้งหมดของสมการข้อจำกัดว่าคำตอบใดเป็นคำตอบที่ดีที่สุด ซึ่งเราจะต้องพยายามหาค่าเป็นไปได้โดยอาศัยเทคนิคที่มีอยู่ โดยที่จุดประสงค์ของโปรแกรมเชิงเส้นก็คือหาค่าของตัวแปรเหล่านี้ที่ทำให้สมการกำหนดเป้าหมายมีค่าที่ดีที่สุดหรือเรียกว่าแบบจำลองที่ใช้สำหรับหาค่าสูงสุดหรือต่ำสุด (Maximize/Minimize) ที่สมการความสัมพันธ์ระหว่าง สมการวัตถุประสงค์ (Objective Function) และ สมการข้อจำกัด (Constraints) เป็นเส้นตรง ซึ่งวิธีโปรแกรมเชิงเส้น (Linear Programming) มีข้อดีคือ มีความสัมพันธ์กันในลักษณะเชิงเส้น (Linear) หรืออย่างน้อยก็สามารถอนุโลมเป็นเชิงเส้นได้ โดยที่มีข้อผิดพลาดเล็กน้อยเท่านั้น แต่ก็มีข้อเสียคือ ตัวแปรทุกค่าต้องคงที่แน่นอน รู้ค่าจริงและสมการที่ได้จะให้ผลลัพธ์เท่าเดิมถ้าสัมประสิทธิ์เพิ่มค่าในอัตราส่วนที่เท่ากัน

2.3.5 Ant Colony System (ACS)

วิธีระบบอาณานิคมมด (Ant Colony System) คือวิธีการหาค่าความเหมาะสมของปัญหาที่มีพื้นฐานมาจากการหาแหล่งอาหารของมด ซึ่งมดทุกตัวจะหาเส้นทางจากรังไปยังแหล่งอาหารโดยใช้ระยะทางที่ใกล้ที่สุด เมื่อมดเดินผ่านเส้นทางจะพ่นฟีโรโมน (Pheromone) ลงบนพื้นเพื่อเป็นข้อมูลให้มดตัวอื่นๆ เส้นทางที่สั้นจะมีจำนวนมดเดินผ่านมากกว่าเส้นทางที่ยาวกว่า ดังนั้นเส้นทางที่สั้นกว่าจะมีปริมาณความเข้มข้นของฟีโรโมนมากกว่าเส้นทางที่ยาวกว่า ในการทำงานของ ACS จะกำหนดให้ค่าของฟีโรโมนตามเส้นทาง เมื่อแต่ละเส้นทางถูกมดเลือกเดินจะมีการปรับปรุงค่าฟีโรโมนในระดับเฉพาะที่เกิดขึ้นการปรับปรุงฟีโรโมนในระดับเฉพาะที่คือ การลดระดับของฟีโรโมนเส้นทางนั้นให้น้อยลง ทำให้มดตัวอื่นมีโอกาสที่จะสำรวจไปยังเส้นทางอื่นที่ไม่เคยไปมาก่อนได้และอาจพบเส้นทางที่ดีกว่าเดิมได้ เมื่อ ACS ทำงานครบรอบ (มดเลือกเดินทุกทาง) จะมีการปรับปรุงค่าฟีโรโมนในระดับครอบคลุมทั้งระบบ ในแต่ละรอบของการทำงานโดยจะใช้ค่าการทำงานของมดตัวที่ดีที่สุดเป็นค่าในการปรับปรุงกระบวนการนี้จะทำเฉพาะรอบการเดินทางที่สั้นที่สุดเท่านั้นจึงจะสามารถปรับปรุงปริมาณฟีโรโมนในระดับรวมได้ ทำให้เส้นทางที่ดีที่สุดมีปริมาณมากขึ้นมากกว่าเส้นทางอื่นๆ และแสดงว่าเป็นเส้นทางที่ดีที่สุด ณ ขณะนั้นด้วย (Sum-Im Thanathip. 2004)

2.3.6 Mixed Integer Programming (MIP)

เป็นการประยุกต์ของเทคนิค Mixed Integer Programming สำหรับการคำนวณการจัดตารางการเดินเครื่องเครื่องกำเนิดไฟฟ้า โดยจะอาศัยวิธี Branch and Bound เป็นอีกวิธีหนึ่ง ซึ่งสำหรับวิธี Branch and Bound จะเป็นการแสดงการคำนวณของปัญหายูนิตคอมมิตเมนต์ อย่างไรก็ตามวิธีการของ Mixed Integer Programming เหมาะสำหรับระบบขนาดเล็กหรือขนาดกลาง เพราะจะแสดงการคำนวณของปัญหายูนิตคอมมิตเมนต์ที่มีเครื่องกำเนิดไฟฟ้าน้อยมากในช่วงเวลาที่สั้นๆ ดังนั้นเทคนิคนี้จึงยังไม่เหมาะสมสำหรับระบบขนาดใหญ่

2.3.7 วิธีลากรางจ์รีแล็กเซชัน (Lagrangian Relaxation: LR)

จากเงื่อนไขต่างๆ เราสามารถเขียนสมการ Lagrange Relaxation ได้ดังนี้

$$L(P,U,\lambda) = F_{TotalCost} + \sum_{t=1}^{T} \lambda_t (P_{Dt}) \sum_{i=1}^{N} U_{it} P_{it} \tag{2.16}$$

วิธีลากรางจ์รีแล็กเซชัน (Lagrangian Relaxation: LR) ที่ใช้ในการแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้านั้นเป็นวิธีที่อาศัยหลักการ Dual Optimization ในการหาคำตอบ หลักการนี้จะแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าโดยการไม่คำนึงถึงเงื่อนไขของปัญหาเป็นเวลาชั่วคราว (Relaxing) และคำนวณปัญหาเสมือนว่าไม่มีเงื่อนไขอยู่ในปัญหา การคำนวณหาค่าที่เหมาะสมที่สุด (Optimization) ของปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้า จากหลักการ Dual Optimization นั้นทำได้โดยการหาค่าสูงสุด (Maximize) ของฟังก์ชันลากรางจ์โดยพิจารณาตัวคูณลากรางจ์ (Lagrangian Multiplier: λ) พร้อมกันกับหาค่าต่ำสุด (Minimize) ของฟังก์ชันลากรางจ์โดยพิจารณาตัวแปรที่ต้องการหาคำตอบของปัญหาในขณะที่กำหนดตัวคูณลากรางจ์มีค่าคงที่นั่นคือ

$$q^*(\lambda) = \max q(\lambda) \tag{2.17}$$

เมื่อ

$$q^*(\lambda) = \min L(P,U,\lambda) \tag{2.18}$$

ขั้นตอนหลักของวิธีการนี้ประกอบด้วย 2 ขั้นตอน คือ

1. ทำการหาค่าตัวคูณลากรางจ์ที่สามารถทำให้ $q(\lambda)$ มีค่ามากขึ้น

2. สมมติว่าตัวคูณลากรางจ์ที่ได้จากขั้นตอนที่ 1 มีค่าคงที่ จากนั้นทำการหาค่าต่ำสุดของฟังก์ชันลากรางจ์ (L) โดยพิจารณาตัวแปร P และ U เมื่อสิ้นสุดขั้นตอนข้างต้น จะได้ผลลัพธ์ที่เรียกว่า Dual Solution (q^*) ซึ่งเป็นผลลัพธ์ของแต่ละรอบการคำนวณ

$$q^*(\lambda) = \sum_{t=1}^{T}\sum_{i=1}^{N}[F_i(P_{it}) - \lambda_t P_{it}]U_{it} + ST_{it}U_{it} + \lambda_t P_{Dt} \quad (2.19)$$

ผลลัพธ์ของ Dual Solution (q^*) นี้ไม่ใช่คำตอบที่แท้จริงของปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้า และขั้นตอนต่อไปหลังจากได้ Dual Solution แล้ว จะทำการคำนวณต่อไปโดยพิจารณาเฉพาะเครื่องกำเนิดไฟฟ้าที่มีสถานะเปิด $(U_i = 1)$ ในผลของ Dual Solution โดยการนำเฉพาะเครื่องกำเนิดไฟฟ้าที่มีสถานะเปิดเหล่านี้มาคำนวณการแก้ปัญหาการจ่ายโหลดอย่างประหยัด (Economic Dispatch) สำหรับแต่ละชั่วโมงในช่วงเวลาทั้งหมดที่พิจารณา และผลลัพธ์ที่ได้แต่ละชั่วโมงในขั้นตอนนี้เมื่อนำมารวมกันแล้วจะถูกเรียกว่า Primal Solution (J^*)

$$J^* = \sum_{t=1}^{T}\sum_{i=1}^{N}[F_i(P_{it}) + (1 - U_{i,t-1})ST_{it}]U_{it} \quad (2.20)$$

เมื่อได้ Primal Solution แล้ว ขั้นตอนต่อไปคือการค่าตัวคูณลากรางจ์ตัวใหม่ที่สามารถทำให้ $q(\lambda)$ ซึ่งก็คือ q^* ของแต่ละรอบการคำนวณมีค่ามากขึ้นกว่าเดิม การหาค่าตัวคูณลากรางจ์ตัวใหม่โดยทั่วไปจะอาศัยหลักการกราเดียนท์ในการปรับเปลี่ยนค่าของตัวคูณลากรางจ์ ซึ่งความสัมพันธ์ที่ใช้ในการหาค่าตัวคูณลากรางจ์ตัวใหม่เป็นดังสมการที่ (2.15)

$$\lambda_t^{iter+1} = \lambda_t^{iter} + \alpha\frac{dq}{d\lambda} \quad (2.21)$$

เมื่อ iter คือจำนวนรอบของการคำนวณ และ α คือค่าที่ใช้กำหนดระยะในการปรับเปลี่ยนค่าตัวคูณลากรางจ์ตัวใหม่ ซึ่งการกำหนดค่า α นี้จะอาศัยค่าของ $\frac{dq}{d\lambda}$ มากำหนด โดยมีหลักการที่ใช้โดยทั่วไปเป็นดังนี้

$\alpha = \alpha_1$ เมื่อ มีค่าเป็นบวก

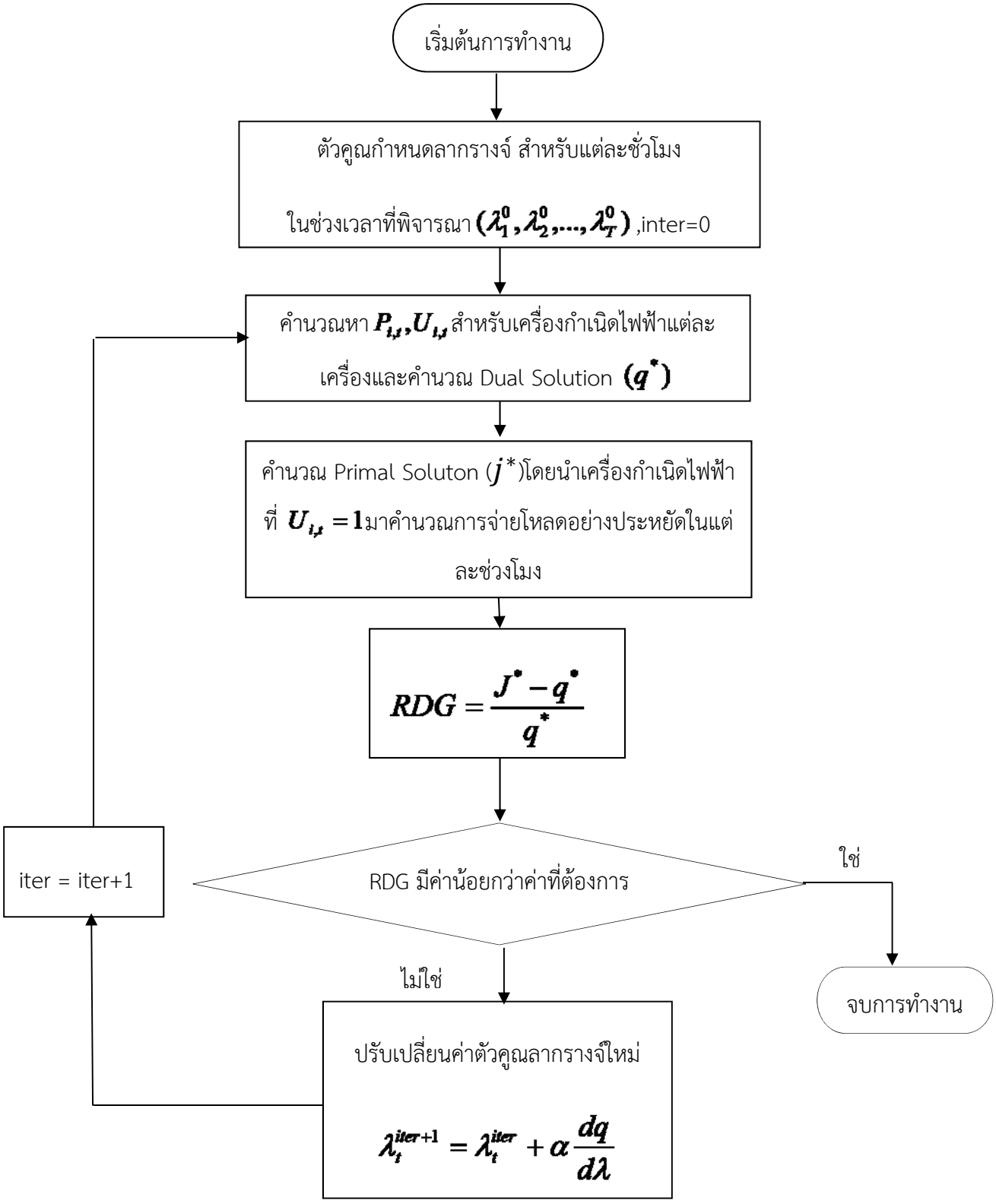
$\alpha = \alpha_2$ เมื่อ มีค่าเป็นลบ

โดยที่ $\alpha_1 >> \alpha_2$

ตัวคูณลากรางจ์ตัวใหม่ที่ได้นี้จะถูกนำกลับไปหา Dual Solution และ Primal Solution ในรอบการคำนวณใหม่อีกครั้งตามขั้นตอนเดิมข้างต้น ซึ่งการคำนวณนี้จะเป็นการคำนวณแบบวนรอบโดยจะอาศัยค่า Relative Duality Gap (RDG) เป็นเกณฑ์ในการหยุดคำนวณ โดยจะหยุดคำนวณเมื่อ RDG มีค่าน้อยกว่าค่าที่ต้องการ เช่นมีค่าน้อยกว่า 0.1 หรือ 0.05 เป็นต้น

$$RDG = \frac{J^* - q^*}{q^*} \tag{2.22}$$

เมื่อสิ้นสุดการคำนวณ ผลลัพธ์ของปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าที่ได้ คือสถานะ(เปิด/ปิด) และกำลังไฟฟ้าของเครื่องกำเนิดไฟฟ้าแต่ละเครื่องที่ได้จากการคำนวณในส่วนของ Primal Solution ในรอบการคำนวณสุดท้าย (ผนึกดี กฤษณา. 2557)



รูปที่ 2.1 ขั้นตอนการแก้ปัญหายูนิตคอมมิตเมนต์โดยใช้วิธีลากรางจ์รีแล็กเซชัน

ตัวอย่างการแก้ปัญหายูนิตคอมมิตเมนต์โดยใช้วิธีลากรางจ์รีแล็กเซชัน

ขั้นตอนการแก้ไขปัญหายูนิตคอมมิตเมนต์จะถูกแสดงไว้ในโจทย์ตัวอย่าง โดยโจทย์ตัวอย่างนี้จะไม่นำค่า startup cost, minimum up time และ minimum down time มาใช้ในการแก้ไขปัญหา

$$F_1 = 400 + 8P_1 + 0.001P_1^2 \quad \$/\text{hr} \quad 100 \le P_1 \le 500 \quad \text{MW}$$
$$F_2 = 80 + 5P_1 + 0.004P_2^2 \quad \$/\text{hr} \quad 50 \le P_2 \le 200 \quad \text{MW}$$
$$F_3 = 200 + 6P_3 + 0.002\,P_3^2 \quad \$/\text{hr} \quad 100 \le P_3 \le 350 \quad \text{MW}$$

รอบที่ 1

ในรอบแรกจะเสมือนว่า $\lambda = 0$, เริ่มด้วยการปรับค่า λ

จาก $$q(\lambda) = \sum_{t=1}^{T}\sum_{i=1}^{N}[F_i(P_{it}) - \lambda_t P_{it}]U_{it} + ST_{it}U_{it} + \lambda_t P_{Dt}$$

$$\frac{\partial q(\lambda)}{\partial \lambda} = P_{Dt} - \sum_{t=1}^{N} U_{it}P_{it} = 0$$

ถ้า $\frac{\partial q(\lambda)}{\partial \lambda}$ เป็นบวก ใช้ $\alpha = 0.01$

ถ้า $\frac{\partial q(\lambda)}{\partial \lambda}$ เป็นลบ ใช้ $\alpha = 0.002$

ชั่วโมงที่ 1 $\lambda_1 = \lambda_1 + (P_{1,load})(0.01) = 300 \times 0.01 = 3$

ชั่วโมงที่ 2 $\lambda_2 = \lambda_2 + (P_{2,load})(0.01) = 800 \times 0.01 = 8$

รอบที่ 2

$\lambda_1 = 3$ และ $\lambda_2 = 8$

Unit 1

$$\frac{\partial q(\lambda)}{\partial P_1} = \frac{\partial(400 + 8P_1 + 0.001P_1^2 - \lambda_t P_1)}{\partial P_1} = 8 + 0.002 - \lambda_t$$

ดังนั้น $P_1 = \frac{\lambda_t - 8}{0.002}$

Unit1	$P_1(cal)$	P_1	$F_1(P_1)$	$F_1(P_1)-\lambda P_1$	U_{1t}
λ_1	-2500	100	1210	910	0
λ_2	0	100	1210	410	0

ขั้นตอนที่ 1 จะทำการหาค่ากำลังไฟฟ้า P_1 ในแต่ละชั่วโมง

ขั้นตอนที่ 2 เมื่อได้ค่ากำลังไฟฟ้า P_1 ในแต่ละชั่วโมงแล้ว หากค่า P_1 ในแต่ละชั่วโมงไม่เป็นไปตามเงื่อนไขของเครื่องกำเนิดไฟฟ้า จะต้องบังคับค่าของ P_1 ให้เป็นไปตามเงื่อนไขของเครื่องกำเนิดไฟฟ้า คือกำหนดให้ค่า P_1 เป็นค่าพิกัดต่ำสุดที่กำหนดในเงื่อนไข ถ้าค่าของ P_1 ที่ได้จากข้างต้นน้อยกว่าค่าพิกัดต่ำสุดในเงื่อนไขหรือกำหนดให้ค่า P_1 เป็นค่าพิกัดสูงสุดในเงื่อนไข ถ้าค่าของ P_1 ที่ได้จากข้างต้นมากกว่าค่าพิกัดสูงสุดในเงื่อนไข

ขั้นตอนที่ 3 จะทำการหาค่าของ U_{it} โดยแทนค่า λ_1, P_1 ที่ได้จากข้างต้นลงใน สมการ $F_1(P_1)-\lambda P_1$ ถ้าค่าที่ได้ออกมาเป็นบวก ค่าต่ำสุดของเทอมนั้นที่เป็นไปได้คือ 0 ซึ่งทำได้โดยกำหนดให้ $U_{it}=0$ หรือหากค่าที่ได้มีค่าไม่เป็นบวก(เป็นค่าลบหรือศูนย์) การที่จะทำให้ค่าของเทอมนั้นต่ำสุดได้โดยกำหนด $U_{it}=1$, หาค่าใน unit 2 และ 3 ต่อไป

Unit 2

$$\frac{\partial q(\lambda)}{\partial P_2}=\frac{\partial(80+5P_2+0.004P_2^2-\lambda_t P_2)}{\partial P_2}=5+0.008-\lambda_t$$

ดังนั้น $P_2=\dfrac{\lambda_t-5}{0.008}$

Unit2	$P_2(cal)$	P_2	$F_2(P_2)$	$F_2(P_2)-\lambda P_2$	U_{2t}
λ_1	-250	50	340	190	0
λ_2	375	200	1240	-360	1

Unit 3

$$\frac{\partial q(\lambda)}{\partial P_{31}}=\frac{\partial(200+6P_1+0.002P_1^2-\lambda_t P_1)}{\partial P_3}=6+0.004-\lambda_t$$

ดังนั้น $P_3=\dfrac{\lambda_t-6}{0.004}$

Unit3	$P_3(cal)$	P_3	$F_3(P_3)$	$F_3(P_3)-\lambda P_3$	U_{3t}
λ_1	-750	100	820	520	0
λ_2	500	350	2545	-255	1

เมื่อสิ้นสุดขั้นตอนในข้างต้น จะสามารถหาค่าของ dual solution $(q^*(\lambda))$ได้โดยการแทนค่า P,λ,U_{it} ลงในฟังก์ชันลากรางจ์ (L) จะได้ค่า $q(\lambda)$ ซึ่งเป็นค่า dual solution $(q^*(\lambda))$ ที่ได้ของแต่ละรอบการคำนวณ

$$q(\lambda)=\sum_{t=1}^{2}\sum_{t=1}^{3}[F_i(P_{it})-\lambda_t P_{it}]*U_{it}+\sum_{t=1}^{2}\lambda_t P_{Dt}$$

$$q(\lambda)=[F_2(P_{22})-\lambda_2 P_{22}]+[F_3(P_{32})-\lambda_2 P_{32}]+3(300)+8(800)$$

$$q(\lambda)=-360-255+900+6400=6685$$

ขั้นตอนต่อไปจะหา primal solution (J^*) โดยการพิจารณาเฉพาะเครื่องที่มีสถานะเปิด $(U_{it}=1)$ ในผลของ dual solution $(q^*(\lambda))$มาคำนวณการจ่ายโหลดอย่างประหยัดโดยจะใช้วิธีแบบดั้งเดิม (ในการทดลองจะใช้ วิธีการ DEA ในการคำนวณการจ่ายโหลดอย่างประหยัด) แต่อย่างไรก็ตามค่าของ U_{it} ที่ได้มาจากผลของ dual solution $(q^*(\lambda))$ ในบางกรณีไม่สามารถนำมาคำนวณการจ่ายโหลดอย่างประหยัดได้หรือไม่ให้คำตอบที่เป็นไปได้ (Feasible Solution) เช่น ในกรณีของโจทย์ตัวอย่างนี้ กรณีที่อาจจะเกิดขึ้นได้ คือ

(1) กรณีที่ค่าของ U_{it} ที่ได้จาก dual solution มีค่าเป็นศูนย์ทั้งหมด

(2) กรณีเครื่องกำเนิดไฟฟ้าที่ได้จาก dual solution มีขนาดพิกัดรวมไม่เพียงพอที่จะจ่ายให้โหลดของระบบ

(3) กรณีที่ความต้องการใช้กำลังไฟฟ้าของระบบมีค่าน้อยกว่าผลรวมของค่าพิกัดต่ำสุดของเครื่องกำเนิดไฟฟ้าทุกเครื่องที่มีสถานะเปิดจาก dual solution

กรณีเหล่านี้จะทำให้ไม่สามารถหาคำตอบของการจ่ายโหลดอย่างประหยัดได้ ดังนั้นเมื่อมีกรณีเหล่านี้เกิดขึ้นในขั้นตอนนี้ จะทำให้ไม่สามารถหาค่าของ J^* ได้ และจากการใช้ค่าของ Relative Duality Gap (RDG) เป็นตัวกำหนดสำหรับการหยุดวนรอบการคำนวณ การกำหนดให้ J^* เป็นค่ามากๆค่าหนึ่งก็เป็นทางออกหนึ่งเมื่อเกิดกกรณีเหล่านี้ เพราะ J^* ที่มีค่ามากๆ จะทำให้ Relative Duality Gap (RDG) มีค่ามาก ในขณะที่การวนรอบของการคำนวณก็จะหยุดต่อเมื่อ RDG มีค่าน้อยกว่าที่ต้องการ ซึ่งในตัวอย่างนี้จะกำหนดให้ J^* มีค่า 10,000 ต่อหนึ่งโรง

$$J^* = 10{,}000 \times 2 = 20{,}000$$

$$RDG = \frac{J^* - q^*}{q^*} = \frac{20{,}000 - 6685}{6685} = 1.99$$

จะเห็นได้ว่าค่า RDG ที่คำนวณได้ยังมีค่ามาก จึงต้องทำการคำนวณรอบต่อไป โดยทำการปรับเปลี่ยนตัวคูณลากรางจ์ใหม่ (ในการทดลองเราจะเปลี่ยนตัวคูณลากรางจ์ด้วยวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม)

$$\frac{\partial q(\lambda)}{\partial \lambda} = P_{Dt} - \sum_{i=1}^{N} U_{it} P_{it}$$

$t = 1$ $\quad \frac{dq}{dt} = 300 > 0$ $\quad$ ดังนั้น $\quad \alpha = 0.01$

$t = 2$ $\quad \frac{dq}{dt} = -200 - 350 + 800 > 0$ $\quad$ ดังนั้น $\quad \alpha = 0.01$

เราก็จะได้ตัวคูณลากรางจ์ใหม่

$t = 1$ $\quad \lambda_1 = 3 + 300(0.01) = 6$

$t = 2$ $\quad \lambda_2 = 8 + 250(0.01) = 10.5$

รอบที่ 3

$t = 1$ $\quad \lambda_1 = 6$

$t = 2$ $\quad \lambda_2 = 10.5$

Unit 1

$$\frac{\partial q(\lambda)}{\partial P_1} = \frac{\partial (400 + 8P_1 + 0.001P_1^2 - \lambda_t P_1)}{\partial P_1} = 8 + 0.002 - \lambda_t$$

ดังนั้น $P_1 = \frac{\lambda_t - 8}{0.002}$

Unit1	$P_1(cal)$	P_1	$F_1(P_1)$	$F_1(P_1) - \lambda P_1$	U_{1t}
λ_1	-1000	100	1210	610	0
λ_2	1250	500	4650	-600	1

Unit 2

$$\frac{\partial q(\lambda)}{\partial P_2} = \frac{\partial(80 + 5P_2 + 0.004P_2^2 - \lambda_t P_2)}{\partial P_2} = 5 + 0.008 - \lambda_t$$

