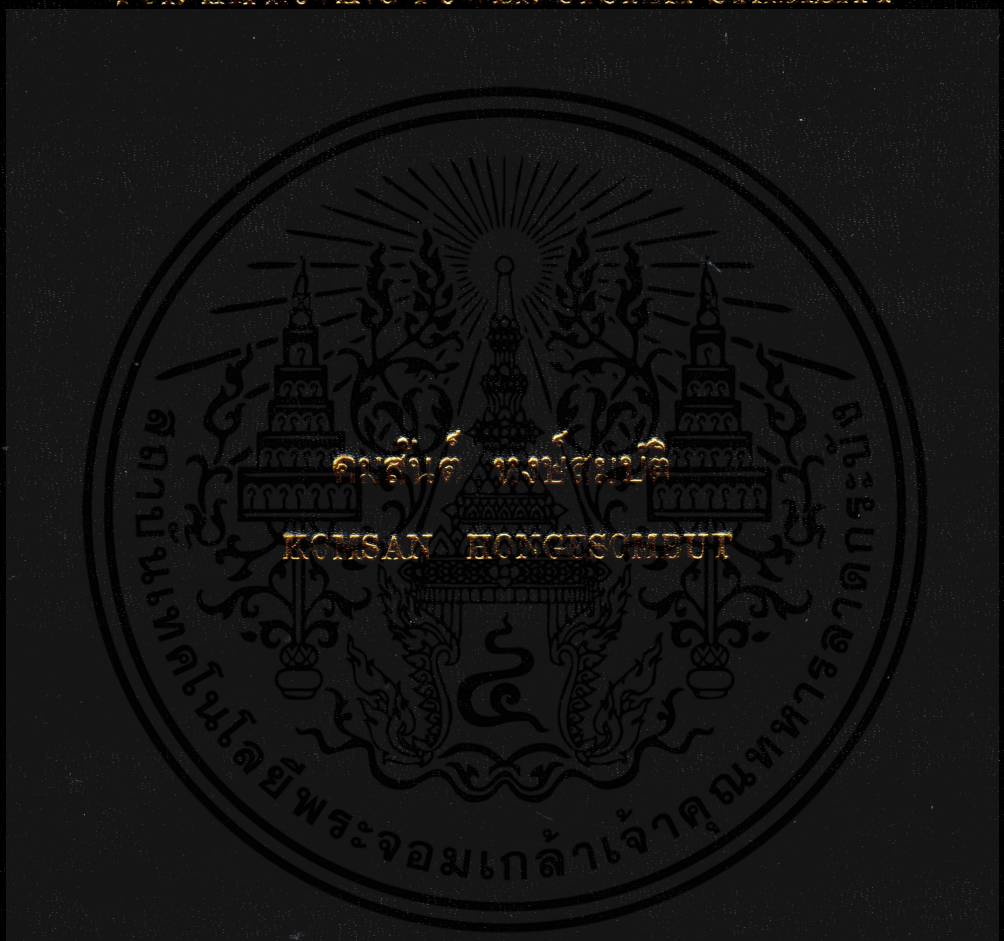
เจเนติกอัลกอริทึมสำหรับช่วยออกแบบตัวควบคุมฟัซซีของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

GENETIC ALGORITHM AIDED FUZZY CONTROLLER DESIGN OF SVC FOR IMPROVING POWER SYSTEM STABILITY



คมสัน หงษ์สมบัติ

KOMSAN HONGSOMBUT

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชา วิศวกรรมไฟฟ้า

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2541

ISBN 974-622-522-7

เจเนติกอัลกอริทึ่มสำหรับช่วยออกแบบตัวควบคุมฟัซซี่ของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

GENETIC ALGORITHM AIDED FUZZY CONTROLLER DESIGN OF SVC FOR IMPROVING POWER SYSTEM STABILITY

T033365

คมสันต์ หงษ์สมบัติ

KOMSAN HONGESOMBUT

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2542

ISBN 974-622-522-7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

GENETIC ALGORITHM AIDED FUZZY CONTROLLER DESIGN OF SVC FOR IMPROVING POWER SYSTEM STABILITY

KOMSAN HONGESOMBUT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIRMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

1999

ISBN 974-622-522-7

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านก
กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปล ถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COPYRIGHT 1999
SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

หัวข้อวิทยานิพนธ์	เจเนติกอัลกอริทึ่มสำหรับช่วยออกแบบตัวควบคุมฟัซซี่ของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง
นักศึกษา	นายคมสันต์ หงษ์สมบัติ
รหัสประจำตัว	40061034
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2542
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ.มณฑล ลีลาจินดาไกรฤกษ์

บทคัดย่อ

วิทยานิพนธ์นี้เสนอวิธีการใหม่ในการออกแบบตัวควบคุมฟัซซี่ของ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังโดยใช้เจเนติกอัลกอริทึ่ม(Genetic Algorithm) แบบจำลองที่ใช้ในวิทยานิพนธ์นี้ประกอบด้วยเครื่องจักรกลไฟฟ้าชิงโครนัส 1 ตัวต่อกับบัสอนันต์(Infinite Bus) ผ่านสายส่งวงจรคู่และมี SVC ติดตั้งอยู่จุดกึ่งกลางของสายส่ง ตัวควบคุม SVC ออกแบบโดยใช้หลักการการควบคุมแบบฟัซซี่ซึ่งอาศัยข้อมูลที่ได้รับจากเฟสเพลน สัญญาณการเปลี่ยนแปลงและอัตราการเปลี่ยนแปลงกำลังไฟฟ้า ณ จุดที่ติดตั้ง SVC ถูกเลือกเป็นอินพุทสำหรับตัวควบคุมฟัซซี่ โดยที่ตัวแปรควบคุมฟัซซี่จะใช้เจเนติกอัลกอริทึ่มในการค้นหาค่าพารามิเตอร์ที่ให้ผลตอบสนองไดนามิกของระบบที่ดีที่สุดหรือใกล้เคียง ผลการจำลองแสดงให้เห็นถึงประสิทธิภาพของวิธีการใหม่ที่นำเสนอ โดยทำการเปรียบเทียบกับผลที่ได้จากการออกแบบโดยใช้วิธีไอเทอร์เรชั่น รวมทั้งได้แสดงผลการจำลองโดยใช้วิธีการใหม่ที่นำเสนอทำงานร่วมกับตัวควบคุมเสถียรภาพระบบไฟฟ้ากำลัง (PSS) ชนิดต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title	Genetic Algorithm Aided Fuzzy Controller Design of SVC for Improving Power Sytem Stability
Student	Mr.Komsan Hongesombut
Student ID.	40061034
Degree	Master of Engineering
Programme	Electrical Engineering
Year	1999
Thesis Advisor	Assoc.Prof.Monthon Leelajindakrirerk

ABSTRACT

This thesis presents a novel approach to design SVC fuzzy controller for improving power system stability using Genetic Algorithm(GA). System modeling considered in this thesis comprises a synchronous machine connected to an infinite bus through double circuit transmission line and SVC is installed at the mid-point. SVC controller is designed by fuzzy control method and utilizing information from phase plane. Electric power deviation and its change at the location of SVC are chosen as input signals to SVC fuzzy controller and search the optimal fuzzy control parameters to obtain the optimal or near optimal dynamic system performance by using GA. Simulation results demonstrate the effectiveness of this novel proposed control both in case SVC only and in case SVC operated together with serveral types of power system stabilizers(PSS) as well as compare the results of designing by conventional approach , iteration method , with GA approach.

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกา
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ สำเร็จลุล่วงได้อย่างดี ด้วยคำแนะนำและคำปรึกษา จาก รศ.มณฑล ลีลาจินดาไกรฤกษ์ ซึ่งเป็นอาจารย์ผู้ควบคุมวิทยานิพนธ์ ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบคุณ Prof.Dr.Yoshibumi Mizutani อาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยโตไก เอกสารข้อมูลและข้อคิดของท่าน มีส่วนช่วยในการแก้ไขปัญหาต่างๆที่เกิดขึ้นระหว่างทำวิทยานิพนธ์เป็นอย่างมาก

ขอขอบพระคุณ บิดา มารดา ที่ให้การสนับสนุนด้านการศึกษา และค่าใช้จ่ายต่างๆ เสมอมา รวมทั้งเป็นกำลังใจให้ผู้วิจัยทำงานวิจัยชิ้นนี้จนประสบความสำเร็จ

ขอขอบคุณ คุณสุมิตรา ศักดิ์ศิรินุกูล บุคคลที่เป็นกำลังของผู้วิจัยตลอดระยะเวลา 10 ปีที่ผ่านมา ความสำเร็จต่างๆในอดีตจนถึงปัจจุบัน ได้รับแรงผลักดันจากบุคคลผู้นี้เป็นอย่างมาก ไม่มีสิ่งใดที่ผู้วิจัยจะตอบแทนให้กับบุคคลผู้นี้ได้ นอกจากความตั้งใจที่จะต้องประสบความสำเร็จในทุกด้าน โดยมีบุคคลผู้นี้อยู่ในใจตลอดไป

ขอขอบคุณ คุณอินทิยา เลิศสุทธิรัศมีวงศ์ เพื่อนที่รู้ใจที่ให้คำปรึกษาและช่วยคลายความกังวลต่างๆของผู้วิจัย ทำให้ปัญหาต่างๆได้ถูกใช้ปัญญาในการแก้ไขลุล่วงด้วยดี

ขอขอบคุณ คุณเขมธัช สุวรรณภักดี วิศวกรส่วนระบบและบริการสื่อสาร ศูนย์ระบบสารสนเทศ สำนักงานใหญ่ การปิโตรเลียมแห่งประเทศไทย ที่อำนวยความสะดวกด้านอุปกรณ์ประกอบการทำวิทยานิพนธ์

สุดท้ายขอขอบคุณบัณฑิตวิทยาลัย ที่ได้ให้ทุนสนับสนุนการทำวิทยานิพนธ์ครั้งนี้

คุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

คมสันต์ หงษ์สมบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

หน้า

บทคัดย่อภาษาไทย.....I

บทคัดย่อภาษาอังกฤษ.....II

กิตติกรรมประกาศ.....III

สารบัญ.....IV

สารบัญตาราง.....X

สารบัญรูป.....XI

คำจำกัดความ.....XIX

บทที่ 1 บทนำ.....1

1.1 ความเป็นมาและความสำคัญของปัญหา.....1

1.2 วัตถุประสงค์ของการทำโครงงาน.....2

1.3 ขอบเขตของโครงงาน.....3

1.4 ขั้นตอนการศึกษา.....3

1.5 ประโยชน์ที่คาดว่าจะได้รับ.....4

บทที่ 2 กระบวนการทางพันธุกรรมและทฤษฎีที่เกี่ยวข้อง.....5

2.1 กระบวนการทางพันธุกรรมเบื้องต้น.....5

2.1.1 การเข้ารหัส DNA.....5

2.1.2 การส่งผ่านข้อมูลทางพันธุกรรม.....7

2.1.3 การรีคอมบิเนชั่น.....8

2.1.4 การมิวเทชั่น.....9

2.2 เจเนติกอัลกอริทึมแบบพื้นฐาน.....11

2.3 ทฤษฎีพื้นฐานและข้อสมมุติฐานที่เกี่ยวข้อง.....14

2.3.1 ทฤษฎีสคีมา.....14

2.3.1.1 ผลกระทบจากการคัดเลือก.....15

2.3.1.2 ผลกระทบจากการครอสโอเวอร์.....16

2.3.1.3 ผลกระทบจากการมิวเทชั่น.....16

2.3.1.4 สมการการเติบโตของสคีมา.....17

2.3.2 ข้อสมมุติฐานบิวล์ดิ้งบล๊อก.....17

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

บทที่ 3 การปรับปรุงเจเนติกอัลกอริทึม.....18

3.1 การแทนโครโมโซม.....18

3.2 ฟังก์ชั่นวัตถุประสงค์และฟังก์ชั่นฟิทเนส.....19

3.2.1 สเกลลิ่งแบบเชิงเส้น.....20

3.2.2 การตัดเศษซิกม่า.....20

3.2.3 สเกลลิ่งด้วยกฎยกกำลัง.....20

3.2.4 แรงค์กิ่ง.....21

3.3 วิธีการคัดเลือก.....21

3.4 ตัวดำเนินการทางเจเนติก.....22

3.4.1 การครอสโอเวอร์.....22

3.4.2 การมิวเทชั่น.....24

3.4.3 การตั้งค่าอัตราการทำงาน.....24

3.4.4 การจัดลำดับใหม่.....24

3.5 การแทนที่.....25

บทที่ 4 การใช้ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง.....26

4.1 การชดเชยกำลังไฟฟ้ารีแอกทีฟด้วย SVC.....26

4.1.1 ผลกระทบของ SVC ต่อระบบไฟฟ้ากำลัง.....26

4.1.2 วิธีการควบคุม SVC.....27

4.1.2.1 การควบคุมด้วย $\Delta\delta$.....29

4.1.2.2 การควบคุมด้วย $\Delta\omega$.....30

4.2 ประโยชน์และชนิดของการชดเชยกำลังไฟฟ้ารีแอกทีฟ.....31

บทที่ 5 วิเคราะห์ระบบจำลอง.....35

5.1 แบบจำลองระบบ.....36

5.2 สมการเครื่องกำเนิดไฟฟ้าซิงโครนัส.....36

5.2.1 สมการกระแสและแรงดันในแกน d-q.....37

5.2.2 สมการฟลักซ์ขดลวดฟิลด์เครื่องกำเนิดไฟฟ้าซิงโครนัส.....41

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

5.2.3 สมการสวิง.....42

5.2.4 สมการกำลังไฟฟ้าเอาท์พุทเครื่องกำเนิดไฟฟ้าซิงโครนัส.....45

5.2.5 สมการแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัส.....46

5.3 แบบจำลอง AVR.....47

5.4 แบบจำลอง GOV.....49

5.5 แบบจำลอง PSS.....50

5.5.1แบบจำลอง ΔP-PSS.....50

5.5.2แบบจำลอง $\Delta\omega$-PSS.....54

5.5.3 แบบจำลอง$\Delta\omega$-PSS ชนิดฟัซซี่.....58

5.6 แบบจำลองตัวควบคุมฟัซซี่ของ SVC.....60

5.7 แบบจำลอง SVC.....61

5.8 สมการการไหลกำลังไฟฟ้าในสภาวะเริ่มต้นปกติ.....62

5.9 นิยามสภาวะการรบกวนและการวิเคราะห์โครงข่ายอิมพีแดนซ์.....68

5.9.1 นิยามสภาวะการรบกวน.....68

5.9.2 การวิเคราะห์โครงข่ายอิมพีแดนซ์.....68

บทที่ 6 การออกแบบตัวควบคุมด้วยวิธีฟัซซี่ซึ่งอาศัยข้อมูลที่ได้รับจากเฟสเพลน.....70

6.1 การแปลงความหมายข้อมูลจากเฟสเพลน.....70

6.2 การคำนวณสัญญาณควบคุม.....72

6.3 การปรับค่าพารามิเตอร์ควบคุมฟัซซี่.....72

6.4 การเชื่อมต่อ SVC กับตัวควบคุมฟัซซี่.....74

6.5 ความสัมพันธ์ของ P,Vt,Vc,ω เมื่อควบคุมด้วย SVC.....75

6.6 การปรับการควบคุม SVC โดยใช้ข้อมูลจากเฟสเพลน.....80

6.7 ผลกระทบต่อปริมาณอื่นๆจากการปรับการควบคุมของ SVC.....81

6.8 การประยุกต์ใช้งาน.....89

บทที่ 7 การออกแบบโปรแกรมและการใช้โปรแกรม.....90

7.1 การออกแบบโปรแกรมสำหรับจำลองระบบ.....90

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

7.1.1 โปรแกรมจำลองระบบกรณีควบคุมด้วย PSS.....90

7.1.2 โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่.....93

7.1.2.1 การหามุม θ.....94

7.1.2.2 การหา μ_N.....96

7.1.2.3 การหา μ_G.....98

7.1.3 โปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน.....101

7.2 การออกแบบโปรแกรมเจเนติกอัลกอริทึ่ม.....101

7.2.1 การจัดการหน่วยความจำ.....102

7.2.2 การครอสโอเวอร์.....102

7.2.3 การมิวเทชั่น.....107

7.2.4 การทำสเกลลิ่ง.....108

7.2.5 การคัดเลือก.....113

7.2.6 การเข้ารหัสแบบเกรย์.....118

7.2.7 อิลิทิสซึม.....120

7.2.8 การปรับค่าความน่าจะเป็นในการครอสโอเวอร์และมิวเทชั่น.....120

7.2.9 โปรแกรมหลัก.....121

7.3 การทดสอบโปรแกรมเจเนติกอัลกอริทึ่ม.....122

7.3.1 สรุป.....128

7.4 การออกแบบโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่.....137

7.5 การใช้งานโปรแกรม.....141

7.5.1 โปรแกรมจำลองระบบกรณีควบคุมด้วย PSS.....141

7.5.2 โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่.....143

7.5.3 โปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน.....146

7.5.4 โปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีไอเทอร์เรชั่น.....148

7.5.5 โปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีเจเนติกอัลกอริทึ่ม.....148

บทที่ 8 ผลการจำลองด้วยคอมพิวเตอร์.....151

8.1 ค่าคงที่ระบบจำลอง.....151

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

8.2 ผลตอบสนองกรณีควบคุมด้วย PSS.....152

8.3 ผลตอบสนองกรณีติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{SVC}.....152

8.4 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่.....169

8.4.1 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่น.....169

8.4.1.1 กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว.....169

8.4.1.2 กรณีควบคุมแบบผสมผสาน.....169

8.4.2 ผลการออกแบบด้วยวิธีเจเนติกอัลกอริทึ่ม.....186

8.4.2.1 กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว.....186

8.4.2.2 กรณีควบคุมแบบผสมผสาน.....187

8.4.3 ผลเปรียบเทียบการออกแบบโดยวิธีไอเทอร์เรชั่นกับการออกแบบโดยใช้โปรแกรม DAFCD.....188

บทที่ 9 บทสรุปและข้อเสนอแนะ.....209

9.1 บทสรุป.....209

9.2 ข้อเสนอแนะ.....215

เอกสารอ้างอิง.....217

ภาคผนวก.....221

ภาคผนวก ก.....222

ภาคผนวก ข.....224

ภาคผนวก ค.....227

ภาคผนวก ง.....280

ภาคผนวก จ.....305

ภาคผนวก ฉ.....341

ภาคผนวก ช.....342

ภาคผนวก ซ.....358

ภาคผนวก ฌ. ผลงานวิชาการที่ได้รับการตีพิมพ์.....361

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

ประวัติผู้เขียน.....378

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกา
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่ หน้า

2.1 การแปลงรหัสเจเนติกจากโคดอนไปเป็นกรดอะมิโน.....6

2.2 ขั้นตอนการคัดเลือกแบบวงล้อรูเลท.....11

4.1เปรียบเทียบคุณลักษณะต่างๆของการชดเชยกำลังไฟฟ้ารีแอกทีฟชนิดควบคุมได้.....33

7.1รายการใน tourneylist ของการคัดเลือกแบบทัวร์นาเมนท์.....117

7.2ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA1 เข้ารหัสแบบไบนารี.....125

7.3ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA2 เข้ารหัสแบบไบนารี.....125

7.4ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA3 เข้ารหัสแบบไบนารี.....125

7.5 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA4 เข้ารหัสแบบไบนารี.....126

7.6 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA1 เข้ารหัสแบบเกรย์.....126

7.7 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA2 เข้ารหัสแบบเกรย์.....126

7.8 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA3 เข้ารหัสแบบเกรย์.....127

7.9 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA4 เข้ารหัสแบบเกรย์.....127

8.1 ค่าคงที่ระบบจำลอง.....151

8.2 ผลสรุปจากการควบคุมด้วย PSS.....152

8.3 ผลสรุปกรณีติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{SVC}.....152

8.4 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่นกรณีควบคุมด้วย SVC ชนิดฟัซซื่อย่างเดียว.....169

8.5 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่นกรณีควบคุมแบบผสมผสาน.....169

8.6 ค่าอินพุทเริ่มต้นโปรแกรม GAFCD.....186

8.7 ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC ชนิดฟัซซื่อย่างเดียว.....186

8.8 ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS1.....187

8.9 ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS2.....187

8.10 ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS3.....188

8.11 ผลเปรียบเทียบการออกแบบโดยวิธีไอเทอร์เรชั่นกับการออกแบบโดยใช้โปรแกรม GAFCD.....188

9.1 ค่ารีแอกแตนซ์ขนานตามชนิดของฟอล์ท.....216

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกา
่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

รูปที่ หน้า

2.1 โครงสร้างคอมพลีเมนทารีของ DNA สายคู่.....6

2.2 การจัดลำดับชั้นของ DNA.....7

2.3 การแปลง DNA ไปเป็นโปรตีน.....7

2.4 กระบวนการแปลงความหมายตามทฤษฎีของ Crick.....8

2.5 แบบจำลอง Holiday สำหรับกระบวนการรีคอมบิเนชั่น.....9

2.6 ชนิดของการมิวเทชั่น.....10

2.7 การคัดเลือกแบบวงล้อรูเลท.....12

2.8 การครอสโอเวอร์แบบ 1 จุด.....12

2.9 การมิวเทชั่นในตำแหน่งบิตที่ 4.....12

2.10 วัฏจักร GA.....13

2.11 อัลกอริทึ่มของ GA แบบพื้นฐาน.....14

2.12 ลูกบาศ์ก 3 มิติ.....15

3.1 ลิงค์สิสแบบเชิงเส้น.....18

3.2 การแทนโครโมโซมแบบกราฟ.....19

3.3 ตัวอย่างการครอสโอเวอร์แบบหลายจุด.....23

3.4 ตัวอย่างการครอสโอเวอร์แบบยูนิฟอร์ม.....23

3.5 ตัวอย่างการจัดลำดับใหม่.....24

4.1การแปลง Y เป็น Δ วงจรสมมูลย์ระบบจำลอง.....26

4.2 กราฟ P-δ.....27

4.3 การแกว่งไกวเมื่อไม่มีแดมปิ้ง.....28

4.4 การควบคุมอย่างง่ายเพื่อเพิ่มแดมปิ้งในระบบ.....28

4.5 บล๊อกไดอะแกรมแบบจำลองของ Phillips-Heffron.....29

4.6 การแกว่งไกวเมื่อควบคุม SVC ด้วย $\Delta\delta$.....30

4.7 การแกว่งไกวเมื่อควบคุม SVC ด้วย $\Delta\omega$.....30

4.8 TCR/FC.....31

4.9 TSC/TCR.....31

5.1 ระบบจำลอง.....36

5.2 ระบบบัสอนันต์ที่ลดเหลือรูปอย่างง่าย.....38

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป(ต่อ)

รูปที่ หน้า

5.3 เครื่องกำเนิดไฟฟ้าซิงโครนัสจ่ายกำลังไฟฟ้าสู่โครงข่ายไฟฟ้ากำลัง.....43

5.4 แบบจำลอง AVR.....47

5.5 แบบจำลอง AVR ส่วนบล็อกป้อนกลับที่มีการปรับปรุง.....48

5.6 แบบจำลอง GOV.....49

5.7 แบบจำลอง ΔP-PSS.....51

5.8 แบบจำลอง ΔP-PSS ที่มีการปรับปรุงใหม่.....51

5.9 แบบจำลอง $\Delta\omega$-PSS.....54

5.10 แบบจำลอง $\Delta\omega$-PSS ที่มีการปรับปรุงใหม่.....55

5.11 แบบจำลอง $\Delta\omega$-PSS ชนิดฟัซซี่.....58

5.12 แบบจำลองตัวควบคุมฟัซซี่ของ SVC.....60

5.13 แบบจำลอง SVC.....61

5.14 วงจรสมมูลย์อย่างง่ายและเฟสเซอร์ไดอะแกรมสภาวะเริ่มต้นปกติ.....62

5.15 การแปลงวงจรสมมูลย์อย่างง่ายจาก Y เป็น Δ.....65

5.16 การไหลของกำลังไฟฟ้ารีแอกทีฟของวงจรสมมูลย์อย่างง่าย.....66

5.17 การรบกวนแบบเล็กน้อย.....68

5.18 ขั้นตอนการลดรูปวงจรสมมูลย์ของแบบจำลองเมื่อเกิดการลัดวงจร.....69

6.1 ตัวควบคุมฟัซซี่ของ SVC.....70

6.2 เฟสเพลน.....71

6.3 ค่าฟังก์ชั่นการเป็นสมาชิกของ θ.....71

6.4 ค่าฟังก์ชั่นการเป็นสมาชิกของ D_r.....72

6.5 เปรียบเทียบผลตอบสนอง 2 กรณีที่มีพื้นที่ใต้กราฟเท่ากัน.....73

6.6 บล็อกไดอะแกรมการเชื่อมต่อ SVC เข้ากับตัวควบคุมฟัซซี่.....74

6.7 วงจรสมมูลย์อย่างง่ายสำหรับการคำนวณ ΔV_d, ΔV_q, ΔI_d, Δi_q.....76

6.8 ความสัมพันธ์ P, ω, Q, Vt และ Vc จากการปรับตัวควบคุม.....79

6.9 ตัวอย่างการปรับการควบคุมของ SVC.....80

6.10 สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC.....83

6.11 เส้นทางการควบคุมบนเฟสเพลน.....89

7.1 โฟลว์ชาร์ตขั้นตอนหลักของโปรแกรมจำลองระบบ.....91

สารบัญรูป(ต่อ)

รูปที่ หน้า

7.2 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมด้วย PSS.....93

7.3 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่.....94

7.4 โฟลว์ชาร์ตขั้นตอนการหามุม θ.....95

7.5 การคำนวณหาค่าฟังก์ชั่นการเป็นสมาชิกของ θ.....96

7.6 การคำนวณหาค่าฟังก์ชั่นการเป็นสมาชิกของ D_r.....98

7.7 โฟลว์ชาร์ตโปรแกรมย่อย svcfuz.....99

7.8 โฟลว์ชาร์ตโปรแกรมย่อย svcsimem.....100

7.9 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน.....101

7.10 การเก็บข้อมูลของโครโมโซม.....102

7.11 การครอสโอเวอร์แบบ 1 จุด.....103

7.12 การครอสโอเวอร์แบบ 2 จุด.....103

7.13 การครอสโอเวอร์แบบยูนิฟอร์ม.....104

7.14 โฟลว์ชาร์ตการครอสโอเวอร์แบบ 1 จุด.....105

7.15 โฟลว์ชาร์ตการครอสโอเวอร์แบบ 2 จุด.....106

7.16 โฟลว์ชาร์ตการครอสโอเวอร์แบบยูนิฟอร์ม.....107

7.17 การมิวเทชั่น.....107

7.18 โฟลว์ชาร์ตการมิวเทชั่น.....108

7.19 สมการเส้นตรงที่เป็นไปได้สำหรับการทำสเกลลิ่งแบบเชิงเส้น.....109

7.20 โฟลว์ชาร์ตการทำสเกลลิ่งแบบเชิงเส้น.....112

7.21 โฟลว์ชาร์ตการทำสเกลลิ่งแบบเอกโปเนนเทียล.....113

7.22 พื้นที่ในวงล้อรูเลทที่แบ่งตามสัดส่วนของค่าฟิทเนส.....114

7.23 โฟลว์ชาร์ตการคัดเลือกแบบวงล้อรูเลท.....115

7.24 พื้นที่ในวงล้อรูเลทเมื่อเกิดการลู่เข้าก่อนกำหนด.....116

7.25 การคัดเลือกแบบทัวร์นาเมนท์เมื่อ q = 3.....116

7.26 การคัดเลือกแบบทัวร์นาเมนท์ในแต่ละรอบของการสลับที่.....117

7.27 โฟลว์ชาร์ตการคัดเลือกแบบทัวร์นาเมนท์.....118

7.28 โฟลว์ชาร์ตการแปลงรหัสแบบแกรย์ไปเป็นรหัสแบบไบนารี.....119

7.29 โฟลว์ชาร์ตการทำอิลิทิสซึม.....120

สารบัญรูป(ต่อ)

รูปที่ หน้า

7.30 โฟลว์ชาร์ตโปรแกรมหลักโปรแกรมเจเนติกอัลกอริทึม.....122

7.31 กราฟฟังก์ชั่นทดสอบ.....123

7.32 กราฟคอนทัวร์ฟังก์ชั่นทดสอบ.....124

7.33 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA1 เข้ารหัสแบบไบนารี.....129

7.34 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA2 เข้ารหัสแบบไบนารี.....130

7.35 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA3 เข้ารหัสแบบไบนารี.....131

7.36 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA4 เข้ารหัสแบบไบนารี.....132

7.37 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA1 เข้ารหัสแบบเกรย์.....133

7.38 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA2 เข้ารหัสแบบเกรย์.....134

7.39 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA3 เข้ารหัสแบบเกรย์.....135

7.40 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA4 เข้ารหัสแบบเกรย์.....136

7.41 โฟลว์ชาร์ตโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่ ด้วยวิธีไอเทอร์เรชั่น.....139

7.42 โฟลว์ชาร์ตโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่ ด้วยวิธีเจเนติกอัลกอริทึม(GAFCD)...140

7.43 ตัวอย่างหน้าจออินพุทสำหรับเลือกชนิดของการรบกวน.....141

7.44 ตัวอย่างหน้าจอเอาท์พุทสำหรับโปรแกรมจำลองระบบกรณีควบคุมด้วย PSS.....142

7.45 หน้าจออินพุทสำหรับเลือกประเภทของการควบคุมสำหรับโปรแกรมจำลองระบบ
กรณีควบคุมด้วย SVC ชนิดฟัซซี่.....144

7.46 หน้าจอการกำหนดค่าพารามิเตอร์ควบคุมฟัซซี่.....145

7.47 หน้าจอเอาท์พุทกรณีควบคุมด้วย SVC ชนิดฟัซซี่.....145

7.48 หน้าจอเอาท์พุทสำหรับโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน.....147

7.49 หน้าจอเอาท์พุทโปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีไอเทอร์เรชั่น.....148

7.50 หน้าจออินพุทสำหรับโปรแกรม GAFCD.....149

7.51 หน้าจอรายงานผลการใส่ค่าพารามิเตอร์ของโปรแกรม GAFCD.....149

7.52 หน้าจอรายงานผลในแต่ละเจนเนอเรชั่นของโปรแกรม GAFCD.....150

8.1 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า.....153

8.2 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า.....154

8.3 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า.....155

8.4 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า.....156

สารบัญรูป(ต่อ)

รูปที่ หน้า

8.5 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า.....157
8.6 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า.....158
8.7 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า.....159
8.8 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า.....160
8.9 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า.....161
8.10 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า.....162
8.11 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า.....163
8.12 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า.....164
8.13 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบรุนแรง เมื่อ Xj = 1.0 เท่า.....165
8.14 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบเล็กน้อยเมื่อ Xj = 1.0 เท่า.....166
8.15 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบรุนแรง เมื่อ Xj = 1.3 เท่า.....167
8.16 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบเล็กน้อย เมื่อ Xj = 1.3 เท่า.....168
8.17 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....170
8.18 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....171
8.19 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....172
8.20 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....173
8.21 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....174
8.22 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....175

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป(ต่อ)

รูปที่ หน้า

8.23 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....176

8.24 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....177

8.25 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....178

8.26 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....179

8.27 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....180

8.28 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....181

8.29 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....182

8.30 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....183

8.31 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....184

8.32 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น.....185

8.33 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว.....189

8.34 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS1.....190

8.35 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS2.....191

8.36 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS3.....192

8.37 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....193

8.38 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....194

สารนิพนธ์นี้เป็นลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย ไม่อนุญาตให้นำไปใช้ประโยชน์ทางการค้า ก่อนได้รับอนุญาตเป็นลายลักษณ์อักษรจากเจ้าของลิขสิทธิ์ทุกครั้งที่มีการนำไปใช้

สารบัญรูป(ต่อ)

รูปที่ หน้า

8.39 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....195

8.40 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....196

8.41 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....197

8.42 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....198

8.43 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....199

8.44 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD.....200

8.45 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....201

8.46 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....202

8.47 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....203

8.48 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....204

8.49 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....205

8.50 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....206

8.51 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....207

8.52 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD.....208

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป(ต่อ)

รูปที่ หน้า

9.1 เปรียบเทียบผลตอบสนองเมื่อควบคุมด้วย PSS แบบต่างๆภายใต้การรบกวนแบบรุนแรง....209

9.2 ผลตอบสนองเมื่อควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม.....210

9.3 ผลตอบสนองเมื่อควบคุมด้วย SVC ชนิดฟัซซี่.....210

9.4 ผลตอบสนองเปรียบเทียบการควบคุม SVC ร่วมกับ PSS แบบต่างๆ.....211

9.5 เส้นทางการเข้าหาคำตอบของวิธีเจเนติกอัลกอริทึ่ม.....213

9.6 เส้นทางการเข้าหาคำตอบของวิธีไอเทอร์เรชั่น.....214

9.7 การแปลงเป็นวงจรสมมูลย์อย่างง่ายระหว่างเกิดฟอล์ทสำหรับฟอล์ทชนิดต่างๆ.....215

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำจำกัดความที่ใช้ในการศึกษา

1. สัญลักษณ์

δ = มุมถ่ายโอนกำลัง (Power angle : rad)

θ_m = มุมทางกลของโรเตอร์ (Mechanical angle of Rotor : rad)

ω = ความเร็ว (speed : rad/s)

ω_0 = ความเร็วซิงโครนัส (Synchronous speed : rad/s)

ω_m = ความเร็วเพลาโรเตอร์ (Rotor shaft velocity : rad/s)

σ = เพอร์มาเนนท์ดรูป (permanent droop)

ϕ_d = ฟลักซ์แม่เหล็กในแกน d (d-axis flux)

ϕ_q = ฟลักซ์แม่เหล็กในแกน q (q-axis flux)

ϕ_f = ฟลักซ์แม่เหล็กในขดลวดฟิลด์ (field flux)

B = ซัสเซบแตนซ์ (susceptance : pu^{-1})

D = สัมประสิทธิ์แดมปิ้ง (Damping coefficient: pu-s/rad)

i_d = กระแสในแกน d (d-axis current : pu)

i_q = กระแสในแกน q (q-axis current : pu)

I = โมเมนต์เฉื่อย (Moment of inertia)

J = ดัชนีการทำงาน (performance index)

K_A = เกนของเอกไซเตอร์ (Exciter gain)

K_f = ลูปเกนป้อนกลับสำหรับควบคุมแรงดัน (voltage control feedback loop gain)

K_S = เกนของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (SVC gain)

M = ค่าคงที่ความเฉื่อย (inertia constant : $pu\text{-}s^2$ / rad)

P_e = กำลังไฟฟ้าเอาท์พุท (electrical power output : pu)

P_i = กำลังไฟฟ้าอินพุท (electrical power input :pu)

P_m = กำลังกลเอาท์พุท (mechanical power output : pu)

P_o = กำลังไฟฟ้าเอาท์พุทเริ่มต้น (initial electrical power output : pu)

T_a = ทอร์กเร่ง (accelerating torque : pu)

T_A = ค่าคงตัวเวลาเอกไซเตอร์ (exciter time constant : s)

T_c = ค่าคงตัวเวลาตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (SVC time constant : s)

T_{do}' = ค่าคงตัวเวลาเปิดวงจรขดลวดฟิลด์ (open circuit time constant of the field : s)

T_e = ทอร์กไฟฟ้าเอาท์พุท (electrical torque output : pu)

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

T_i = ทอร์กไฟฟ้าอินพุท (electrical torque input : pu)

T_f = ค่าคงตัวเวลาลูปป้อนกลับควบคุมแรงดัน (voltage control feedback loop time constant : s)

T_g = ค่าคงตัวเวลา GOV (governor time constant : s)

T_m = ค่าคงตัวเวลาวงจรการวัด (measuring circuit time constant : s)

T_r = ค่าคงตัวเวลาวงจรรีเซต (reset circuit time constant : s)

T_t = ค่าคงตัวเวลาเทอร์ไบน์ (turbine time constant : s)

T_v = ค่าคงตัวเวลาวงจรวัดของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (measuring circuit time constant of SVC : s)

V_c = แรงดันที่ตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (SVC voltage : pu)

V_d = แรงดันในแกน d (d-axix voltage : pu)

V_q = แรงดันในแกน q (q-axis voltage : pu)

V_f = แรงดันขดลวดฟิลด์ (field voltage : pu)

V_o = แรงดันที่บัสอนันต์ (Infinite bus voltage : pu)

V_t = แรงดันที่ขั้วเครื่องกำเนิดไฟฟ้า (Terminal voltage : pu)

x_d = ซิงโครนัสรีแอกแตนซ์ในแกน d (Synchronous reactance d-axis : pu)

x_q = ซิงโครนัสรีแอกแตนซ์ในแกน q (Synchronous reactance q-axis : pu)

x_d' = ทรานเซียนท์รีแอกแตนซ์ในแกน d (Transient reactance d-axis : pu)

x_q' = ทรานเซียนท์รีแอกแตนซ์ในแกน q (Transient reactance q-axis : pu)

2. คำย่อ

AVR = ตัวควบคุมแรงดันไฟฟ้าอัตโนมัติ (Automatic Voltage Regulator)

GA = เจเนติกอัลกอริทึ่ม (Genetic Algorithm)

GOV = ตัวควบคุมความเร็วเครื่องกำเนิดไฟฟ้า (Gorvernor)

PSS = ตัวควบคุมเสถียรภาพระบบไฟฟ้ากำลัง (Power System Stabilizer)

SVC = ตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (Static Var Compensator)

SSR = การเลือกแบบสุ่มโดยมีการแทนที่ (Stochastic Sampling with Replacement)

SSPR = การเลือกแบบสุ่มโดยมีการแทนที่บางส่วน (Stochastic Sampling with Partial Replacement)

SUS = การเลือกแบบยูนิเวอร์ซอลโดยสุ่ม (Stochastic Universal Sampling)

TCR = รีแอกเตอร์ควบคุมแบบไทริสเตอร์ (Thyristor Controll Reactor)

TSC = ตัวเก็บประจุสวิทช์แบบไทริสเตอร์ (Thyristor Switching Capacitor)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ในกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากความต้องการพลังงานไฟฟ้าในปัจจุบันและอนาคตมีแนวโน้มสูงขึ้นมากกว่าในอดีตหลายเท่าตัวการเชื่อมต่อระบบไฟฟ้าต่างๆเข้าไว้ด้วยกันจึงเป็นสิ่งจำเป็นตามหลักเศรษฐศาสตร์ของระบบไฟฟ้ากำลัง [1] ส่งผลให้ระบบไฟฟ้ากำลังมีความซับซ้อนมากขึ้น ปัญหาที่ต้องให้ความสำคัญเป็นอย่างมากก็คือ การศึกษาเรื่องเสถียรภาพของระบบ เพราะหากระบบไฟฟ้ากำลังไม่มีเสถียรภาพแล้ว จะส่งผลกระทบต่อผู้ใช้ไฟ เช่น เกิดปัญหาไฟดับเป็นบริเวณกว้าง ซึ่งจะสร้างความไม่สะดวกแก่ผู้ใช้ไฟ เกิดความเสียหายแก่ผลผลิตอุตสาหกรรม และสิ้นเปลืองค่าใช้จ่ายในการทำให้ระบบกลับสู่สภาพปกติ

ปัจจุบันการศึกษาพฤติกรรมต่างๆในระบบไฟฟ้ากำลังโดยการจำลองด้วยคอมพิวเตอร์เป็นที่ยอมรับกัน [2,3] เนื่องจากมีความถูกต้องในย่านที่สามารถยอมรับได้และสามารถทราบพฤติกรรมต่างๆที่เกิดขึ้นกับระบบไฟฟ้ากำลังได้ล่วงหน้า ทำให้สามารถเตรียมการแก้ไขได้ล่วงหน้าในระหว่างการวางแผนการออกแบบ โดยมีค่าใช้จ่ายที่ต่ำมากกว่าวิธีอื่น [3] วิทยานิพนธ์ฉบับนี้จึงมุ่งศึกษาและปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังโดยการจำลองด้วยคอมพิวเตอร์

การปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังสามารถทำได้หลายวิธี วิธีการหนึ่งที่รู้จักกันเป็นอย่างดี คือ การควบคุมตัวควบคุมแรงดันไฟฟ้าอัตโนมัติ (AVR) ร่วมกับตัวควบคุมเสถียรภาพระบบไฟฟ้ากำลัง (PSS) [40] ซึ่งปกติแล้วตัวควบคุมจะเป็นแบบเกนคงที่ (Fixed gain) และมีการชดเชยแบบนำหน้าและล้าหลัง (Lead-Lag Compensate) การควบคุมวิธีนี้ไม่สามารถสร้างแดมปิ้ง (damping) ให้แก่ระบบได้มากพอ เมื่อโครงข่ายไฟฟ้าที่ควบคุมเป็นชนิดสายส่งระยะไกล [4] นอกจากนี้แล้วตัวควบคุมชนิดนี้ยังไม่สามารถทำงานได้ในย่านกว้าง เนื่องจากการออกแบบตัวควบคุมชนิดนี้ จะพิจารณาการออกแบบเฉพาะจุดทำงานใดจุดหนึ่ง ดังนั้นเมื่อจุดทำงานของระบบเปลี่ยนแปลงไป ซึ่งในระบบไฟฟ้ากำลังการเปลี่ยนแปลงจุดทำงานมักจะไม่เป็นเชิงเส้น ตัวควบคุมชนิดนี้อาจให้ผลตอบสนองระบบที่ไม่ดีพอ หรือบางครั้งไม่สามารถควบคุมเสถียรภาพได้

ปัจจุบันอิเลกทรอนิกส์กำลังมีความรุดหน้าเป็นอย่างมาก ทำให้ตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (SVC) ชนิดไทริสเตอร์ซึ่งตอบสนองการควบคุมได้เร็ว มีการใช้กันอย่างแพร่หลายในอุตสาหกรรม โดยมีวัตถุประสงค์เพื่อปรับปรุงเพาเวอร์แฟกเตอร์ การทำสมดุลโหลด การควบคุมแรงดัน [5,6] การเพิ่มความสามารถในการส่งถ่ายกำลังไฟฟ้าในสายส่ง [7] นอกจากนี้ SVC ยังสามารถนำมาใช้ในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง [30-33] แต่ก็มีบทความวิจัยกล่าวถึงไม่มาก

นัก วิทยานิพนธ์ฉบับนี้จึงสนใจการใช้ SVC เข้าไปแก้ไขโครงข่ายไฟฟ้าโดยตรงเพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

การควบคุม SVC วิธีเดิม จะมีเฉพาะลูปควบคุมแรงดัน ซึ่งมีข้อเสีย คือ ไม่สามารถสร้างแดมปิ้งในระบบได้มากพอ [8,9] ด้วยเหตุนี้จึงมีบทความวิจัยมากมายกล่าวถึงการสร้างสัญญาณควบคุมเสริมเพิ่มเข้าไปในลูปควบคุมแรงดันของ SVC เพื่อเพิ่มแดมปิ้งของระบบ เช่น การควบคุม SVC แบบ PD , PID [10] ซึ่งการควบคุมด้วยวิธีดังกล่าวไม่สามารถทำงานได้ในย่านกว้าง เนื่องจากตัวควบคุมเป็นแบบเกนคงที่ จึงมีการพัฒนาตัวควบคุมที่สามารถปรับค่าเกนได้ด้วยตัวเอง (Self-Tuning) ซึ่งให้ผลตอบสนองที่ดีและสามารถทำงานได้ในย่านที่กว้างขึ้น แต่มีข้อเสีย คือ ต้องอาศัยการคำนวณสัญญาณควบคุมที่ซับซ้อน ดังนั้นหากระบบไฟฟ้ากำลังมีความซับซ้อนมาก เวลาที่ใช้ในการคำนวณสัญญาณควบคุมก็เพิ่มมากขึ้น วิธีการควบคุมวิธีนี้จึงอาจไม่สามารถนำมาใช้ได้ [11,12]

วิทยานิพนธ์ฉบับนี้จึงได้เสนอ วิธีการสร้างสัญญาณควบคุมเสริมเพิ่มเข้าไปในลูปควบคุมแรงดันของ SVC โดยใช้ตัวควบคุมชนิดฟัซซี่ที่อาศัยข้อมูลจากเฟสเพลน [4] ซึ่งสามารถขจัดปัญหาขั้นตอนการคำนวณที่ซับซ้อน และตัวควบคุมชนิดฟัซซี่ยังสามารถตัดสินสร้างสัญญาณควบคุมเสริม ที่สามารถควบคุมได้ในย่านกว้าง นอกจากนี้ยังสามารถปรับค่าฟังก์ชั่นการเป็นสมาชิกของฟัซซี่ให้เหมาะสม เพื่อให้ได้ผลตอบสนองระบบที่ดีที่สุด โดยใช้เจเนติกอัลกอริทึ่มซึ่งสามารถลดเวลาในการออกแบบตัวควบคุมชนิดฟัซซี่ลงอย่างมากเมื่อเปรียบเทียบกับวิธีไอเทอร์เรชั่น [13] การใช้เจเนติกอัลกอริทึ่มมีข้อดีอีกประการ คือ สามารถสร้างความหลากหลายในการควบคุม เช่น สามารถนำไปใช้ในการปรับค่าพารามิเตอร์ของตัวควบคุมที่เหมาะสมของตัวควบคุมแบบเก่ากับตัวควบคุมชนิดฟัซซี่ ซึ่งการคำนวณด้วยวิธีปกติไม่สามารถทำได้ เนื่องจากเจเนติกอัลกอริทึ่มสนใจเฉพาะคำตอบที่ให้ค่าเหมาะสมสูงสุดโดยไม่สนใจขั้นตอนการคำนวณสมการ [13-15] การใช้เจเนติกอัลกอริทึ่มจึงเป็นเครื่องมือสำคัญในการพัฒนาเทคนิคการควบคุม SVC แบบใหม่ที่ดีต่อไปในอนาคต

1.2 วัตถุประสงค์ของการทำโครงงาน

วัตถุประสงค์ของโครงงานนี้ คือ ออกแบบตัวควบคุมชนิดฟัซซี่เพื่อสร้างสัญญาณควบคุมเสริมเพิ่มเข้าลูปควบคุมแรงดันของ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง ภายใต้การรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย ซึ่งสามารถทำงานได้ในย่านกว้างและให้ผลตอบสนองที่ดี โดยใช้เจเนติกอัลกอริทึ่มเป็นเครื่องมือช่วยออกแบบตัวควบคุมชนิดฟัซซี่ เพื่อเลือกพารามิเตอร์ควบคุมของตัวควบคุมชนิดฟัซซี่ที่เหมาะสมที่ให้ผลตอบสนองระบบได้ดีที่สุด วัตถุประสงค์ของการใช้เจเนติกอัลกอริทึ่ม เพื่อลดเวลาในการออกแบบตัวควบคุมชนิดฟัซซี่ และสร้างความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลากหลายในการควบคุมให้เป็นไปอย่างมีประสิทธิภาพ ก่อให้เกิดแนวทางพัฒนาและความก้าวหน้าในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังต่อไป

1.3 ขอบเขตของโครงงาน

โครงงานนี้มีขอบเขตในการเขียนโปรแกรมคอมพิวเตอร์ เพื่อจำลองระบบไฟฟ้ากำลังชนิดเครื่องกำเนิดไฟฟ้าซิงโครนัสเดี่ยวต่อกับบัสอนันต์ และปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังโดยใช้ SVC ซึ่งมีการสร้างสัญญาณควบคุมเสริมเพิ่มเข้าไปในลูปควบคุมแรงดันจากตัวควบคุมชนิดฟัซซี่ ในส่วนของโปรแกรมช่วยออกแบบตัวควบคุมฟัซซี่ (GAFCD) จะใช้เจเนติกอัลกอริทึ่ม ซึ่งมีการปรับปรุงตัวดำเนินการเจเนติกและเทคนิคการจัดการหน่วยความจำของโปรแกรม จากนั้นจะศึกษาการทำงานร่วมกันของตัวควบคุมแบบเก่ากับตัวควบคุมชนิดฟัซซี่ที่สร้างขึ้นต่อการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง โดยใช้ภาษาซีในการเขียนโปรแกรม

1.4 ขั้นตอนการศึกษา

ขั้นตอนการศึกษาต่างๆ สามารถแบ่งเป็นหัวข้อได้ดังต่อไปนี้

1. ศึกษาแบบจำลอง ซึ่งประกอบด้วย
   - 1.1 ระบบเครื่องกำเนิดไฟฟ้าซิงโคนัสเดี่ยวต่อกับบัสอนันต์
   - 1.2 ตัวควบคุมเครื่องกำเนิดไฟฟ้า เช่น AVR , GOV และ PSS ชนิดต่างๆ
   - 1.3 ตัวควบคุม SVC ชนิดต่างๆ เช่น SVC ที่ควบคุมด้วยการควบคุม PD , PID และ Self-tuning
2. ศึกษาทฤษฎีพื้นฐานของการควบคุมแบบฟัซซี่ และการประยุกต์ใช้งานฟัซซี่ในระบบไฟฟ้ากำลัง
3. ศึกษาทฤษฎีพื้นฐานของเจเนติกอัลกอริทึ่ม และการประยุกต์ใช้งานเจเนติกอัลกอริทึ่มในระบบไฟฟ้ากำลัง
4. เขียนโปรแกรมจำลองระบบเครื่องกำเนิดไฟฟ้าซิงโครนัสเดี่ยวต่อกับบัสอนันต์ โดยใช้การควบคุมด้วย PSS ร่วมกับ AVR
5. พัฒนาโปรแกรมโดยใช้ SVC ที่ควบคุมด้วยลูปควบคุมแรงดันอย่างเดียว ควบคุมเสถียรภาพของระบบ และ SVC ที่มีการสร้างสัญญาณควบคุมเสริมจากตัวควบคุมชนิดฟัซซี่ โดยใช้วิธีไอเทอร์เรชั่นในการออกแบบ
6. เขียนโปรแกรมเจเนติกอัลกอริทึ่มและปรับปรุงตัวดำเนินการเจเนติก โดยทำการทดสอบกับฟังก์ชั่นทดสอบมาตรฐาน เพื่อหาวิธีที่เหมาะสมและทดสอบประสิทธิภาพของโปรแกรมที่เขียนขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7. แก้ไขโปรแกรมเจเนติกอัลกอริทึ่มให้สามารถทำงานร่วมกับโปรแกรมออกแบบตัวควบคุมชนิดฟัซซี่

8. บันทึกผลการทดสอบจากการควบคุมด้วยวิธีการต่างๆ และเปรียบเทียบผลที่ได้จากการออกแบบตัวควบคุมด้วยวิธีเจเนติกอัลกอริทึ่มกับวิธีไอเทอร์เรชั่น

9. สรุปผลการทดลอง

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถจำลองพฤติกรรมของเครื่องกำเนิดไฟฟ้าซิงโครนัสต่อการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย และสามารถปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วยตัวชดเชยกำลังไฟฟ้าแบบสถิตย์โดยใช้วิธีควบคุมฟัซซี่แบบอาศัยข้อมูลจากเฟสเพลน

2. วิธีควบคุมแบบฟัซซี่ที่นำเสนอเป็นวิธีที่ง่าย ไม่มีความซับซ้อน แต่ให้ผลลัพธ์ที่ดี และมีความรวดเร็วในการควบคุม จึงมีความเป็นไปได้ที่จะประยุกต์ใช้วิธีนี้กับระบบไฟฟ้ากำลังที่มีปัญหาเรื่องเสถียรภาพและประยุกต์ใช้กับการควบคุมแบบเวลาจริง

3. การจำลองบนคอมพิวเตอร์ทำให้สามารถทราบปัญหาเกี่ยวกับเสถียรภาพได้ล่วงหน้า และสามารถทำการทดสอบวิธีควบคุมที่สามารถแก้ไขปัญหาเสถียรภาพได้อย่างมีประสิทธิภาพ โดยมีค่าใช้จ่ายที่ต่ำ นอกจากนี้ยังมีประโยชน์ต่อผู้ที่สนใจและศึกษาเกี่ยวกับเสถียรภาพระบบไฟฟ้ากำลังทำให้เกิดความเข้าใจในปัญหาที่เกิดขึ้นและเป็นแนวทางสำหรับการประยุกต์ใช้การควบคุมแบบใหม่กับระบบไฟฟ้ากำลัง

4. การใช้เจเนติกอัลกอริทึ่ม ทำให้ลดเวลาในการหาค่าพารามิเตอร์ควบคุมที่เหมาะสมสูงสุด ซึ่งแต่เดิมการใช้วิธีไอเทอร์เรชั่นจำเป็นต้องใช้เครื่องคอมพิวเตอร์ที่มีประสิทธิภาพสูง เช่น เครื่องเมนเฟรม หรือ ซุปเปอร์คอมพิวเตอร์ ทำให้เกิดค่าใช้จ่ายที่สูง ดังนั้นสำหรับประเทศไทย ที่มีข้อจำกัดทางด้านงบประมาณและเครื่องมือที่ทันสมัย ทำให้การศึกษาปัญหาเสถียรภาพมีข้อจำกัดและไม่ก่อให้เกิดการพัฒนาความรู้ในสาขาวิชานี้เท่าที่ควร วิธีที่นำเสนอนี้สามารถใช้เครื่องคอมพิวเตอร์ส่วนบุคคลในการแก้ปัญหาได้โดยให้คำตอบเดียวกันหรือใกล้เคียงกับการใช้วิธีไอเทอร์เรชั่นและมีความรวดเร็วกว่า ซึ่งจะช่วยให้เกิดการพัฒนาความรู้ในสาขาวิชานี้ต่อไปในอนาคต

5. ในการออกแบบวิธีควบคุมแบบใหม่ พารามิเตอร์ควบคุมที่ให้ผลตอบสนองดีที่สุด คือสิ่งที่เราไม่ทราบคำตอบมาก่อน บางครั้งการคำนวณด้วยสมการคณิตศาสตร์แบบเดิมอาจยากหรือไม่สามารถทำได้ เจเนติกอัลกอริทึ่มจึงเป็นวิธีหนึ่งที่สามารถแก้ไขปัญหานี้ได้

6. วิธีเจเนติกอัลกอริทึ่มที่นำเสนอมีความยืดหยุ่นในการใช้งานสูง ทำให้สามารถทดลองปรับปรุงและหาวิธีควบคุมแบบใหม่ที่สามารถปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังที่ดีกว่า โดยทดสอบกับโปรแกรมที่เขียนขึ้น ซึ่งจะทำให้ทราบคำตอบที่ต้องการได้อย่างรวดเร็ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
กระบวนการทางพันธุกรรมและทฤษฎีที่เกี่ยวข้อง

2.1 กระบวนการทางพันธุกรรมเบื้องต้น

เซลของสิ่งมีชีวิตประกอบด้วยส่วนที่สำคัญ คือ นิวเคลียส ซึ่งบรรจุสารพันธุกรรมที่ทำหน้าที่ควบคุมและถ่ายทอดลักษณะของสิ่งมีชีวิต ภายในนิวเคลียส ประกอบด้วยส่วนเล็กๆที่เรียกว่า โครโมโซม โดยโครโมโซมแต่ละคู่ในนิวเคลียสเป็นโครโมโซมที่ได้รับมาจากฝ่ายพ่อและแม่ผู้ให้กำเนิดฝ่ายละแท่ง ภายในโครโมโซมแต่ละแท่งนี้ ประกอบไปด้วยสารเคมีที่เป็นสารพันธุกรรม เรียกว่า DNA (Deoxyribonucleic acid) โดย DNA ประกอบขึ้นจากหน่วยพื้นฐานที่เรียกว่า นิวคลีโอไทด์ (Nucleotide) โครงสร้างของ DNA ประกอบขึ้นจากสายของชุดนิวคลีโอไทด์ 2 สาย พันกันเป็นเกลียวคู่ลักษณะบันไดเวียน ในแต่ละส่วนของ DNA จะเป็นที่อยู่ของยีนส์ ซึ่งเป็นส่วนที่บรรจุข้อมูลสำหรับการสร้างโปรตีนแต่ละชนิด

2.1.1 การเข้ารหัส DNA

นิวคลีโอไทด์ เป็นองค์ประกอบพื้นฐานของ DNA ซึ่งสามารถแบ่งเป็นเบส 4 ชนิดตามโครงสร้างทางเคมี คือ อะดีไนน์(Adenine : A) กวาไนน์ (Guanine : G) ไซโตไซน์ (Cytosine : C) และ ไทมีนน์ (Thymine : T) โดย A และ G เป็น purine C และ G เป็น pyrimidines ตามทฤษฎีการจับคู่เบสของ Watson และ Crick G จะจับคู่กับ C และ A จะจับคู่กับ T หรือ ยูราซิล (Uracil : U*) ใน RNA ลักษณะการจับคู่แบบนี้จะทำให้พันธะไฮโดรเจนของ purine-pyrimidines ในสายคอมพลีเมนทารี DNA (Complementary DNA)เสถียร รูปที่ 2.1 แสดงสายคอมพลีเมนทารี DNA ในลักษณะของเฮลิกสายคู่ (double-strand helix)

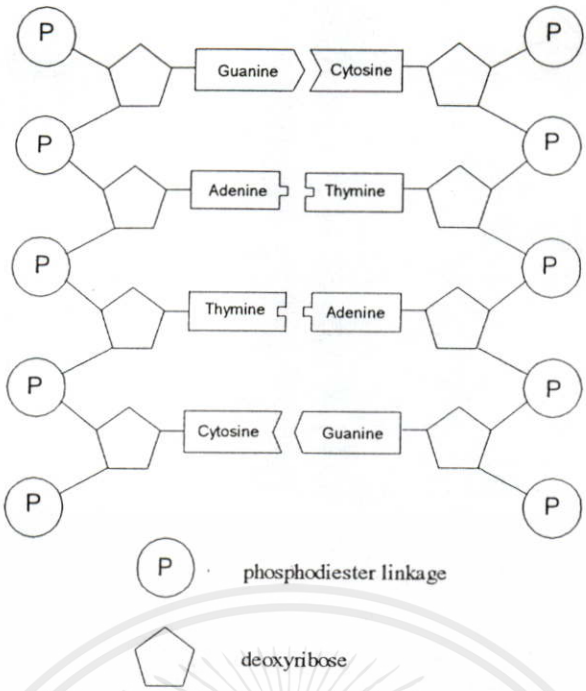
รหัส 3 ตัวของนิวคลีโอไทด์จะเป็นตัวกำหนดโคดอน (Codon) ซึ่งจะเข้าคู่กับแอนทิโคดอน (Anticodon) ใน tRNA (transfer RNA) และช่วยในการส่งข้อมูลเจเนติกในรูปแบบของกรดอะมิโน(Amino acid) ตารางที่ 2.1 แสดงการแปลความหมายจากโคดอนเป็นกรดอะมิโน โดยมีข้อสังเกต คือ กรดอะมิโนเพียง 20 ชนิดเท่านั้นจากความเป็นไปได้ 64 ชนิดที่ถูกแปลความหมาย กรดอะมิโนชนิดเดียวกัน อาจเข้ารหัสได้จากโคดอนหลายแบบ และ มีโคดอน 3 ตัว คือ UGA , UAA และ UAG ที่ไม่ตรงกับกรดอะมิโนชนิดใดเลย แต่โคดอน 3 ตัวนี้ จะทำหน้าที่เป็นสัญญาณหยุดการแปลความหมายในกระบวนการสร้างโปลีเปบไทด์ (Polypeptide) จาก RNA

การจัดลำดับชั้นของ DNA สามารถสรุปได้ในรูปที่ 2.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

---

* T จะบรรจุอยู่ใน DNA เท่านั้น โดยใน mRNA (messenger RNA) จะบรรจุนิวคลีโอไทด์ U แทน

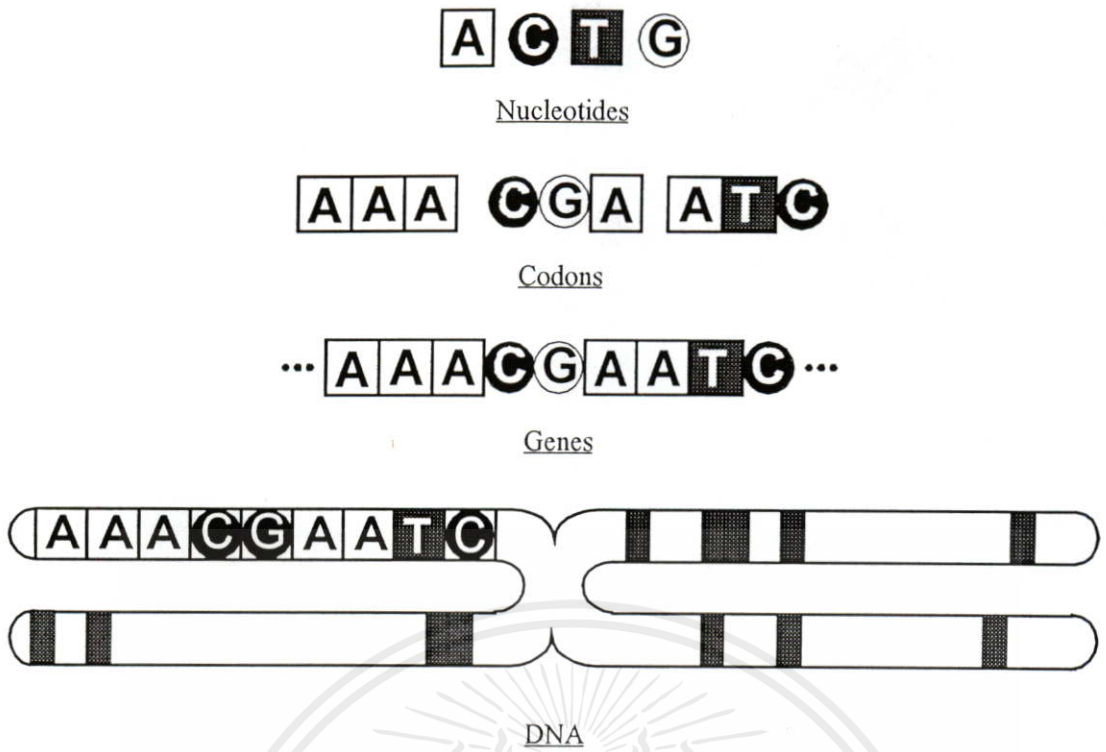


รูปที่ 2.1 โครงสร้างคอมพลีเมนทารีของ DNA สายคู่

ตารางที่ 2.1 การแปลงรหัสเจเนติกจากโคดอนไปเป็นกรดอะมิโน

	ส่วนที่ 3 ของโคดอน			
	U	C	A	G
UU	Phenylalanine	Phenylalanine	Leucine	Leucine
UC	Serine	Serine	Serine	Serine
UA	Phenylalanine	Phenylalanine	stop	stop
UG	Cysteine	Cysteine	stop	Tryptophan
CU	Leucine	Leucine	Leucine	Leucine
CC	Proline	Proline	Proline	Proline
CA	Histidine	Histidine	Glutamine	Glutamine
CG	Arginine	Arginine	Arginine	Arginine
AU	Isolecine	Isolecine	Isolecine	Methinonine
AC	Threonine	Threonine	Threonine	Threonine
AA	Asparagine	Asparagine	Lysine	Lysine
AG	Serine	Serine	Arginine	Arginine
GU	Valine	Valine	Valine	Valine
GC	Alanine	Alanine	Alanine	Alanine
GA	Aspartate	Aspartate	Glutamate	Glutamate
GG	Glycine	Glycine	Glycine	Glycine

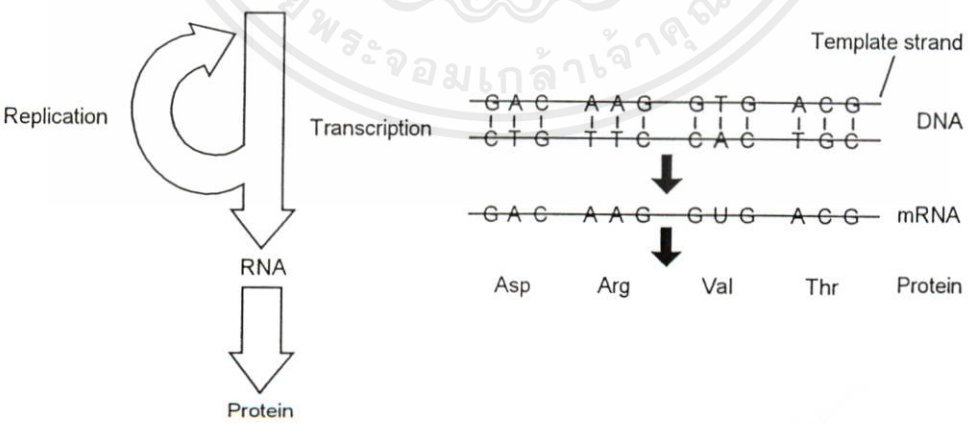
สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.2 การจัดลำดับชั้นของ DNA

2.1.2 การส่งผ่านข้อมูลทางพันธุกรรม

กระบวนการส่งผ่านข้อมูลทางพันธุกรรมหรือข้อมูลเจเนติก ประกอบด้วย 3 กระบวนการหลัก คือ การลอกแบบ (Replication) การถอดแบบ (Transcription) และการแปลความหมาย (Translation) กระบวนการทั้ง 3 นี้แสดงในรูปที่ 2.3



รูปที่ 2.3 การแปลง DNA ไปเป็นโปรตีน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การลอกแบบ

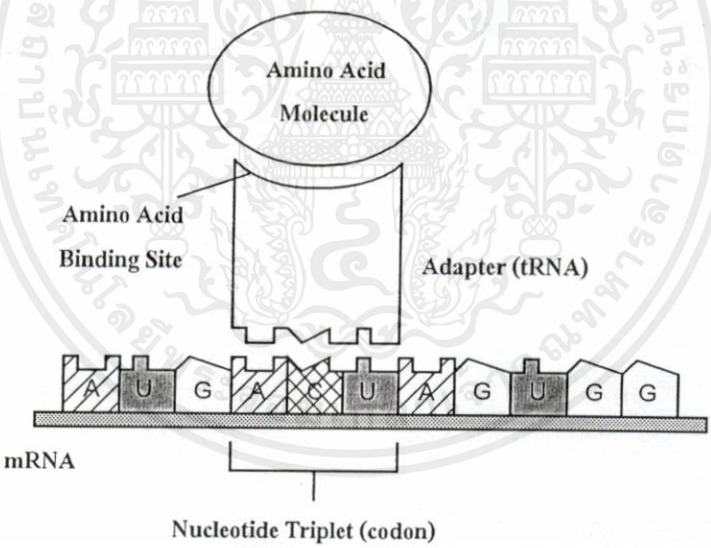
กระบวนการนี้มีหน้าที่รักษาข้อมูลเจเนติก ระหว่างกระบวนการนี้สาย DNA พ่อแม่จะแยกจากกัน แต่ละสายจะทำหน้าที่เป็นแม่พิมพ์สำหรับการสังเคราะห์สายคอมพลีเมนทารีใหม่ รูปแบบการรักษาข้อมูลเจเนติกชนิดนี้ เป็นไปแบบกึ่งอนุรักษ์(Semi-conservative)

- การถอดแบบ

การติดต่อสื่อสารของข้อมูลเจเนติกจากนิวเคลียสไปยังไซโตปลาสซึม (Cytoplasm) จะอาศัยการถอดแบบจาก DNA ไปเป็น mRNA โดยกระบวนการนี้เริ่มจาก DNA สายคู่แยกตัวจากกัน DNA สายหนึ่งจะทำหน้าที่เป็นแม่พิมพ์สังเคราะห์ mRNA ขึ้น mRNA ที่ได้จะมีลักษณะเป็นสายเดี่ยว ทำหน้าที่เป็นพาหนะสำหรับการแปลความหมายกรดนิวคลีอิกไปเป็นลำดับของโปรตีน

- การแปลความหมาย

รหัสข้อมูลเจเนติกใน mRNA จะถูกแปลความหมายบนไรโบโซมในรูปของโปรตีนที่มีลำดับของกรดอะมิโนแน่นอน ในขั้นตอนนี้ tRNA ทำหน้าที่เป็นอะแดปเตอร์ที่คอยจำโคดอนและแทรกกรดอะมิโนในตำแหน่งลำดับที่เหมาะสมในโปลีเปปไทด์ รูปที่ 2.4 แสดงกระบวนการแปลความหมายตามทฤษฎีของ Crick