ดังนั้น $P_2 = \frac{\lambda_t - 5}{0.008}$

Unit2	$P_2(cal)$	P_2	$F_2(P_2)$	$F_2(P_2) - \lambda P_2$	U_{2t}
λ_1	125	125	767.5	17.5	0
λ_2	687.5	200	-860	-860	1

Unit 3

$$\frac{\partial q(\lambda)}{\partial P_3} = \frac{\partial(200 + 6P_1 + 0.002P_1^2 - \lambda_t P_1)}{\partial P_3} = 6 + 0.004 - \lambda_t$$

ดังนั้น $P_3 = \frac{\lambda_t - 6}{0.004}$

Unit3	$P_3(cal)$	P_3	$F_3(P_3)$	$F_3(P_3) - \lambda P_3$	U_{3t}
λ_1	0	100	820	220	0
λ_2	1125	350	2545	-1130	1

$$q(\lambda) = \sum_{t=1}^{2}\sum_{t=1}^{3}[F_i(P_{it}) - \lambda_t P_{it}] * U_{it} + \sum_{t=1}^{2}\lambda_t P_{Dt}$$

$$q(\lambda) = -600 - 860 - 1130 + 6(300) + 10.5(800) = 7610$$

ทำการคำนวณการจ่ายโหลดอย่างประหยัดโดยวิธีแบบดั้งเดิม ซึ่งในโจทย์ตัวอย่างจะไม่พูดถึงวิธีคำนวณดังกล่าวเพราะในการทดลองจะใช้วิธีการดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม ในการคำนวณการจ่ายโหลดอย่างประหยัด จึงนำค่าที่ได้มาแสดง

$$P_1^{ed} = 250, P_2^{ed} = 200, P_3^{ed} = 350$$

จากสมการ

$$q(\lambda) = \sum_{t=1}^{2}\sum_{t=1}^{3}[F_i(P_{it}) - \lambda_t P_{it}] * U_{it} + \sum_{t=1}^{2}\lambda_t P_{Dt}$$

$$J^* = F_1(P_{12}) + F_2(P_{22}) + F_3(P_{32}) + 10{,}000$$

$$J^* = 2462.5 + 1240 + 2545 + 10{,}000 = 16247.5$$

$$RDG = \frac{J^* - q^*}{q^*} = \frac{16247.5 - 7610}{7610} = 1.14$$

จะเห็นได้ว่าค่า RDG ที่ได้ยังมีค่ามากอยู่จึงทำการปรับเปลี่ยนตัวคูณลากรางจ์ใหม่

$$\frac{\partial q(\lambda)}{\partial \lambda} = P_{Dt} - \sum_{i=1}^{N} U_{it} P_{it}$$

$t = 1 \quad \lambda_1 = 6 + 300(0.01) = 9$

$t = 2 \quad \lambda_2 = 10.5 + 250(0.01) = 10$

จากนั้นก็ทำการเพิ่มค่าจำนวนรอบขึ้นไปจนผลลัพธ์ได้ตามน้อยกว่าค่าที่ต้องการ ซึ่งเราอาจจะกำหนดให้ RDG มีค่าน้อยกว่า 0.001 เป็นต้น

2.3.8 วิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (Differential Evolution Algorithm)

วิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม หรือ DEA เป็นขั้นตอนของการวิวัฒนาการแบบใหม่ที่ใช้ค่าตัวแปรจริงและอาศัยวิธีการมิวเตชัน (Mutation) เป็นวิธีการค้นหา ซึ่งวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (DEA) จะมีหลักการทำงานหรือการพัฒนาร่วมกันกับ Conventional Genetic Algorithm (CGA) โดยทั้งสองวิธีการนี้จะรักษาประชากรที่มีศักยภาพและใช้กลไกการคัดเลือกในการคัดเลือกบุคคลที่ดีที่สุด แต่สองวิธีการนี้จะมีข้อแตกต่างกันดังนี้

- DEA ทำงานโดยตรงในจุดที่เวคเตอร์ตั้งอยู่ในขณะที่ CGA อาศัยหลักการของสตริงที่เป็นเลขฐานสอง

- CGA จะอาศัยหลัก Recombination ในสำรวจและค้นพื้นที่ ในขณะที่ DEA จะใช้วิธีพิเศษของ Mutation ที่เป็นตัวดำเนินการที่โดดเด่น

- DEA เป็นส่วนหนึ่งของวิวัฒนาการ โดยจะมุ่งเน้นการเชื่อมโยงระหว่างบุคคลกับลูกหลาน ในขณะที่ CGA จะยังคงรักษาการเชื่อมโยงทางพันธุกรรมไว้

Differential Evolution Algorithm (DEA) จะทำงานแบบขนาน โดยใช้ประชากร P ขนาด N_p ในประชากร $P^{(G)}$ จะประกอบด้วย สมาชิกที่อยู่ในรูปเวกเตอร์ $X_i^{(G)}$ ในทุกๆเจนเนอเรชัน G ระหว่างขั้นตอนการเพิ่มประสิทธิภาพ DEA จะรักษาประชากร $P^{(G)}$ ของเวกเตอร์ N_p ของปัญหา (ผนึกดี กฤษณา.2557)

$$P^{(G)} = [X_1^{(G)}, \ldots, X_i^{(G)}, \ldots, X_{N_p}^{(G)}]; i = 1, \ldots, N_p \tag{2.23}$$

โดยแต่ละ $X_i^{(G)}$ จะมีมิติเท่ากับ D มิติ เป็นพารามิเตอร์เลขจำนวนเต็ม การแก้ไขปัญหาขึ้นอยู่กับพารามิเตอร์ D

$$X_i^{(G)} = [X_{1,i}^{(G)}, \ldots, X_{j,i}^{(G)}, \ldots, X_{D,t}^{(G)}]; i = 1, \ldots, N_p; j = 1, \ldots, D \tag{2.24}$$

2.3.8.1 การสมมติค่าเริ่มต้น (Initialization)

ขั้นตอนแรกของการเพิ่มประสิทธิภาพ DEA คือการกำหนดค่าที่เป็นไปได้เริ่มต้น โดยจะกำหนดขอบเขตของตัวแปรในการตัดสินใจแต่ละตัวของค่าที่เป็นไปได้เริ่มต้น จากนั้นให้สุ่มหาประชากรคำตอบที่

เป็นไปได้เริ่มต้นโดยกำหนดให้โอกาสที่จะถูกเลือกของคาตอบมีค่าสม่ำเสมอ (Uniform Probability Distribution) โดยคำตอบแต่ละคำตอบจะมีมิติเท่ากับ D และจำนวนคำตอบที่เป็นไปได้เริ่มต้นเท่ากับ N_p จากนั้นคำนวณหา Function Value ของแต่ละคำตอบเริ่มต้นที่เป็นไปได้ ดังสมการนี้

$$X_{(j,i)}^{(G=0)} = Xj^{\min} + rand_j[0,1]*(X_j^{\max} - X_j^{\min}) \quad (2.25)$$

เมื่อ $i = 1,\ldots,N_p$

$i = 1,\ldots,D$

$X_{(j,i)}^{(G=0)}$ คือ ค่าเริ่มต้นของพารามิเตอร์ j^{th} ของเวกเตอร์อิสระ i^{th}

$X_j^{\min}, X_j^{\max}$ คือ ค่าต่ำสุดและค่าสูงสุดของพารามิเตอร์ j^{th}

2.3.8.2 การมิวเตชัน (Mutation)

กระบวนการ DEA จะดำเนินการตามหลักการพันธุกรรมพื้นฐาน 3 ขั้นตอน คือ มิวเตชัน(Mutation) ครอสโอเวอร์ (Crossover) และการคัดเลือก (Selection) เพื่อสร้างประชากรในเจนเนอ เรชันถัดไปคือ $P^{(G+1)}$ โดยใช้ประชากรในปัจจุบัน $P^{(G)}$

การมิวเตชันจะเริ่มต้นด้วยการกำหนดเวกเตอร์เป้าหมาย $X_{(i,G)}$ โดยที่ $i = 1,2,\ldots,N_p$ แล้วทาการสุ่มเลือกจำนวน 3 เวกเตอร์ $X_{(r1G),}X_{(r2G),}X_{(r3G)}$ จากประชากรตั้งต้นโดยที่จะต้องไม่ซ้ำกับเวกเตอร์เป้าหมาย แล้วจึงจะคำนวณหามิวแตนท์เวกเตอร์ $(V_{i,G+1})$ จากความสัมพันธ์

$$V_{i,G+1} = X_{r1,G} + F(Xr_{r2,G} - X_{(r3,G)}); i = 1,2,\ldots,N_p \quad (2.26)$$

เมื่อ $X_{i,G}$ คือ เวกเตอร์เป้าหมาย

$V_{i,G+1}$ คือ มิวแตนท์เวกเตอร์

F คือ จำนวนจริงที่มีค่าคงที่และมีค่าระหว่าง 0 ถึง 2

$X_{r1G,}X_{r2G,}X_{r3G}$ คือ Random Vector

โดยที่ $i, r1, r2, r3 \in \{1, \ldots, N_p\}$ และ $i \neq r1 \neq r2 \neq r3$

2.3.8.3 การข้ามสายพันธุ์ (Crossover)

ในกระบวนการข้ามสายพันธุ์ (Crossover) จะช่วยเพิ่มความหลากหลายของมิวแตนท์เวกเตอร์ (Mutant Vector) ในรุ่นปัจจุบัน โดยจะสร้างเวกเตอร์ทดลอง (Trial Vector) $u_{(j,i)}^{(G)}$ ด้วยการผสมผสานระหว่างมิวแตนท์เวกเตอร์ V_i กับเวกเตอร์เป้าหมาย (X_i) โดยสามารถหาค่าเวกเตอร์ทดลองได้จากสมการ

$$U_i^{(G)} = u_{j,i}^{(G)} = \begin{cases} V_{j,i}^{(G)}, if \;\; rand_j[0,1] \leq CR \; or \;\; j = s \\ X_{j,i}^{(G)}, \qquad Otherwise \end{cases} \tag{2.27}$$

เมื่อ $u_{(j,i)}^{(G)}$ คือ เวกเตอร์ทดลอง (Trial Vector)

$V_{j,i}^{(G)}$ คือ มิวแตนท์เวกเตอร์ (Mutant Vector)

$rand_j$ คือ ค่าที่สุ่มขึ้นมา มีค่า (0,1)

S คือ ค่า Index จากการสุ่มเลือกมีค่าเป็นเลขจำนวนเต็มระหว่าง 1,2,...,D ซึ่งทำให้มั่นใจได้ว่าอย่างน้อย 1 พารามิเตอร์มาจาก Mutant Vector

CR คือ Crossover Constant มีค่าเป็นเลขจำนวนจริงระหว่าง 0 ถึง 1

2.3.8.4 การคัดเลือก (Selection)

กระบวนการคัดเลือก (Selection) จะเป็นขั้นตอนสุดท้ายในการหาคำตอบ โดยมีวิธีการคือ เปรียบเทียบค่าของฟังก์ชันของเวกเตอร์ทดลอง (Trial Vector) กับเวกเตอร์เป้าหมาย (Target Vector) ถ้าเวกเตอร์ทดลองให้ค่าของฟังก์ชันที่ดีกว่า มันก็จะแทนที่เวกเตอร์เป้าหมายในเจอเนอเรชันถัดไป จากนั้นก็จะทาซ้ำขั้นตอนที่ 2 ถึงขั้นตอนที่ 3 จนครบทุกเวกเตอร์ในเจนเนอเรชันปัจจุบัน และแทนที่เจนเนอเรชันปัจจุบันด้วยเจนเนอเรชันต่อไป แล้วทำซ้ำกระบวนการทั้งหมดจนถึง Stopping Criteria

$$X_i^{(G+1)} = \begin{cases} U_i^{(G)} \qquad if \qquad f(U_i^{(G)} \leq f(X_i^{(G)}) \\ X_i^{(G)} \; otherwise \end{cases} \tag{2.28}$$

ขั้นตอนการทำงานของ Differential Evolution Algorithm (DEA)

ขั้นตอนที่ 1 รับค่าพารามิเตอร์ควบคุมต่างๆ F CR N_p D

ขั้นตอนที่ 2 สมมติค่าประชากรเริ่มต้นโดยเป็นเวกเตอร์ขนาด D จำนวน N_p เวกเตอร์

ขั้นตอนที่ 3 ประเมินค่าประชากรที่สมมติได้จากขั้นตอนที่ 2

ขั้นตอนที่ 4 สร้างเวกเตอร์มิวเตชันจากประชากรที่มีจานวน N_p เวกเตอร์

ขั้นตอนที่ 5 สร้างเวกเตอร์ทดลองจากการครอสโอเวอร์จำนวน N_p เวกเตอร์

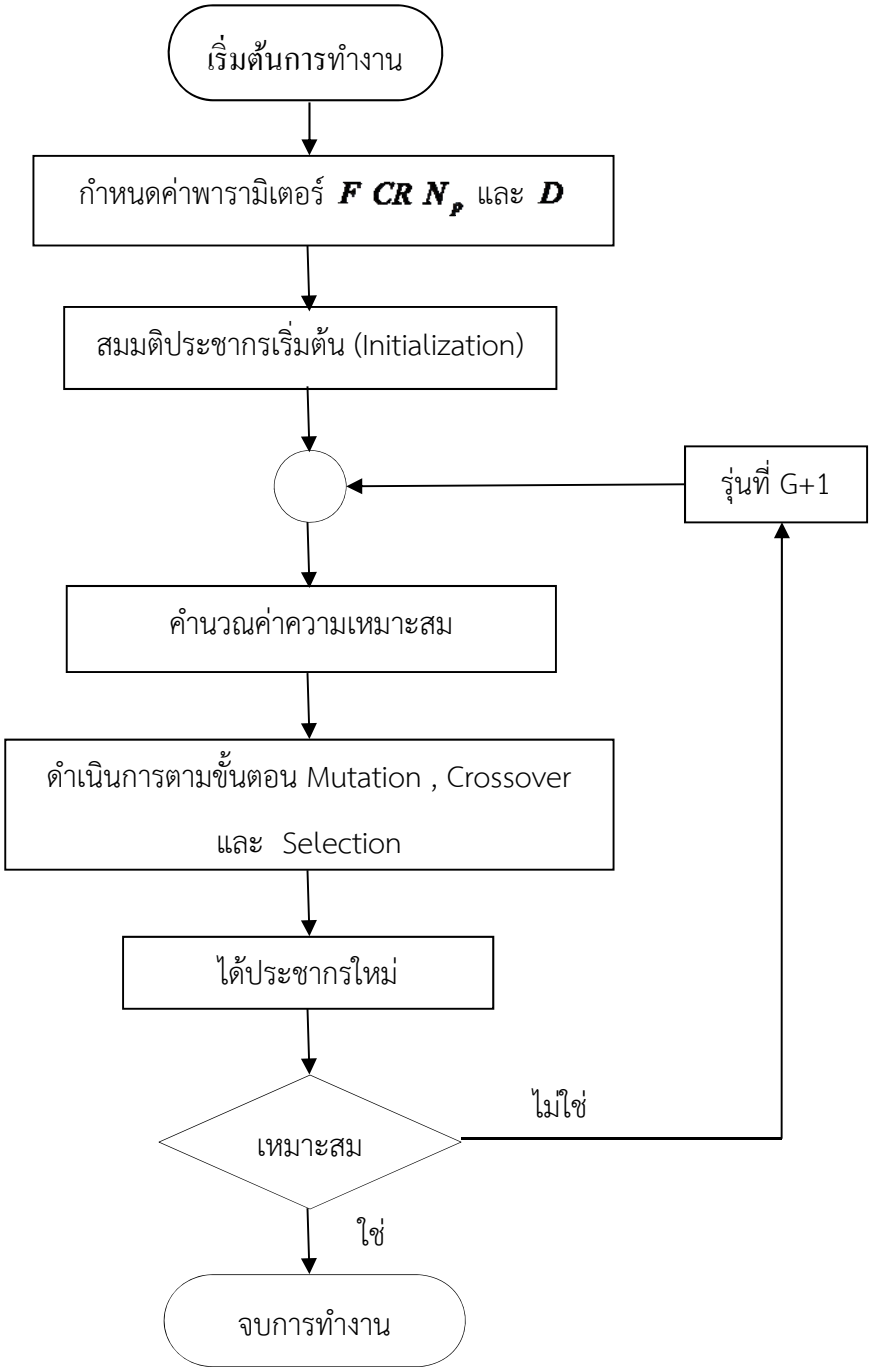
ขั้นตอนที่ 6 ประเมินค่าเวกเตอร์ทดลองที่สร้างขึ้น

ขั้นตอนที่ 7 ทำการคัดเลือกโดยเปรียบเทียบค่าระหว่างประชากรเดิมกับเวกเตอร์ทดลองที่สร้างขึ้น

ขั้นตอนที่ 8 นำประชากรที่ทำการคัดเลือกแล้วใช้เป็นประชากรสาหรับในรอบการคำนวณถัดไป

ขั้นตอนที่ 9 ทำซ้ำขั้นตอนที่ 4 ถึงขั้นตอนที่ 8 จนกว่าค่าที่ดีที่สุดของประชากรจะเป็นค่าที่เหมาะสม หรือจบการคำนวณรอบสูงสุด

จากขั้นตอนการทำงานของ DEA สามารถสรุปการทำงานได้ดังรูป 2.2



รูปที่ 2.2 แผนผังแสดงการทำงาน Differential Evolution Algorithm (DEA)

ตัวอย่างการทำงานของดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (DEA)

ให้ฟังก์ชั่นวัตถุประสงค์ (Objective Function) :

$$f(x) = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8$$

ขั้นตอนที่ 1 กำหนดพารามิเตอร์ที่ใช้ในการควบคุม

ตารางที่ 2.1 แสดงค่าพารามิเตอร์ที่ใช้ในการควบคุม

จำนวนตัวแปรทั้งหมด (Decision Parameter, D	8
จำนวนประชากร (Population size, N_p)	5
Scaling mutation factor F	0.7
Crossover constant CR	0.6

หมายเหตุ: N_p ควรมีค่ามากกว่า D เพื่อเพิ่มความหลากหลายของคำตอบ

ขั้นตอนที่ 2 ทำการ Initialize ประชาการ P ด้วยสมการ (2.25)

ตารางที่ 2.2 แสดงค่าที่ได้จากการ Initialize ประชากร P

Parameters/Individuals	Individual 1	Individual 2	Individual 3	Individual 4	Individual 5
Parameter 1 (x_1)	0.95	0.57	0.18	0.92	0.6
Parameter 2 (x_2)	0.43	0.88	0.29	0.87	0.79
Parameter 3 (x_3)	0.38	0.93	0.99	0.65	0.28
Parameter 4 (x_4)	0.78	0.74	0.86	0.47	0.34
Parameter 5 (x_5)	0.64	0.81	0.39	0.38	0.56
Parameter 6 (x_6)	0.55	0.66	0.42	0.82	0.93
Parameter 7 (x_7)	0.71	0.59	0.56	0.21	0.86
Parameter 8 (x_8)	0.82	0.28	0.8	0.33	0.33
Fitness $f(x)$	5.26	5.46	4.49	4.65	4.69

โดย Fitness Function ก็คือ ฟังก์ชั่นวัตถุประสงค์(Objective function)

ขั้นตอนที่ 3 เลือก ฟังก์ชั่นเป้าหมาย(target vector,i) และ random vector ($r1, r2$ และ $r3$) จากประชากรที่สมมติขึ้นในขั้นตอนที่ 2 โดยที่ $i, r1, r2$ และ $r3$ จะต้องอยู่ใน $\{1,..., N_p\}$ และ $i \neq r1 \neq r2 \neq r3$ ในที่นี้เราจะเลือก

ตารางที่ 2.3 แสดงการเลือกฟังก์ชั่นเป้าหมายและ random vector

i	1
$r1$	5
$r2$	3
$r3$	4

ขั้นตอนที่ 4 ทำการหา mutant vector $(V_{i,G+1})$ โดยใช้สมการ (2.26) ได้ผลดังตารางที่ 2.4

ตารางที่ 2.4 แสดงผลการสร้าง mutant vector หรือกระบวนการ mutation

	X_{r1}	X_{r2}	X_{r3}	$X_{r2} - X_{r3}$	$F(X_{r2} - X_{r3})$	$X_{r1} + F(X_{r2} - X_{r3})$
Parameter 1 (x_1)	0.6	0.18	0.92	-0.74	-0.518	0.082
Parameter 2 (x_2)	0.79	0.29	0.29	-0.58	-0.406	0.384
Parameter 3 (x_3)	0.28	0.99	0.99	0.34	0.238	0.518
Parameter 4 (x_4)	0.34	0.86	0.86	0.39	0.273	0.613
Parameter 5 (x_5)	0.56	0.39	0.39	0.01	0.007	0.567
Parameter 6 (x_6)	0.93	0.42	0.42	-0.4	-0.28	0.65
Parameter 7 (x_7)	0.86	0.56	0.56	0.35	0.245	1.105
Parameter 8 (x_8)	0.33	0.8	0.8	0.47	0.329	0.659
Fitness $f(x)$	4.69	4.49	4.65	-	-	4.578

ขั้นตอนที่ 5 ทำการ crossover โดยใช้สมการ (2.27) (ถ้าสุ่มค่าแล้วน้อยกว่าหรือเท่ากับค่า CR ให้เลือก mutant vector $V_{j,i}^{(G)}$ ถ้านอกจากนั้นให้เลือก target vector $X_{j,i}^{(G)}$) จะได้ผลดังตารางที่ 2.5

ตารางที่ 2.5 แสดงกระบวนการ crossover

	Target vector	Mutant vector	Trial vector	Random[0,1]
Parameter 1 (x_1)	0.95	0.082	0.082	0.43
Parameter 2 (x_2)	0.43	0.384	0.384	0.15
Parameter 3 (x_3)	0.38	0.518	0.38	0.78
Parameter 4 (x_4)	0.78	0.613	0.613	0.44
Parameter 5 (x_5)	0.64	0.567	0.64	0.91
Parameter 6 (x_6)	0.55	0.65	0.65	0.27
Parameter 7 (x_7)	0.71	1.105	0.71	0.66
Parameter 8 (x_8)	0.82	0.659	0.659	0.35
Fitness $f(x)$	5.26	4.578	4.118	-

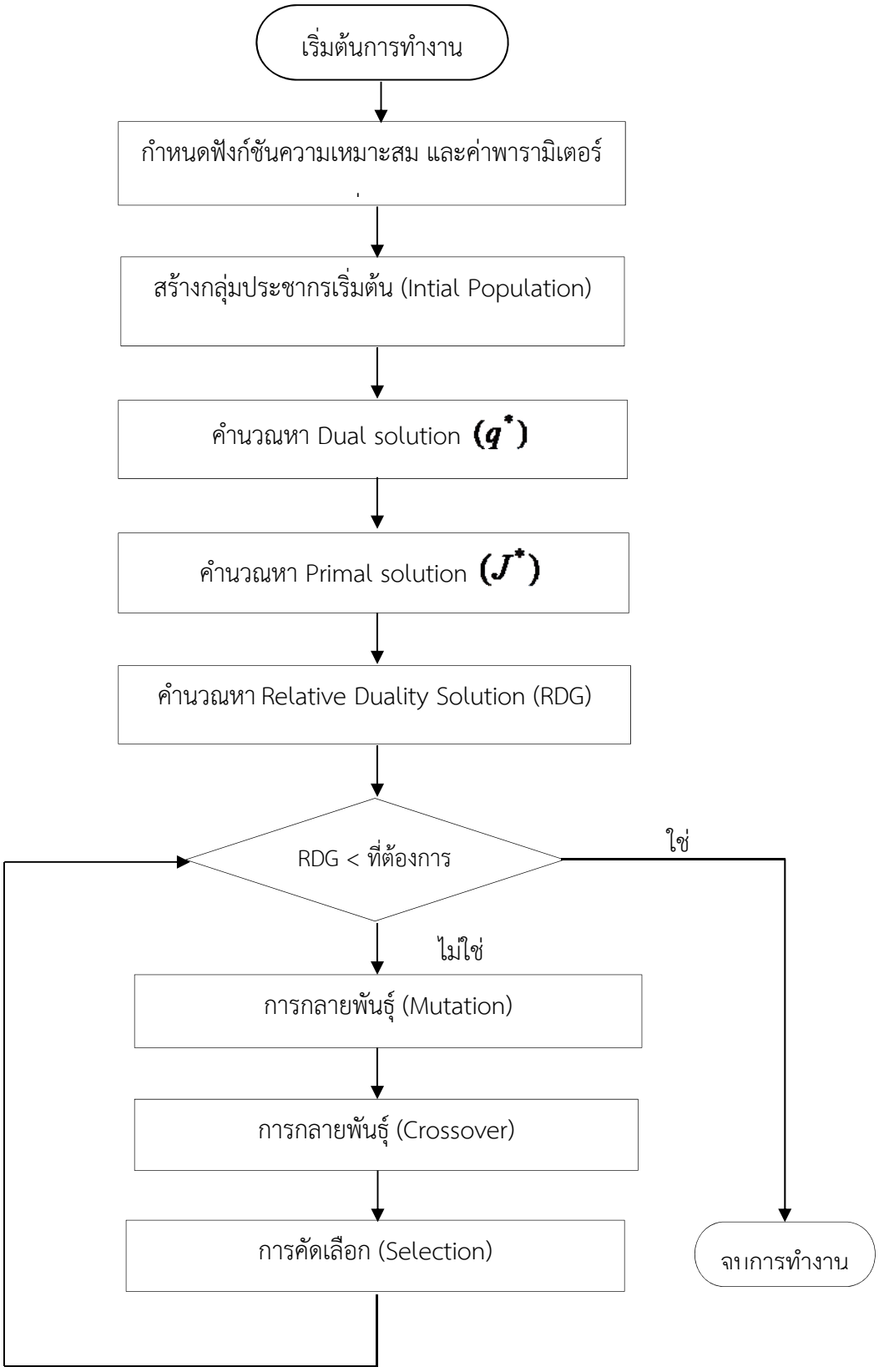
ตารางที่ 2.6 แสดงประชากรในรุ่นที่ 2

Parameters/Individuals	Individual 1	Individual 2	Individual 3	Individual 4	Individual 5
Parameter 1 (x_1)	0.082				
Parameter 2 (x_2)	0.384				
Parameter 3 (x_3)	0.38				
Parameter 4 (x_4)	0.613				
Parameter 5 (x_5)	0.64				
Parameter 6 (x_6)	0.65				
Parameter 7 (x_7)	0.71				
Parameter 8 (x_8)	0.659				
Fitness $f(x)$	4.118				

ขั้นตอนที่ 7 ทำซ้ำในขั้นตอนที่ 3-6 อีกครั้ง จนได้ค่าที่เหมาะสมสำหรับคำตอบ(DEA จะทำการคำนวณแบบขนาน ซึ่งจะทำให้ได้ประชากรในรุ่นที่ 2 มาพร้อมกัน และนำมาคำนวณค่าที่เหมาะสม หากยังไม่พบค่าตามที่ต้องการก็จะทำการสร้างประชากรในรุ่นที่ 3 ต่อไป)

2.4 การประยุกต์ใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (Lagrangian Relaxation combined with Differential Evolution Algorithm: LR-DEA) เป็นวิธีที่ประยุกต์รวมวิธีลากรางจ์รีแล็กเซชัน (LR) กับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (DEA) เข้าด้วยกัน โดยการนำวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมมาใช้ในการปรับเปลี่ยนค่าของตัวคูณลากรางจ์ให้กับวิธีลากรางจ์รีแล็กเซชันในการแก้ปัญหายูนิตคอมมิตเมนต์ ซึ่งวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมมาใช้ในการปรับเปลี่ยนค่าตัวคูณลากรางจ์ให้กับวิธีลากรางจ์รีแล็กเซชันนี้ จะทำให้การลู่เข้าหาคำตอบของวิธีลากรางจ์รีแล็กเซชันกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม ดีกว่าการลู่เข้าหาคำตอบของวิธีลากรางจ์รีแล็กเซชัน และช่วยทำให้คุณภาพของผลลัพธ์ที่ได้ดีขึ้นกว่าใช้วิธีลากรางจ์รีแล็กเซชันในการแก้ปัญหาการแก้ปัญหายูนิตคอมมิตเมนต์โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมโดยสรุปเป็นขั้นตอนดังรูปที่ 2.3



รูปที่ 2.3 การแก้ปัญหายูนิตคอมมิตเมนต์โดยวิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

2.5 ทบทวนงานวิจัยที่เกี่ยวข้องกับโครงงาน

Thanathip Sum-Im ได้ทำการศึกษาวิจัยเกี่ยวกับเรื่อง “Lagrangian Relaxation Combined with Differential Evolution Algorithm for Unit Commitment Problem” โดยงานวิจัยนี้มีจุดประสงค์เพื่อแก้ไขปัญหาการวางแผนการเดินเครื่องกำเนิดไฟฟ้า โดยเลือกใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (LR-DEA) เพื่อให้ได้แผนการเดินเครื่องกำเนิดไฟฟ้าที่มีต้นทุนในการผลิตที่ต่ำที่สุด โดยพิจารณาข้อจำกัดคือ กำลังไฟฟ้าสมดุล กำลังไฟฟ้าสำรอง ข้อจำกัดของเครื่องกำเนิดไฟฟ้า เพื่อนำมาทดสอบกับเครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต ระยะเวลาการจัดตารางเดินเครื่องเครื่องกำเนิดไฟฟ้า 24 ชั่วโมง แล้วนำผลลัพธ์ที่ได้จากการแก้ปัญหาวิธีนี้มาเปรียบเทียบกับผลลัพธ์ที่ได้จากการแก้ปัญหาวิธีการอื่นๆ ซึ่งผลสรุปจากการทำวิจัยพบว่าการใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (LR-DEA) มีค่าใช้จ่ายในการผลิตต่ำที่สุด เมื่อเทียบกับวิธีการอื่น

นายพุฒิพงศ์ ชยันโต นายภาณุวัฒน์ เสาวพันธ์ และนายอนวัช วาฤทธิ์ มหาวิทยาลัยศรีนครินทรวิโรฒ ได้ทำการศึกษาวิจัยเกี่ยวกับเรื่อง “การจัดตารางเดินเครื่องเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม” โดยงานวิจัยนี้มีจุดประสงค์เพื่อศึกษาและแก้ไขปัญหาการจัดตารางเดินการเครื่องเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม โดยพัฒนาโปรแกรมเพื่อให้สะดวกต่อการคำนวณและการแก้ไขปัญหายูนิตคอมมิตเมนต์ โดยพิจารณาข้อจำกัดและเงื่อนไขต่างๆ ได้แก่ กำลังไฟฟ้าสมดุล กำลังการผลิตสำรอง ข้อจำกัดของเครื่องกำเนิดไฟฟ้า เวลาเดินเครื่องกำเนิดไฟฟ้าอย่างน้อยที่สุด เวลาหยุดเดินเครื่องอย่างน้อยที่สุด และการสูญเสียในสายส่ง โดยนำมาทดสอบกับเครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต ระยะเวลาการจัดตารางเดินเครื่องเครื่องกำเนิดไฟฟ้า 24 ชั่วโมง ซึ่งผลทดสอบกับโปรแกรมสำหรับการแก้ไขปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าที่สร้างขึ้น ผลที่ได้สรุปว่าวิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม สามารถแก้ไขปัญหาการจัดตารางการเดินเครื่องเครื่องกำเนิดไฟฟ้าได้อย่างมีประสิทธิภาพสามารถจัดตารางการเดินเครื่องเครื่องกำเนิดไฟฟ้าให้คุ้มค่าในทางด้านเศรษฐศาสตร์ และสามารถผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลดได้

DAISUKE MURATA and SUSUMU YAMASHIRO ได้ทำการศึกษาวิจัยเรื่อง Unit Commitment Scheduling by Lagrange Relaxation Method Taking into Account Transmission Losses โดยงานวิจัยนี้มีวัตถุประสงค์เพื่อแก้ปัญหาตารางการเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันโดยเปรียบเทียบการไม่คิดการสูญเสียในสายส่งกับการนำการสูญเสียในสายส่งมาคิดในระบบ จากผลการวิจัย

พบว่า การแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธี Lagrange Relaxation ทั้งที่มีและไม่มีการคิดการสูญเสียในสายส่งนั้น การไม่คิดการสูญเสียในสายส่งอาจจะทำให้ไม่สามารถกำหนดตารางการเดินเครื่องกำเนิดไฟฟ้าใช้งานจริงได้ แต่การนำการสูญเสียในสายส่งมาคิดในระบบ จะเสมือนเป็นการจำลองการสูญเสียที่มีอยู่จริงมาคิดร่วม เพื่อเป็นการเพิ่มประสิทธิภาพให้กับตารางการเดินเครื่องกำเนิดไฟฟ้าและสามารถนำไปใช้งานได้จริง

บทที่ 3
ขั้นตอนและวิธีการดำเนินงาน

3.1 ขั้นตอนการดำเนินงานโครงการ

ในการศึกษาและจัดทำโครงงานนี้ได้แบ่งขั้นตอนในการดำเนินงานดังนี้

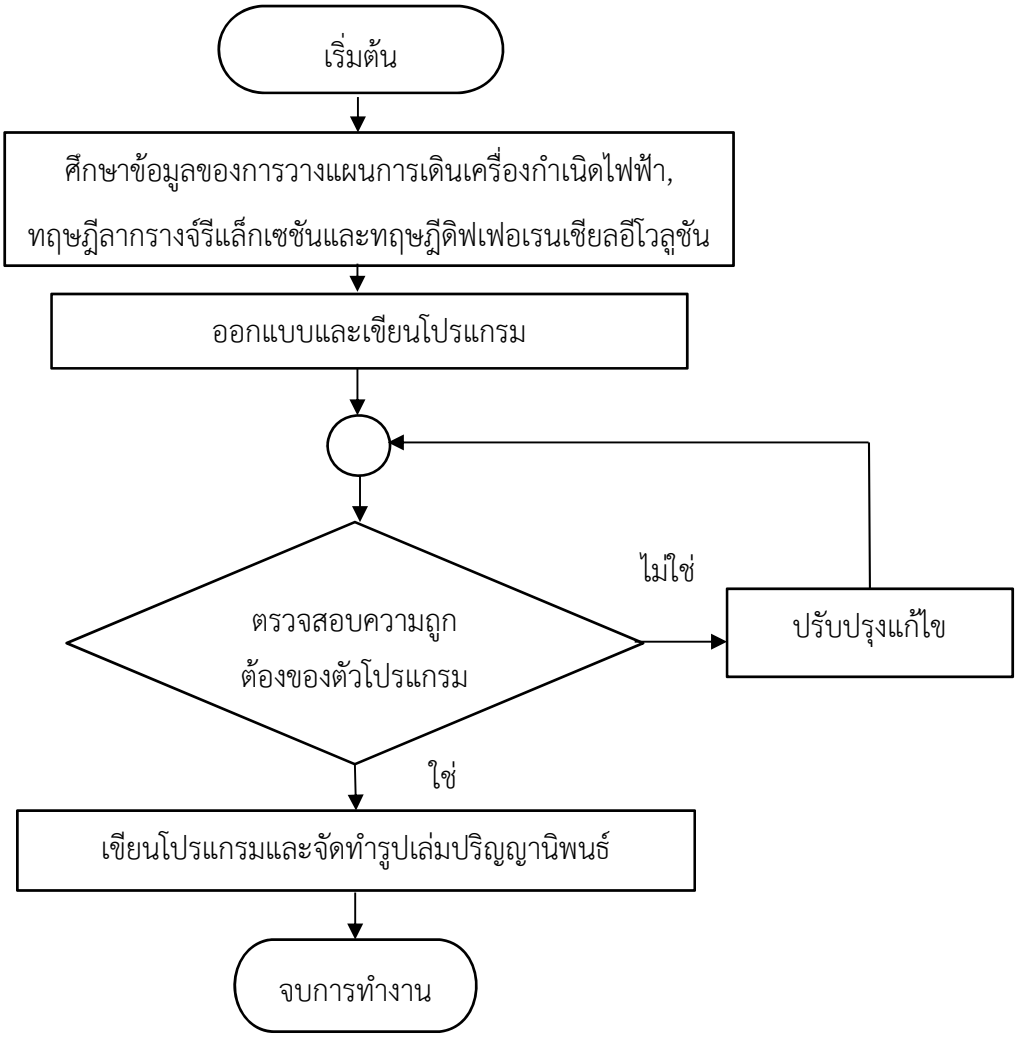
3.1.1 ศึกษาเกี่ยวกับเอกสารที่เกี่ยวข้องกับการคำนวณค่าการจัดตารางเดินเครื่องกำเนิดไฟฟ้าจากหนังสือ และเอกสารต่างๆ

3.1.2 ศึกษาทฤษฎีและการทำงานของวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม (DEA)

3.1.3 ออกแบบ และเขียนโปรแกรมคอมพิวเตอร์

3.1.4 วิเคราะห์ ตรวจสอบความถูกต้อง และสรุปผลที่ได้จากการศึกษา

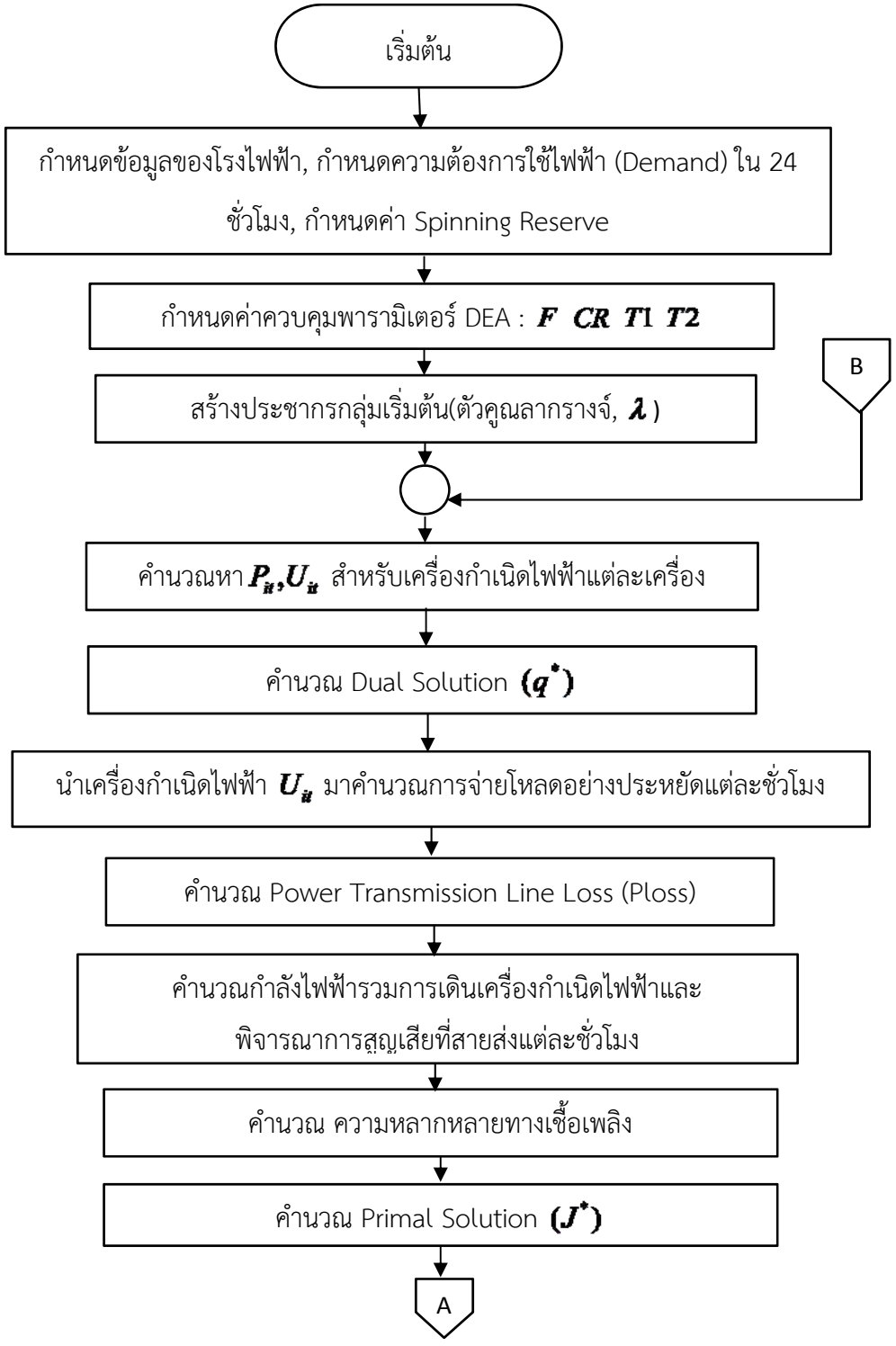
3.1.5 เขียนโครงงาน และจัดทำรูปเล่ม



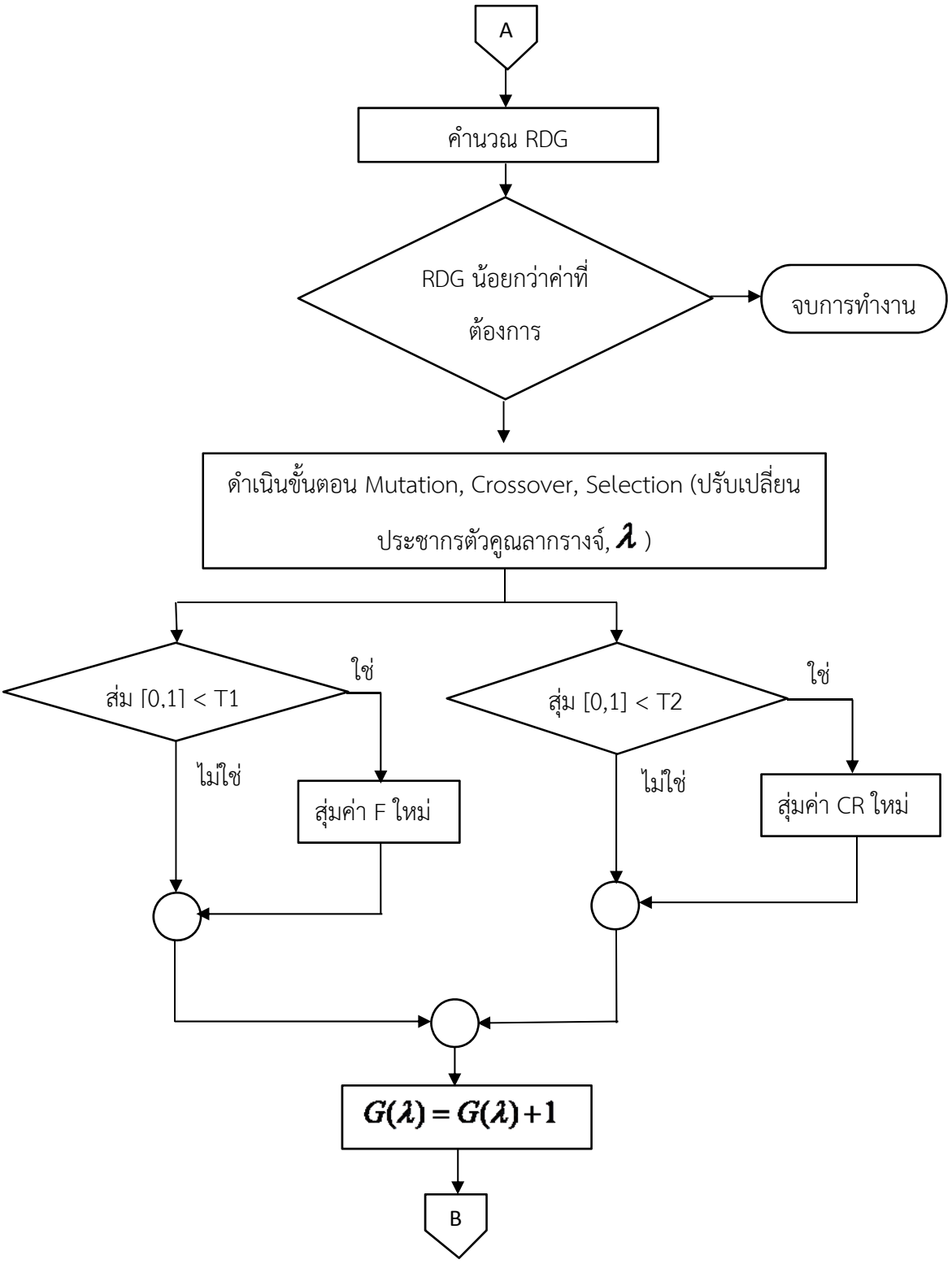
รูปที่ 3.1 แผนผังแสดงขั้นตอนการทำงาน

3.2 ขั้นตอนและวิธีการดำเนินงาน ออกแบบโปรแกรมคอมพิวเตอร์

โครงงานนี้ทำการศึกษาการแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าโดยการเขียนโปรแกรมด้วยโปรแกรม MATLAB R2011a โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม โดยการทำงานของโปรแกรมจะแสดงดังรูปที่ 3.2



รูปที่ 3.2 แผนผังหลักการคิดโปรแกรม



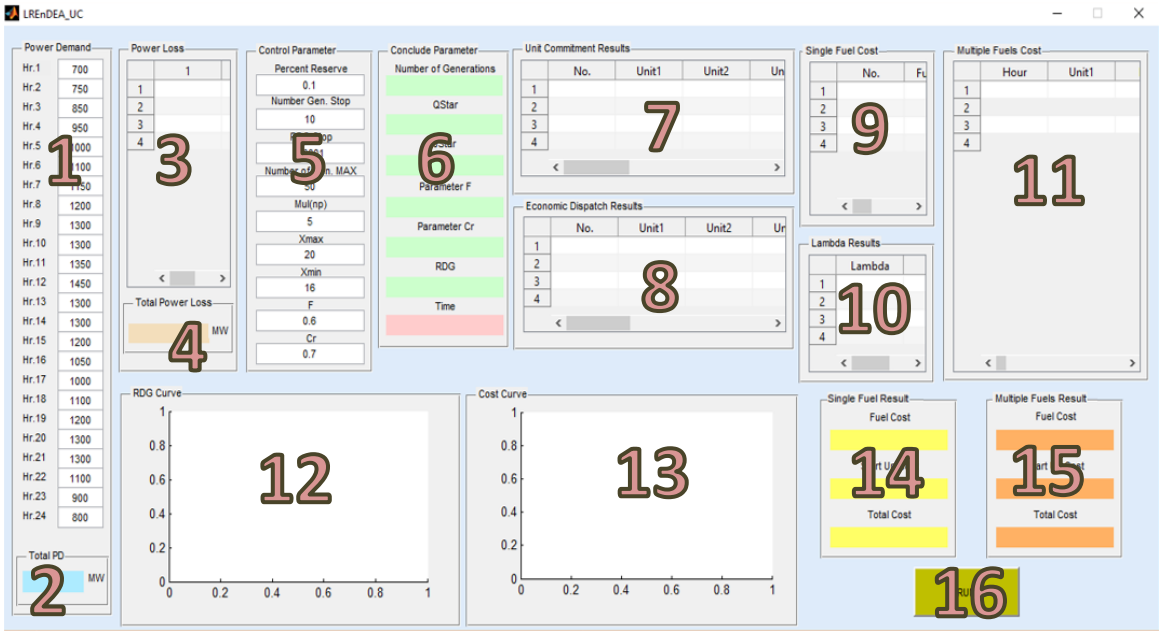
รูปที่ 3.2 แผนผังหลักการคิดโปรแกรม (ต่อ)

3.3 กำหนดปัญหา Unit Commitment

การกำหนดปัญหาจะใช้วิธีลากรางจ์ร่วมกับดิฟเฟอเรนเชียลอีโวลูชันในการแก้ปัญหาการเดินเครื่องกำเนิดไฟฟ้า โดยกำลังไฟฟ้าที่ผลิตออกมาของแต่ละยูนิตให้ราคาต้นทุนต่ำที่สุด ขณะที่ค่า Fitness Function พิจารณาจาก RDG และ Primal Solution

ในส่วนของการผลิตกำลังไฟฟ้าเพื่อจ่ายให้โหลดด้วย DEA เพื่อให้ได้ราคาต้นทุนที่ค่าต่ำที่สุด โดยที่ค่ากำลังไฟฟ้าที่ออกมานั้นจะต้องผ่านเงื่อนไขข้อจำกัดดังนี้ คือ กำลังไฟฟ้าสมดุล กำลังการผลิตสำรอง การสูญเสียในสายส่ง และขอบเขตการผลิตในแต่ละหน่วย

3.4 วิธีการใช้หน้าต่างของโปรแกรม



รูปที่ 3.3 หน้าต่างโปรแกรมสำหรับการแก้ไขปัญหา

ในส่วนของการใช้หน้าต่างโปรแกรม LREnDEA_UC มีวิธีการใช้ในแต่ละส่วนดังรายละเอียดต่อไปนี้

หมายเลข 1 ส่วนสำหรับป้อนค่าความต้องการใช้กำลังไฟฟ้าในแต่ละชั่วโมง (Power Demand)

หมายเลข 2 ส่วนแสดงผลรวมค่าความต้องการใช้กำลังไฟฟ้าทั้งหมดใน 24 ชั่วโมง (Total Power Demand)

หมายเลข 3 ส่วนแสดงตารางผลลัพธ์กำลังสูญเสียในแต่ละชั่วโมง (Power Loss)

หมายเลข 4 ส่วนแสดงผลรวมกำลังสูญเสียทั้งหมดใน 24 ชั่วโมง (Total Power Loss)

หมายเลข 5 ส่วนสำหรับป้อนค่า Parameter ต่างๆ ได้แก่ กำลังการผลิตสำรอง (Percentage of Power Reserve), จำนวนรอบที่จะหยุดการคำนวณเมื่อไม่เจอค่าใหม่ (Max.Repeat Generation), เกณฑ์ในการหยุดการคำนวณ (RDG Criteria), จำนวนรอบสูงสุดในการคำนวณ (Max. Generation), ตัวคูณ Parameter,(Mul (N_p)),ขอบขาบน $(X_{\max})$, ขอบขาล่าง $(X_{\min})$, Scaling Mutation Factor (F) และCrossover Constant (CR)

หมายเลข 6 ส่วนแสดงข้อมูลสรุปของผลลัพธ์ในการคำนวณมี 7 ค่า ได้แก่ จำนวนรอบในการคำนวณ (Number of Generations), Dual solution (q^*), Primal Solution (J^*), Scaling Mutation Factor (F), Crossover Constant (CR), เกณฑ์ในการหยุดการคำนวณ (RDG) และเวลาที่ใช้ในการคำนวณ (Calculation Time)

หมายเลข 7 ส่วนแสดงตารางผลลัพธ์สถานะเครื่องกำเนิดไฟฟ้าที่ได้จากการแก้ไขปัญหา UC โดยการจัดตารางเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

หมายเลข 8 ส่วนแสดงตารางผลลัพธ์กำลังไฟฟ้าของเครื่องกำเนิดไฟฟ้าที่ได้จากการแก้ไขปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม

หมายเลข 9 ส่วนแสดงตารางผลลัพธ์ราคาต้นทุนการผลิตรายชั่วโมง ซึ่งประกอบด้วย Fuel Cost, Startup Cost และ Total Cost ที่ได้จากการวางแผนเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมแบบเชื้อเพลิงเดียว

หมายเลข 10 ส่วนแสดงค่าที่ λ ใช้ในแต่ละชั่วโมง ที่ได้จากกระบวนการ DEA ที่ใช้การวางแผนเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมได้ในรอบนั้น

หมายเลข 11 ส่วนแสดงค่าตารางผลลัพธ์ราคาต้นทุนการผลิตรายชั่วโมง ซึ่งประกอบด้วย Fuel Cost, Startup Cost และ Total Cost แบบความหลากหลายทางเชื้อเพลิง

หมายเลข 12 ส่วนแสดงกราฟ Relative Duality Gap (RDG) กับจำนวนรอบในการคำนวณ

หมายเลข 13 ส่วนแสดงกราฟ ต้นทุนในการผลิต กับจำนวนรอบในการคำนวณ

หมายเลข 14 ส่วนแสดงผลลัพธ์ของข้อมูล 3 ค่า คือ Fuel Cost, Startup Cost และ Total Cost แบบเชื้อเพลิงเดียว