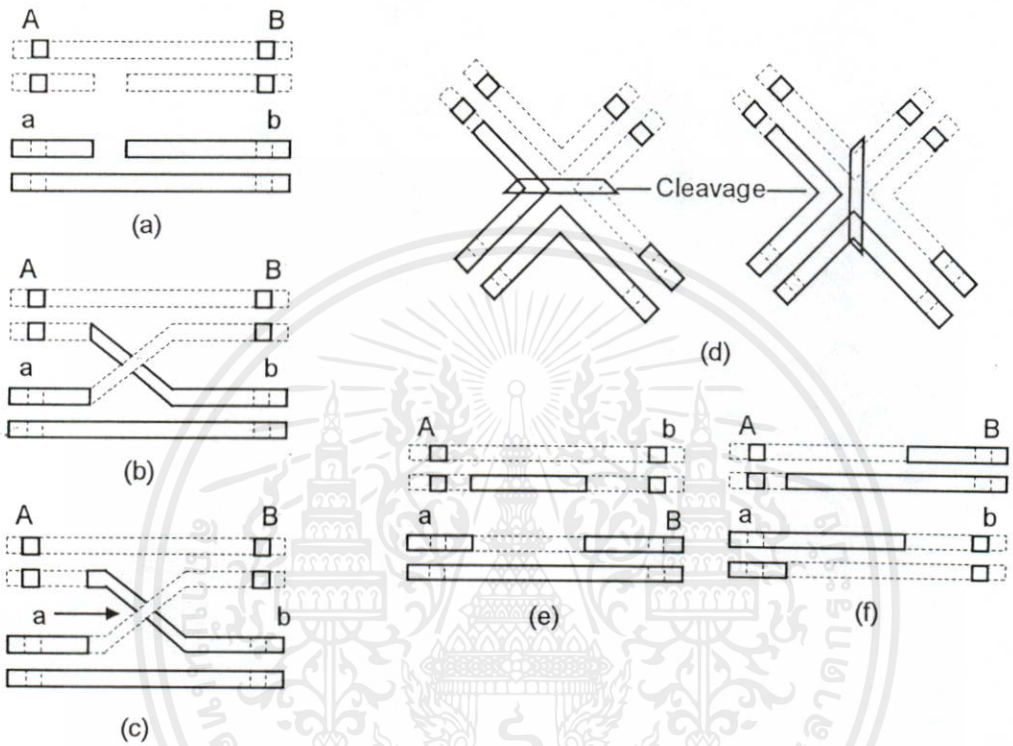


รูปที่ 2.4 กระบวนการแปลความหมายตามทฤษฎีของ Crick

2.1.3 การรีคอมบิเนชั่น (Recombination)

เป็นกระบวนการแลกเปลี่ยนข้อมูลเจเนติก ซึ่งเกิดจากการจับคู่ระหว่างสาย DNA ที่แยกออกมาจากโฮสท์ (Host) กับสาย DNA ผู้ให้ (Donor) โดยขึ้นกับตำแหน่งของการครอสโอเวอร์ การจับคู่ดังกล่าวทำให้เกิดโครงสร้างตามแบบจำลองของ Holliday ซึ่งสามารถสรุปได้ดังนี้

1. สาย DNA แยกตัวกันและ จัดเรียงตัวในแนวตรง (รูปที่ 2.5a)
2. เกิดการไขว้กันของสาย DNA เกิดเป็น Holliday intermediate (รูปที่2.5b)
3. จุดครอสโอเวอร์เกิดการเคลื่อนที่ สาย DNA ที่แยกกันถูกซ่อมแซม (รูปที่2.5c)
4. Holliday intermediate สามารถเกิดการตัด (Cleavage)ได้ 2 ทิศทาง (รูปที่ 2.5d)
5. ผลจากการตัดในแนวนอนและแนวตั้ง แสดงในรูปที่ 2.5e และ รูปที่2.5f ตามลำดับ



รูปที่ 2.5 แบบจำลอง Holliday สำหรับกระบวนการรีคอมบิเนชั่น

2.1.4 การมิวเทชั่น

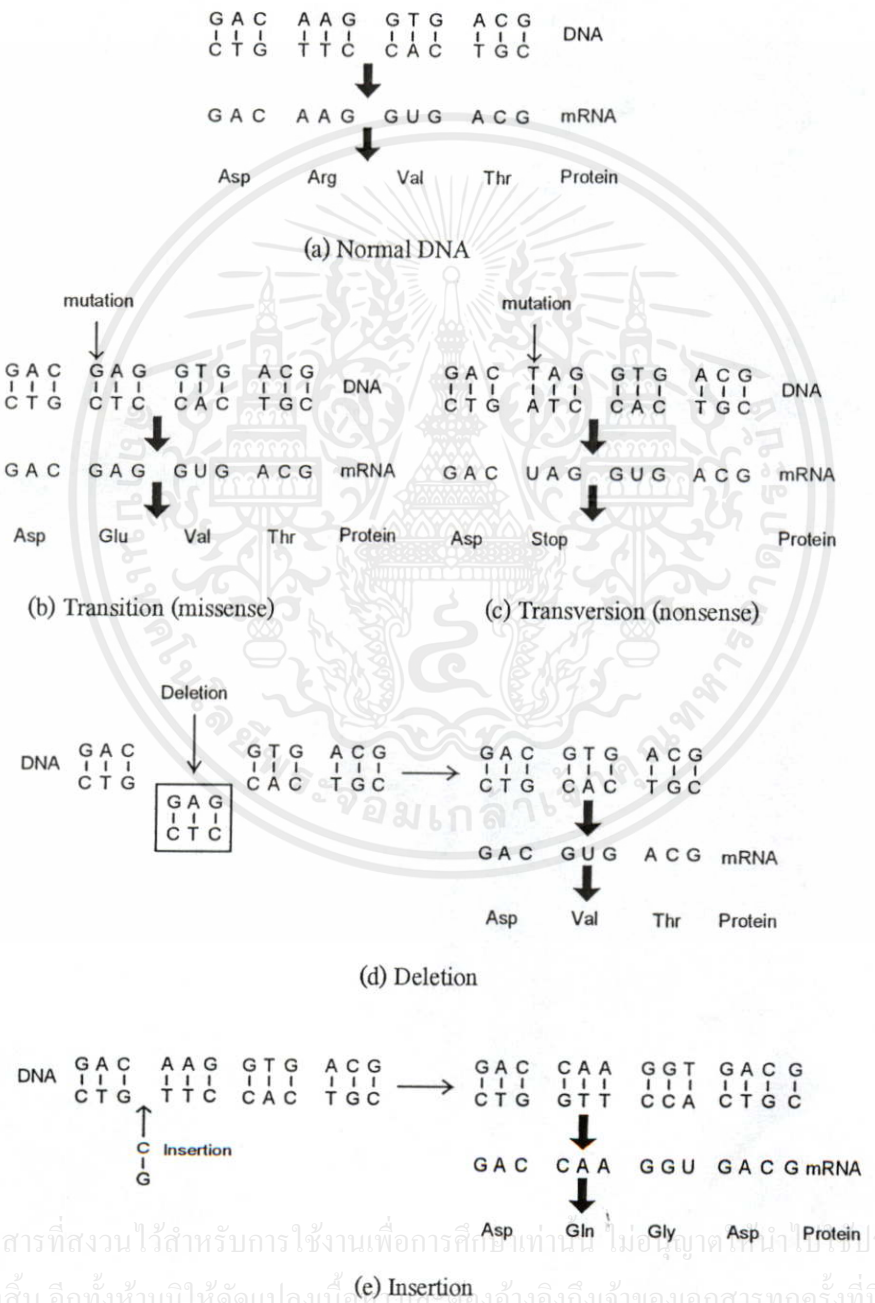
การมิวเทชั่น คือ การเปลี่ยนแปลงลำดับของ DNA ซึ่งจะมีผลต่อลักษณะภายนอกที่ปรากฏ หรือฟีโนไทป์ (Phenotype) การมิวเทชั่นนั้นปกติเกิดขึ้นได้ยาก และมีทั้งข้อดีและข้อเสีย ข้อดี คือ ทำให้เกิดการพัฒนาพันธุ์ ข้อเสียคือ ทำให้เกิดความผิดปกติของเซล ซึ่งส่วนใหญ่ในธรรมชาติจะมีข้อเสียมากกว่าข้อดี

การมิวเทชั่นแบบจุด (Point mutation) แบ่งเป็น 2 ชนิด คือ การมิวเทชั่นแบบทรานสิชั่น (Transition mutation) และการมิวเทชั่นแบบทรานเวิร์สชั่น (Transversion mutation) การมิวเทชั่นแบบทรานสิชั่น (รูปที่ 2.6b) purine จะถูกแทนที่ด้วย purine และ pyrimidines จะถูกแทนที่ด้วย pyrimidines เช่น T-A จะถูกแทนที่ด้วย C-G หรือ A-T ถูกแทนที่ด้วย G-C การมิวเทชั่นแบบทรานเวิร์สชั่น purine จะถูกแทนที่ด้วย pyrimidines หรือ pyrimidines จะถูกแทนที่ด้วย purine เช่น T-A

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะถูกแทนที่ด้วย A-T หรือ G-C , C-G จะถูกแทนที่ด้วย G-C หรือ A-T นอกจากนี้การมิวเทชั่นยังสามารถจัดเป็นกลุ่มๆตามลักษณะผลกระทบ คือ ถ้าการมิวเทชั่นไม่มีผลกระทบต่อคุณสมบัติของโคดอน จะเรียกว่า missense แต่ถ้ามีผลกระทบต่อโคดอนโดยเปลี่ยนไปเป็นโคดอนหยุด (stop codon) จะเรียกว่า nonsense

นอกจากนี้ยังมีการมิวเทชั่นแบบเลื่อนกรอบ (Frameshift mutation) ซึ่งแบ่งเป็น การมิวเทชั่นแบบลบ (deletion mutaion) (รูปที่ 2.6d) และการมิวเทชั่นแบบแทรก (insertion mutation) (รูปที่ 2.6e)



(a) Normal DNA

(b) Transition (missense)

(c) Transversion (nonsense)

(d) Deletion

(e) Insertion

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.6 ชนิดของการมิวเทชั่น

2.2 เจเนติกอัลกอริทึมแบบพื้นฐาน

หลักการพื้นฐานของ GA ถูกนำเสนอครั้งแรกโดย Holland [16] ซึ่งมีแรงดลใจจากกลไกการคัดเลือกในธรรมชาติ โดยผู้ที่แข็งแรงกว่าจะเป็นผู้ชนะในการแข่งขัน ปัญหาต่างๆที่ต้องการหาคำตอบที่ดีที่สุด สามารถแก้ไขได้ด้วยหลักการพัฒนาของ GA โดยคำตอบที่ดีที่สุดจะถูกพบจากสมาชิกสุดท้ายที่เป็นผู้ชนะ

ปัญหาโดยทั่วไปจะประกอบด้วยกลุ่มของพารามิเตอร์ ดังนั้นเมื่อใช้ GA พารามิเตอร์แต่ละตัวจะเปรียบเสมือนยีนส์ และกลุ่มของพารามิเตอร์ก็คือโครโมโซม ค่าฟิทเนสซึ่งเป็นค่าบวกจะเป็นตัววัดความดีของโครโมโซมนั้นๆ

การประยุกต์ใช้ GA ในทางปฏิบัติ จะเริ่มจากการสร้างประชากรของสมาชิกเริ่มต้นโดยสุ่ม ขนาดของประชากรนั้นไม่มีการกำหนดตายตัว ขึ้นกับลักษณะของปัญหาที่แก้ไข จากนั้นสร้างประชากรรุ่นลูกหลานซึ่งจะเป็นสมาชิกในเจนเนอเรชั่นถัดไปจากสมาชิกประชากรเจนเนอเรชั่นปัจจุบัน โดยคาดหวังว่าจะได้โครโมโซมที่ดีกว่าเดิมในเจนเนอเรชั่นใหม่นี้ และโครโมโซมที่ดีนี้จะสร้างประชากรรุ่นลูกหลานที่ดีเป็นจำนวนมากต่อไป

วิธีการคัดเลือกแบบวงล้อรูเลทเป็นวิธีการที่ใช้กันแพร่หลายมากที่สุด [17] โดยอาศัยกลไกการคัดเลือกตามสัดส่วน ขั้นตอนต่างๆสรุปไว้ในตารางที่ 2.2

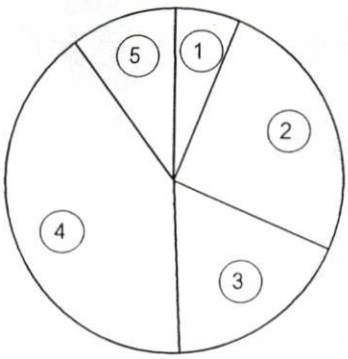
ตารางที่ 2.2 ขั้นตอนการคัดเลือกแบบวงล้อรูเลท

- บวกค่าฟิทเนสของสมาชิกทุกตัว (F_{SUM})
- สร้างตัวเลขโดยสุ่ม (n) ระหว่าง 0 และ F_{SUM}
- กลับไปเริ่มที่สมาชิกตัวแรก แล้วบวกค่าฟิทเนสของสมาชิกถัดไป จนกระทั่งผลรวมฟิทเนสที่ได้มีค่ามากกว่าหรือเท่ากับ F_{SUM}

ตัวอย่างเช่น ในรูปที่ 2.7 เส้นรอบวงของวงล้อรูเลท คือ F_{SUM} ของโครโมโซม 5 ตัว โครโมโซมที่ 4 มีค่าฟิทเนสสูงสุดจึงครอบครองเนื้อที่ในวงล้อรูเลทมากที่สุด ในขณะที่โครโมโซมที่ 1 มีค่าฟิทเนสต่ำสุด จึงครอบครองเนื้อที่ในวงล้อรูเลทต่ำสุด ในการเลือกโครโมโซม จะเริ่มจากการสร้างตัวเลขสุ่มในช่วง $[0,F_{SUM}]$ เนื้อที่ของโครโมโซมตัวใดที่ครอบคลุมค่าเลขสุ่มนี้ก็จะถูกเลือก

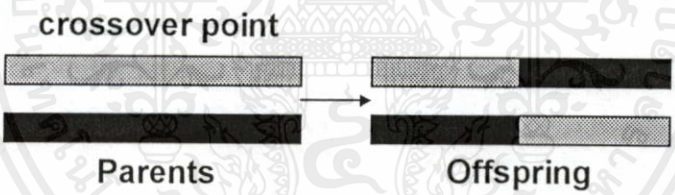
วงรอบการพัฒนาจะเกิดซ้ำไปเรื่อยๆ จนกระทั่งเงื่อนไขการหยุดที่กำหนดเป็นจริง ซึ่งเงื่อนไขการหยุดนี้อาจเป็น จำนวนเจนเนอเรชั่นสูงสุด ความเบี่ยงเบนของสมาชิกระหว่างเจนเนอเรชั่น หรือค่าฟิทเนสที่มีการกำหนดไว้ล่วงหน้า

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



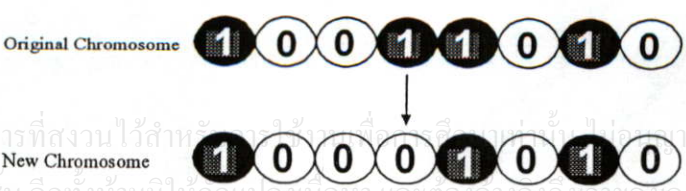
รูปที่ 2.7 การคัดเลือกแบบวงล้อรูเลท

เพื่อให้เกิดการพัฒนาในแต่ละวงรอบของ GA จะมีการใช้ตัวดำเนินการเจเนติก คือ การครอสโอเวอร์และการมิวเทชั่น ซึ่งการคัดเลือกอาจพิจารณาว่าเป็นตัวดำเนินการเจเนติกตัวหนึ่งก็ได้ รูปที่ 2.8 แสดงให้เห็นถึงกระบวนการครอสโอเวอร์แบบ 1 จุด โดยตำแหน่งครอสโอเวอร์จะถูกเลือกโดยสุ่ม ส่วนของโครโมโซมที่อยู่ถัดไปทางขวาจากจุดครอสโอเวอร์จะมีการแลกเปลี่ยนกันเพื่อสร้างโครโมโซมลูกหลาน อัตราการทำงาน (Operation rate) หรือความน่าจะเป็นในการครอสโอเวอร์ (P_c) ปกติจะมีค่าระหว่าง 0.6 ถึง 1.0 [15]



รูปที่ 2.8 การครอสโอเวอร์แบบ 1 จุด

สำหรับการมิวเทชั่น จะถูกนำมาใช้หลังจากการทำครอสโอเวอร์ โดยจะทำการเปลี่ยนแปลงค่าบางบิตโดยสุ่ม ด้วยความน่าจะเป็นในการมิวเทชั่น (P_m) ที่มีค่าน้อยๆ ซึ่งปกติมีค่าไม่เกิน 0.1 [15] รูปที่ 2.9 แสดงการมิวเทชั่นของโครโมโซมในตำแหน่งบิตที่ 4



สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

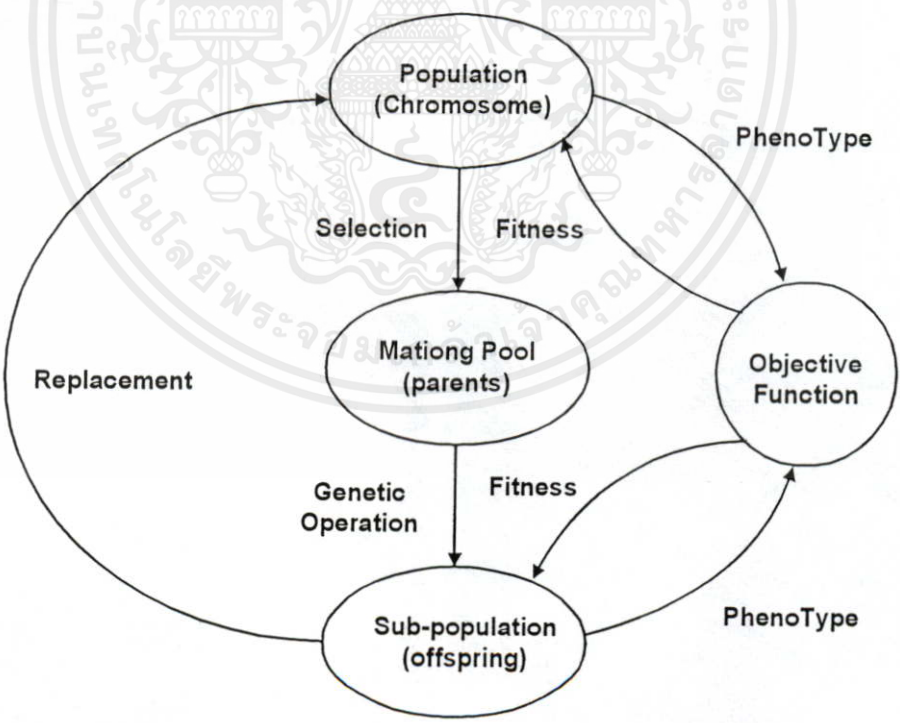
รูปที่ 2.9 การมิวเทชั่นในตำแหน่งบิตที่ 4

การเลือกค่า P_c และ P_m สำหรับควบคุมกระบวนการ GA ไม่มีการกำหนดไว้แน่นอน ส่วนใหญ่แล้วจะขึ้นอยู่กับธรรมชาติของฟังก์ชั่นวัตถุประสงค์ แต่ก็มีการแนะนำไว้เป็นแนวทางดังนี้ [18]

1. สำหรับประชากรขนาดใหญ่ (100)
   - อัตราการครอสโอเวอร์หรือความน่าจะเป็นในการครอสโอเวอร์ (P_c) ควรเป็น 0.6
   - อัตราการมิวเทชั่นหรือความน่าจะเป็นในการมิวเทชั่น (P_m) ควรเป็น 0.001
2. สำหรับประชากรขนาดเล็ก (30)
   - อัตราการครอสโอเวอร์หรือความน่าจะเป็นในการครอสโอเวอร์ (P_c) ควรเป็น 0.9
   - อัตราการมิวเทชั่นหรือความน่าจะเป็นในการมิวเทชั่น (P_m) ควรเป็น 0.01

การตั้งค่า P_c และ P_m ตามข้อแนะนำข้างบนนี้ มีข้อสังเกต คือ ความหลากหลายของประชากรเกิดขึ้นได้จาก จำนวนประชากรมากๆ หรือ การใช้อัตราการครอสโอเวอร์และมิวเทชั่นสูงๆ แต่ในทางปฏิบัติจะต้องคำนึงถึงเวลาในการประมวลผลด้วย ดังนั้นเมื่อขนาดประชากรมาก อัตราการครอสโอเวอร์และมิวเทชั่น ก็ควรมีค่าต่ำกว่าอัตราการครอสโอเวอร์และมิวเทชั่นกรณีขนาดประชากรไม่มากนัก