หมายเลข 15 ส่วนแสดงผลลัพธ์ของข้อมูล 3 ค่า คือ Fuel Cost, Startup Cost และ Total Cost แบบหลากหลายทางเชื้อเพลิง

หมายเลข 16 ปุ่มกดสำหรับให้โปรแกรมเริ่มการทำงาน

3.5 แผนผังการดำเนินงานโครงการ

ตารางที่ 3.1 ตารางแสดงระยะเวลาดำเนินงานของโครงการ

ขั้นตอน	ระยะเวลา (เดือน)								
	ส.ค. 2559	ก.ย. 2559	ต.ค. 2559	พ.ย. 2559	ธ.ค. 2559	ม.ค. 2560	ก.พ. 2560	มี.ค. 2560	เม.ย. 2560
นำเสนอหัวข้อโครงงาน									
ศึกษาข้อมูลทางทฤษฎี									
ออกแบบ วิเคราะห์ และปรับปรุงโปรแกรม									
สรุปผล และจัดทำรูปเล่มโครงงานวิศวกรรม									

3.6 รายละเอียดของเครื่องคำนวณที่ใช้

สำหรับการสร้างโปรแกรมการคำนวณการวางแผนเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม และการทดสอบปัญหาผู้จัดทำได้ใช้คอมพิวเตอร์ ซึ่งมีคุณลักษณะ ดังตารางที่ 3.2

ตารางที่ 3.2 คุณลักษณะคอมพิวเตอร์ที่ใช้ในโครงงาน

หัวข้อ	รายละเอียด
รุ่น	DELL INSPIRON N5110
CPU	Intel® Core™ i5-2450M CPU @ 2.50GHz
ระบบประมวลผล	Window 10
หน่วยความจำ	4.00 GB
โปรแกรมคำนวณ	MATLAB R2011a

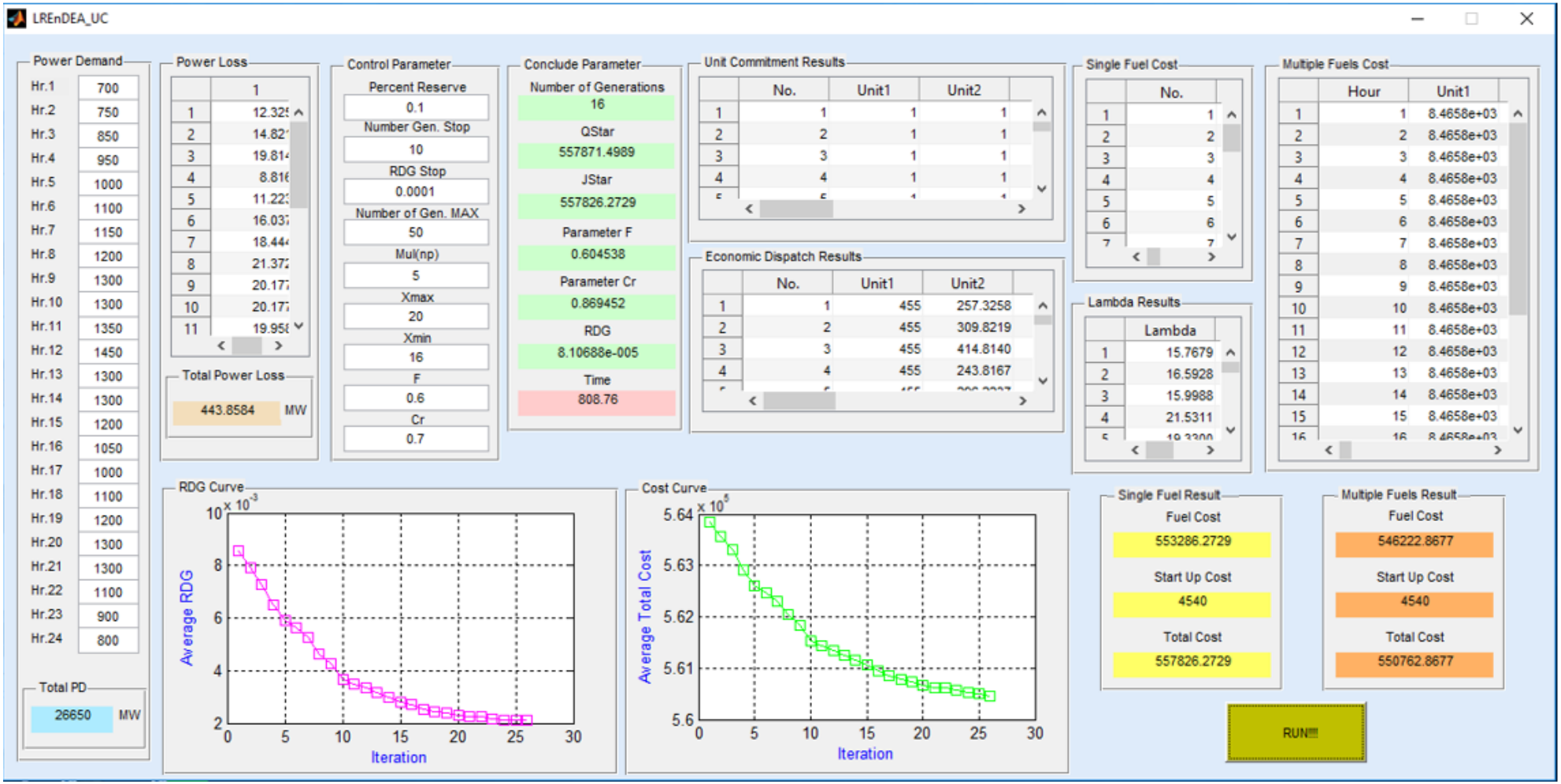
บทที่ 4
ผลการทดลอง

ในการทดลองในโครงงานวิศวกรรมนี้เป็นการแก้ปัญหาการจัดตารางเดินเครื่องกำเนิดไฟฟ้าด้วยวิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชัน โดยใช้โปรแกรม MATLAB R2011a ซึ่งประมวลผลบน Intel® Core™ i5-2450M CPU @ 2.50GHz RAM 4.00 GB

ระบบที่ใช้ทดสอบ คือ เครื่องกำเนิดไฟฟ้า 10 ยูนิต แสดงในภาคผนวก ก ดังตารางที่ ก1.1 ระยะเวลาทำงาน 24 ชั่วโมง โดยมีการพิจารณาการสูญเสียในสายส่ง ซึ่งมีความต้องการใช้ไฟฟ้ารายชั่วโมงแสดงในภาคผนวก ก ดังตารางที่ ก1.2 และพิจารณาการเลือกใช้เชื้อเพลิงได้หลายชนิดแสดงในภาคผนวก ก ดังตารางที่ ก1.3

4.1 ผลการทดลองในการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า

ในการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชัน มีการพิจารณาการสูญเสียในสายส่ง และมีการพิจารณาการเลือกใช้เชื้อเพลิงได้หลายชนิด โดยใช้เครื่องกำเนิดไฟฟ้า 10 ยูนิต ทำงานเป็นเวลา 24 ชั่วโมงสำหรับการทดสอบความถูกต้องของโปรแกรม ได้ทำการทดลองหาคำตอบเป็นจำนวน 100 ครั้ง โดยผลการทดลองถูกแสดงในตารางที่ 4.1



รูปที่ 4.1 แสดงหน้าต่างการทำงานของโปรแกรมการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า

ตารางที่ 4.1 แสดงผลของค่าใช้จ่ายในแต่ละรอบการคำนวณสำหรับการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าจำนวน 100 รอบ

Iterations	Total Cost of Single Fuel ($)	Total Cost of Multiple Fuels($)	Total Loss (MW)	RDG (10^{-5})	Time (sec)
1	559793.7092	553280.2900	472.6923	5.5488	811.93340
2	559869.2565	552813.2021	432.6785	5.1644	490.4537
3	561033.6919	554077.8983	439.5192	3.5631	446.5320
4	561238.3236	554486.8109	461.4509	6.5141	306.5474
5	560072.6913	553229.6029	454.8256	2.5776	470.5641
6	560072.6913	553229.6029	454.8256	6.0579	503.9008
7	560072.6913	553229.6029	454.8256	9.5807	742.7116
8	559978.3456	553117.7919	454.6068	6.2161	745.2114
9	558671.4503	552240.0155	473.8539	9.0110	730.0287
10	561033.6919	554077.8983	439.5192	9.8027	292.0691
11	560072.6913	553229.6029	454.8256	7.3469	470.4394
12	561096.0330	554509.4873	470.3235	8.8868	476.2069
13	558908.2560	551964.9067	447.9848	1.6536	545.4318
14	561238.3236	554486.8109	461.4509	6.5418	543.4542
15	558671.4503	552240.0155	473.8539	5.1728	492.4640
16	558787.2734	551611.1631	428.5520	5.3073	578.1949
17	561012.3164	553830.8171	424.6377	9.9401	5.3073
18	559112.1231	552578.7347	473.8163	4.8607	500.0988
19	559793.7092	553280.2900	472.6923	5.5486	792.5794
20	559869.2565	552813.2021	432.6785	5.1644	493.6135
21	561033.6919	554077.8983	439.5192	3.5631	438.7010
22	561238.3236	554486.8109	461.4509	6.5141	316.2202
23	560072.6913	553229.6029	454.8256	2.5776	460.1664
24	560072.6913	553229.6029	454.8256	6.0579	504.6204
25	560072.6913	553229.6029	454.8256	9.5807	728.4488

ตารางที่ 4.1 แสดงผลของค่าใช้จ่ายในแต่ละรอบการคำนวณสำหรับการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าจำนวน 100 รอบ (ต่อ)

Iterations	Total Cost of Single Fuel ($)	Total Cost of Multiple Fuels($)	Total Loss (MW)	RDG (10^{-5})	Time (sec)
26	558671.4503	552240.0155	473.8539	9.0110	716.1324
27	561033.6919	554077.8983	439.5192	9.8027	287.0406
28	560072.6913	553229.6029	454.8256	7.3469	453.4338
29	559793.7092	553280.2900	472.6923	5.5486	780.5265
30	559869.2565	552813.2021	432.6785	5.1644	470.8690
31	561033.6919	554077.8983	439.5192	3.5631	422.9348
32	561238.3236	554486.8109	461.4509	6.5141	303.4936
33	560072.6913	553229.6029	454.8256	2.5776	438.7967
34	560072.6913	553229.6029	454.8256	6.0579	480.7485
35	560072.6913	553229.6029	454.8256	9.5807	412.2236
36	559978.3456	553117.7919	454.6068	6.2161	698.3337
37	558671.4503	552240.0155	473.8539	9.0110	674.1371
38	561033.6919	554077.8983	439.5192	9.8027	275.4729
39	560072.6913	553229.6029	454.8256	7.3469	437.8382
40	561096.0330	554509.4873	470.3235	8.8868	547.8673
41	559793.7092	553280.2900	472.6923	5.5486	1092.6090
42	559869.2565	552813.2021	432.6785	5.1644	705.4820
43	561033.6919	554077.8983	439.5192	3.5631	449.3010
44	561238.3236	554486.8109	461.4509	6.5141	316.5997
45	560072.6913	553229.6029	454.8256	2.5776	494.7303
46	560072.6913	553229.6029	454.8256	9.5807	505.0914
47	559978.3456	553117.7919	454.6068	6.2161	885.6426
48	558671.4503	552240.0155	473.8539	9.0110	831.3938
49	561033.6919	554077.8983	439.5192	9.8027	332.1689
50	560072.6913	553229.6029	454.8256	7.3469	542.6813

ตารางที่ 4.1 แสดงผลของค่าใช้จ่ายในแต่ละรอบการคำนวณสำหรับการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าจำนวน 100 รอบ (ต่อ)

Iterations	Total Cost of Single Fuel ($)	Total Cost of Multiple Fuels($)	Total Loss (MW)	RDG (10^{-5})	Time (sec)
51	559793.7092	553280.2900	472.6923	5.5486	1092.9301
52	559869.2565	552813.2021	432.6785	5.1644	693.9714
53	561033.6919	554077.8983	439.5192	3.5631	557.4170
54	561238.3236	554486.8109	461.4509	6.5141	356.8235
55	560072.6913	553229.6029	454.8256	2.5776	494.3067
56	560072.6913	553229.6029	454.8256	6.0579	555.7780
57	560072.6913	553229.6029	454.8256	9.5807	475.4490
58	559978.3456	553117.7919	454.6068	6.2161	793.1102
59	558671.4503	552240.0155	473.8539	9.0110	815.3068
60	561033.6919	554077.8983	439.5192	9.8027	307.4747
61	560072.6913	553229.6029	454.8256	7.3469	521.3491
62	561096.0330	554509.4873	470.3235	8.8868	493.7491
63	558908.2560	551964.9067	447.9848	1.6536	586.5154
64	561238.3236	554486.8109	461.4509	6.5418	565.0399
65	558671.4503	552240.0155	473.8539	5.1728	514.3256
66	558787.2734	551611.1631	428.5520	5.3073	607.8563
67	561012.3164	553830.8171	424.6377	9.9401	353.0938
68	559112.1231	552578.7347	473.8163	4.8607	683.7576
69	559663.0586	553043.1643	472.7123	7.0663	528.4263
70	561033.6919	554077.8983	439.5192	2.7234	743.9742
71	561096.0330	554509.4873	470.3235	3.6645	428.9501
72	559793.7092	553280.2900	472.6923	9.0727	660.9168
73	563211.3967	556183.2501	449.9708	4.3950	324.1040
74	560072.6913	553229.6029	454.8256	7.2515	609.0419
75	560072.6913	553229.6029	454.8256	3.5773	514.4903

ตารางที่ 4.1 แสดงผลของค่าใช้จ่ายในแต่ละรอบการคำนวณสำหรับการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าจำนวน 100 รอบ (ต่อ)

Iterations	Total Cost of Single Fuel ($)	Total Cost of Multiple Fuels($)	Total Loss (MW)	RDG (10^{-5})	Time (sec)
76	560072.6913	553229.6029	454.8256	2.3894	374.6574
77	560072.6913	553229.6029	454.8256	1.7118	594.2862
78	559869.2565	552813.2021	432.6785	1.6186	585.7964
79	560072.6913	553229.6029	454.8256	6.0136	378.6652
80	560072.6913	553229.6029	454.8256	7.5205	477.3837
81	559793.7092	553280.2900	472.6923	5.5123	428.1423
82	558671.4503	552240.0155	473.8539	4.6466	298.9511
83	558908.2560	551964.9067	447.9848	3.4971	463.3383
84	559793.7092	553280.2900	472.6923	4.2010	670.5457
85	561012.3164	553830.8171	424.6377	2.7926	610.3935
86	558671.4503	552240.0155	473.8539	3.3433	772.3899
87	560072.6913	553229.6029	454.8256	3.3997	625.5800
88	557826.2729	550762.8677	443.8584	8.1069	808.7596
89	559793.7092	553280.2900	472.6923	3.2163	514.7026
90	561033.6919	554077.8983	439.5192	1.1533	393.6080
91	561238.3236	554486.8109	461.4509	1.6165	548.9137
92	559757.4043	553154.9753	472.9312	4.1652	601.1075
93	559498.0034	553018.9944	473.9509	5.4681	1165.6995
94	564556.9884	557425.0672	431.8203	9.2502	336.5890
95	558671.4503	552240.0155	473.8539	3.7221	708.8242
96	561033.6919	554077.8983	439.5192	3.5631	479.0828
97	559869.2565	552813.2021	432.6785	5.1644	514.0481
98	559793.7092	553280.2900	472.6923	5.5486	838.6046
99	559793.7092	553280.2900	472.6923	5.5486	930.5038
100	561096.0330	554509.4873	470.3235	8.8868	540.2581

จากการคำนวณทั้งหมด 100 รอบ ค่าที่ได้จากการคำนวณในรอบที่ 88 เป็นค่าที่ดีที่สุด กล่าวคือ มีค่าใช้จ่ายในการผลิตสำหรับการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงเป็นจำนวนเงิน 550762.8677 $ ซึ่งต่ำกว่าแบบเชื้อเพลิงเดียว โดยมีค่าใช้จ่ายในการผลิตเท่ากับ 557826.2729 $ อีกทั้งยังเพียงพอต่อความต้องการสำหรับการใช้ไฟฟ้า โดยสามารถแสดงผลทางการคำนวณได้ดังตารางที่ 4.2 – 4.7

ตารางที่ 4.2 แสดงผลของสถานะของเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมง

เวลา (ชั่วโมง)	สถานะของเครื่องกำเนิดไฟฟ้า 24 ชั่วโมง									
	Unit1	Unit2	Unit3	Unit4	Unit5	Unit6	Unit7	Unit8	Unit9	Unit10
1	1	1	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0
3	1	1	0	0	0	0	0	0	0	0
4	1	1	1	1	0	0	0	0	0	0
5	1	1	1	1	0	0	0	0	0	0
6	1	1	1	1	0	0	0	0	0	0
7	1	1	1	1	0	0	0	0	0	0
8	1	1	1	1	1	0	0	0	0	0
9	1	1	1	1	1	0	0	0	0	0
10	1	1	1	1	1	0	0	0	0	0
11	1	1	1	1	1	0	0	0	0	0
12	1	1	1	1	1	0	1	0	0	0
13	1	1	1	1	1	0	1	0	0	0
14	1	1	1	1	1	0	1	0	0	0
15	1	1	1	1	1	0	0	0	0	0
16	1	1	1	1	1	0	0	0	0	0
17	1	1	1	1	1	0	0	0	0	0
18	1	1	1	1	1	0	0	0	0	0
19	1	1	1	1	1	0	0	0	0	0
20	1	1	1	1	1	0	0	0	0	0
21	1	1	1	1	1	0	0	0	0	0
22	1	1	1	1	1	0	0	0	0	0
23	1	1	1	1	1	0	0	0	0	0
24	1	1	1	1	1	0	0	0	0	0

ตารางที่ 4.3 สรุปค่ากำลังไฟฟ้า ราคาค่าใช้จ่ายสำหรับระบบที่มีการเดินเครื่องกำเนิดไฟฟ้า 10 ยูนิตและมีการพิจารณาความสูญเสียที่สายส่ง ในรอบการคำนวณที่ให้ค่าที่ดีที่สุดจากการคำนวณทั้งหมด 100 รอบแบบเชื้อเพลิงเดียว

Hour	Load (MW)	Generation schedule (MW)										Transmission Losses(MW)	Fuel Cost Of Single Fuel ($)	Start Up Cost of Single Fuel ($)	Total Cost Of Single Fuel ($)
		Unit1	Unit2	Unit3	Unit4	Unit5	Unit6	Unit7	Unit8	Unit9	Unit10				
1	700	455	257.3258	0	0	0	0	0	0	0	0	12.3258	13897.7932	0	13897.7932
2	750	455	309.8219	0	0	0	0	0	0	0	0	14.8219	14813.1047	0	14813.1047
3	850	455	414.814	0	0	0	0	0	0	0	0	19.814	16648.8535	0	16648.8535
4	950	455	243.8167	130	130	0	0	0	0	0	0	8.8167	19414.9853	2220	21634.9853
5	1000	455	296.2237	130	130	0	0	0	0	0	0	11.2237	20328.3037	0	20328.3037
6	1100	455	401.0377	130	130	0	0	0	0	0	0	16.0377	22160.0490	0	22160.0490
7	1150	455	453.4447	130	130	0	0	0	0	0	0	18.4447	23078.4759	0	23078.4759
8	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24991.0545	1800	26791.0545
9	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26650.3415	0	26650.3415
10	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26650.3415	0	26650.3415
11	1350	455	455	130	130	199.9581	0	0	0	0	0	19.9581	27654.0666	0	27654.0666
12	1450	455	455	130	130	250	0	54.6096	0	0	0	24.6096	30726.7360	520	31246.7360
13	1300	455	455	130	97.0834	162	0	25	0	0	0	24.0834	27466.7072	0	27466.7072
14	1300	455	455	130	97.0834	162	0	25	0	0	0	24.0834	27466.7072	0	27466.7072
15	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24991.0545	0	24991.0545
16	1050	455	413.3637	20	20	162	0	0	0	0	0	20.3637	22412.9445	0	22412.9445
17	1000	455	360.849	20	20	162	0	0	0	0	0	17.849	21493.9360	0	21493.9360
18	1100	455	455	30.8785	20	162	0	0	0	0	0	22.8785	23324.4844	0	23324.4844
19	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24991.0545	0	24991.0545
20	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26650.3415	0	26650.3415
21	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26650.3415	0	26650.3415
22	1100	455	455	30.8785	20	162	0	0	0	0	0	22.8785	23324.4844	0	23324.4844
23	900	455	255.8195	20	20	162	0	0	0	0	0	12.8195	19661.0487	0	19661.0487
24	800	455	151.0241	20	20	162	0	0	0	0	0	8.0241	17839.0630	0	17839.0630
The total cost of 10 unit 24 hour system													553286.2729	4540	557826.2729

จากสมการที่ (2.9) $$\sum_{i=1}^{N}(P_i) = P_{load} + P_{loss}$$

$$P_{load} = \sum_{i=1}^{N}(P_i) - P_{loss}$$

$P_{load(t=1)}$ = (455+257.3258) - 12.3258 = 700 MW

$P_{load(t=2)}$ = (455+309.8219) - 14.8219 = 750 MW

$P_{load(t=3)}$ = (455+414.8140) - 19.8140 = 850 MW

$P_{load(t=4)}$ = (455+243.8167+130+130) - 8.8167 = 950 MW

$P_{load(t=5)}$ = (455+296.2237+130+130) - 11.2237 = 1000 MW

$P_{load(t=6)}$ = (455+401.0377+130+130) - 16.0377 = 1100 MW

$P_{load(t=7)}$ = (455+453.4447+130+130) - 18.4447 = 1150 MW

$P_{load(t=8)}$ = (455+455+129.3726+20+162) - 21.3726 = 1200 MW

$P_{load(t=9)}$ = (455+455+130+118.1771+162) - 20.1771 = 1300 MW

$P_{load(t=10)}$ = (455+455+130+118.1771+162) - 20.1771 = 1300 MW

$P_{load(t=11)}$ = (455+455+130+130+199.9581) - 19.9581 = 1350 MW

$P_{load(t=12)}$ = (455+455+130+130+250+54.6096) - 24.6096 = 1450 MW

$P_{load(t=13)}$ = (455+455+130+97.0834+162+25) - 24.0834 = 1300 MW

$P_{load(t=14)}$ = (455+455+130+97.0834+162+25) - 24.0834 = 1300 MW

$P_{load(t=15)}$ = (455+455+129.3726+20+162) - 21.3726 = 1200 MW

$P_{load(t=16)}$ = (455+413.3637+20+20+162) - 20.3637 = 1050 MW

$P_{load(t=17)}$ = (455+360.8490+20+20+162) - 17.8490 = 1000 MW

$P_{load(t=18)}$ = (455+455+30.8785+20+162) - 22.8785 = 1100 MW

$P_{load(t=19)}$ = (455+455+129.3726+20+162) - 21.3726 = 1200 MW

$P_{load(t=20)}$ = (455+455+130+118.1771+162) - 20.1771 = 1300 MW

$P_{load(t=21)}$ = (455+455+130+118.1771+162) - 20.1771 = 1300 MW

$P_{load(t=22)}$ = (455+455+30.8785+20+162) - 22.8785 = 1100 MW

$P_{load(t=23)}$ = (455+255.8195+20+20+162) - 12.8195 = 900 MW

$P_{load(t=24)}$ = (455+151.0241+20+20+162) - 8.0241 = 800 MW

จากการคำนวณของระบบที่มีการเดินเครื่องกำเนิดไฟฟ้า 10 ยูนิต พบว่า ค่ากำลังไฟฟ้ารวมที่ผลิตในแต่ละชั่วโมง เมื่อนำมาลบด้วยค่ากำลังไฟฟ้าความสูญเสียที่สายส่งในแต่ละชั่วโมงที่คำนวณได้ จะมีค่าเท่ากับความต้องการกำลังไฟฟ้าในแต่ละชั่วโมงนั้นๆ

ตารางที่ 4.4 แสดงราคาค่าใช้จ่ายในการผลิต ($) สำหรับการเดินเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมงแบบเชื้อเพลิงเดียวของรอบที่ดีที่สุด

เวลา(ชั่วโมง)	Fuel Cost of Single Fuel ($)	Start Up Cost of Single Fuel ($)	Total Cost of Single Fuel ($)
1	13897.7932	0	13897.7932
2	14813.1047	0	14813.1047
3	16648.8535	0	16648.8535
4	19414.9853	2220	21634.9853
5	20328.3037	0	20328.3037
6	22160.0490	0	22160.0490
7	23078.4759	0	23078.4759
8	24991.0545	1800	26791.0545
9	26650.3415	0	26650.3415
10	26650.3415	0	26650.3415
11	27654.0666	0	27654.0666
12	30726.7360	520	31246.7360
13	27466.7072	0	27466.7072
14	27466.7072	0	27466.7072
15	24991.0545	0	24991.0545
16	22412.9445	0	22412.9445
17	21493.9360	0	21493.9360
18	23324.4844	0	23324.4844
19	24991.0545	0	24991.0545
20	26650.3415	0	26650.3415
21	26650.3415	0	26650.3415
22	23324.4844	0	23324.4844
23	19661.0487	0	19661.0487
24	17839.0630	0	17839.0630
Total	553286.2729	4540	557826.2729

ตารางที่ 4.5 แสดงราคาค่าใช้จ่ายในการผลิต ($) สำหรับการเดินเครื่องกำเนิดไฟฟ้าในเวลา 24 ชั่วโมงแบบหลายเชื้อเพลิงของรอบที่ดีที่สุด