รูปที่ 2.10 และ 2.11 เป็นการสรุปขั้นตอนของ GA แบบพื้นฐาน



รูปที่ 2.10 วัฏจักร GA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Genetic Algorithm()
{
// start with an initial time
t:=0;
// initialize a usually random population of individuals
init_population P(t);
// evaluate fitness of all initial individuals of population
evaluate P(t);
// evolution cycle
while not terminated do
      // increase the time counter
      t := t + 1;
     // select a sub-population for offspring production
      P' := select_parents P(t);
     // perturb the mated population stochastically
     mutate P'(t);
     // evaluate its new fitness
     evaluate P'(t);
     // select the survivors from actual fitness
     P := survive P , P'(t);
}
```

รูปที่ 2.11 อัลกอริทึ่มของ GA แบบพื้นฐาน

2.3 ทฤษฎีพื้นฐานและข้อสมมุติฐานที่เกี่ยวข้อง

2.3.1 ทฤษฎีสคีมา

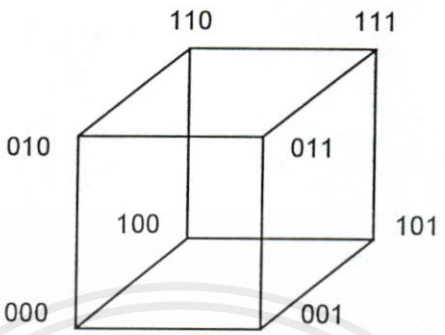
พิจารณารูปที่ 2.12 ซึ่งสมมุติรูปลูกบาศก์แทนอวกาศการค้นหาของปัญหาที่เข้ารหัสด้วยตัวเลขขนาด 3 บิต สตริง 000 แทนจุดกำเนิด สัญลักษณ์ " * " แทน " don't care " ซึ่งอาจมีค่าเป็น 1 หรือ 0 ก็ได้ สคีมาหนึ่งตัวจะแทนด้วยสตริงทุกตัวที่มีตัวเลขที่ทราบค่าแน่นอนในตำแหน่งเดียวกัน ยกเว้น * ซึ่งจะพบว่า จะมี $2^{\tau} - 1$ สตริง เมื่อ τ คือจำนวน don't care หรือ * ในสคีมา รหัสไบนารีทุกตัว คือ โครโมโซมที่สอดคล้องกับมุมของรูปลูกบาศก์และมีสมาชิก $2^{L} - 1$ ไฮเปอร์เพลนที่แตกต่างกัน เมื่อ L คือความยาวของรหัสไบนารี โดยมีอวกาศการค้นหาทั้งหมดเท่ากับ 3^{L}

GA เป็นการค้นหาจากประชากร จุดตัวอย่างของสมาชิกในประชากร จะให้ข้อมูลเกี่ยวกับไฮเปอร์เพลนมากมาย นอกจากนี้ ไฮเปอร์เพลนที่มีลำดับ* ต่ำ จะสามารถสุ่มจากประชากรได้หลาย

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

* ลำดับของไฮเปอร์เพลน คือ จำนวนของค่าบิตจริงที่ปรากฏในสคีมา

จุดไฮเปอร์เพลนต่างๆที่แตกต่างกันจะถูกคำนวณในลักษณะขนานไปกับการคำนวณสตริงแต่ละตัว ซึ่งจะทำให้ได้ข้อมูลสถิติเกี่ยวกับซับเซตของไฮเปอร์เพลนบางตัว ตามทฤษฎีแนะนำไว้ว่า ตลอดกระบวนการรีโพรดักชั่นและรีคอมบิเนชั่น สคีมาของไฮเปอร์เพลนที่แข่งขันกัน จะเพิ่มขึ้นหรือลดลงตามค่าฟิทเนสของสตริงที่บรรจุอยู่ในไฮเปอร์เพลน



รูปที่ 2.12 ลูกบาศก์ 3 มิติ

2.3.1.1 ผลกระทบจากการคัดเลือก

ค่าฟิทเนสของสคีมา S แทนด้วย $f(S,t)$ ซึ่งถูกกำหนดโดยสตริงทุกตัวในประชากรที่ตรงกับสคีมา S ζ (S,t) คือ จำนวนสตริงที่ตรงกับสคีมา S ในเจนเนอเรชั่นปัจจุบัน F(t) คือค่าฟิทเนสเฉลี่ย ดังนั้นจำนวนของสคีมา S ในเจนเนอเรชั่นถัดไปจะได้

$$\zeta(S,t+1) = \zeta(S,t)\frac{f(S,t)}{F(t)} \tag{2.1}$$

เมื่อ

$$\varepsilon = \frac{f(S,t) - F(t)}{F(t)} \tag{2.2}$$

ถ้า $\varepsilon > 0$ หมายความว่า ค่าฟิทเนสของสคีมา S มีค่าสูงกว่าค่าฟิทเนสเฉลี่ย เมื่อแทนสมการที่(2.2) ลงในสมการที่ (2.1) จะได้

$$\zeta(S,t) = \zeta(S,0)(1+\varepsilon)^t \tag{2.3}$$

จะพบว่าสคีมาที่มีค่าฟิทเนสสูงกว่าค่าฟิทเนสเฉลี่ยมีการเพิ่มขึ้นในลักษณะเป็นเอกโปเนนเทียลในเจนเนอเรชั่นถัดไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1.2 ผลกระทบจากการครอสโอเวอร์

ความน่าจะเป็นในการทำลายสคีมา S คือ

$$p_d(S) = \frac{\delta(S)}{L-1} \tag{2.4}$$

เมื่อ L คือ ความยาวโครโมโซม δ คือ ระยะห่างระหว่างตำแหน่งตัวเลขคงที่ที่อยู่นอกสุด เช่น *000* จะมี δ = 4-2 = 2 หรือ 1*00* จะมี δ = 4-1 = 3 ดังนั้นความน่าจะเป็นในการอยู่รอดของสคีมา S คือ

$$p_s(S) = 1 - \frac{\delta(S)}{L-1} \tag{2.5}$$

สมมุติให้อัตราการครอสโอเวอร์หรือความน่าจะเป็นในการครอสโอเวอร์มีค่าเป็น P_c ดังนั้นความน่าจะเป็นในการอยู่รอดของสคีมา S จะเป็น

$$p_s(S) = 1 - p_c\frac{\delta(S)}{L-1} \tag{2.6}$$

สมการที่ (2.6) สามารถเขียนได้ใหม่ ถ้า สคีมา S ยังคงอยู่รอด แม้ว่าตำแหน่งการครอสโอเวอร์ถูกเลือกระหว่างตำแหน่งค่าคงที่

$$p_s(S) \geq 1 - p_c\frac{\delta(S)}{L-1} \tag{2.7}$$

2.3.1.3 ผลกระทบจากการมิวเทชั่น

ให้ความน่าจะเป็นในการครอสโอเวอร์ คือ P_m ดังนั้นความน่าจะเป็นในการอยู่รอดของบิต 1 บิต คือ 1-P_m ถ้า O(S) คือ ลำดับของสคีมา ความน่าจะเป็นในการอยู่รอดของสคีมา S สำหรับการมิวเทชั่น คือ

$$p_s(S) = (1 - p_m)^{o(S)} \tag{2.8}$$

เนื่องจากว่า $P_m << 1$ ดังนั้นจึงสามารถประมาณสมการ (2.8) ได้ดังนี้

$$p_s(S) \approx 1 - o(S)p_m \tag{2.9}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

2.3.1.4 สมการการเติบโตของสคีมา

เมื่อรวมผลกระทบจากการคัดเลือก การครอสโอเวอร์ และการมิวเทชั่น สมการการเติบโตของสคีมา S สามารถเขียนได้ดังนี้

$$\zeta(S,t+1) \geq \zeta(S,t)\frac{f(S,t)}{F(t)}\left[1 - p_c\frac{\delta(S)}{L-1} - o(S)p_m\right] \qquad (2.10)$$

2.3.2 ข้อสมมุติฐานบิวล์ดิ้งบล๊อก (Building Block Hypothesis)

บิวล์ดิ้งบล๊อก คือ เจเนติกอัลกอริทึ่มที่ทำงานใกล้จุดที่ดีที่สุด โดยมีระยะระหว่างตำแหน่งข้างเคียงที่สั้น ลำดับต่ำ มีสคีมาที่มีประสิทธิภาพในการทำงานสูง [19]

ตัวดำเนินการเจเนติก เช่น ครอสโอเวอร์ และ มิวเทชั่น มีความสามารถที่จะสร้างหรือสนับสนุน ต่อสตริงที่ให้คำตอบที่ดีที่สุด การครอสโอเวอร์ มีแนวโน้มอนุรักษ์ข้อมูลเจเนติกที่มีอยู่ใน สตริงสำหรับการครอสโอเวอร์ ดังนั้นเมื่อสตริงที่ทำการครอสโอเวอร์เหมือนกัน ความสามารถในการสร้างบิวล์ดิ้งบล๊อกใหม่ๆก็ลดน้อยลง ในขณะที่การมิวเทชั่นเป็นตัวดำเนินการที่ไม่อนุรักษ์และมีความสามารถในการสร้างบิวล์ดิ้งบล๊อกได้อย่างรุนแรง

นอกเหนือจากนี้ การคัดเลือกก็เป็นกระบวนการสำคัญ เนื่องจากว่าการคัดเลือกมีแนวโน้มที่จะลำเอียงต่อบิวล์ดิ้งบล๊อกที่มีค่าฟิทเนสสูงๆ

ข้อสมมุติฐานนี้แนะนำว่า ปัญหาของการเข้ารหัสสำหรับ GA มีความสำคัญต่อความสามารถในการทำงานของ GA และแนวคิดในการเข้ารหัสควรจะทำให้บิวล์ดิ้งบล๊อกสั้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

33365

บทที่ 3
การปรับปรุงเจเนติกอัลกอริทึม

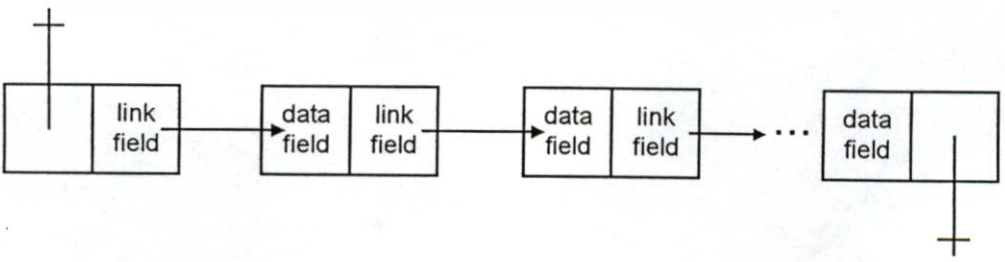
กลไกของ GA ไม่ได้ถูกควบคุมด้วยสมการดิฟเฟอเรนเทียลหรือมีพฤติกรรมเหมือนกับฟังก์ชั่นชนิดต่อเนื่องทั่วไป กลไกของ GA มีความสามารถที่เป็นเอกลักษณ์เฉพาะตัวในการออฟติไมซ์ระบบที่มีความซับซ้อน และไม่สามารถใช้วิธีการคำนวณด้วยสมการคณิตศาสตร์แบบปกติได้ การใช้ GA แบบมาตรฐานอาจไม่มีความยืดหยุ่นในการนำไปใช้งานจริง เนื่องจากในทางวิศวกรรมมักจะมีข้อกำหนดต่างๆมากมายสำหรับการออกแบบ ปัญหานี้จะเห็นเด่นชัดมากขึ้น หากปัญหาที่ทำการแก้ไขมีความซับซ้อน มีข้อจำกัดต่างๆมากมาย หรือมีลักษณะทำงานแบบมัลติทาส์กกิ้ง (multi tasking) เนื้อหาในบทนี้จึงได้แนะนำวิธีการต่างๆในการปรับปรุง GA ที่จำเป็น

3.1 การแทนโครโมโซม

การเข้ารหัสโครโมโซมอาจมีได้หลายวิธีขึ้นกับลักษณะของปัญหาที่ทำการแก้ไข วิธีการเข้ารหัสแบบไบนารีเป็นวิธีการหนึ่งที่ใช้กันอย่างกว้างขวาง เนื่องจากว่ามีความง่ายและสามารถติดตามผลได้ง่าย แต่ก็มีผลงานวิจัยที่ได้ทดลองและพบว่า การเข้ารหัสแบบเกรย์ (Gray code) ให้ผลทำงานที่ดีกว่าการเข้ารหัสแบบไบนารีอยู่เล็กน้อย

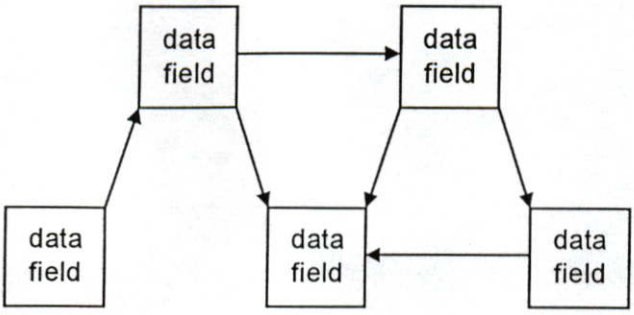
ปัจจุบัน การจัดการกับโครโมโซมโดยใช้ค่าจริง [21] ได้รับความสนใจมากขึ้น ซึ่งการแทนด้วย floating point จะให้ผลลัพธ์ที่รวดเร็วกว่าวิธีอื่น นอกจากนี้ประสิทธิภาพในการทำงานของ GA ยังขึ้นอยู่กับ ความสามารถของตัวดำเนินการเจเนติกพิเศษ แต่อย่างไรก็ตามในบางกรณีการใช้รหัสค่าจริงอาจไม่ให้ผลดีเท่าที่ควรก็ได้

สำหรับปัญหาที่มีการค้นหาในลักษณะมีลำดับ มักใช้การแทนโครโมโซมที่พิจารณาจากลำดับ การแทนโครโมโซมด้วยวิธีนี้แบบง่ายที่สุด คือ ลิงค์ลิสแบบเชิงเส้น ซึ่งแสดงในรูปที่ 3.1



รูปที่ 3.1 ลิงค์ลิสแบบเชิงเส้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.2 การแทนโครโมโซมแบบกราฟ

ในกรณีที่ความยาวโครโมโซมมีการเปลี่ยนแปลงได้ อาจใช้การแทนโครโมโซมแบบกราฟ ดังในรูปที่ 3.2 ข้อดีของการเข้ารหัสโดยพิจารณาจากลำดับเมื่อเปรียบเทียบกับการเข้ารหัสแบบตัวอักษร คือ สามารถหลีกเลี่ยงคำตอบที่ไม่ถูกต้องที่เกิดจากการทำครอสโอเวอร์ได้

ในบางกรณี อาจมีการใช้ดัชนีเป็นโครโมโซมแทนการใช้ค่าจริง ในลักษณะของ look-up table ซึ่งวิธีนี้จะมีประโยชน์สำหรับการคัดเลือกที่ไม่เป็นเชิงเส้น

จากที่กล่าวมาทั้งหมด จะพบว่าการปรับปรุงวิธีการแทนโครโมโซมมีมากมายหลายวิธี ซึ่งการจะเลือกใช้วิธีใดนั้นจะต้องเลือกให้เหมาะสมกับแต่ละปัญหา เพราะการเลือกรูปแบบที่เหมาะสม จะช่วยเพิ่มประสิทธิภาพและลดภาระในการคำนวณเมื่อมีการนำไปใช้งานจริง

3.2 ฟังก์ชั่นวัตถุประสงค์และฟังก์ชั่นฟิทเนส

ในปัญหาต่างๆทั่วไป ฟังก์ชั่นวัตถุประสงค์อาจเป็นการหาค่าสูงสุด หรือค่าต่ำสุดก็ได้ ซึ่งฟังก์ชั่นวัตถุประสงค์เหล่านี้อาจมีค่าเป็นลบในบางค่า ใน GA ฟังก์ชั่นฟิทเนส ซึ่งเป็นตัววัดสถานะของโครโมโซม จะมีค่าเป็นบวกเสมอ ดังนั้นจะต้องมีการแมป (map) ฟังก์ชั่นวัตถุประสงค์เป็นฟังก์ชั่นฟิทเนส [22]

ในทางปฏิบัติ ปัญหาที่มีการหาค่าต่ำสุดสามารถเปลี่ยนเป็นปัญหาในการหาค่าสูงสุดได้ เช่น ให้ g(x) เป็นฟังก์ชั่นวัตถุประสงค์ที่ต้องการหาค่าต่ำสุด f(x) เป็นฟังก์ชั่นฟิทเนสสำหรับหาค่าสูงสุดของ GA ความสัมพันธ์ดังกล่าว เป็นไปตามสมการ (3.1)

$$\begin{aligned} f(x) &= C_{\max} - g(x) \quad \text{เมื่อ} \quad g(x) < C_{\max} \\ f(x) &= 0 \qquad\qquad\quad \text{เมื่อ} \quad g(x) \geq C_{\max} \end{aligned} \tag{3.1}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

หรือ กรณีหาค่าต่ำสุดของฟังก์ชั่น g(x) ซึ่งมีค่าเป็นบวกเสมอ ฟังก์ชั่นฟิทเนสอาจเขียนเป็นปัญหาค่าสูงสุดได้ดังสมการที่ (3.2)

$$f(x) = A / g(x) \qquad \text{เมื่อ} \quad A \geq 0 \tag{3.2}$$

สำหรับฟังก์ชั่นวัตถุประสงค์ในการหาค่าสูงสุด อาจเขียนเป็นฟังก์ชั่นฟิทเนสได้ดังนี้

$$\begin{aligned} f(x) &= u(x) + C_{\min} \quad \text{เมื่อ} \quad u(x) + C_{\min} \rangle 0 \\ f(x) &= 0 \qquad\qquad\quad \text{เมื่อ} \quad u(x) + C_{\min} \leq 0 \end{aligned} \tag{3.3}$$

3.2.1 สเกลลิ่งแบบเชิงเส้น

ค่าฟิทเนส f_i ของแต่ละโครโมโซมที่ทำสเกลลิ่ง จะมีค่าสัมพันธ์อย่างเป็นเชิงเส้นกับค่าฟิทเนสเดิม f_i' ดังนี้

$$f_i = af_i' + b \tag{3.4}$$

โดยที่ a และ b ถูกเลือกค่าที่ทำให้ ค่าเฉลี่ยฟิทเนสเดิมเท่ากับค่าเฉลี่ยฟิทเนสใหม่หลังทำสเกลลิ่ง และค่าฟิทเนสในการสเกลสูงสุดถูกกำหนดจากค่าคงที่ที่คูณกับค่าเฉลี่ยฟิทเนส

วิธีการทำสเกลลิ่งแบบเชิงเส้นนี้จะช่วยลดผลกระทบจากการลู่เข้าก่อนกำหนด เนื่องจากบางโครโมโซมมีค่าฟิทเนสสูงกว่าปกติจากโครโมโซมอื่นๆ แต่อย่างไรก็ตามจะต้องหลีกเลี่ยงการเกิดค่าฟิทเนสที่อาจมีค่าเป็นลบจากการใช้การสเกลลิ่งวิธีนี้ ดังนั้น การเลือกค่า a และ b จึงต้องพิจารณาย่านของค่าฟิทเนสเดิมด้วย [22]

3.2.2 การตัดเศษซิกม่า (Sigma Truncation)

วิธีการนี้ใช้เพื่อหลีกเลี่ยงการเกิดฟิทเนสค่าลบ โดยค่าฟิทเนส f_i คำนวณตามสมการนี้

$$f_i = o_i - (\bar{o} - c\sigma) \tag{3.5}$$

เมื่อ c เป็นจำนวนเต็มที่มีค่าน้อยๆ $\bar{o}$ คือค่าเฉลี่ยของฟังก์ชั่นวัตถุประสงค์ σ คือส่วนเบี่ยงเบนมาตรฐานของประชากร

เพื่อป้องกันการเกิดฟิทเนสค่าลบของ f ดังนั้น $f \langle 0$ จะถูกกำหนดให้เป็นศูนย์ โครโมโซมที่ค่าฟิทเนสน้อยกว่า c (จำนวนเต็มค่าน้อยๆระหว่าง 1 ถึง 5) ส่วนเบี่ยงเบนมาตรฐานจากค่าเฉลี่ยฟิทเนสจะไม่ถูกเลือก

3.2.3 สเกลลิ่งด้วยกฎยกกำลัง

การทำสเกลลิ่งวิธีนี้ ทำได้โดยการยกกำลังค่าฟังก์ชั่นวัตถุประสงค์ o_i

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$f_i = o_i^k \tag{3.6}$$

เมื่อ k มีค่าขึ้นอยู่กับปัญหาที่ต้องการแก้ไขหรืออาจมีค่าเปลี่ยนแปลงค่าได้ขณะทำการรัน

3.2.4 แรงค์กิ้ง

วิธีนี้ค่าฟิทเนสไม่ได้มีความสัมพันธ์โดยตรงกับค่าของฟังก์ชั่นวัตถุประสงค์ แต่จะเป็นการจัดลำดับ (rank) ของค่าฟังก์ชั่นวัตถุประสงค์ [23]

การใช้วิธีการนี้จะช่วยหลีกเลี่ยงการเกิดการลู่เข้าก่อนกำหนด และช่วยเพิ่มความเร็วในการค้นหาคำตอบเมื่อประชากรใกล้คอนเวอร์เจนซ์ (convergence)

3.3 วิธีการคัดเลือก

การคัดเลือกเป็นกระบวนการที่ใช้ในการเลือกสมาชิกแต่ละตัวในประชากรสำหรับกระบวนการรีโพรดักชั่น โครโมโซมลูกหลานที่ดีเกิดขึ้นได้จากกลไกการคัดเลือกโครโมโซมพ่อแม่ที่ดี ดังนั้นจึงมีมาตรการการวัดการทำงานของอัลกอริทึมการคัดเลือกดังนี้ [15,27]

- ความลำเอียง (Bias) คือ ค่าสัมบูรณ์ของความแตกต่างระหว่างค่าจริงกับความน่าจะเป็นที่คาดหวังไว้สำหรับการคัดเลือกของสมาชิกแต่ละตัว ความลำเอียงจะมีค่าเป็นศูนย์เมื่อความน่าจะเป็นในการถูกเลือกของสมาชิกแต่ละตัวเท่ากับความน่าจะเป็นที่คาดหวังไว้

- การกระจาย (Spread) คือ ย่านของจำนวนทดลองที่เป็นไปได้ที่สมาชิกตัวหนึ่งจะถูกเลือก ถ้า ตัวอย่างเช่น g(i) คือ จำนวนทดสอบของสมาชิกลำดับที่ i การกระจายต่ำสุด คือ การกระจายที่น้อยที่สุดที่ยินยอมให้มีความลำเอียงเป็นศูนย์ตามทฤษฎี

$$g(i) \in \{\lfloor et(i) \rfloor, \lceil et(i) \rceil\} \tag{3.7}$$

เมื่อ $et(i)$ คือ จำนวนทดสอบที่คาดหวังไว้ของสมาชิกแต่ละตัวที่ i , $\lfloor et(i) \rfloor$ คือ ขอบเขตล่าง และ $\lceil et(i) \rceil$ คือ ขอบเขตบน ดังนั้น การกระจายจะเป็นตัววัดความเที่ยงของวิธีการคัดเลือก

- ประสิทธิภาพ จะขึ้นอยู่กับความซับซ้อนของเวลาทั้งหมดสำหรับอัลกอริทึม

วิธีการคัดเลือกที่ดีควรจะมีความลำเอียงเป็นศูนย์ มีการกระจายที่ต่ำ และไม่เพิ่มความซับซ้อนของเวลาในการประมวลผล GA มากเกินไป

เทคนิคการคัดเลือกต่างๆมักจะใช้กลไกของวงล้อรูเลท วิธีการคัดเลือกแบบวงล้อรูเลทอย่างง่าย คือ การเลือกแบบสุ่มโดยมีการแทนที่(SSR : Stochastic Sampling with Replacment) วิธีนี้ขนาดเซกเมนท์และความน่าจะเป็นในการเลือกจะคงที่ตลอดช่วงการคัดเลือก SSR มีแนวโน้มให้ความลำเอียงเป็นศูนย์แต่มีแนวโน้มที่เพิ่มการกระจายซึ่งมีไม่จำกัด

การเลือกแบบสุ่มโดยมีการแทนที่บางส่วน(SSPR : Stochastic Sampling with Partial Replacement) เป็นวิธีที่มีการปรับขนาดเซกเมนท์ของโครโมโซมถ้าโครโมโซมนั้นมีการถูกเลือกในแต่ละครั้งที่โครโมโซมถูกเลือกจะมีการลดขนาดเซกเมนท์ด้วยแฟกเตอร์ค่าหนึ่ง ถ้าขนาดเซกเมนท์ของโครโมโซมมีค่าเป็นลบ ขนาดเซกเมนท์ก็จะถูกกำหนดเป็นศูนย์ วิธีนี้จะกำหนดขอบเขตบนของการกระจาย $\lceil et(i) \rceil$ แต่ขอบเขตล่างจะมีค่าเป็นศูนย์และมีความลำเอียงที่สูงขึ้น วิธีการคัดเลือกแบบวงล้อรูเรทมีความซับซ้อนของเวลาเป็น NlogN เมื่อ N คือ ขนาดของประชากร

การเลือกแบบยูนิเวอร์ซอลโดยสุ่ม (SUS : Stochastic Universal Sampling) เป็นวิธีที่มีการกระจายต่ำ ความลำเอียงเป็นศูนย์ และความซับซ้อนของเวลาเป็น N วิธีคัดเลือกแบบ SUS จะใช้ตัวชี้ N ตัวที่มีระยะห่างเท่ากันในการเลือกประชากรในครั้งเดียว ประชากรจะถูกสลับแบบสุ่มและจะสุ่มตัวเลขค่าหนึ่งในช่วง [0,Fsum/N] เป็น ptr ดังนั้นตัวชี้ทั้ง N ตัวจะมีค่าเป็น [ptr,ptr+Fsum/N,…,ptr+(N-1)Fsum/N] โครโมโซมที่ถูกเลือกคือโครโมโซมที่คลอบคลุมพื้นที่ในตำแหน่งของตัวชี้

3.4 ตัวดำเนินการเจเนติก

3.4.1 การครอสโอเวอร์

แม้ว่าการครอสโอเวอร์แบบ 1 จุด เป็นวิธีการที่ได้แนวคิดมาจากกระบวนการทางชีววิทยา แต่ก็มีข้อเสียในบางสถานการณ์ที่สคีมาบางตัวไม่สามารถที่จะรวมกันได้ ตัวอย่างเช่น มีสคีมา 2 ตัว คือ S1 และ S2

S1 = 1 0 1 * * * * 1

S2 = * * * * 1 1 * *

มีโครโมโซมอยู่ 2 ตัว คือ I1 และ I2 ที่ตรงกับ S1 และ S2 ตามลำดับ

I1 = 1 0 1 1 0 0 0 1

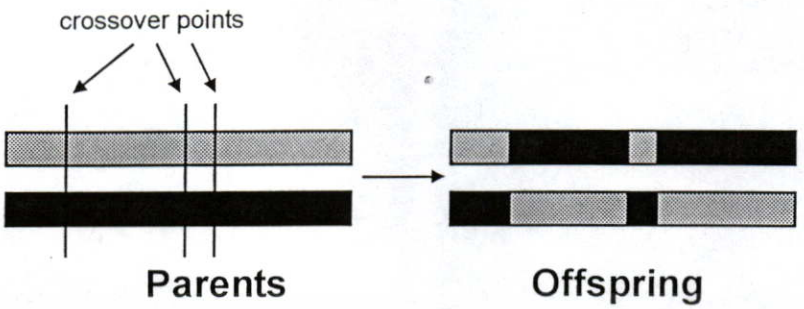
I2 = 0 1 1 0 1 1 0 0

ถ้าใช้การครอสโอเวอร์แบบ 1 จุด จะไม่มีทางเป็นไปได้เลยที่จะมีโครโมโซมตรงกับสคีมา S3

S3 = 1 0 1 * 1 1 * 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การครอสโอเวอร์แบบหลายจุดจะช่วยแก้ปัญหานี้ได้ โดยจะช่วยเพิ่มความสามารถในการสร้างโครโมโซมลูกหลาน ตัวอย่างการทำงานของการครอสโอเวอร์แบบหลายจุดนี้แสดงในรูปที่ 3.3 โดยที่ตำแหน่งของการครอสโอเวอร์จะถูกเลือกโดยวิธีการสุ่ม



รูปที่ 3.3 ตัวอย่างการครอสโอเวอร์แบบหลายจุด

เพื่อให้เข้าใจในการเพิ่มความสามารถของการสร้างโครโมโซมลูกหลานของการครอสโอเวอร์แบบหลายจุด จะยกตัวอย่างเช่น สมมุติว่ามีการใช้การครอสโอเวอร์ I1 และ I2 แบบ 2 จุด ทำให้ได้โครโมโซมลูกหลาน คือ I3 และ I4 จะพบว่า I3 ตรงกับสคีมา S3

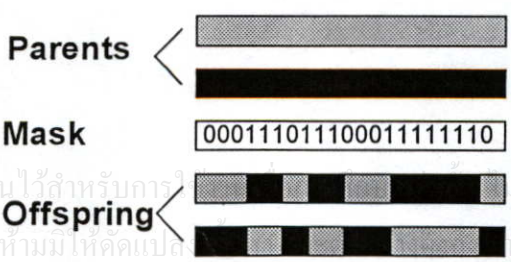
I1 = 1 0 1 1| 0 0| 0 1

I2 = 0 1 1 0| 1 1| 0 0

I3 = 1 0 1 1 1 1 0 1

I4 = 0 1 1 0 0 0 0 0

วิธีการครอสโอเวอร์อีกวิธี คือ การครอสโอเวอร์แบบยูนิฟอร์ม [28] ซึ่งวิธีนี้จะสร้างโครโมโซมลูกหลานโดยขึ้นอยู่กับการสร้าง mask โดยการสุ่ม การทำงานของการครอสโอเวอร์วิธีนี้แสดงในรูปที่ 3.4 จากรูปที่ 3.4 จะสังเกตได้ว่าการครอสโอเวอร์แบบยูนิฟอร์มจะมีการแลกเปลี่ยนกันระหว่างบิต มากกว่าที่จะแลกเปลี่ยนกันเป็นส่วน (segment) ทำให้การครอสโอเวอร์แบบยูนิฟอร์มมีประสิทธิภาพมากกว่าสำหรับบางปัญหา เช่น ช่วยลดการทำลายบิวล์ดิ้งบล๊อก



สารนี้เป็นเอกสารที่สงวนไว้สำหรับการ ...อนุญาตให้นำไปใช้ประโยชน์ด้านกา
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปล... ของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.4 ตัวอย่างของการครอสโอเวอร์แบบยูนิฟอร์ม

จากตัวอย่างการครอสโอเวอร์แบบต่างๆที่ยกมา จะพบว่าการครอสโอเวอร์ยังคงอยู่บนแนวคิดพื้นฐาน คือ ต้องการให้มีการแลกเปลี่ยนข้อมูลกันระหว่างโครโมโซม และมีข้อสังเกตว่าการออกแบบวิธีการครอสโอเวอร์ที่มีประสิทธิภาพ จะช่วยเพิ่มการคอนเวอร์เจนซ์ให้เร็วขึ้น

3.4.2 การมิวเทชั่น

การมิวเทชั่นเป็นการสร้างความหลากหลายให้แก่โครโมโซม แต่โดยทั่วไปแล้วการมิวเทชั่นจะออกแบบไว้สำหรับโครโมโซมที่เข้ารหัสแบบไบนารีเท่านั้น การมิวเทชั่นแบบสุ่มเป็นวิธีหนึ่งมีการใช้กันในกรณีที่ใช้โครโมโซมแบบจำนวนจริง โดยเป็นไปตามสมการที่ (3.8)

$$g = g + \psi(\mu, \sigma) \tag{3.8}$$

เมื่อ g คือ ยีนส์ที่เป็นค่าจริง ψ คือ ฟังก์ชั่นสุ่ม ซึ่งอาจจะเป็น Gaussian หรือ การสุ่มแบบกระจายปกติ (normal distribution) μ, σ คือ คือ ค่าเฉลี่ยและส่วนเบี่ยงเบนซึ่งสัมพันธ์กับฟังก์ชั่นสุ่ม ตามลำดับ

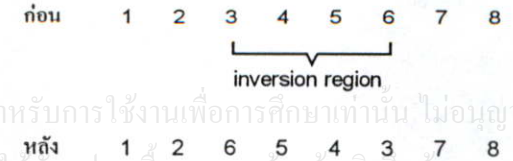
3.4.3 การตั้งอัตราการทำงาน

การตั้งอัตราทำงานหรือความน่าจะเป็นในการครอสโอเวอร์และมิวเทชั่น ยังเป็นสิ่งที่มีการโต้แย้งกันอยู่ การเพิ่มความน่าจะเป็นในการครอสโอเวอร์จะเป็นสาเหตุให้เกิดการรีคอมบิเนชั่นของบิวล์ดิ้งบล๊อกเพิ่มขึ้น แต่ในขณะเดียวกันก็จะเพิ่มการทำลายโครโมโซมที่ดีด้วย ในขณะที่ การเพิ่มความน่าจะเป็นในการมิวเทชั่น จะทำให้การค้นหาคำตอบเป็นไปในลักษณะสุ่มมากขึ้น

อย่างไรก็ตาม ความน่าจะเป็นทั้งสองนี้อาจปรับเปลี่ยนไปตามเจนเนอเรชั่น ดังเช่น ข้อแนะนำของ Davis [25] ที่แนะนำให้ลดความน่าจะเป็นในการครอสโอเวอร์ และเพิ่มความน่าจะเป็นในการมิวเทชั่นอย่างเป็นเชิงเส้นตามเจนเนอเรชั่น

3.4.4 การจัดลำดับใหม่

วัตถุประสงค์ของการจัดลำดับใหม่ เพื่อหาลำดับของยีนส์ที่ทำให้การพัฒนาคำตอบดีที่สุด กระบวนการนี้มีรู้จักกันดีในชื่อ อินเวอร์ชั่น (Inversion) [26] ขั้นตอนนี้เริ่มจากการสุ่มตำแหน่งที่จะมีการอินเวอร์ชั่น ดังในรูปที่ 3.5 แสดงลำดับก่อนและหลังการทำอินเวอร์ชั่น โดยตำแหน่งการอินเวอร์ชั่นถูกเลือกที่ 3 และ 6



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.5 ตัวอย่างการจัดลำดับใหม่

3.5 การแทนที่

ในกรณีการแทนที่แบบเจนเนอเรชั่น (Generation replacement) หลังจากที่สร้างประชากรลูกหลานแล้ว จะมีการแทนที่โครโมโซมในประชากรเจนเนอเรชั่นปัจจุบันด้วยโครโมโซมลูกหลานทุกตัว เช่น ถ้าหากขนาดประชากรมีค่าเท่ากับ N วิธีนี้ก็จะต้องสร้างประชากรลูกหลานเท่ากับ N ด้วย วิธีการแทนที่ดังกล่าวนี้อาจทำให้โครโมโซมที่ดีเสียหายในเจนเนอเรชั่นถัดไป ดังนั้นจึงมักจะมีการทำอิลิทิสซึม โดยการทำสำเนาโครโมโซมที่ดีที่สุด 1 ตัว หรือ 2-3 ตัว ในเจนเนอเรชั่นถัดไป อิลิทิสซึมจะช่วยเพิ่มความเร็วในการหาคำตอบที่ดีที่สุดถ้าหากกำหนดจำนวนสำเนาโครโมโซมที่ดีที่สุดได้อย่างสมดุล

เนื่องจากว่าการสร้างประชากรลูกหลานที่มีขนาดใหญ่ จะทำให้การคำนวณในแต่ละรอบของ GA นานขึ้น จึงมีวิธีการแทนที่อื่นๆที่สร้างประชากรลูกหลานจำนวนน้อยลงและช่วยเพิ่มความเร็วในการคำนวณ เช่น การแทนที่โครโมโซมพ่อแม่ด้วยโครโมโซมลูกหลานลูกตรง โดยอาจเลือกแทนที่โครโมโซมที่เลวที่สุดเมื่อมีการแทรกโครโมโซมใหม่เข้ามาในประชากร หรืออาจแทนที่โครโมโซมที่อาศัยอยู่ในประชากรเป็นเวลานาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การใช้ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

การชดเชยกำลังไฟฟ้ารีแอกทีฟเพื่อรักษาแรงดันในระบบไฟฟ้ากำลัง เป็นที่รู้จักกันเป็นอย่างดี ในยุคปัจจุบันที่อิเลกทรอนิกส์กำลังมีความรุดหน้าไปอย่างรวดเร็ว ทำให้การออกแบบตัวชดเชยกำลังไฟฟ้ารีแอกทีฟโดยใช้ไทริสเตอร์ [32,33] สามารถนำมาใช้ได้ในทางปฏิบัติและมีความประหยัดในทางเศรษฐศาสตร์

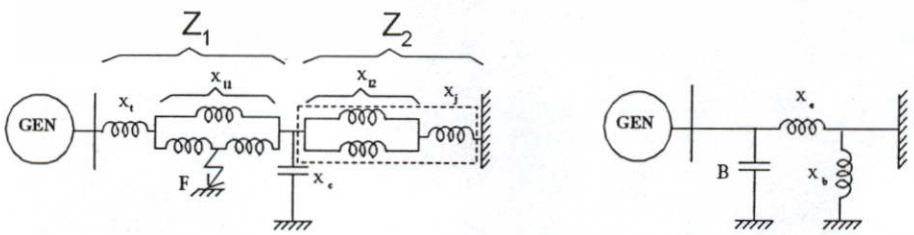
วัตถุประสงค์แรกของการประยุกต์ใช้ SVC คือ ใช้รักษาแรงดันไฟฟ้าที่บัส ให้คงที่หรือใกล้เคียงระดับคงที่ค่าหนึ่ง นอกเหนือจากนี้แล้ว SVC ยังอาจใช้ในการปรับปรุงเสถียรภาพทรานเซียนส์และเสถียรภาพภาวะอยู่ตัว โดยจะช่วยเพิ่มแดมปิ้งในการแกว่งไกว [30,31] หลักการของการปรับปรุงเสถียรภาพทรานเซียนส์ด้วย SVC สามารถอธิบายได้โดยใช้กฎพื้นที่เท่ากัน [9]

การใช้แรงดันเป็นสัญญาณควบคุม SVC แต่เพียงอย่างเดียว ไม่สามารถทำให้เกิดแดมปิ้งในระบบได้มากพบ การเพิ่มแดมปิ้งในระบบสามารถทำได้โดยสร้างสัญญาณควบคุมเสริม เพิ่มเข้าไปในลูปควบคุมแรงดัน นอกจากนี้ยังพบว่า ตำแหน่งที่ติดตั้ง SVC ที่สามารถสร้างแดมปิ้งในระบบได้สูงสุด คือ บริเวณจุดกึ่งกลางของสายส่ง [8,9]

4.1 การชดเชยกำลังไฟฟ้ารีแอกทีฟด้วย SVC

4.1.1 ผลกระทบของ SVC ต่อระบบไฟฟ้ากำลัง

SVC สามารถแทนด้วยซัสเซบแตนซ์ B_{SVC} ที่ปรับค่าได้ เมื่อ SVC จ่ายกำลังไฟฟ้ารีแอกทีฟ B_{SVC} จะเป็นคาปาซิทีฟและ $B_{SVC} > 0$ เมื่อ SVC รับกำลังไฟฟ้ารีแอกทีฟ SVC จะทำหน้าที่เป็นโหลดอินดักทีฟและ $B_{SVC} < 0$



รูปที่ 4.1 การแปลง Y เป็น Δ วงจรสมมูลย์ระบบจำลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

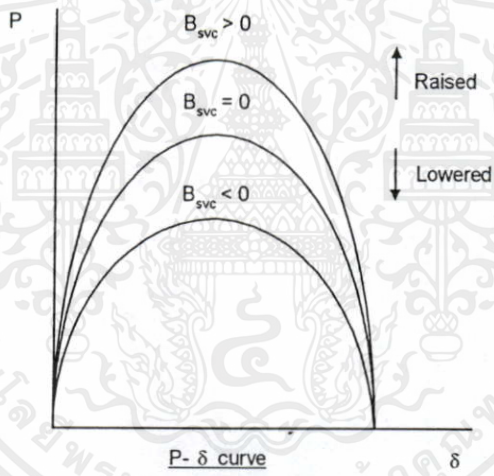
พิจารณาวงจรสมมูลย์ระบบจำลองในรูปที่ 4.1(a) ซึ่งต่อแบบ Y และสามารถแปลงเป็น Δ ได้ในรูปที่ 4.1(b) รีแอกแตนซ์ถ่ายโอน x_e สามารถคำนวณได้จาก

$$x_e = Z_1 + Z_2 - B_{svc}Z_1Z_2 \tag{4.1}$$

จากสมการที่ (4.1) เมื่อ SVC เป็นโหลดคาปาซิทีฟ $B_{SVC} > 0$ ทำให้ $x_e < Z_1+Z_2$ และเมื่อ SVC เป็นโหลดอินดักทีฟ $B_{SVC} < 0$ ทำให้ $x_e > Z_1+Z_2$ ความสัมพันธ์ P-δ สามารถเขียนได้ดังนี้

$$P = \frac{V_t V_b \sin\delta}{x_e} \tag{4.2}$$

เมื่อ $B_{SVC} > 0$ กราฟ P-δ จะถูกยกขึ้น และ $B_{SVC} < 0$ กราฟ P-δ จะถูกยกลง ดังแสดงในรูปที่ 4.2



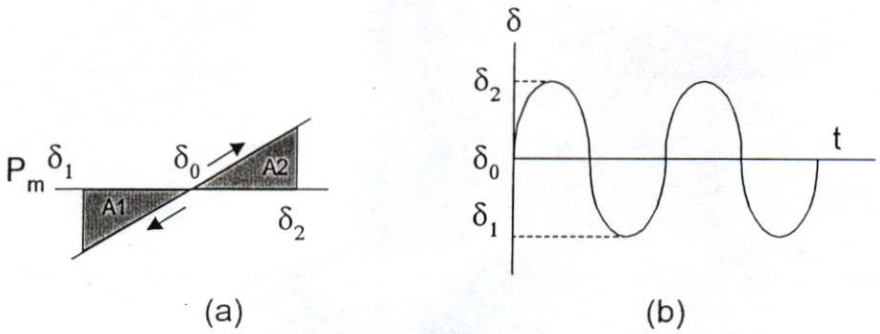
รูปที่ 4.2 กราฟ P-δ

จะพบว่า เมื่อเปลี่ยนแปลงเอาพุทของ SVC จะทำให้คุณลักษณะถ่ายโอนกำลัง หรือ กราฟ P-δ ของระบบเปลี่ยนแปลง ดังนั้นผลกระทบของ SVC ต่อไดนามิกของระบบไฟฟ้ากำลัง จึงสามารถอธิบายได้จากการเปลี่ยนแปลงของกราฟ P-δ แบบไดนามิก

4.1.2 วิธีการควบคุม SVC

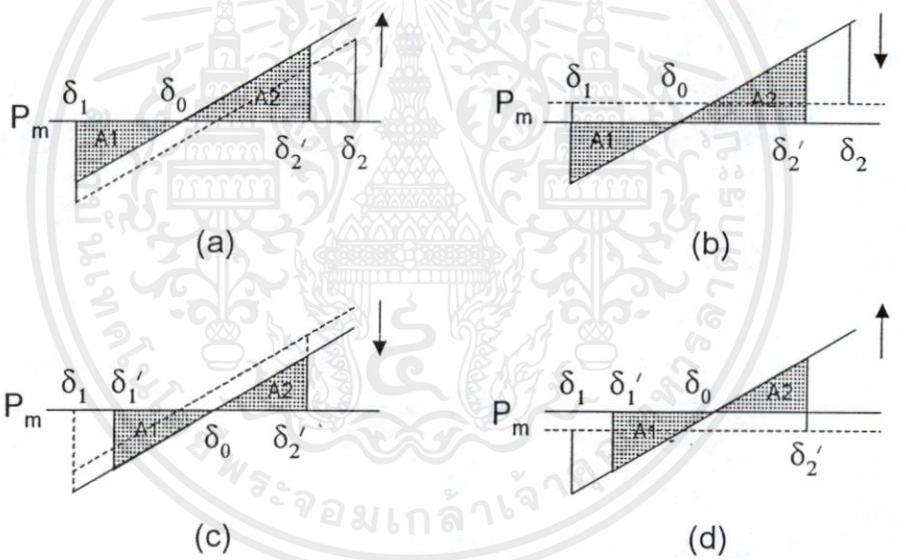
เมื่อระบบในรูปที่ 4.1 ไม่มี SVC และไม่มีแดมปิ้ง จากนั้นสมมุติให้เกิดการรบกวนแบบรุนแรง มุม δ ของเครื่องกำเนิดไฟฟ้าจะแกว่งด้วยขนาดคงที่ ดังแสดงในรูปที่ 4.3(b) ตามกฎพื้นที่เท่ากัน การแกว่งไกวจะเกิดขึ้นโดยพื้นที่ $A_1 = A_2$ ดังในรูปที่ 4.3(a) เมื่อพื้นที่ A_1 คือ พื้นที่เร่งเมื่อ δ กำลังเพิ่มขึ้น และ $\Delta\omega > 0$ พื้นที่A_2 คือ พื้นที่เร่ง เมื่อ δ กำลังลดลง และ $\Delta\omega < 0$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.3 การแกว่งไกวเมื่อไม่มีแดมปิ้ง

พื้นที่เร่งหรือพลังงานทรานเซียนส์ในรูปที่ 4.3(a) จะต้องถูกลดลงเมื่อเกิดการแกว่งไกวด้วยวิธีการควบคุม เพื่อทำให้ระบบกลับเข้าสู่จุดสมดุล δ_0 เมื่อ δ กำลังเพิ่มขึ้น และ $\Delta\omega > 0$ กราฟ P-δ ควรจะถูกยกขึ้นดังรูปที่ 4.4(a) และหรือ P_m ควรจะลดลงดังรูปที่ 4.4(b) เมื่อ δ กำลังลดลง และ $\Delta\omega < 0$ กราฟ P-δ ควรจะถูกยกลงดังรูปที่ 4.4(c) และหรือ P_m ควรจะเพิ่มขึ้นดังรูปที่ 4.4(d)



รูปที่ 4.4 การควบคุมอย่างง่ายเพื่อเพิ่มแดมปิ้งในระบบ

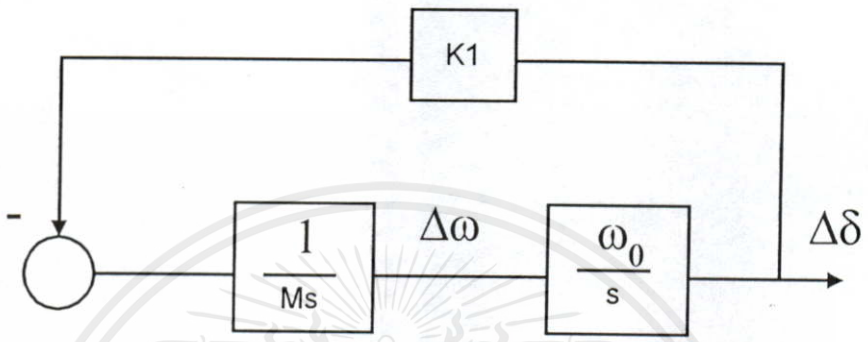
จากรูปที่ 4.4 พบว่าเมื่อเปลี่ยนแปลงกราฟ P-δ หรือมีการปรับกำลังกลอินพุท P_m การแกว่งไกวของมุม δ คือ δ_1 -> δ_2' -> δ_1' ซึ่งจะใกล้เคียงกับ δ_0 มากกว่าการแกว่งไกวที่มีขนาดคงที่ ซึ่งแกว่งไกวในช่วง δ_1 <-> δ_2

GOV ซึ่งทำหน้าที่ควบคุมกำลังกลของเครื่องกำเนิดไฟฟ้า มีผลกระทบต่อแดมปิ้งในระบบน้อยมาก เนื่องจากผลตอบสนองที่ช้าของ GOV กราฟ P-δ สามารถเพิ่มได้จากการใช้ PSS ควบคุม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ร่วมกับ AVR ในการปรับแรงดันฟิลด์ของเอกไซเตอร์ [34] หรือใช้ SVC โดยการปรับ B_{svc} แต่ข้อได้เปรียบของการใช้ SVC คือ สามารถปรับกราฟ P-δ ได้กว้างกว่า และมีความเร็วสูง

จากแบบจำลองของ Phillips-Heffron [20] ในรูปที่ 4.5 ซึ่งใช้ศึกษาแดมปิ้งของระบบไฟฟ้ากำลัง K1 คือ สัมประสิทธิ์ซิงโครไนซ์ทอร์ก M คือ ค่าคงที่ความเฉื่อย จะพบว่ามีตัวแปรสถานะ 2 ตัว คือ $\Delta\omega$ และ $\Delta\delta$ ดังนั้นการปรับเอาพุท B_{svc} ของ SVC อาจควบคุมให้สอดคล้องกับตัวแปรสถานะทั้งสองนี้ได้



รูปที่ 4.5 บล๊อกไดอะแกรมแบบจำลองของ Phillips-Heffron

4.1.2.1 การควบคุมด้วย $\Delta\delta$

วิธีนี้เอาพุท B_{svc} ของ SVC ควบคุมจากสมการ

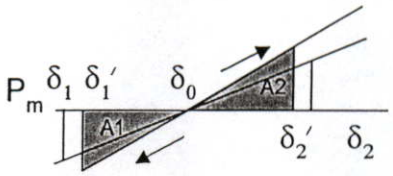
$$Bsvc = K_{\Delta\delta}\Delta\delta \tag{4.3}$$

เมื่อ $K_{\Delta\delta}$ คือ เกนของการควบคุม และ $\Delta\delta = \delta - \delta_0$ ลูปควบคุมของ SVC มีความไวมากเมื่อเทียบกับการแกว่งไกวของระบบ ดังนั้นเวลาหน่วงในลูปควบคุมของ SVC จึงตัดทิ้งออกไปได้

สมมุติว่าระบบซึ่งมี SVC ต่ออยู่ ถูกรบกวนด้วยการรบกวนแบบเล็กน้อย เมื่อ $\Delta\delta > 0$ B_{svc} จะมีค่าเป็นบวก ดังนั้นกราฟ P-δ จะถูกยกขึ้น เมื่อ $\Delta\delta < 0$ B_{svc} จะมีค่าเป็นลบ ดังนั้นกราฟ P-δ จะถูกยกลง