เวลา(ชั่วโมง)	Fuel Cost of Multiple Fuels ($)	Start Up Cost of Multiple Fuels ($)	Total Cost of Multiple Fuels ($)
1	13789.1247	0	13789.1247
2	14687.0419	0	14687.0419
3	16487.6716	0	16487.6716
4	19110.3381	2220	21330.3381
5	20006.3202	0	20006.3202
6	21803.0634	0	21803.0634
7	22703.8244	0	22703.8244
8	24706.3961	1800	26506.3961
9	26285.0037	0	26285.0037
10	26285.0037	0	26285.0037
11	27278.8891	0	27278.8891
12	30281.4007	520	30801.4007
13	27054.2660	0	27054.2660
14	27054.2660	0	27054.2660
15	24706.3961	0	24706.3961
16	22164.9187	0	22164.9187
17	21263.5281	0	21263.5281
18	23067.7596	0	23067.7596
19	24706.3961	0	24706.3961
20	26285.0037	0	26285.0037
21	26285.0037	0	26285.0037
22	23067.7596	0	23067.7596
23	19465.5456	0	19465.5456
24	17677.9471	0	17677.9471
Total	546222.8677	4540	550762.8677

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด

เครื่องกำเนิดที่	ชั่วโมงที่ 1		ชั่วโมงที่ 2		ชั่วโมงที่ 3		ชั่วโมงที่ 4	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.80	1	8465.80	1	8465.80
2	2	5323.30	2	6221.20	2	8021.80	2	5092.50
3	0	0	0	0	0	0	3	2857.60
4	0	0	0	0	0	0	3	2694.50
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด (ต่อ)

เครื่องกำเนิดที่	ชั่วโมงที่ 5		ชั่วโมงที่ 6		ชั่วโมงที่ 7		ชั่วโมงที่ 8	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.80	1	8465.80	1	8465.80
2	2	5988.50	2	7785.20	2	8686.00	2	8712.70
3	3	2857.60	3	2857.60	3	2857.60	3	2847.20
4	3	2694.50	3	2694.50	3	2694.50	3	934.79
5	0	0	0	0	0	0	1	3745.90
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด (ต่อ)

เครื่องกำเนิดที่	ชั่วโมงที่ 9		ชั่วโมงที่ 10		ชั่วโมงที่ 11		ชั่วโมงที่ 12	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.80	1	8465.80	1	8465.80
2	2	8712.70	2	8712.70	2	8712.70	2	8712.70
3	3	2857.60	3	2857.60	3	2857.60	3	2857.60
4	3	2503.00	3	2503.00	3	2694.50	1	2694.50
5	1	3745.90	1	3745.90	1	4548.30	1	5623.80
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	2	1927.10
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด (ต่อ)

เครื่องกำเนิดที่	ชั่วโมงที่ 13		ชั่วโมงที่ 14		ชั่วโมงที่ 15		ชั่วโมงที่ 16	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.80	1	8465.80	1	8465.80
2	2	8712.70	2	8712.70	2	8712.70	2	7996.90
3	3	2857.60	3	2857.60	3	2847.20	2	1021.50
4	3	2162.90	3	2162.90	3	934.79	3	934.79
5	1	3745.90	1	3745.90	1	3745.90	1	3745.90
6	0	0	0	0	0	0	0	0
7	2	1109.40	2	1109.40	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด (ต่อ)

เครื่องกำเนิดที่	ชั่วโมงที่ 17		ชั่วโมงที่ 18		ชั่วโมงที่ 19		ชั่วโมงที่ 20	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.80	1	8465.80	1	8465.80
2	2	7095.50	2	8712.70	2	8712.70	2	8712.70
3	2	1021.50	2	1208.60	3	2847.20	3	2857.60
4	3	934.79	3	934.79	3	934.792	3	2503.00
5	1	3745.90	1	3745.90	1	3745.90	1	3745.90
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.6 แสดงราคาค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าของแต่ละชั่วโมงและชนิดของเชื้อเพลิงที่ใช้จากการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด (ต่อ)

เครื่องกำเนิดที่	ชั่วโมงที่ 21		ชั่วโมงที่ 22		ชั่วโมงที่ 23		ชั่วโมงที่ 24	
	F	GEN	F	GEN	F	GEN	F	GEN
1	1	8465.80	1	8465.8000	1	8465.80	1	8465.80
2	2	8712.70	2	8712.7000	2	5297.60	2	3510.00
3	3	2857.60	2	1208.6000	2	1021.50	2	1021.50
4	3	2503.00	3	934.79	3	934.79	3	934.79
5	1	3745.90	1	3745.90	1	3745.90	1	3745.90
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0

ตารางที่ 4.7 สรุปค่ากำลังไฟฟ้า ราคาค่าใช้จ่ายสำหรับระบบที่มีการเดินเครื่องกำเนิดไฟฟ้า 10 ยูนิตและมีการพิจารณาความสูญเสียที่สายส่ง ในรอบการคำนวณที่ 88 ให้ค่าที่ดีที่สุดจากการคำนวณทั้งหมด 100 รอบแบบหลายเชื้อเพลิง

Hour	Load (MW)	Generation schedule (MW)										Transmission Losses (MW)	Fuel Cost Of Multiple Fuels ($)	Start Up Cost of Multiple Fuels($)	Total Cost Of Multiple Fuels($)
		Unit1	Unit2	Unit3	Unit4	Unit5	Unit6	Unit7	Unit8	Unit9	Unit10				
1	700	455	257.3258	0	0	0	0	0	0	0	0	12.3258	13789.1247	0	13789.1247
2	750	455	309.8219	0	0	0	0	0	0	0	0	14.8219	14687.0419	0	14687.0419
3	850	455	414.814	0	0	0	0	0	0	0	0	19.814	16487.6716	0	16487.6716
4	950	455	243.8167	130	130	0	0	0	0	0	0	8.8167	19110.3381	2220	21330.3381
5	1000	455	296.2237	130	130	0	0	0	0	0	0	11.2237	20006.3202	0	20006.3202
6	1100	455	401.0377	130	130	0	0	0	0	0	0	16.0377	21803.0634	0	21803.0634
7	1150	455	453.4447	130	130	0	0	0	0	0	0	18.4447	22703.8244	0	22703.8244
8	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24706.3961	1800	26506.3961
9	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26285.0037	0	26285.0037
10	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26285.0037	0	26285.0037
11	1350	455	455	130	130	199.9581	0	0	0	0	0	19.9581	27278.8891	0	27278.8891
12	1450	455	455	130	130	250	0	54.6096	0	0	0	24.6096	30281.4007	520	30801.4007
13	1300	455	455	130	97.0834	162	0	25	0	0	0	24.0834	27054.2660	0	27054.2660
14	1300	455	455	130	97.0834	162	0	25	0	0	0	24.0834	27054.2660	0	27054.2660
15	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24706.3961	0	24706.3961
16	1050	455	413.3637	20	20	162	0	0	0	0	0	20.3637	22164.9187	0	22164.9187
17	1000	455	360.849	20	20	162	0	0	0	0	0	17.849	21263.5281	0	21263.5281
18	1100	455	455	30.8785	20	162	0	0	0	0	0	22.8785	23067.7596	0	23067.7596
19	1200	455	455	129.3726	20	162	0	0	0	0	0	21.3726	24706.3961	0	24706.3961
20	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26285.0037	0	26285.0037
21	1300	455	455	130	118.1771	162	0	0	0	0	0	20.1771	26285.0037	0	26285.0037
22	1100	455	455	30.8785	20	162	0	0	0	0	0	22.8785	23067.7596	0	23067.7596
23	900	455	255.8195	20	20	162	0	0	0	0	0	12.8195	19465.5456	0	19465.5456
24	800	455	151.0241	20	20	162	0	0	0	0	0	8.0241	17677.9471	0	17677.9471
The total cost of 10 unit 24 hour system													546222.8677	4540	550762.8677

ตัวอย่างการคำนวณราคาต้นทุนรวมในกรณีที่ความต้องการกำลังไฟฟ้าเท่ากับ 700 MW (ชั่วโมงที่ 1) ซึ่งในชั่วโมงที่ 1 มีเครื่องกำเนิดไฟฟ้าที่เดินเครื่องทั้งหมด 2 ยูนิต คือ ยูนิตที่ 1 และ 2 ซึ่งยูนิตที่ 1 ได้เลือกใช้เชื้อเพลิงที่ 1 และยูนิตที่ 2 ได้เลือกใช้เชื้อเพลิงที่ 2 สามารถคำนวณได้จากสมการที่ (2.10), (2.11) และ(2.13)

$$F_1P_1 = (1000) + ((16.19)(455)) + ((0.00048)(455)^2) = 8465.80\$$$

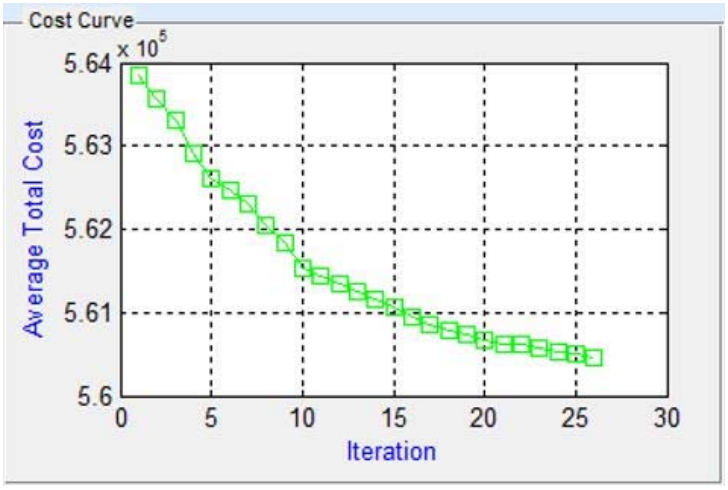
$$F_2P_1 = (945) + ((16.94)(257.3258)) + ((0.00029)(257.3258)^2) = 5323.30\$$$

$$F_T = 13789124626\$/hr$$

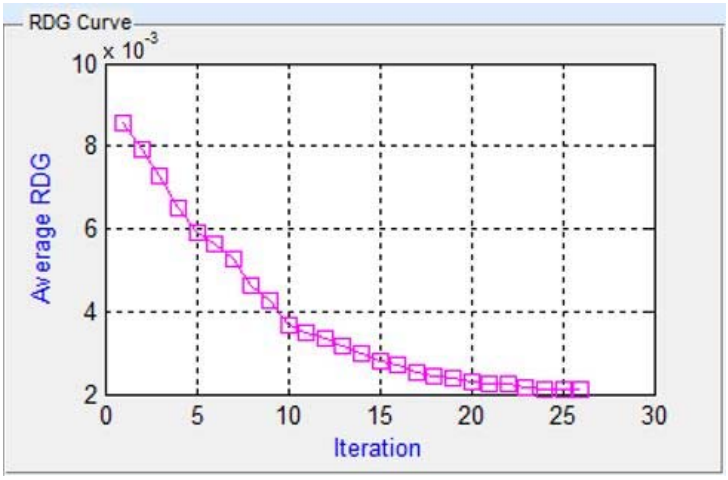
ตารางที่ 4.8 สรุปผลการคำนวณสำหรับการเดินเครื่องกำเนิดไฟฟ้าที่ค่าใช้จ่ายในการผลิตต่ำที่สุด

Results	The Best Solution
Dual Solution	557871.4989
Primal Solution	557826.2729
Total Cost of Single Fuel ($)	557826.2729
Total Cost of Multiple Fuels ($)	550762.8677
Parameter F	0.6045
Parameter CR	0.8694
RDG (10^{-5})	8.1069
Calculation Time (Sec)	808.7596

จากตารางที่ 4.8 พบว่าผลการทดลองที่ดีที่สุดจากการคำนวณทั้งหมด 100 รอบ ทำให้ได้ค่าใช้จ่ายในการผลิตที่ต่ำที่สุดแบบเชื้อเพลิงเดียวคือ 557826.2729 $ ค่า RDG ที่ได้คือ 8.1069 ค่า Parameter F ที่เหมาะสมอยู่ในช่วง [0.6-0.7] ค่า Parameter CR ที่เหมาะสมอยู่ในช่วง [0.7-0.8] และเวลาที่ใช้ในการคำนวณคือ 808.759577 วินาที



รูปที่ 4.2 กราฟการลู่เข้าหาคำตอบของราคาค่าใช้จ่ายในการผลิตในการแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า



รูปที่ 4.3 กราฟการลู่เข้าหาคำตอบของค่าช่องว่าง (Relative Duality Gap) ในการแก้ปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้า

ตารางที่ 4.9 แสดงค่าทางสถิติในการผลิตสำหรับการเดินเครื่องกำเนิดไฟฟ้าเป็นเวลา 24 ชั่วโมงในการคำนวณ 100 รอบ

Results	Best	Worst	Average
Total Cost of Single Fuel ($)	557826.2729	564556.9884	560518.8348
Total Cost of Multiple Fuels ($)	550762.8677	557425.0672	554093.9675
Total Loss (MW)	443.8584	431.8203	437.8394
RDG (10^{-5})	8.1069	9.2502	8.6786
Calculation Time(sec)	808.7596	336.5890	572.6730

ตารางที่ 4.10 ตารางเปรียบเทียบของการแก้ไขปัญหาระหว่างการพิจารณาการเลือกใช้เชื้อเพลิงเดียวกับการพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด

Results	พิจารณาการเลือกใช้เชื้อเพลิงเดียว	พิจารณาการเลือกใช้เชื้อเพลิงหลายชนิด
Fuel Cost ($)	553286.2729	546222.8677
Start Up Cost ($)	4540	4540
Total Cost ($)	557826.2729	550762.8677

จากตารางที่ 4.10 พบว่า เมื่อพิจารณาการเลือกใช้เชื้อเพลิงเดียว จะมีค่าราคาของเชื้อเพลิงคือ 553286.2729($) ค่าต้นทุนการเริ่มเดินเครื่องกำเนิดไฟฟ้าคือ 4540($) และค่าใช้จ่ายทั้งหมดในการผลิตคือ 557826.2729($) และเมื่อพิจารณาการเลือกใช้เชื้อเพลิงหลายชนิดจะมีค่าราคาของเชื้อเพลิงคือ 546222.8677($) ค่าต้นทุนการเริ่มเดินเครื่องกำเนิดไฟฟ้าคือ 4540($) และค่าใช้จ่ายทั้งหมดในการผลิตคือ 550762.8677($) จะเห็นได้ว่ามีค่าใช้จ่ายในการผลิตลดลงจากเดิม 7063.4052 ($) หรือลดลง 1.27%

4.2 วิเคราะห์ผลการทดลอง

ในการทดลองครั้งนี้เป็นการทดลองการแก้ไขปัญหาการจัดตารางการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงของระบบผลิตไฟฟ้า โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม โดยใช้เครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต ทำงานเป็นเวลา 24 ชั่วโมง และกำลังการผลิตสำรอง 10% ซึ่งจากการทดลองพบว่า ค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงมีค่าใช้จ่ายต่ำกว่าค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้าแบบเชื้อเพลิงเดียว เนื่องจาก เครื่องกำเนิดไฟฟ้ามีความหลากหลายทางเชื้อเพลิง จึงทำให้เครื่องกำเนิดไฟฟ้าสามารถเลือกเชื้อเพลิงที่ทำให้ค่าใช้จ่ายสำหรับการเดินเครื่องกำเนิดไฟฟ้ามีค่าที่ต่ำที่สุด

บทที่ 5
สรุปผลการทดลองและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

โครงงานนี้จะได้นำเสนอการประยุกต์ใช้โปรแกรมคอมพิวเตอร์ LREnDEA_UC เพื่อจัดตารางการเดินเครื่องกำเนิดไฟฟ้าของระบบผลิตกำลังไฟฟ้าแบบหลายเชื้อเพลิงโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม ซึ่งผู้จัดทำโครงงานได้ศึกษา พัฒนา และออกแบบหน้าต่างของโปรแกรมโดยใช้โปรแกรม MATLAB 2011a ร่วมกับ Graphical User Interface (GUI) ซึ่งได้กำหนดช่วงพารามิเตอร์ดังนี้ $X_{\max} = 20, X_{\min} = 16, F = 0.6, CR = 0.7$ เพื่อความสะดวกต่อการใช้งานกับคอมพิวเตอร์ โดยสามารถสรุปผลได้ดังนี้คือ

จากผลการทดลองที่ได้ทั้งหมด จะเห็นได้ว่าการจัดตารางเดินเครื่องกำเนิดไฟฟ้าแบบหลายเชื้อเพลิงโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม โดยการทดลองใช้เครื่องกำเนิดไฟฟ้า 10 ยูนิต ทำงานเป็นเวลา 24 ชั่วโมง ทำการทดลองโดยมีเงื่อนไขว่าค่า Relative Duality Gap (RDG) อยู่ในช่วง 10^{-5} มีกำลังสำรอง (Spinning Reserve) เท่ากับ 10% มีการคิดค่าความสูญเสียในสายส่ง และพิจารณาหลายเชื้อเพลิง ซึ่งผลลัพธ์ของต้นทุนในการผลิตกำลังไฟฟ้าที่ได้จะมีค่าต่ำที่สุดและการทดลองในแต่ละรอบการคำนวณจะมีค่าแตกต่างกันไป เนื่องจากแต่ละรอบในการคำนวณค่า λ ที่ได้ทำการสุ่มมีค่าไม่เหมือนกันจึงส่งผลให้ค่าราคาเชื้อเพลิงไม่เท่ากัน และการพิจารณาแบบหลายเชื้อเพลิงทำให้ได้ค่าใช้จ่ายในการเดินเครื่องกำเนิดไฟฟ้าต่ำกว่าการพิจารณาแบบเชื้อเพลิงเดียว

5.2 ข้อเสนอแนะ

จากการที่ได้ออกแบบ และเขียนโปรแกรมในการคำนวณการจัดตารางเดินเครื่องกำเนิดไฟฟ้า การประมวลผลของโปรแกรมจะเร็วได้ขึ้นอยู่กับจำนวนรอบในการคำนวณ ประสิทธิภาพของคอมพิวเตอร์ ผลจากตัวแปรต่างๆ ที่เกี่ยวข้องกับการคำนวณ และสำคัญที่สุดคือ รุ่นของโปรแกรมที่ใช้ ค่าที่ได้จากการทดลองนั้นอาจไม่ใช่ค่าใช้จ่ายรวมในการผลิตกำลังไฟฟ้าที่ต่ำที่สุด เนื่องจากการใช้ LREnDEA_UC เป็นการสุ่มค่าของ Initial ในแต่ละรอบการคำนวณจะมีค่าไม่เท่ากัน คำตอบที่ได้อาจยังไม่เป็นคำตอบที่ดีที่สุด ควรมีการทดลองซ้ำเพื่อให้ได้คำตอบที่ดีที่สุด

หากมีงานวิจัยเพื่อพัฒนาในครั้งต่อไปในอนาคต กรณีศึกษา(Case Study) ควรจะมีการพิจารณาในเรื่องของการใช้กำลังไฟฟ้าสำรองหมุนเวียน (Renewable Energy) เช่น พลังงานแสงอาทิตย์ พลังงานลม และพลังงานน้ำ เป็นต้น เพื่อเพิ่มประสิทธิภาพในการทำงานของโปรแกรมนี้ให้สามารถนำไปใช้จริงได้

เอกสารอ้างอิง

สร้อยพลู วรภพ; และ เขื่อนเพชร ไววิทย์. (2554). *การหาค่ายูนิตคอมมิตเมนต์ โดยใช้วิธีรีแล็กเซชันแบบลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม*. ปริญญานิพนธ์วิศวกรรมศาสตร์บัณฑิต. มหาวิทยาลัยศรีนครินทรวิโรฒ.

ผนึกดี กฤษณา; บัวบาน ชนกานต์; และ เรืองจิตร ชุติภา. (2557). *การวางแผนเดินเครื่องโรงไฟฟ้าของระบบผลิตไฟฟ้าโดยใช้วิธีลากรางจ์ร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึมแบบปรับตัวเองได้*. ปริญญานิพนธ์วิศวกรรมศาสตร์บัณฑิต. มหาวิทยาลัยศรีนครินทรวิโรฒ.

นิตย์ เพ็ชรรักษ์. (2557). การวิเคราะห์ระบบไฟฟ้ากำลัง. พิมพ์ครั้งที่ 1. กรุงเทพมหานคร:จุฬาลงกรณ์มหาวิทยาลัย, หน้า 61-112

นายพุฒิพงศ์ ชยันโต; นายภาณุวัฒน์ เสาวพันธ์; และนายอนวัช วาฤทธิ์. (2558). การจัดตารางเดินเครื่องเครื่องกำเนิดไฟฟ้าโดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม. ปริญญานิพนธ์วิศวกรรมศาสตร์บัณฑิต. มหาวิทยาลัยศรีนครินทรวิโรฒ

Murata, Daisuke; & Yamashiro, Susumu. (2005). Unit Commitment Scheduling by Lagrange Relaxation Method Taking into Account Transmission Losses. Electrical Engineering in Japan. 152(4): 27-33.

Chao-Lung Chiang, “Improved Genetic Algorithm for Power Economic Dispatch of Units with Valve-Point Effects and Multiple Fuels.” IEEE Trans. No Power System, vol. 20, no. 4, pp. 1690-1699, 2005.

Sum-Im, Thanathip. (2004). Economic Dispatch by Ant Colony Search Algorithm. *IEEE Conference on Cybernetics and Intelligent Systems*.

Sum-Im, Thanatip. (2009). *A Novel Differential Evolution Algorithmic Approach to Transmission Expansion Planning*. thesis. Brunel University.

Sum-Im, Thanathip. (2014). Lagrangian Relaxation Combined with Differential Evolution Algorithm for Unit Commitment Problem. *IEEE Emerging Technology and Factory Automation (ETFA)*. pp. 1-6.

ภาคผนวก

ภาคผนวก ก

ภาคผนวก ก ข้อมูลระบบไฟฟ้าที่ใช้ทดลอง

ตารางที่ ก1.1 แสดงรายละเอียดเกี่ยวกับข้อมูลเครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต

Unit	1	2	3	4	5
P_{max} (MW)	455	455	130	130	162
P_{min} (MW)	150	150	20	20	25
a ($/h)	1000	970	700	680	450
b (MWh)	16.19	17.26	16.60	16.50	19.70
c ($/MW2-h)	0.00048	0.00031	0.002	0.00211	0.00398
T_{up} (hr)	8	8	5	5	6
T_{down} (hr)	8	8	5	5	6
S_h(Hot Start)($)	4500	5000	550	560	900
S_c(Cold Start)($)	9000	10000	1100	1120	1800
$T_{Cold\ Start}$ (hr)	5	5	4	4	4
Initial State (hr)	8	8	-5	-5	-6

ตารางที่ ก1.1 แสดงรายละเอียดเกี่ยวกับข้อมูลเครื่องกำเนิดไฟฟ้าจำนวน 10 ยูนิต(ต่อ)

Unit	6	7	8	9	10
P_{max} (MW)	80	85	55	55	55
P_{min} (MW)	20	25	10	10	10
a ($/h)	370	480	660	665	670
b (MWh)	22.26	27.74	25.92	27.27	27.79
c ($/MW2-h)	0.00712	0.0079	0.00413	0.00222	0.00173
T_{up} (hr)	3	3	1	1	1
T_{down} (hr)	3	3	1	1	1
S_h(Hot Start)($)	170	260	30	30	30
S_c(Cold Start)($)	340	520	60	60	60
$T_{Cold\ Start}$ (hr)	2	2	0	0	0
Initial State (hr)	-3	-3	-1	-1	-1

ค่าสัมประสิทธิ์การสูญเสีย

$$B = \begin{bmatrix} 0.2179 & 0.1062 & -0.0027 & -0.0077 & 0.0076 & 0.0345 & 0.2179 & 0.1062 & -0.0027 & -0.0077 \\ 0.1062 & 0.1639 & -0.0009 & -0.0128 & 0.0040 & 0.0268 & 0.1062 & 0.1639 & -0.0009 & -0.0128 \\ -0.0027 & -0.0009 & 0.2586 & -0.1054 & -0.0983 & -0.0649 & -0.0027 & -0.0009 & 0.2586 & -0.1054 \\ -0.0077 & -0.0128 & -0.1054 & 0.1919 & 0.0720 & 0.0366 & -0.0077 & -0.0128 & -0.1054 & 0.1919 \\ 0.0076 & 0.0040 & -0.0983 & 0.0720 & 0.1601 & -0.0007 & 0.0076 & 0.0040 & -0.0983 & 0.0720 \\ 0.0345 & 0.0268 & -0.0649 & 0.0366 & -0.0007 & 0.2568 & 0.0345 & 0.0268 & -0.0649 & 0.0366 \\ 0.2179 & 0.1062 & -0.0027 & -0.0077 & 0.0076 & 0.0345 & 0.2179 & 0.1062 & -0.0027 & -0.0077 \\ 0.1062 & 0.1639 & -0.0009 & -0.0128 & 0.0040 & 0.0268 & 0.1062 & 0.1639 & -0.0009 & -0.0128 \\ -0.0027 & -0.0009 & 0.2586 & -0.1054 & -0.0983 & -0.0649 & -0.0027 & -0.0009 & 0.2586 & -0.1054 \\ -0.0077 & -0.0128 & -0.1054 & 0.1919 & 0.0720 & 0.0366 & -0.0077 & -0.0128 & -0.1054 & 0.1919 \end{bmatrix}$$

$$B_{i0} = \begin{bmatrix} -0.0001 & 0.0016 & -0.0043 & 0.0028 & 0.0012 & 0.0036 & -0.0001 & 0.0016 & -0.0043 & 0.0028 \end{bmatrix}$$

$$B_{00} = \begin{bmatrix} 0.1407 \end{bmatrix}$$

ตารางที่ ก1.2 แสดงข้อมูลความต้องการกำลังไฟฟ้าในเวลา 24 ชั่วโมง และกำลังการผลิตสำรอง 10%

Hour	Load (MW)	Reserve (MW)
1	700	70
2	750	75
3	850	85
4	950	95
5	1000	100
6	1100	110
7	1150	115
8	1200	120
9	1300	130
10	1300	130
11	1350	135
12	1450	145
13	1300	130
14	1300	130
15	1200	120
16	1050	105
17	1000	100
18	1100	110
19	1200	120
20	1300	130
21	1300	130
22	1100	110
23	900	90
24	800	80

ตารางที่ ก1.3 แสดงข้อมูลของเครื่องกำเนิดไฟฟ้าที่ใช้ทดสอบโดยพิจารณาแบบหลายเชื้อเพลิง