จะพบว่า SVC ที่ควบคุมด้วย $\Delta\delta$ สามารถลดขนาดการแกว่งไกวลงเป็น δ_1' <--> δ_2' แทนที่จะเป็น δ_1 <--> δ_2 ดังแสดงในรูปที่ 4.6 การเพิ่มของสัมประสิทธิ์ซิงโครไนซ์ทอร์กแสดงได้จากการเพิ่มความชันของกราฟ P-δ จะสังเกตได้ชัดเจนว่า SVC ควบคุมด้วย $\Delta\delta$ ไม่มีผลต่อแดมปิ้งของระบบโดยจะยังมีการแกว่งไกวอยู่แต่มีขนาดที่ลดลงเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.6 การแกว่งไกวเมื่อควบคุม SVC ด้วย $\Delta\delta$

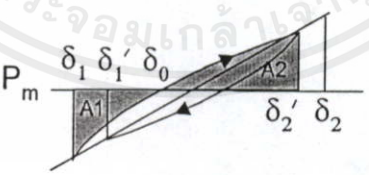
การควบคุม SVC ด้วยแรงดันที่บัสของ SVC ไม่สามารถสร้างแดมปิ้งในระบบได้มากพอ [2,7] เนื่องจากการเปลี่ยนแปลงแรงดันบัสของ SVC เกือบมีเฟสเดียวกับ $\Delta\delta$ ดังนั้นการควบคุม SVC ด้วยแรงดันบัสจึงเป็นการควบคุม SVC ด้วย $\Delta\delta$ วิธีหนึ่ง

4.1.2.2 การควบคุมด้วย $\Delta\omega$

วิธีนี้เอาพุท B_{SVC} ของ SVC ควบคุมจากสมการ

$$B_{SVC} = K_{\Delta\omega}\Delta\omega \tag{4.4}$$

เมื่อ $K_{\Delta\omega}$ คือ เกนของการควบคุม ผลกระทบต่อกราฟ P-δ แสดงในรูปที่ 4.7 เมื่อ δ กำลังเพิ่มขึ้น และ $\Delta\omega > 0$ B_{SVC} จะมีค่าเป็นบวก ดังนั้นกราฟ P-δ จะถูกยกขึ้น เมื่อ δ กำลังลดลง และ $\Delta\omega < 0$ B_{SVC} จะมีค่าเป็นลบ ดังนั้นกราฟ P-δ จะถูกยกลง จากการเปลี่ยนแปลงแบบไดนามิกของกราฟ P-δ ที่สอดคล้องกับ $\Delta\omega$ จะสังเกตได้ชัดเจนว่า สามารถเพิ่มแดมปิ้งในระบบได้ โดยมุม δ มีการเปลี่ยนแปลง δ_1 --> δ_2' --> δ_1' ดังรูปที่ 4.7 ซึ่งใกล้เคียงจุดสมดุล δ_0 มากกว่าการที่ไม่มี SVC ซึ่งมีการแกว่งในช่วง δ_1 <--> δ_2



รูปที่ 4.7 การแกว่งไกวเมื่อควบคุม SVC ด้วย $\Delta\omega$

แดมปิ้งในระบบสามารถสูงขึ้นได้จากสัญญาณควบคุมเสริมเพิ่มจากลูปควบคุมแรงดันของ SVC โดยสัญญาณควบคุมเสริมนี้ควรจะมีเฟสเดียวกับ $\Delta\omega$ แต่ปกติแล้วสัญญาณ $\Delta\omega$ จะไม่มีณ ตำแหน่งที่ติดตั้ง SVC โดยตรง ดังนั้นสัญญาณอื่นๆที่อาจนำมาใช้แทนได้ เช่น ความถี่ กำลังไฟฟ้า กระแส เป็นต้น

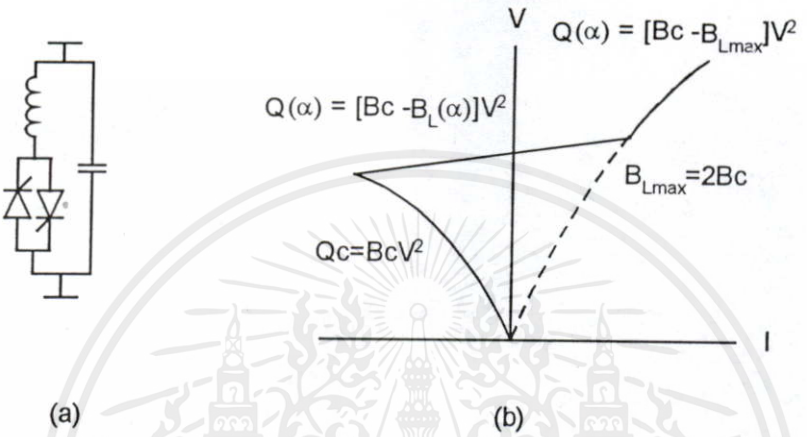
สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของสารนิพนธ์นี้ทุกครั้งที่มีการนำไปใช้

4.2 ชนิดและประโยชน์ของ SVC

4.2.1 ชนิดของ SVC

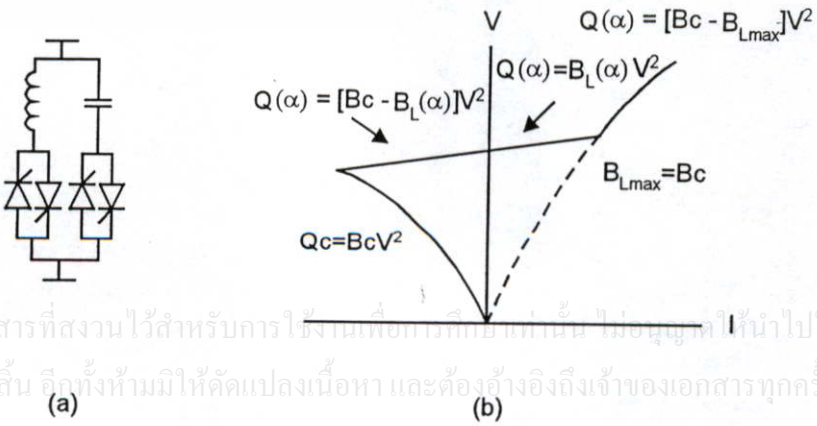
ตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ สามารถแบ่งได้เป็น 2 ชนิด คือ

1. ไทริสเตอร์คอนโทลรีแอกเตอร์ (TCR : Thyristor controlled reactor) และ ตัวเก็บประจุคงที่ (FC : Fixed Capacitor) หรือเรียกย่อว่า TCR/FC ตัวเก็บประจุคงที่จะใช้การสวิทช์ทางกล โดยมีรูปแบบการต่อแสดงในรูปที่ 4.8(a) และมีกราฟคุณลักษณะ V-I แสดงในรูปที่ 4.8(b)



รูปที่ 4.8 TCR/FC

2. ไทริสเตอร์สวิทช์คาปาซิเตอร์(TSC : Thyristor swithched capacitor) และ ไทริสเตอร์คอนโทรลรีแอกเตอร์ หรือเรียกย่อว่า TSC/TCR การใช้ TSC ร่วมกับ TCR เป็นการแก้ปัญหาของการใช้ TCR/FC ในเรื่อง การสร้างฮาร์โมนิก คุณลักษณะเรื่องของความสูญเสีย สมรรถนะในการทำงานเมื่อเกิดการรบกวนแบบรุนแรง รูปแบบการต่อ TSC/TCR แสดงในรูปที่ 4.9(a) และกราฟคุณลักษณะ V-I แสดงในรูปที่ 4.9(b)



รูปที่ 4.9 TSC/TCR

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจาก TSC สามารถสวิทช์ เปิด/ปิด ตัวเก็บประจุได้อย่างรวดเร็ว ในขณะที่ตัวเก็บประจุแบบคงที่ไม่สามารถสวิทช์ เปิด/ปิด ได้อย่างทันทีทันใด ทำให้ B_{Lmax} ของ TSC/TCR มีค่าน้อยกว่า B_{Lmax} ของ TCR/FC เมื่อจ่ายกำลังไฟฟ้ารีแอกทีฟได้เท่ากัน

4.2.2 ประโยชน์ของ SVC

ประโยชน์ของ SVC ในระบบไฟฟ้ากำลังสามารถสรุปได้ดังนี้

1. รักษาแรงดันให้อยู่ในระดับคงที่หรือใกล้เคียง

(a) ภายใต้ภาวะการเปลี่ยนแปลงอย่างช้าๆของโหลด

(b) แก้ไขการเปลี่ยนแปลงแรงดันเนื่องจากเหตุการณ์ที่ไม่พึงประสงค์ เช่น การปลดโหลด การล้มเหลวของเครื่องกำเนิดไฟฟ้าบางตัวในระบบไฟฟ้า

(c) ลดการกระพริบของแรงดันเนื่องจากการเปลี่ยนแปลงขึ้นลงของโหลด เช่น โหลดเตาหลอมแบบอาร์ก

2. ปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

(a) ปรับปรุงเสถียรภาพแรงดัน ณ จุดสำคัญ เช่น จุดกึ่งกลางของสายส่งระยะไกล

(b) ช่วยเหลือในการปรับปรุงแดมปิ้งของระบบ

3. ช่วยปรับปรุงค่าเพาเวอร์แฟกเตอร์

4. แก้ไขปัญหาเฟสไม่สมดุล

4.3 เปรียบเทียบคุณลักษณะการชดเชยกำลังไฟฟ้ารีแอกทีฟชนิดควบคุมได้แบบต่างๆ

การชดเชยกำลังไฟฟ้ารีแอกทีฟชนิดควบคุมได้แบบต่างๆ ได้แก่ ซิงโครนัสคอนเดนเซอร์ (SD : Synchronous condensor), รีแอกเตอร์แบบอิ่มตัว (SR : Saturated reactor), TCR, TSC, TCR/FC และ TSC/TCR มีคุณลักษณะต่างๆที่ไม่เหมือนกัน ดังสรุปในตารางที่ 4.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.1 เปรียบเทียบคุณลักษณะต่างๆของการชดเชยกำลังไฟฟ้ารีแอกทีฟชนิดควบคุมได้

คุณลักษณะ	ซิงโครนัสคอนเดนเซอร์	TCR (มีตัวเก็บประจุขนาน เมื่อจำเป็น)	TSC (มีTCR เมื่อจำเป็น)	รีแอกเตอร์แบบอิ่มตัวหลายเฟส
การสร้างและติดตั้ง	เครื่องจักรกลหมุน	เป็นอุปกรณ์ไทริสเตอร์ และประกอบด้วยรีแอกเตอร์แบบสถิตย์ และตัวเก็บประจุคงที่	เป็นอุปกรณ์ไทริสเตอร์และประกอบด้วยตัวเก็บประจุแบบสถิตย์	เป็นอุปกรณ์ประกอบด้วยหม้อแปลงและตัวเก็บประจุขนาน
ความสามารถในการ จ่าย/รับ กำลังรีแอกทีฟ	ได้ทั้ง นำหน้า/ล้าหลัง	ได้ทั้ง นำหน้า/ล้าหลัง (ทางอ้อม)	ได้ทั้ง นำหน้า/ล้าหลัง (ทางอ้อม)	ได้ทั้ง ล้าหลัง/นำหน้า (ทางอ้อม)
การควบคุม	ได้อย่างต่อเนื่อง	ได้อย่างต่อเนื่อง	ไม่ต่อเนื่อง (ถ้ามี TCR ทำงานร่วมจะทำได้ต่อเนื่องและละเอียด)	ได้อย่างต่อเนื่อง
เวลาการตอบสนอง	ช้า	ไว แต่ขึ้นกับคุณลักษณะของระบบ	ไว แต่ขึ้นกับคุณลักษณะของระบบ	ไว แต่ขึ้นกับคุณลักษณะของระบบ และจะถูกทำให้ช้าลงด้วยตัวเก็บประจุแก้ไขความชัน
ฮาร์โมนิกส์	ไม่เกิดเลย	มีบ้าง จำเป็นต้องใช้ ฟิลเตอร์ขึ้นกับสภาพของระบบ	ไม่เกิดเลย	มีบ้างเล็กน้อย เฉพาะฮาร์โมนิก ที่ 17 และ 19
การสูญเสีย	มีปานกลาง	มีน้อย แต่จะเพิ่มขึ้นเมื่อกระแสเพิ่มขึ้น	มีน้อย แต่จะเพิ่มขึ้นเมื่อกระแสเพิ่มขึ้น	มีน้อย แต่จะเพิ่มขึ้นเมื่อกระแสเพิ่มขึ้น

ตารางที่ 4.1 (ต่อ)

คุณลักษณะ	Synchronous condenser	TCR (with shunt capacitor เมื่อจำเป็น)	TSC (with TCR เมื่อจำเป็น)	Poly phase saturated reactor
การควบคุมโดยใช้ โปรแกรม	ดี แต่ต้องอยู่ในช่วงจำกัดของผลตอบสนองความเร็ว	ดีมาก	ดี แต่ต้องทำงานร่วมกับ TCR	ไม่ดีนัก
ความสามารถในการทำสมดุลเฟส	จำกัด	ดี	จำกัด	จำกัด
ขีดจำกัดในเรื่องแรงดันเกิน	ดี	ปานกลาง	ไม่ดี	ดี แต่ต้องอยู่ภายในขีดจำกัดของตัวเก็บประจุแก้ไขความชัน
ความเฉื่อยหมุน	มี	ไม่มี	ไม่มี	ไม่มี
ความถูกต้องแม่นยำในการชดเชย	ดี	ดีมาก	ดี และจะดีมาก ถ้าทำงานร่วมกับ TCR	ดี
การต่อตรงกับ EHV	ไม่ได้	รีแอกเตอร์ไม่ได้ ตัวเก็บประจุ ได้	ไม่ได้	SR ไม่ได้ ตัวเก็บประจุ ได้
การสตาร์ท	ช้า	เร็ว และมีกระแสทรานเซียนส์เล็กน้อย	เร็ว แต่มีกระแสทรานเซียนส์	เร็ว แต่มีกระแสทรานเซียนส์

บทที่ 5
วิเคราะห์ระบบจำลอง

เสถียรภาพของระบบไฟฟ้ากำลังจะพิจารณาจากผลตอบสนองการเปลี่ยนแปลงความเร็วเครื่องกำเนิดไฟฟ้าซิงโครนัสเป็นหลัก ถ้าเครื่องกำเนิดไฟฟ้าสามารถกลับมาหมุนที่ความเร็วซิงโครนัสได้อีกครั้งหลังจากถูกสิ่งรบกวน ไม่ว่าจะเป็นการรบกวนแบบรุนแรงหรือการรบกวนแบบเล็กน้อย จะถือว่าระบบมีเสถียรภาพ การศึกษาผลตอบสนองของเครื่องกำเนิดไฟฟ้าต่อสิ่งรบกวนจะต้องมีการสร้างแบบจำลอง โดยแบบจำลองที่ใช้จะต้องให้ผลที่ใกล้เคียงกับระบบจริง ด้วยเหตุนี้แบบจำลองที่ใช้จึงต้องรวมอุปกรณ์ควบคุมเครื่องกำเนิดไฟฟ้าต่างๆ คือ GOV (Governor) , AVR (Automatic voltage regulator) และ PSS (Power system stablizer) การติดตั้ง GOV มีวัตถุประสงค์เพื่อควบคุมความเร็วของเครื่องกำเนิดไฟฟ้า โดยควบคุมการปิดเปิดวาล์วของสตีมหรือน้ำที่เทอร์ไบน์ ซึ่งจะมีผลต่อความถี่ไฟของระบบ AVR มีวัตถุประสงค์เพื่อควบคุมแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าให้คงที่ และยังเพิ่มขีดจำกัดเสถียรภาพสภาวะอยู่ตัวด้วย เพราะ AVR จะช่วยลดผลจากปฏิกิริยาอาร์เมเจอร์ (Armature reaction) เป็นผลให้ค่ารีแอกแตนซ์ของเครื่องกำเนิดไฟฟ้ามีค่าลดลงจากค่า x_d เหลือค่าประมาณ x_d' ซึ่งจะทำให้เครื่องกำเนิดไฟฟ้าสามารถจ่ายกำลังไฟฟ้าได้เต็มอัตราพิกัด

ในระบบควบคุมแบบเดิมจะมีการใช้ PSS สร้างสัญญาณควบคุมเสริมให้แก่ AVR ซึ่งจะช่วยเพิ่มขีดจำกัดเสถียรภาพ [38] แต่การใช้ PSS ควบคุมร่วมกับ AVR ไม่มีประสิทธิภาพมากนักหากระบบไฟฟ้ากำลังที่ควบคุมเป็นแบบชนิดสายส่งระยะไกล ด้วยเหตุนี้เอง การติดตั้ง SVC จะมีส่วนช่วยปรับปรุงการไหลของกำลังไฟฟ้าและปรับปรุงเสถียรภาพของระบบด้วย [39] แต่การติดตั้ง SVC แต่เพียงอย่างเดียวไม่สามารถเพิ่มเสถียรภาพของระบบได้ จึงต้องมีการสร้างสัญญาณควบคุมเสริมเพิ่มเข้ากับการควบคุมแบบเดิม ซึ่งผลลัพธ์ที่ได้จะเห็นได้จากผลการจำลองในบทที่ 8

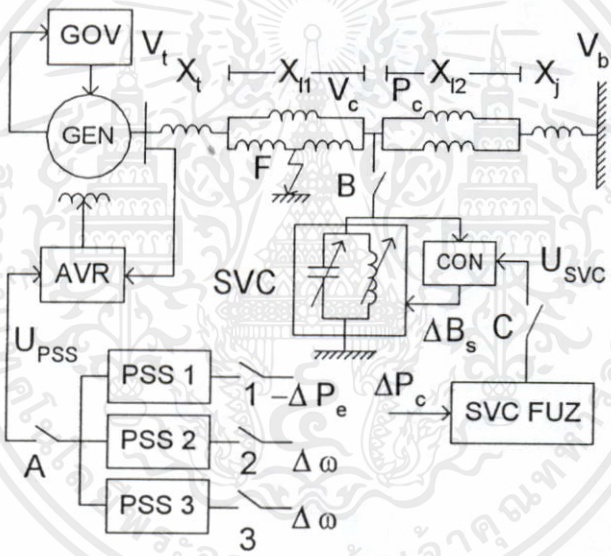
แบบจำลองระบบที่ใช้ในการวิเคราะห์เป็นระบบบัสอนันต์ ที่มีเครื่องกำเนิดไฟฟ้า 1 ตัว โดยบล๊อกไดอะแกรมของตัวควบคุมต่างๆ เป็นบล๊อกมาตรฐานจากระบบจริงที่มีการใช้อยู่ในปัจจุบัน วิธีการใหม่ในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง จะเป็นการสร้างสัญญาณเสริมเข้ากับระบบควบคุมเดิม และทดสอบผลโดยการจำลองบนคอมพิวเตอร์ ซึ่งจะมีการเปรียบเทียบระบบควบคุมด้วย PSS แบบเดิม กับระบบควบคุมด้วย SVC แบบใหม่

เนื้อหาในบทนี้ จะเป็นการอธิบายการสร้างสมการแบบจำลองเครื่องกำเนิดไฟฟ้าซิงโครน นัส และสมการของบล๊อกไดอะแกรมต่างๆ เพื่อที่จะนำมาสร้างแบบจำลองทั้งหมดของระบบ และนำไปใช้ในการสร้างเป็นโปรแกรมเพื่อทดสอบวิธีการควบคุมแบบต่างๆต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.1 แบบจำลองระบบ

แบบจำลองระบบที่ใช้ในการวิเคราะห์ ประกอบด้วยเครื่องกำเนิดไฟฟ้าซิงโครนัส 1 ตัว ต่อกับบัสอนันต์ (Infinite bus) ผ่านสายส่งวงจรคู่ ที่เครื่องกำเนิดไฟฟ้าซิงโครนัสจะมีอุปกรณ์ควบคุมความเร็วของเครื่องกำเนิดไฟฟ้า คือ GOV และมี AVR เป็นเครื่องควบคุมแรงดันอัตโนมัติ โดยมี SVC (Static Var Compensator) ติดตั้งอยู่จุดกึ่งกลางสายส่ง SVC จะได้รับสัญญาณควบคุมจากสองแหล่งคือ แรงดัน V_c และสัญญาณควบคุมเสริม U_{SVC} ซึ่งรับมาจากตัวควบคุมแบบฟัซซี่โดยมีสวิทช์ C เป็นตัวปลดหรือต่อสัญญาณ U_{SVC} กับ SVC สวิทช์ B เป็นตัวปลดหรือต่อ SVC เข้ากับระบบสายส่ง นอกจากนี้ที่ AVR ยังมีสัญญาณควบคุมเสริม U_{PSS} ที่รับมาจาก PSS โดยมีสวิทช์ A เป็นตัวปลดหรือต่อสัญญาณ U_{PSS} PSS ที่ใช้ในการวิเคราะห์มี 3 ชนิดด้วยกัน คือ ΔP-PSS , $\Delta\omega$-PSS , $\Delta\omega$-PSS ชนิดฟัซซี่ โดยมีสวิทช์ 1 , 2 , 3 เป็นตัวเลือกชนิดของ PSS ที่จะนำมาควบคุมร่วมกับ AVR แบบจำลองระบบทั้งหมดดังกล่าวแสดงในรูปที่ 5



รูปที่ 5.1 ระบบจำลอง

5.2 สมการเครื่องกำเนิดไฟฟ้าซิงโครนัส

ในหัวข้อนี้จะกล่าวถึงสมการคณิตศาสตร์ต่างๆที่จะสร้างเป็นแบบจำลองของเครื่องกำเนิดไฟฟ้าซิงโครนัส โดยจะพิจารณาผลกระทบจากปัจจัยต่างๆทั้งภายในและภายนอก เช่น ฟลักซ์ของขดลวดฟิลด์ , ค่าความเสียดทานต่างๆ เป็นต้น แต่เพื่อลดความซับซ้อนของแบบจำลองคณิตศาสตร์เครื่องกำเนิดไฟฟ้าซิงโครนัส จึงจำเป็นที่จะต้องตัดผลจากปัจจัยบางตัวที่มีผลกระทบน้อยมากต่อความละเอียดแม่นยำของแบบจำลอง เช่น ความต้านทานในขดลวดอาร์เมเจอร์ เป็นต้น ดังนั้นจึงเชื่อ

ได้ว่าแบบจำลองที่สร้างขึ้นนี้มีความน่าเชื่อถือสูงและสามารถนำมาอธิบายปรากฏการณ์ต่างๆของเครื่องกำเนิดไฟฟ้าซิงโครนัสได้เป็นอย่างดี

สมการต่างๆของแบบจำลองคณิตศาสตร์เครื่องกำเนิดไฟฟ้าซิงโครนัสที่จะทำการวิเคราะห์ได้แก่ สมการกระแสและแรงดันในแกน d-q , สมการฟลักซ์ขดลวดฟิลด์เครื่องกำเนิดไฟฟ้าซิงโครนัส , สมการสวิง , สมการกำลังไฟฟ้าเอาท์พุทเครื่องกำเนิดไฟฟ้าซิงโครนัส และสมการแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัส ซึ่งสมการต่างๆเหล่านี้ได้ทำการวิเคราะห์ใหม่ โดยเลียนแบบวิธีจาก [41]

5.2.1 สมการกระแสและแรงดันในแกน d-q

สมการแรงดัน V_t ที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัสสามารถจัดให้อยู่ในรูปของสมการแรงดันในแกน d-q โดยการใช้การแปลงของ Park ถ้าไม่คิดผลของความต้านทาน R ในอาร์เมเจอร์ดิริเวทีฟของฟลักซ์เกี่ยวคล้อง $p\phi_d$, $p\phi_q$ และค่าคงตัวเวลาซับทรานส์เซียน สมการที่ (5.1)-สมการที่(5.6) จึงลดรูปเหลืออย่างง่าย ดังต่อไปนี้ [41]

$$V_d = -\omega\phi_q \tag{5.1}$$

$$V_q = \omega\phi_d \tag{5.2}$$

$$V_f = \frac{d\phi_f}{dt} + V_{fr} \tag{5.3}$$

เมื่อ

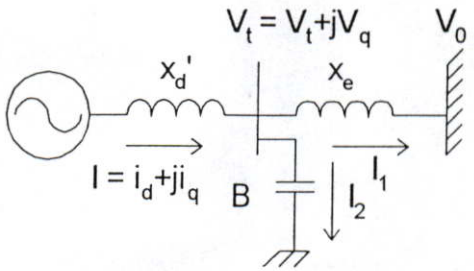
$$\omega\phi_q = -x_q i_q \tag{5.4}$$

$$\omega\phi_d = V_{fr} - x_d i_d \tag{5.5}$$

$$\phi_f = T_{d0}'\left[V_{fr} - \left(x_d - x_d'\right)i_d\right] \tag{5.6}$$

พิจารณารูปที่ 5.2 เป็นวงจรที่ลดเหลือรูปอย่างง่ายของระบบบัสอนันต์ โดยตัดตัวควบคุมออกจากระบบทั้งหมด ขั้นตอนการวิเคราะห์มีลักษณะคล้ายกับ [41]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.2 ระบบบัสอนันต์ที่ลดเหลือรูปอย่างง่าย

จากทิศทางกระแสที่กำกับในรูปที่ 5.2 สามารถเขียนสมการกระแสได้ดังต่อไปนี้

$$I = i_d + ji_q = I_1 + I_2 \tag{5.7}$$

$$i_d + ji_q = V_t(jB) + (V_t - V_0)jx_e \tag{5.8}$$

โดยที่

$$V_t = V_d + jV_q \tag{5.9}$$

$$V_0 = V_0 \sin\delta + jV_0 \cos\delta \tag{5.10}$$

แทนค่า V_t และ V_0 จากสมการที่ (5.9) และ (5.10) ลงในสมการที่ (5.8)

$$i_d + ji_q = (V_d + jV_q)(jB) + \frac{V_d + jV_q - V_0 \sin\delta - jV_0 \cos\delta}{jx_e} \tag{5.11}$$

$$i_d + ji_q = jV_dB - V_qB + \frac{-jV_d + V_q + jV_0 \sin\delta - V_0 \cos\delta}{x_e} \tag{5.12}$$

จากสมการที่ (5.12) เทียบสัมประสิทธิ์ของส่วนจริงจะได้

$$i_d = -V_qB + \frac{V_q - V_0 \cos\delta}{x_e} \tag{5.13}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$i_d = \frac{-V_q B + V_q - V_0 \cos\delta}{x_e} \tag{5.14}$$

$$i_d x_e = (1 - x_e B)V_q - V_0 \cos\delta \tag{5.15}$$

$$i_d = \frac{(1 - x_e B)V_q - V_0 \cos\delta}{x_e} \tag{5.16}$$

จากสมการที่ (5.2) และ (5.5)

$$V_q = V_{fr} - x_d i_d$$

จากสมการที่ (5.6) ทำการย้ายข้างใหม่เพื่อหา V_{fr} จะได้

$$V_{fr} = \frac{\phi_f}{T_{do}'} + (x_d - x_d')i_d \tag{5.18}$$

แทนค่า V_{fr} จากสมการที่ (5.18) ลงในสมการที่ (5.17)

$$V_q = \frac{\phi_f}{T_{do}'} - x_d' i_d \tag{5.19}$$

แทนค่า V_q จากสมการที่ (5.19) ลงในสมการที่ (5.16)

$$i_d = \frac{(1 - x_e B)\phi_f}{x_e T_{do}'} - \frac{(1 - x_e B)x_d' i_d}{x_e} - \frac{V_0 \cos\delta}{x_e} \tag{5.20}$$

$$i_d x_e = \frac{(1 - x_e B)\phi_f}{T_{do}'} - (1 - x_e B)x_d' i_d - V_0 \cos\delta \tag{5.21}$$

$$i_d = \frac{(1 - x_e B)\phi_f}{T_{do}'[(1 - x_e B)x_d' + x_e]} - \frac{V_0 \cos\delta}{[(1 - x_e B)x_d' + x_e]} \tag{5.22}$$

เมื่อกำหนดให้

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การผลิตซ้ำ ทั้งนี้ยังรวมถึงทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$X_d' = (1 - x_e B)x_d' + x_e \tag{5.23}$$

สมการที่ (5.22) สามารถเขียนใหม่ได้ดังนี้

$$i_d = \frac{(1 - x_e B)\phi_f}{X_d' T_{do}'} - \frac{V_0 \cos\delta}{X_d'} \tag{5.24}$$

จากสมการที่ (5.19)

$$V_q = \frac{\phi_f}{T_{do}'} - x_d' i_d$$

แทนค่า i_d จากสมการที่ (5.24) ลงในสมการที่ (5.19)

$$V_q = \frac{\phi_f}{T_{do}'} - \frac{(1 - x_e B)x_d'\phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \tag{5.25}$$

$$V_q = \frac{[X_d' - (1 - x_e B)x_d']\phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \tag{5.26}$$

แทนค่า X_d' จากสมการที่ (5.23) ลงในสมการ (5.26)

$$V_q = \frac{[(1 - x_e B)x_d' + x_e - (1 - x_e B)x_d']\phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \tag{5.27}$$

$$V_q = \frac{x_e \phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \tag{5.28}$$

เทียบสัมประสิทธิ์ส่วนจินตภาพจากสมการที่ (5.12) จะได้

$$i_q = V_d B + \frac{-V_d + V_0 \sin\delta}{x_e} \tag{5.29}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$i_q = i_q x_q B + \frac{(-i_q x_q + V_0 \sin\delta)}{x_e} \tag{5.30}$$

$$i_q x_q = i_q x_q x_e B - i_q x_q + V_0 \sin\delta \tag{5.31}$$

$$[(1 - x_e B)x_q + x_e]\, i_q = V_0 \sin\delta \tag{5.32}$$

$$i_q = \frac{V_0 \sin\delta}{(1 - x_e B)x_q + x_e} \tag{5.33}$$

เมื่อกำหนดให้

$$X_q = (1 - x_e B)x_q + x_e \tag{5.34}$$

แทนค่า X_q จากสมการที่ (5.34) ลงในสมการที่ (5.33) จะได้

$$i_q = \frac{V_0 \sin\delta}{X_q} \tag{5.35}$$

จากสมการที่ (5.1) และ (5.4)

$$V_d = x_q i_q \tag{5.36}$$

$$V_d = \frac{x_q V_0 \sin\delta}{X_q} \tag{5.37}$$

5.2.2 สมการฟลักซ์ขดลวดฟิลด์เครื่องกำเนิดไฟฟ้าซิงโครนัส

จากสมการที่ (5.18)

$$V_{fr} = \frac{\phi_f}{T_{do}'} + (x_d - x_d')i_d$$

แทนค่า i_d จากสมการที่ (5.24) ลงในสมการที่ (5.18)

$$V_{fr} = \frac{\phi_f}{T_{do}'} + \frac{(x_d - x_d')(1 - x_e B)\phi_f}{X_d' T_{do}'} - \frac{(x_d - x_d')V_0 \sin\delta}{X_d'} \tag{5.38}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แทนค่า $X_d{}'$ เฉพาะที่เศษของสมการที่ (5.38)

$$V_{fr} = \frac{[(1-x_eB)x_d{}' + x_e + (1-x_eB)x_d - (1-x_eB)x_d{}']\phi_f}{X_d{}'T_{do}{}'} - \frac{(x_d - x_d{}')V_0\cos\delta}{X_d{}'} \tag{5.39}$$

$$V_{fr} = \frac{[(1-x_eB)x_d + x_e]\phi_f}{X_d{}'T_{do}{}'} - \frac{(x_d - x_d{}')V_0\cos}{X_d{}'} \tag{5.40}$$

$$V_{fr} = \frac{X_d\phi_f}{X_d{}'T_{do}{}'} - \frac{(x_d - x_d{}')V_0\cos}{X_d{}'} \tag{5.41}$$

จากสมการที่ (5.3) ทำการย้ายข้างเพื่อจัดรูปใหม่จะได้

$$\frac{d\phi_f}{dt} = V_f - V_{fr} \tag{5.42}$$

แทนค่า V_{fr} จากสมการที่ (5.41) ในสมการที่ (5.42)

$$\frac{d\phi_f}{dt} = V_f - \frac{X_d\phi_f}{X_d{}'T_{do}{}'} - \frac{(x_d - x_d{}')V_0\cos\delta}{X_d{}'} \tag{5.43}$$

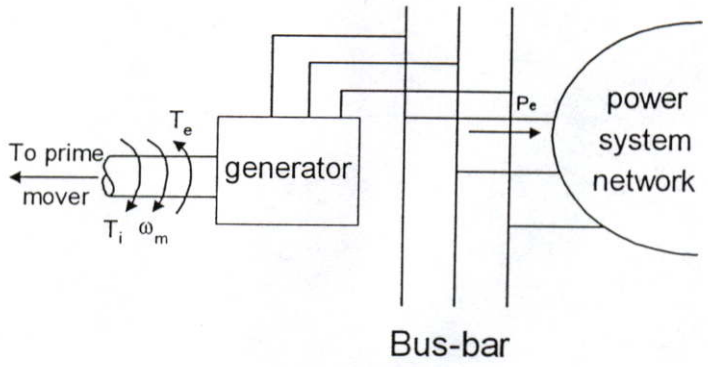
เมื่อ

$$X_d = (1-X_eB)x_d + x_e \tag{5.44}$$

5.2.3 สมการสวิง

พิจารณารูปที่ 5.3 เครื่องกำเนิดไฟฟ้าซิงโครนัสได้รับพลังงานกลอินพุท P_i จากเทอร์ไบน์ด้วยทอร์ก T_i และจ่ายพลังงานไฟฟ้าเอาท์พุท P_e โดยทำให้เกิดทอร์กทางแม่เหล็กไฟฟ้า T_e ซึ่งมีทิศทางสวนทางกับทอร์ก T_i

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกา
่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.3 เครื่องกำเนิดไฟฟ้าซิงโครนัสจ่ายกำลังไฟฟ้าสู่โครงข่ายไฟฟ้ากำลัง

เมื่อเกิดสิ่งรบกวน จะเกิดการการเร่งหรือหน่วงของโรเตอร์โดยขึ้นกับทอร์กเร่ง T_a ซึ่งกำหนดโดยสมการที่ (5.45)

$$T_a = T_i - T_e \tag{5.45}$$

$$T_a = I\frac{d^2\theta_m}{dt^2} + D\frac{d\theta_m}{dt} \tag{5.46}$$

$$T_a = I\frac{d\omega_m}{dt^2} + D\omega_m \tag{5.47}$$

$D\omega_m$ คือ ทอร์กหน่วง ซึ่งมีสาเหตุจาก ความเสียดทานที่แบริ่ง (Bearing) ของโรเตอร์ , ความเสียดทานจากแรงลม , ความสูญเสียทางแม่เหล็ก และทอร์กอื่นๆที่มีทิศทางสวนทางกับทิศทางการหมุนของโรเตอร์

สมการ (5.45) และ (5.47) เมื่อคูณด้วย ω_m ทั้งสองข้างจะได้

$$P_a = P_i - P_e \tag{5.48}$$

$$P_a = \omega_m I\frac{d\omega_m}{dt} + D\omega_m \tag{5.49}$$

$$P_a = M\frac{d\omega_m}{dt} + D\omega_m \tag{5.50}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อ M' มีหน่วยเป็น pu-sec^2/mech.radian มุม θ_m จะมีการเปลี่ยนแปลงตามเวลาอย่างต่อเนื่อง ทำให้สามารถเขียนสมการการขจัดเชิงมุม δ_m ซึ่งวัดจากแกนอ้างอิงที่หมุนไปด้วยความเร็วซิงโครนัส ω_{0m} (mech.rad/sec) ได้ดังสมการที่ (5.51)

$$\delta_m = \theta_m - \omega_{om}t \tag{5.51}$$

$$\frac{d\delta_m}{dt} = \frac{d\theta_m}{dt} - \omega_{om} = \omega_m - \omega_{om} \tag{5.52}$$

$$\frac{d\delta_m}{dt} = \omega_m - \omega_{om} = \Delta\omega_m \tag{5.53}$$

สมการที่ (5.52) จัดรูปใหม่จะได้

$$\frac{d\theta_m}{dt} = \frac{d\delta_m}{dt} + \omega_{om} \tag{5.54}$$

จะเป็นการสะดวกกว่าถ้าการวัดมุมต่างๆเป็นมุมทางไฟฟ้า ดังนั้นการแปลงมุมทางกลเป็นมุมทางไฟฟ้าสามารถทำได้โดยใช้สมการ

$$\delta = \frac{P}{2}\delta_m \text{ (electrical radian)} \tag{5.55}$$

$$\omega = \frac{P}{2}\omega_m \text{ (elecrical radian/s)} \tag{5.56}$$

เมื่อคูณ P/2 ทั้งสองข้างในสมการที่ (5.53) และ (5.54) จะได้

$$\frac{d\theta}{dt} = \omega_0 + \frac{d\delta}{dt} \tag{5.57}$$

$$\frac{d\delta}{dt} = \Delta\omega \tag{5.58}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\frac{d\theta}{dt} = \omega_0 + \Delta\omega \tag{5.59}$$

สมการที่ (5.50) เมื่อคูณทั้งสองข้างด้วย P/2 แล้วจัดรูปสมการใหม่ จะได้

$$M\frac{d^2\delta}{dt}+\frac{Dd\delta}{dt}=P_i-P_e \tag{5.60}$$

เมื่อ

$$M=\frac{M'}{P/2}\ (\text{pu.sec}^2/\text{elect.rad}) \tag{5.61}$$

สมการ (5.60) เขียนใหม่ได้เป็น

$$M\frac{d^2\delta}{dt}+D\Delta\omega=P_i-P_e \tag{5.62}$$

จากความสัมพันธ์

$$P_i-P_e=(P_0+\Delta P_i)-(P_0+\Delta P_e) \tag{5.63}$$

$$P_i-P_e=\Delta P_i-\Delta P_e \tag{5.64}$$

แทนค่า P_i-P_e จากสมการที่ (5.64) ลงในสมการที่ (5.62) แล้วจัดรูปใหม่จะได้

$$\frac{d\omega}{dt}=\frac{1}{M}(\Delta P_i-D\Delta\omega-\Delta P_e) \tag{5.65}$$

5.2.4 สมการกำลังไฟฟ้าเอาท์พุทเครื่องกำเนิดไฟฟ้าซิงโครนัส

กำลังไฟฟ้าเอาท์พุทของเครื่องกำเนิดไฟฟ้าซิงโครนัสสามารถเขียนได้ดังสมการที่ (5.66)

$$P_e=i_dV_d+i_qV_q \tag{5.66}$$

แทนค่า V_d, i_d, V_q, i_q จากสมการที่ (5.37) , (5.16) , (5.28) , (5.35) ลงในสมการที่ (5.66)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$P_e = \left(\frac{(1 - x_e B)\phi_f}{X_d' T_{do}'} - \frac{V_0 \cos\delta}{X_d'} \right) \frac{x_q V_0 \sin\delta}{X_q} + \frac{V_0 \sin\delta}{X_q} \left(\frac{x_e \phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \right) \tag{5.67}$$

$$P_e = \frac{(1 - x_e B) x_q \phi_f V_0 \sin\delta}{X_q X_d' T_{do}'} - \frac{x_q V_0^2 \sin\delta\cos\delta}{X_q X_d'} + \frac{x_e \phi_f V_0 \sin\delta}{X_q X_d' T_{do}'} + \frac{x_d' V_0^2 \sin\delta\cos\delta}{X_q X_d'} \tag{5.68}$$

ทำการจัดพจน์ในสมการ (5.68) ใหม่จะได้

$$P_e = \frac{V_0 \sin\delta\, \phi_f [(1 - x_e B) x_q + x_e]}{X_q X_d' T_{do}'} - \frac{V_0^2 \sin\delta\cos\delta (x_d' - x_q)}{X_q X_d'} \tag{5.69}$$

$$P_e = \frac{V_0 \sin\delta\, \phi_f}{X_d' T_{do}'} - \frac{2V_o^2 \sin\delta\cos\delta (x_d' - x_q)}{2 X_q X_d'} \tag{5.70}$$

สมการที่ (5.70) จัดให้เหลือในรูปอย่างง่ายจะได้

$$P_e = \frac{V_0 \sin\delta\, \phi_f}{X_d' T_{do}'} - \frac{(x_d' - x_q) V_0^2 \sin 2\delta}{2 X_q X_d'} \tag{5.71}$$

5.2.5 สมการแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัส

ความสัมพันธ์ของแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัสกับแรงดันในแกน d-q ของเครื่องกำเนิดไฟฟ้าสามารถเขียนได้ดังนี้

$$V_t = \sqrt{V_d^2 + V_q^2} \tag{5.72}$$

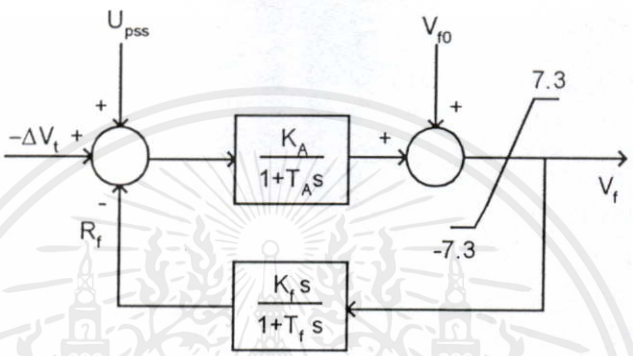
แทนค่า V_d, V_q จากสมการที่ (5.37) , (5.28) ลงในสมการที่ (5.72)

$$V_t = \sqrt{\left(\frac{x_q V_0 \sin\delta}{X_q} \right)^2 + \left(\frac{x_e \phi_f}{X_d' T_{do}'} + \frac{x_d' V_0 \sin\delta}{X_d'} \right)^2} \tag{5.73}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.3 แบบจำลอง AVR

บล๊อกไดอะแกรม AVR ในรูปที่ 5.1 สามารถแทนด้วยแบบด้วยจำลอง AVR ในรูปที่ 5.4 หน้าที่ของ AVR คือ ควบคุมขนาดของแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าซิงโครนัส จากแบบจำลองในรูปที่ 5.4 จะสังเกตได้ว่า Upss เป็นสัญญาณควบคุมเสริมที่ออกจาก PSS โดยจะรวมกับสัญญาณการเปลี่ยนแปลงแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้า ΔV_t วัตถุประสงค์ของสัญญาณควบคุมเสริม U_{PSS} คือ สร้างทอร์กหน่วงให้กับระบบโดยอาศัยการขยายสัญญาณผ่าน AVR เนื่องจากว่ามีค่าเกนของเอกไซเตอร์ที่สูง



รูปที่ 5.4 แบบจำลอง AVR

การวิเคราะห์สมการจากรูปที่ 5.4 สามารถทำได้โดยเขียนสมการในแต่ละบล๊อกดังนี้

$$V_f = \frac{K_A}{1+T_A s}(U_{PSS} - \Delta V_t - R_f) + V_{f0} \tag{5.74}$$

$$V_f + T_A s V_f = K_A(U_{PSS} - \Delta V_t - R_f) + V_{f0} \tag{5.75}$$

$$T_A\ sVf = K_A(U_{PSS} - \Delta V_t - R_f) - (V_f - V_{f0}) \tag{5.76}$$

$$sV_f = \frac{K_A}{T_A}(U_{PSS} - \Delta V_t - R_f) - \frac{(V_f - V_{f0})}{T_A} \tag{5.77}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.77) จะได้

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\frac{dV_f}{dt} = \frac{K_A}{T_A}(U_{PSS} - \Delta V_t - R_f) - \frac{(V_f - V_{f0})}{T_A} \tag{5.78}$$

เขียนสมการจากบล๊อกไดอะแกรมส่วนป้อนกลับรูปที่ 5.4 ได้ดังนี้

$$R_f = \frac{K_f s}{1 + T_f s} V_f \tag{5.79}$$

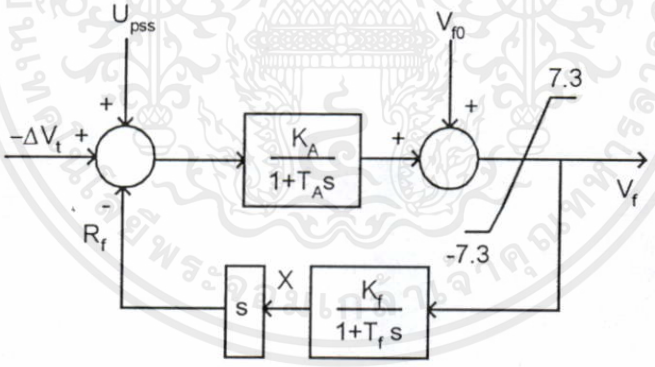
$$R_f + T_f s R_f = K_f s V_f \tag{5.80}$$

$$sR_f = \frac{K_f}{T_f} sV_f - \frac{R_f}{T_f} \tag{5.81}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.81) จะได้

$$\frac{dR_f}{dt} = \frac{K_f}{T_f}\frac{dV_f}{dt} - \frac{R_f}{T_f} \tag{5.82}$$

หรือทำการจัดรูปส่วนป้อนกลับใหม่ ดังรูปที่ 5.5 แล้วทำการหาสมการบล๊อกป้อนกลับใหม่อีกรอบ ซึ่งเป็นวิธีที่ใช้ในโปรแกรมที่เขียนขึ้นในบทที่ 7



รูปที่ 5.5 แบบจำลอง AVR ส่วนบล๊อกป้อนกลับที่มีการปรับปรุง

จากรูปที่ 5.5 ส่วนป้อนกลับสามารถเขียนเป็นสมการได้ดังนี้

$$X = \frac{K_f}{1 + T_f s} V_f \tag{5.83}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$X + T_f s X = K_f V_f \tag{5.84}$$

$$sX = \frac{K_f V_f - X}{T_f} \tag{5.85}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.85) จะได้

$$\frac{dX}{dt} = \frac{K_f V_f - X}{T_f} \tag{5.86}$$

จากส่วนป้อนกลับในรูปที่ 5.5 เขียนสมการต่อได้เป็น

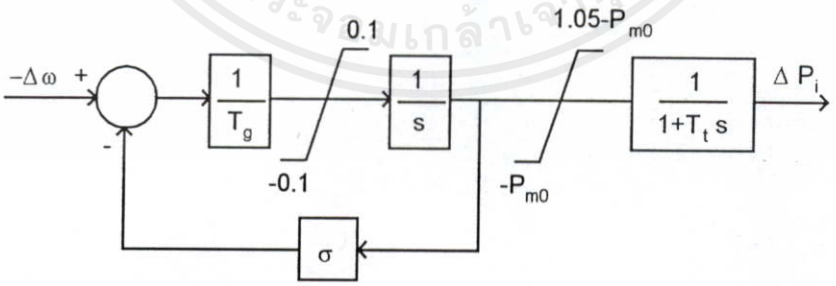
$$R_f = sX \tag{5.87}$$

$$R_f = \frac{K_f V_f - X}{T_f} \tag{5.88}$$

$$R_f = \frac{K_f ZZ - X}{T_f} \tag{5.89}$$

5.4 แบบจำลอง GOV

บล๊อกไดอะแกรม GOV ในรูปที่ 5.1 สามารถแทนด้วยแบบจำลอง GOV ในรูปที่ 5.6 วัตถุประสงค์ของการใช้ GOV คือ ใช้ปรับความเร็วของเครื่องกำเนิดไฟฟ้าซิงโครนันให้มีความเร็วคงที่เท่ากับความเร็วซิงโครนัส



รูปที่ 5.6 แบบจำลอง GOV

จากรูปที่ 5.6 สามารถเขียนสมการจากบล๊อกไดอะแกรมต่างๆได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$g = \frac{1}{T_g s}(-\Delta\omega - \sigma g) \tag{5.90}$$

$$sg = \frac{1}{T_g}(-\Delta\omega - \sigma g) \tag{5.91}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.91) จะได้

$$\frac{dg}{dt} = \frac{1}{T_g}(-\Delta\omega - \sigma g) \tag{5.92}$$

ส่วนที่เหลือของรูปที่ 5.6 เขียนเป็นสมการได้ดังนี้

$$\Delta P_i = \frac{g}{1 + T_t s} \tag{5.93}$$

$$\Delta P_i + T_t s\Delta P_i = g \tag{5.94}$$

$$s\Delta P_i = \frac{1}{T_t}(g - \Delta P_i) \tag{5.95}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.95) จะได้

$$\frac{d\Delta P_i}{dt} = \frac{1}{T_t}(g - \Delta P_i) \tag{5.96}$$

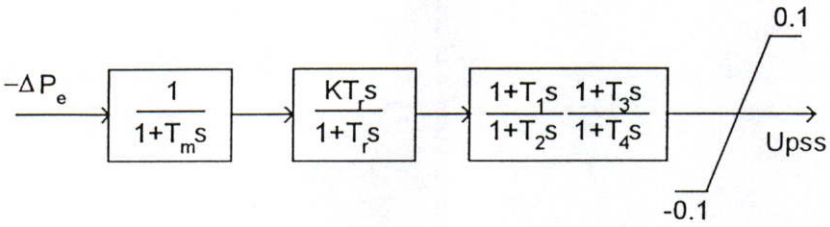
5.5 แบบจำลอง PSS

บล๊อกไดอะแกรม PSS ในรูปที่ 5.1 ซึ่งมีอยู่ 3 บล๊อก คือ PSS1 , PSS2 และ PSS3 สามารถแทนด้วยแบบจำลอง PSS 3 ชนิด คือ ΔP-PSS , $\Delta\omega$-PSS และ $\Delta\omega$-PSS ชนิดฟัซซี่ เรียงตามลำดับ ΔP-PSS และ $\Delta\omega$-PSS เป็น PSS แบบเกนคงที่ ซึ่งออกแบบด้วยวิธีการชดเชยแบบนำหน้าล้าหลัง (Lead-Lag Compensation) ส่วน $\Delta\omega$-PSS ชนิดฟัซซี่ ออกแบบตามวิธีในบทที่ 6

5.5.1 แบบจำลอง ΔP-PSS

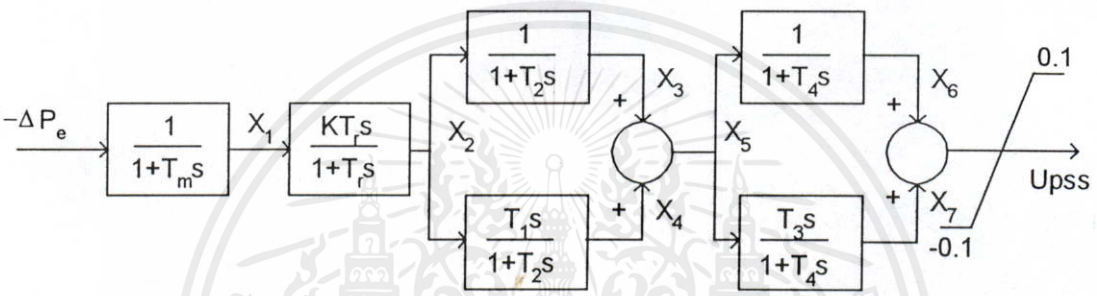
PSS1 สามารถแทนด้วยแบบจำลอง ΔP-PSS ในรูปที่ 5.7

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.7 แบบจำลอง ΔP-PSS

เพื่อให้เกิดความง่ายและสะดวกในการนำสมการที่เขียนขึ้นไปใช้งาน แบบจำลองในรูปที่ 5.7 จึงปรับปรุงใหม่ได้ดังรูปที่ 5.8



รูปที่ 5.8 แบบจำลอง ΔP-PSS ที่ทำการปรับปรุงใหม่

จากรูปที่ 5.8 สามารถเขียนเป็นสมการได้ดังนี้

$$X_1 = \frac{-\Delta P_e}{1+T_m s} \tag{5.97}$$

$$X_1 + T_m s X_1 = -\Delta P_e \tag{5.98}$$

$$sX_1 = \frac{-\Delta P_e - X_1}{T_m} \tag{5.99}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.99)

$$\frac{dX_1}{dt} = \frac{-\Delta P_e - X1}{T_m} \tag{5.100}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เขียนสมการตามรูปที่ 5.8 ในบล๊อกถัดไปจะได้

$$X_2 = \frac{KT_r s}{1 + T_r s} X_1 \tag{5.101}$$

$$X_2 + T_r s X_2 = KT_r s X_1 \tag{5.102}$$

$$sX_2 = \frac{KT_r s X_1 - X_2}{T_r} \tag{5.103}$$

แทนค่า sX1 จากสมการที่ (5.99) ในสมการที่ (5.103) จะได้

$$sX_2 = K\frac{(-\Delta P_e - X_1)}{T_m} - \frac{X_2}{T_r} \tag{5.104}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.104) จะได้

$$\frac{dX_2}{dt} = K\frac{(-\Delta P_e - X_1)}{T_m} - \frac{X_2}{T_r} \tag{5.105}$$

สมการในบล๊อกถัดไปในรูปที่ 5.8 เขียนได้ดังนี้

$$X_3 = \frac{X_2}{1 + T_2 s} \tag{5.106}$$

$$X_3 + T_2 s X_3 = X_2 \tag{5.107}$$

$$sX_3 = \frac{X_2 - X_3}{T_2} \tag{5.108}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.108) จะได้

$$\frac{dX_3}{dt} = \frac{X_2 - X_3}{T_2} \tag{5.109}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการ [illegible] เท่านั้น ไม่อนุญาตให้นำไปใช้ประ [illegible] ในกา
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมการในบล๊อกถัดไปในรูปที่ 5.8 เขียนได้ดังนี้

$$X_4 = \frac{T_1 s}{1 + T_2 s} X_3 \tag{5.110}$$

$$X_4 + T_2 sX_4 = T_1 sX_3 \tag{5.111}$$

$$sX_4 = \frac{T_1 sX_3}{T_2} - \frac{X_4}{T_2} \tag{5.112}$$

แทนค่า sX3 จากสมการที่ (5.108) ในสมการที่ (5.112)

$$sX_4 = \frac{T_1(X_2 - X_3)}{T_2 T_2} - \frac{X_4}{T_2} \tag{5.113}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.113)

$$\frac{dX_4}{dt} = \frac{T_1(X_2 - X_3)}{T_2 T_2} - \frac{X_4}{T_2} \tag{5.114}$$

สมการในบล๊อกถัดไปในรูปที่ 5.8 เขียนสมการได้ดังนี้

$$X_5 = X_3 + X_4 \tag{5.115}$$

$$X_6 = \frac{X_5}{1 + T_4 s} \tag{5.116}$$

$$X_6 + T_4 sX_6 = X_5 \tag{5.117}$$

$$sX_6 = \frac{X_5 - X_6}{T_4} \tag{5.118}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.118) จะได้

$$\frac{dX_6}{dt} = \frac{X_5 - X_6}{T_4} \tag{5.119}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมการในบล๊อกถัดไปในรูปที่ 5.8 เขียนได้ดังนี้

$$X_7 = \frac{T_3\, s}{1 + T_4\, s} X_5 \tag{5.120}$$

$$X_7 + T_4\, sX_7 = T_3\, sX_5 \tag{5.121}$$

$$sX_7 = \frac{T_3\, sX_5}{T_4} - \frac{X_7}{T_4} \tag{5.122}$$

สมการที่ (5.115) คูณด้วย s ทั้งสองข้าง จะได้ sX5 จากนั้นแทนลงในสมการที่ (5.122)

$$sX_7 = \frac{T_3}{T_4}\left[\frac{X_2 - X_3}{T_2} + \left[\frac{T_1(X_2 - X_3)}{T_2\, T_2} - \frac{X_4}{T_2}\right]\right] - \frac{X_7}{T_4} \tag{5.123}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.123) จะได้

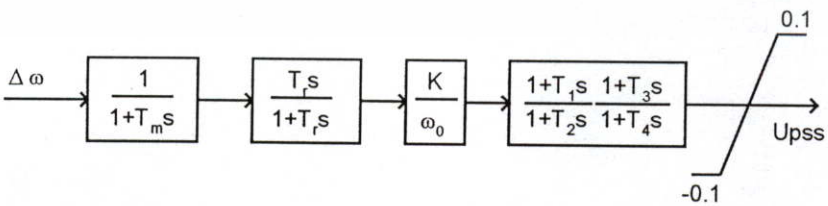
$$\frac{dX_7}{dt} = \frac{T_3}{T_4}\left[\frac{X_2 - X_3}{T_2} + \left[\frac{T_1(X_2 - X_3)}{T_2\, T_2} - \frac{X_4}{T_2}\right]\right] - \frac{X_7}{T_4} \tag{5.124}$$

จากรูปที่ 5.6 ผลรวมสุดท้ายของสัญญาณควบคุมเสริม U_{PSS} คือ

$$U_{PSS} = X_6 + X_7 \tag{5.125}$$

5.5.2 แบบจำลอง $\Delta\omega$-PSS

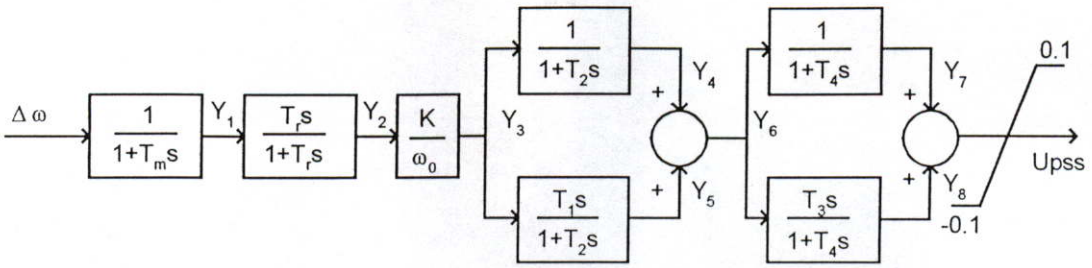
บล๊อกไดอะแกรม PSS2 ในรูปที่ 5.1 แทนด้วยแบบจำลอง $\Delta\omega$-PSS ดังแสดงในรูปที่ 5.9



รูปที่ 5.9 แบบจำลอง $\Delta\omega$-PSS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การแสวงหาผลประโยชน์ อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อความสะดวกและง่ายในการนำไปใช้งานแบบจำลองในรูปที่ 5.9 จึงปรับปรุงใหม่ได้ดังรูปที่ 5.10



รูปที่ 5.10 แบบจำลอง $\Delta\omega$-PSS ที่ทำการปรับปรุงใหม่

จากรูปที่ 5.10 เขียนเป็นสมการได้ดังนี้

$$Y_1 = \frac{\Delta\omega}{1 + T_m s} \tag{5.126}$$

$$Y_1 + T_m s Y_1 = \Delta\omega \tag{5.127}$$

$$sY_1 = \frac{\Delta\omega - Y_1}{T_m} \tag{5.128}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.128) จะได้

$$\frac{dY_1}{dt} = \frac{\Delta\omega - Y_1}{T_m} \tag{5.129}$$

สมการในบล๊อกถัดไปของรูปที่ 5.10 เขียนได้ดังนี้

$$Y_2 = \frac{T_r s}{1 + T_r s} Y_1 \tag{5.130}$$

$$Y_2 + T_r s Y_2 = T_r s Y_1 \tag{5.131}$$

$$sY_2 = \frac{T_r s Y_1 - Y_2}{T_r} \tag{5.132}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.132) จะได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\frac{dY_2}{dt} = \frac{T_r sY_1 - Y_2}{T_r} \tag{5.133}$$

สมการในบล๊อกถัดไปของรูปที่ 5.10 เขียนได้ดังนี้

$$Y_3 = \frac{KY_2}{\omega_0} \tag{5.134}$$

$$Y_4 = \frac{Y_3}{1 + T_2 s} \tag{5.135}$$

$$Y_4 + T_2 sY_4 = Y_3 \tag{5.136}$$

$$sY_4 = \frac{Y_3 - Y_4}{T_4} \tag{5.137}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.137) จะได้

$$\frac{dY_4}{dt} = \frac{Y_3 - Y_4}{T_4} \tag{5.138}$$

สมการในบล๊อกถัดไปของรูปที่ 5.10 เขียนได้ดังนี้

$$Y_5 = \frac{T_1 s}{1 + T_2 s} Y_3 \tag{5.139}$$

$$Y_5 + T_2 sY_5 = T_1 sY_3 \tag{5.140}$$

$$sY_5 = \frac{T_1 sY_3}{T_2} - \frac{Y_5}{T_2} \tag{5.141}$$

คูณสมการที่ (5.134) ด้วย s ทั้งสองข้าง จากนั้นแทนค่า sY3 ลงในสมการ (5.141)

$$sY_5 = \frac{T_1 K sY_2}{T_2 \omega_0} - \frac{Y_5}{T_2} \tag{5.142}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แทนค่า sY2 จากสมการที่ (5.132) ลงในสมการที่ (5.142)

$$sY_5 = \frac{T_1 K}{T_2 \omega_0}\left[\frac{T_1 sY_1 - Y_2}{T_r}\right] - \frac{Y_5}{T_2} \tag{5.143}$$

แทนค่า sY1 จากสมการที่ (5.128) ลงในสมการที่ (5.143)

$$sY_5 = \frac{T_1 K}{T_2 \omega_0}\left[\frac{T_1}{T_r}\frac{(\Delta\omega - Y_1)}{T_m} - \frac{Y_2}{T_r}\right] - \frac{Y_5}{T_2} \tag{5.144}$$

ทำการแปลงอินเวิอร์สลาปลาสสมการที่ (5.144) จะได้

$$\frac{dY_5}{dt} = \frac{T_1 K}{T_2 \omega_0}\left[\frac{T_1}{T_r}\frac{(\Delta\omega - Y_1)}{T_m} - \frac{Y_2}{T_r}\right] - \frac{Y_5}{T_2} \tag{5.145}$$

บล๊อกไดอะแกรมถัดไปในรูปที่ 5.10 เขียนได้ดังนี้

$$Y_6 = Y_4 + Y_5 \tag{5.146}$$

$$Y_7 = \frac{Y_6}{1 + T_4 s} \tag{5.147}$$

$$Y_7 + T_4 sY_7 = Y_6 \tag{5.148}$$

$$sY_7 = \frac{Y_6 - Y_7}{T_4} \tag{5.149}$$

ทำการแปลงอินเวิอร์สลาปลาสสมการที่ (5.149) จะได้

$$\frac{dY_7}{dt} = \frac{Y_6 - Y_7}{T_4} \tag{5.150}$$

สมการในบล๊อกถัดไปของรูปที่ 5.10 เขียนได้ดังนี้

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$Y_8 = \frac{T_3\, s}{1 + T_4\, s} Y_6 \tag{5.151}$$

$$Y_8 + T_4\, sY_8 = T_3\, sY_6 \tag{5.152}$$

$$sY_8 = \frac{T_3\, sY_6}{T_4} - \frac{Y_8}{T_4} \tag{5.153}$$

สมการที่ (5.146) คูณด้วย s ทั้งสองข้าง จากนั้นแทนค่า sY6 ลงในสมการที่ (5.153)

$$sY_8 = \frac{T_3}{T_4}\left[\frac{Y_3 - Y_4}{T_4} + \frac{T_1\, K}{T_2\, \omega_0}\left[\frac{T_1}{T_r}\frac{(\Delta\omega - Y_1)}{T_m} - \frac{Y_2}{T_r}\right] - \frac{Y_5}{T_2}\right] - \frac{Y_8}{T_4} \tag{5.154}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.154) จะได้

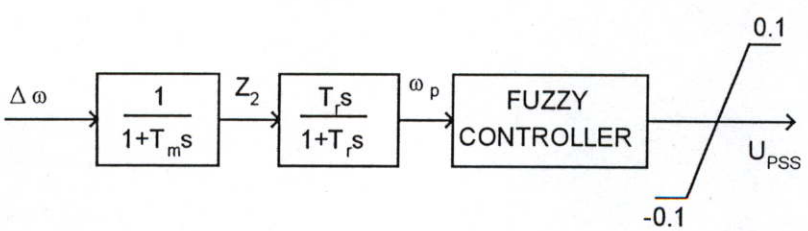
$$\frac{dY_8}{dt} = \frac{T_3}{T_4}\left[\frac{Y_3 - Y_4}{T_4} + \frac{T_1\, K}{T_2\, \omega_0}\left[\frac{T_1}{T_r}\frac{(\Delta\omega - Y_1)}{T_m} - \frac{Y_2}{T_r}\right] - \frac{Y_5}{T_2}\right] - \frac{Y_8}{T_4} \tag{5.155}$$

จากรูปที่ 5.10 ผลรวมสุดท้ายของสัญญาณควบคุมเสริม U_{PSS} คือ

$$U_{PSS} = Y_7 + Y_8 \tag{5.156}$$

5.5.3 แบบจำลอง $\Delta\omega$-PSS ชนิดฟัซซี่

บล๊อกไดอะแกรม PSS3 ในรูปที่ 5.1 สามารถแทนด้วยแบบจำลอง $\Delta\omega$-PSS ชนิดฟัซซี่ดังแสดงในรูปที่ 5.11



รูปที่ 5.11 แบบจำลอง $\Delta\omega$-PSS ชนิดฟัซซี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 5.11 สามารถเขียนเป็นสมการได้ดังนี้

$$Z_2 = \frac{\Delta\omega}{1 + T_m s} \tag{5.157}$$

$$Z_2 + T_m s Z_2 = \Delta\omega \tag{5.158}$$

$$sZ_2 = \frac{\Delta\omega - Z_2}{T_m} \tag{5.159}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.159)

$$\frac{dZ_2}{dt} = \frac{\Delta\omega - Z_2}{T_m} \tag{5.160}$$

บล๊อกไดอะแกรมถัดไปในรูปที่ 5.11 เขียนเป็นสมการได้ดังนี้

$$\omega_p = \frac{T_r s}{1 + T_r s} Z_2 \tag{5.161}$$

$$\omega_p + T_r s \omega_p = T_r s Z_2 \tag{5.162}$$

$$s\omega_p = \frac{T_r s Z_2 - \omega_p}{T_r} \tag{5.163}$$

แทนค่า sZ2 จากสมการที่ (5.159) ลงในสมการที่ (5.163)

$$s\omega_p = \frac{\Delta\omega - Z_2}{T_m} - \frac{\omega_p}{T_r} \tag{5.164}$$

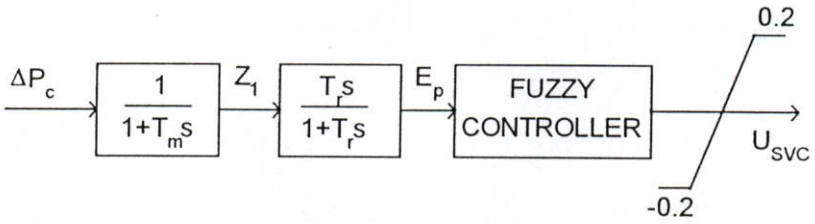
ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.164)

$$\frac{d\omega_p}{dt} = \frac{\Delta\omega - Z_2}{T_m} - \frac{\omega_p}{T_r} \tag{5.165}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.6 แบบจำลองตัวควบคุมฟัซซี่ของ SVC

บล๊อกไดอะแกรม SVC FUZ ในรูปที่ 5.1 สามารถแทนด้วยแบบจำลองตัวควบคุมฟัซซี่ของ SVC ดังในรูปที่ 5.12



รูปที่ 5.12 แบบจำลองตัวควบคุมฟัซซี่ของ SVC

จากรูปที่ 5.12 สามารถเขียนเป็นสมการได้ดังนี้

$$Z_1 = \frac{\Delta P_c}{1 + T_m s} \tag{5.166}$$

$$Z_1 + T_m s Z_1 = \Delta P_c \tag{5.167}$$

$$sZ_1 = \frac{\Delta P_c - Z_1}{T_m} \tag{5.168}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.168) จะได้

$$\frac{dZ_1}{dt} = \frac{\Delta P_c - Z_1}{T_m} \tag{5.169}$$

บล๊อกไดอะแกรมถัดไปในรูปที่ 5.12 สามารถเขียนเป็นสมการได้ดังนี้

$$E_p = \frac{T_r s}{1 + T_r s} Z_1 \tag{5.170}$$

$$E_p + T_r s E_p = T_r s Z_1 \tag{5.171}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$sE_p = \frac{T_r\, sZ_1 - E_p}{T_r} \tag{5.172}$$

แทนค่า sZ1 จากสมการที่ (5.168) ลงในสมการที่ (5.172)

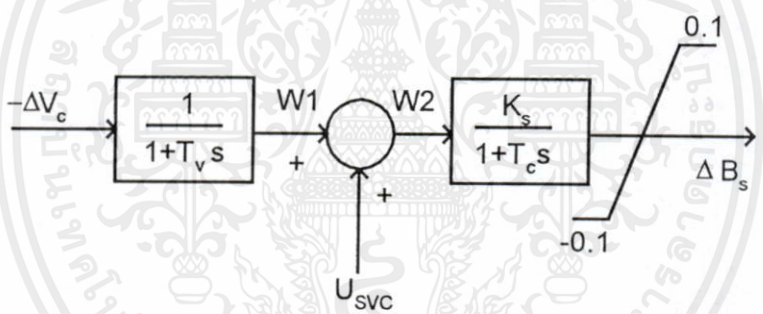
$$sE_p = \frac{\Delta P_c - Z_1}{T_m} - \frac{E_p}{T_r} \tag{5.173}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.173) จะได้

$$\frac{dE_p}{dt} = \frac{\Delta P_c - Z_1}{T_m} - \frac{E_p}{T_r} \tag{5.174}$$

5.7 แบบจำลอง SVC

บล๊อกไดอะแกรม SVC ในรูปที่ 5.1 สามารถแทนด้วยแบบจำลอง SVC ดังรูปที่ 5.13



รูปที่ 5.13 แบบจำลอง SVC

จากรูปที่ 5.13 สามารถเขียนสมการได้ดังนี้

$$W1 = \frac{-\Delta V_c}{1 + T_v s} \tag{5.175}$$

$$W1 + T_v sW1 = -\Delta V_c \tag{5.176}$$

$$sW1 = \frac{-\Delta V_c - W1}{T_v} \tag{5.177}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.177) จะได้

$$\frac{dW1}{dt} = \frac{-\Delta V_c - W1}{T_v} \tag{5.178}$$

บล๊อกไดอะแกรมถัดไปในรูปที่ 5.13 เขียนเป็นสมการได้ดังนี้

$$W2 = W1 + U_{SVC} \tag{5.179}$$

$$\Delta B_S = \frac{K_S W_2}{1 + T_c s} \tag{5.180}$$

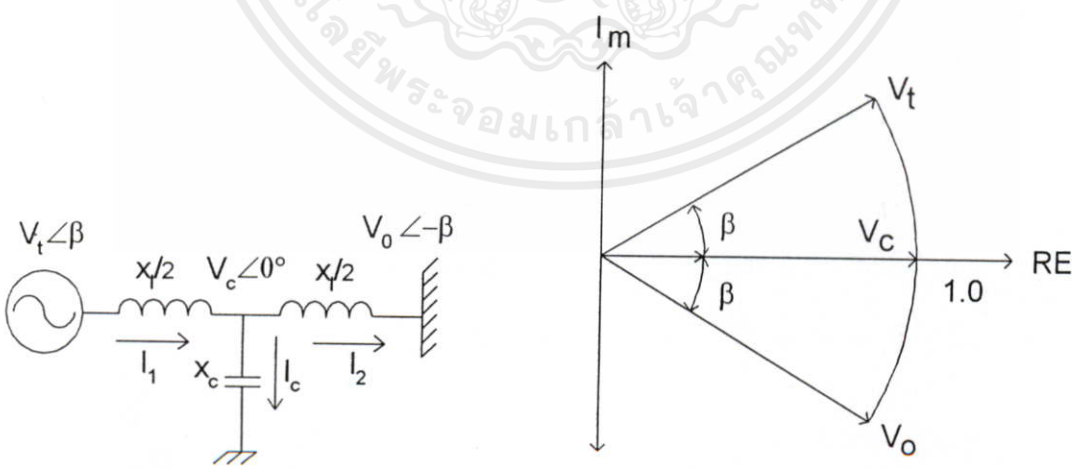
$$\Delta B_S + T_c s \Delta B_s = K_S (W1 + U_{SVC}) \tag{5.181}$$

$$s\Delta B_s = \frac{K_S (W1 + U_{SVC}) - \Delta B_S}{T_c} \tag{5.182}$$

ทำการแปลงอินเวิร์สลาปลาสสมการที่ (5.182) จะได้

$$\frac{d\Delta B_s}{dt} = \frac{K_S (W1 + U_{SVC}) - \Delta B_S}{T_c} \tag{5.183}$$

5.8 สมการการไหลของกำลังไฟฟ้าในสภาวะเริ่มต้นปกติ



(a) (b)

รูปที่ 5.14 วงจรสมมูลย์อย่างง่ายและเฟสเซอร์ไดอะแกรมสภาวะเริ่มต้นปกติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การแสวงหาผลกำไร ในทางการค้า ไม่ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในสภาวะเริ่มต้นปกติวงจรสมมูลย์และเฟสเซอร์ไดอะแกรมเป็นไปตามรูปที่ 5.14 ขนาดของแรงดันที่บัสต่างๆและกำลังไฟฟ้าจริงเริ่มต้นมีค่าดังนี้

V_t = 1.0 (pu) : ขนาดแรงดันที่ขั้วเครื่องกำเนิดไฟฟ้าเริ่มต้น

V_c = 1.0 (pu) : ขนาดแรงดันที่บัสติดตั้ง SVC เริ่มต้น

V_0 = 1.0 (pu) : ขนาดแรงดันที่บัสอนันต์เริ่มต้น

P_0 = 1.0 (pu) : ขนาดกำลังไฟฟ้าจริงเริ่มต้น

พิจารณาจากวงจรสมมูลย์อย่างง่ายในรูปที่ 5.14 (a) กระแส I1 มีค่าดังนี้

$$I_1 = \frac{V_t\angle\beta - V_c\angle 0^\circ}{j\,x_l/2} \tag{5.184}$$

$$I_1 = \frac{\cos\beta + j\sin\beta - 1.0}{j\,x_l/2} \tag{5.185}$$

$$I_1 = \frac{2}{x_l}\sin\beta - j\frac{2}{x_l}(\cos\beta - 1.0) \tag{5.186}$$

กระแส I2 มีค่าดังนี้

$$I_2 = \frac{V_c\angle\beta - V_0\angle 0^\circ}{j\,x_l/2} \tag{5.187}$$

$$I_2 = \frac{1.0 - \cos\beta + j\sin\beta}{j\,x_l/2} \tag{5.188}$$

$$I_2 = \frac{2}{x_l}\sin\beta - j\frac{2}{x_l}(1.0 - \cos\beta\ ) \tag{5.189}$$

พิจารณาทิศทางการไหลของกระแสในรูป 5.14 (a) จะได้

$$I_1 = I_2 + I_c \tag{5.190}$$

สมการกำลังไฟฟ้าสามารถเขียนได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$S_1 = P_1 + jQ_1 = V_t \bar{I}_1 \tag{5.191}$$

$$S_2 = P_2 + jQ_2 = V_c \bar{I}_2 \tag{5.192}$$

แทนค่ากระแสและแรงดัน สมการที่ (5.191) เขียนใหม่ได้ดังนี้

$$P_1 + jQ_1 = (\cos\beta \ + j\sin\beta \ )\frac{\cos\beta \ - j\sin\beta \ - 1.0}{-j\,x_l \ / \ 2} \tag{5.193}$$

$$P_1 + jQ_1 = \frac{2}{x_l}\sin\beta \ + j\frac{2(1.0-\cos\beta \ )}{x_l} \tag{5.194}$$

สมการที่ (5.194) เทียบสัมประสิทธิ์ส่วนจริงจะได้

$$P_1 = \frac{2}{x_l}\sin\beta \tag{5.195}$$

โดยที่

$$\beta \ = \ \sin^{-1}(\ \frac{P_1 x_l}{2}\ ) \tag{5.196}$$

เทียบสัมประสิทธิ์ส่วนจินตภาพสมการที่ (5.194) จะได้

$$Q_1 = \frac{2(1.0-\cos\beta \ )}{x_l} \tag{5.197}$$

กระแส I_c จากรูปที่ 5.14 (a) มีค่าดังนี้

$$I_c = I_1 - I_2 \tag{5.198}$$

$$I_c = \frac{4}{x_l}(\cos\beta \ - \ 1.0) \tag{5.199}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าพิจารณาที่ SVC โดยตรงกระแส I_c จะมีค่าเท่ากับ

$$I_c = \frac{V_c \angle 0\,^\circ}{-j\,x_c} = \frac{1.0}{-j\,x_c} \tag{5.200}$$

$$I_c = j\frac{1}{x_c} \tag{5.201}$$

เทียบสัมประสิทธิ์สมการที่ (5.199) กับ สมการที่ (5.201) จะได้

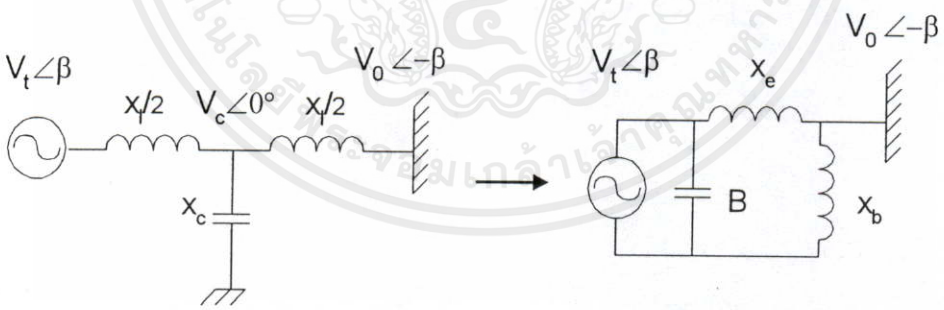
$$\frac{1}{x_c} = \frac{4}{x_l}(1.0 - \cos\beta\ ) \tag{5.202}$$

$$x_c = \frac{x_l}{4.0(1.0 - \cos\beta\ )} \tag{5.203}$$

$$B_{S0} = \frac{1}{x_c} \tag{5.204}$$

$$B_{S0} = \frac{4.0(1.0 - \cos\beta\ )}{x_l} \tag{5.205}$$

วงจรสมมูลย์อย่างง่ายในรูปที่ 5.14 สามารถแปลงจาก Y เป็น Δ ได้ดังรูปที่ 5.15



รูปที่ 5.15 การแปลงวงจรสมมูลย์อย่างง่ายจาก Y เป็น Δ

ค่ารีแอกแตนซ์ในวงจรสมมูลย์แบบ Δ หาได้จาก

$$x_e = \frac{-x_l{}^2/4 - j\,x_c(j\,x_l/2 + j\,x_l/2)}{-j\,x_c} \tag{5.206}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$x_e = j\frac{x_c x_l - x_l^2/4}{x_c} \tag{5.207}$$

$$x_b = \frac{-x_l^2/4 - j\,x_c(j\,x_l/2 + j\,x_l/2)}{j\,x_l} \tag{5.208}$$

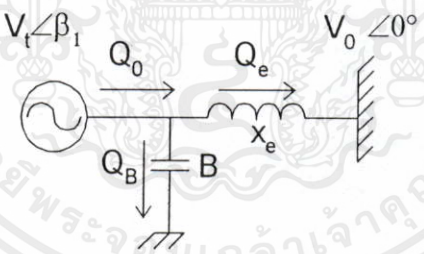
$$x_b = -j\frac{4x_c x_l - x_l^2}{2x_l} \tag{5.209}$$

$$-j\frac{1}{\omega C} = -j\frac{1}{B} \tag{5.210}$$

$$B = \omega C = -\frac{1}{x_b} \tag{5.211}$$

$$B = \frac{2x_l}{4x_c x_l - x_l^2} \tag{5.212}$$

การไหลของกำลังไฟฟ้ารีแอกทีฟของวงจรสมมูลย์อย่างง่าย แสดงได้ดังรูปที่ 5.16



รูปที่ 5.16 การไหลของกำลังไฟฟ้ารีแอกทีฟของวงจรสมมูลย์อย่างง่าย

จากรูปที่ 5.16 สามารถหา Q_0 ได้ดังนี้

$$Q_0 = Q_e + Q_B \tag{5.213}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำลังไฟฟ้าปรากฏมีค่าดังนี้

$$S_e = P_0 + jQ_0 = V_t\bar{I}_e \tag{5.214}$$

$$S_e = (\cos\beta_1 + j\sin\beta_1)\frac{\cos\beta_1 - j\sin\beta_1 - 1.0}{-jx_e} \tag{5.215}$$

$$S_e = \frac{1}{x_e}\sin\beta_1 + j\frac{(\cos\beta_1 - 1.0)}{x_e} \tag{5.216}$$

เทียบสัมประสิทธิ์ส่วนจริงสมการที่ (5.216) จะได้

$$P_0 = \frac{1}{x_e}\sin\beta_1 \tag{5.217}$$

เทียบสัมประสิทธิ์ส่วนจินตภาพสมการที่ (5.216) จะได้

$$Q_e = \frac{1}{x_e}(\cos\beta_1 - 1.0) \tag{5.218}$$

Q_B สามารถหาได้จาก

$$Q_B = \frac{1}{B} = \frac{1}{2x_1}(4x_1x_e - x_1^2) \tag{5.219}$$

ดังนั้น Q_0 หาได้จากผลรวมสมการที่ (5.218) และสมการที่ (5.219)

$$Q_0 = \frac{1}{x_e}(\cos\beta_1 - 1.0) + \frac{1}{2x_1}(4x_ex_1 - x_1^2) \tag{5.220}$$

โดยที่

$$\beta_1 = \sin^{-1} P_0x_e \tag{5.221}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

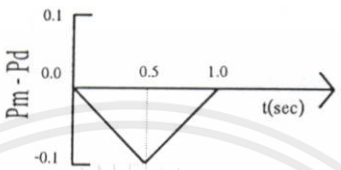
5.9 นิยามสภาวะการรบกวนและการวิเคราะห์โครงข่ายอิมพีแดนซ์

5.9.1 นิยามสภาวะการรบกวน

การรบกวนต่างๆที่ใช้ทดสอบนิยามไว้ดังต่อไปนี้

1. การรบกวนแบบรุนแรง คือ การเกิดฟอล์ทชนิด 3 เฟสลงดิน ในตำแหน่ง F ของระบบจำลองในรูปที่ 5.1 และฟอล์ทนี้ถูกขจัดหลังจาก 0.1 วินาทีผ่านไป

2. การรบกวนแบบเล็กน้อย คือ การเปลี่ยนแปลงของกำลังอินพุทเครื่องกำเนิดไฟฟ้า ดังแสดงในรูปที่ 5.17

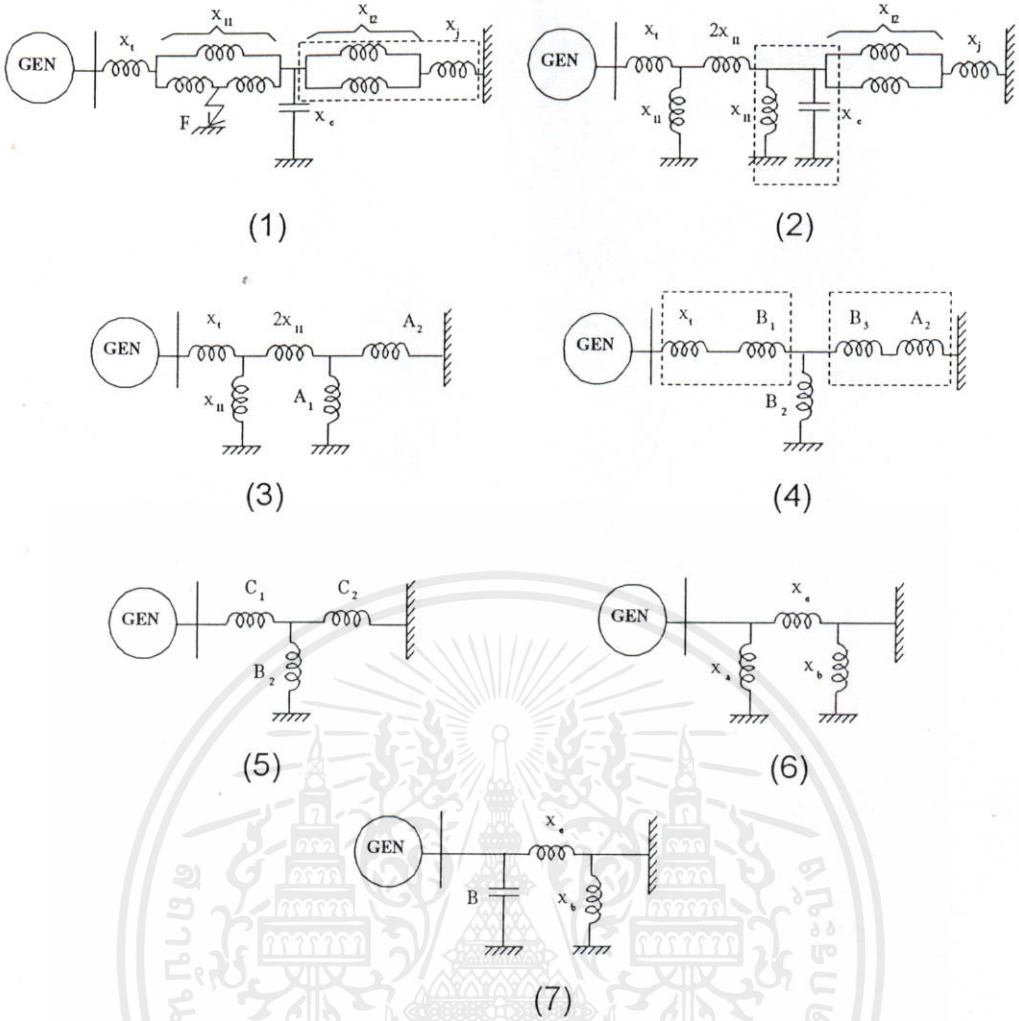


รูปที่ 5.17 การรบกวนแบบเล็กน้อย

5.9.2 การวิเคราะห์โครงข่ายอิมพีแดนซ์

จากการวิเคราะห์การไหลของกำลังไฟฟ้าเริ่มต้นสภาวะปกติในหัวข้อที่ 5.8 จะสังเกตได้ว่าวงจรสมมูลย์ที่ใช้ในการคำนวณจะลดรูปสุดท้ายในรูปแบบการต่อแบบ Δ เนื่องจากง่ายในการคำนวณ ในการจำลองจะมีการสมมุติให้เกิดการรบกวนแบบรุนแรงด้วยการลัดวงจรในตำแหน่ง F ของรูปที่ 5.1 ซึ่งหากพิจารณาให้ดีแล้ว การลัดวงจรจะทำให้วงจรสมมูลย์ Y มีค่าเปลี่ยนแปลงไปจากเดิม เพื่อให้การคำนวณการไหลของกำลังไฟฟ้า ยังสามารถใช้สมการที่ได้วิเคราะห์แล้วในหัวข้อ 5.8 ดังนั้นวงจรสมมูลย์สุดท้ายจึงทำการลดรูปสุดท้ายในลักษณะการต่อแบบ Δ การลดรูปวงจรสมมูลย์ดังกล่าวมีขั้นตอนแสดงในรูปที่ 5.18

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.18 ขั้นตอนการลดรูปวงจรสมมูลย์ของแบบจำลองเมื่อเกิดการลัดวงจร

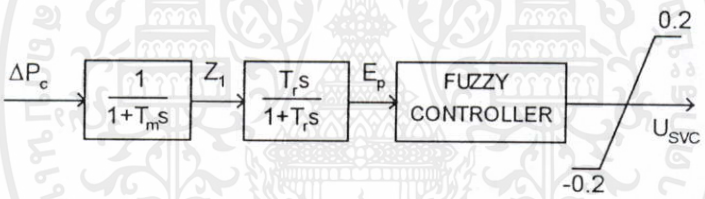
สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
การออกแบบตัวควบคุมฟัซซี่ชนิดเฟสเพลน

ปกติตัวควบคุมฟัซซี่ทั่วไปจะมีกฎการควบคุม IF-THEN มากมาย ซึ่งกฎเหล่านี้ต้องอาศัยผู้ชำนาญในการสร้างขึ้นมา [35,36,40] จึงจะทำให้ได้ผลตอบสนองระบบที่ดี เนื้อหาในบทนี้ จะเสนอการออกแบบตัวควบคุมฟัซซี่วิธีใหม่โดยใช้การแปลงข้อมูลที่ได้รับจากเฟสเพลนแทน ซึ่งจะพบว่า ตัวควบคุมฟัซซี่วิธีใหม่นี้ไม่มีความซับซ้อน จึงมีความเป็นไปได้ที่จะนำมาใช้ในการควบคุมแบบเวลาจริง

6.1 การแปลงความหมายข้อมูลจากเฟสเพลน

รูปที่ 6.1 แสดงตัวควบคุมฟัซซี่ของ SVC โดยสัญญาณ ΔP_C ณ ตำแหน่งที่ติดตั้ง SVC จะเป็นสัญญาณอินพุท สัญญาณนี้จะผ่านบล๊อกการวัดที่มีค่าคงตัวเวลา T_m และผ่านบล๊อกวงจรรีเซตที่มีค่าคงตัวเวลา T_r สัญญาณที่จะป้อนเข้าสู่ตัวควบคุมฟัซซี่และนำไปใช้คำนวณต่อไป คือ Ep



รูปที่ 6.1 ตัวควบคุมฟัซซี่ของ SVC

ภายในบล๊อกตัวควบคุมฟัซซี่ จะต้องคำนวณหา $E^{\bullet}_p$ โดยเป็นไปตามสมการที่ (6.1)

$$E_P^{\bullet}(k) = \frac{E_P(k) - E_P(k-1)}{T} \qquad (6.1)$$

k = 1,2,3.....,N เมื่อ T คือเวลาสุ่ม , N คือจำนวนข้อมูล

$E^{\bullet}_p$ ที่คำนวณได้จะถูกคูณกับแฟกเตอร์ปรับสเกล $\alpha 1$ เป็น $\alpha 1\, E^{\bullet}_p$ โดย E_p และ $\alpha 1\, E^{\bullet}_p$ จะเป็นตัวกำหนดตำแหน่งของจุด P(k) บนเฟสเพลนในรูปที่ 6.2

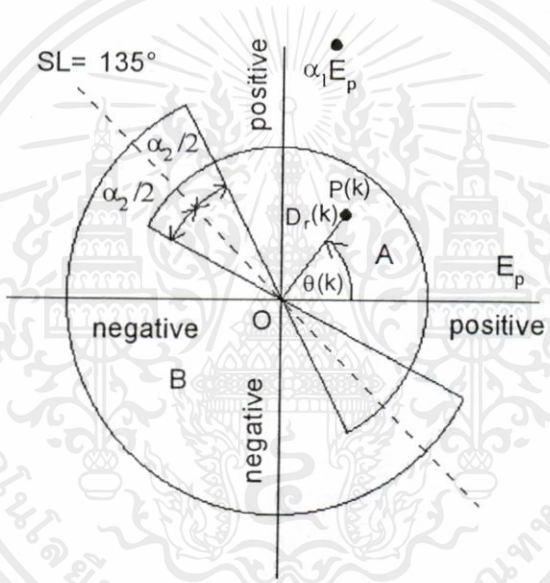
ในรูปที่ 6.2 จุด O เป็นจุดที่ระบบสมดุล ดังนั้นการทำงานของการควบคุมวิธีนี้จะพยายามเลื่อนตำแหน่งจุด P(k) ให้เข้าสู่จุดศูนย์กลาง O ให้เร็วที่สุดเท่าที่จะทำได้ ข้อมูลต่างๆที่แปลงความหมายจากเฟสเพลนแสดงไว้ในสมการที่ (6.2) , (6.3) และ (6.4)

$$P(k) = D_r(k)\angle\theta(k) \tag{6.2}$$

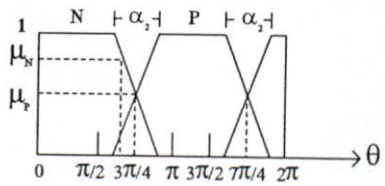
$$D_r(k) = \sqrt{Ep(k)^2 + (\alpha_1 \dot{E}_P(k))^2} \tag{6.3}$$

$$\theta(k) = \tan^{-1}(\alpha_1 \dot{E}_P(k) / E_P(k)) \tag{6.4}$$

พื้นที่บนเฟสเพลนถูกออกเป็น 2 ส่วนคือส่วนA และB โดยที่พื้นที่ส่วน A เป็นส่วนที่ SVC จ่ายกำลังไฟฟ้ารีแอกทีฟ พื้นที่ส่วน B เป็นส่วนที่ SVC รับกำลังไฟฟ้ารีแอกทีฟ α_2 เป็นมุมทับซ้อนของพื้นที่ทั้งสองส่วน SL คือสวิทชิ่งลายน์(Switching Line)ซึ่งทำมุมทวนเข็มนาฬิกา 135° พื้นที่ทั้งสองนี้ถูกกำหนดโดยฟังก์ชั่นการเป็นสมาชิกของ θ คือ $\mu_N(\theta)$, $\mu_P(\theta)$ ดังแสดงในรูปที่ 6.3



รูปที่ 6.2 เฟสเพลน

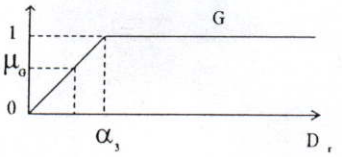


รูปที่ 6.3 ค่าฟังก์ชั่นการเป็นสมาชิกของ θ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นอกจากนี้ D_r ยังเป็นตัวบ่งบอกถึงระดับความรุนแรงของของรบกวน หมายความว่า ถ้า D_r ห่างจากจุด O มาก การรบกวนก็จะรุนแรง การที่จะเลื่อนจุด P(k) ที่เลื่อนออกไปกลับมายังจุด O จึงต้องใช้

สัญญาณควบคุมเสริมที่มีขนาดความแรงมากขึ้นด้วย แต่ถ้าจุด P(k) เลื่อนไปจากจุด O ไม่มากนัก ระดับความแรงของสัญญาณควบคุมเสริมก็จะลดลง ความแรงของสัญญาณควบคุมเสริมนี้ถูกกำหนดโดยค่าฟังก์ชั่นการเป็นสมาชิกของ D_r ในรูปที่ 6.4 โดยมี $\alpha 3$ เป็นพารามิเตอร์ควบคุมฟัซซี่



รูปที่ 6.4 ค่าฟังก์ชั่นการเป็นสมาชิกของ D_r

6.2 การคำนวณสัญญาณควบคุม

จากหัวข้อ 6.1 จะพบว่าสัญญาณควบคุมเสริมที่สร้างขึ้นจะขึ้นอยู่กับ (1) ตำแหน่งของจุด P (k) บนเฟสเพลนว่าอยู่ในพื้นที่ส่วนใด ซึ่งจะเป็นตัวบ่งบอกว่าสัญญาณควบคุมเสริมที่สร้างขึ้นจะควบคุมการจ่ายหรือรับกำลังไฟฟ้ารีแอกทีฟ (2) ระยะทางของจุด P(k) กับจุด O ซึ่งจะเป็นตัวบ่งบอกความแรงสัญญาณที่สร้างขึ้นว่า การจ่ายหรือรับกำลังไฟฟ้ารีแอกทีฟจะมีปริมาณมากน้อยเพียงใด

สัญญาณควบคุมเสริม U_{SVC} ดังกล่าว สามารถเขียนเป็นสมการได้ดังสมการที่ (6.5) ซึ่งได้จากการนำค่า $\mu_N(\theta)$ และ $\mu_G(D_r)$ จากรูปที่ 6.3 และ 6.4 มาเขียนในลักษณะการหาจุดศูนย์ถ่วง

$$U_{SVC}(k) = \frac{\mu_N(k) - \mu_P(k)}{\mu_N(k) + \mu_P(k)} \mu_G(k) U_{max}$$

$$= (2\mu_N(k) - 1)\mu_G(k) U_{max} \qquad (6.5)$$

6.3 การปรับค่าพารามิเตอร์ควบคุมฟัซซี่

พารามิเตอร์ควบคุมฟัซซี่ในที่นี้มี 3 ตัว คือ $\alpha 1$, $\alpha 2$ และ $\alpha 3$ การปรับแต่งพารามิเตอร์ควบคุมนี้จะอาศัยสมการดัชนีทำงาน (Performacne index : J) ดังสมการที่ (6.6) โดยผลตอบสนองที่ดีจะให้ค่าดัชนีทำงาน J ต่ำที่สุด

$$J = \sum_{i=1}^{2} (\beta - 1)i + (2 - \beta)\left(\sum_{k=1}^{N} \left(|\Delta\omega(k)| t_k + |\dot{U}(k)| \right) \right) \qquad (6.6)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

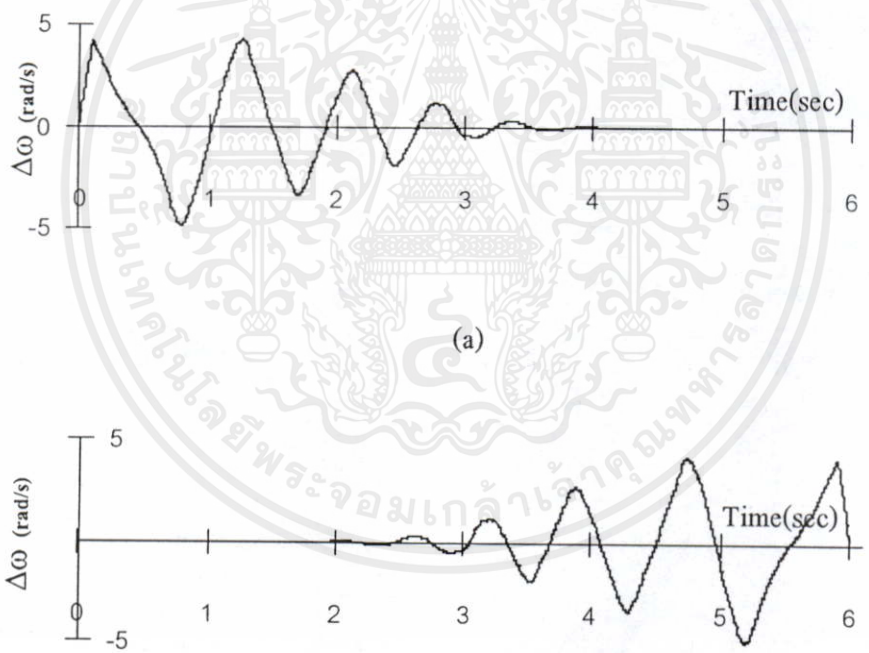
โดยที่ i = 1 คือ การรบกวนแบบรุนแรง

i = 2 คือ การรบกวนแบบเล็กน้อย

β คือ ค่าคงที่ช่วง [0,1]

$$\dot{U}(k) = \frac{U_{SVC}(k) - U_{SVC}(k-1)}{T}$$

ดัชนีทำงาน J จะ เป็นตัวเลขสำหรับใช้ในการพิจารณาตัดสินใจเลือกผลตอบสนองที่ดีที่สุด โดยปัญหานี้พิจารณาเสถียรภาพจากการเปลี่ยนแปลงความเร็วเครื่องกำเนิดไฟฟ้าซิงโครนัส $\Delta\omega$ โดยจะนำค่า $\Delta\omega$ คูณกับเวลาขณะนั้น ซึ่งเป็นการให้น้ำหนักของเวลาไม่เท่ากันโดยจะให้น้ำหนักช่วงแรกต่ำกว่าช่วงหลัง เพื่อเลือกผลตอบสนองที่มีการแกว่งไกวช่วงแรกและเข้าสู่สภาวะอยู่ตัวในตอนท้าย ในรูปที่ 6.5 แสดงให้เห็นถึงผลตอบสนองที่มีพื้นที่ใต้กราฟเท่ากัน ซึ่งในกรณีแรก(รูปที่ 9.5(a)) จะให้ค่าดัชนีทำงานต่ำกว่ากรณีที่สอง (รูปที่ 9.5(b)) จากตัวอย่างนี้จะพบว่า การใช้สมการดัชนีทำงานนี้ สามารถเลือกผลตอบสนองที่ถูกต้องตามต้องการได้