Unit	Generation		Fuel Type	Cost Coefficients		
	$P_{\min}$	$P_{\max}$		a	b	c
1	150	455	1	1000	16.19	0.00048
			2	1050	17.0	0.00051
2	150	455	1	970	17.26	0.00031
			2	945	16.94	0.00029
			3	1000	17.32	0.00035
3	20	130	1	700	16.6	0.0020
			2	680	17.0	0.0038
			3	750	15.9	0.0024
4	20	130	1	680	16.5	0.00211
			2	730	17.2	0.0028
			3	620	15.7	0.00198
5	162	250	1	450	19.7	0.00398
			2	500	19.9	0.0042
			3	560	20.2	0.0040

ตารางที่ ก1.3 แสดงข้อมูลของเครื่องกำเนิดไฟฟ้าที่ใช้ทดสอบโดยพิจารณาแบบหลายเชื้อเพลิง (ต่อ)

Unit	Generation		Fuel Type	Cost Coefficients		
	$P_{\min}$	$P_{\max}$		a	b	c
6	20	80	1	370	22.26	0.00712
			2	420	23.0	0.0080
			3	330	21.87	0.00685
7	25	85	1	480	27.74	0.00079
			2	420	27.56	0.00068
			3	500	27.95	0.00083
8	10	55	1	660	25.92	0.00431
			2	720	27.21	0.00532
			3	600	25.43	0.00392
9	10	55	1	665	27.27	0.00222
			2	700	28.21	0.00284
			3	680	27.56	0.00295
10	10	55	1	670	27.79	0.00173
			2	520	27.28	0.00154
			3	700	28.0	0.00240