(a)

(b)

รูปที่ 6.5 เปรียบเทียบผลตอบสนอง 2 กรณีที่มีพื้นที่ใต้กราฟเท่ากัน

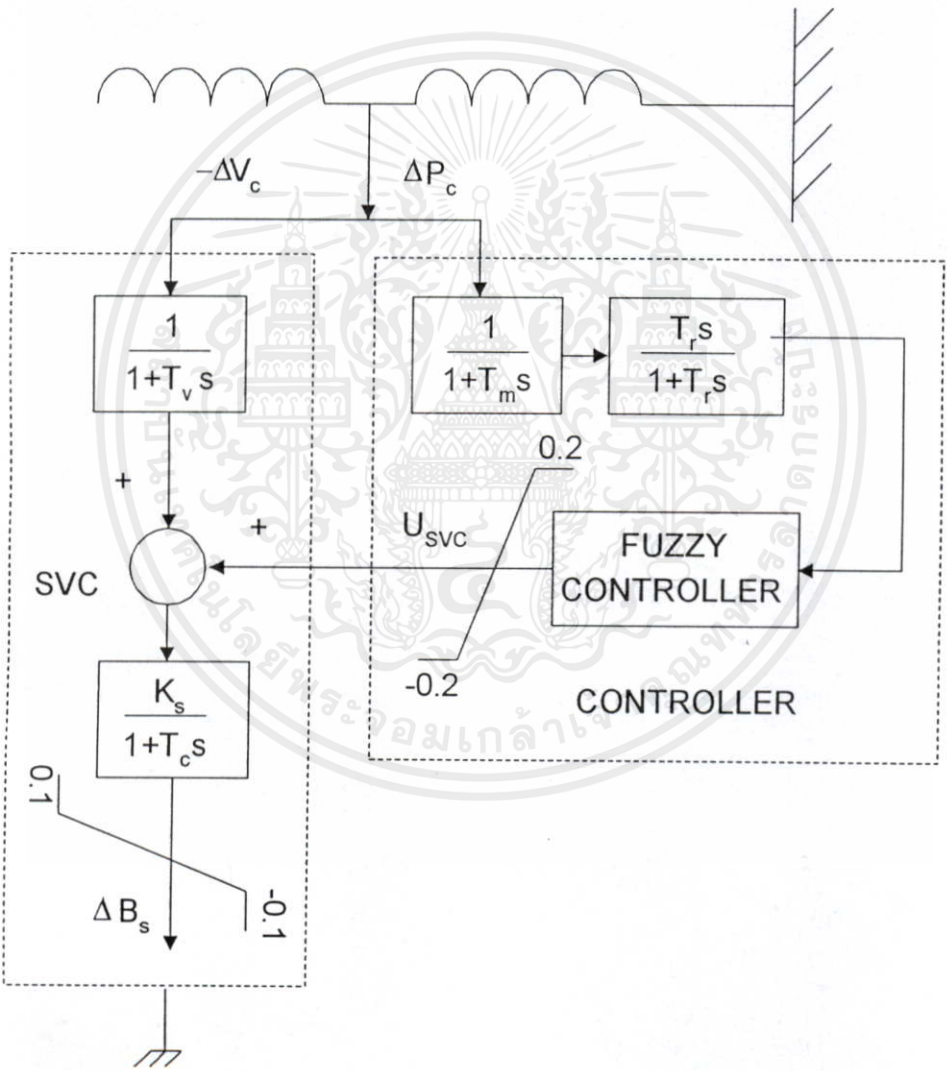
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

จากสมการที่ (6.6) ดัชนีทำงาน J จะมีการคิดถ่วงน้ำหนักของการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย โดยกำหนดที่ค่า β เพื่อให้ตัวควบคุมสามารถควบคุมได้ทั้งสองสภาวะ

ค่าพารามิเตอร์ควบคุมฟัซซี่ที่ต้องการ คือ พารามิเตอร์ที่ให้ค่าดัชนีทำงาน J ต่ำที่สุด ซึ่งจะให้ผลตอบสนองที่ดีที่สุดนั่นเอง

6.4 การเชื่อมต่อ SVC กับตัวควบคุมฟัซซี่

SVC ที่ใช้ในวิทยานิพนธ์นี้เป็นชนิด TSC/TCR ซึ่งมีผลตอบสนองการควบคุมที่เร็ว (ประมาณ 0.01 วินาที) [5] การเชื่อมต่อ SVC กับตัวควบคุมฟัซซี่แสดงในรูปที่ 6.6 อินพุทของ SVC คือ แรงดันตำแหน่งที่ติดตั้ง SVC และอินพุทของตัวควบคุมฟัซซี่คือ กำลังไฟฟ้า ที่ตำแหน่งเดียวกัน



รูปที่ 6.6 บล๊อกไดอะแกรมการเชื่อมต่อ SVC เข้ากับตัวควบคุมฟัซซี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.5 ความสัมพันธ์ของ P,Vt,Vc,ω เมื่อควบคุมด้วย SVC

จากสมการสวิง

$$\frac{d\omega}{dt} = \frac{1}{M}(\Delta P_i - D\Delta\omega - \Delta P_e)$$

ปริมาณที่ควบคุมการเร่งหรือหน่วงของเครื่องกำเนิดไฟฟ้าซิงโครนัส คือ ΔP_i , $D\Delta\omega$ และ ΔP_e เนื่องจากผลตอบสนองการควบคุมของ GOV ช้ามาก ΔP_i จึงไม่สามารถควบคุมการเร่งหรือหน่วงได้ทัน $D\Delta\omega$ มีทิศทางต้านการเปลี่ยนแปลงของความเร็วเครื่องกำเนิดไฟฟ้าซิงโครนัส แต่เนื่องจาก D มีค่าน้อยมาก ปริมาณ $D\Delta\omega$ นี้จึงมีผลน้อย ดังนั้นการควบคุม ΔP_e จึงมีผลต่อการเร่งหรือหน่วงมากที่สุด ทั้งการควบคุมปริมาณและความเร็วในการตอบสนอง

จากการวิเคราะห์แบบจำลองระบบในบทที่ 5 จะได้

$$P_e = V_d i_d + V_q i_q$$

$$P_e = V_d(\delta, B_s)\, i_d(\delta, \phi f, B_s) + V_q(\delta, \phi f, B_s)\, i_q(\delta, B_s)$$

$$Q_e = V_q\, i_d - V_d i_q$$

$$Q_e = V_q(\delta, \phi f, B_s)\, i_d(\delta, \phi f, B_s) - V_d(\delta, B_s)\, i_q(\delta, B_s)$$

$$V_t = \sqrt{V_d{}^2 + V_q{}^2}$$

$$V_c{}^2 = (V_d + (x_t + x_{l1})\, i_q)^2 + (V_q - (x_t + x_{l1})\, i_d)^2$$

เมื่อทำสมการข้างบนเป็นเชิงเส้นจะได้

$$\Delta P_e = V_{d0}\Delta\, i_d + i_{d0}\Delta V_d + V_{q0}\Delta\, i_q + i_{q0}\Delta V_q \qquad (6.7)$$

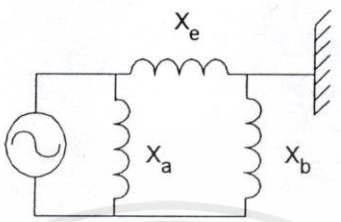
$$\Delta Q_e = V_{q0}\Delta\, i_d + i_{d0}\Delta V_q - V_{d0}\Delta\, i_q - i_{q0}\Delta V_d \qquad (6.8)$$

$$\Delta V_t = \frac{V_{d0}}{V_{t0}}\Delta V_d + \frac{V_{d0}}{V_{t0}}\Delta V_q \qquad (6.9)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\Delta V_{c0} = \frac{V_{d0} + (x_t + x_{l1})\, i_{q0}}{V_{c0}} \Delta V_d + (x_t + x_{l1}) \Delta\, i_q + \frac{V_{q0} + (x_t + x_{l1})\, i_{d0}}{V_{c0}} \Delta V_q + (x_t + x_{l1}) \Delta\, i_d \tag{6.10}$$

ΔV_d , ΔV_q , Δi_d ,$\Delta\, i_q$ เมื่อติดตั้ง SVC คำนวณจากวงจรสมมูลย์อย่างง่ายในรูปที่ 6.7 โดยรายละเอียดในการคำนวณดูได้ในบทที่ 5



รูปที่ 6.7 วงจรสมมูลย์อย่างง่ายสำหรับการคำนวณ ΔV_d , ΔV_q , Δi_d , Δi_q

จากผลการคำนวณในบทที่ 5 จะได้

$$V_d = \frac{x_q V_0 \sin\delta}{X_q} \tag{6.11}$$

โดยที่

$$X_q = (1 - x_e x_a) x_q + x_e \tag{6.12}$$

$$x_e = \frac{4 x_c x_l - x_l^2}{4 x_c} \tag{6.13}$$

$$x_a = \frac{2 x_l}{4 x_c x_l - x_l^2} \tag{6.14}$$

ค่า X_C ของ SVC คำนวณได้จาก

$$x_c = x_{c0} + \Delta x_c = \frac{1}{B_{s0} + \Delta B_s} = \frac{1}{B_s} \tag{6.15}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แทนค่า x_e ,x_a จากสมการ (6.13) และ (6.14) ลงในสมการ (6.12) จะได้

$$X_q = x_q - \frac{x_q x_l}{2} B_s + x_l - \frac{x_l^2}{4} B_s$$

$$X_q = x_q + x_l - B_s(\frac{x_q x_l}{2} + \frac{x_l^2}{4})$$

$$X_q = A_1 - A_2 B_s \tag{6.16}$$

แทนค่า X_q จากสมการ (6.16) ในสมการ (6.11)

$$V_d = \frac{x_q V_0 \sin\delta}{A_1 - A_2 B_s} \tag{6.17}$$

เมื่อทำสมการที่ (6.17) เป็นเชิงเส้นจะได้

$$\Delta V_d = \frac{x_q V_0 \cos\delta_0}{A_1 - A_2 B_{s0}} \Delta\delta + \frac{A_2 x_q V_0 \sin\delta_0}{A_1 - A_2 B_{s0}} \Delta B_s \tag{6.18}$$

โดยที่

$$A_1 = x_q + x_l \tag{6.19}$$

$$A_2 = \frac{x_l^2}{4} + \frac{x_q x_l}{2} \tag{6.20}$$

สมการ i_d จากบทที่ 5 เขียนได้ดังนี้

$$i_d = \frac{\phi f (1 - x_e x_a)}{X_d' T_{d0}'} - \frac{V_0 \cos\delta}{X_d'} \tag{6.21}$$

โดยที่

$$X_d' = (1 - x_e x_a) x_d' + x_e \tag{6.22}$$

แทนค่า x_e , x_a จากสมการ (6.13) และ (6.14)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$X_d' = x_d' - \frac{x_d' x_l}{2} B_s + x_l - \frac{x_l^2}{4} B_s \tag{6.23}$$

$$X_d' = x_d' + x_l - B_s\left(\frac{x_d' x_l}{2} + \frac{x_l^2}{4}\right) \tag{6.24}$$

$$X_d' = A_3 - A_4 B_s \tag{6.25}$$

แทนค่าจากสมการ (6.25) ในสมการ (6.21) จะได้

$$i_d = \frac{\phi f (1 - x_e x_a)}{(A_3 - A_4 B_s) T_{d0}'} - \frac{V_0 \cos\delta}{(A_3 - A_4 B_s)} \tag{6.26}$$

เมื่อทำสมการ (6.26) เป็นเชิงเส้นจะได้

$$\Delta i_d = \frac{V_0 \sin\delta_0}{X_d'} \Delta\delta + \frac{1 - x_e x_a}{X_d' T_{d0}'} \Delta\phi f + \left(\frac{-x_l / 2\phi_{f0}}{X_d' T_{d0}'} + \frac{(1 - x_e x_a) A_4 \phi_{f0}}{X_d'^2 T_{d0}'} - \frac{A_4 V_0 \cos\delta_0}{X_d'^2}\right) \Delta B_s \tag{6.27}$$

สมการ i_q และ V_q คือ

$$i_q = \frac{V_0 \sin\delta}{X_q} \tag{6.28}$$

$$V_q = \frac{x_e \phi_f}{X_d' T_{d0}'} + \frac{x_d' V_0 \cos\delta}{X_d'} \tag{6.29}$$

สมการที่ (6.28) และ (6.29) เมื่อทำเป็นเชิงเส้นจะได้

$$\Delta i_q = \frac{V_0 \cos\delta_0}{X_q} \Delta\delta + \frac{A_4 V_0 \cos\delta_0}{X_q^2} \Delta B_s \tag{6.30}$$

$$\Delta V_q = \frac{-x_d' V_0 \sin\delta_0}{X_d'} \Delta\delta + \frac{x_e}{X_d' T_{d0}'} \Delta\phi f + \left(\frac{-x_l^2 / 4\phi_{f0}}{X_d' T_{d0}'} + \frac{x_e A_4 \phi_{f0}}{X_d'^2 T_{d0}'} - \frac{A_4 x_d' V_0 \cos\delta_0}{X_d'^2}\right) \Delta B_s \tag{6.31}$$

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ ดัดแปลง เผยแพร่ หรือแสวงหาประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลลัพธ์ที่ได้จากสมการข้างบนสามารถสรุปได้ดังนี้

$$\Delta P_e = V_{d0}\Delta\, i_d + i_{d0}\Delta V_d + V_{q0}\Delta\, i_q + i_{q0}\Delta V_q$$

$$\Delta Q_e = V_{q0}\Delta\, i_d + i_{d0}\Delta V_q - V_{d0}\Delta\, i_q - i_{q0}\Delta V_d$$

$$\Delta V_t = \frac{V_{d0}}{V_{t0}}\Delta V_d + \frac{V_{d0}}{V_{t0}}\Delta V_q$$

$$\Delta V_{c0} = \frac{V_{d0} + (x_t + x_{l1})\, i_{q0}}{V_{c0}}\Delta V_d + (x_t + x_{l1})\Delta\, i_q + \frac{V_{q0} + (x_t + x_{l1})\, i_{d0}}{V_{c0}}\Delta V_q + (x_t + x_{l1})\Delta\, i_d$$

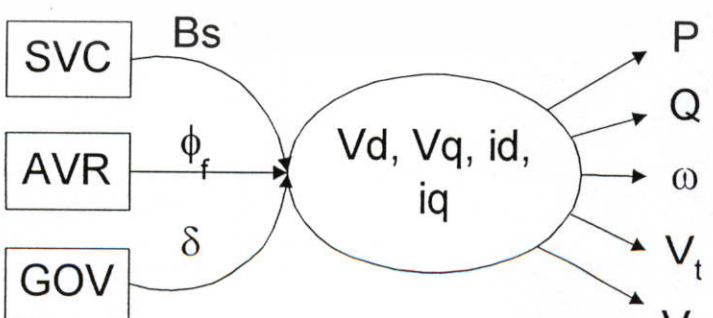
$$\Delta V_d = \frac{x_q V_0 \cos\delta_0}{A_1 - A_2 B_{s0}}\Delta\delta + \frac{A_2 x_q V_0 \sin\delta_0}{A_1 - A_2 B_{s0}}\Delta B_s$$

$$\Delta\, i_d = \frac{V_0 \sin\delta_0}{X_d'}\Delta\delta + \frac{1 - x_e x_a}{X_d' T_{d0}'}\Delta\phi f + \left(\frac{-x_l / 2\phi_{f0}}{X_d' T_{d0}'} + \frac{(1 - x_e x_a) A_4 \phi_{f0}}{X_d'^2 T_{d0}'} - \frac{A_4 V_0 \cos\delta_0}{X_d'^2}\right)\Delta B_s$$

$$\Delta\, i_q = \frac{V_0 \cos\delta_0}{X_q}\Delta\delta + \frac{A_4 V_0 \cos\delta_0}{X_q^2}\Delta B_s$$

$$\Delta\, V_q = \frac{-x_d' V_0 \sin\delta_0}{X_d'}\Delta\delta + \frac{x_e}{X_d' T_{d0}'}\Delta\phi f + \left(\frac{-x_l^2 / 4\phi_{f0}}{X_d' T_{d0}'} + \frac{x_e A_4 \phi_{f0}}{X_d'^2 T_{d0}'} - \frac{A_4 x_d' V_0 \cos\delta_0}{X_d'^2}\right)\Delta B_s$$

ความสัมพันธ์ของ P,Q,ω,Vt และ Vc ผ่าน V_d V_q i_d i_q โดยการปรับ B_S ,ϕ_f ,δ จาก ตัวควบคุม SVC , AVR และ GOV แสดงไว้ในรูปที่ 6.8



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 6.8 ความสัมพันธ์ P,Q,ω,Vt และ Vc จากการปรับตัวควบคุม

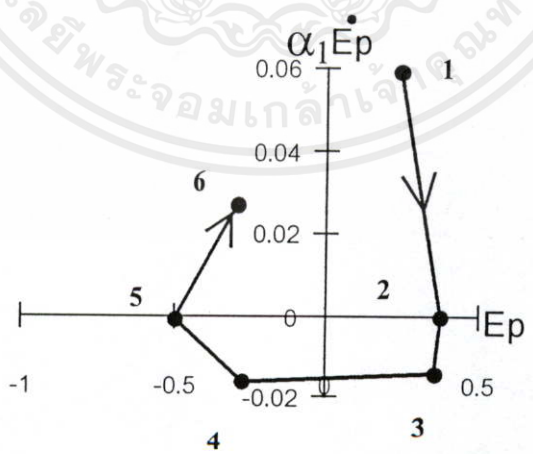
6.6 การปรับการควบคุม SVC โดยใช้ข้อมูลจากเฟสเพลน

ΔP_C เป็นสัญญาณอินพุทเข้าบล๊อกไดอะแกรมตัวควบคุมฟัซซี่ของ SVC โดยจะผ่านบล๊อกวงจรการวัดและบล๊อกวงจรรีเซต และได้สัญญาณ E_P เป็นอินพุทสำหรับบล๊อกตัวควบคุมฟัซซี่ เนื่องจากสัญญาณ $\Delta P_C = \Delta P_e$ และเกือบมีเฟสเดียวกับ E_P ดังนั้นการอธิบายการปรับการควบคุม SVC และผลกระทบจากการปรับ SVC จะอาศัยสัญญาณ ΔP_C เป็นหลัก

กระบวนการปรับการควบคุมและผลกระทบจากการปรับ SVC แสดงในตารางที่ 6.1 โดยตำแหน่งที่ 1-6 ในตารางที่ 6.1 มีความสัมพันธ์กับตำแหน่งในรูปที่ 6.9 ซึ่งข้อมูลจากตารางที่ 6.1 นี้จะเป็นแนวทางในการออกแบบกฎการควบคุมฟัซซี่ ดังอธิบายรายละเอียดไว้ในภาคผนวก ข.

ตารางที่ 6.1 การปรับการควบคุมและผลกระทบจากการปรับของ SVC

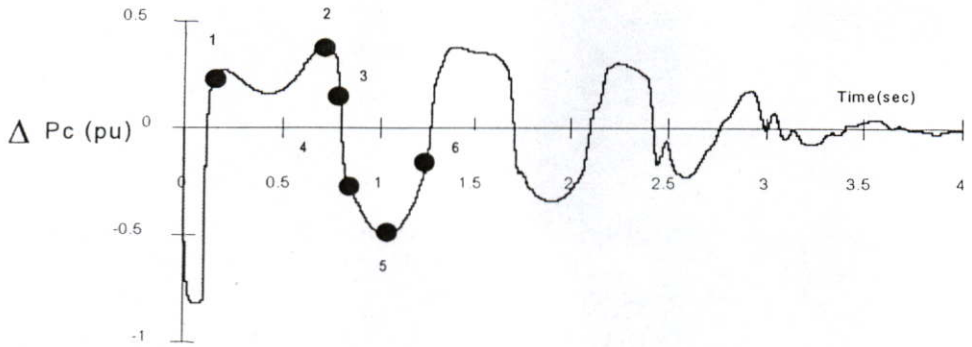
ตำแหน่ง	เงื่อนไข	การปรับ SVC	ผลกระทบ
1	$\Delta P_e > 0$, $\Delta \dot{P}_e > 0$	จ่าย Q	หน่วง ω , Vc สูงขึ้น
2	$\Delta P_e = \max$, $\Delta \dot{P}_e = 0$	จ่าย Q	หน่วง ω , Vc สูงขึ้น
3	$\Delta P_e > 0$, $\Delta \dot{P}_e < 0$	จ่าย Q ลดลง	หน่วง ω , Vc สูงขึ้น
4	$\Delta P_e < 0$, $\Delta \dot{P}_e < 0$	รับ Q	เร่ง ω , Vc ต่ำลง
5	$\Delta P_e = -\max$, $\Delta \dot{P}_e = 0$	รับ Q	เร่ง ω , Vc ต่ำลง
6	$\Delta P_e < 0$, $\Delta \dot{P}_e > 0$	รับ Q ลดลง	เร่ง ω , Vc ต่ำลง



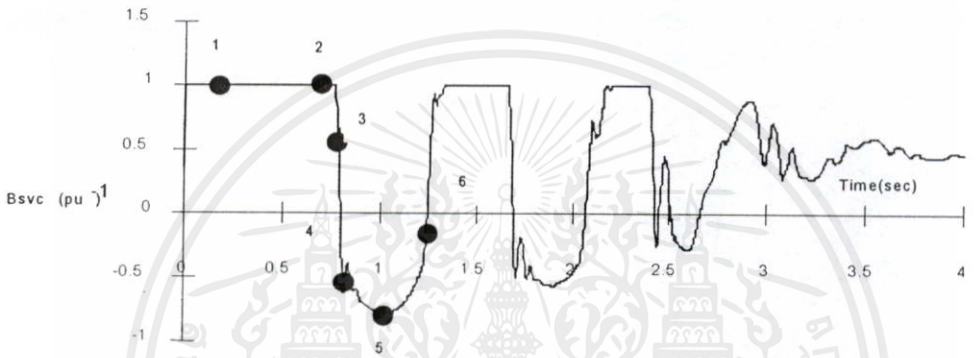
(a)

รูปที่ 6.9 ตัวอย่างการปรับการควบคุมของ SVC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การขาย การจำหน่าย การแสวงหาผลประโยชน์ทางการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(b)



(c)

รูปที่ 6.9 (ต่อ) ตัวอย่างการปรับการควบคุมของ SVC

6.7 ผลกระทบต่อปริมาณอื่นๆจากการปรับการควบคุมของ SVC

ตัวอย่างนี้จะทำการตั้งค่า $\alpha 1 = 0.012$, $\alpha 2 = 90^{\circ}$ และ $\alpha 3 = 0.24$ โดยใช้การคำนวณสัญญาณควบคุม U_{SVC} ตามหัวข้อ 6.2 และกระบวนการในหัวข้อ 6.6 ผลของการปรับ B_{SVC} ต่อปริมาณอื่นที่สำคัญโดยพิจารณาแนวโน้มการเปลี่ยนแปลงเป็นหลัก แสดงในตารางที่ 6.2 ซึ่งสัมพันธ์กับการวิเคราะห์สมการในหัวข้อ 6.5 สัญญาณของปริมาณต่างๆแสดงไว้ในรูปที่ 6.10

ตารางที่ 6.2 แนวโน้มของปริมาณอื่นๆที่สำคัญเมื่อปรับ B_{SVC}

B_{SVC}	V_d	V_q	i_q	i_q	P_e	V_C
เพิ่ม ↑	เพิ่ม ↑	ลด ↓	เพิ่ม ↑	เพิ่ม ↑	เพิ่ม ↑	เพิ่ม ↑
ลด ↓	ลด ↓	เพิ่ม ↑	ลด ↓	ลด ↓	ลด ↓	ลด ↓

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กราฟสัญญาณต่างๆในรูปที่ 6.10 อธิบายได้ดังต่อไปนี้

1. ช่วง ΔP_e เป็นบวก เป็นช่วงที่ปรับ SVC เป็นโหลดคาปาซิทีฟ ทำให้ x_e มีค่าต่ำลงและทำให้เกิดการส่งถ่ายพลังงานออกจากตัวเครื่องกำเนิดไฟฟ้ามากขึ้น ซึ่งจะเป็นการหน่วงความเร็ว ω ไม่ให้มีการเปลี่ยนแปลงเพิ่มขึ้น ในช่วงนี้แรงดัน V_t จะมีค่าสูงขึ้นและมีทิศทางเพิ่มขึ้นตาม V_d

2. ในช่วง ΔP_e เป็นลบ เป็นช่วงที่ปรับ SVC เป็นโหลดอินดักทีฟ ทำให้ x_e มีค่าสูงขึ้นและทำให้เกิดการส่งถ่ายพลังงานออกจากตัวเครื่องกำเนิดไฟฟ้าลดลง ซึ่งจะเป็นการเร่งความเร็ว ω ให้มีค่าสูงขึ้น ในช่วงนี้แรงดัน V_t จะมีค่าต่ำลงแต่แรงดันจะไม่ตกลงตาม P_e มากนัก เนื่องจากแรงดัน V_q ในช่วงนี้มีการเปลี่ยนแปลงในทิศทางเพิ่มขึ้น

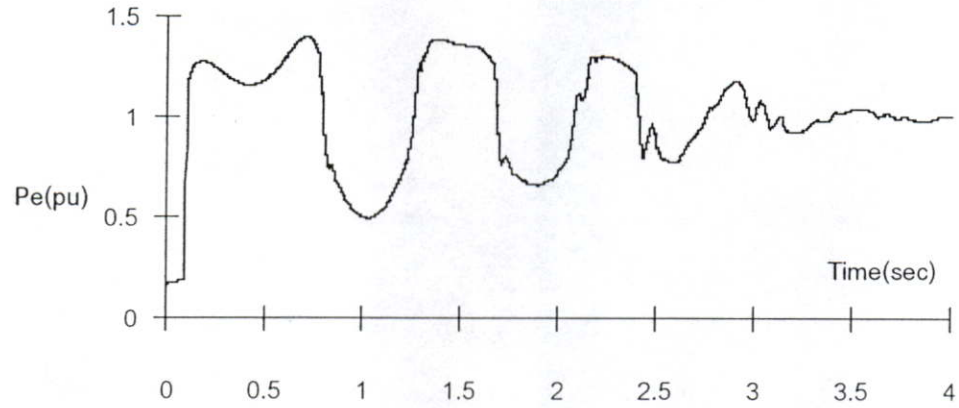
3. ปริมาณกำลังไฟฟ้ารีแอกทีฟที่เครื่องกำเนิดไฟฟ้าจ่าย Q_e หลังจากขจัดฟอล์ทมีค่าลดลง เพราะมีการจ่ายกำลังไฟฟ้ารีแอกทีฟจากตัว SVC เข้าสู่ระบบ

4. ในช่วง ΔP_e เป็นบวก SVC จะมีการจ่ายค่ากำลังไฟฟ้ารีแอกทีฟในปริมาณที่ค่อนข้างคงที่ ทำให้กำลังไฟฟ้ารีแอกทีฟที่จ่ายออกมาจากเครื่องกำเนิดไฟฟ้า Q_e มีการเปลี่ยนแปลงตาม P_e แต่ในช่วง ΔP_e เป็นลบ ปริมาณ Q_e จะไม่ตกลงมากนัก เนื่องจากในช่วงนี้แรงดัน V_t และ V_e ค่อนข้างคงที่

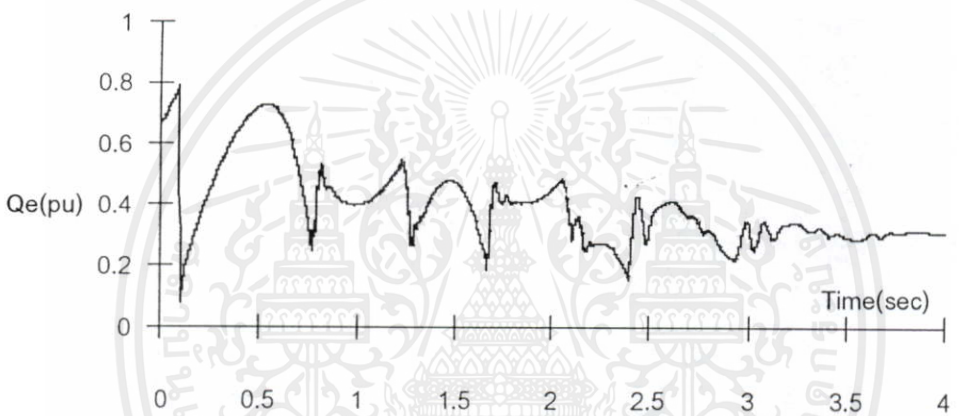
5. การเพิ่มขึ้นหรือลดลงของแรงดัน Vt มีทิศทางเดียวกับการปรับ ϕ_f เพิ่มขึ้นหรือลดลง โดย ϕ_f จะมีการปรับเพิ่มขึ้นหรือลดลงในย่ายแคบๆ

เส้นทางการควบคุม SVC บนเฟสเพลน แสดงไว้ในรูปที่ 6.11 โดยในสภาวะปกติระบบจะทำงานที่จุด (0,0) บนเฟสเพลน เมื่อเกิดฟอล์ท ณ เวลา t = 0 วินาที E_p จะมีค่าน้อยกว่าศูนย์และเลื่อนออกมาอยู่คอร์ดแดรนท์ที่ 3 หลังจาก 0.1 วินาทีทันทีที่ฟอล์ทถูกขจัด จะต้องปรับ SVC เป็นโหลดคาปาซิทีฟทันที เพื่อเป็นการระบายเอาพลังงานส่วนเกินที่อยู่ในตัวเครื่องกำเนิดไฟฟ้าออกมาให้มากที่สุด ส่งผลให้ความเร็ว ω ของเครื่องกำเนิดไฟฟ้าไม่เพิ่มสูงขึ้นจนทำให้มุม δ เกินค่าวิกฤต การปรับ SVC เป็นโหลดคาปาซิทีฟทำให้ E_p มีค่าเพิ่มขึ้น และ $\alpha 1 E_p^{\bullet}$ มีค่าลดลง ซึ่งจะเลื่อนจุดทำงานให้กลับเข้ามาใกล้จุด(0,0) มากที่สุด เส้นทางการควบคุมบนเฟสเพลนในรูปที่ 6.11 มีความสอดคล้องกับกราฟสัญญาณในรูปที่ 6.10 ทุกประการ ดังนั้นเมื่อเครื่องกำเนิดไฟฟ้าซิงโครนัสกลับมาทำงานที่จุดสมดุลอีกครั้ง จุดทำงานบนเฟสเพลนก็จะกลับมาอยู่ที่จุด (0,0) เช่นเดิม

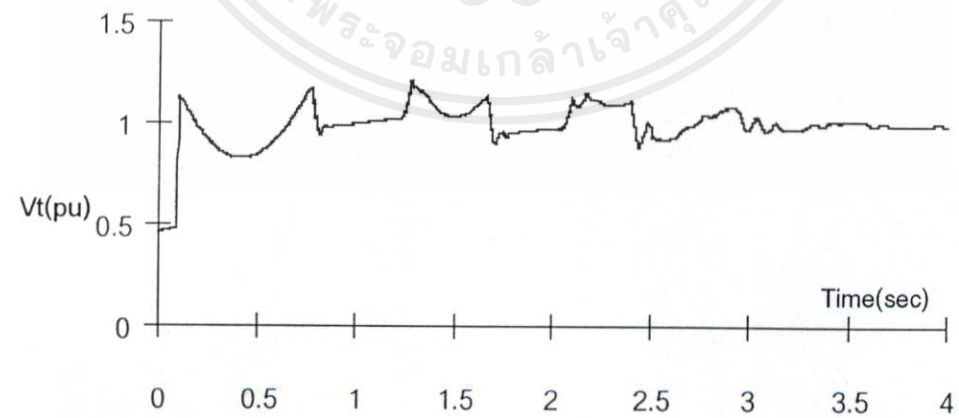
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(a)



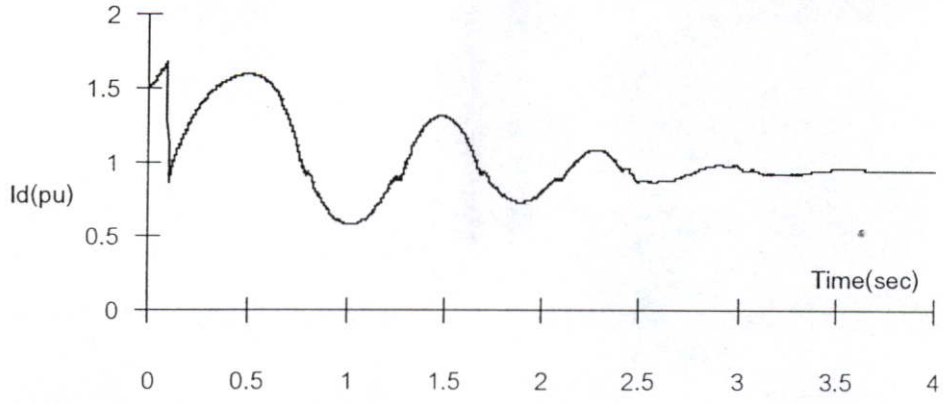
(b)



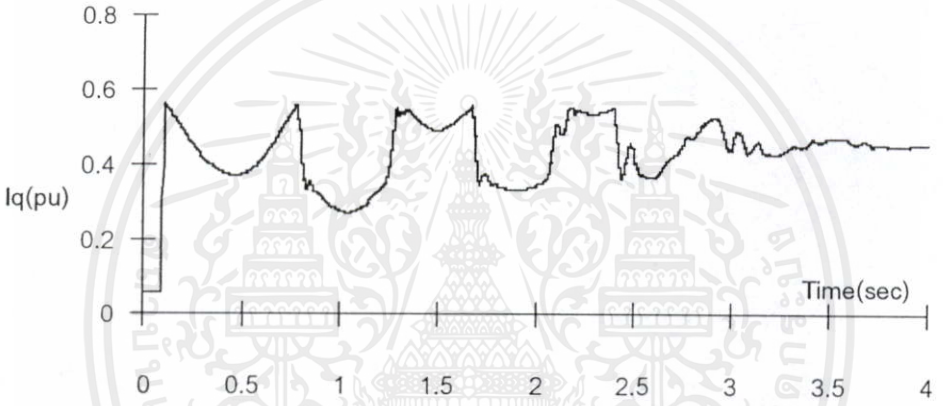
(c)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

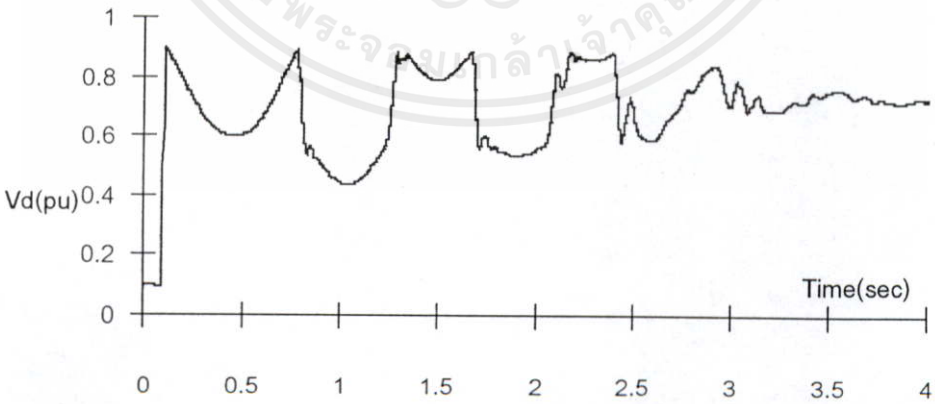
รูปที่ 6.10 สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



(d)



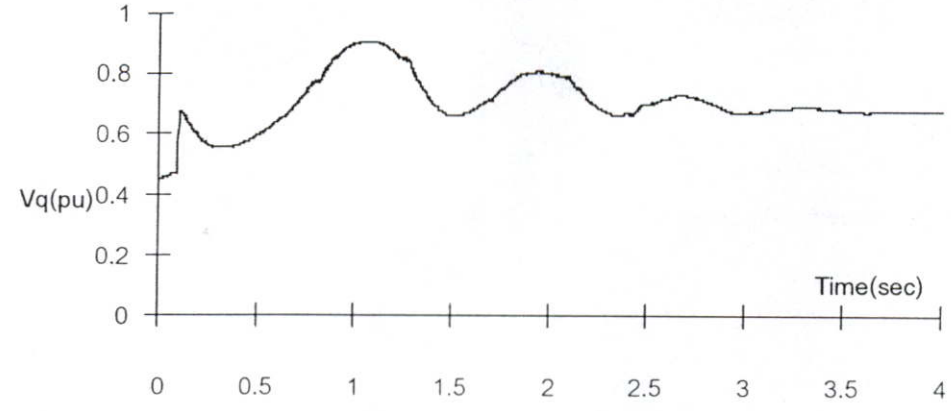
(e)



(f)

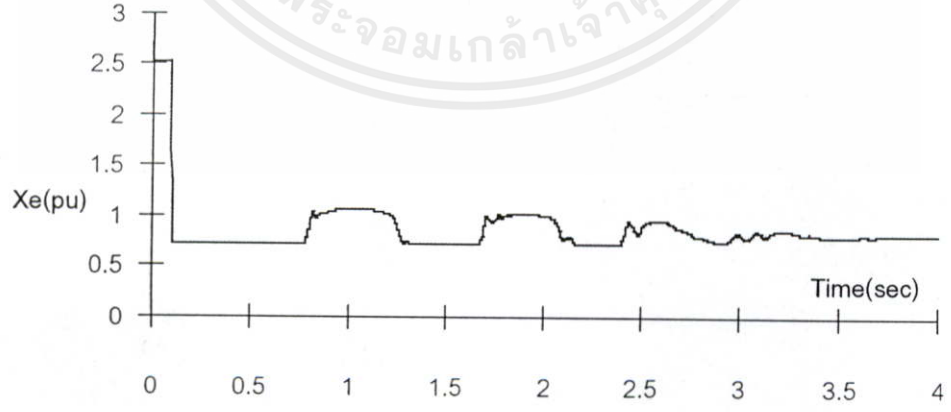
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 6.10 (ต่อ) สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



(g)

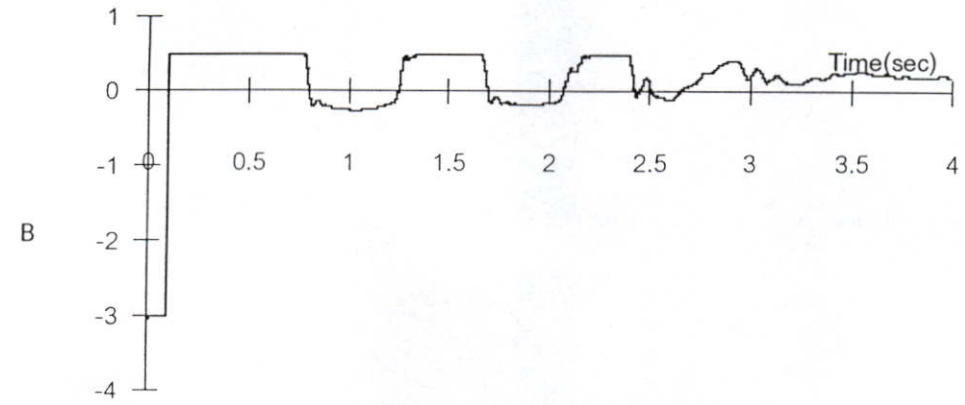
(h)



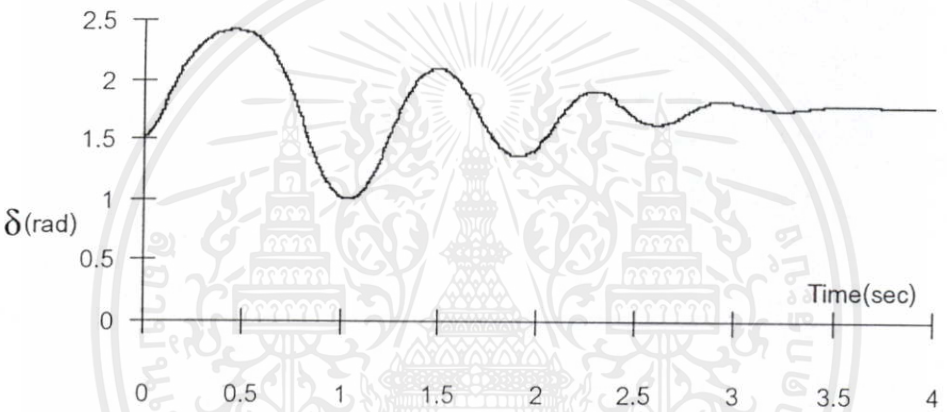
(i)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

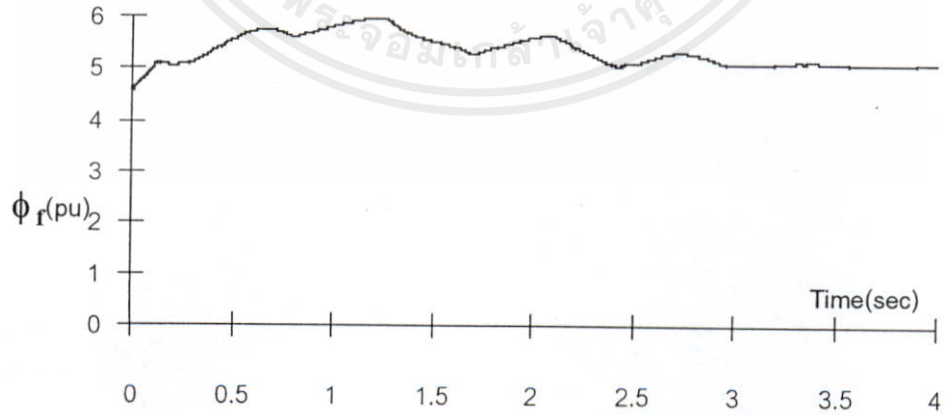
รูปที่ 6.10 (ต่อ) สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



(j)



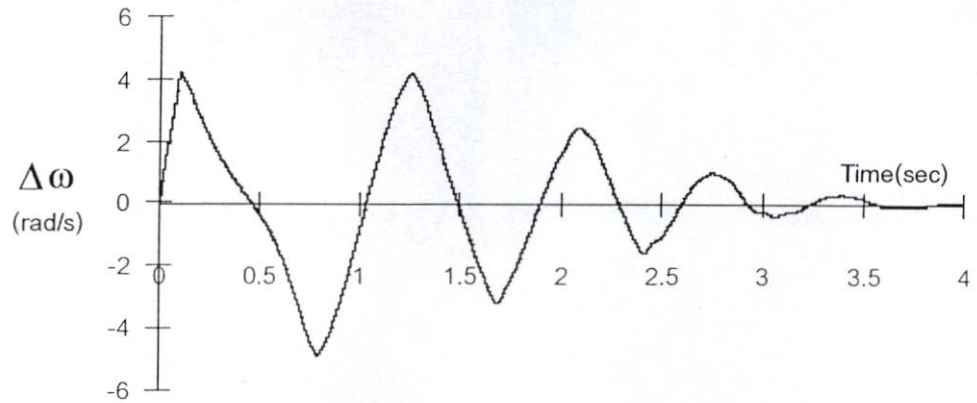
(k)



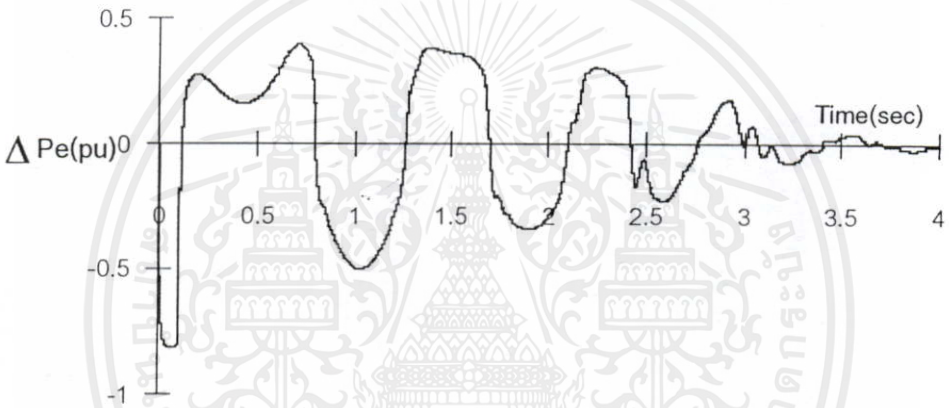
(l)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

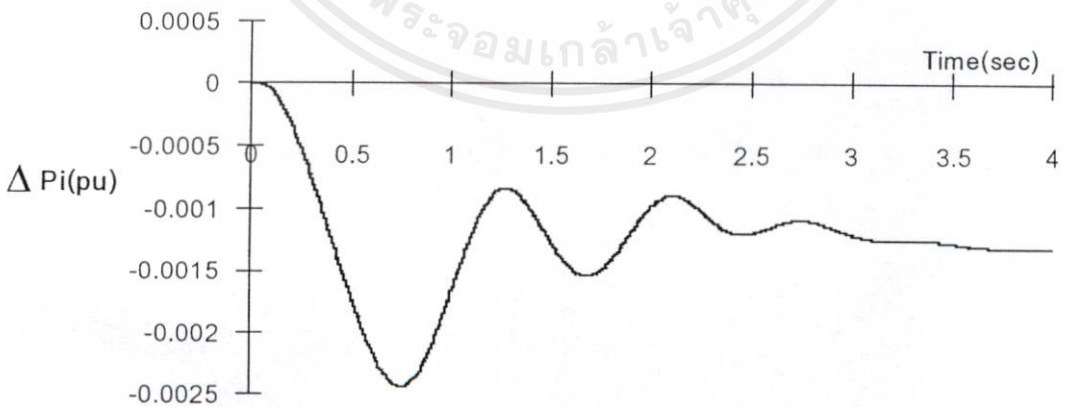
รูปที่ 6.10 (ต่อ) สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



(m)



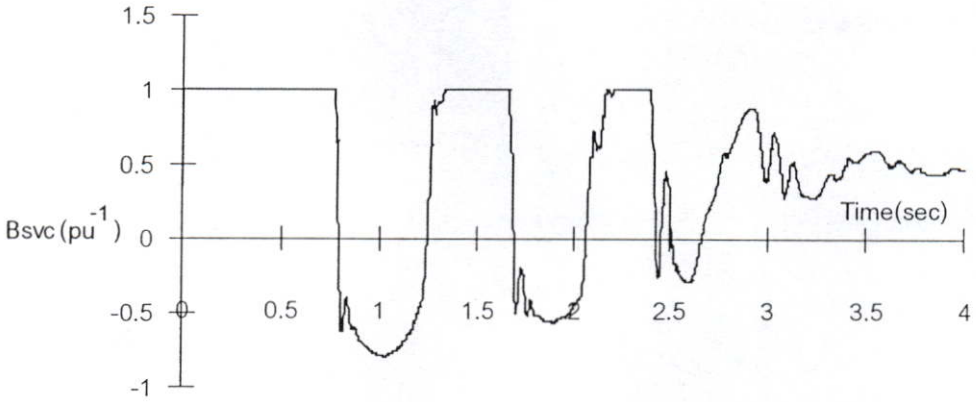
(n)



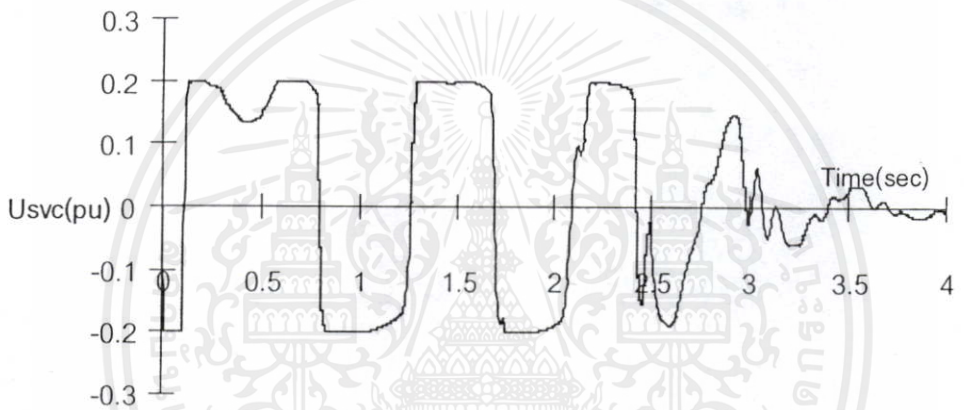
(o)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

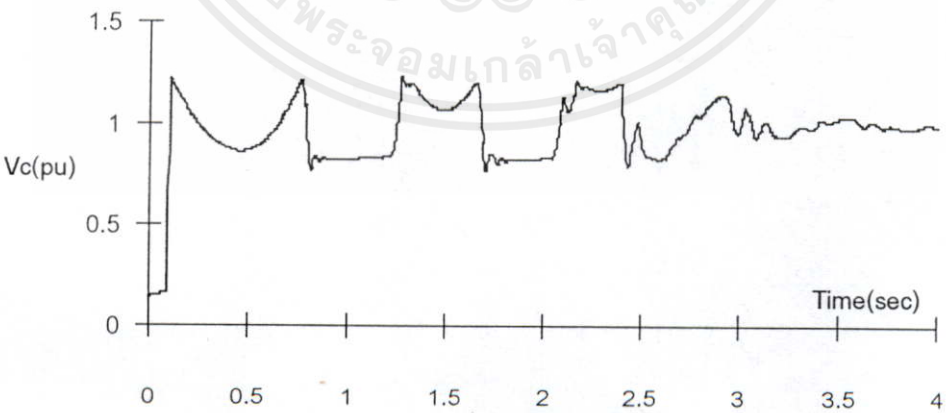
รูปที่ 6.10 (ต่อ) สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



(p)



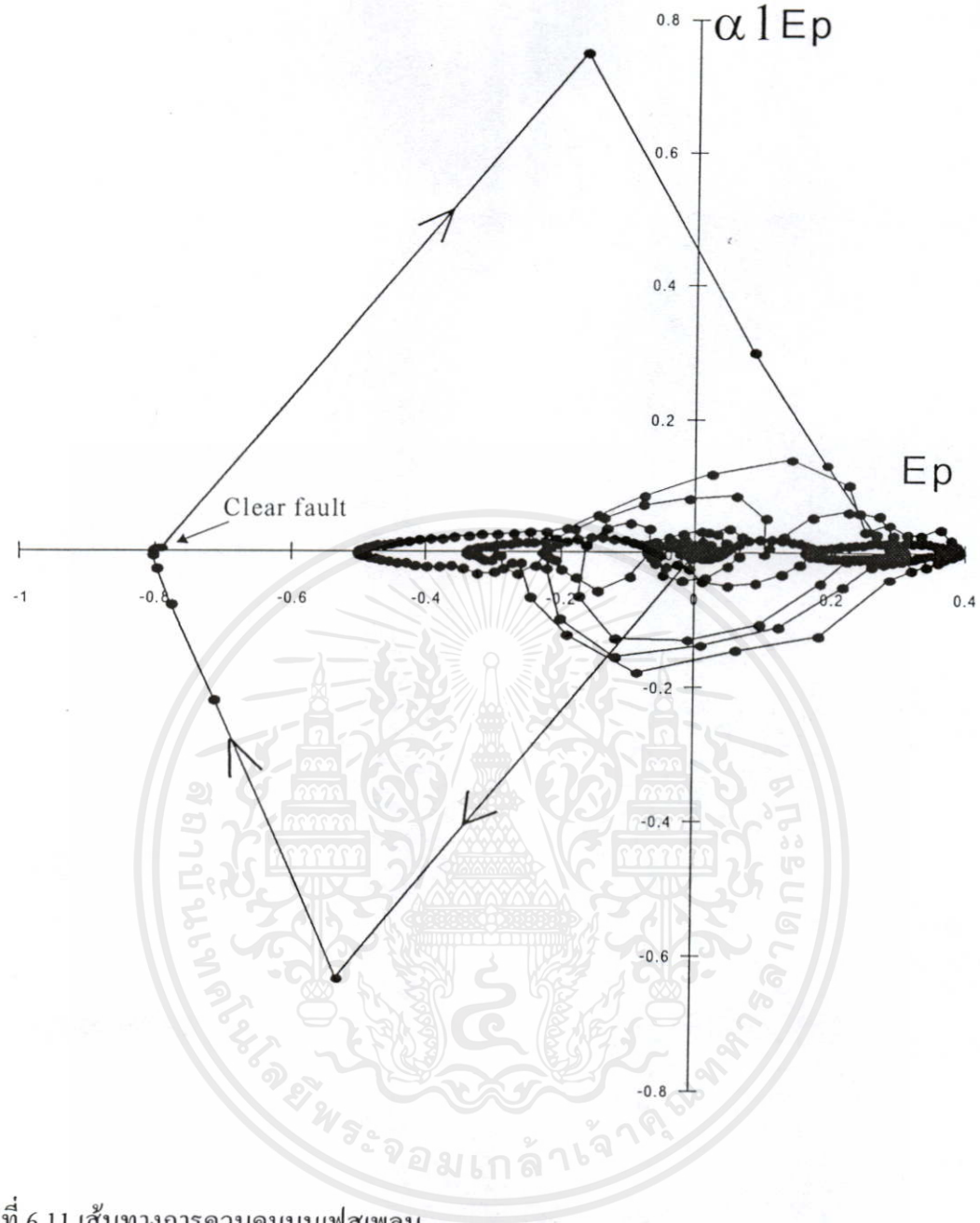
(q)



(r)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 6.10 (ต่อ) สัญญาณของปริมาณต่างๆเมื่อควบคุมด้วยตัวควบคุมฟัซซี่ของ SVC



รูปที่ 6.11 เส้นทางการควบคุมบนเฟสเพลน

6.8 การประยุกต์ใช้งาน

ในหัวข้อ 6.1-6.3 ได้กล่าวถึงตัวควบคุมฟัซซี่ที่อาศัยข้อมูลจากเฟสเพลน โดยนำไปประยุกต์ใช้กับ SVC วิธีการดังกล่าวนอกจากจะใช้กับ SVC แล้ว ยังอาจนำไปใช้กับ PSS ซึ่งจะควบคุมร่วมกับ AVR โดยหลักการของตัวควบคุมฟัซซี่ยังคงเหมือนเดิม เพียงแต่สัญญาณอินพุทป้อนเข้าตัวควบคุมฟัซซี่ของ SVC ใช้ ΔP_e แต่ถ้านำมาใช้กับ PSS จะใช้ $\Delta\omega$ แทน ซึ่งจะพบในบทต่อไปสำหรับ $\Delta\omega$-PSS ชนิดฟัซซี่ ที่ใช้หลักการออกแบบเช่นเดียวกับตัวควบคุมฟัซซี่ของ SVC นั่นเอง

บทที่ 7
การออกแบบโปรแกรมและวิธีใช้โปรแกรม

เนื้อหาบทนี้เป็นการอธิบายขั้นตอนการออกแบบโปรแกรมและวิธีใช้โปรแกรม โดยแบ่งเป็น 4 ส่วน คือ ส่วนแรกเป็นการออกแบบโปรแกรมจำลองระบบ ซึ่งประกอบด้วย โปรแกรมจำลองระบบกรณีควบคุมด้วย PSS โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่ โปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน ส่วนที่สอง เป็นการออกแบบโปรแกรมเจเนติกอัลกอริทึ่ม ซึ่งได้ทำการทดสอบประสิทธิภาพของโปรแกรมด้วยการใช้หาค่าสูงสุดของฟังก์ชั่นทดสอบมาตรฐาน ส่วนที่สามเป็นการปรับปรุงโปรแกรมส่วนแรกเพื่อนำมาใช้ในโปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC โดยใช้วิธีไอเทอร์เรชั่นและวิธีเจเนติกอัลกอริทึ่ม และส่วนที่สี่เป็นวิธีใช้โปรแกรมต่างๆดังที่กล่าวมาข้างต้น

7.1 การออกแบบโปรแกรมสำหรับจำลองระบบ

โปรแกรมสำหรับจำลองระบบที่เขียนขึ้นประกอบด้วยการจำลองระบบ 3 กรณี คือ

1. กรณีควบคุมด้วย PSS อย่างเดียว ได้แก่ การควบคุมด้วย ΔP-PSS , $\Delta\omega$-PSS และ $\Delta\omega$-PSS ชนิดฟัซซี่
2. กรณีควบคุมด้วย SVC ชนิดฟัซซี่ อย่างเดียว
3. กรณีควบคุมแบบผสมผสานระหว่าง SVC ชนิดฟัซซี่กับ PSS ทั้ง 3 ชนิด

ขั้นตอนการทำงานหลักของโปรแกรมจำลองระบบทั้ง 3 กรณี เป็นไปตามโฟลว์ชาร์ตรูปที่ 7.1 โดยมีบางขั้นตอนภายในบล๊อก A ที่แตกต่างกันเล็กน้อยตามวิธีการควบคุม ส่วนสมการระบบที่ใช้คำนวณแสดงไว้ในภาคผนวก ก.

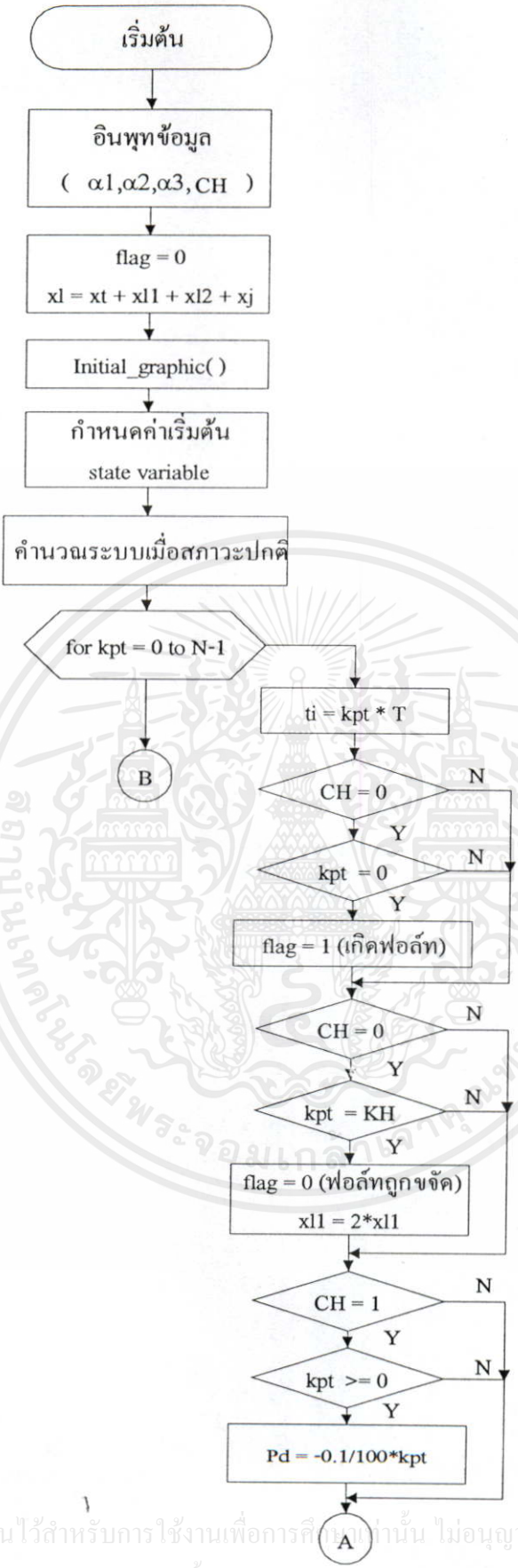
7.1.1 โปรแกรมจำลองระบบกรณีควบคุมด้วย PSS

ระบบจำลองกรณีควบคุมด้วย PSS อย่างเดียว มีความแตกต่างในส่วนของขั้นตอนภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1 ดังต่อไปนี้

1. ในการคำนวณสมการโครงข่ายไฟฟ้าของระบบ จะไม่คิดผลของ SVC ที่ติดอยู่ ณ กึ่งกลางสายส่ง (เปิดวงจร C ณ จุดกึ่งกลางสายส่ง ในวงจรสมมูลย์)
2. ไม่มีสัญญาณควบคุม Usvc (Usvc = 0)
3. มีสัญญาณควบคุม Upss (Upss $\neq$ 0)

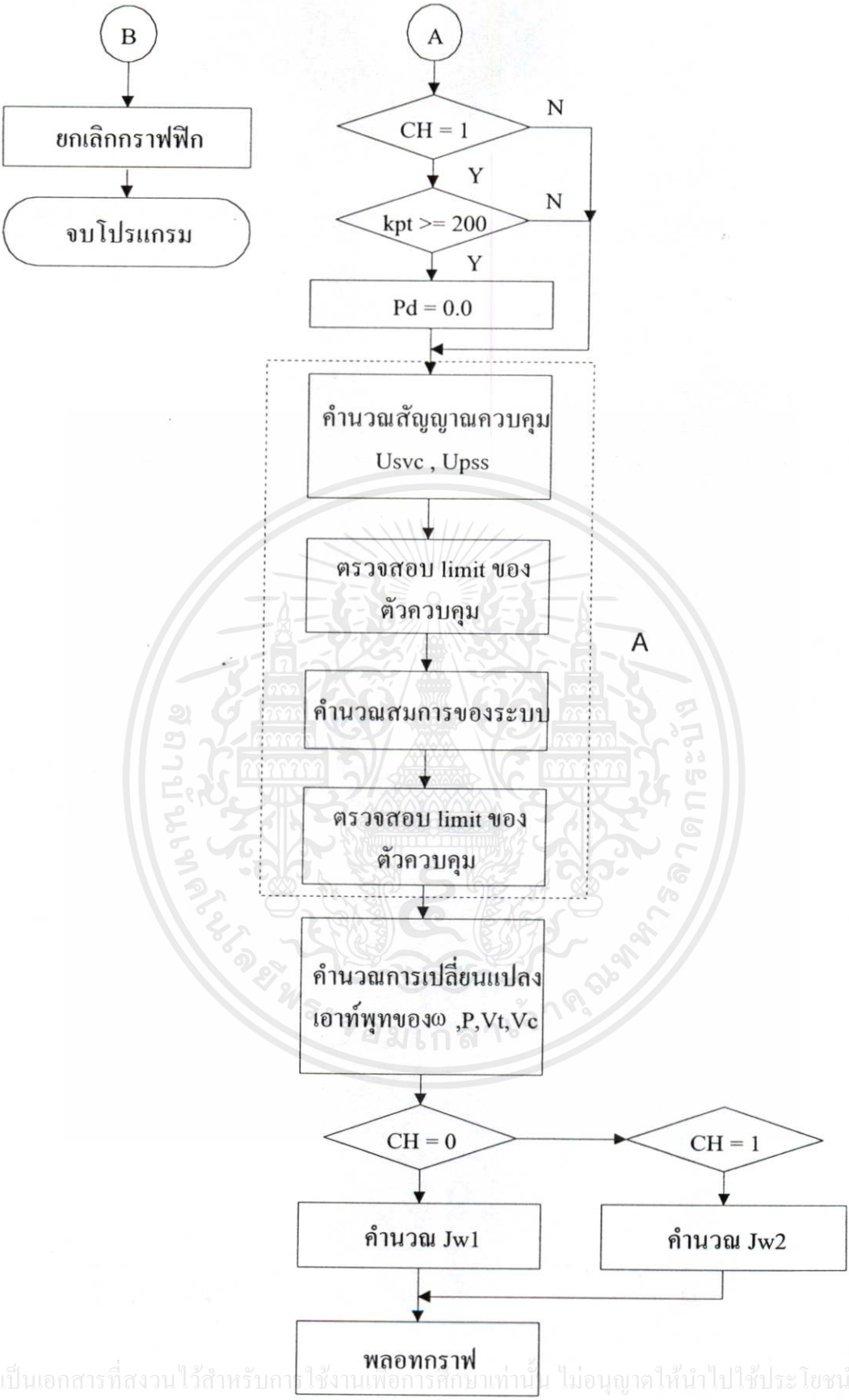
ขั้นตอนที่แตกต่างกันดังกล่าวแสดงในโฟลว์ชาร์ตรูปที่ 7.2 ซึ่งแสดงเฉพาะขั้นตอนที่เปลี่ยนไปภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



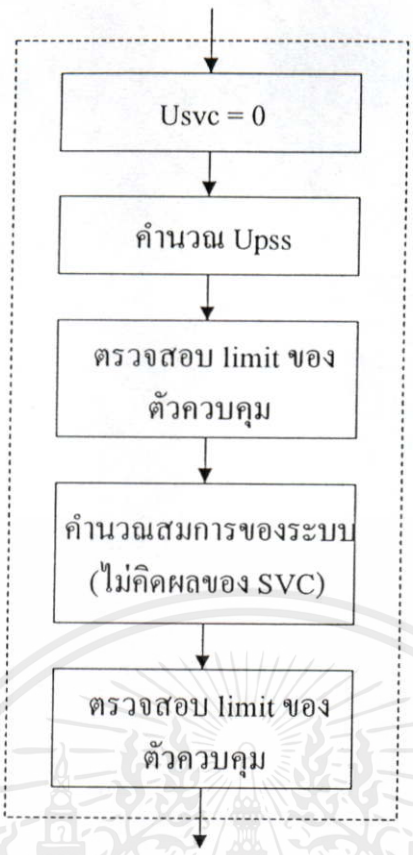
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7.1 โฟลว์ชาร์ตขั้นตอนหลักสำหรับโปรแกรมจำลองระบบ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7.1 (ต่อ)



รูปที่ 7.2 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมด้วย PSS

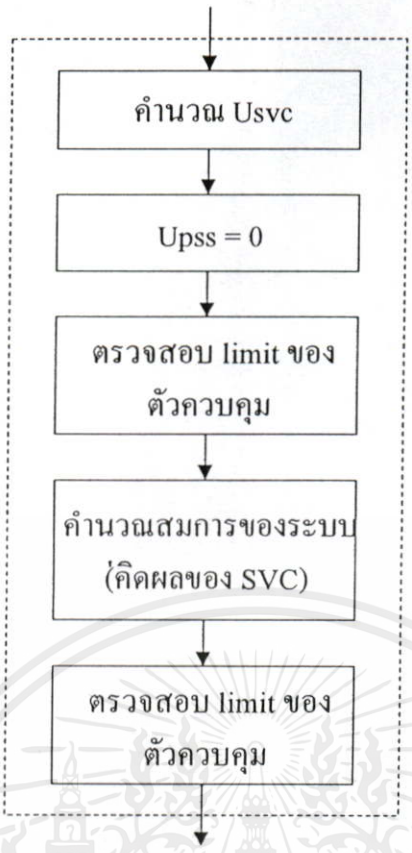
7.1.2 โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่

ระบบจำลองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว มีความแตกต่างในส่วนของขั้นตอนภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1 ดังต่อไปนี้

1. ในการคำนวณสมการโครงข่ายไฟฟ้าของระบบ จะคิดผลของ SVC ที่ติดอยู่ ณ กึ่งกลางสายส่ง (แทนด้วย C ณ จุดกึ่งกลางสายส่ง ในวงจรสมมูลย์)
2. มีสัญญาณควบคุม Usvc (Usvc $\neq$ 0)
3. ไม่มีสัญญาณควบคุม Upss (Upss = 0)

ขั้นตอนที่แตกต่างกันดังกล่าวแสดงในโฟลว์ชาร์ตรูปที่ 7.3 ซึ่งแสดงเฉพาะขั้นตอนที่เปลี่ยนไปภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



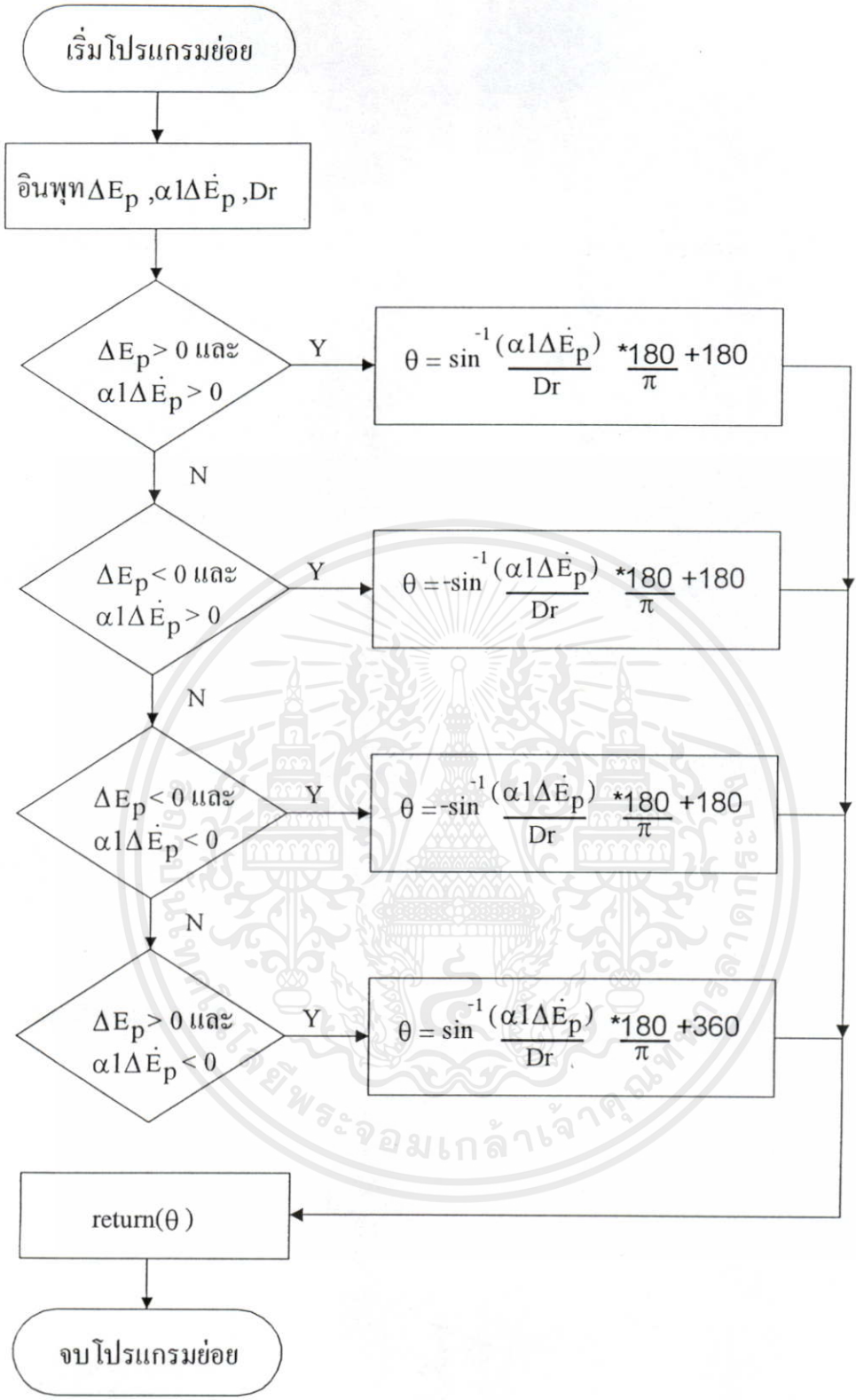
รูปที่ 7.3 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่

ในขั้นตอนการคำนวณ Usvc จะต้องมีการหาค่า θ , μ_N และ μ_G การหาค่าดังกล่าวสามารถอธิบายได้ดังต่อไปนี้

7.1.2.1 การหามุม θ

ข้อมูล ΔE_p และ $\alpha 1 \Delta E_p^{\bullet}$ เป็นตัวกำหนดจุด P(k) บนเฟสเพลนในรูปที่ 6.1 ซึ่ง มุมθ สามารถหาได้จากขั้นตอนตามโฟลว์ชาร์ตรูปที่ 7.4

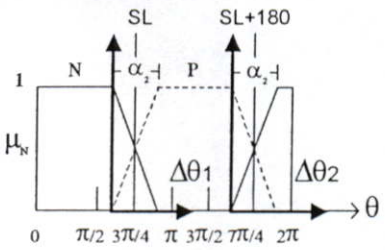
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.4 โฟลว์ชาร์ตขั้นตอนการหามุม θ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.1.2.2 การหา μ_N



รูปที่ 7.5 การคำนวณหาค่าฟังก์ชั่นการเป็นสมาชิกของ θ

ในการหาค่า μ_N สามารถพิจารณาได้ 5 กรณี ต่อไปนี้

<u>กรณีที่ 1</u> $0 \leq \theta < (SL- \alpha 2/2)$

$$\mu_N(\theta) = 1.0 \tag{7.1}$$

<u>กรณีที่ 2</u> $(SL- \alpha 2/2) \leq \theta < (SL+ \alpha 2/2)$

$$\mu_N(\Delta\theta 1) = \mu_N(\theta) = a\Delta\theta 1 + b \tag{7.2}$$

หาค่า b โดยแทนค่า $\Delta\theta 1 = 0$ จะได้

$$\mu_N(0) = \mu_N(0-(SL- \alpha 2/2)) = b = 1.0 \tag{7.3}$$

ความชัน a สามารถพิจารณาได้จากรูปที่ 7.5

$$a = -1/\alpha 2 \tag{7.4}$$

แทนค่า a , b ลงในสมการที่ (7.4)

$$\mu_N(\Delta\theta 1) = \mu_N(\theta) = (-1/\alpha 2)\Delta\theta 1 + 1.0 \tag{7.5}$$

แทนค่า $\Delta\theta 1 = \theta-(SL-\alpha 2/2)$ ลงในสมการที่ (7.5)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\mu_N(\theta) = (-1/\alpha 2)[\ \theta-(SL-\alpha 2/2)] + 1.0$$

$$\mu_N(\theta) = (-\theta+SL) / \alpha 2 + 0.5 \qquad (7.6)$$

<u>กรณีที่ 3</u> $(SL+\alpha 2/2) \leq \theta < (SL+180-\alpha 2/2)$

$$\mu_N(\theta) = 0.0 \qquad (7.7)$$

<u>กรณีที่ 4</u> $(SL+180-\alpha 2/2) \leq \theta < (SL+180+\alpha 2/2)$

$$\mu_N(\Delta\theta 2) = \mu_N(\theta) = a\Delta\theta 2 + b \qquad (7.8)$$

หาค่า b โดยแทนค่า $\Delta\theta 2 = 0$ จะได้

$$\mu_N(0) = \mu_N(0-(SL+180-\alpha 2/2)) = b = 0.0 \qquad (7.9)$$

ความชัน a สามารถพิจารณาได้จากรูปที่ 7.5

$$a = 1/\alpha 2 \qquad (7.10)$$

แทนค่า a , b ลงในสมการที่ (7.4)

$$\mu_N(\Delta\theta 2) = \mu_N(\theta) = (1/\alpha 2)\Delta\theta 2 \qquad (7.11)$$

แทนค่า $\Delta\theta 2 = \theta-(SL+180-\alpha 2/2)$ ลงในสมการที่ (7.5)

$$\mu_N(\theta) = (1/\alpha 2)[\ \theta-(SL+180-\alpha 2/2)]$$

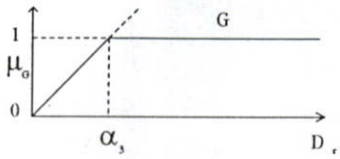
$$\mu_N(\theta) = [\theta-(SL+180)] / \alpha 2 + 0.5 \qquad (7.12)$$

<u>กรณีที่ 5</u> $(SL+180+\alpha 2/2) \leq \theta < 360$

$$\mu_N(\theta) = 1.0 \qquad (7.13)$$

สงวนลิขสิทธิ์ ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.1.2.3 การหา μ_G



รูปที่ 7.6 การคำนวณหาค่าฟังก์ชั่นการเป็นสมาชิกของ Dr

ในการหาค่า μ_G สามารถพิจารณาได้ 2 กรณี ต่อไปนี้

<u>กรณีที่ 1</u> $0 \leq Dr \leq \alpha 3$

พิจารณาจากรูปที่ 7.6 จะได้

$$\mu_G(Dr) < 1.0$$
$$\mu_G(Dr) = Dr/\alpha 3 \qquad (7.14)$$

<u>กรณีที่ 2</u> $Dr > \alpha 3$

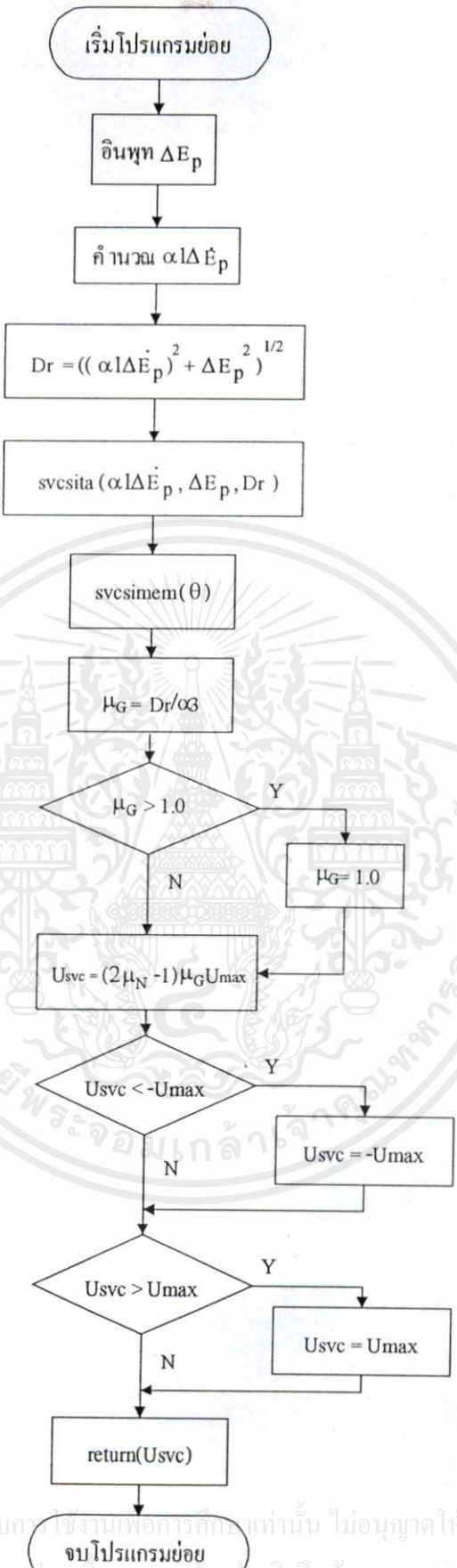
$$\mu_G(Dr) > 1.0$$
$$\mu_G(Dr) = 1.0 \qquad (7.15)$$

โปรแกรมที่เขียนขึ้นเพื่อคำนวณ Usvc ประกอบด้วยโปรแกรมย่อย 3 ส่วน คือ

1. svcfuz ทำหน้าที่ คำนวณ $\mu_G(Dr)$ และ Usvc
2. svcsita ทำหน้าที่ คำนวณหามุม θ
3. svcsimem ทำหน้าที่ คำนวณ $\mu_N(\theta)$

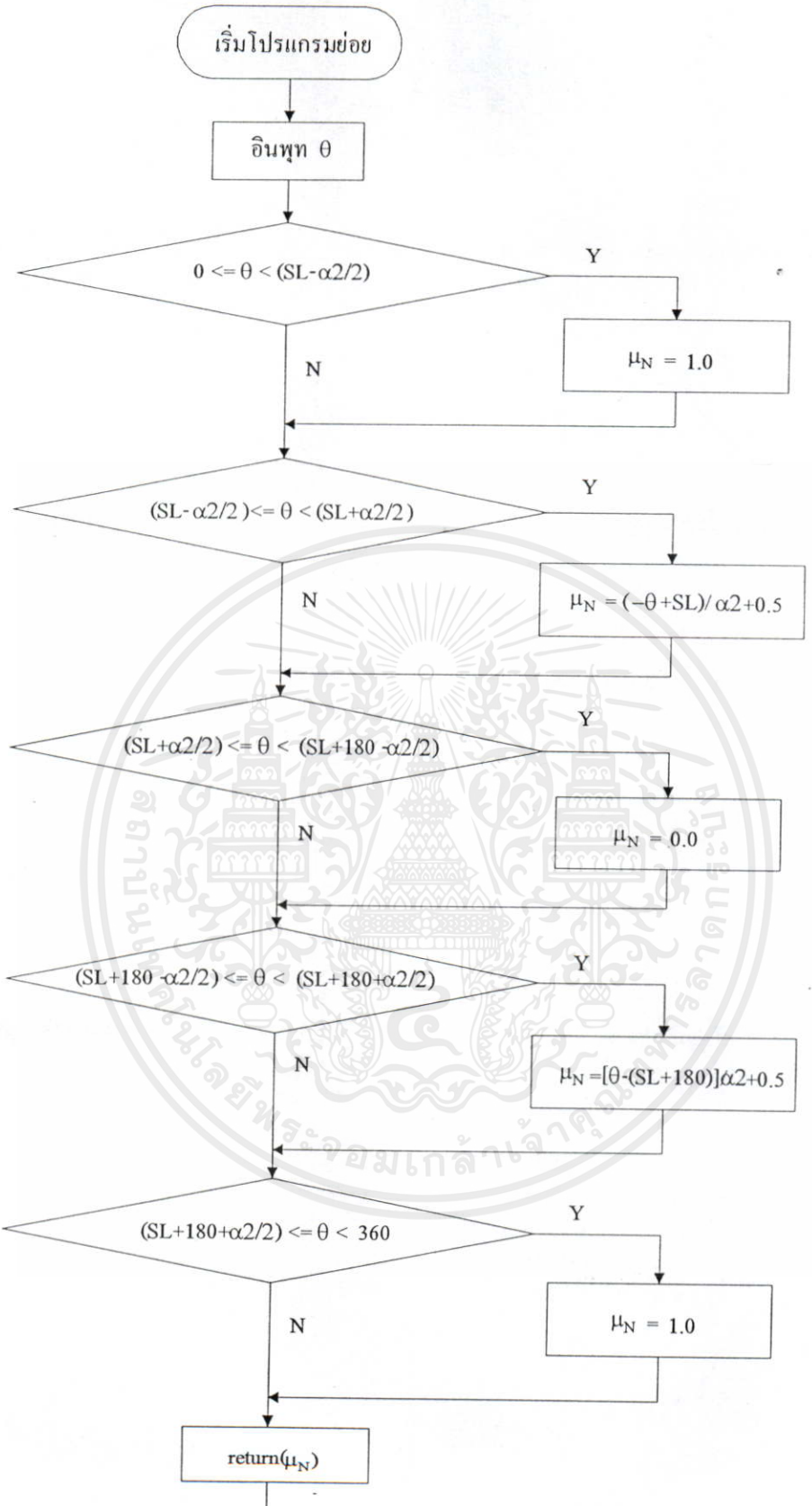
โปรแกรมย่อย svcfuz มีขั้นตอนตามโฟลว์ชาร์ตรูปที่ 7.7 โปรแกรมย่อย svcsita มีขั้นตอนตามโฟลว์ชาร์ตรูปที่ 7.4 และโปรแกรมย่อย svcsimem มีขั้นตอนตามโฟลว์ชาร์ตรูปที่ 7.8 รายละเอียดโปรแกรมดูได้ในภาคผนวก ค.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.7 โฟลว์ชาร์ตโปรแกรมย่อย svcfuz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.8 โฟลว์ชาร์ตโปรแกรมย่อย svcsimem

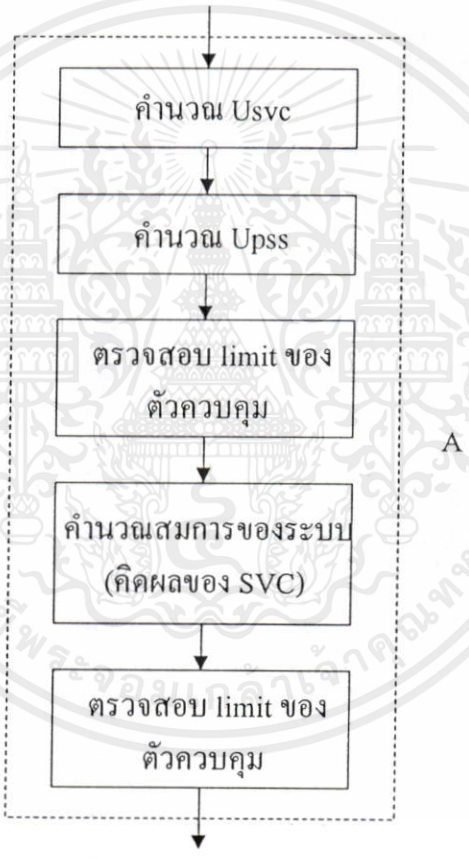
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.1.3 โปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน

ระบบจำลองกรณีควบคุมแบบผสมผสานระหว่าง SVC ชนิดฟัซซี่กับ PSS ทั้ง 3 ชนิด มีความแตกต่างในส่วนของขั้นตอนภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1 ดังต่อไปนี้

1. ในการคำนวณสมการโครงข่ายไฟฟ้าของระบบ จะคิดผลของ SVC ที่ติดอยู่ ณ กึ่งกลางสายส่ง (แทนด้วย C ณ จุดกึ่งกลางสายส่ง ในวงจรสมมูลย์)
2. มีสัญญาณควบคุม Usvc (Usvc $\neq$ 0)
3. มีสัญญาณควบคุม Upss (Upss $\neq$ 0)

ขั้นตอนที่แตกต่างกันดังกล่าวแสดงในโฟลว์ชาร์ตรูปที่ 7.9 ซึ่งแสดงเฉพาะขั้นตอนที่เปลี่ยนไปภายในบล๊อก A ของโฟลว์ชาร์ตรูปที่ 7.1



รูปที่ 7.9 โฟลว์ชาร์ตโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน

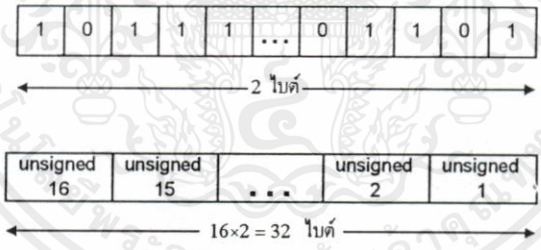
7.2 การออกแบบโปรแกรมเจเนติกอัลกอริทึ่ม

เนื่องจากการใช้งานเจเนติกอัลกอริทึ่ม บ่อยครั้งที่ต้องอาศัยเวลาในการประมวลผลเป็นเวลานาน ดังนั้นถ้าปัญหานั้นต้องอาศัยโครโมโซมที่มีขนาดยาว การรันอาจผิดพลาดได้เนื่องจากหน่วยความจำไม่พอ การออกแบบโปรแกรมเจเนติกอัลกอริทึ่มนี้จึงได้มีการคำนึงถึงปัญหานี้ด้วย

และเลือกใช้วิธีการจัดการและเก็บข้อมูลแบบบิตแทนที่การจัดการและเก็บข้อมูลแบบอาเรย์ โปรแกรมที่เขียนขึ้นประกอบด้วย การครอสโอเวอร์แบบ 1 จุด แบบ 2 จุด และแบบยูนิฟอร์ม การมิวเทชั่น การทำสเกลลิ่งแบบเชิงเส้น การทำสเกลลิ่งแบบเอกโปเนนเทียล การเลือกแบบรูเลท และการเลือกแบบทัวร์นาเมนท์

7.2.1 การจัดการหน่วยความจำ

เพื่อเป็นการลดการใช้หน่วยความจำของโปรแกรมเจเนติก ในกรณีที่การแก้ปัญหาต้องใช้โครโมโซมที่มีขนาดยาวและต้องใช้การรันเป็นเวลานาน โปรแกรมที่เขียนขึ้นจึงใช้ข้อมูลชนิด unsigned เก็บข้อมูลของโครโมโซมแบบบิต และใช้พอยท์เตอร์ในการเข้าถึงตำแหน่งข้อมูลที่เก็บแทนการเก็บข้อมูลของโครโมโซมโดยใช้อาร์เรย์ เหตุผลที่สำคัญที่ใช้พอยท์เตอร์แทนอาร์เรย์ก็คือ เมื่อเก็บข้อมูลแบบอาร์เรย์ โปรแกรมจะทำการจองเนื้อที่หน่วยความจำไว้ล่วงหน้า แม้ว่าข้อมูลนั้นจะไม่ได้ใช้งาน ซึ่งจะทำให้สิ้นเปลืองเนื้อที่หน่วยความจำโดยไม่จำเป็น แต่การใช้พอยท์เตอร์ เนื้อที่หน่วยความจำจะถูกจองไว้ก็ต่อเมื่อมีการเรียกใช้ และสามารถคืนเนื้อที่หน่วยความจำได้ตลอดเวลาได้ตามต้องการ วิธีการเก็บข้อมูลของแต่ละโครโมโซม แสดงในรูปที่ 7.10 ซึ่งเป็นตัวอย่างการเก็บข้อมูลสำหรับโครโมโซมขนาด 16 บิต จากรูปที่ 7.10 จะสังเกตได้ว่า การเก็บข้อมูลแบบบิตจะใช้หน่วยความจำเพียง 2 ไบต์ ในขณะที่การเก็บข้อมูลแบบอาร์เรย์จะใช้หน่วยความจำ 32 ไบต์



รูปที่ 7.10 การเก็บข้อมูลของโครโมโซม

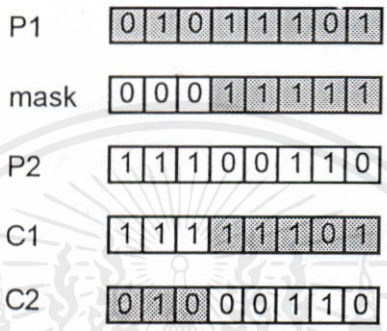
โปรแกรมย่อยที่เกี่ยวข้องกับการจัดการหน่วยความจำของโปรแกรมเจเนติกที่เขียนขึ้น คือ initmalloc() ซึ่งทำหน้าที่จองพื้นที่หน่วยความจำที่จำเป็นต้องใช้ รายละเอียดของโปรแกรมย่อยนี้สามารถดูได้จากภาคผนวก ช.

7.2.2 การครอสโอเวอร์

โปรแกรมย่อยสำหรับการครอสโอเวอร์ที่เขียนขึ้นมีด้วยกัน 3 วิธี คือ การครอสโอเวอร์แบบ 1 จุด การครอสโอเวอร์แบบ 2 จุด และการครอสโอเวอร์แบบยูนิฟอร์ม โดยแต่ละวิธีมีขั้นตอนดังต่อไปนี้

1. การครอสโอเวอร์แบบ 1 จุด

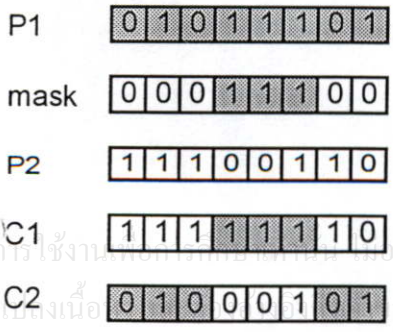
การทำงานของการครอสโอเวอร์วิธีนี้แสดงไว้ในรูปที่ 7.11 โดยจะเริ่มจากการสร้าง mask ด้วยวิธีการสุ่มในลักษณะคล้ายกับรูปที่ 7.11 สำหรับ C1 ตำแหน่งบิตที่เป็น 1 ของ mask จะสำเนาโครโมโซม P1 บางส่วน และตำแหน่งที่เป็น 0 จะสำเนาโครโมโซม P2 บางส่วน สำหรับ C2 ตำแหน่งบิตที่เป็น 1 ของ mask จะสำเนาโครโมโซม P2 บางส่วน และตำแหน่งที่เป็น 0 ของ mask จะสำเนาโครโมโซม P1 บางส่วน โฟลว์ชาร์ตของการครอสโอเวอร์แบบ 1 จุด แสดงในรูปที่ 7.15 และมีรายละเอียดในโปรแกรมย่อยชื่อ one_point_crossover ในภาคผนวก จ.



รูปที่ 7.11 การครอสโอเวอร์แบบ 1 จุด

2. การครอสโอเวอร์แบบ 2 จุด

การทำงานของการครอสโอเวอร์วิธีนี้แสดงไว้ในรูปที่ 7.12 โดยจะเริ่มจากการสร้าง mask ด้วยวิธีการสุ่มลักษณะคล้ายกับรูปที่ 7.12 สำหรับ C1 ตำแหน่งบิตที่เป็น 1 ของ mask จะสำเนาโครโมโซม P1 บางส่วน และตำแหน่งที่เป็น 0 จะสำเนาโครโมโซม P2 บางส่วน สำหรับ C2 ตำแหน่งบิตที่เป็น 1 ของ mask จะสำเนาโครโมโซม P2 บางส่วน และตำแหน่งที่เป็น 0 ของ mask จะสำเนาโครโมโซม P1 บางส่วน โฟลว์ชาร์ตของการครอสโอเวอร์แบบ 2 จุด แสดงในรูปที่ 7.15 และมีรายละเอียดในโปรแกรมย่อยชื่อ two_point_crossover ในภาคผนวก จ.

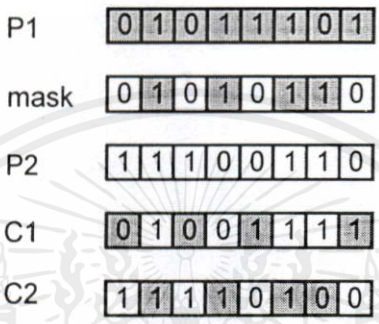


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.12 การครอสโอเวอร์แบบ 2 จุด

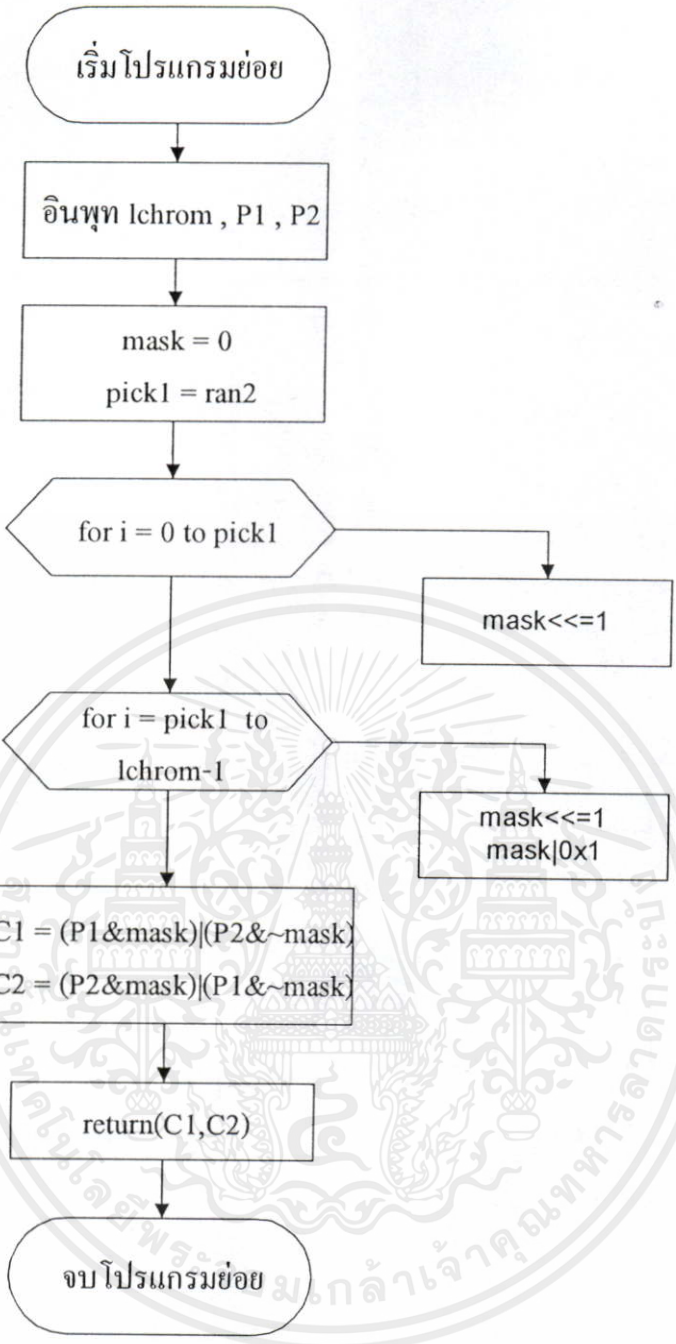
3. การครอสโอเวอร์แบบยูนิฟอร์ม

การทำงานของการครอสโอเวอร์วิธีนี้แสดงในรูปที่ 7.14 โดยจะเริ่มจากการสร้าง mask ด้วยวิธีการสุ่มลักษณะคล้ายรูปที่ 7.13 สำหรับ C1 บิตที่เป็น 0 ของ mask จะสำเนาโครโมโซม P1 บางส่วน บิตที่เป็น 1 ของ mask จะสำเนาโครโมโซม P2 บางส่วน สำหรับ C2 บิตที่เป็น 0 ของ mask จะสำเนาโครโมโซม P2 บางส่วน บิตที่เป็น 1 จะสำเนาโครโมโซม P1 บางส่วน โฟลว์ชาร์ตของการครอสโอเวอร์แบบยูนิฟอร์มแสดงในรูปที่ 7.16 และมีรายละเอียดในโปรแกรมย่อยชื่อ uniform_crossover ในภาคผนวก จ.



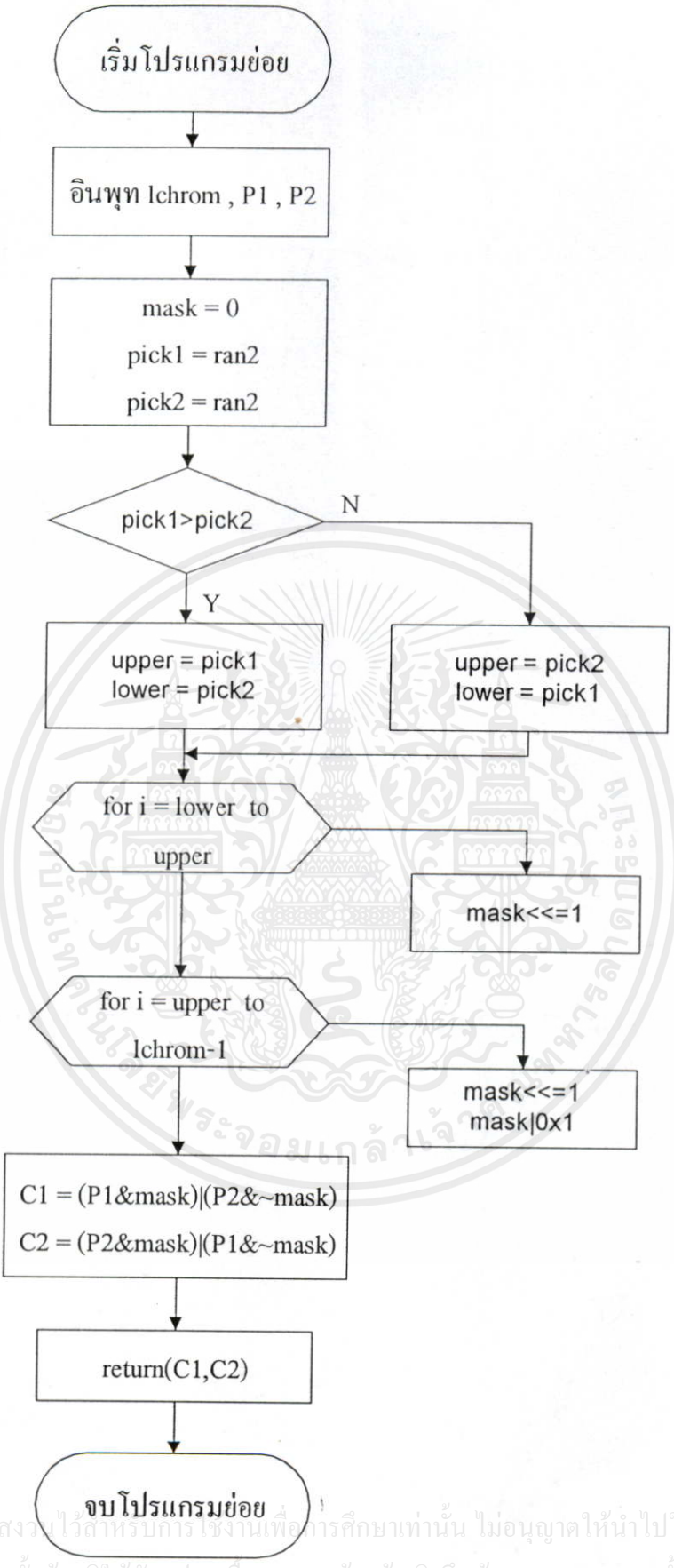
รูปที่ 7.13 การครอสโอเวอร์แบบยูนิฟอร์ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



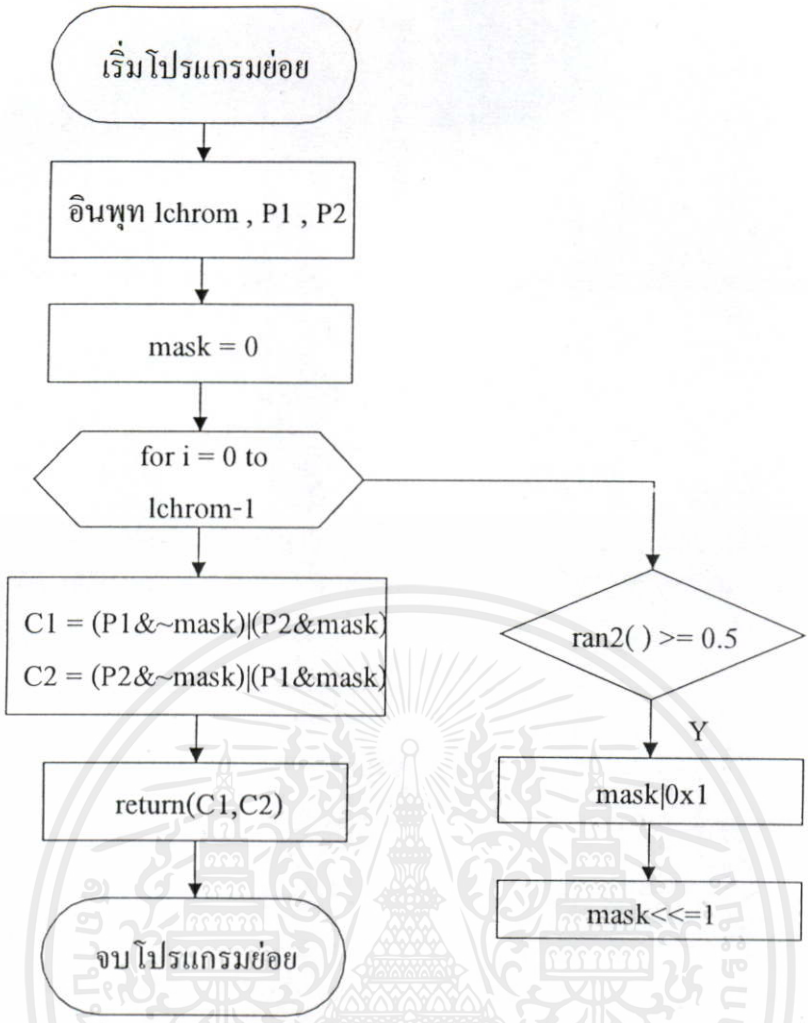
รูปที่ 7.14 โฟลว์ชาร์ตการครอสโอเวอร์แบบ 1 จุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

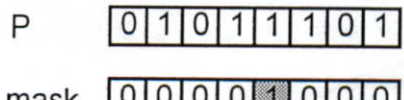
รูปที่ 7.15 โฟลว์ชาร์ตการครอสโอเวอร์แบบ 2 จุด



รูปที่ 7.16 โฟลว์ชาร์ตการครอสโอเวอร์แบบยูนิฟอร์ม

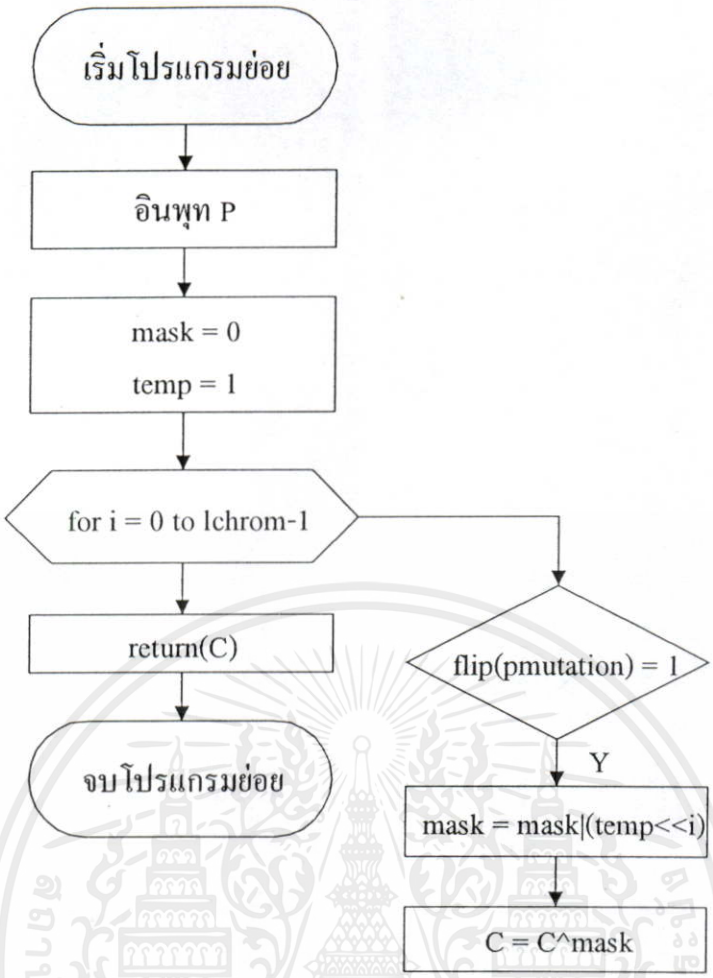
7.2.3 การมิวเทชั่น

ขั้นตอนการมิวเทชั่นแสดงในรูปที่ 7.17 โดยจะเริ่มจากการสร้าง mask ด้วยวิธีการสุ่มลักษณะคล้ายรูปที่ 7.17 ตำแหน่งบิตที่เป็น 1 ของ mask จะทำการกลับค่าของ P1 จาก 1 เป็น 0 หรือจาก 0 เป็น 1 โฟลว์ชาร์ตของการมิวเทชั่นแสดงในรูปที่ 7.18 และมีรายละเอียดในโปรแกรมย่อยชื่อ mutation ในภาคผนวก จ.



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7.17 การมิวเทชั่น



รูปที่ 7.18 โฟลว์ชาร์ตการมิวเทชั่น

7.2.4 การทำสเกลลิ่ง

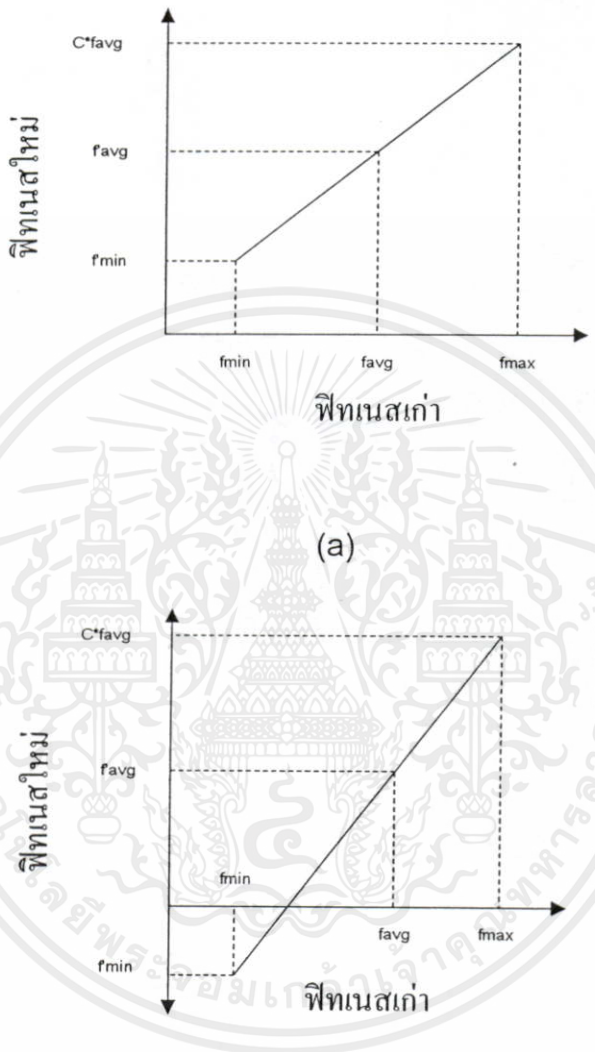
วัตถุประสงค์ของการทำสเกลลิ่งก็เพื่อป้องกันการเกิดการลู่เข้าก่อนกำหนด (Premature) อันเนื่องจากการครอบครองพื้นที่วงล้อรูเลทของประชากรบางตัวที่มีค่าฟิทเนสสูงมากๆ ทำให้ประชากรที่มีค่าฟิทเนสต่ำมากๆไม่มีโอกาสถูกเลือก ซึ่งบางครั้งประชากรที่มีค่าฟิทเนสต่ำๆ อาจมีบางส่วนเป็นข้อมูลดีก็ได้ การทำสเกลลิ่งจะเป็นการปรับค่าฟิทเนสใหม่ทั้งหมด ก่อนเข้าสู่การคัดเลือกด้วยวิธีวงล้อรูเลท ซึ่งจะช่วยเพิ่มโอกาสสำหรับประชากรที่มีค่าฟิทเนสต่ำ ได้มีโอกาสถูกเลือกมากขึ้น

สำหรับโปรแกรมที่เขียนขึ้น จะมีวิธีการทำสเกลลิ่ง 2 วิธี คือ การทำสเกลลิ่งแบบเชิ่งเส้น และการทำสเกลลิ่งแบบเอกโปเนนเทียล ซึ่งมีรายละเอียดดังต่อไปนี้

1. การทำสเกลลิ่งแบบเชิ่งเส้น

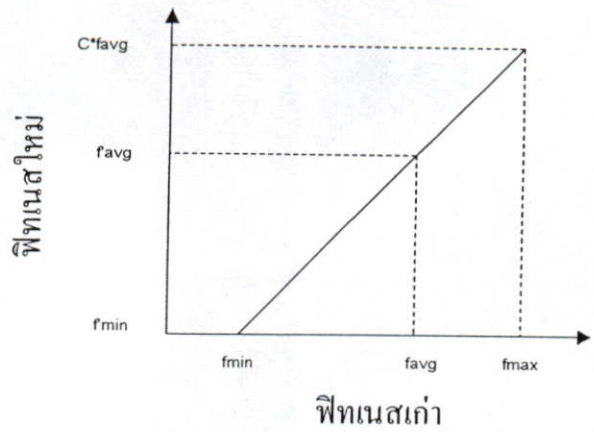
วิธีการทำสเกลลิ่งวิธีนี้ จะแปลงค่าฟิทเนสใหม่โดยใช้สมการเส้นตรง $y = ax+b$ โดย x คือ ค่าฟิทเนสเก่า และ y คือค่าฟิทเนสใหม่ ข้อมูลที่ใช้ในการหาสมการเส้นตรงนี้ คือ fmin , favg และ

fmax ในเจนเนอเรชั่นรุ่นก่อน โดยค่าฟิทเนสใหม่ที่ได้จะต้องมีค่าเป็นบวกเสมอ แต่ก็มีความเป็นไปได้ที่จะได้สมการเส้นตรงที่แปลงค่าฟิทเนสใหม่แล้วได้ค่าลบ ดังกรณีรูปที่ 7.19(b) ซึ่งสามารถแก้ไขได้โดยแปลงสมการเส้นตรงไปเป็นรูปที่ 7.19(c)



รูปที่ 7.19 สมการเส้นตรงที่เป็นไปได้สำหรับการทำสเกลลิ่งแบบเชิงเส้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(c)

รูปที่ 7.19 (ต่อ) สมการเส้นตรงที่เป็นไปได้สำหรับการทำสเกลลิ่งแบบเชิงเส้น

การสร้างสมการเส้นตรงสามารถพิจารณาได้เป็น 2 กรณีดังต่อไปนี้

<u>กรณีที่ 1</u> ค่าฟิทเนสใหม่มีค่าเป็นบวกทุกค่า (รูปที่ 7.19(a))

จากรูป 7.19(a) ให้ fmultiple = C , Umax = fmax , Uavg = favg และ Umin = fmin สมการต่างๆพิจาณาได้ดังนี้

$$delta = U\max - Uavg \tag{7.16}$$

ค่าความชัน a สามารถพิจารณาได้จากรูปที่ 7.19(a)

$$a = \frac{(fmultiple * Uavg) - Uavg}{delta} = \frac{(fmultiple - 1)Uavg}{delta} \tag{7.17}$$

จากสมการ y = ax+b จัดรูปใหม่เป็น b = y-ax

และ ปกติค่า favg จะมีค่าเท่ากับ f'avg (ค่าเฉลี่ยของฟิทเนสหลังจากปรับค่าแล้วไม่เปลี่ยนแปลงจากค่าเฉลี่ยฟิทเนสเดิม) ดังนั้น จากรูปที่ 7.20(a) x = favg = Uavg แล้ว y = f'avg = Uavg

แทนค่า a จากสมการ (7.17) และค่า x และ y จะได้

$$b = Uavg - \frac{(fmultiple - 1)Uavg * Uavg}{delta} \tag{7.18}$$

$$b = \frac{Uavg(U\max - fmultiple * Uavg)}{delta} \tag{7.19}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>กรณีที่ 2</u> ค่าฟิทเนสใหม่บางค่าติดลบ (รูปที่ 7.19(b))

ในเหตุการณ์ปกติที่ค่าฟิทเนสใหม่เป็นบวกทุกค่าจะเกิดขึ้นเมื่อ

$$y = ax + b \rangle 0 \tag{7.20}$$

เมื่อ แทน x = Umin จะได้

$$aU\min + b \rangle 0 \tag{7.21}$$

$$U\min \rangle -\frac{b}{a} \tag{7.22}$$

แทนค่า a , b จากสมการ (7.17) และ (7.19) ลงในสมการ (7.22) จะได้

$$U\min \rangle \frac{fmultiple - U\max}{(fmultiple - 1)} \tag{7.23}$$

ซึ่งถ้าตรวจสอบแล้วเงื่อนไขสมการ (7.23) แล้วเป็นจริงก็จะเกิดกรณีรูปที่ 7.19(a) แต่ถ้าไม่เป็นจริงก็จะเกิดกรณีรูปที่ 7.19(b) ในกรณีที่เกิดเหตุการณ์ดังรูปที่ 7.19(b) จะต้องมีการแปลงสมการเส้นตรงไปเป็นรูปที่ 7.19(c) โดยสามารถหาสมการเส้นตรงของรูป 7.19(c) ได้ดังนี้

$$delta = Uavg - U\min \tag{7.24}$$

ความชัน a หาได้จากการพิจารณารูปที่ 7.19(c)

$$a = \frac{Uavg}{delta} \tag{7.25}$$

จากสมการ y = ax+b จัดรูปใหม่เป็น b = y-ax

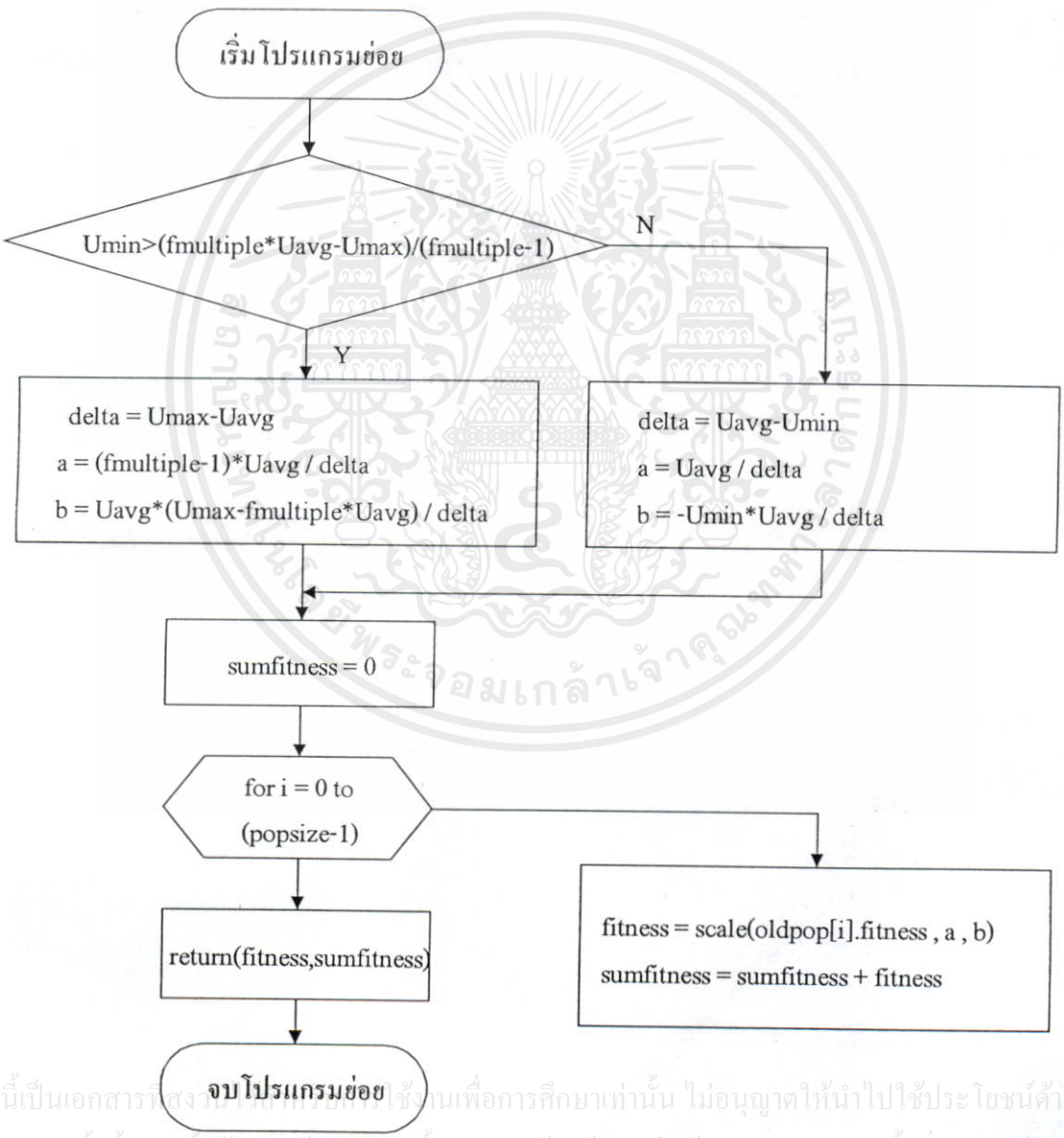
และ ปกติค่า favg จะมีค่าเท่ากับ f avg (ค่าเฉลี่ยของฟิทเนสหลังจากปรับค่าแล้วไม่เปลี่ยนแปลงจากค่าเฉลี่ยฟิทเนสเดิม) ดังนั้น จากรูปที่ 7.19(c) x = favg = Uavg แล้ว y = f avg = Uavg แทนค่า a จากสมการ (7.25) และค่า x และ y จะได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีต่างๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$b = Uavg - \frac{Uavg * Uavg}{delta} \tag{7.26}$$

$$b = -\frac{U\min * Uavg}{delta} \tag{7.27}$$

จากสมการที่ได้ทำการวิเคราะห์กรณีต่างๆที่เป็นไปได้ของสมการเส้นตรง สามารถนำมาเขียนเป็นโปรแกรมย่อยสำหรับการทำสเกลลิ่งแบบเชิงเส้น ได้ตามโฟลว์ชาร์ตรูปที่ 7.20 ซึ่งมีรายละเอียดในโปรแกรมย่อยชื่อ prescale , scale , scalepop ในภาคผนวก จ.

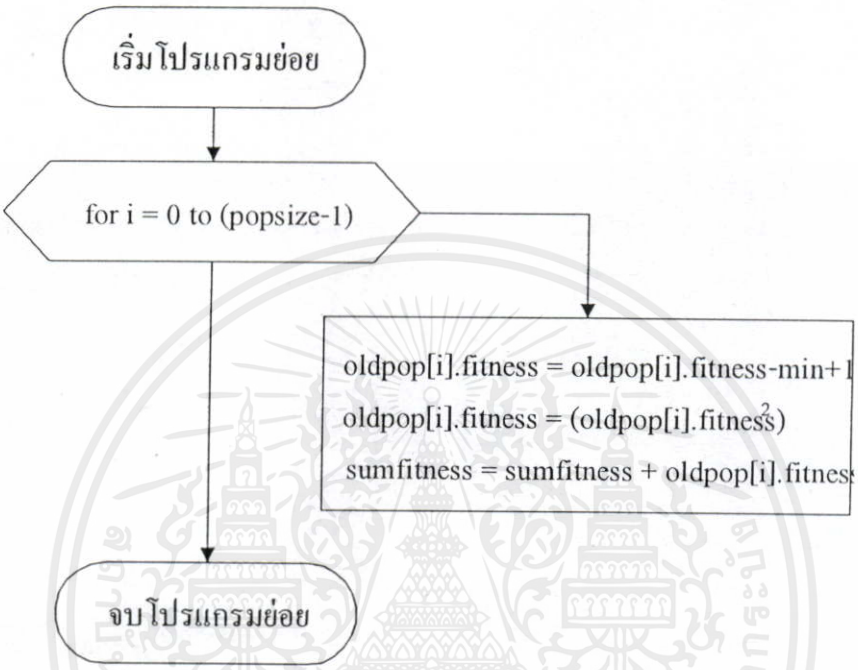


รูปที่ 7.20 โฟลว์ชาร์ตการทำสเกลลิ่งแบบเชิงเส้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การทำสเกลลิ่งแบบเอกโปเนนเทียล

วิธีนี้สามารถแก้ไขปัญหาการลู่เข้าจุดเหมาะสมเฉพาะที่ (Local hill) ซึ่งไม่ใช่จุดที่ให้ค่าฟิทเนสสูงสุดได้ดี ทำให้มีโอกาสหลุดออกจากจุดเหมาะสมเฉพาะที่มากขึ้น เนื่องจากประชากรที่มีค่าฟิทเนสต่ำๆ มีโอกาสถูกเลือกมากขึ้น โฟลว์ชาร์ตของการทำสเกลลิ่งแบบเอกโปเนนเทียลแสดงในรูปที่ 7.21 ซึ่งมีรายละเอียดในโปรแกรมย่อยชื่อ preselected_expo_scaling ในภาคผนวก จ.



รูปที่ 7.21 โฟลว์ชาร์ตการทำสเกลลิ่งแบบเอกโปเนนเทียล

7.2.5 การคัดเลือก

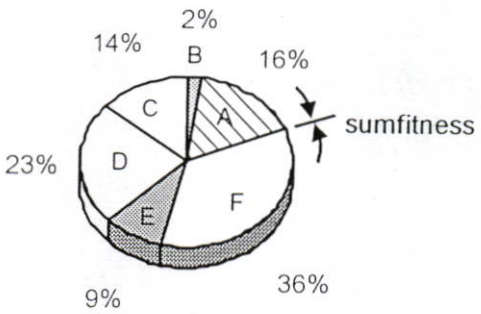
การคัดเลือกที่เขียนขึ้นในโปรแกรมมี 2 วิธี คือ การคัดเลือกแบบวงล้อรูเลท (Roulette wheel selection) และการคัดเลือกแบบทัวร์นาเมนท์ (Tournament selection) รายละเอียดของวิธีการคัดเลือกทั้งสอง สามารถอธิบายได้ดังต่อไปนี้

1. การคัดเลือกแบบวงล้อรูเลท

วิธีการนี้จะเป็นการคัดเลือกตามส่วน โดยพิจารณาจากค่าฟิทเนสของประชากรแต่ละตัว ดังเช่น ในรูปที่ 7.22 ซึ่งมีประชากรทั้งหมด 6 ตัว อันดับแรกจะแบ่งส่วนของพื้นที่รูเลทตามสัดส่วนของค่าฟิทเนส โดยให้เส้นรอบวงของวงล้อรูเลท คือ ผลรวมของค่าฟิทเนสของประชากรแต่ละตัว จากนั้นจะสุ่มค่าตั้งแต่ 0-sumfitness ถ้าตัวเลขที่สุ่มนั้นไปหยุดอยู่ในส่วนพื้นที่ใด ประชากรตัวนั้นก็จะถูกเลือก ดังนั้นประชากรที่มีค่าฟิทเนสสูงก็จะมีโอกาสถูกเลือกมากกว่าประชากรที่มีค่าฟิทเนส

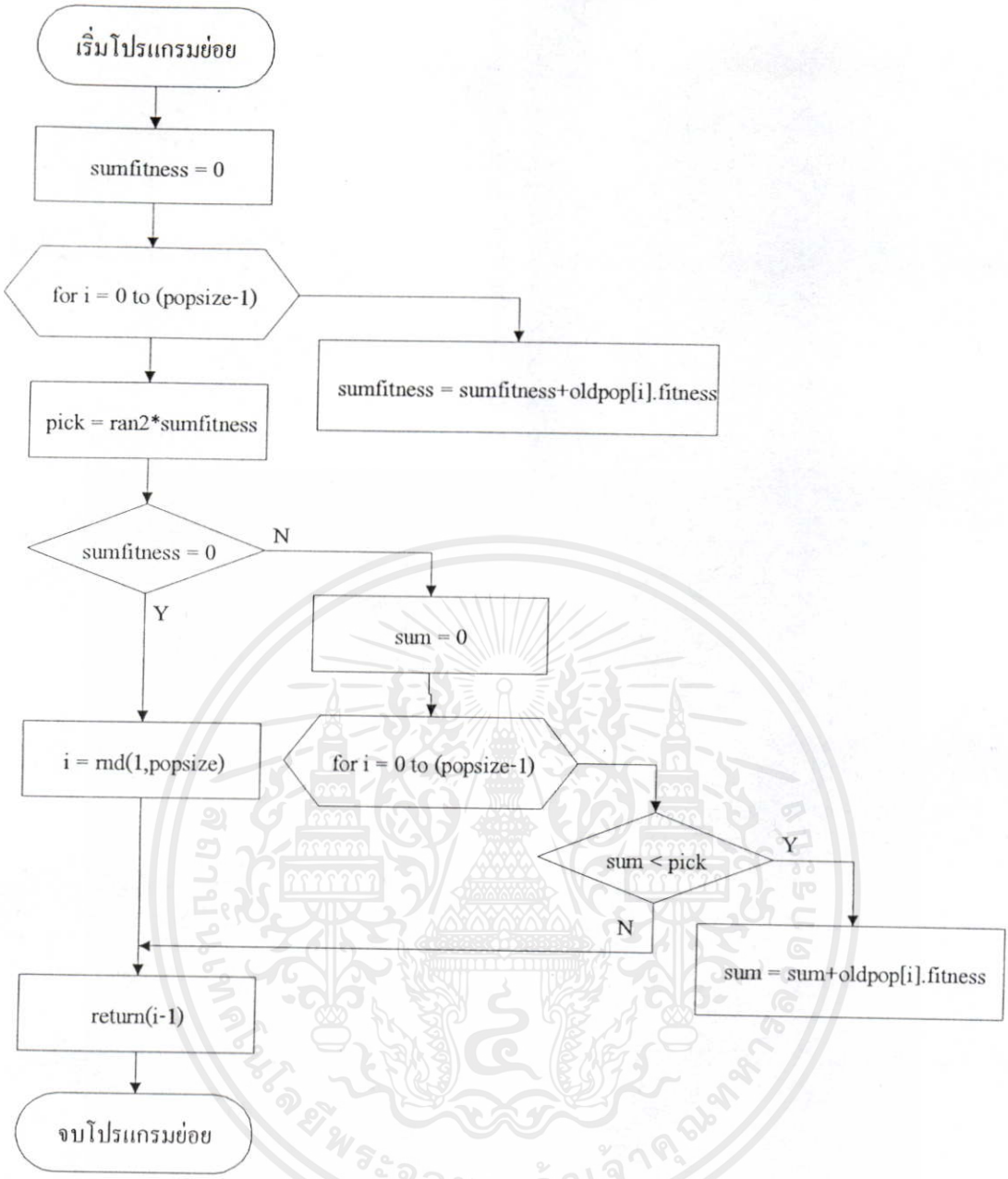
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่ำ วิธีการดังกล่าวเขียนเป็นโฟลว์ชาร์ตได้ดังรูปที่ 7.23 และมีรายละเอียดในโปรแกรมย่อยชื่อ selected ในภาคผนวก ง.



รูปที่ 7.22 พื้นที่ในวงล้อรูเลทที่แบ่งตามสัดส่วนของค่าฟิทเนส

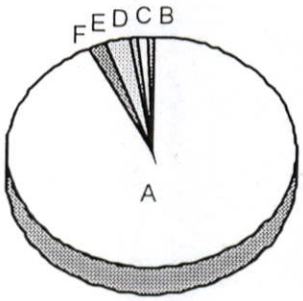
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.23 โฟลว์ชาร์ตการคัดเลือกแบบวงล้อรูเลท

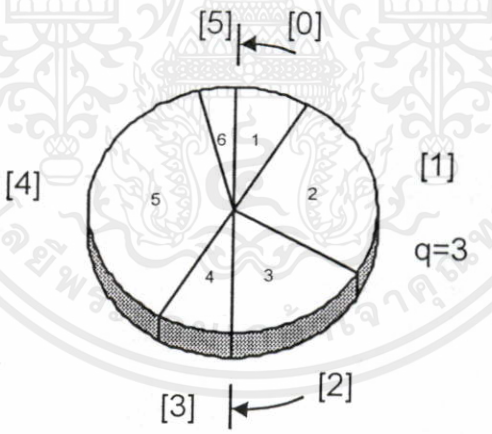
2. การคัดเลือกแบบทัวร์นาเมนท์

การคัดเลือกแบบวงล้อรูเลท อาจทำให้เกิดการลู่เข้าก่อนกำหนดได้ เช่น ในกรณีรูปที่ 7.24 มีประชากรตัวหนึ่ง ที่มีค่าฟิทเนสสูงกว่าค่าฟิทเนสของประชากรตัวอื่นมากๆ ในกรณีนี้จะเกิดการทำสำเนาประชากร A มากเกินไป เนื่องจากประชากร A ครอบครองพื้นที่ในวงล้อรูเลทมากทำให้ประชากรตัวอื่น ซึ่งอาจมีข้อมูลดีและนำไปสู่คำตอบที่ดีที่สุด มีโอกาสถูกเลือกน้อย ซึ่งถ้าปล่อยให้เกิดเหตุการณ์เช่นนี้เกิดต่อไป ประชากรทุกตัวในเจนเนอเรชั่นถัดไปก็จะเป็น A ทั้งหมด ซึ่งไม่ใช่คำตอบที่ดีที่สุดตามที่ต้องการ



รูปที่ 7.24 พื้นที่ในวงล้อรูเลทเมื่อเกิดการลู่เข้าก่อนกำหนด

การคัดเลือกแบบทัวนาร์เมนท์จะช่วยแก้ปัญหานี้ โดยจะทำการแบ่งการแข่งขันของประชากรทั้งหมดออกเป็นกลุ่มๆ ตามขนาดของทัวร์นาเมนท์ (tornament size : q) ดังเช่น รูปที่ 7.25 กำหนด q = 3 การแข่งขันของประชากรก็จะแบ่งเป็นกลุ่มละ 3 จำนวน 2 กลุ่ม ประชากรที่มีค่าฟิทเนสสูงสุดในแต่ละกลุ่มก็จะถูกเลือก ซึ่งจะสังเกตได้ว่า ถ้าใช้วิธีการคัดเลือกวิธีนี้กับรูปที่ 7.24 ประชากรที่มีค่าฟิทเนสน้อยก็จะมีโอกาสถูกเลือกมากขึ้น ปัญหาที่เกิดจากการครอบครองพื้นที่วงล้อรูเลทมากเกินไปของประชากรบางตัวที่ไม่ใช่คำตอบที่ดีที่สุดก็จะลดน้อยลง



รูปที่ 7.25 การคัดเลือกแบบทัวร์นาเมนท์ เมื่อ q = 3

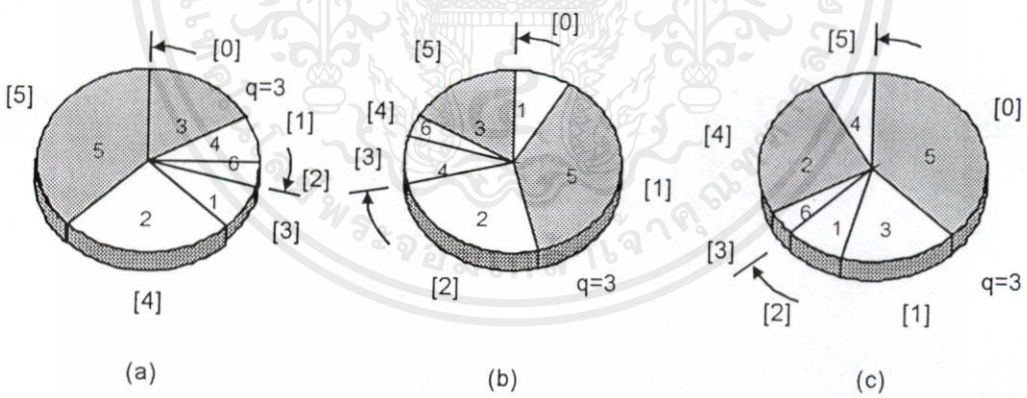
เพื่อให้เกิดความเข้าใจในการทำงานของการคัดเลือกแบบทัวร์นาเมนท์ จะยกตัวอย่างโดยใช้ข้อมูลในตารางที่ 7.1 ซึ่งกำหนด q = 3 ในตารางที่ 7.1 ตัวแปร tourneylist จะเก็บรายการของประชากรในแต่ละครั้งที่มีการสลับที่ โดยสมมุติว่ามีประชากร 6 ตัว ซึ่งมีค่าฟิทเนสตามตารางที่ 7.1 ในช่องเริ่มต้น โดยตัวเลขข้างหน้าในช่องแสดงค่าฟิทเนส และตัวเลขในวงเล็บคือตัวเลขกำกับประชากร ค่าฟิทเนสดังกล่าวเมื่อนำมาแบ่งส่วนพื้นที่จะได้ตามรูปที่ 7.25 ก่อนการคัดเลือกแต่ละครั้งจะมีการ

สลับที่ใน torneylist พื้นที่แรเงาในรูปที่ 7.26 แสดงส่วนที่ถูกเลือกในการสลับที่แต่ละครั้ง โดยรูปที่ 7.26(a) - 7.26(c) คือการสลับที่ครั้งที่ 1-3 ในการสลับที่ครั้งที่ 1 ประชากร 3 และ 5 จะถูกเลือก การสลับที่ครั้งที่ 2 ประชากร 5 และ 3 จะถูกเลือก และการสลับที่ครั้งที่ 3 ประชากร 5 และ 2 จะถูกเลือก

จากวิธีดังกล่าว สามารถเขียนเป็นโฟลว์ชาร์ตสำหรับการคัดเลือกแบบทัวร์นาเมนท์ได้ตามรูปที่ 7.27 รายละเอียดของโปรแกรมที่เขียนขึ้นสามารถดูได้จากโปรแกรมย่อย select_memory , select_free , ts_selected , ts_reset และ selected ในภาคผนวก จ.

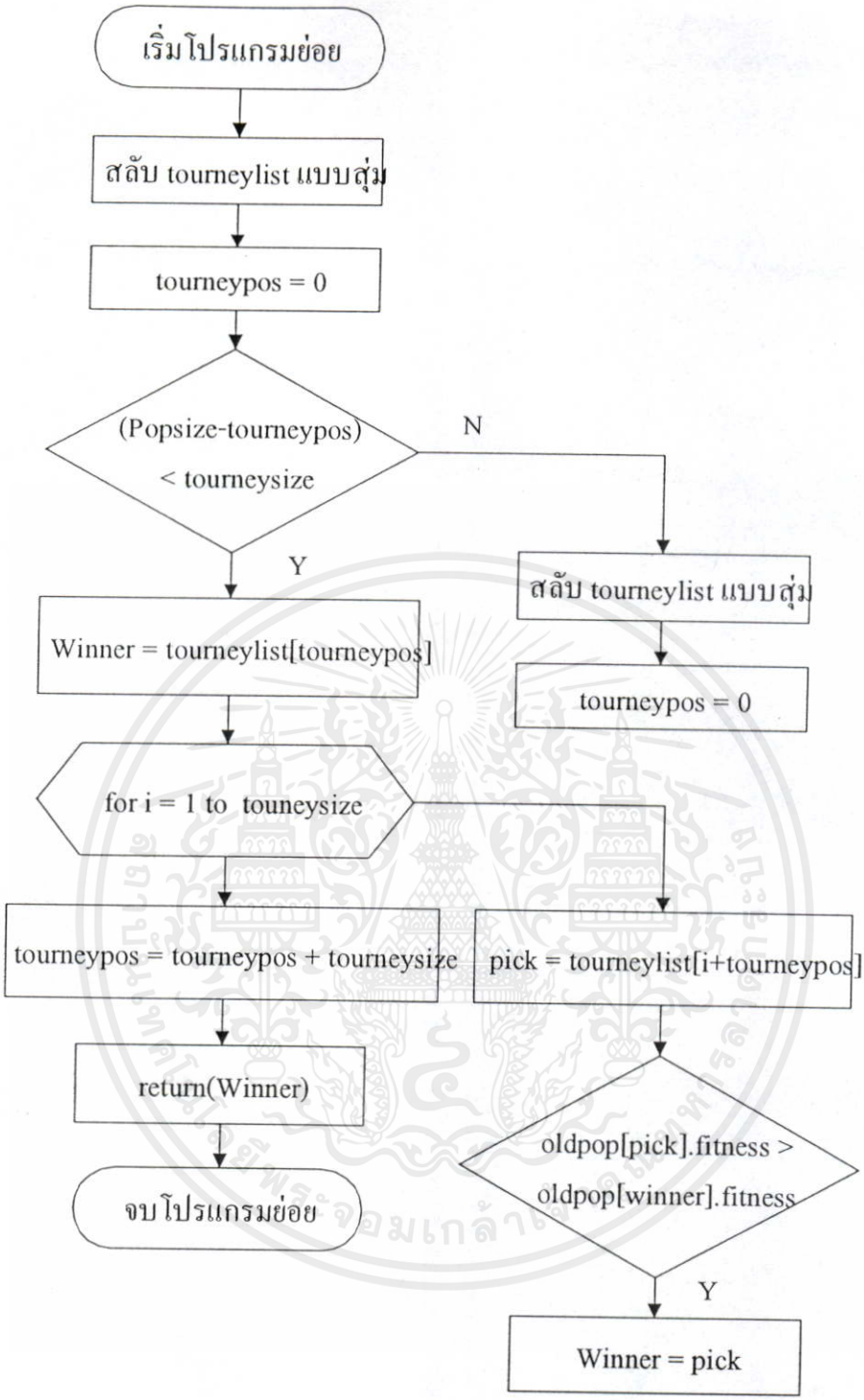
ตารางที่ **7.1** รายการใน tourneylist ของการคัดเลือกแบบทัวร์นาเมนท์

รายการ	เริ่มต้น	สลับที่ครั้งที่ 1	สลับที่ครั้งที่ 2	สลับที่ครั้งที่ 3
tourneylist[0]	1/12 (1)	2/12 (3)	1/12 (1)	4.5/12 (5)
tourneylist[1]	3/12 (2)	1/12 (4)	4.5/12 (5)	2/12 (3)
tourneylist[2]	2/12 (3)	0.5/12 (6)	3/12 (2)	1/12 (1)
tourneylist[3]	1/12 (4)	1/12 (1)	1/12 (4)	0.5/12 (6)
tourneylist[4]	4.5/12 (5)	3/12 (2)	0.5/12 (6)	3/12 (2)
tourneylist[5]	0.5/12 (6)	4.5/12 (5)	2/12 (3)	1/12 (4)



รูปที่ 7.26 การคัดเลือกแบบทัวร์นาเมนท์ในแต่ละรอบของการสลับที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



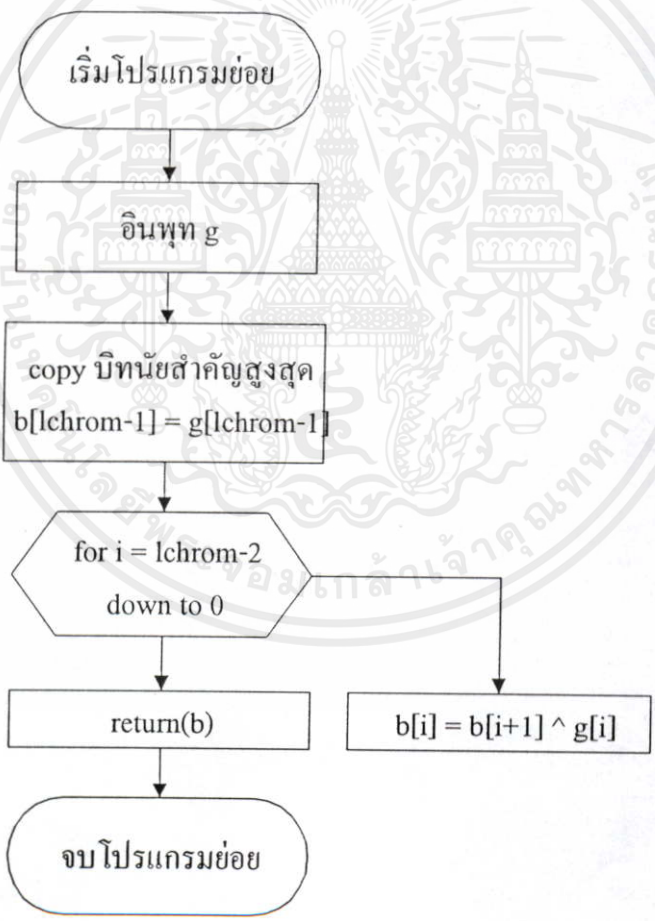
รูปที่ 7.27 โฟลว์ชาร์ตการคัดเลือกแบบทัวร์นาเมนท์

7.2.6 การเข้ารหัสแบบเกรย์ (Gray Code)

รหัสแบบเกรย์ คือ รหัสไบนารีที่มีการจัดลำดับใหม่ ให้มีตำแหน่งของบิตข้างเคียงต่างกันเพียง 1 บิต ตัวอย่างเช่น รหัสไบนารีของ [0...7] คือ { 000 , 001 , 010 , 011 , 100 , 101 , 110 , 111 } ในขณะที่รหัสแบบเกรย์ คือ { 000 , 001 , 011 , 010 , 110 , 111 , 101 , 100 } จากตัวอย่างนี้จะ

สังเกตได้ว่า การที่จะเปลี่ยนค่าจาก 0 เป็น 7 สำหรับรหัสแบบไบนารีจะต้องเปลี่ยนบิตถึง 3 ครั้ง แต่สำหรับรหัสแบบเกรย์เปลี่ยนบิตเพียง 1 ครั้งเท่านั้น ทำให้รหัสแบบเกรย์มีส่วนช่วยปรับปรุงประสิทธิภาพในการมิวเทชั่นค่าที่อยู่ข้างเคียง ซึ่งมีข้อยืนยันจาก Hollstien [9] ที่ใช้รหัสแบบเกรย์สำหรับ GA ในการแก้ปัญหาหาค่าสูงสุดของฟังก์ชั่น 2 ตัวแปร ว่าการใช้รหัสแบบเกรย์ให้ผลที่ดีกว่าการใช้รหัสแบบไบนารี

ในโปรแกรมที่เขียนขึ้น โครโมโซมแต่ละตัวจะเป็นรหัสแบบเกรย์ ดังนั้นในขั้นตอนของการถอดรหัสจะต้องมีการถอดเป็นค่าเลขฐานสิบ โดยจะมีขั้นตอนจาก แปลงรหัสแบบเกรย์ไปเป็นรหัสแบบไบนารีก่อน แล้วจึงแปลงรหัสแบบไบนารีเป็นเลขฐานสิบ การแปลงรหัสแบบเกรย์ไปเป็นรหัสแบบไบนารีมีขั้นตอนตามโฟลว์ชาร์ตรูปที่ 7.28 รายละเอียดโปรแกรมในส่วนนี้ดูได้จากโปรแกรมย่อยชื่อ objfunc ในภาคผนวก ง.



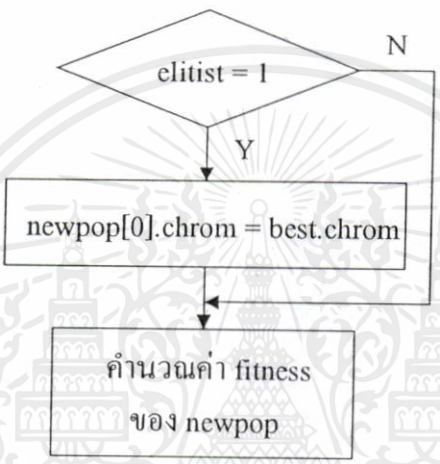
รูปที่ 7.28 โฟลว์ชาร์ตการแปลงรหัสแบบเกรย์ไปเป็นรหัสแบบไบนารี

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.2.7 อิลิทิสซึม (Elitism)

อิลิทิสซึม เป็นการรักษาโครโมโซมที่ดีที่สุดไว้ในแต่ละเจนเนอเรชั่น เพื่อเป็นหลักประกันว่าโครโมโซมที่ดีที่สุดในเจนเนอเรชั่นถัดไปจะดีกว่าเจนเนอเรชั่นก่อนหน้านี้เสมอ

ในโปรแกรมที่เขียนขึ้นจะมีการเก็บโครโมโซมที่ดีที่สุดไว้ 1 ตัว โดยจะเก็บไว้ที่ตำแหน่ง newpop[0].chrom เสมอ ตัวแปร elitist จะเป็นตัวกำหนดว่า ในโปรแกรมจะมีการเก็บโครโมโซมที่ดีที่สุดไว้หรือไม่ ซึ่งสามารถกำหนดได้ในตอนเริ่มต้นของโปรแกรม ขั้นตอนดังกล่าวเขียนเป็นโฟลว์ชาร์ตได้ดังรูปที่ 7.29 รายละเอียดโปรแกรมในส่วนนี้ดูได้จากโปรแกรมย่อยชื่อ generation ในภาคผนวก ง.



รูปที่ 7.29 โฟลว์ชาร์ตการทำอิลิทิสซึม

7.2.8 การปรับค่าความน่าจะเป็นในการครอสโอเวอร์และมิวเทชั่น

โดยปกติแล้วความน่าจะเป็นในการครอสโอเวอร์จะอยู่ในช่วง 0.6-0.9 และความน่าจะเป็นในการมิวเทชั่นจะไม่เกิน 0.1 [10] ในตอนเริ่มต้นของเจนเนอเรชั่นแรกๆนั้นต้องการความหลากหลายของประชากร ดังนั้นความน่าจะเป็นในการครอสโอเวอร์ควรจะมีค่าสูงและความน่าจะเป็นในการมิวเทชั่นควรจะมีค่าต่ำ แต่เมื่อเจนเนอเรชั่นสูงๆ ผลของการครอสโอเวอร์จะน้อยลงและอาจนำไปสู่การลู่เข้าก่อนกำหนดได้ ดังนั้นความหลากหลายในช่วงนี้สามารถปรับได้โดยเพิ่มความน่าจะเป็นในการมิวเทชั่นให้สูงขึ้น และลดความน่าจะเป็นในการครอสโอเวอร์ลง

ในโปรแกรมที่เขียนขึ้นจะปรับค่าความน่าจะเป็นในการครอสโอเวอร์ลดลงอย่างเป็นเชิงเส้นจนถึง 0.6 และความน่าจะเป็นในการมิวเทชั่นเพิ่มขึ้นอย่างเป็นเชิงเส้นจนถึง 0.1 โดยเป็นไปตามสมการ (7.28) และ (7.29) [37,38] รายละเอียดโปรแกรมในส่วนนี้ดูได้จากโปรแกรมย่อยชื่อ generation ในภาคผนวก จ.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$P_c(t) = P_c(t-1) - \left[P_c(0) - 0.6\right] / Max\ gen \tag{7.28}$$

$$P_m(t) = P_m(t-1) + \left[0.1 - P_m(0)\right] / Max\ \text{gen} \tag{7.29}$$

โดยที่

$P_c(t)$ คือ ความน่าจะเป็นสำหรับการครอสโอเวอร์ ณ เวลาใดๆ

$P_c(t-1)$ คือ ความน่าจะเป็นสำหรับการครอสโอเวอร์ ณ เวลาก่อนหน้านั้น

$P_c(0)$ คือ ความน่าจะเป็นสำหรับการครอสโอเวอร์ ณ เวลาเริ่มต้น

$P_m(t)$ คือ ความน่าจะเป็นสำหรับการมิวเทชั่น ณ เวลาใดๆ

$P_m(t-1)$ คือ ความน่าจะเป็นสำหรับการมิวเทชั่น ณ เวลาก่อนหน้านั้น

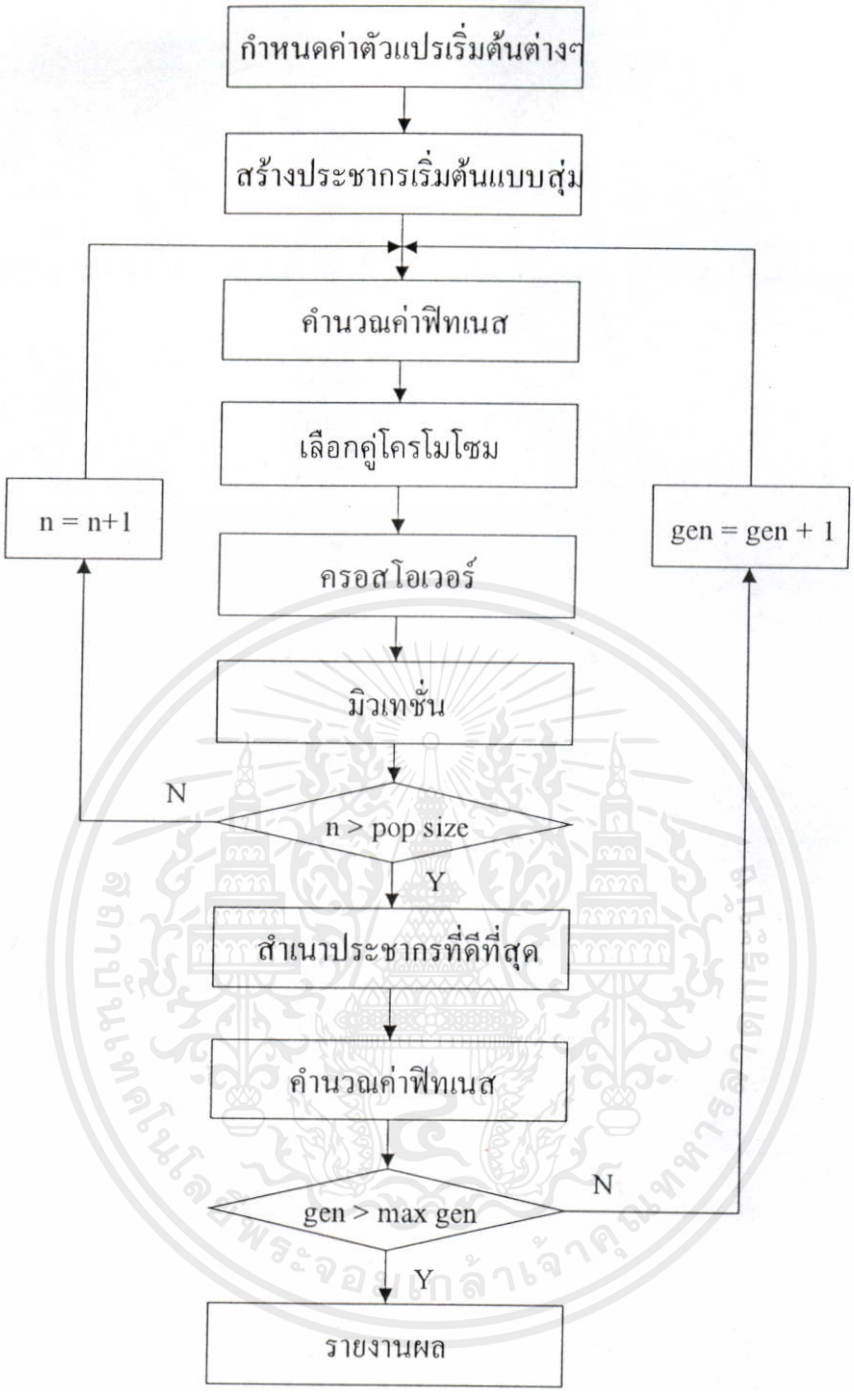
$P_m(0)$ คือ ความน่าจะเป็นสำหรับการมิวเทชั่น ณ เวลาเริ่มต้น

Max gen คือ จำนวนเจนเนอเรชั่นสูงสุด

7.2.9 โปรแกรมหลัก

โปรแกรมหลักของโปรแกรมเจเนติกอัลกอริทึมที่เขียนขึ้นแสดงในโฟลว์ชาร์ตรูปที่ 7.30

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.30 โฟลว์ชาร์ตโปรแกรมหลักโปรแกรมเจเนติกอัลกอริทึ่ม

7.3 การทดสอบโปรแกรมเจเนติกอัลกอริทึ่ม

ในหัวข้อนี้จะเป็นการทดสอบโปรแกรมเจเนติกอัลกอริทึ่มที่เขียนขึ้นว่ามีประสิทธิภาพมากแค่ไหน ซึ่งจะพิจารณาจากความถูกต้องของคำตอบและความเร็วในการเจอคำตอบ โดยจะทำการทดสอบด้วยการหาค่าสูงสุดของฟังก์ชั่นทดสอบที่เป็นฟังก์ชั่นมาตรฐานดังสมการที่ (7.30) [14] ซึ่งมีสมการดังนี้

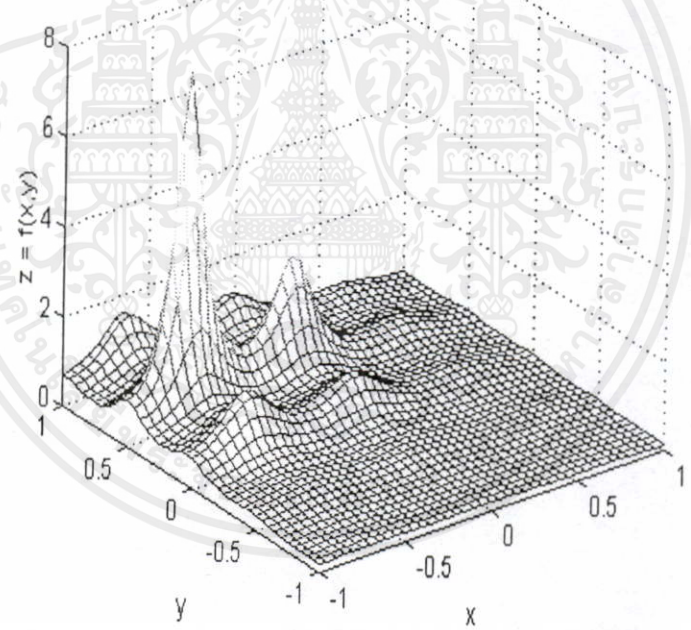
$$f(x,y) = \frac{1}{(x+0.5)^2 + 2(y-0.5)^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.8} \qquad (7.30)$$

โดยที่ $-1 \le x \le 1$, $-1 \le y \le 1$

จากสมการที่ (7.30) ฟังก์ชั่นดังกล่าวสามารถนำมาเขียนเป็นกราฟได้ดังรูปที่ 7.31 และเส้นคอนทัวร์ได้ดังรูปที่ 7.32

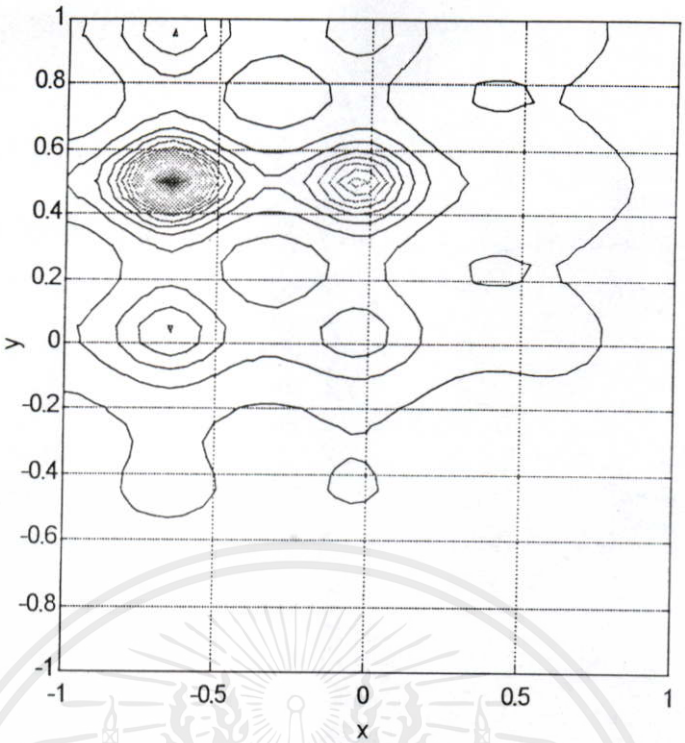
จากกราฟรูปที่ 7.31 และ 7.32 จะพบว่าค่าสูงสุดของฟังก์ชั่นทดสอบตามสมการที่ (7.30) อยู่ที่ x = -0.65500935 และ y = 0.5 โดยมีค่ายอด f(x,y) = 7.94680649 [10]

ในโปรแกรมเจเนติกอัลกอริทึ่มจะแทนตัวแปร x และ y ด้วยโครโมโซมความยาว 28 บิต ดังนั้นโครโมโซมทั้งหมดจะมีขนาด 56 บิต เจนเนอเรชั่นสูงสุดกำหนดเป็น 300 เจนเนอเรชั่น และขนาดประชากรเท่ากับ 60



รูปที่ 7.31 กราฟฟังก์ชั่นทดสอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.32 กราฟคอนทัวร์ฟังก์ชั่นทดสอบ

การทดสอบโปรแกรมเจเนติกอัลกอริทึมจะทดสอบด้วยเจเนติกอัลกอริทึม 4 วิธี คือ

1. GA1 เป็น SGA ใช้การครอสโอเวอร์แบบ 1 จุด
2. GA2 เป็น SGA ที่มีการทำสเกลลิ่งแบบเชิงเส้นและอิลิทิสซึม
3. GA3 เป็น GA ที่มีการปรับปรุง คือ ใช้การครอสโอเวอร์แบบยูนิฟอร์ม ความน่าจะเป็นในการครอสโอเวอร์ 0.8-0.6 ความน่าจะเป็นในการมิวเทชั่น 0.01-0.1 และการคัดเลือกแบบทัวร์นาเมนท์ โดยใช้ขนาดทัวร์นาเมนท์ (q) เท่ากับ 2 [37,38]
4. GA4 เป็น GA เหมือนกับ GA3 แต่ใช้ ขนาดทัวร์นาเมนท์ (q) เท่ากับ 4

ผลการทดสอบต่างๆจะทดสอบเปรียบเทียบเมื่อเข้ารหัสแบบไบนารีและแบบเกรย์ โดยใช้เจเนติกอัลกอริทึม 4 วิธีดังกล่าว ตารางที่ 7.2 -7.9 เป็นผลการทดสอบแต่ละวิธี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 7.2 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA1 เข้ารหัสแบบไบนารี

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65507386	0.49980511	69	7.94672219
2	-0.65534306	0.50010224	87	7.94668414
3	-0.65495434	0.50030612	390	7.94660503
4	-0.65481289	0.50005001	107	7.94676645
5	-0.65485177	0.50002333	75	7.94678300
6	-0.65497479	0.49995924	150	7.94680189
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94672712

ตารางที่ 7.3 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA2 เข้ารหัสแบบไบนารี

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65501062	0.49999999	490	7.94680648
2	-0.65500926	0.50000001	311	7.94680646
3	-0.65500935	0.50000000	256	7.94680649
4	0.95501184	0.49999904	496	7.94680648
5	-0.65500937	0.50000000	179	7.94680649
6	-0.65502930	0.50000000	329	7.94680613
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680642

ตารางที่ 7.4 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA3 เข้ารหัสแบบไบนารี

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65502930	0.50000000	79	7.94680613
2	-0.65502930	0.50000000	103	7.94680613
3	-0.65500935	0.50000000	85	7.94680649
4	-0.65500935	0.50000000	178	7.94680649
5	-0.65502930	0.50000000	133	7.94680613
6	-0.65500935	0.50000000	92	7.94680649
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680631

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 7.5 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA4 เข้ารหัสแบบไบนารี

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65500935	0.50000000	159	7.94680649
2	-0.65500935	0.50000000	96	7.94680649
3	-0.65501023	0.50000000	35	7.94680649
4	-0.65502930	0.50000000	178	7.94680613
5	-0.65502930	0.50000000	72	7.94680613
6	-0.65502930	0.50000000	45	7.94680613
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680631

ตารางที่ 7.6 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA1 เข้ารหัสแบบเกรย์

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65497750	0.49996946	349	7.94680360
2	-0.65474206	0.50016399	325	7.94668520
3	-0.5515238	0.50008952	437	7.94677109
4	-0.65485646	0.49941928	293	7.94607031
5	-0.65509286	0.49956404	85	7.94639716
6	-0.63556537	0.50024890	111	7.94636617
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94651559

ตารางที่ 7.7 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA2 เข้ารหัสแบบเกรย์

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65500935	0.50000000	459	7.94680649
2	-0.65500935	0.49999998	446	7.94680649
3	-0.65500926	0.49999988	469	7.94680649
4	-0.65500936	0.49999998	459	7.94680649
5	-0.65500934	0.49999998	473	7.94680649
6	-0.65500940	0.49999998	498	7.94680649
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680649

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 7.8 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA3 เข้ารหัสแบบเกรย์

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65500935	0.50000000	136	7.94680649
2	-0.65500935	0.50000000	121	7.94680649
3	-0.65500935	0.50000000	175	7.94680649
4	-0.65500935	0.50000000	108	7.94680649
5	-0.65500935	0.50000000	115	7.94680649
6	-0.65500935	0.50000000	135	7.94680649
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680649

ตารางที่ 7.9 ผลการแก้ปัญหาฟังก์ชั่นทดสอบด้วย GA4 เข้ารหัสแบบเกรย์

ครั้งที่	x	y	Gen	ฟิทเนส
1	-0.65500935	0.50000000	59	7.94680649
2	-0.65500935	0.50000000	63	7.94680649
3	-0.65500935	0.50000000	54	7.94680649
4	-0.65500935	0.50000000	63	7.94680649
5	-0.65500935	0.50000000	68	7.94680649
6	-0.65500935	0.50000000	71	7.94680649
ค่าเฉลี่ยฟิทเนส 6 ครั้ง				7.94680649

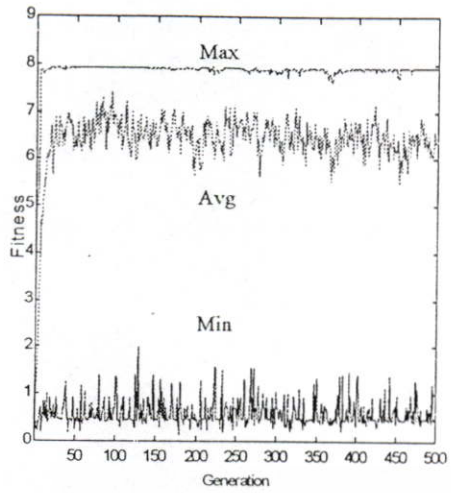
ผลการแก้ปัญหาฟังก์ชั่นทดสอบโดยแสดงค่า Max , Min , Avg ในแต่ละเจนเนอเรชั่น จนถึงเจนเนอเรชั่นสูงสุด แสดงในรูปที่ 7.33 -7.40

จากผลการทดสอบในตารางที่ 7.2-7.9 และรูปที่ 7.33 -7.40 จะพบว่า GA1 ซึ่งเป็น SGA ให้คำตอบยังไม่ดี แต่เมื่อมีการปรับปรุงเจเนติกเป็น GA2 คำตอบที่ได้จะดีขึ้น และเมื่อใช้ GA3 และ GA4 คำตอบที่ได้ก็ดียิ่งขึ้นไปอีก โดย GA3 และ GA4 จะต่างกันที่เวลาในการลู่เข้าของคำตอบ ซึ่งเป็นการแสดงข้อดีของการใช้การคัดเลือกแบบทัวร์นาเมนท์ คือ สามารถปรับพฤติกรรมการคัดเลือกได้จากขนาดทัวร์นาเมนท์ (ขนาดทัวร์นาเมนท์น้อยจะลู่เข้าคำตอบช้า ขนาดทัวร์นาเมนท์มากจะลู่เข้าเข้าตอบได้เร็ว แต่มีความเสี่ยงสูงในการเกิดการลู่เข้าก่อนกำหนด) และจากการเปลี่ยนวิธีการเข้ารหัสจากไบนารีเป็นเกรย์ จะพบว่าคำตอบที่ได้จากการเข้ารหัสแบบเกรย์ดีกว่าการเข้ารหัสแบบไบนารี โดยเฉพาะเมื่อใช้ร่วมกับ GA3 หรือ GA4 จะให้คำตอบที่ถูกต้อง 100%

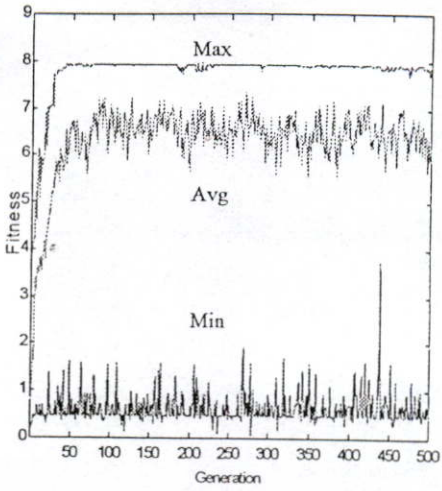
7.3.1 สรุป

จากผลการทดสอบโปรแกรมเจเนติกอัลกอริทึมดังกล่าว พบว่าการใช้ GA3 หรือ GA4 ร่วมกับการเข้ารหัสแบบเกรย์ให้คำตอบที่ดีที่สุด โดยที่ GA4 จะลู่เข้าคำตอบเร็วกว่า GA3 แต่ก็มีความเสี่ยงต่อการเกิดการลู่เข้าก่อนกำหนด โดยปกติแล้วขนาดของทัวร์นาเมนท์จะกำหนดไว้เท่ากับ 2 เพื่อลดความเสี่ยงต่อการเกิดการลู่เข้าก่อนกำหนด ยกเว้นบางกรณีที่ต้องการความเร็วแต่ไม่ต้องการคำตอบที่ถูกต้องมากนัก หรือต้องการรู้ย่านคำตอบ การปรับพฤติกรรมการคัดเลือกก็สามารถทำได้โดยการปรับขนาดทัวร์นาเมนท์ให้สูงขึ้น

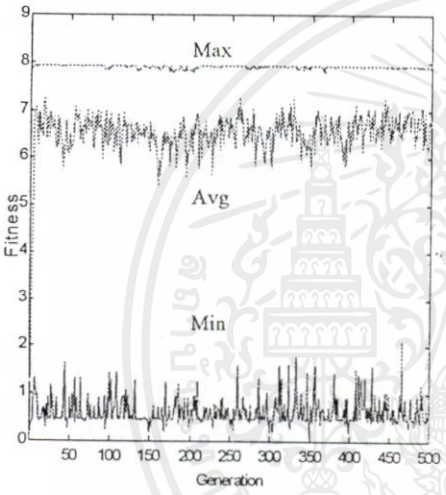
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



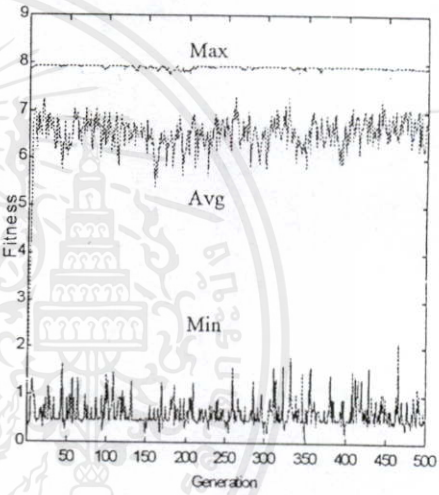
(1)



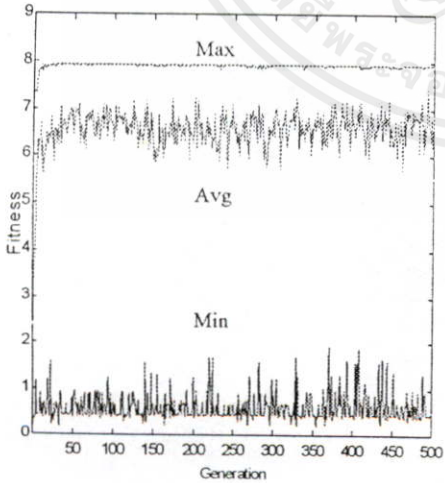
(2)



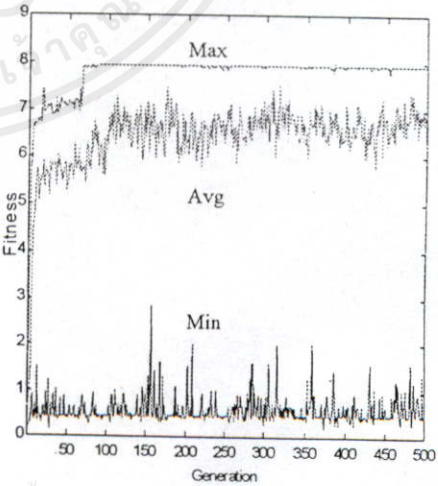
(3)



(4)



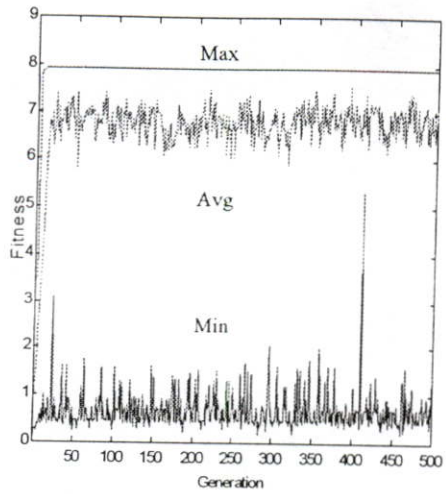
(5)



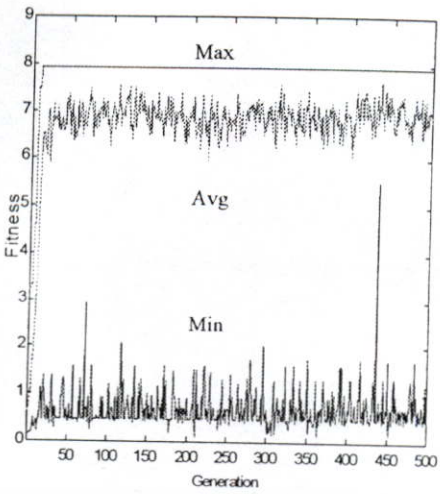
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

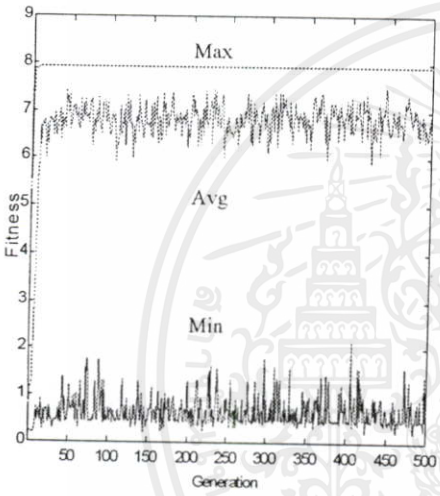
รูปที่ 7.33 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA1 เข้ารหัสแบบไบนารี



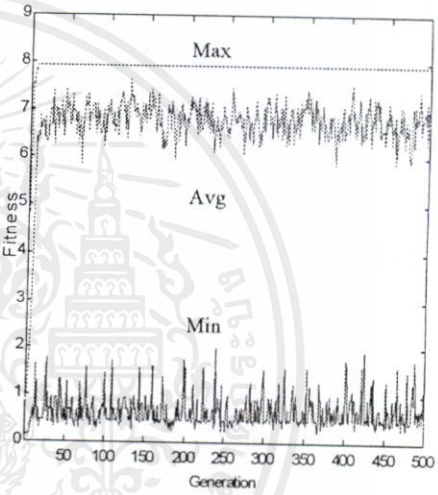
(1)



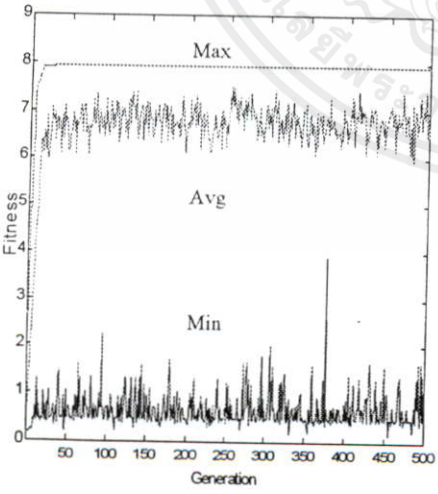
(2)



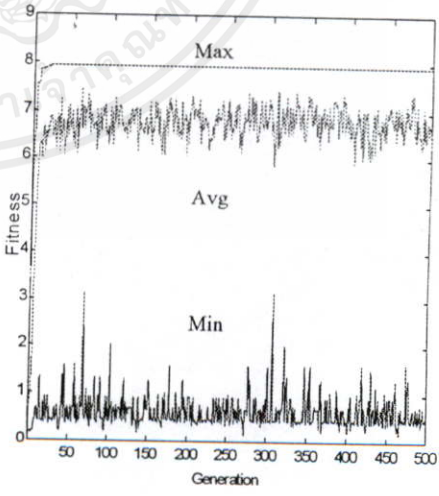