ภาคผนวก ข

ภาคผนวก ข โปรแกรมที่ใช้ทดลอง

โปรแกรมหน้าต่าง GUI ที่ใช้ในการแก้ไขปัญหาการจัดตารางการเดินเครื่องเครื่องกำเนิดไฟฟ้า ของระบบการเดินเครื่องเครื่องกำเนิดไฟฟ้า10ยูนิต โดยใช้วิธีลากรางจ์รีแล็กเซชันร่วมกับวิธีดิฟเฟอเรนเชียลอีโวลูชันอัลกอริทึม พิจารณาการสูญเสียที่สายส่ง และพิจารณาความหลากหลายทางเชื้อเพลิง

```
function varargout = LRDEA_UC(varargin)
%LRDEA_UC M-file for LRDEA_UC.fig
%      LRDEA_UC, by itself, creates a new LRDEA_UC or raises the existing
%      singleton*.
%
%      H = LRDEA_UC returns the handle to a new LRDEA_UC or the handle to
%      the existing singleton*.
%
%      LRDEA_UC('Property','Value',...) creates a new LRDEA_UC using the
%      given property value pairs. Unrecognized properties are passed via
%      varargin to LRDEA_UC_OpeningFcn.  This calling syntax produces a
%      warning when there is an existing singleton*.
%
%      LRDEA_UC('CALLBACK') and LRDEA_UC('CALLBACK',hObject,...) call the
%      local function named CALLBACK in LRDEA_UC.M with the given input
%      arguments.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help LRDEA_UC
% Last Modified by GUIDE v2.5 24-Apr-2016 21:46:11
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
```

```
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @LRDEA_UC_OpeningFcn, ...
                   'gui_OutputFcn',  @LRDEA_UC_OutputFcn, ...
                   'gui_LayoutFcn',  [], ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before LRDEA_UC is made visible.
function LRDEA_UC_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)
% Choose default command line output for LRDEA_UC
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes LRDEA_UC wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = LRDEA_UC_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double
% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as a double
% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit5_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double
% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as a double
% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as a double
% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as a double


% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a double
% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as a double
% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit11 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit11 as a double
% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12 as a double
% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit13 as text
%        str2double(get(hObject,'String')) returns contents of edit13 as a double
% --- Executes during object creation, after setting all properties.
```

```
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit14 as text
%        str2double(get(hObject,'String')) returns contents of edit14 as a double
% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit15 as text
%        str2double(get(hObject,'String')) returns contents of edit15 as a double
% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit16 as text
%        str2double(get(hObject,'String')) returns contents of edit16 as a double
% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit17 as text
%        str2double(get(hObject,'String')) returns contents of edit17 as a double
% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit18 as text
%        str2double(get(hObject,'String')) returns contents of edit18 as a double
% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 % Hints: get(hObject,'String') returns contents of edit19 as text
%        str2double(get(hObject,'String')) returns contents of edit19 as a double
% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit20 as text
%        str2double(get(hObject,'String')) returns contents of edit20 as a double
% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit21 as text
%        str2double(get(hObject,'String')) returns contents of edit21 as a double
% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
function edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit22 as text
%        str2double(get(hObject,'String')) returns contents of edit22 as a double
% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit23 as text
%        str2double(get(hObject,'String')) returns contents of edit23 as a double
% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit24_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit24 as text
%        str2double(get(hObject,'String')) returns contents of edit24 as a double
% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

tic
%ข้อมูล Gennerator
%--------------------------------------------------------------------------
%gen1
gamma1=1000;
    B1=16.19;
    r1=0.00048;
    P1min=150;
    P1max=455;
%gen2
gamma2=970;
```

```
    B2=17.26;
    r2=0.00031;
    P2min=150;
    P2max=455;
%gen3
gamma3=700;
    B3=16.6;
    r3=0.002;
    P3min=20;
    P3max=130;
%gen4
gamma4=680;
    B4=16.5;
    r4=0.00211;
    P4min=20;
    P4max=130;
%gen5
gamma5=450;
    B5=19.7;
    r5=0.00398;
    P5min=162;
    P5max=250;
%gen6
gamma6=370;
    B6=22.26;
    r6=0.00712;
    P6min=20;
    P6max=80;
%gen7
gamma7=480;
    B7=27.74;
    r7=0.00079;
    P7min=25;
    P7max=85;
```

```
%gen8
gamma8=660;
    B8=25.92;
    r8=0.00431;
    P8min=10;
    P8max=55;
%gen9
gamma9=665;
    B9=27.27;
    r9=0.00222;
    P9min=10;
    P9max=55;
%gen10
gamma10=670;
    B10=27.79;
    r10=0.00173;
    P10min=10;
    P10max=55;
%-------------------------------------------------------------------------
initB=(10^-3)*[0.2179 0.1062 -0.0027 -0.0077 0.0076 0.0345 0.2179 0.1062 -0.0027 -0.0077
     0.1062 0.1639 -0.0009 -0.0128 0.0040 0.0268 0.1062 0.1639 -0.0009 -0.0128
     -0.0027 -0.0009 0.2586 -0.1054 -0.0983 -0.0649 -0.0027 -0.0009 0.2586 -0.1054
     -0.0077 -0.0128 -0.1054 0.1919 0.0720 0.0366 -0.0077 -0.0128 -0.1054 0.1919
     0.0076 0.0040 -0.0983 0.0720 0.1601 -0.0007 0.0076 0.0040 -0.0983 0.0720
     0.0345 0.0268 -0.0649 0.0366 -0.0007 0.2568 0.0345 0.0268 -0.0649 0.0366
     0.2179 0.1062 -0.0027 -0.0077 0.0076 0.0345 0.2179 0.1062 -0.0027 -0.0077
     0.1062 0.1639 -0.0009 -0.0128 0.0040 0.0268 0.1062 0.1639 -0.0009 -0.0128
     -0.0027 -0.0009 0.2586 -0.1054 -0.0983 -0.0649 -0.0027 -0.0009 0.2586 -0.1054
     -0.0077 -0.0128 -0.1054 0.1919 0.0720 0.0366 -0.0077 -0.0128 -0.1054 0.1919];
initB00=0.1407;
initBi0=[-0.0001 0.0016 -0.0043 0.0028 0.0012 0.0036 -0.0001 0.0016 -0.0043 0.0028];
%-------------------------------------------------------------------------
%percentreserve
percentr = str2double(get(handles.edit25,'string'));
```

```
PL=[700,750,850,950,1000,1100,1150,1200,1300,1300,1350,1450,1300,1300,1200,1050,1000,1100,1200,1
300,1300,1100,900,800];
PL(1,1)=str2double(get(handles.edit1,'string'));
PL(1,2)=str2double(get(handles.edit2,'string'));
PL(1,3)=str2double(get(handles.edit3,'string'));
PL(1,4)=str2double(get(handles.edit4,'string'));
PL(1,5)=str2double(get(handles.edit5,'string'));
PL(1,6)=str2double(get(handles.edit6,'string'));
PL(1,7)=str2double(get(handles.edit7,'string'));
PL(1,8)=str2double(get(handles.edit8,'string'));
PL(1,9)=str2double(get(handles.edit9,'string'));
PL(1,10)=str2double(get(handles.edit10,'string'));
PL(1,11)=str2double(get(handles.edit11,'string'));
PL(1,12)=str2double(get(handles.edit12,'string'));
PL(1,13)=str2double(get(handles.edit13,'string'));
PL(1,14)=str2double(get(handles.edit14,'string'));
PL(1,15)=str2double(get(handles.edit15,'string'));
PL(1,16)=str2double(get(handles.edit16,'string'));
PL(1,17)=str2double(get(handles.edit17,'string'));
PL(1,18)=str2double(get(handles.edit18,'string'));
PL(1,19)=str2double(get(handles.edit19,'string'));
PL(1,20)=str2double(get(handles.edit20,'string'));
PL(1,21)=str2double(get(handles.edit21,'string'));
PL(1,22)=str2double(get(handles.edit22,'string'));
PL(1,23)=str2double(get(handles.edit23,'string'));
PL(1,24)=str2double(get(handles.edit24,'string'));
TotalPL=0;
TotalPL=sum(PL);
%--------------------------------------------------------------------
%Initial value
InitST_Cost=[9000,10000,1100,1120,1800,340,520,60,60,60];
InitStateHr=[8,8,-5,-5,-6,-3,-3,-1,-1,-1];
TOnMin=[8,8,5,5,6,3,3,1,1,1];
TDownMin=[8,8,5,5,6,3,3,1,1,1];
```

```
%---------------------------------------------------------------------
%PL reserve
PLReserve=PL*percentr;
T=[1:24];
disp('-------------Display Results--------------');
disp(['Initial Lagrange Multiplier']);
disp('Power Demand(PL)');
disp(' T     PL');
disp(num2str([T;PL]'));
keep=1;
check=1;
stop=0;
rdgkeep=1;
RDGgraph=[];
Numjstar=[];
Numgraph1=[];
rdgala=[];
Jstarala=[];
%intitial---------------------------------------------------------------
D = 24;
rdgstop = str2double(get(handles.edit0026,'string'));
ngs = str2double(get(handles.edit27,'string'));
nogm = str2double(get(handles.edit28,'string'));
Xmax = str2double(get(handles.edit29,'string'));
Xmin = str2double(get(handles.edit30,'string'));
F = str2double(get(handles.edit31,'string'));
Cr1 = str2double(get(handles.edit32,'string'));
np1 = str2double(get(handles.edit33,'string'));
Cr=ones(1,D).*Cr1;
np=D.*np1;
indi1=(ones(np,D).*Xmin)+((rand(np,D)).*((ones(np,D).*Xmax)-(ones(np,D).*Xmin)));
%รอบgeneration
for aaa3=1:nogm
%---------------------------------------------------------------------
```

```
%รอบการคำนวณRDG
for ibest=1:np
lambda=indi1(ibest,:);
    %Generator No.1
    P1=(lambda-B1)/(2*r1);
    U1=(P1>=P1max);
    U1old=U1;
    PD1old=U1old.*P1max;
    %Generator No.2
    P2=(lambda-B2)/(2*r2);
    U2=(P2>=P2max);
    U2old=U2;
    PD2old=U2old.*P2max;
    %Generator No.3
    P3=(lambda-B3)/(2*r3);
    U3=(P3>=P3max);
    U3old=U3;
    PD3old=U3old.*P3max;
    %Generator No.4
    P4=(lambda-B4)/(2*r4);
    U4=(P4>=P4max);
    U4old=U4;
    PD4old=U4old.*P4max;
    %Generator No.5
    P5=(lambda-B5)/(2*r5);
    U5=(P5>=P5max);
    U5old=U5;
    PD5old=U5old.*P5max;
    %Generator No.6
    P6=(lambda-B6)/(2*r6);
    U6=(P6>=P6max);
    U6old=U6;
    PD6old=U6old.*P6max;
    %Generator No.7
```

```
    P7=(lambda-B7)/(2*r7);
    U7=(P7>=P7max);
    U7old=U7;
    PD7old=U7old.*P7max;
    %Generator No.8
    P8=(lambda-B8)/(2*r8);
    U8=(P8>=P8max);
    U8old=U8;
    PD8old=U8old.*P8max;
    %Generator No.9
    P9=(lambda-B9)/(2*r9);
    U9=(P9>=P9max);
    U9old=U9;
    PD9old=U9old.*P9max;
    %Generator No.10
    P10=(lambda-B10)/(2*r10);
    U10=(P10>=P10max);
    U10old=U10;
    PD10old=U10old.*P10max;
    Pmax=[P1max,P2max,P3max,P4max,P5max,P6max,P7max,P8max,P9max,P10max];
    U=[U1;U2;U3;U4;U5;U6;U7;U8;U9;U10]';
    Unew=U;
    %----------------------------Update U Step 1----------------------------
    %Check if sum Pmax >PL and replace with Pmax and Update U

PU=[U1*P1max;U2*P2max;U3*P3max;U4*P4max;U5*P5max;U6*P6max;U7*P7max;U8*P8max;U9*P9max;
U10*P10max]';
    lessPLindex=find(sum(PU')<(PL+PLReserve));
    lessPLvalue=U(lessPLindex,:);
PLsel=[P1max*lessPLvalue(:,1),P2max*lessPLvalue(:,2),P3max*lessPLvalue(:,3),P4max*lessPLvalue(:,4),P
5max*lessPLvalue(:,5),P6max*lessPLvalue(:,6),P7max*lessPLvalue(:,7),P8max*lessPLvalue(:,8),P9max*les
sPLvalue(:,9),P10max*lessPLvalue(:,10)];
    difvalue=PL(lessPLindex)-sum(PLsel');
```

```
    fillIndex=lessPLvalue~=1;
PLfill=[P1max*fillIndex(:,1),P2max*fillIndex(:,2),P3max*fillIndex(:,3),P4max*fillIndex(:,4),P5max*fillIndex(:,
5),P6max*fillIndex(:,6),P7max*fillIndex(:,7),P8max*fillIndex(:,8),P9max*fillIndex(:,9),P10max*fillIndex(:,10)]
;
    for i=1:size(PLfill,1)
        srt=sort(PLfill(i,:),'descend');
        cums=cumsum(srt);
        needfill=srt(1:find(cums==min(min(cums(difvalue(i)<cums)))));
        for j=1:length(needfill)
            if needfill(j)~=0
                Unew(lessPLindex(i),PLfill(i,:)==needfill(j))=1;
             end
        end
    end
    U=Unew;

U1=U(:,1)';U2=U(:,2)';U3=U(:,3)';U4=U(:,4)';U5=U(:,5)';U6=U(:,6)';U7=U(:,7)';U8=U(:,8)';U9=U(:,9)';U10=U(:,10
)';
    %-------------------------Update U Step 2-----------------------------
    %Check TOnmin and TDownMin value
    numon=zeros(1,10);
    numdown=zeros(1,10);
    numon(find(InitStateHr>0))=InitStateHr(find(InitStateHr>0));
    for i=1:size(U,1)
       for j=1:size(U,2)
            if U(i,j)==1
               if numon(j)<TOnMin(j)
                    numon(j)=numon(j)+1;
                end
            else
                if numon(j)<TOnMin(j)&numon(j)>0
                    Unew(i,j)=1;
                    numon(j)=numon(j)+1;
                else
```

```
                numon(j)=0;
            end
        end
    end

end

Ux=Unew;
numdown(find(InitStateHr<0))=abs(InitStateHr(find(InitStateHr<0)));
for i=1:24
    for j=1:size(Ux,2)
        if Ux(i,j)==0
            if i==24 & numdown(j)< TDownMin(j)
                st=i-numdown(j);
                if i-numdown(j)<=0
                st=1;
                end
                Unew(st:i,j)=1;
            end
                numdown(j)=numdown(j)+1;
        else
            if numdown(j)<TDownMin(j)
                st=i-numdown(j);
                if i-numdown(j)<=0
                st=1;
                end
                Unew(st:i,j)=1;
            end
            numdown(j)=0;
        end
    end
end
%PUnew=[Unew(:,1)*P1max,Unew(:,2)*P2max,Unew(:,3)*P3max,Unew(:,4)*P4max,Unew(:,5)*P5max,Une
w(:,6)*P6max,Unew(:,7)*P7max,Unew(:,8)*P8max,Unew(:,9)*P9max,Unew(:,10)*P10max];
```

```
    U=Unew;
U1=U(:,1)';U2=U(:,2)';U3=U(:,3)';U4=U(:,4)';U5=U(:,5)';U6=U(:,6)';U7=U(:,7)';U8=U(:,8)';U9=U(:,9)';U10=U(:,10
)';
    P1(find((U1old~=U1)==1))=P1max;
    P2(find((U2old~=U2)==1))=P2max;
    P3(find((U3old~=U3)==1))=P3max;
    P4(find((U4old~=U4)==1))=P4max;
    P5(find((U5old~=U5)==1))=P5max;
    P6(find((U6old~=U6)==1))=P6max;
    P7(find((U7old~=U7)==1))=P7max;
    P8(find((U8old~=U7)==1))=P8max;
    P9(find((U9old~=U9)==1))=P9max;
    P10(find((U10old~=U10)==1))=P10max;

    P1new=P1;
    P1new(P1>=P1max)=P1max;
    P1new(P1<=P1min)=P1min;
    CP1=gamma1+B1*P1new+r1*P1new.^2;
    GenTable1=[T;P1;U1;CP1-lambda.*P1new]';
   % disp(' T              P              U      CP-lambda*P');
   % disp(num2str(GenTable1));
    P2new=P2;
    P2new(P2>=P2max)=P2max;
    P2new(P2<=P2min)=P2min;
    CP2=gamma2+B2*P2new+r2*P2new.^2;
    GenTable2=[T;P2;U2;CP2-lambda.*P2new]';
   % disp(' T              P              U      CP-lambda*P');
   % disp(num2str(GenTable2));
    P3new=P3;
    P3new(P3>=P3max)=P3max;
    P3new(P3<=P3min)=P3min;
    CP3=gamma3+B3*P3new+r3*P3new.^2;
    GenTable3=[T;P3;U3;CP3-lambda.*P3new]';
  %  disp(' T              P              U      CP-lambda*P');
```

```
%  disp(num2str(GenTable3));
  P4new=P4;
  P4new(P4>=P4max)=P4max;
  P4new(P4<=P4min)=P4min;
  CP4=gamma4+B4*P4new+r4*P4new.^2;
  GenTable4=[T;P4;U4;CP4-lambda.*P4new]';
 % disp(' T                P                U       CP-lambda*P');
 % disp(num2str(GenTable4));
  P5new=P5;
  P5new(P5>=P5max)=P5max;
  P5new(P5<=P5min)=P5min;
  CP5=gamma5+B5*P5new+r5*P5new.^2;
  GenTable5=[T;P5;U5;CP5-lambda.*P5new]';
% disp(' T                P                U       CP-lambda*P');
% disp(num2str(GenTable5));
  P6new=P6;
  P6new(P6>=P6max)=P6max;
  P6new(P6<=P6min)=P6min;
  CP6=gamma6+B6*P6new+r6*P6new.^2;
  GenTable6=[T;P6;U6;CP6-lambda.*P6new]';
% disp(' T                P                U       CP-lambda*P');
% disp(num2str(GenTable6));
  P7new=P7;
  P7new(P7>=P7max)=P7max;
  P7new(P7<=P7min)=P7min;
  CP7=gamma7+B7*P7new+r7*P7new.^2;
  GenTable7=[T;P7;U7;CP7-lambda.*P7new]';
 % disp(' T                P                U       CP-lambda*P');
 % disp(num2str(GenTable7));
  P8new=P8;
  P8new(P8>=P8max)=P8max;
  P8new(P8<=P8min)=P8min;
  CP8=gamma8+B8*P8new+r8*P8new.^2;
  GenTable8=[T;P8;U8;CP8-lambda.*P8new]';
```

```
% disp(' T              P              U      CP-lambda*P');
% disp(num2str(GenTable8));
 P9new=P9;
 P9new(P9>=P9max)=P9max;
 P9new(P9<=P9min)=P9min;
 CP9=gamma9+B9*P9new+r9*P9new.^2;
 GenTable9=[T;P9;U9;CP9-lambda.*P9new]';
% disp(' T              P              U      CP-lambda*P');
% disp(num2str(GenTable9));
 P10new=P10;
 P10new(P10>=P10max)=P10max;
 P10new(P10<=P10min)=P10min;
 CP10=gamma10+B10*P10new+r10*P10new.^2;
 GenTable10=[T;P10;U10;CP10-lambda.*P10new]';
% disp(' T              P              U      CP-lambda*P');
% disp(num2str(GenTable10));
 PD1=U1.*P1max;
 PD2=U2.*P2max;
 PD3=U3.*P3max;
 PD4=U4.*P4max;
 PD5=U5.*P5max;
 PD6=U6.*P6max;
 PD7=U7.*P7max;
 PD8=U8.*P8max;
 PD9=U9.*P9max;
 PD10=U10.*P10max;
 P=[PD1;PD2;PD3;PD4;PD5;PD6;PD7;PD8;PD9;PD10]';
 %////////////////////q*,J*,RDG Calculation/////////////////////////
 InitState=InitStateHr>0;
 U_ST1=[InitState(1),U1];
 U_ST2=[InitState(2),U2];
 U_ST3=[InitState(3),U3];
 U_ST4=[InitState(4),U4];
 U_ST5=[InitState(5),U5];
```

```
    U_ST6=[InitState(6),U6];
    U_ST7=[InitState(7),U7];
    U_ST8=[InitState(8),U8];
    U_ST9=[InitState(9),U9];
    U_ST10=[InitState(10),U10];
%-------------------------------------------------------------------------
    %Find Start Up Position
numStartup1=0;numStartup2=0;numStartup3=0;numStartup4=0;numStartup5=0;numStartup6=0;numS
tartup7=0;numStartup8=0;numStartup9=0;numStartup10=0;
    k=1;
    for i=2:25
        startup1(k)=(U_ST1(i)-U_ST1(i-1))==1;
        startup2(k)=(U_ST2(i)-U_ST2(i-1))==1;
        startup3(k)=(U_ST3(i)-U_ST3(i-1))==1;
        startup4(k)=(U_ST4(i)-U_ST4(i-1))==1;
        startup5(k)=(U_ST5(i)-U_ST5(i-1))==1;
        startup6(k)=(U_ST6(i)-U_ST6(i-1))==1;
        startup7(k)=(U_ST7(i)-U_ST7(i-1))==1;
        startup8(k)=(U_ST8(i)-U_ST8(i-1))==1;
        startup9(k)=(U_ST9(i)-U_ST9(i-1))==1;
        startup10(k)=(U_ST10(i)-U_ST10(i-1))==1;
        k=k+1;
    end
ST_Cost1=zeros(1,24);ST_Cost2=zeros(1,24);ST_Cost3=zeros(1,24);ST_Cost4=zeros(1,24);ST_Cost5=zero
s(1,24);ST_Cost6=zeros(1,24);ST_Cost7=zeros(1,24);ST_Cost8=zeros(1,24);ST_Cost9=zeros(1,24);ST_Cost
10=zeros(1,24);

sumofclose1=0;sumofclose2=0;sumofclose3=5;sumofclose4=5;sumofclose5=6;sumofclose6=3;sumofc
lose7=3;sumofclose8=1;sumofclose9=1;sumofclose10=1;
    for i=1:24
        if U1(i)==0
            sumofclose1=sumofclose1+1;
        end
        if U2(i)==0
```

```
    sumofclose2=sumofclose2+1;
end
if U3(i)==0
    sumofclose3=sumofclose3+1;
end
if U4(i)==0
    sumofclose4=sumofclose4+1;
end
if U5(i)==0
    sumofclose5=sumofclose5+1;
end
if U6(i)==0
    sumofclose6=sumofclose6+1;
end
if U7(i)==0
    sumofclose7=sumofclose7+1;
end
if U8(i)==0
    sumofclose8=sumofclose8+1;
end
if U9(i)==0
    sumofclose9=sumofclose9+1;
end
if U10(i)==0
    sumofclose10=sumofclose10+1;
end
if startup1(i)==1
    if sumofclose1>5
        ST_Cost1(i)=startup1(i)*InitST_Cost(1);
    else
        ST_Cost1(i)=startup1(i)*InitST_Cost(1)/2;
    end
    sumofclose1=0;
else
```

```
    ST_Cost1(i)=0;
end
if startup2(i)==1
    if sumofclose2>5
        ST_Cost2(i)=startup2(i)*InitST_Cost(2);
    else
        ST_Cost2(i)=startup2(i)*InitST_Cost(2)/2;
    end
    sumofclose2=0;
else
    ST_Cost2(i)=0;
end
if startup3(i)==1
    if sumofclose3>4
        ST_Cost3(i)=startup3(i)*InitST_Cost(3);
    else
        ST_Cost3(i)=startup3(i)*InitST_Cost(3)/2;
    end
    sumofclose3=0;
else
    ST_Cost3(i)=0;
end
if startup4(i)==1
    if sumofclose4>4
        ST_Cost4(i)=startup4(i)*InitST_Cost(4);
    else
        ST_Cost4(i)=startup4(i)*InitST_Cost(4)/2;
    end
    sumofclose4=0;
else
    ST_Cost4(i)=0;
end
if startup5(i)==1
    if sumofclose5>4
```

```
        ST_Cost5(i)=startup5(i)*InitST_Cost(5);
    else
        ST_Cost5(i)=startup5(i)*InitST_Cost(5)/2;
    end
    sumofclose5=0;
else
    ST_Cost5(i)=0;
end
if startup6(i)==1
    if sumofclose6>2
        ST_Cost6(i)=startup6(i)*InitST_Cost(6);
    else
        ST_Cost6(i)=startup6(i)*InitST_Cost(6)/2;
    end
    sumofclose6=0;
else
    ST_Cost6(i)=0;
end
if startup7(i)==1
    if sumofclose7>2
        ST_Cost7(i)=startup7(i)*InitST_Cost(7);
    else
        ST_Cost7(i)=startup7(i)*InitST_Cost(7)/2;
    end
    sumofclose7=0;
else
    ST_Cost7(i)=0;
end
if startup8(i)==1
    if sumofclose8>0
        ST_Cost8(i)=startup8(i)*InitST_Cost(8);
    else
        ST_Cost8(i)=startup8(i)*InitST_Cost(8);
    end
```

```
            sumofclose8=0;
        else
            ST_Cost8(i)=0;
        end
        if startup9(i)==1
            if sumofclose9>0
                ST_Cost9(i)=startup9(i)*InitST_Cost(9);
            else
                ST_Cost9(i)=startup9(i)*InitST_Cost(9);
            end
            sumofclose9=0;
        else
            ST_Cost9(i)=0;
        end
        if startup10(i)==1
            if sumofclose10>0
                ST_Cost10(i)=startup10(i)*InitST_Cost(10);
            else
                ST_Cost10(i)=startup10(i)*InitST_Cost(10);
            end
            sumofclose10=0;
        else
            ST_Cost10(i)=0;
        end
    end
%-------------------------------------------------------------------------
    %Start-Up Cost Calculation
ST_CostByTime=ST_Cost1+ST_Cost2+ST_Cost3+ST_Cost4+ST_Cost5+ST_Cost6+ST_Cost7+ST_Cost8+S
T_Cost9+ST_Cost10;
ST_CostByUnit=[sum(ST_Cost1),sum(ST_Cost2),sum(ST_Cost3),sum(ST_Cost4),sum(ST_Cost5),sum(ST_C
ost6),sum(ST_Cost7),sum(ST_Cost8),sum(ST_Cost9),sum(ST_Cost10)];
%ST_CostByTime=startup1*InitST_Cost(1)+startup2*InitST_Cost(2)+startup3*InitST_Cost(3)+startup4*Ini
tST_Cost(4)...
```

```
    %
+startup5*InitST_Cost(5)+startup6*InitST_Cost(6)+startup7*InitST_Cost(7)+startup8*InitST_Cost(8)+start
up9*InitST_Cost(9)+startup10*InitST_Cost(10);
%ST_CostByUnit=[sum(startup1*InitST_Cost(1)),sum(startup2*InitST_Cost(2)),sum(startup3*InitST_Cost(3
)),...
    %
sum(startup4*InitST_Cost(4)),sum(startup5*InitST_Cost(5)),sum(startup6*InitST_Cost(6)),sum(startup7*Ini
tST_Cost(7)),sum(startup8*InitST_Cost(8)),sum(startup9*InitST_Cost(9)),sum(startup10*InitST_Cost(10))];
%---------------------------------------------------------------------------
    %q* Calculation
    qstar=sum((CP1-lambda.*P1new).*U1)+sum((CP2-lambda.*P2new).*U2)+sum((CP3-
lambda.*P3new).*U3)+sum((CP4-lambda.*P4new).*U4)...
        +sum((CP5-lambda.*P5new).*U5)+sum((CP6-lambda.*P6new).*U6)+sum((CP7-
lambda.*P7new).*U7)+sum((CP8-lambda.*P8new).*U8)+sum((CP9-lambda.*P9new).*U9)+sum((CP10-
lambda.*P10new).*U10)+sum(ST_CostByTime)+sum(lambda.*PL);
%---------------------------------------------------------------------------
    %Pedc (Power Economic Dispacth Calculation)
 gamma=[gamma1,gamma2,gamma3,gamma4,gamma5,gamma6,gamma7,gamma8,gamma9,gamma10];
    B=[B1,B2,B3,B4,B5,B6,B7,B8,B9,B10];
    r=[r1,r2,r3,r4,r5,r6,r7,r8,r9,r10];
    Pmin=[P1min,P2min,P3min,P4min,P5min,P6min,P7min,P8min,P9min,P10min];
    Pmax=[P1max,P2max,P3max,P4max,P5max,P6max,P7max,P8max,P9max,P10max];
    PmaxSel=zeros(10);
    Pedc=zeros(24,10);
    CPedcByTime=zeros(24,1);
    InitCost=0; %use if all U=0
    for t =1:24
        PmaxSel=zeros(10);
        PminSel=zeros(10);
        PmaxSel=Pmax(U(t,:)==1);
        PminSel=Pmin(U(t,:)==1);
        gammaSel=gamma(U(t,:)==1);
        BSel=B(U(t,:)==1);
        rSel=r(U(t,:)==1);
```

```
index=find(U(t,:)==1);
if (sum(U(t,:))==1)
    if(sum(PmaxSel)>=PL(t))
        Pedc(t,index)=PL(t);
        CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
        CPedcByTime(t)=CPSel;
    else
        CPedcByTime(t)=InitCost;
        Pedc(t,:)=0;
    end
elseif(sum(U(t,:))==0)||sum(PmaxSel)<PL(t)
    CPedcByTime(t)=InitCost;
    Pedc(t,:)=0;
else
    ptemp=PL(t);
    sumcp=zeros(length(PmaxSel),1);
    pmax=PmaxSel;
    pmin=PminSel;
    gammasel=gammaSel;
    bsel=BSel;
    rsel=rSel;
    PedcTempAll=zeros(length(pmax),length(pmax));
    for i=1:length(pmax)
        difp=sum(pmax)-ptemp;
        pedcTemp=pmax;
        for j=length(pmax):-1:1
            if(difp<pmax(j))
                if(pmax(j)-difp)>pmin(j)
                    pedcTemp(j)=pmax(j)-difp;
                    difp=0;
                else
                    pedcTemp(j)=pmin(j);
                    difp=difp-pmax(j)+pmin(j);
                end
```

```
                else
                    pedcTemp(j)=pmin(j);
                    difp=difp-pmax(j)+pmin(j);
                end
            end
            PedcTempAll(i,:)=pedcTemp;
            cp=0;
            cp=gammasel+bsel.*pedcTemp+rsel.*pedcTemp.^2;
            sumcp(i)=sum(cp);
            pmax=circshift(pmax,[1 1]);
            pmin=circshift(pmin,[1 1]);
            gammasel=circshift(gammasel,[1 1]);
            bsel=circshift(bsel,[1 1]);
            rsel=circshift(rsel,[1 1]);
        end
        lsindex=find(sumcp==min(sumcp));
        pedcTemp=circshift(PedcTempAll(lsindex(1),:),[1,-(lsindex(1)-1)]);

        Pedc(t,index)=pedcTemp;
        CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
        CPedcByTime(t)=sum(CPSel);
    end
end
%-------------------------------------------------------------------------
%Transmission Line Loss Calculation
Plossindex=zeros(1,24);
for t =1:24;
    Ploss=0;
    pwg=sum(Pedc);
    NG=length(pwg);
        for i=1:NG-1
            for j=i+1:NG
                Ploss=Ploss+Pedc(t,i)*initB(i,j)*Pedc(t,j);
            end
```

```
            Ploss=Ploss+initBi0(i)*Pedc(t,i);
        end
        Plossindex(t)=Ploss+initB00;
    end
%---------------------------------------------------------------------
%Pedc (Power Economic Dispacth Include Transmission Line Loss Calculation)
    PmaxSel=zeros(10);
    Pedc=zeros(24,10);
    CPedcByTime=zeros(24,1);
    InitCost=0;
    for t =1:24
        PmaxSel=zeros(10);
        PminSel=zeros(10);
        PmaxSel=Pmax(U(t,:)==1);
        PminSel=Pmin(U(t,:)==1);
        gammaSel=gamma(U(t,:)==1);
        BSel=B(U(t,:)==1);
        rSel=r(U(t,:)==1);
        index=find(U(t,:)==1);
        if (sum(U(t,:))==1)
            if(sum(PmaxSel)>=PL(t)+Plossindex(t))
                Pedc(t,index)=PL(t)+Plossindex(t);
                CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
                CPedcByTime(t)=CPSel;
            else
                CPedcByTime(t)=InitCost;
                Pedc(t,:)=0;
            end
        elseif(sum(U(t,:))==0)||sum(PmaxSel)<PL(t)+Plossindex(t)
            CPedcByTime(t)=InitCost;
            Pedc(t,:)=0;
        else
            ptemp=PL(t);
            sumcp=zeros(length(PmaxSel),1);
```

```
pmax=PmaxSel;
pmin=PminSel;
gammasel=gammaSel;
bsel=BSel;
rsel=rSel;
PedcTempAll=zeros(length(pmax),length(pmax));
for i=1:length(pmax)
    difp=sum(pmax)-ptemp-Plossindex(t);
    pedcTemp=pmax;
    for j=length(pmax):-1:1
        if(difp<pmax(j))
            if(pmax(j)-difp)>pmin(j)
                pedcTemp(j)=pmax(j)-difp;
                difp=0;
            else
                pedcTemp(j)=pmin(j);
                difp=difp-pmax(j)+pmin(j);
            end
        else
            pedcTemp(j)=pmin(j);
            difp=difp-pmax(j)+pmin(j);
        end
    end
    PedcTempAll(i,:)=pedcTemp;
    cp=0;
    cp=gammasel+bsel.*pedcTemp+rsel.*pedcTemp.^2;
    sumcp(i)=sum(cp);
    pmax=circshift(pmax,[1 1]);
    pmin=circshift(pmin,[1 1]);
    gammasel=circshift(gammasel,[1 1]);
    bsel=circshift(bsel,[1 1]);
    rsel=circshift(rsel,[1 1]);
end
lsindex=find(sumcp==min(sumcp));
```

```
            pedcTemp=circshift(PedcTempAll(lsindex(1),:),[1,-(lsindex(1)-1)]);

            Pedc(t,index)=pedcTemp;
            CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
            CPedcByTime(t)=sum(CPSel);
        end
    end
%-------------------------------------------------------------------------
    %J* Calculation
    Jstar=sum(CPedcByTime)+sum(ST_CostByTime);
    %RDG Calculation
    rdg=abs(Jstar-qstar)/qstar;
    JStar(ibest)=Jstar;
    QStar(ibest)=qstar;
    RDG(ibest)=rdg;
    PedcTABLE(:,:,ibest)=[T;Pedc'];
    %Diplay Value
    CostTABLE(:,:,ibest)=[T;CPedcByTime';ST_CostByTime;(CPedcByTime'+ST_CostByTime)]';
    %TotalCostTABLE(iter)=['     Total    ',num2str(sum(CPedcByTime)),'
',num2str(sum(ST_CostByTime)),'    ',num2str(sum(CPedcByTime)+sum(ST_CostByTime))];
    TBC=num2str([T;CPedcByTime';ST_CostByTime;(CPedcByTime'+ST_CostByTime)]');
%-------------------------------------------------------------------------
    %RDG Constraints
    if rdg<rdgstop & keep==1
        Jstarkeep=Jstar;
        Qstarkeep=qstar;
        rdgkeep=rdg;
        keep=keep+1;
        lambdakeep=lambda;
        Numgen=aaa3;
        Numround=ibest;
        keepF=F;
        keepCr=Cr;
        uctablekeep=[T;U1;U2;U3;U4;U5;U6;U7;U8;U9;U10]';
```

```
    qtablekeep=num2str([T;PD1;PD2;PD3;PD4;PD5;PD6;PD7;PD8;PD9;P10]');
    edtable=([T;Pedc']');
    lossTable=num2str([T;PL;Plossindex]');
    Plossindex1=Plossindex;
    SumPlossindex=sum(Plossindex);
    totaledtable=TBC;
    ccc=num2str(sum(CPedcByTime));
    ccc1=num2str(sum(ST_CostByTime));
    ccc2=num2str(sum(CPedcByTime)+sum(ST_CostByTime));
    chikeep=([T;CPedcByTime';ST_CostByTime;(CPedcByTime'+ST_CostByTime)]');
end
if rdg<rdgstop & keep==2
    if Jstar<Jstarkeep
        Jstarkeep=Jstar;
        Qstarkeep=qstar;
        rdgkeep=rdg;
        lambdakeep=lambda;
        Numgen=aaa3;
        Numround=ibest;
        keepF=F;
        keepCr=Cr;
        uctablekeep=[T;U1;U2;U3;U4;U5;U6;U7;U8;U9;U10]';
        qtablekeep=num2str([T;PD1;PD2;PD3;PD4;PD5;PD6;PD7;PD8;PD9;P10]');
        edtable=([T;Pedc']');
        lossTable=num2str([T;PL;Plossindex]');
        Plossindex1=Plossindex;
        SumPlossindex=sum(Plossindex);
        totaledtable=TBC;
        ccc=num2str(sum(CPedcByTime));
        ccc1=num2str(sum(ST_CostByTime));
        ccc2=num2str(sum(CPedcByTime)+sum(ST_CostByTime));
        chikeep=([T;CPedcByTime';ST_CostByTime;(CPedcByTime'+ST_CostByTime)]');
    end
end
```

```
%-------------------------------------------------------------------------
rdgala=[rdgala,rdg];
Jstarala=[Jstarala,Jstar];
%-------------------------------------------------------------------------
end
%-------------------------------------------------------------------------
rdgalamin=mean(rdgala);
Jstaralamin=mean(Jstarala);
Numgraph=aaa3;
Numgraph1=[Numgraph1,Numgraph];
RDGgraph=[RDGgraph,rdgalamin];
Numjstar=[Numjstar,Jstaralamin];
rdgala=[];
Jstarala=[];
%-------------------------------------------------------------------------
    if rdgkeep<rdgstop & check==1
        Jstarcheck=Jstarkeep;
        check=check+1;
    end
    if rdgkeep<rdgstop & check==2
        if Jstarcheck==Jstarkeep
            stop=stop+1;
        else
            Jstarcheck=Jstarkeep;
            stop=0;
        end
    end
    if stop==ngs
        break
    end
%-------------------------------------------------------------------------
for aaa2=1:np
%-------------------------------------------------------------------------
%/////////////// PART DIFFERENTIAL EVOLUTION ALGORITHM ////////////////////
```

```
%สุ่มค่า r1 r2 r3
%-------------------------------------------------------------------------
tgv=indi1(aaa2,:);
fmtv=randperm(np);
xr1=fmtv(1,1);
xr2=fmtv(1,2);
xr3=fmtv(1,3);
while xr1==aaa2 & xr2==aaa2 & xr3==aaa2
fmtv=randperm(np);

xr1=fmtv(1,1);
xr2=fmtv(1,2);
xr3=fmtv(1,3);
end

xr11=indi1(xr1,:);
xr22=indi1(xr2,:);
xr33=indi1(xr3,:);

%ค่าหา mutant vector
%-------------------------------------------------------------------------
mtv=abs(xr11+((xr22-xr33).*F));
%หาค่า trial vector
%-------------------------------------------------------------------------
Rcro=rand(1,D);
for rcc=1:D
tgv1=tgv(1,rcc);
mtv1=mtv(1,rcc);
Cr1=Cr(1,rcc);
Rcro1=Rcro(1,rcc);
if Rcro1 <= Cr1
    tavv=mtv1;
else
    tavv=tgv1;
```

```
end
if ~isempty(tavv)
trial(rcc) = tavv;
end
end
%---------------------------------------------------------------------------
for aaa=1:2
    if aaa==1
        lambda=tgv;
    else
        lambda=trial;
    end
%---------------------------------------------------------------------------
    %qstar
    T=[1:24];

    %Generator No.1
    P1=(lambda-B1)/(2*r1);
    U1=(P1>=P1max);
    U1old=U1;
    PD1old=U1old.*P1max;
    %Generator No.2
    P2=(lambda-B2)/(2*r2);
    U2=(P2>=P2max);
    U2old=U2;
    PD2old=U2old.*P2max;
    %Generator No.3
    P3=(lambda-B3)/(2*r3);
    U3=(P3>=P3max);
    U3old=U3;
    PD3old=U3old.*P3max;
    %Generator No.4
    P4=(lambda-B4)/(2*r4);
    U4=(P4>=P4max);
```

```
U4old=U4;
PD4old=U4old.*P4max;
%Generator No.5
P5=(lambda-B5)/(2*r5);
U5=(P5>=P5max);
U5old=U5;
PD5old=U5old.*P5max;
%Generator No.6
P6=(lambda-B6)/(2*r6);
U6=(P6>=P6max);
U6old=U6;
PD6old=U6old.*P6max;
%Generator No.7
P7=(lambda-B7)/(2*r7);
U7=(P7>=P7max);
U7old=U7;
PD7old=U7old.*P7max;
%Generator No.8
P8=(lambda-B8)/(2*r8);
U8=(P8>=P8max);
U8old=U8;
PD8old=U8old.*P8max;
%Generator No.9
gamma9=665;
P9=(lambda-B9)/(2*r9);
U9=(P9>=P9max);
U9old=U9;
PD9old=U9old.*P9max;
%Generator No.10
P10=(lambda-B10)/(2*r10);
U10=(P10>=P10max);
U10old=U10;
PD10old=U10old.*P10max;
Pmax=[P1max,P2max,P3max,P4max,P5max,P6max,P7max,P8max,P9max,P10max];
```

```
    U=[U1;U2;U3;U4;U5;U6;U7;U8;U9;U10]';
    Unew=U;
    %-------------------------Update U Step 1-----------------------------
    %Check if sum Pmax >PL and replace with Pmax and Update U

PU=[U1*P1max;U2*P2max;U3*P3max;U4*P4max;U5*P5max;U6*P6max;U7*P7max;U8*P8max;U9*P9max;
U10*P10max]';
    lessPLindex=find(sum(PU')<(PL+PLReserve));
    lessPLvalue=U(lessPLindex,:);

PLsel=[P1max*lessPLvalue(:,1),P2max*lessPLvalue(:,2),P3max*lessPLvalue(:,3),P4max*lessPLvalue(:,4),P
5max*lessPLvalue(:,5),P6max*lessPLvalue(:,6),P7max*lessPLvalue(:,7),P8max*lessPLvalue(:,8),P9max*les
sPLvalue(:,9),P10max*lessPLvalue(:,10)];
    difvalue=PL(lessPLindex)-sum(PLsel');
    fillIndex=lessPLvalue~=1;

PLfill=[P1max*fillIndex(:,1),P2max*fillIndex(:,2),P3max*fillIndex(:,3),P4max*fillIndex(:,4),P5max*fillIndex(:,
5),P6max*fillIndex(:,6),P7max*fillIndex(:,7),P8max*fillIndex(:,8),P9max*fillIndex(:,9),P10max*fillIndex(:,10)]
;
    for i=1:size(PLfill,1)
        srt=sort(PLfill(i,:),'descend');
        cums=cumsum(srt);
        needfill=srt(1:find(cums==min(min(cums(difvalue(i)<cums)))));
        for j=1:length(needfill)
            if needfill(j)~=0
                Unew(lessPLindex(i),PLfill(i,:)==needfill(j))=1;
             end
        end
    end
    U=Unew;
U1=U(:,1)';U2=U(:,2)';U3=U(:,3)';U4=U(:,4)';U5=U(:,5)';U6=U(:,6)';U7=U(:,7)';U8=U(:,8)';U9=U(:,9)';U10=U(:,10
)';
    %-----------------------------Update U Step 2---------------------------
    %Check TOnmin and TDownMin value
```

```
 numon=zeros(1,10);
 numdown=zeros(1,10);
 numon(find(InitStateHr>0))=InitStateHr(find(InitStateHr>0));
 for i=1:size(U,1)
    for j=1:size(U,2)
        if U(i,j)==1
           if numon(j)<TOnMin(j)
               numon(j)=numon(j)+1;
            end
        else
            if numon(j)<TOnMin(j)&numon(j)>0
               Unew(i,j)=1;
               numon(j)=numon(j)+1;
            else
               numon(j)=0;
            end
        end
    end
 end
Ux=Unew;
numdown(find(InitStateHr<0))=abs(InitStateHr(find(InitStateHr<0)));
for i=1:24
    for j=1:size(Ux,2)
        if Ux(i,j)==0
            if i==24 & numdown(j)< TDownMin(j)
               st=i-numdown(j);
               if i-numdown(j)<=0
               st=1;
               end
               Unew(st:i,j)=1;
            end
               numdown(j)=numdown(j)+1;
        else
            if numdown(j)<TDownMin(j)
```

```
                st=i-numdown(j);
                if i-numdown(j)<=0
                st=1;
                end
                Unew(st:i,j)=1;
            end
            numdown(j)=0;
        end
    end
end

%PUnew=[Unew(:,1)*P1max,Unew(:,2)*P2max,Unew(:,3)*P3max,Unew(:,4)*P4max,Unew(:,5)*P5max,Unew(:,6)*P6max,Unew(:,7)*P7max,Unew(:,8)*P8max,Unew(:,9)*P9max,Unew(:,10)*P10max];
    U=Unew;

U1=U(:,1)';U2=U(:,2)';U3=U(:,3)';U4=U(:,4)';U5=U(:,5)';U6=U(:,6)';U7=U(:,7)';U8=U(:,8)';U9=U(:,9)';U10=U(:,10)';
    P1(find((U1old~=U1)==1))=P1max;
    P2(find((U2old~=U2)==1))=P2max;
    P3(find((U3old~=U3)==1))=P3max;
    P4(find((U4old~=U4)==1))=P4max;
    P5(find((U5old~=U5)==1))=P5max;
    P6(find((U6old~=U6)==1))=P6max;
    P7(find((U7old~=U7)==1))=P7max;
    P8(find((U8old~=U7)==1))=P8max;
    P9(find((U9old~=U9)==1))=P9max;
    P10(find((U10old~=U10)==1))=P10max;
    P1new=P1;
    P1new(P1>=P1max)=P1max;
    P1new(P1<=P1min)=P1min;
    CP1=gamma1+B1*P1new+r1*P1new.^2;
    GenTable1=[T;P1;U1;CP1-lambda.*P1new]';
   % disp(' T                P                U       CP-lambda*P');
   % disp(num2str(GenTable1));
```

```
  P2new=P2;
  P2new(P2>=P2max)=P2max;
  P2new(P2<=P2min)=P2min;
  CP2=gamma2+B2*P2new+r2*P2new.^2;
  GenTable2=[T;P2;U2;CP2-lambda.*P2new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable2));
  P3new=P3;
  P3new(P3>=P3max)=P3max;
  P3new(P3<=P3min)=P3min;
  CP3=gamma3+B3*P3new+r3*P3new.^2;
  GenTable3=[T;P3;U3;CP3-lambda.*P3new]';
%  disp(' T              P              U      CP-lambda*P');
%  disp(num2str(GenTable3));
  P4new=P4;
  P4new(P4>=P4max)=P4max;
  P4new(P4<=P4min)=P4min;
  CP4=gamma4+B4*P4new+r4*P4new.^2;
  GenTable4=[T;P4;U4;CP4-lambda.*P4new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable4));
  P5new=P5;
  P5new(P5>=P5max)=P5max;
  P5new(P5<=P5min)=P5min;
  CP5=gamma5+B5*P5new+r5*P5new.^2;
  GenTable5=[T;P5;U5;CP5-lambda.*P5new]';
% disp(' T              P              U      CP-lambda*P');
% disp(num2str(GenTable5));
  P6new=P6;
  P6new(P6>=P6max)=P6max;
  P6new(P6<=P6min)=P6min;
  CP6=gamma6+B6*P6new+r6*P6new.^2;
  GenTable6=[T;P6;U6;CP6-lambda.*P6new]';
% disp(' T              P              U      CP-lambda*P');
```

```
% disp(num2str(GenTable6));
  P7new=P7;
  P7new(P7>=P7max)=P7max;
  P7new(P7<=P7min)=P7min;
  CP7=gamma7+B7*P7new+r7*P7new.^2;
  GenTable7=[T;P7;U7;CP7-lambda.*P7new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable7));
  P8new=P8;
  P8new(P8>=P8max)=P8max;
  P8new(P8<=P8min)=P8min;
  CP8=gamma8+B8*P8new+r8*P8new.^2;
  GenTable8=[T;P8;U8;CP8-lambda.*P8new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable8));
  P9new=P9;
  P9new(P9>=P9max)=P9max;
  P9new(P9<=P9min)=P9min;
  CP9=gamma9+B9*P9new+r9*P9new.^2;
  GenTable9=[T;P9;U9;CP9-lambda.*P9new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable9));
  P10new=P10;
  P10new(P10>=P10max)=P10max;
  P10new(P10<=P10min)=P10min;
  CP10=gamma10+B10*P10new+r10*P10new.^2;
  GenTable10=[T;P10;U10;CP10-lambda.*P10new]';
 % disp(' T              P              U      CP-lambda*P');
 % disp(num2str(GenTable10));
  PD1=U1.*P1max;
  PD2=U2.*P2max;
  PD3=U3.*P3max;
  PD4=U4.*P4max;
  PD5=U5.*P5max;
```

```
    PD6=U6.*P6max;
    PD7=U7.*P7max;
    PD8=U8.*P8max;
    PD9=U9.*P9max;
    PD10=U10.*P10max;
    %//////////////////q*,J*,RDG Calculation/////////////////////////////
    InitState=InitStateHr>0;
    U_ST1=[InitState(1),U1];
    U_ST2=[InitState(2),U2];
    U_ST3=[InitState(3),U3];
    U_ST4=[InitState(4),U4];
    U_ST5=[InitState(5),U5];
    U_ST6=[InitState(6),U6];
    U_ST7=[InitState(7),U7];
    U_ST8=[InitState(8),U8];
    U_ST9=[InitState(9),U9];
    U_ST10=[InitState(10),U10];
%-------------------------------------------------------------------------
    %Find Start Up Position
numStartup1=0;numStartup2=0;numStartup3=0;numStartup4=0;numStartup5=0;numStartup6=0;numS
tartup7=0;numStartup8=0;numStartup9=0;numStartup10=0;
    k=1;
    for i=2:25
        startup1(k)=(U_ST1(i)-U_ST1(i-1))==1;
        startup2(k)=(U_ST2(i)-U_ST2(i-1))==1;
        startup3(k)=(U_ST3(i)-U_ST3(i-1))==1;
        startup4(k)=(U_ST4(i)-U_ST4(i-1))==1;
        startup5(k)=(U_ST5(i)-U_ST5(i-1))==1;
        startup6(k)=(U_ST6(i)-U_ST6(i-1))==1;
        startup7(k)=(U_ST7(i)-U_ST7(i-1))==1;
        startup8(k)=(U_ST8(i)-U_ST8(i-1))==1;
        startup9(k)=(U_ST9(i)-U_ST9(i-1))==1;
        startup10(k)=(U_ST10(i)-U_ST10(i-1))==1;
        k=k+1;
```

```
    end
ST_Cost1=zeros(1,24);ST_Cost2=zeros(1,24);ST_Cost3=zeros(1,24);ST_Cost4=zeros(1,24);ST_Cost5=zeros(1,24);ST_Cost6=zeros(1,24);ST_Cost7=zeros(1,24);ST_Cost8=zeros(1,24);ST_Cost9=zeros(1,24);ST_Cost10=zeros(1,24);
sumofclose1=0;sumofclose2=0;sumofclose3=5;sumofclose4=5;sumofclose5=6;sumofclose6=3;sumofclose7=3;sumofclose8=1;sumofclose9=1;sumofclose10=1;
    for i=1:24
        if U1(i)==0
            sumofclose1=sumofclose1+1;
        end
        if U2(i)==0
            sumofclose2=sumofclose2+1;
        end
        if U3(i)==0
            sumofclose3=sumofclose3+1;
        end
        if U4(i)==0
            sumofclose4=sumofclose4+1;
        end
        if U5(i)==0
            sumofclose5=sumofclose5+1;
        end
        if U6(i)==0
            sumofclose6=sumofclose6+1;
        end
        if U7(i)==0
            sumofclose7=sumofclose7+1;
        end
        if U8(i)==0
            sumofclose8=sumofclose8+1;
        end
        if U9(i)==0
            sumofclose9=sumofclose9+1;
        end
```

```
if U10(i)==0
    sumofclose10=sumofclose10+1;
end
if startup1(i)==1
    if sumofclose1>5
        ST_Cost1(i)=startup1(i)*InitST_Cost(1);
    else
        ST_Cost1(i)=startup1(i)*InitST_Cost(1)/2;
    end
    sumofclose1=0;
else
    ST_Cost1(i)=0;
end
if startup2(i)==1
    if sumofclose2>5
        ST_Cost2(i)=startup2(i)*InitST_Cost(2);
    else
        ST_Cost2(i)=startup2(i)*InitST_Cost(2)/2;
    end
    sumofclose2=0;
else
    ST_Cost2(i)=0;
end
if startup3(i)==1
    if sumofclose3>4
        ST_Cost3(i)=startup3(i)*InitST_Cost(3);
    else
        ST_Cost3(i)=startup3(i)*InitST_Cost(3)/2;
    end
    sumofclose3=0;
else
    ST_Cost3(i)=0;
end
if startup4(i)==1
```

```
    if sumofclose4>4
        ST_Cost4(i)=startup4(i)*InitST_Cost(4);
    else
        ST_Cost4(i)=startup4(i)*InitST_Cost(4)/2;
    end
    sumofclose4=0;
else
    ST_Cost4(i)=0;
end
if startup5(i)==1
    if sumofclose5>4
        ST_Cost5(i)=startup5(i)*InitST_Cost(5);
    else
        ST_Cost5(i)=startup5(i)*InitST_Cost(5)/2;
    end
    sumofclose5=0;
else
    ST_Cost5(i)=0;
end
if startup6(i)==1
    if sumofclose6>2
        ST_Cost6(i)=startup6(i)*InitST_Cost(6);
    else
        ST_Cost6(i)=startup6(i)*InitST_Cost(6)/2;
    end
    sumofclose6=0;
else
    ST_Cost6(i)=0;
end
if startup7(i)==1
    if sumofclose7>2
        ST_Cost7(i)=startup7(i)*InitST_Cost(7);
    else
        ST_Cost7(i)=startup7(i)*InitST_Cost(7)/2;
```

```
    end
    sumofclose7=0;
else
    ST_Cost7(i)=0;
end
if startup8(i)==1
    if sumofclose8>0
        ST_Cost8(i)=startup8(i)*InitST_Cost(8);
    else
        ST_Cost8(i)=startup8(i)*InitST_Cost(8);
    end
    sumofclose8=0;
else
    ST_Cost8(i)=0;
end
if startup9(i)==1
    if sumofclose9>0
        ST_Cost9(i)=startup9(i)*InitST_Cost(9);
    else
        ST_Cost9(i)=startup9(i)*InitST_Cost(9);
    end
    sumofclose9=0;
else
    ST_Cost9(i)=0;
end
if startup10(i)==1
    if sumofclose10>0
        ST_Cost10(i)=startup10(i)*InitST_Cost(10);
    else
        ST_Cost10(i)=startup10(i)*InitST_Cost(10);
    end
    sumofclose10=0;
else
    ST_Cost10(i)=0;
```

```
        end
    end
%-------------------------------------------------------------------------
    %Start-Up Cost Calculation

ST_CostByTime=ST_Cost1+ST_Cost2+ST_Cost3+ST_Cost4+ST_Cost5+ST_Cost6+ST_Cost7+ST_Cost8+S
T_Cost9+ST_Cost10;

ST_CostByUnit=[sum(ST_Cost1),sum(ST_Cost2),sum(ST_Cost3),sum(ST_Cost4),sum(ST_Cost5),sum(ST_C
ost6),sum(ST_Cost7),sum(ST_Cost8),sum(ST_Cost9),sum(ST_Cost10)];

%ST_CostByTime=startup1*InitST_Cost(1)+startup2*InitST_Cost(2)+startup3*InitST_Cost(3)+startup4*Ini
tST_Cost(4)...
    %
+startup5*InitST_Cost(5)+startup6*InitST_Cost(6)+startup7*InitST_Cost(7)+startup8*InitST_Cost(8)+start
up9*InitST_Cost(9)+startup10*InitST_Cost(10);

%ST_CostByUnit=[sum(startup1*InitST_Cost(1)),sum(startup2*InitST_Cost(2)),sum(startup3*InitST_Cost(3
)),...
    %
sum(startup4*InitST_Cost(4)),sum(startup5*InitST_Cost(5)),sum(startup6*InitST_Cost(6)),sum(startup7*Ini
tST_Cost(7)),sum(startup8*InitST_Cost(8)),sum(startup9*InitST_Cost(9)),sum(startup10*InitST_Cost(10))];
%-------------------------------------------------------------------------
    %q* Calculation
    qstar=sum((CP1-lambda.*P1new).*U1)+sum((CP2-lambda.*P2new).*U2)+sum((CP3-
lambda.*P3new).*U3)+sum((CP4-lambda.*P4new).*U4)...
        +sum((CP5-lambda.*P5new).*U5)+sum((CP6-lambda.*P6new).*U6)+sum((CP7-
lambda.*P7new).*U7)+sum((CP8-lambda.*P8new).*U8)+sum((CP9-lambda.*P9new).*U9)+sum((CP10-
lambda.*P10new).*U10)+sum(ST_CostByTime)+sum(lambda.*PL);
%-------------------------------------------------------------------------
    %Pedc (Power Economic Dispacth Calculation)

gamma=[gamma1,gamma2,gamma3,gamma4,gamma5,gamma6,gamma7,gamma8,gamma9,gamma10];
    B=[B1,B2,B3,B4,B5,B6,B7,B8,B9,B10];
```

```
r=[r1,r2,r3,r4,r5,r6,r7,r8,r9,r10];
Pmin=[P1min,P2min,P3min,P4min,P5min,P6min,P7min,P8min,P9min,P10min];
Pmax=[P1max,P2max,P3max,P4max,P5max,P6max,P7max,P8max,P9max,P10max];
PmaxSel=zeros(10);
Pedc=zeros(24,10);
CPedcByTime=zeros(24,1);
InitCost=0; %use if all U=0
for t =1:24
    PmaxSel=zeros(10);
    PminSel=zeros(10);
    PmaxSel=Pmax(U(t,:)==1);
    PminSel=Pmin(U(t,:)==1);
    gammaSel=gamma(U(t,:)==1);
    BSel=B(U(t,:)==1);
    rSel=r(U(t,:)==1);
    index=find(U(t,:)==1);
    if (sum(U(t,:))==1)
        if(sum(PmaxSel)>=PL(t))
            Pedc(t,index)=PL(t);
            CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
            CPedcByTime(t)=CPSel;
        else
            CPedcByTime(t)=InitCost;
            Pedc(t,:)=0;
        end
    elseif(sum(U(t,:))==0)||sum(PmaxSel)<PL(t)
        CPedcByTime(t)=InitCost;
        Pedc(t,:)=0;
    else
        ptemp=PL(t);
        sumcp=zeros(length(PmaxSel),1);
        pmax=PmaxSel;
        pmin=PminSel;
        gammasel=gammaSel;
```

```
bsel=BSel;
rsel=rSel;
PedcTempAll=zeros(length(pmax),length(pmax));
for i=1:length(pmax)
    difp=sum(pmax)-ptemp;
    pedcTemp=pmax;
    for j=length(pmax):-1:1
        if(difp<pmax(j))
            if(pmax(j)-difp)>pmin(j)
                pedcTemp(j)=pmax(j)-difp;
                difp=0;
            else
                pedcTemp(j)=pmin(j);
                difp=difp-pmax(j)+pmin(j);
            end
        else
            pedcTemp(j)=pmin(j);
            difp=difp-pmax(j)+pmin(j);
        end
    end
    PedcTempAll(i,:)=pedcTemp;
    cp=0;
    cp=gammasel+bsel.*pedcTemp+rsel.*pedcTemp.^2;
    sumcp(i)=sum(cp);
    pmax=circshift(pmax,[1 1]);
    pmin=circshift(pmin,[1 1]);
    gammasel=circshift(gammasel,[1 1]);
    bsel=circshift(bsel,[1 1]);
    rsel=circshift(rsel,[1 1]);
end
lsindex=find(sumcp==min(sumcp));
pedcTemp=circshift(PedcTempAll(lsindex(1),:),[1,-(lsindex(1)-1)]);

Pedc(t,index)=pedcTemp;
```

```
            CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
            CPedcByTime(t)=sum(CPSel);
        end
    end
%---------------------------------------------------------------------------
    %Transmission Line Loss Calculation
    Plossindex=zeros(1,24);
    for t =1:24;
        Ploss=0;
        pwg=sum(Pedc);
        NG=length(pwg);
            for i=1:NG-1
                for j=i+1:NG
                        Ploss=Ploss+Pedc(t,i)*initB(i,j)*Pedc(t,j);
                end
                Ploss=Ploss+initBi0(i)*Pedc(t,i);
            end
            Plossindex(t)=Ploss+initB00;
    end
%---------------------------------------------------------------------------
%Pedc (Power Economic Dispacth Include Transmission Line Loss Calculation)
    PmaxSel=zeros(10);
    Pedc=zeros(24,10);
    CPedcByTime=zeros(24,1);
    InitCost=0;
    for t =1:24
        PmaxSel=zeros(10);
        PminSel=zeros(10);
        PmaxSel=Pmax(U(t,:)==1);
        PminSel=Pmin(U(t,:)==1);
        gammaSel=gamma(U(t,:)==1);
        BSel=B(U(t,:)==1);
        rSel=r(U(t,:)==1);
        index=find(U(t,:)==1);
```

```
if (sum(U(t,:))==1)
    if(sum(PmaxSel)>=PL(t)+Plossindex(t))
        Pedc(t,index)=PL(t)+Plossindex(t);
        CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
        CPedcByTime(t)=CPSel;
    else
        CPedcByTime(t)=InitCost;
        Pedc(t,:)=0;
    end
elseif(sum(U(t,:))==0)||sum(PmaxSel)<PL(t)+Plossindex(t)
    CPedcByTime(t)=InitCost;
    Pedc(t,:)=0;
else
    ptemp=PL(t);
    sumcp=zeros(length(PmaxSel),1);
    pmax=PmaxSel;
    pmin=PminSel;
    gammasel=gammaSel;
    bsel=BSel;
    rsel=rSel;
    PedcTempAll=zeros(length(pmax),length(pmax));
    for i=1:length(pmax)
        difp=sum(pmax)-ptemp-Plossindex(t);
        pedcTemp=pmax;
        for j=length(pmax):-1:1
            if(difp<pmax(j))
                if(pmax(j)-difp)>pmin(j)
                    pedcTemp(j)=pmax(j)-difp;
                    difp=0;
                else
                    pedcTemp(j)=pmin(j);
                    difp=difp-pmax(j)+pmin(j);
                end
            else
```

```
                pedcTemp(j)=pmin(j);
                difp=difp-pmax(j)+pmin(j);
            end
        end
        PedcTempAll(i,:)=pedcTemp;
        cp=0;
        cp=gammasel+bsel.*pedcTemp+rsel.*pedcTemp.^2;
        sumcp(i)=sum(cp);
        pmax=circshift(pmax,[1 1]);
        pmin=circshift(pmin,[1 1]);
        gammasel=circshift(gammasel,[1 1]);
        bsel=circshift(bsel,[1 1]);
        rsel=circshift(rsel,[1 1]);
    end
    lsindex=find(sumcp==min(sumcp));
    pedcTemp=circshift(PedcTempAll(lsindex(1),:),[1,-(lsindex(1)-1)]);

    Pedc(t,index)=pedcTemp;
    CPSel=gammaSel+BSel.*Pedc(t,index)+rSel.*Pedc(t,index).^2;
    CPedcByTime(t)=sum(CPSel);
  end
 end
%-------------------------------------------------------------------------
 %J* calculation
 Jstar=sum(CPedcByTime)+sum(ST_CostByTime);
 rdg=abs(Jstar-qstar)/qstar;
  if aaa==1
     rdgx1=rdg;
     jstarx1=Jstar;
     qstarx1=qstar;
  else
     rdgx2=rdg;
     jstarx2=Jstar;
     qstarx2=qstar;
```

```
    end
end

if rdgx2<rdgx1  & jstarx2<jstarx1
    pop=trial;
    else
    pop=tgv;
end

if aaa2==1
gendea=pop;
else
gendea=[gendea;pop];
end
end
%-------------------------------------------------------------------------
T1=rand();
T2=rand();
if T1<0.1
    F=0.6+(rand(1)*(1-0.6));
end
if T2<0.1
    Cr=(0.7+(rand(1)*(0.9-0.7)))*ones(1,D);
end
indi1=gendea;
end
%-------------------------------------------------------------------------
%Find Min Max of Fuel in edTable
[m,n] = size(edtable);
for i = 1:n-1
        FuelTable(:,i) = edtable(:,i+1);
end
%-----------------------------------------------
%multi gen. info
```

```
FuelC = zeros(1,3);
gamma=[0.00048,0.00051,0.00031,0.00029,0.00035,0.0020,0.0038,0.0024,0.00211,0.0028,0.00198,0.0039
8,0.0042,0.0040,0.00712,0.0080,0.00685,0.00079,0.00068,0.00083,0.00431,0.00532,0.00392,0.00222,0.00
284,0.00295,0.00173,0.00154,0.00240]
alpha=[1000,1050,970,945,1000,700,680,750,680,730,620,450,500,560,370,420,330,480,420,500,660,720,
600,665,700,680,670,520,700]
beta=[16.19,17.00,17.26,16.94,17.32,16.6,17.0,15.9,16.5,17.2,15.7,19.7,19.9,20.2,22.26,23.00,21.87,27.74,
27.56,27.95,25.92,27.21,25.43,27.27,28.21,27.56,27.79,27.28,28.0]
Pmin=[150,150,20,20,162,20,25,10,10,10]
Pmax=[455,455,130,130,250,80,85,55,55,55]
arvalue = 1;
unitvalue = 1;
FuelCost = zeros(24,10);
FuelNum = zeros(24,10);
for t = 1:24;
    unitvalue = 1;
    for i = 1:29
        if i <= 2
            if (FuelTable(t,unitvalue) ~= 0)
            FuelC(1,arvalue) = alpha(i) + beta(i)*FuelTable(t,unitvalue) +
gamma(i)*(FuelTable(t,unitvalue)^2);
            else
                FuelC(1,arvalue) = 0;
                FuelNum(t,unitvalue) = 0;
            end
            if arvalue == 2
                if FuelC(1,arvalue) < FuelC(1,arvalue-1)
                    FuelCost(t,unitvalue) = FuelC(1,arvalue);
                    FuelNum(t,unitvalue) = arvalue;

                else
                    FuelCost(t,unitvalue) = FuelC(1,arvalue-1);
                    FuelNum(t,unitvalue) = arvalue-1;
                end
```

```
            FuelC = zeros(1,3);
            arvalue = 0;
            unitvalue = unitvalue + 1;
         end
         arvalue = arvalue + 1;
      else
         if (FuelTable(t,unitvalue) ~= 0)
          FuelC(1,arvalue) = alpha(i) + beta(i)*FuelTable(t,unitvalue) +
gamma(i)*(FuelTable(t,unitvalue)^2);
          else
            FuelC(1,arvalue) = 0;
            FuelNum(t,unitvalue) = 0;
         end
         if arvalue == 3
            if FuelC(1,arvalue) < FuelC(1,arvalue-2) && FuelC(1,arvalue) < FuelC(1,arvalue-1)
               FuelCost(t,unitvalue) = FuelC(1,arvalue);
               FuelNum(t,unitvalue) = arvalue;

            elseif FuelC(1,arvalue-1) < FuelC(1,arvalue-2) && FuelC(1,arvalue-1) < FuelC(1,arvalue)
               FuelCost(t,unitvalue) = FuelC(1,arvalue-1);
               FuelNum(t,unitvalue) = arvalue-1;

            elseif FuelC(1,arvalue-2) < FuelC(1,arvalue) && FuelC(1,arvalue-2) < FuelC(1,arvalue-1)
               FuelCost(t,unitvalue) = FuelC(1,arvalue-2);
               FuelNum(t,unitvalue) = arvalue-2;
            end
            FuelC = zeros(1,3);
            arvalue = 0;
            unitvalue = unitvalue + 1;
         end
         arvalue = arvalue + 1;
      end
   end
end
```

```
SumCost = zeros(24,1);
for i = 1:24
    SumCost(i,1) = sum(FuelCost(i,:));
end

MultiFuelData = zeros(24,24);
k = 1;
l = 1;
for i= 1:22
    if i == 1
        for j = 1:24
            MultiFuelData(j,1) = j;
        end
    elseif i<=21
        if mod(i,2)==0
            MultiFuelData(:,i) = FuelCost(:,k);
            k = k+1;
        else
            MultiFuelData(:,i) = FuelNum(:,l);
            l = l+1;
        end
    else
        MultiFuelData(:,i) = (SumCost(:,1));
        MultiFuelData(:,i+1) = (chikeep(:,3));
        MultiFuelData(:,i+2) = (MultiFuelData(:,i)+MultiFuelData(:,i+1));
    end
end

TotalMulCost = (sum(MultiFuelData(:,22)));
TotalStart = (sum(MultiFuelData(:,23)));
Totalall = (sum(MultiFuelData(:,24)));
displayData = zeros(24,4);
displayData(:,1) =  (MultiFuelData(:,1));
displayData(:,2) =  (MultiFuelData(:,22));
```

```
displayData(:,3) =  (MultiFuelData(:,23));
displayData(:,4) =  (MultiFuelData(:,24));
disp(MultiFuelData(:,22));

%---------------------------------------------------------------------
%แสดงผลลัพธ์
axes(handles.axes1);
plot(Numgraph1,RDGgraph,'-sm')
grid;
xlabel('Iteration','color','b');
ylabel('Average RDG','color','b');
axes(handles.axes2);
plot(Numgraph1,Numjstar,'-sg')
grid;
xlabel('Iteration','color','b');
ylabel('Average Total Cost','color','b');
lambdakeep1=lambdakeep
disp(['Numgen = ',num2str(Numgen)]);
disp(['Numround = ',num2str(Numround)]);
disp(['QStar = ',num2str(Qstarkeep)]);
disp(['JStar = ',num2str(Jstarkeep)]);
disp(['RDG = ',num2str(rdgkeep)]);
disp(['Parameter F = ',num2str(keepF)]);
disp(['Parameter Cr = ',num2str(keepCr)]);
keepCr1=keepCr(1,1);
set(handles.text35,'string',Numgen);
set(handles.text39,'string',num2str(Qstarkeep));
set(handles.text41,'string',num2str(Jstarkeep));
set(handles.text49,'string',rdgkeep);
set(handles.text43,'string',keepF);
set(handles.text45,'string',keepCr1);
set(handles.anstime,'string',ccc2);
set(handles.text54,'string',ccc);
set(handles.text57,'string',ccc1);
```

```
set(handles.text65,'string',num2str(SumPlossindex));
set(handles.text68,'string',num2str(TotalPL));
set(handles.uitable7,'Data',uctablekeep);
set(handles.uitable8,'Data',edtable);
set(handles.uitable13,'Data',lambdakeep.');
set(handles.uitable15,'Data',Plossindex1.');
set(handles.uitable19,'Data',MultiFuelData);
set(handles.text71,'string',num2str(TotalMulCost));
set(handles.text73,'string',num2str(TotalStart));
set(handles.text75,'string',num2str(Totalall));

    disp(['================Power Transmission Loss=================']);
    disp(' T   Power Demand(PL)  Power Transmission Line Loss(PLoss)');
    disp(lossTable);
    disp(['Total Power Demand = ',num2str(TotalPL)]);
    disp(['Total Power Transmission Line Loss = ',num2str(SumPlossindex)]);
    disp(['=========================================================']);
    disp(['====================Unit Commitment Table=====================']);
    disp('     T    U1    U2    U3   U4    U5    U6    U7    U8    U9    U10');
    disp(uctablekeep);
    disp(['=================================================================']);
    disp(['   ================================Solution of Dual
Problem============================']);
    disp('     T     P1      P2       P3      P4     P5      P6       P7       P8      P9      P10');
    disp(qtablekeep);
    disp(['
================================================================================
==========']);
    disp(['   ================================The best Economic
Dispatch==========================']);
   disp('      T     P1      P2       P3      P4     P5      P6       P7       P8      P9      P10');
   disp(edtable);
```

```
   disp(['
==========================================================================
==========']);
   disp(['==========The best Cost Comparation==============']);
   disp(' T        FuelCost        StartUpCost        TotalCost');
   disp(totaledtable);
   disp(['Total  ',ccc,'            ',ccc1,'     ',ccc2]);
   disp(['==========The best Cost Comparation of Multi Fuel==============']);
   disp('T         FuelCost        StartUpCost        TotalCost');
   for i = 1:24
       fprintf('%d        %f      %d
%f\n',MultiFuelData(i,1),MultiFuelData(i,22),MultiFuelData(i,23),MultiFuelData(i,24));
   end
   fprintf('Total      %s      %s       %s\n',num2str(TotalMulCost),num2str(TotalStart),num2str(Totalall));
   disp(['================================================']);
   toc;
   set(handles.text53,'string',toc);
   chi=chikeep;
   set(handles.uitable12,'Data',chi)

function edit25_Callback(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit25 as text
%        str2double(get(hObject,'String')) returns contents of edit25 as a double
% --- Executes during object creation, after setting all properties.
function edit25_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end
function edit0026_Callback(hObject, eventdata, handles)
% hObject    handle to edit0026 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit0026 as text
%        str2double(get(hObject,'String')) returns contents of edit0026 as a double
% --- Executes during object creation, after setting all properties.
function edit0026_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit0026 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit27_Callback(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit27 as text
%        str2double(get(hObject,'String')) returns contents of edit27 as a double
% --- Executes during object creation, after setting all properties.
function edit27_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
function edit28_Callback(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit28 as text
%        str2double(get(hObject,'String')) returns contents of edit28 as a double
% --- Executes during object creation, after setting all properties.
function edit28_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit29_Callback(hObject, eventdata, handles)
% hObject    handle to edit29 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit29 as text
%        str2double(get(hObject,'String')) returns contents of edit29 as a double
% --- Executes during object creation, after setting all properties.
function edit29_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit29 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit30_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit30 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit30 as text
%        str2double(get(hObject,'String')) returns contents of edit30 as a double
% --- Executes during object creation, after setting all properties.
function edit30_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit30 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit31_Callback(hObject, eventdata, handles)
% hObject    handle to edit31 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit31 as text
%        str2double(get(hObject,'String')) returns contents of edit31 as a double
% --- Executes during object creation, after setting all properties.
function edit31_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit31 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit32_Callback(hObject, eventdata, handles)
% hObject    handle to edit32 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit32 as text
%        str2double(get(hObject,'String')) returns contents of edit32 as a double
% --- Executes during object creation, after setting all properties.
function edit32_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit32 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit33_Callback(hObject, eventdata, handles)
% hObject    handle to edit33 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit33 as text
%        str2double(get(hObject,'String')) returns contents of edit33 as a double
% --- Executes during object creation, after setting all properties.
function edit33_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit33 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes during object creation, after setting all properties.
function text46_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text46 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

ประวัติย่อผู้ทำโครงงาน

ประวัติย่อผู้ทำโครงงาน

ชื่อ ชื่อสกุล	นางสาววนัชพร ชมปรารภ
วันเดือนปีเกิด	28 สิงหาคม 2538
สถานที่เกิด	เขตหนองแขม กรุงเทพฯ
ที่อยู่ปัจจุบัน	65 ซอยเพชรเกษม 81 ถนนมาเจริญ แขวงหนองแขม เขตหนองแขม กรุงเทพฯ 10160
หมายเลขโทรศัพท์ติดต่อ	099-217-5626

ประวัติการศึกษา

พ.ศ. 2555	มัธยมศึกษาปีที่ 6 จากโรงเรียนมัธยมวัดหนองแขม กรุงเทพมหานคร
พ.ศ.2560	กำลังศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ

ประวัติย่อผู้ทำโครงงาน

ชื่อ ชื่อสกุล	นางสาวสุทัตตา สันตยากร
วันเดือนปีเกิด	16 กันยายน 2537
สถานที่เกิด	อำเภอดำเนินสะดวก จังหวัดราชบุรี
ที่อยู่ปัจจุบัน	52 หมู่ 4 ต.ศรีสุราษฎร์ อ.ดำเนินสะดวก จ.ราชบุรี 70130
หมายเลขโทรศัพท์ติดต่อ	094-957-7540

ประวัติการศึกษา

พ.ศ. 2555	มัธยมศึกษาปีที่ 6 จากโรงเรียน เบญจมราชูทิศ ราชบุรี
พ.ศ.2560	กำลังศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