(3)



(4)



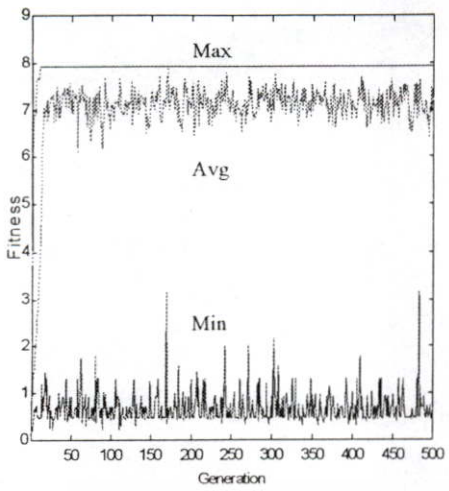
(5)



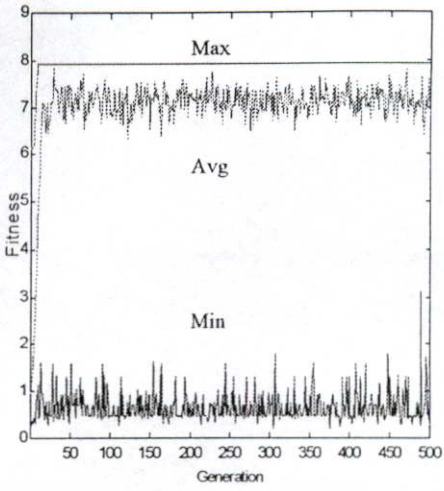
(6)

รูปที่ 7.34 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA2 เข้ารหัสแบบไบนารี

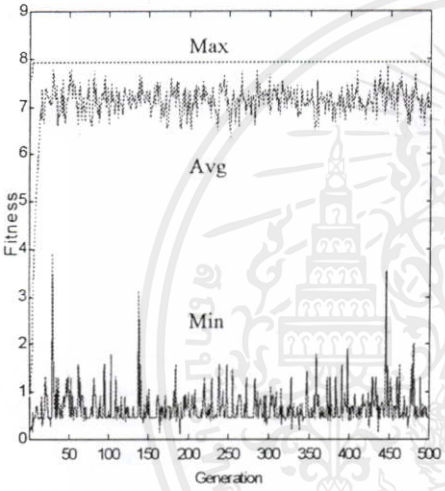
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



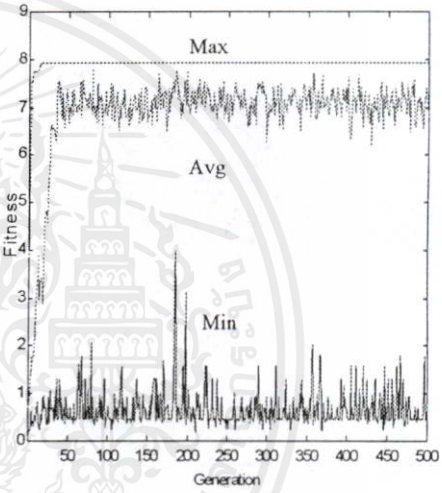
(1)



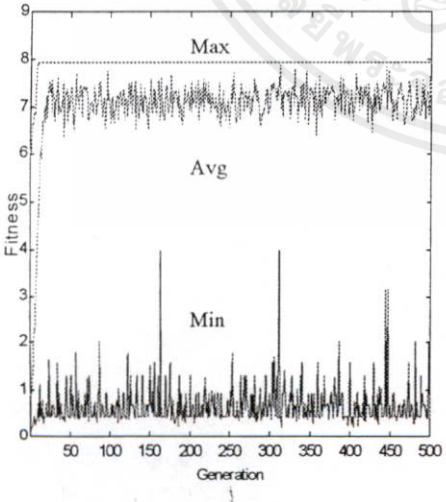
(2)



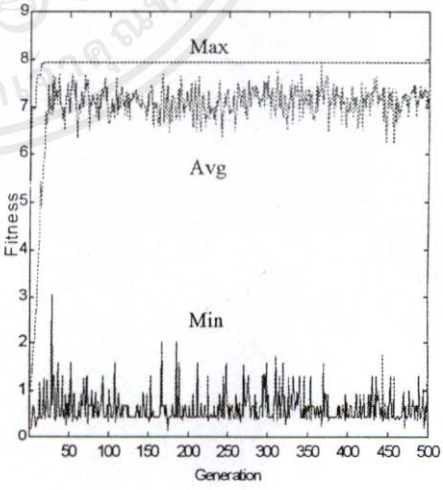
(3)



(4)



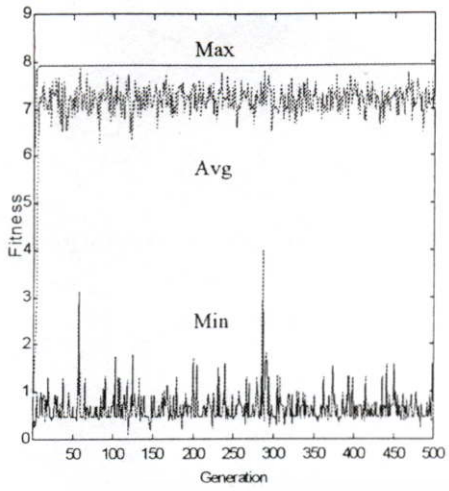
(5)



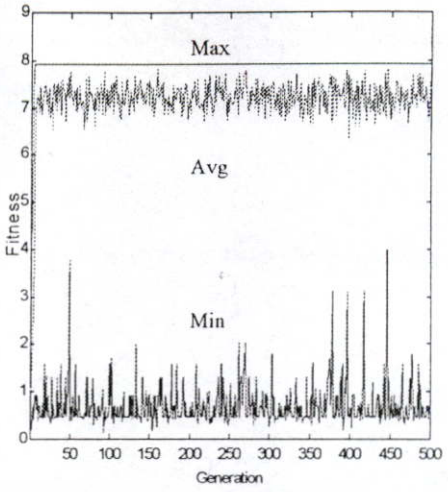
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

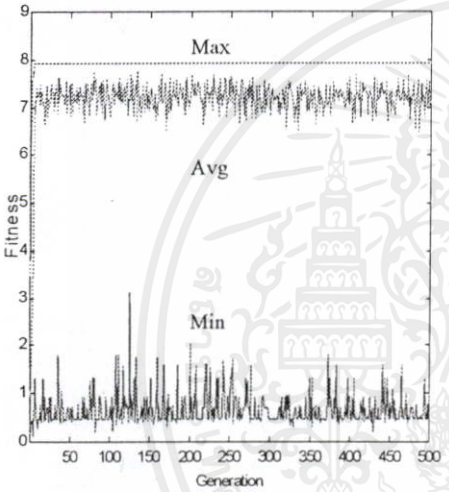
รูปที่ 7.35 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA3 เข้ารหัสแบบไบนารี



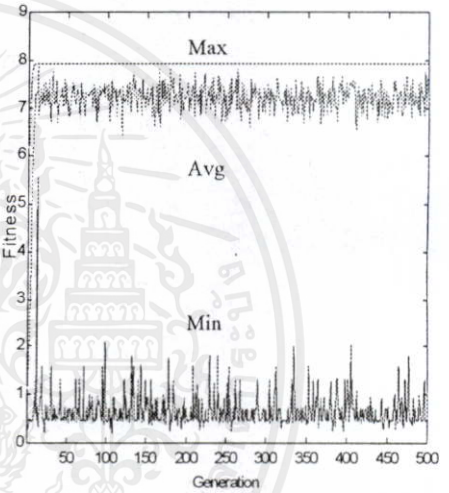
(1)



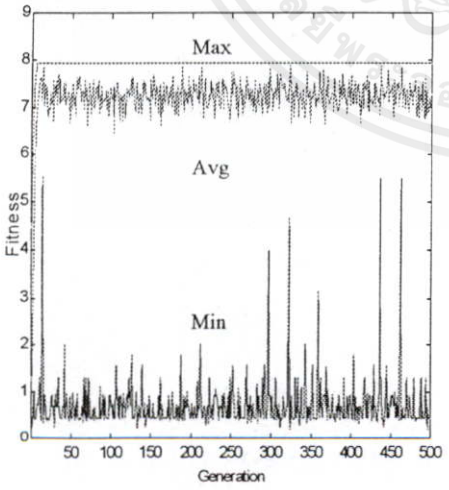
(2)



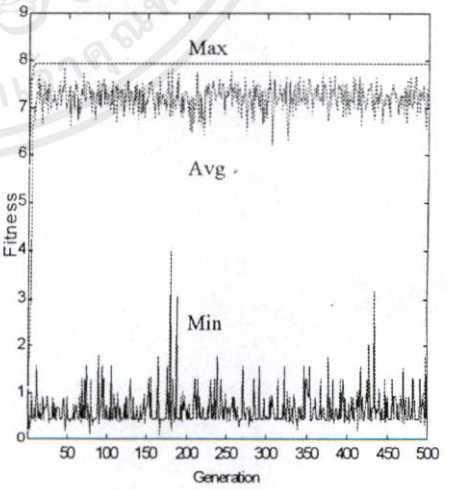
(3)



(4)



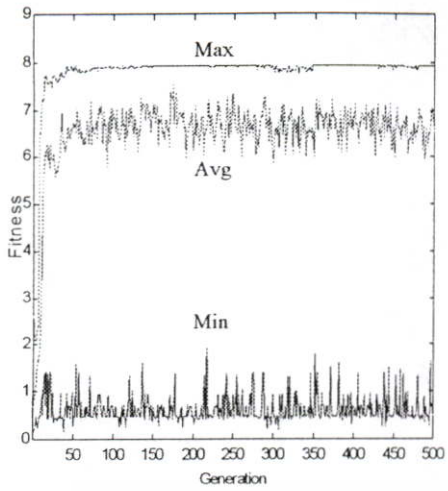
(5)



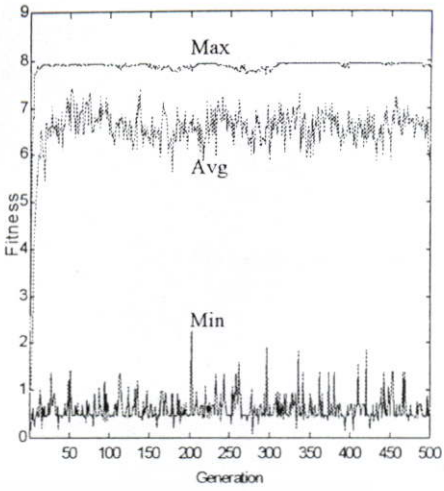
(6)

รูปที่ 7.36 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA4 เข้ารหัสแบบไบนารี

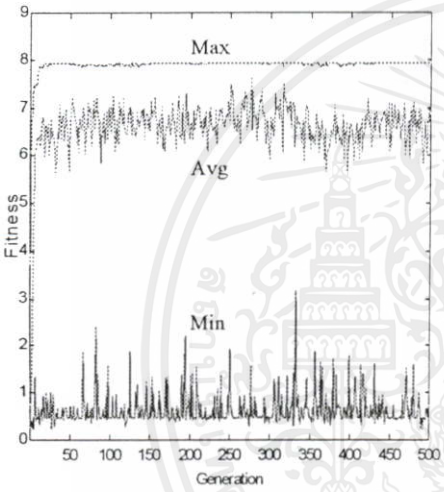
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารนี้ทุกครั้งที่มีการนำไปใช้



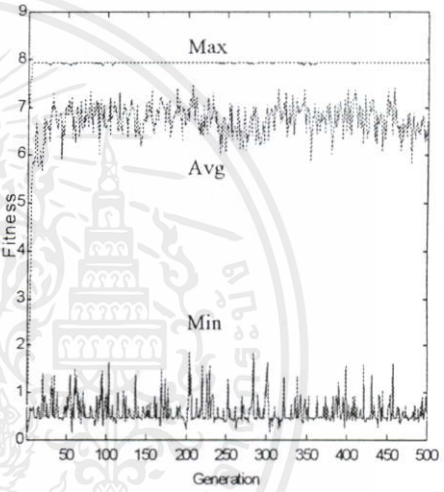
(1)



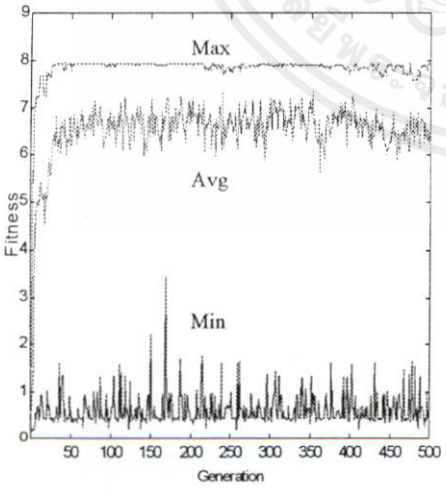
(2)



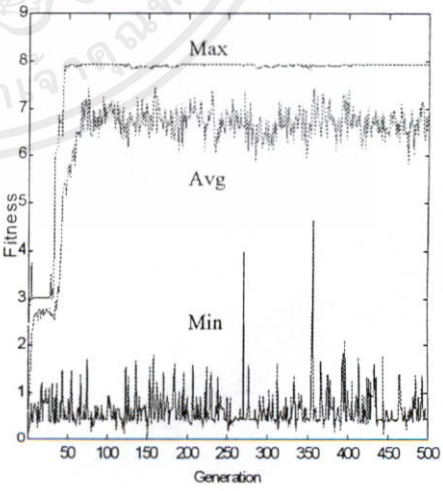
(3)



(4)



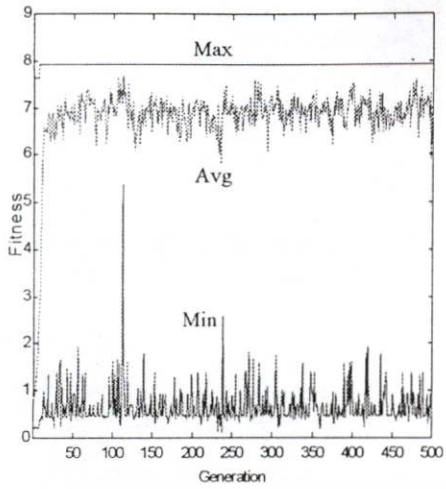
(5)



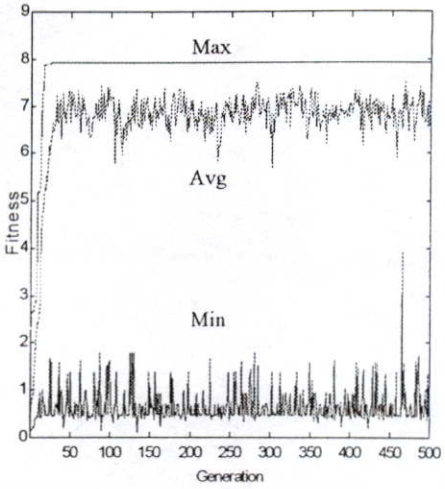
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

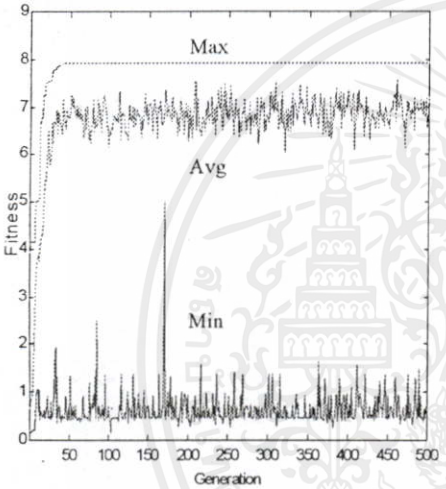
รูปที่ 7.37 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA1 เข้ารหัสแบบเกรย์



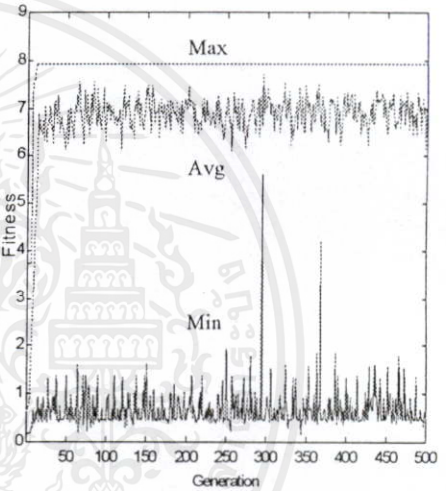
(1)



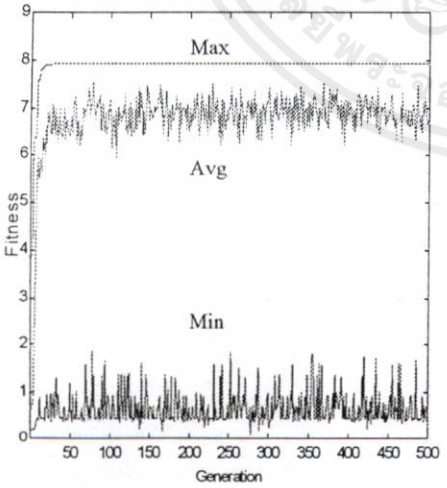
(2)



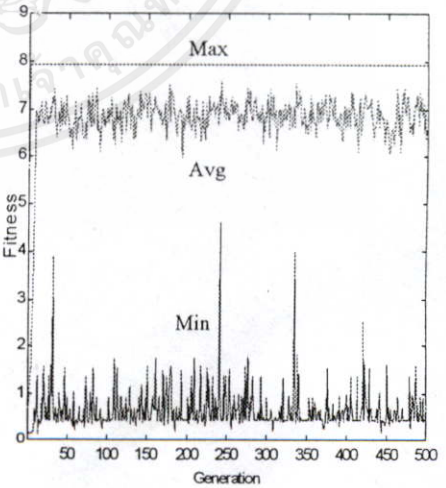
(3)



(4)



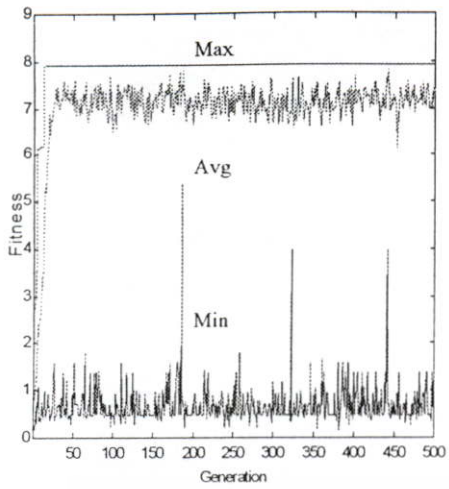
(5)



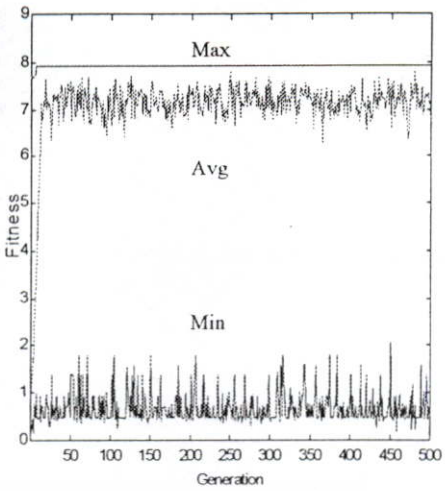
(6)

รูปที่ 7.38 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA2 เข้ารหัสแบบเกรย์

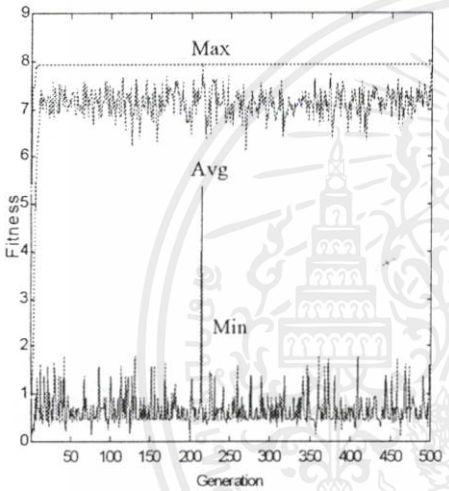
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารนี้ทุกครั้งที่มีการนำไปใช้



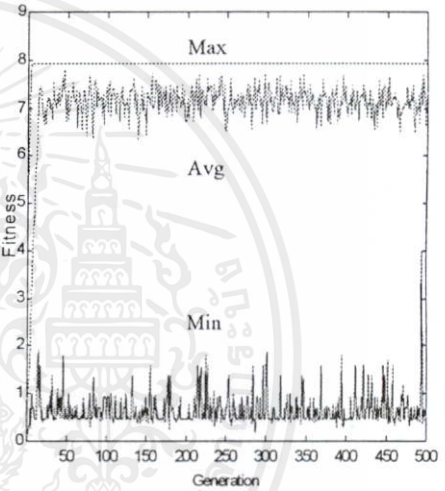
(1)



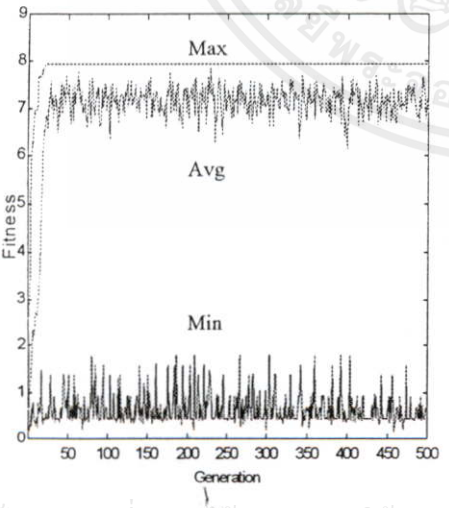
(2)



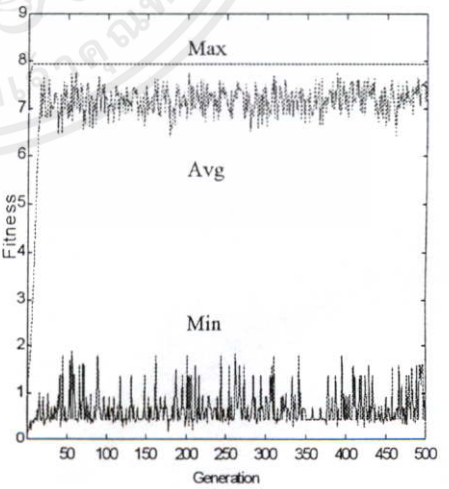
(3)



(4)



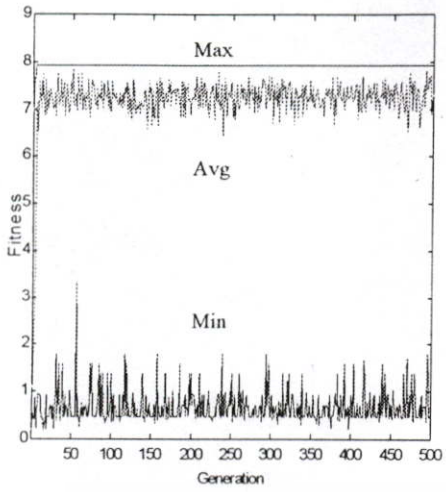
(5)



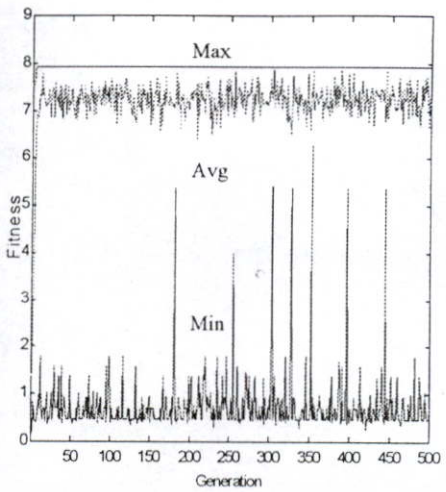
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

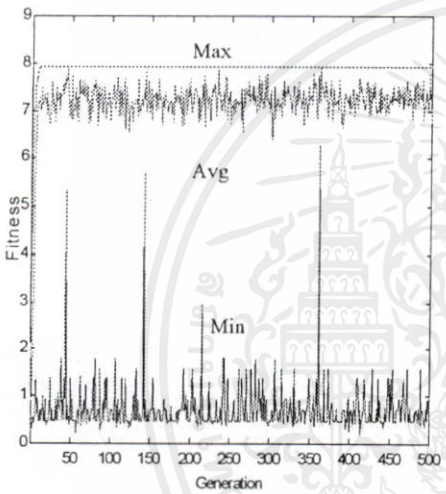
รูปที่ 7.39 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA3 เข้ารหัสแบบเกรย์



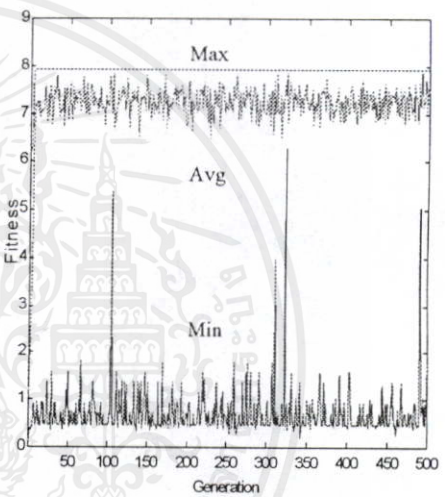
(1)



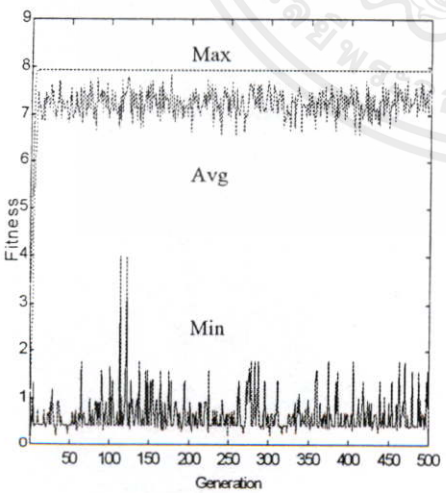
(2)



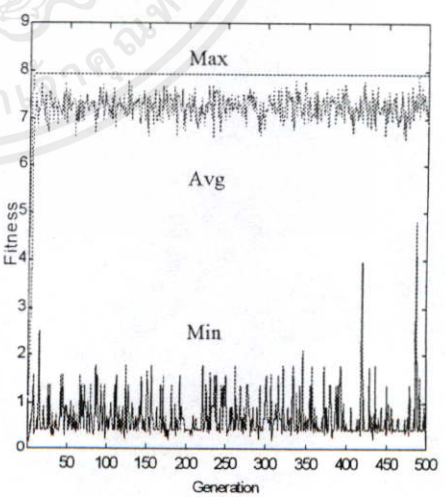
(3)



(4)



(5)



(6)

รูปที่ 7.40 ผลการแก้ปัญหาฟังก์ชั่นทดสอบเมื่อใช้ GA4 เข้ารหัสแบบเกรย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆ ทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารนี้ทุกครั้งที่มีการนำไปใช้

7.4 การออกแบบโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่

โปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่ที่เขียนขึ้นมี 2 วิธี คือ วิธีไอเทอร์เรชั่น และวิธีเจเนติกอัลกอริทึ่ม โดยทั้ง 2 วิธีจะดัดแปลงโปรแกรมหลักของโปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่ และโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน เป็นฟังก์ชั่นที่คืนค่าดัชนีทำงานของแต่ละวิธีควบคุม โดยฟังก์ชั่นนี้ชื่อว่า performance_index รายละเอียดสำหรับฟังก์ชั่นนี้ดูได้จากภาคผนวก ง.

รายละเอียดสำหรับโปรแกรมออกแบบทั้ง 2 วิธี มีดังนี้

1. วิธีไอเทอร์เรชั่น

วิธีนี้จะตรวจสอบข้อมูลทุกค่าเพื่อหาคำตอบที่ดีที่สุด จึงมีข้อจำกัดในการกำหนดช่วงการเพิ่มของข้อมูลแต่ละตัว โปรแกรมที่เขียนขึ้นมีการกำหนดช่วงการเพิ่มของข้อมูลแต่ละตัวดังนี้

- $\alpha 1$ มีค่า 0.0-1.0 ข้อมูลเพิ่มครั้งละ 0.01
- $\alpha 2$ มีค่า 0.0-90.0 ข้อมูลเพิ่มครั้งละ 1.0
- $\alpha 3$ มีค่า 0.0-1.0 ข้อมูลเพิ่มครั้งละ 0.10

ดังนั้นข้อมูลทั้งหมดที่ต้องตรวจสอบ คือ 101×91×101 = 928,291 ข้อมูล โดยโปรแกรมที่เขียนขึ้นใช้เวลาในการตรวจสอบข้อมูล 1 ตัวประมาณ 1 วินาที ดังนั้นเวลาที่ต้องใช้ คือ 928,291/3,600 = 257.859 ชั่วโมง วิธีไอเทอร์เรชั่นนี้สามารถเขียนเป็นโฟลว์ชาร์ตได้ดังรูปที่ 7.41 รายละเอียดโปรแกรมนี้ดูได้จากโปรแกรม it_m.c ในภาคผนวก ฉ.

2. วิธีเจเนติกอัลกอริทึ่ม

โปรแกรมที่เขียนขึ้นนี้ เรียกว่า โปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC โดยวิธีเจเนติกอัลกอริทึ่ม (Genetic Aided Fuzzy Controller Design : GAFCD) ซึ่งมีโปรแกรมหลักเหมือนกับโฟลว์ชาร์ตรูปที่ 7.31 แต่มีบางส่วนของโปรแกรมที่ต้องมีการดัดแปลง คือ การคำนวณหาความยาวโครโมโซม และการถอดรหัสจากโครโมโซม การคำนวณค่าฟิทเนส

- การคำนวณความยาวโครโมโซม

การคำนวณความยาวโครโมโซมจะใช้สมการ (7.31)

$$L \geq \frac{\ln\left[\frac{\alpha_{max} - \alpha_{min}}{\text{step size}}\right]}{\ln 2} \qquad (7.31)$$

เมื่อ L คือ ความยาวสตริง α_{max} คือ ค่าขอบเขตบน

α_{min} คือ ค่าขอบเขตล่าง step size คือ ความละเอียดต่อช่วง เช่น 0.01

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมที่เขียนขึ้นมี step size ของ $\alpha_1 = 0.001$, $\alpha_2 = 1.0$, $\alpha_3 = 0.01$ ความยาวสตริงของแต่ละจีนจึงเป็น 10,7,7 ตามลำดับ ดังนั้นโครโมโซมจึงมีความยาวเท่ากับ 24 บิต

- การถอดรหัสจากโครโมโซม

การถอดรหัสโครโมโซมจะแปลงรหัสเกรย์เป็นรหัสไบนารีก่อนแล้วใช้สมการที่ (7.32) และ (7.33) แปลงรหัสไบนารีเป็นเลขฐานสิบ โดยแบ่งเป็น 2 กรณี

<u>กรณี 1</u> การเพิ่มแต่ละขั้นเป็นทศนิยม เช่น 0.001

$$\alpha_i = (\text{double})[(\text{int})(X_i(10^P)/2^L)]/10^P \tag{7.32}$$

<u>กรณี 2</u> การเพิ่มแต่ละขั้นเป็นจำนวนเต็ม เช่น 1.0

$$\alpha_i = (\text{double})[(\text{int})(X_i(\alpha_{i\,max}-\alpha_{i\,min})/2^L)] \tag{7.33}$$

สมการที่ (7.32) และ (7.33) จะต้องอยู่ภายใต้เงื่อนไข $2^L \geq 10^P$

โดยที่ i = 1,2,3

X_i คือ ค่าฐานสิบที่ถอดรหัสจากเลขไบนารี่ P คือ ตำแหน่งความละเอียดทศนิยม

- การคำนวณค่าฟิทเนส

การคำนวณค่าฟิทเนสมีขั้นตอนดังนี้ คือ ถอดรหัสโครโมโซมเป็นเลขฐานสิบ และคำนวณค่าดัชนีทำงานจากฟังก์ชั่น performance_index ซึ่งเป็นไปตามสมการที่ (7.34) หลังจากนั้นค่าฟิทเนสหาได้จากสมการที่ (7.35) โดยที่สมการที่ (7.34) ดัดแปลงมาจาก [4] และสมการที่ (7.35) เป็นการแปลงปัญหาการหาค่าต่ำสุดในสมการที่ (7.34) เป็นปัญหาการหาค่าสูงสุดในสมการที่ (7.35)

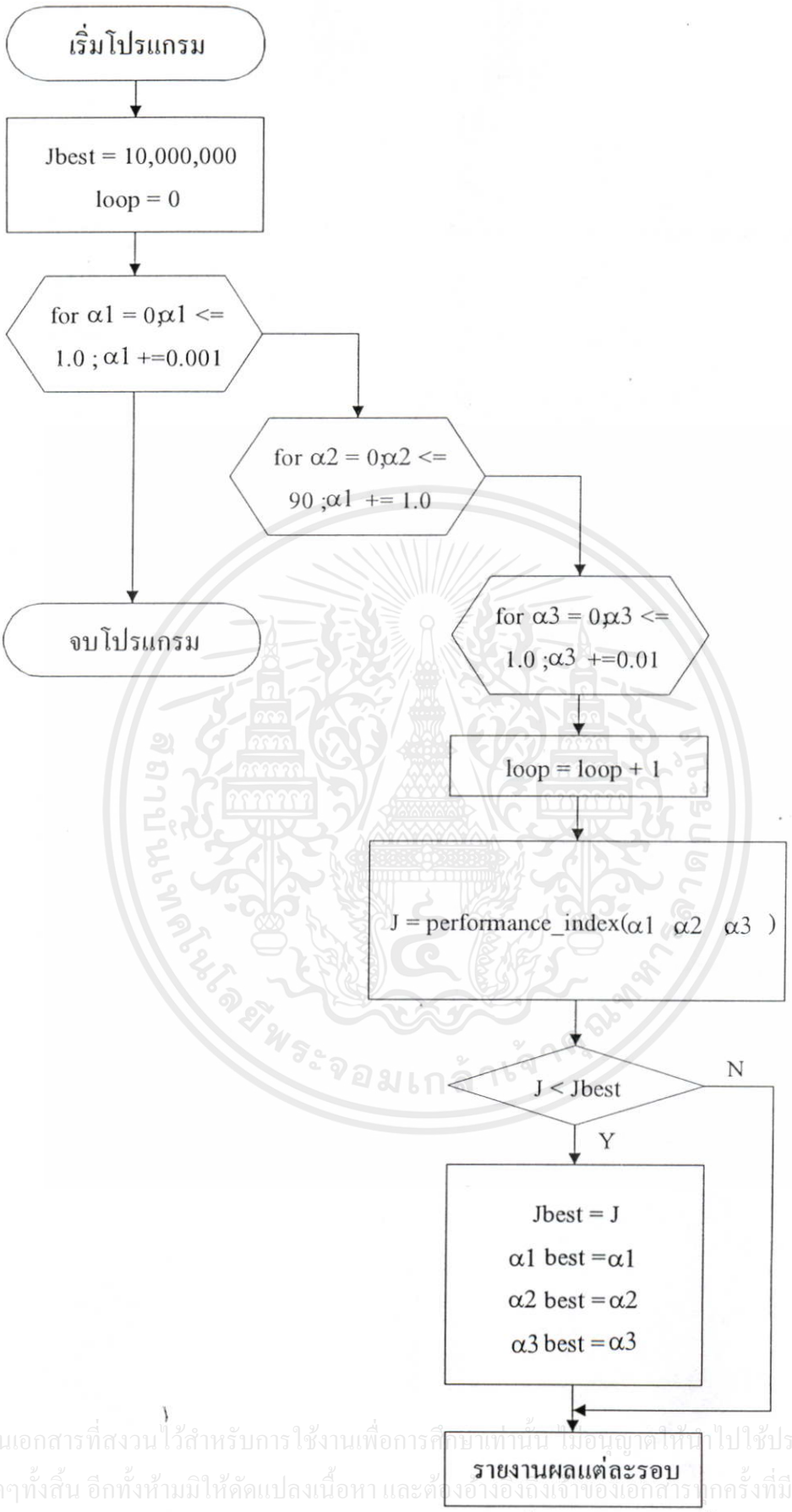
$$J = \sum_{i=1}^{2}(\beta-1)i + (2-\beta)\left(\sum_{k=1}^{N}\left(|\Delta\omega(k)|\,t_k + |\dot{U(k)}|\right)\right) \tag{7.34}$$

$$F = \rho \Big/ \left(\sum_{i=1}^{2}(\beta-1)i + (2-\beta)\left(\sum_{k=1}^{N}\left(|\Delta\omega(k)|t_k + |\dot{U(k)}|\right)\right)\right) \tag{7.35}$$

โดยที่ i = 1 คือ การรบกวนแบบรุนแรง i = 2 คือ การรบกวนแบบเล็กน้อย

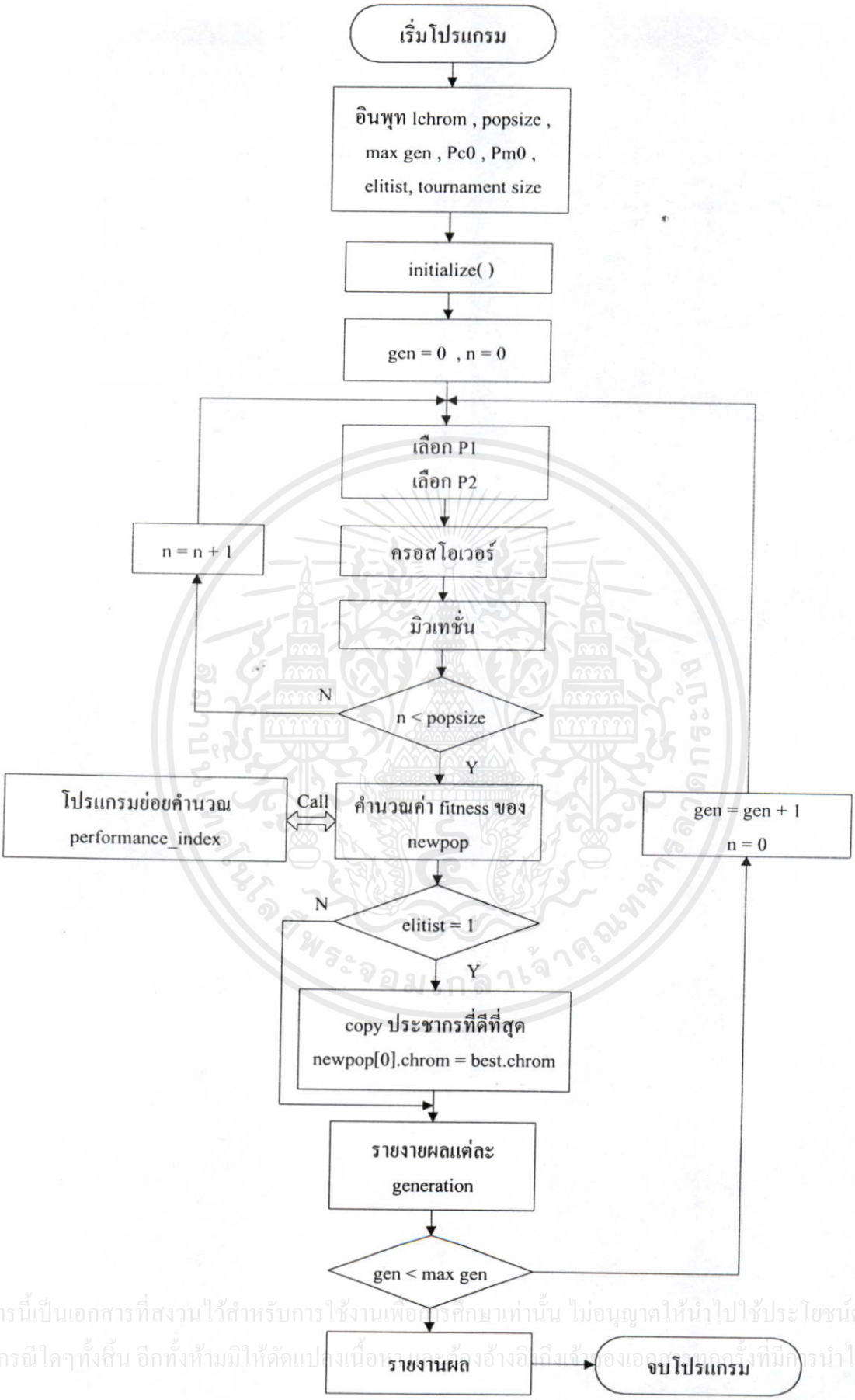
ρ คือ แฟกเตอร์ปรับสเกล β คือ ค่าคงที่ช่วง [0,1]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7.41 โฟลว์ชาร์ตโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่ ด้วยวิธีไอเทอร์เรชั่น



รูปที่ 7.42 โฟลว์ชาร์ตโปรแกรมช่วยออกแบบ SVC ชนิดฟัซซี่ ด้วยวิธีเจเนติกอัลกอริทึม (GAFCD)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ในกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียดในส่วนของขั้นตอนการถอดรหัสจากโครโมโซมและคำนวณค่าฟิทเนส ดูได้จากโปรแกรมในภาคผนวก ฉ. ในโปรแกรมย่อยชื่อ objfunc โฟลว์ชาร์ตโปรแกรม GAFCD แสดงในรูปที่ 7.42 รายละเอียดโปรแกรมหลักของ GAFCD ดูได้จากโปรแกรมชื่อ gafcd.c ในภาคผนวก ฉ.

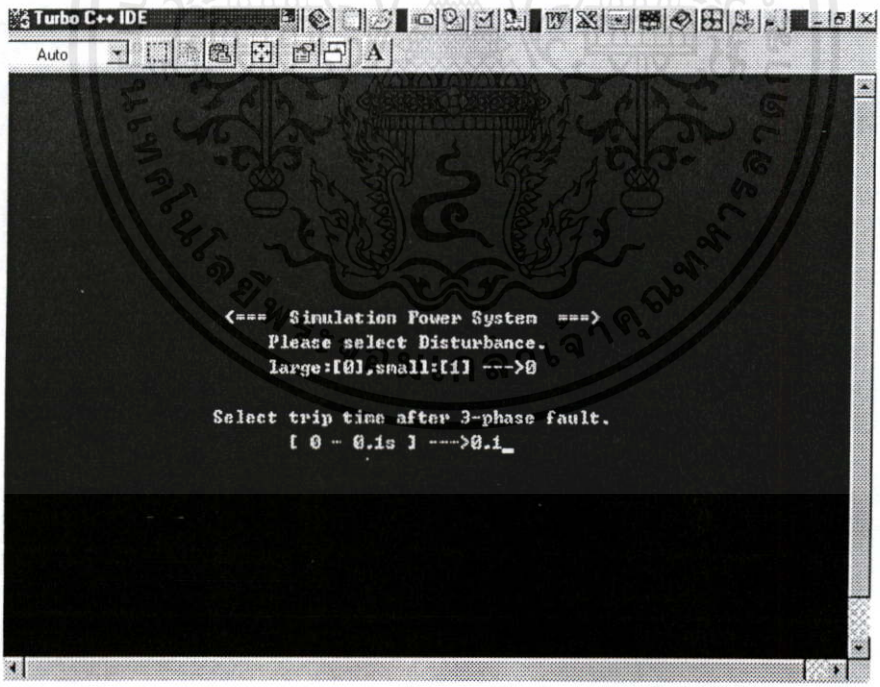
7.5 การใช้งานโปรแกรม

ในหัวข้อนี้จะอธิบายวิธีการใช้โปรแกรมต่างๆ ดังต่อไปนี้

7.5.1 โปรแกรมจำลองระบบกรณีควบคุมด้วย PSS

กรณีนี้ได้แก่ โปรแกรมจำลองระบบกรณีควบคุมด้วย ΔP-PSS โปรแกรมจำลองระบบกรณีควบคุมด้วย $\Delta\omega$-PSS และโปรแกรมจำลองระบบกรณีควบคุมด้วย $\Delta\omega$-PSS ชนิดฟัซซี่ การใช้งานโปรแกรมทั้งสามมีลักษณะเหมือนกัน คือ เลือกชนิดของการรบกวน ดังแสดงในรูปที่ 7.43

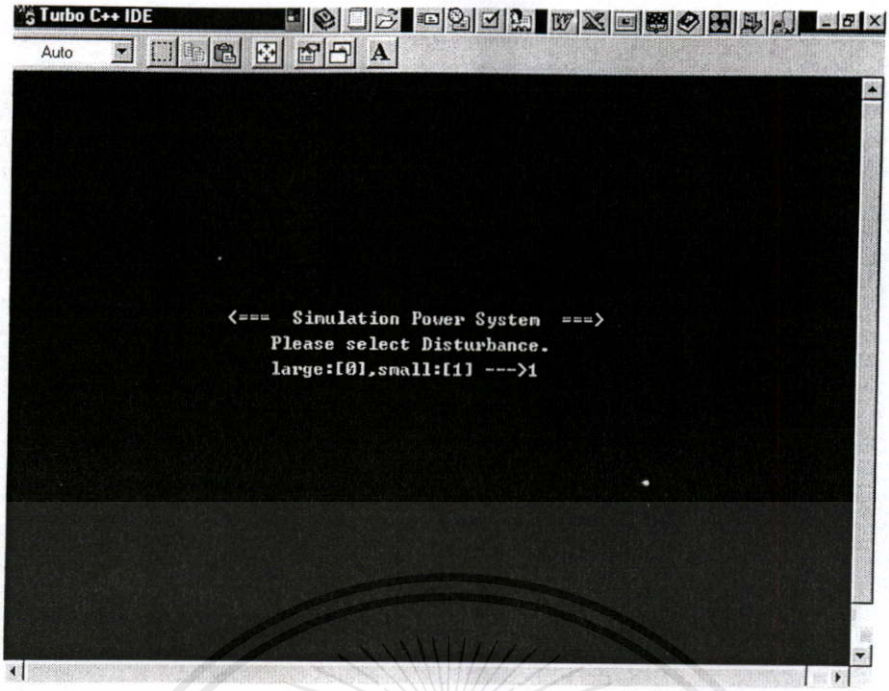
ตัวอย่างหน้าจออินพุทและเอาท์พุทของโปรแกรมจำลองระบบกรณีควบคุมด้วย PSS แสดงในรูปที่ 7.43 และ 7.44 โดย (a) เป็นการรบกวนแบบรุนแรง และ (b) คือการรบกวนแบบเล็กน้อย



(a)

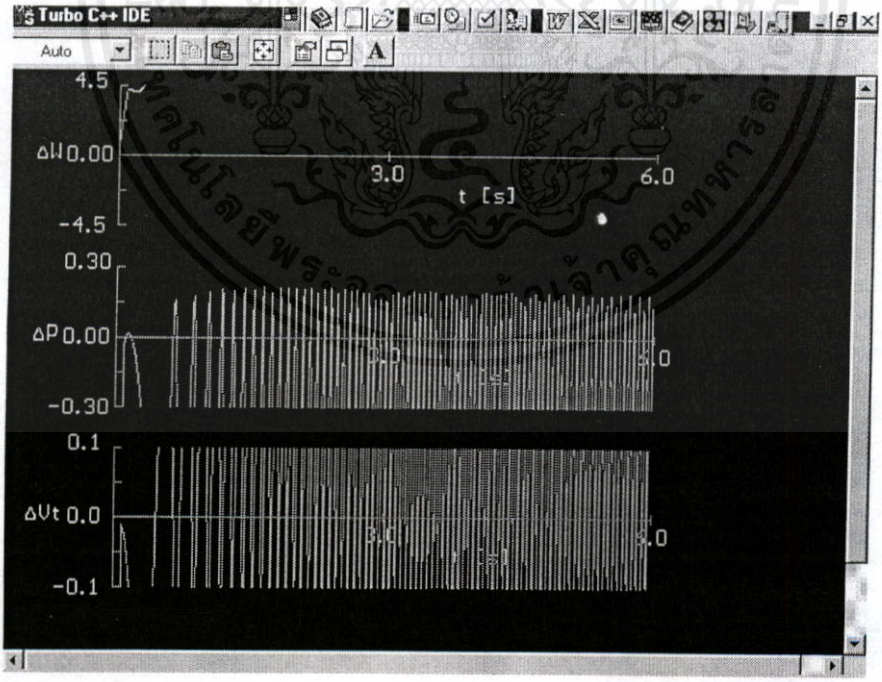
รูปที่ 7.43 ตัวอย่างหน้าจออินพุทสำหรับเลือกชนิดของการรบกวน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(b)

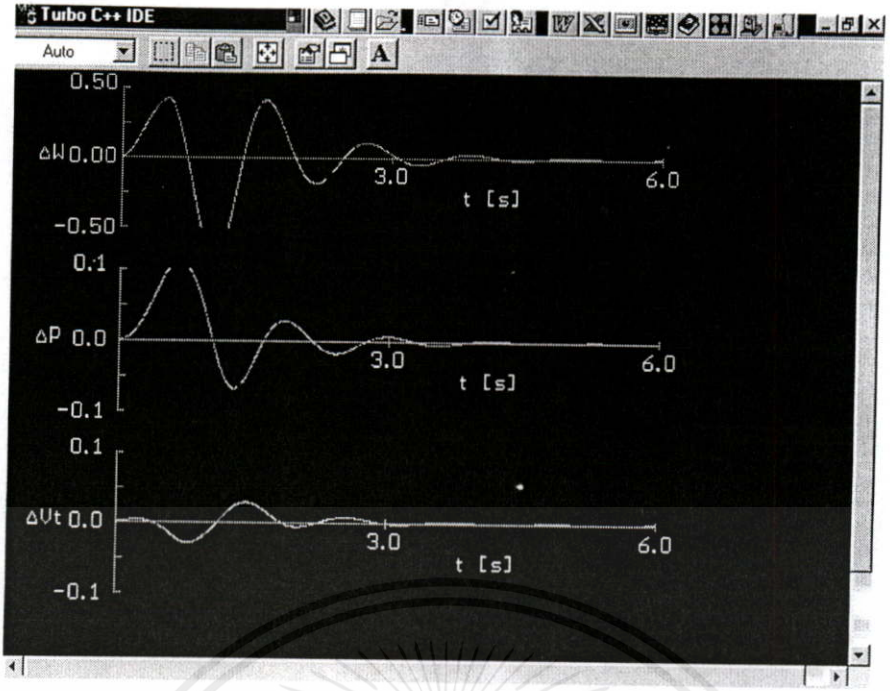
รูปที่ 7.43 (ต่อ)



(a)

รูปที่ 7.44 ตัวอย่างหน้าจอเอาท์พุทสำหรับโปรแกรมจำลองระบบกรณีควบคุมด้วย PSS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(b)

รูปที่ 7.44 (ต่อ)

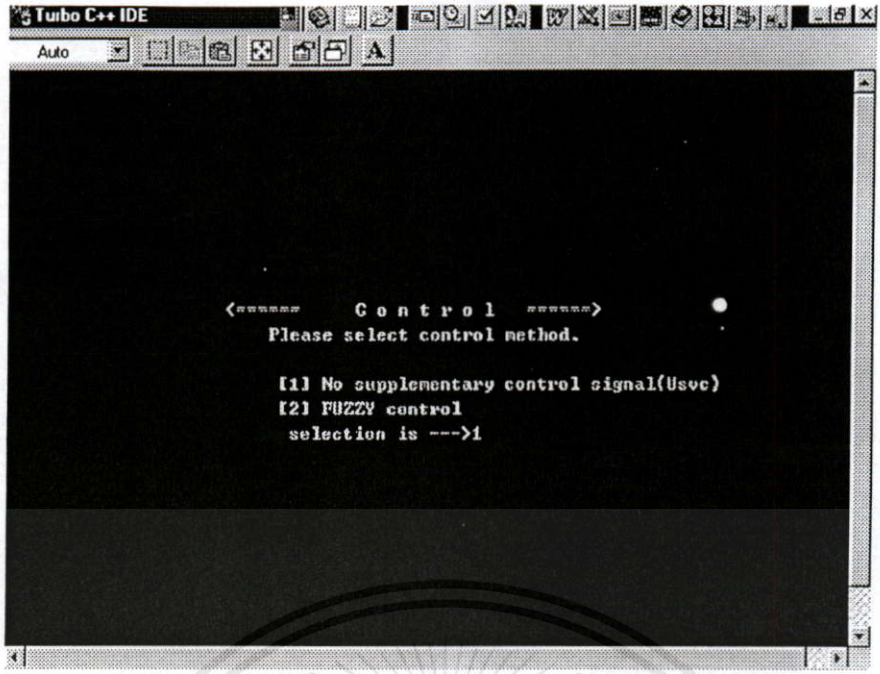
7.5.2 โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่

การใช้งานโปรแกรมนี้มีขั้นตอนต่อไปนี้

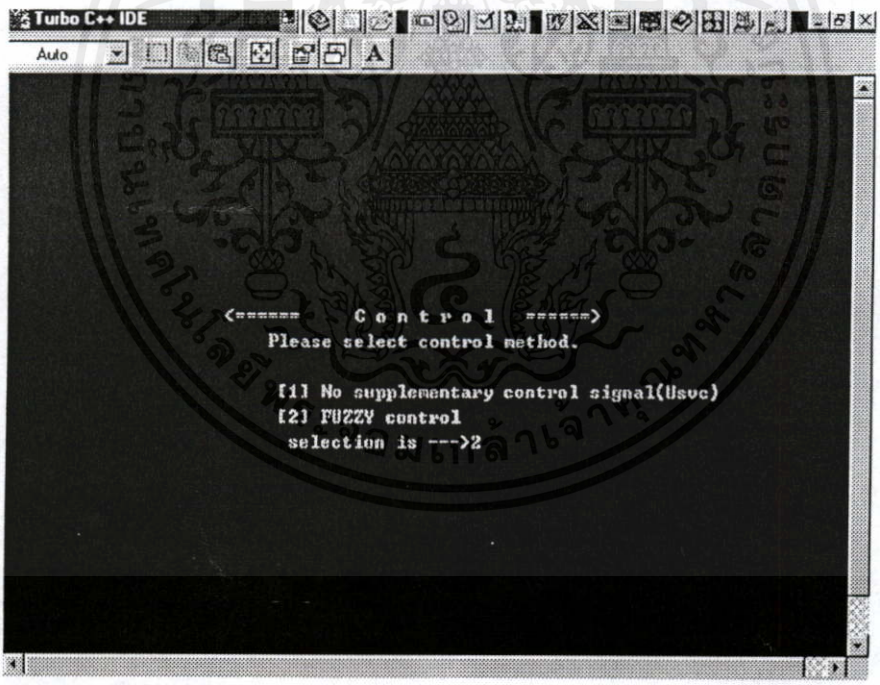
1. เลือกประเภทของการควบคุม ซึ่งมีอยู่ 2 ประเภท คือ (1) เมื่อไม่มีสัญญาณควบคุมเสริม U_{SVC} (2) เมื่อมีสัญญาณควบคุมเสริม U_{SVC} (Fuzzy control) หน้าจออินพุทการเลือกประเภทของการควบคุมแสดงในรูปที่ 7.45
2. เลือกชนิดของการรบกวน ซึ่งมี 2 ชนิด คือ การรบกวนแบบรุนแรง และการรบกวนแบบเล็กน้อย หน้าจออินพุทการเลือกชนิดของการรบกวนมีลักษณะเดียวกับรูปที่ 7.43
3. กำหนดค่าพารามิเตอร์ควบคุมฟัซซี่ โดยมีหน้าจออินพุทแสดงในรูปที่ 7.46

ตัวอย่างหน้าจอเอาท์พุทแสดงในรูปที่ 7.47 โดย (a) เป็นการรบกวนแบบรุนแรง และ (b) คือการรบกวนแบบเล็กน้อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



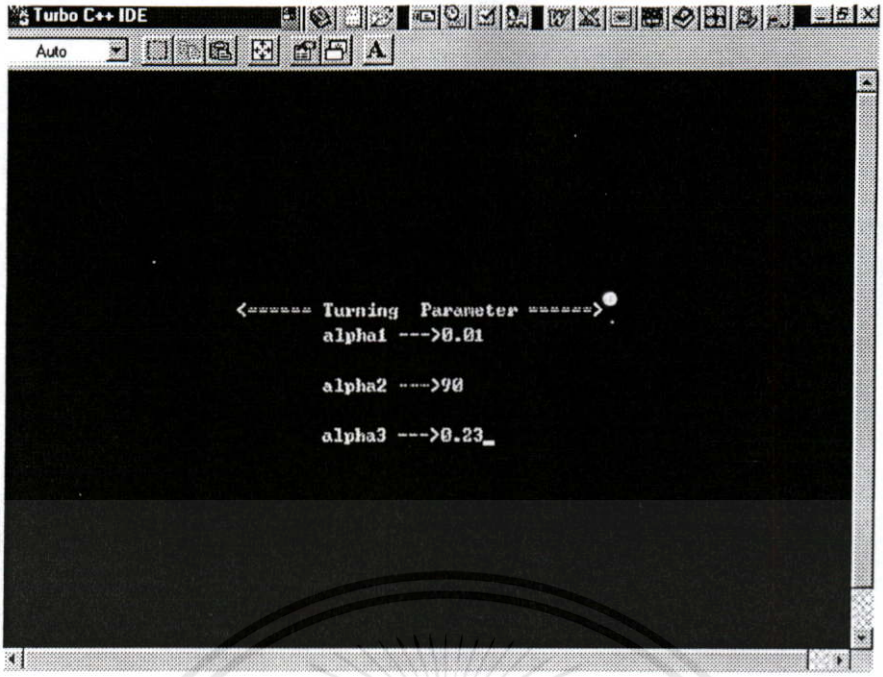
(a)



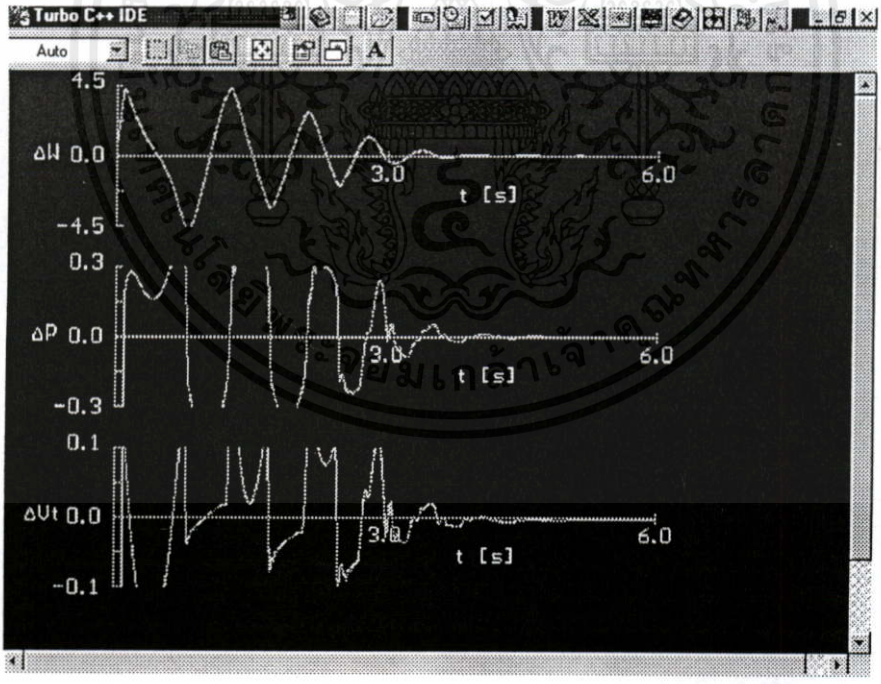
(b)

รูปที่ 7.45 หน้าจออินพุทสำหรับเลือกประเภทของการควบคุมสำหรับโปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย ... ว่ากรณีใดๆทั้งสิ้น ... ต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



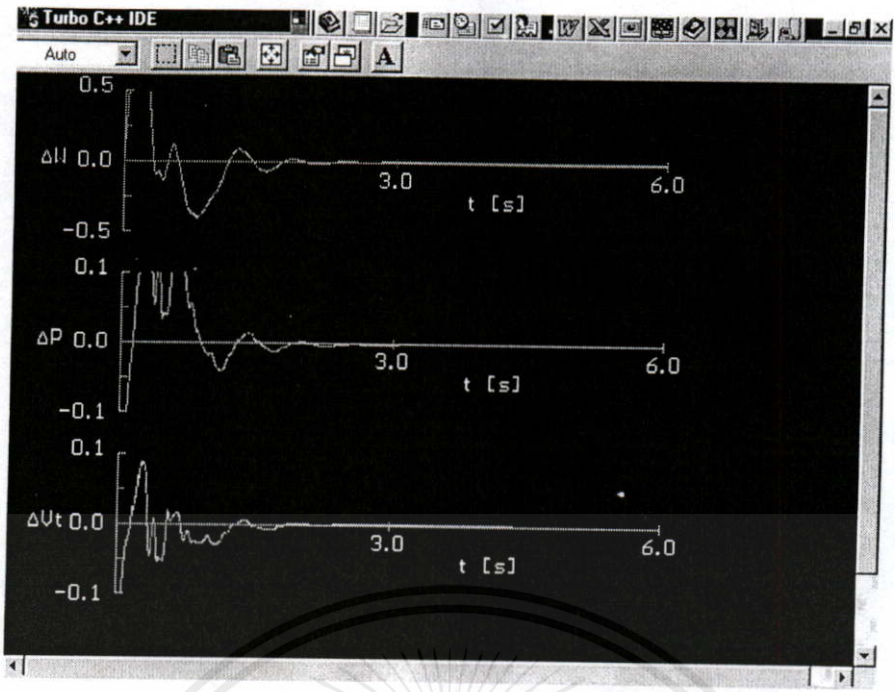
รูปที่ 7.46 หน้อจอการกำหนดค่าพารามิเตอร์ควบคุมฟัซซี่



(a)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการ ... ถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 7.47 หน้าจอเอาท์พุทกรณีควบคุมด้วย SVC ชนิดฟัซซี่



(b)

รูปที่ 7.47 (ต่อ)

7.5.3 โปรแกรมจำลองกรณีควบคุมแบบผสมผสาน

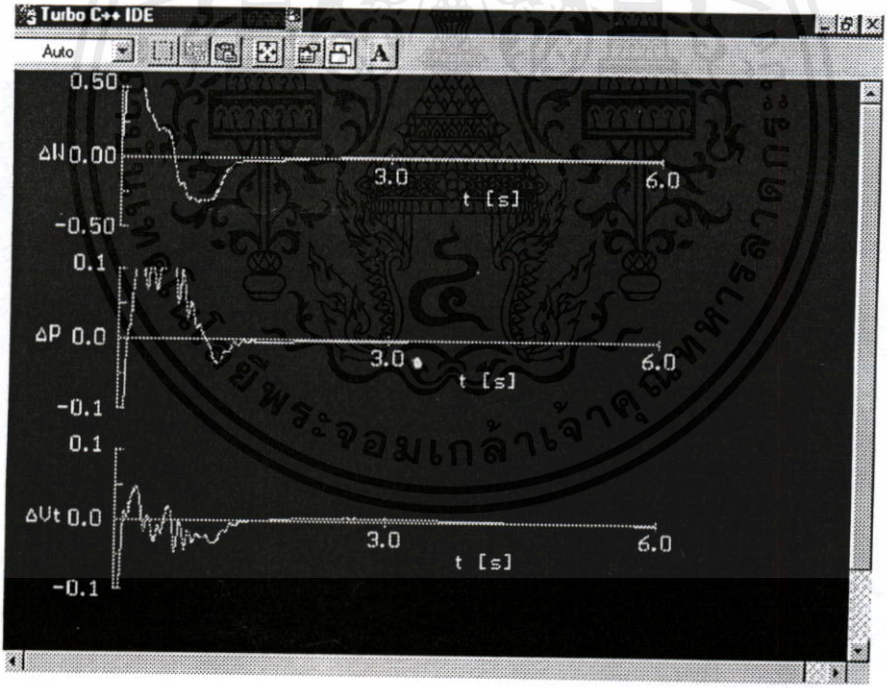
กรณีนี้ได้แก่ โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่ร่วมกับ ΔP-PSS โปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่ร่วมกับ $\Delta\omega$-PSS และโปรแกรมจำลองระบบกรณีควบคุมด้วย SVC ชนิดฟัซซี่ร่วมกับ $\Delta\omega$-PSS ชนิดฟัซซี่ ขั้นตอนการใช้งานโปรแกรมมีดังต่อไปนี้

1. เลือกชนิดของการรบกวน ซึ่งมี 2 ชนิด คือ การรบกวนแบบรุนแรง และการรบกวนแบบเล็กน้อย หน้าจออินพุทการเลือกชนิดของการรบกวนมีลักษณะเดียวกับรูปที่ 7.43
2. กำหนดค่าพารามิเตอร์ควบคุมฟัซซี่ โดยมีหน้าจออินพุทลักษณะเดียวกับรูปที่ 7.46

ตัวอย่างหน้าจอเอาท์พุทสำหรับโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน แสดงในรูปที่ 7.47 โดย (a) คือการรบกวนแบบรุนแรง และ (b) คือการรบกวนแบบเล็กน้อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(a)



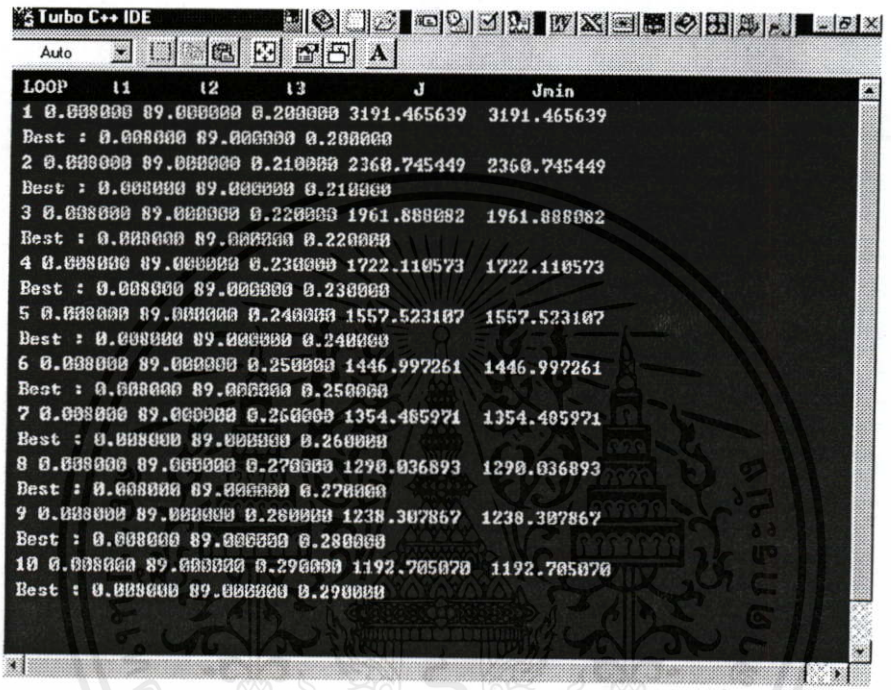
(b)

รูปที่ 7.48 หน้าจอเอาท์พุทสำหรับโปรแกรมจำลองระบบกรณีควบคุมแบบผสมผสาน

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.5.4 โปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC โดยวิธีไอเทอร์เรชั่น

การใช้งานให้รันโปรแกรม it_m.c ซึ่งจะได้หน้าจอเอาท์พุทบนจอดังรูปที่ 7.49 การเปลี่ยนวิธีการควบคุม ให้แก้ไขโปรแกรม it_m.c ที่บรรทัด #include โดยเลือกไฟล์ที่บรรจุฟังก์ชั่น performance_index ของแต่ละวิธี รายละเอียดไฟล์ดังกล่าว ดูได้จากคำอธิบายโปรแกรม it_m.c ในภาคผนวก ฉ.

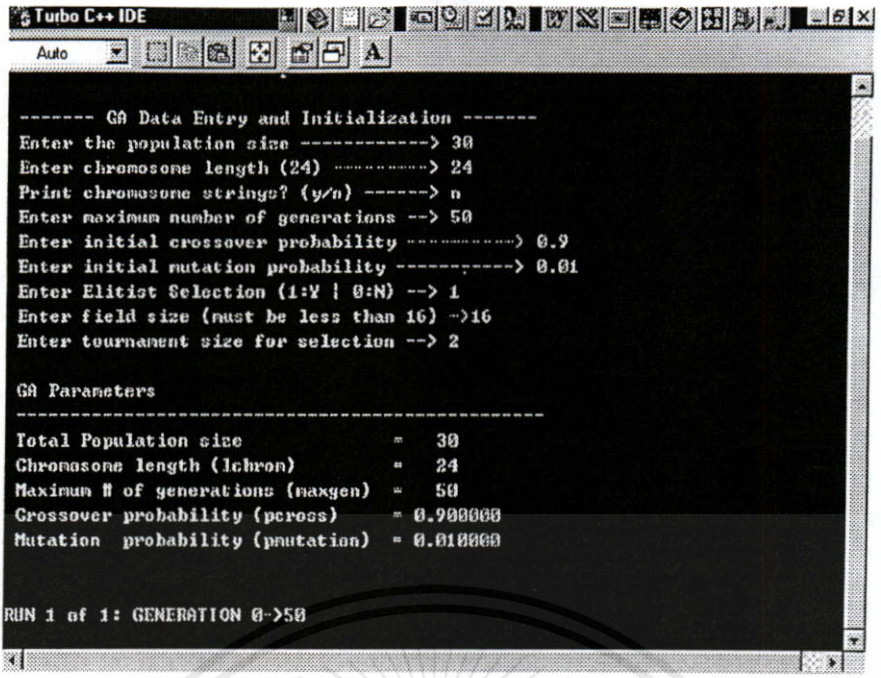


รูปที่ 7.49 หน้าจอเอาท์พุทโปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีไอเทอร์เรชั่น

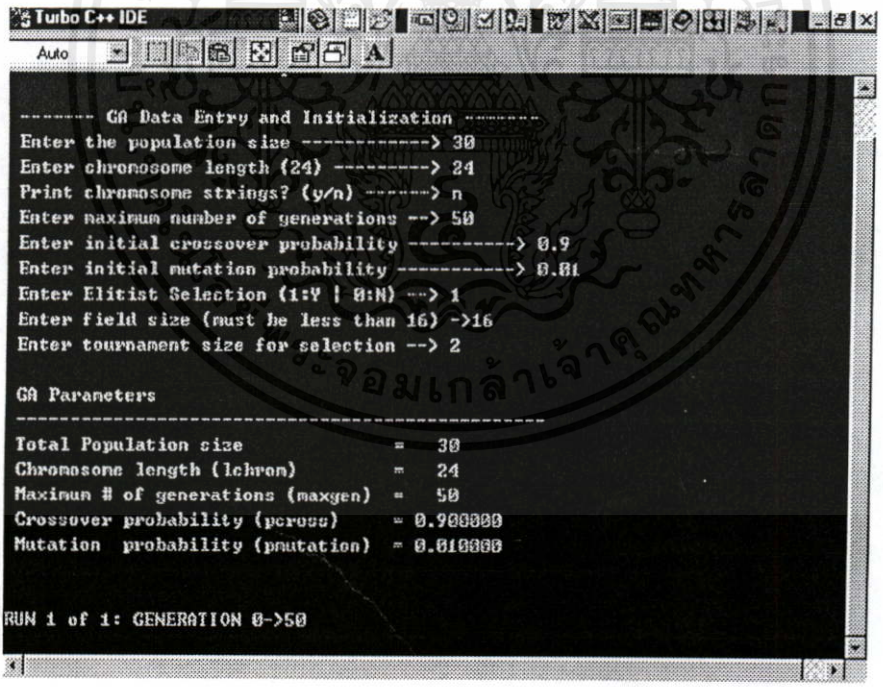
7.5.5 โปรแกรมออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีเจเนติกอัลกอริทึ่ม (GAFCD)

การใช้งานโปรแกรมนี้ เริ่มจากรันโปรแกรม gafcd.c แล้วจะปรากฏหน้าจออินพุทให้ใส่ค่าพารามิเตอร์ ซึ่งได้แก่ จำนวนครั้งที่จะรันโปรแกรม ขนาดประชากร ความยาวโครโมโซม จำนวนเจนเนอเรชั่นสูงสุด ค่าความน่าจะเป็นในการครอสโอเวอร์และมิวเทชั่นเริ่มต้น ตั้งค่าอิลิทิสซึม และขนาดทัวร์นาเมนท์ หน้าจออินพุทดังกล่าวแสดงในรูปที่ 7.50 เมื่อใส่อินพุทครบถ้วนแล้ว หน้าจอรายงานผลการใส่ค่าพารามิเตอร์จะปรากฏขึ้นมาดังรูป 7.51 ซึ่งโปรแกรมจะทำงานต่อไปเรื่อยๆจนกว่าจะถึงค่าเจนเนอเรชั่นสูงสุด โดยหน้าจอรายงานผลในแต่ละเจนเนอเรชั่น แสดงดังรูปที่ 7.52

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

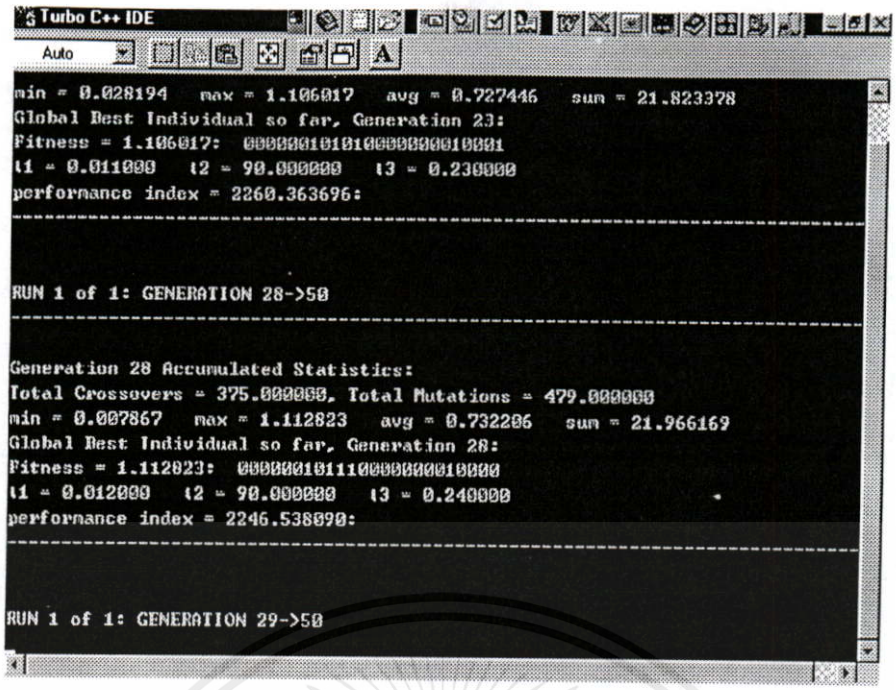


รูปที่ 7.50 หน้าจออินพุทสำหรับโปรแกรม GAFCD



รูปที่ 7.51 หน้าจอรายงานผลการใส่ค่าพารามิเตอร์ของโปรแกรม GAFCD

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านกา
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.52 หน้าจอรายงานผลในแต่ละเจนเนอเรชั่นของโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 8
ผลการจำลองด้วยคอมพิวเตอร์

เนื้อหาในบทนี้จะแสดงผลการจำลองระบบด้วยวิธีการควบคุมแบบต่างๆตามที่ได้กล่าวมาในบทที่ 7 โดยจะแบ่งเป็น 3 ส่วน คือ ส่วนแรก แสดงผลตอบสนองระบบกรณีควบคุมด้วย PSS อย่างเดียว ส่วนที่สอง แสดงผลตอบสนองเมื่อติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{SVC} และส่วนที่สาม เป็นการควบคุมด้วย SVC ชนิดฟัซซี่ โดยจะแสดงทั้งผลตอบสนองเมื่อออกแบบด้วยวิธีไอเทอร์เรชั่น และวิธีเจเนติกอัลกอริโดยใช้โปรแกรม GAFCD

8.1 ค่าคงที่ระบบจำลอง

แบบจำลองที่ใช้ทดสอบ แสดงในรูปที่ 5.1 โดยจะใช้ค่าคงที่ต่างๆตามตารางที่ 8.1 [4]

ตารางที่ 8.1 ค่าคงที่ระบบจำลอง

ชนิดอุปกรณ์	ค่าตัวแปร
เครื่องกำเนิดไฟฟ้า (GEN)	T'_{do}= 5.6(s), M = 0.0191(pu-s), D = 0.00531(pu), X'_d=0.25(pu), X_d=1 .62(pu), X_q = 1.6(pu), V_t= 1.0(pu), V_b = 1.0(pu), ω_o = 377(rad/s) , P_o=1.0(pu)
สายส่ง	X_t = 0.14 (pu), X_j = 0.05(pu), X_{l1} = 0.21(pu), X_{l2} = 0.30(pu)
AVR	K_A = 200, T_A = 0.05(s), K_f = 0.07, T_f = 1.25(s)
GOV	σ = 0.05, T_g = 0.02(s), T_t = 2.0(s)
SVC	K_S = 50, T_C = 0.05(s), T_V = 0.01(s), V_C = 1.0(pu)
PSS1 (ΔP-PSS)	K = 1.0, T_m = 0.01(s), T_r = 4.0(s), T_1 = 0.04(s), T_2 = 0.05(s), T_3 = T_4 = 0.0(s)
PSS2 ($\Delta\omega$-PSS)	K = 13 , ω_0 = 377(rad/s) , T_m = 0.01(s) , T_r = 4.0(s) , T_1 = 0.3(s), T_2 = 0.02(s),T_3 = 0.2(s), T_4 = 0.02(s)
PSS3 ($\Delta\omega$-PSS ชนิดฟัซซี่)	T_m = 0.056(s), T_r = 4.0(s)
SVC FUZ	T_m = 0.01(s), T_r = 4.0(s), T = 0.01(s), N = 1201

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.2 ผลตอบสนองกรณีควบคุมด้วย PSS

ผลตอบสนองกรณีควบคุมด้วย PSS1 ทั้งกรณีการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.1 และ 8.2 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.3 และ 8.4 ผลตอบสนองกรณีควบคุมด้วย PSS2 ทั้งกรณีการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.5 และ 8.6 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.7 และ 8.8 ผลตอบสนองกรณีควบคุมด้วย PSS3 ทั้งกรณีการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.9 และ 8.10 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.11 และ 8.12 ผลสรุปจากการควบคุมด้วย PSS ทั้ง 3 ชนิด แสดงในตารางที่ 8.2

ตารางที่ 8.2 ผลสรุปจากการควบคุมด้วย PSS

ชนิดการควบคุม	X_j = 1.0 เท่า		X_j = 1.3 เท่า	
	large	small	large	small
PSS1	✗	✓	✗	✓
PSS2	✗	✓	✗	✓
PSS3	✗	✓	✗	✓

✓ เสถียร ✗ ไม่เสถียร – มีแนวโน้มเสถียร

PSS3 ใช้ $\alpha w1 = 0.31$ $\alpha w2 = 11.0^{\circ}$ $\alpha w3 = 0.89$

8.3 ผลตอบสนองกรณีติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc}

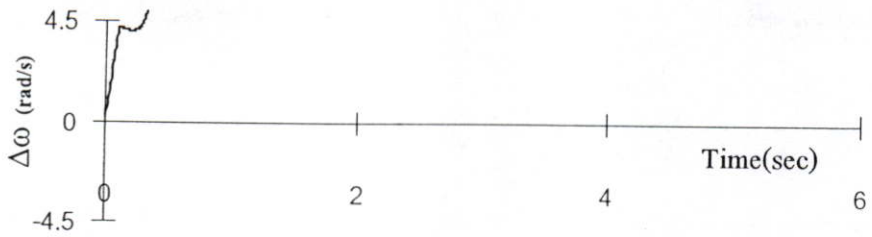
ผลตอบสนองกรณีติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc} ทั้งกรณีการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.13 และ 8.14 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.15 และ 8.16 ผลสรุปจากการควบคุมวิธีนี้แสดงในตารางที่ 8.3

ตารางที่ 8.3 ผลสรุปกรณีติดตั้ง SVC โดยไม่มีสัญญาณควบคุมเสริม U_{svc}

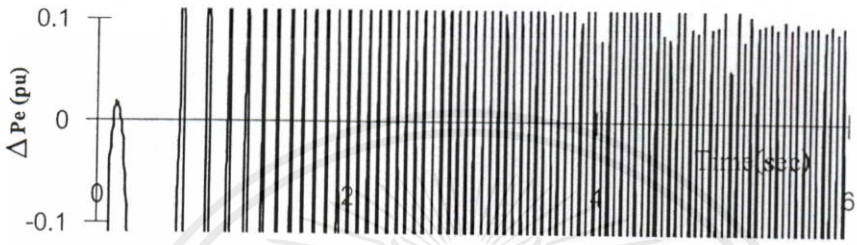
ชนิดการควบคุม	X_j = 1.0 เท่า		X_j = 1.3 เท่า	
	large	small	large	small
SVC (No U_{svc})	✗	–	✗	–

✓ เสถียร ✗ ไม่เสถียร – มีแนวโน้มเสถียร

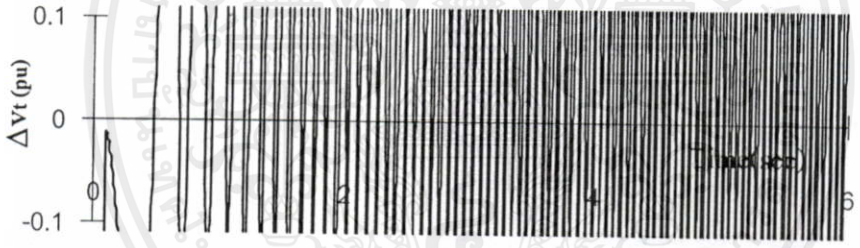
เอกสารนี้มีลิขสิทธิ์ สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



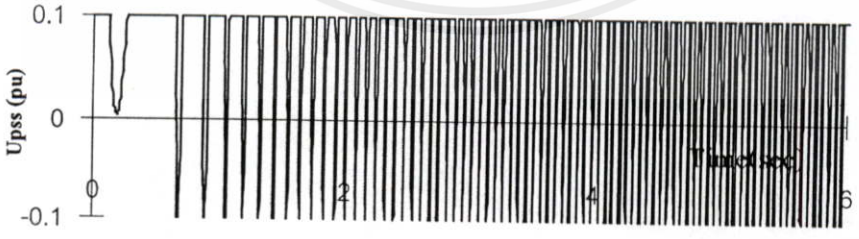
(a)



(b)



(c)

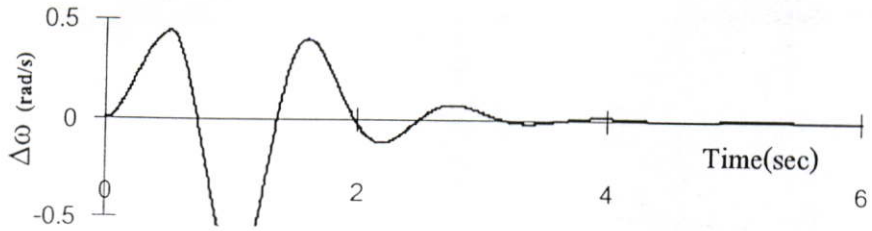


(d)

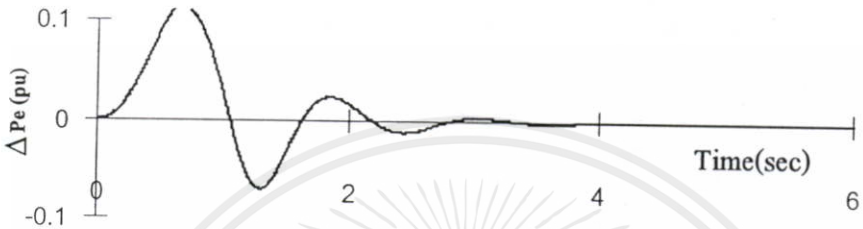
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.1 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า

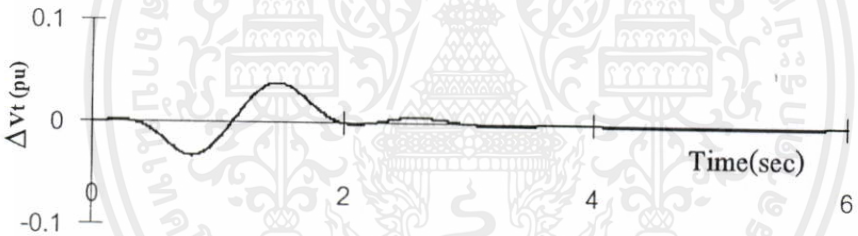
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนำไปใช้



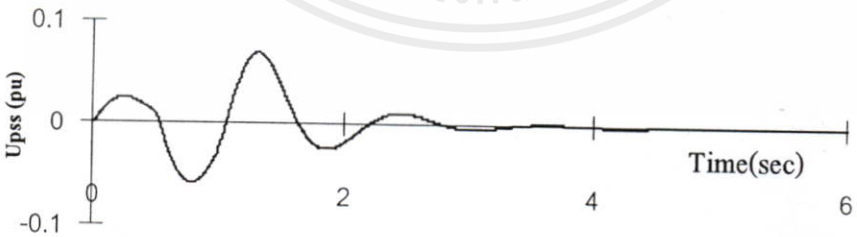
(a)



(b)



(c)

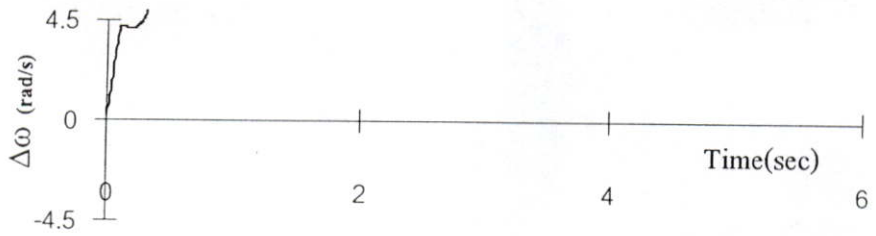


(d)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

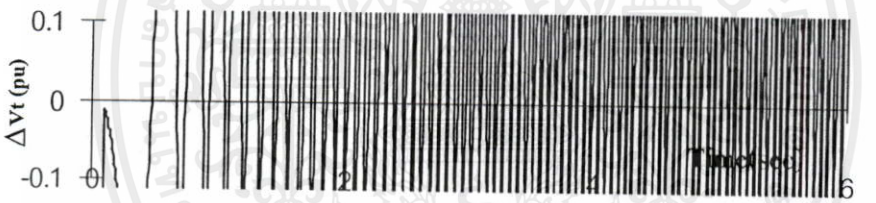
รูปที่ 8.2 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

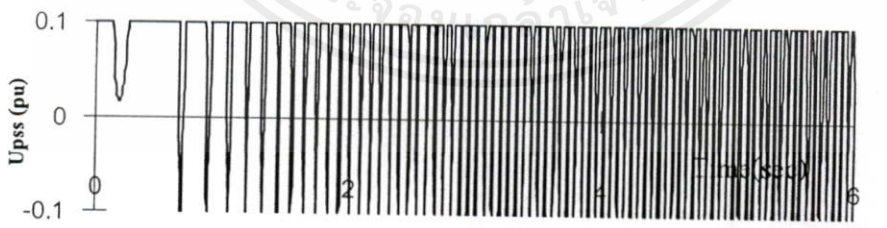


(a)

(b)



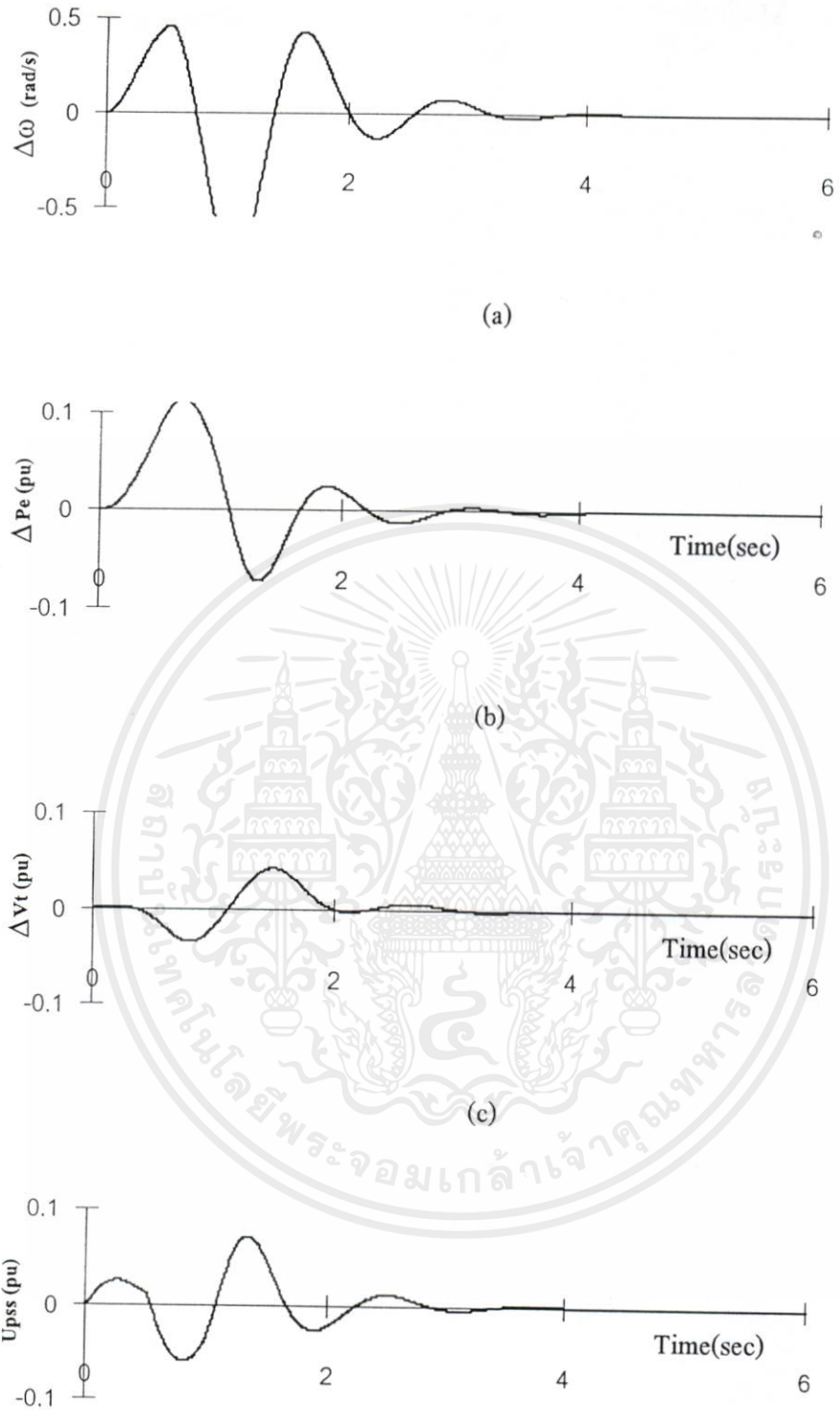
(c)



(d)

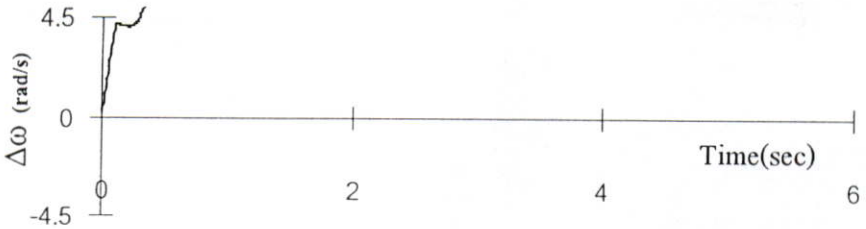
รูปที่ 8.3 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

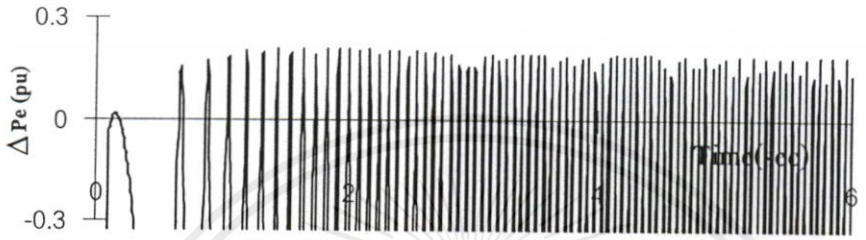


รูปที่ 8.4 ผลตอบสนองกรณีควบคุมด้วย PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า

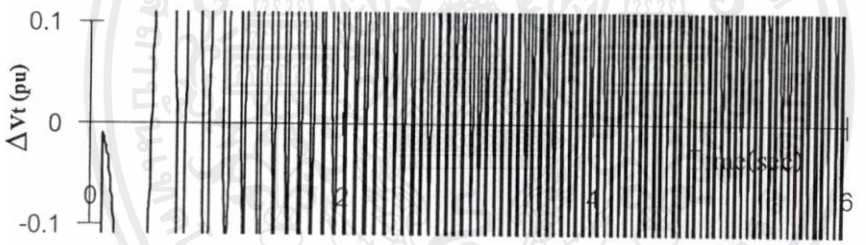
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



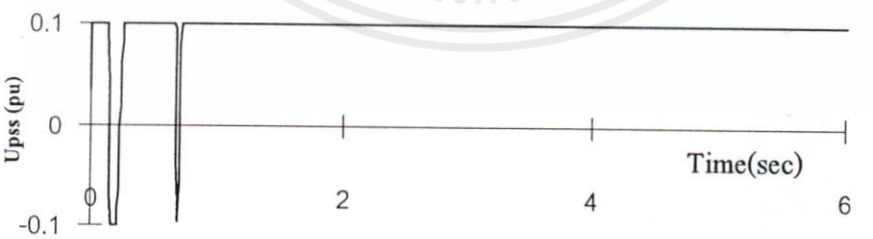
(a)



(b)



(c)

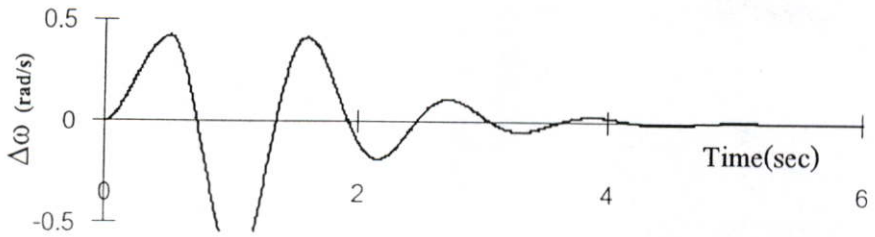


(d)

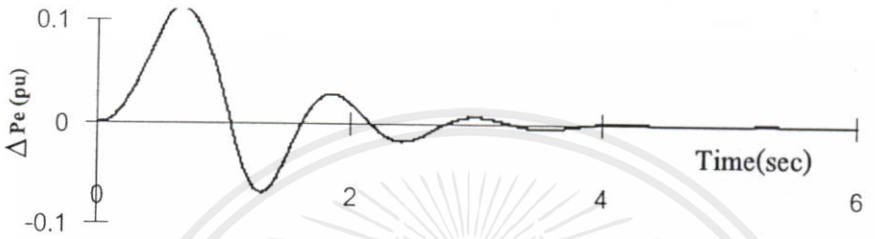
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.5 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_e = 1.0 เท่า

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

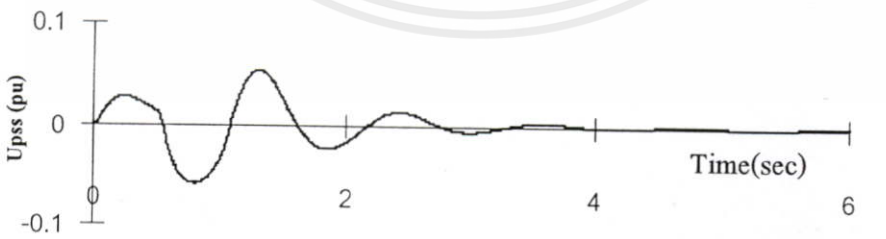


(a)



(b)

(c)

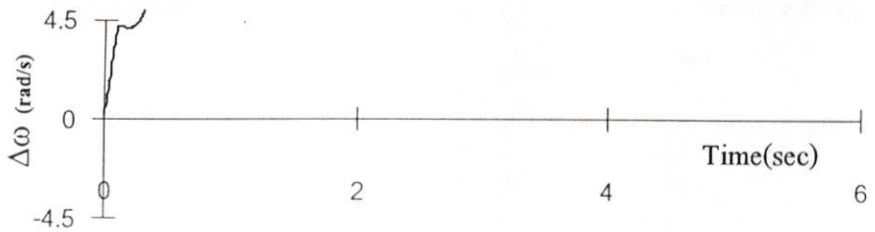


(d)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

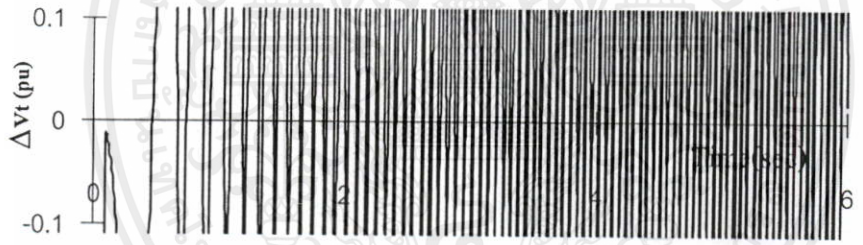
รูปที่ 8.6 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า

ว่าการนำไปใช้ ทุกกรณี ยกเว้นการนำไปใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้

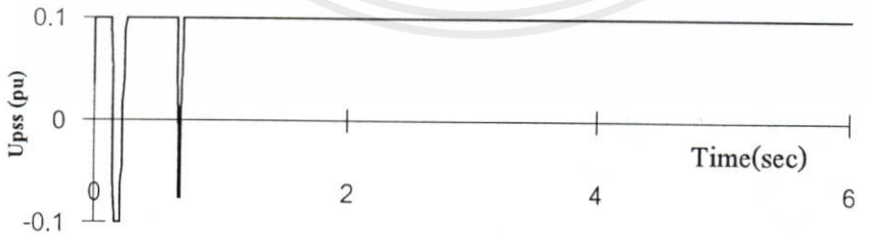


(a)

(b)



(c)

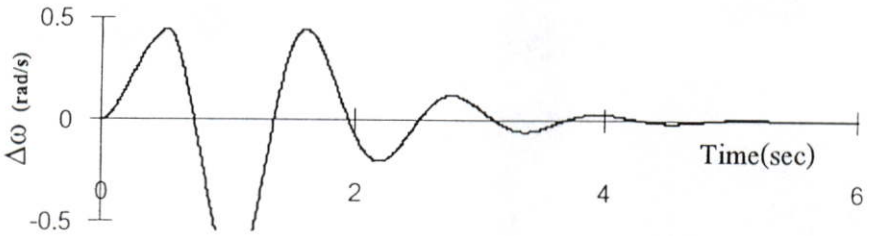


(d)

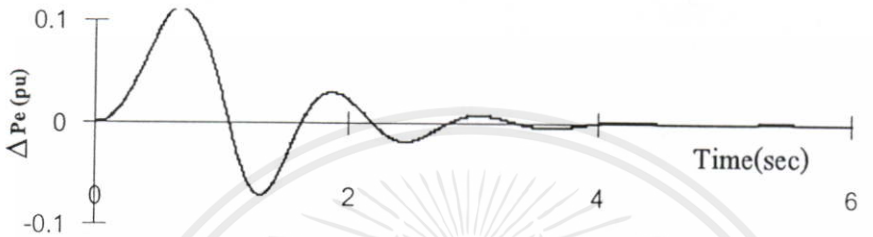
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.7 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า

ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนำไปใช้

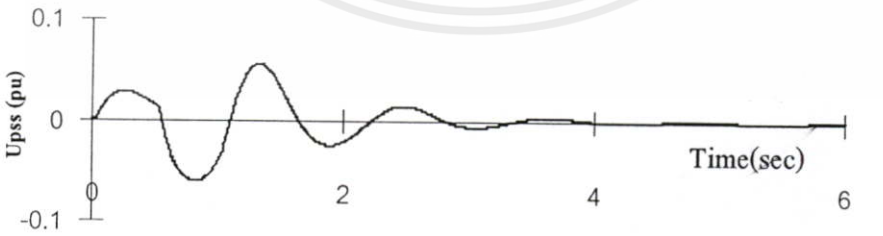


(a)



(b)

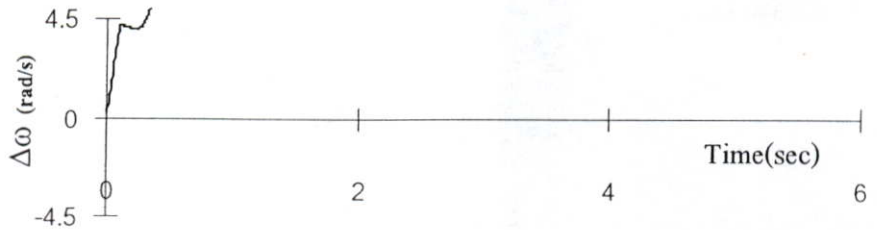
(c)



(d)

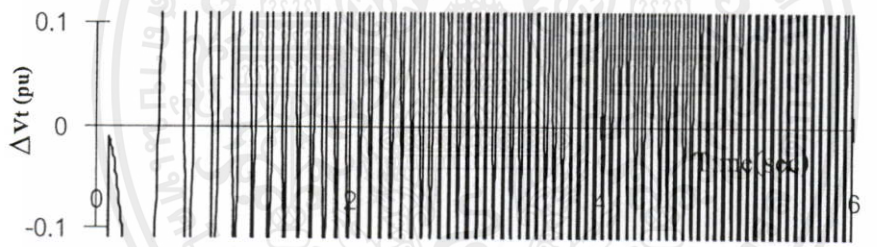
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.8 ผลตอบสนองกรณีควบคุมด้วย PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า

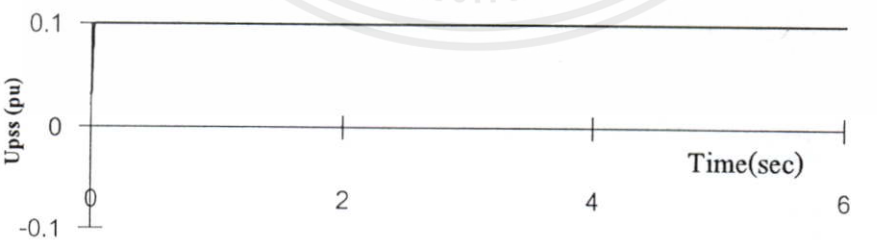


(a)

(b)



(c)

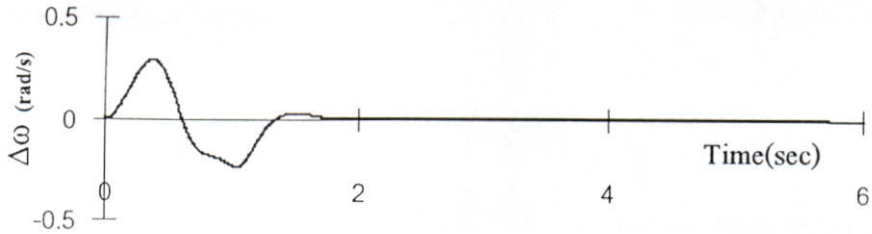


(d)

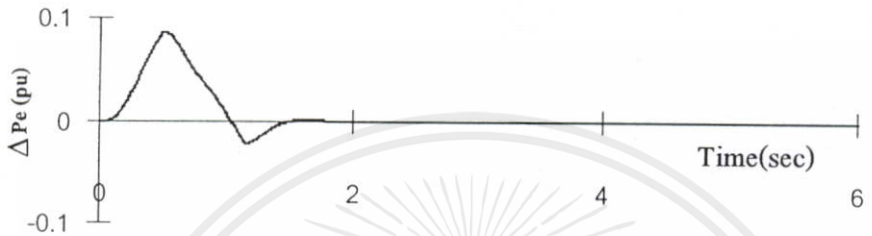
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.9 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_e = 1.0 เท่า

ค้าทุกกรณีโดยไม่ได้รับอนุญาตเป็นลายลักษณ์อักษรจากจุฬาลงกรณ์มหาวิทยาลัย

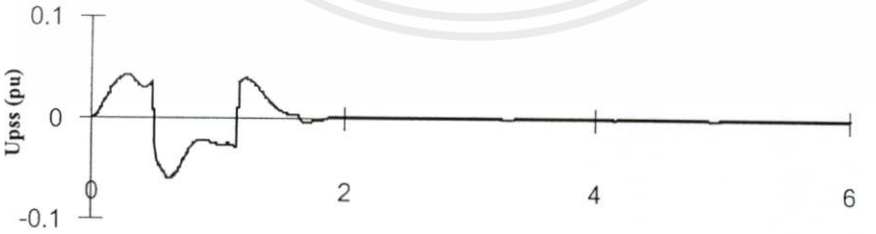


(a)



(b)

(c)

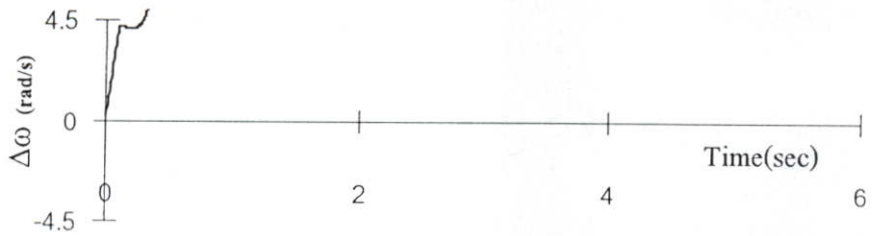


(d)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

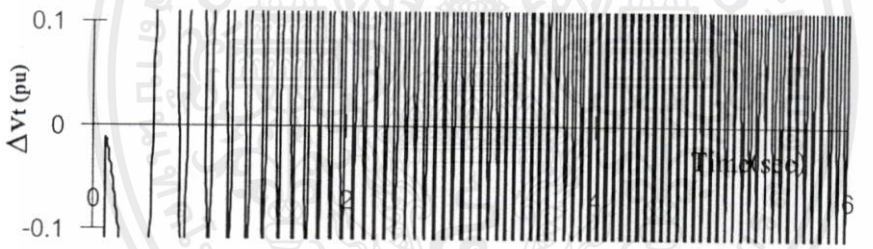
รูปที่ 8.10 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

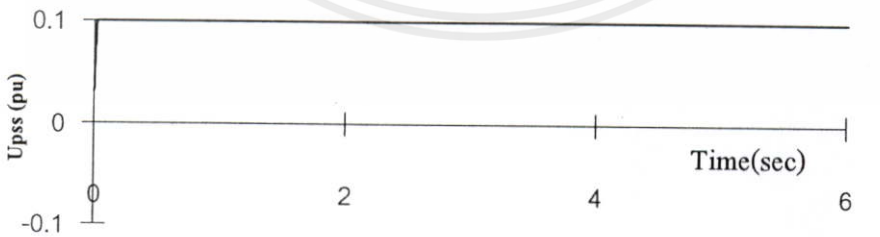


(a)

(b)



(c)

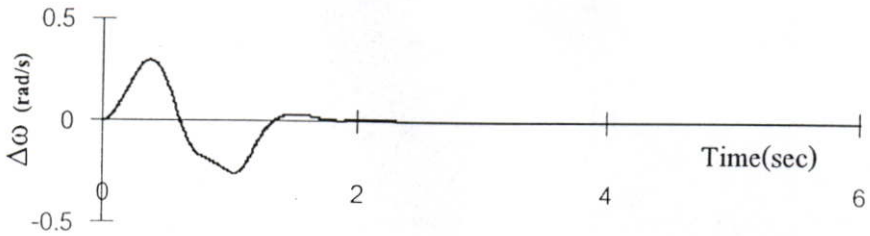


(d)

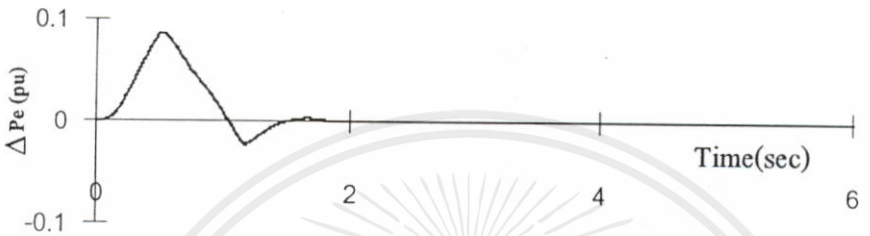
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.11 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า

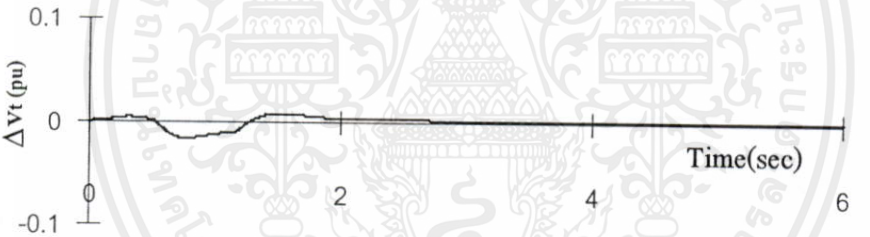
ว่าการนำไปใช้ในทางใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้



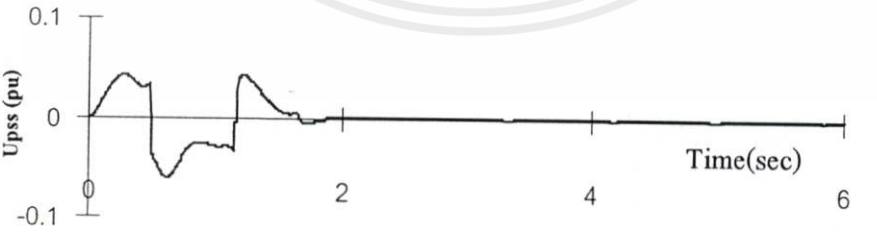
(a)



(b)



(c)

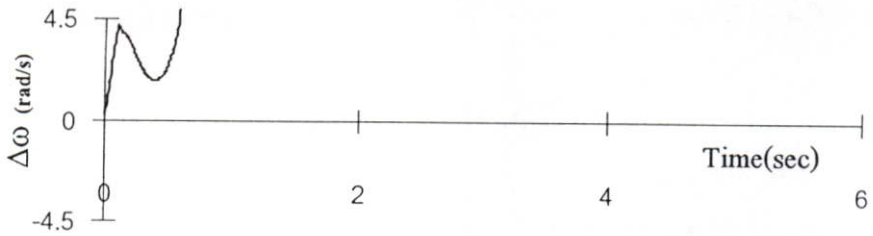


(d)

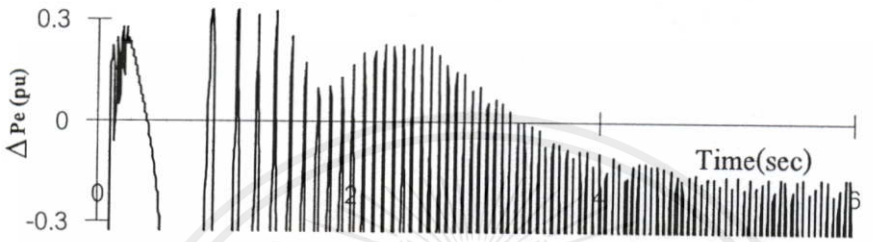
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.12 ผลตอบสนองกรณีควบคุมด้วย PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า

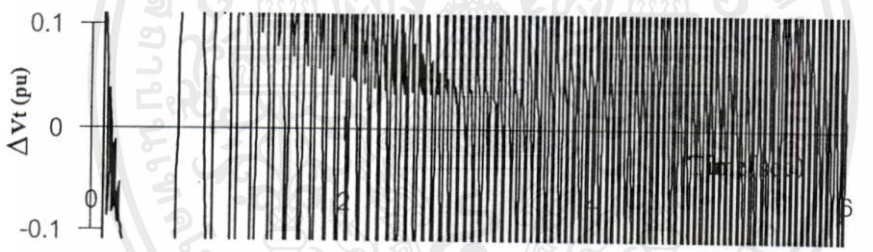
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้



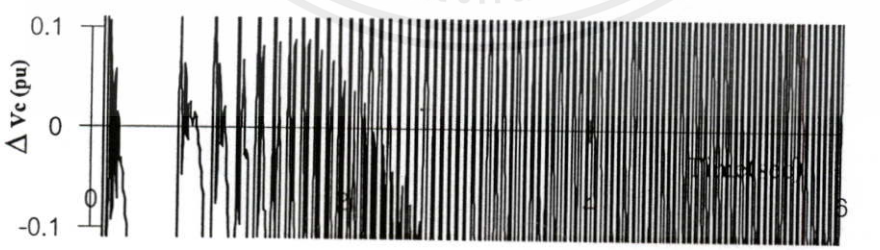
(a)



(b)



(c)

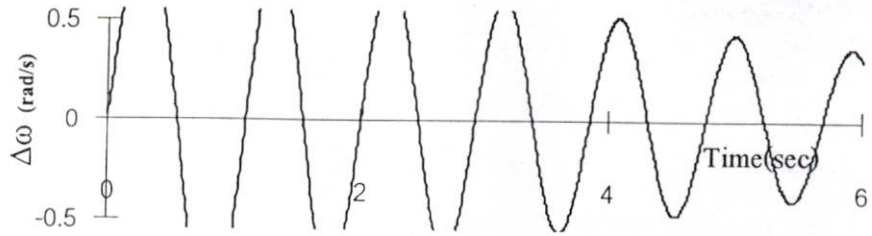


(d)

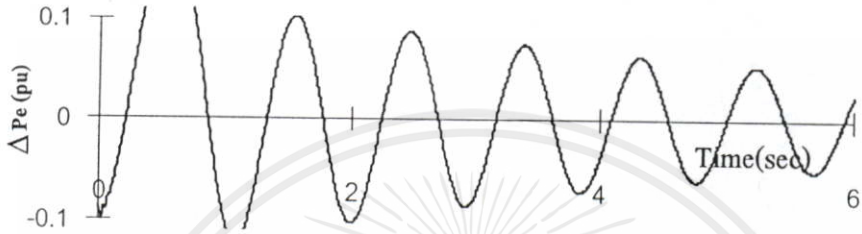
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.13 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มี สัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

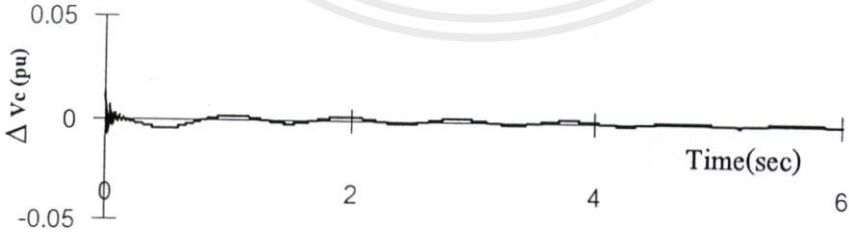


(a)



(b)

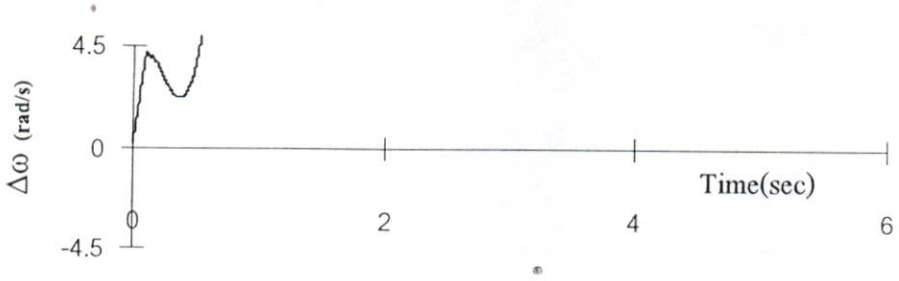
(c)



(d)

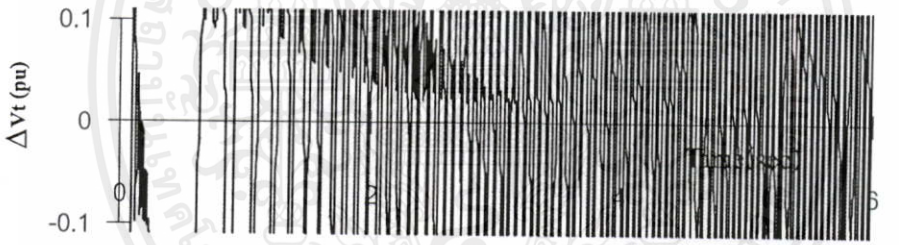
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนำไปใช้

รูปที่ 8.14 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มี สัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า

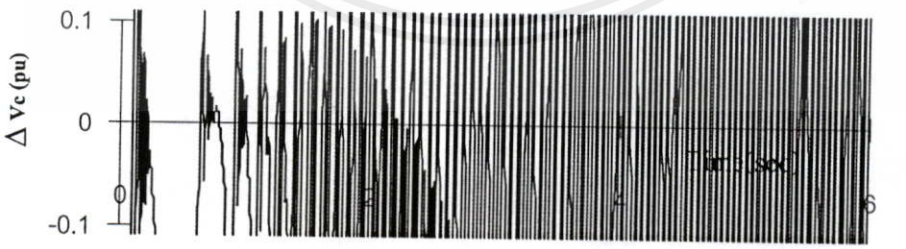


(a)

(b)



(c)

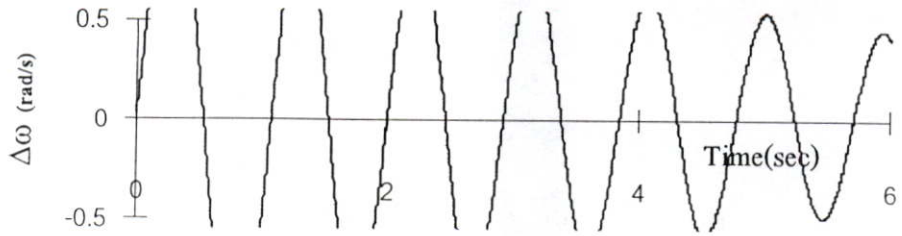


(d)

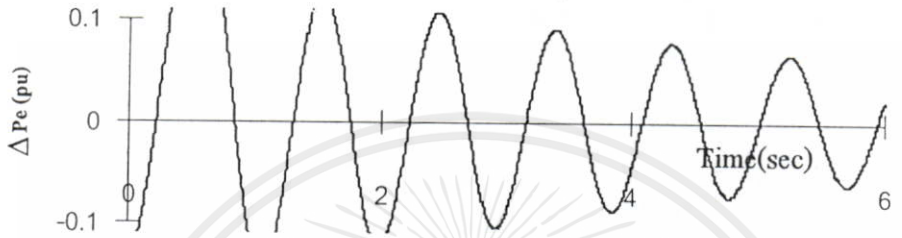
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.15 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มี สัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า

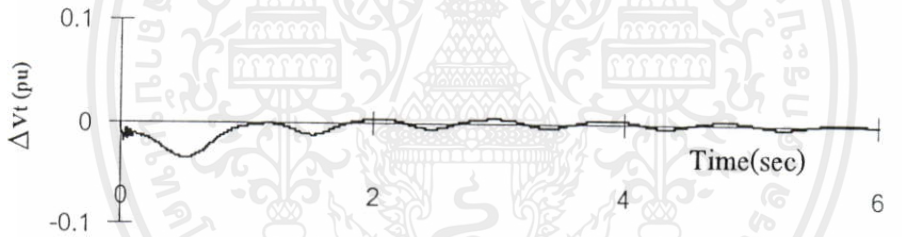
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



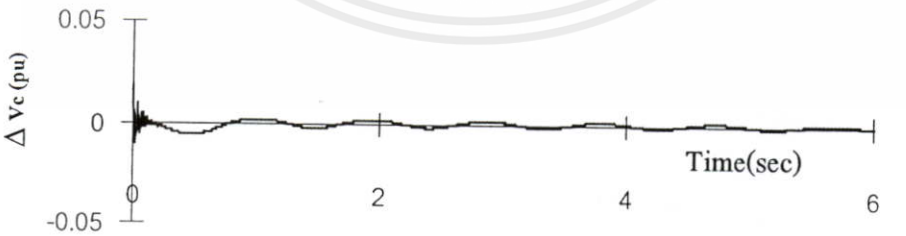
(a)



(b)



(c)



(d)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

รูปที่ 8.16 ผลตอบสนองกรณีควบคุมด้วย SVC โดยไม่มี สัญญาณควบคุมเสริม U_{svc} ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.3$ เท่า

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.4 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่

ผลตอบสนองกรณีนี้จะแบ่งออกเป็น 2 กลุ่ม คือ (1) กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว (2) กรณีควบคุมแบบผสมผสานระหว่าง SVC ชนิดฟัซซี่กับ PSS ทั้ง 3 ชนิดในหัวข้อ 8.2 โดยจะทำการเปรียบเทียบกับวิธีการออกแบบ 2 วิธี คือ (1) วิธีไอเทอร์เรชั่น (2) วิธีเจเนติกอัลกอริทึม โดยใช้โปรแกรม GAFCD

8.4.1 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่น

8.4.1.1 กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว

ผลการออกแบบด้วยวิธีไอเทอร์เรชั่นสำหรับการควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว แสดงในตารางที่ 8.4 ผลตอบสนองจากการใช้ค่าพารามิเตอร์ควบคุมฟัซซี่ในตารางที่ 8.4 ต่อการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.17 และ 8.18 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.19 และ 8.20

ตารางที่ 8.4 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่น กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว

ชนิดการควบคุม	$\alpha 1$	$\alpha 2$	$\alpha 3$	$J(X_j = 1.0$ เท่า)	$J(X_j = 1.3$ เท่า)
SVC	0.01	90°	0.23	2313.1667	3114.2698

8.4.1.2 กรณีควบคุมแบบผสมผสาน

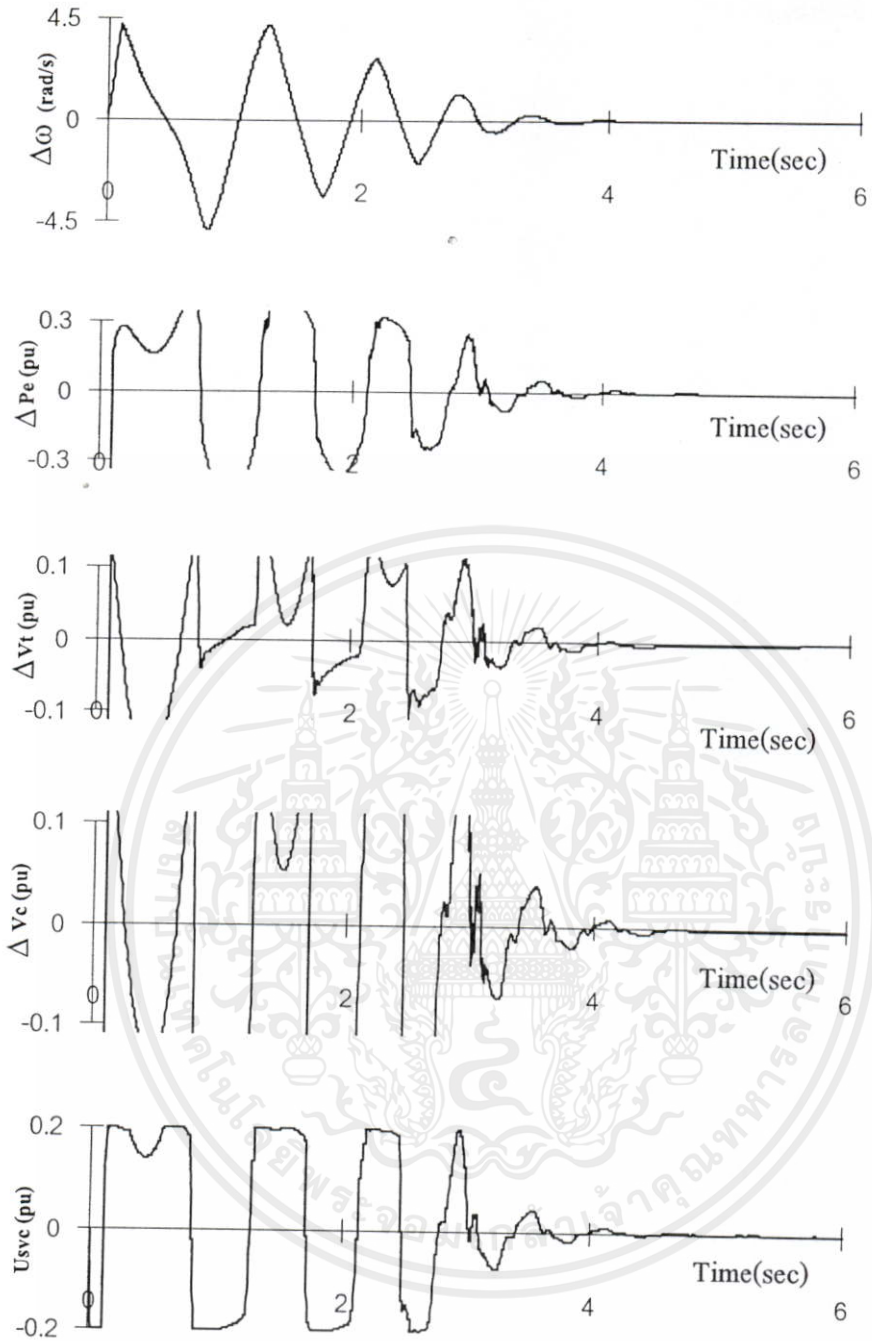
ผลการออกแบบด้วยวิธีไอเทอร์เรชั่นสำหรับการควบคุมแบบผสมผสาน แสดงในตารางที่ 8.5 ผลตอบสนองจากการใช้ค่าพารามิเตอร์ควบคุมฟัซซี่ในตารางที่ 8.5 ต่อการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า แสดงในรูปที่ 8.21 และ 8.26 เมื่อ X_j = 1.3 เท่า แสดงในรูปที่ 8.27 และ 8.32

ตารางที่ 8.5 ผลการออกแบบด้วยวิธีไอเทอร์เรชั่น กรณีควบคุมแบบผสมผสาน

ชนิดการควบคุม	$\alpha 1$	$\alpha 2$	$\alpha 3$	$J(X_j = 1.0$ เท่า)	$J(X_j = 1.3$ เท่า)
SVC+PSS1	0.01	90°	0.23	1267.3309	387800.3114
SVC+PSS2	0.01	85°	0.24	1089.6946	408544.8784
SVC+PSS3	0.01	90°	0.29	819.8666	984.7628

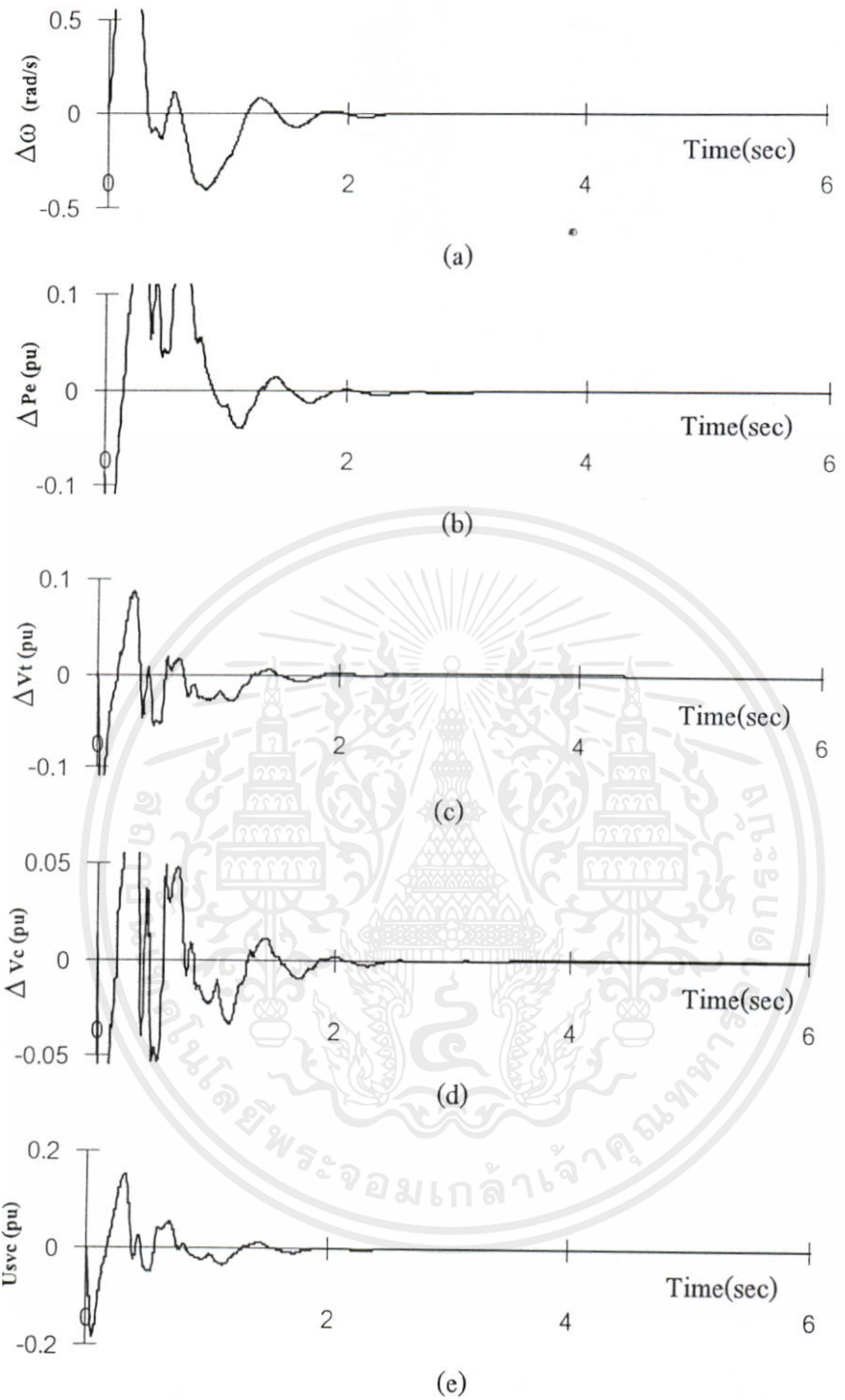
PSS3 ใช้ $\alpha w1 = 0.31$ $\alpha w2 = 11.0^\circ$ $\alpha w3 = 0.89$

สารนิพนธ์นี้เป็นลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



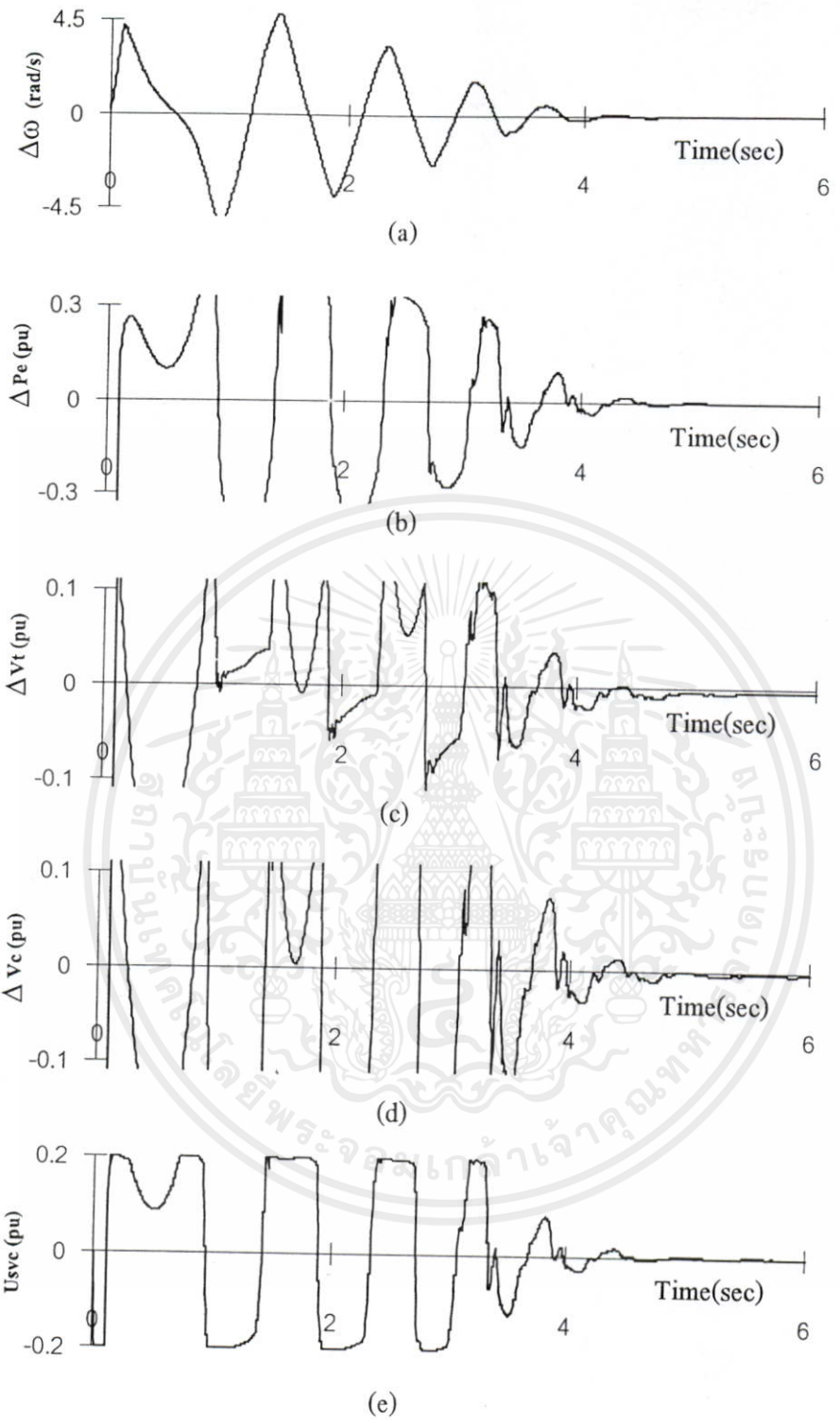
รูปที่ 8.17 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



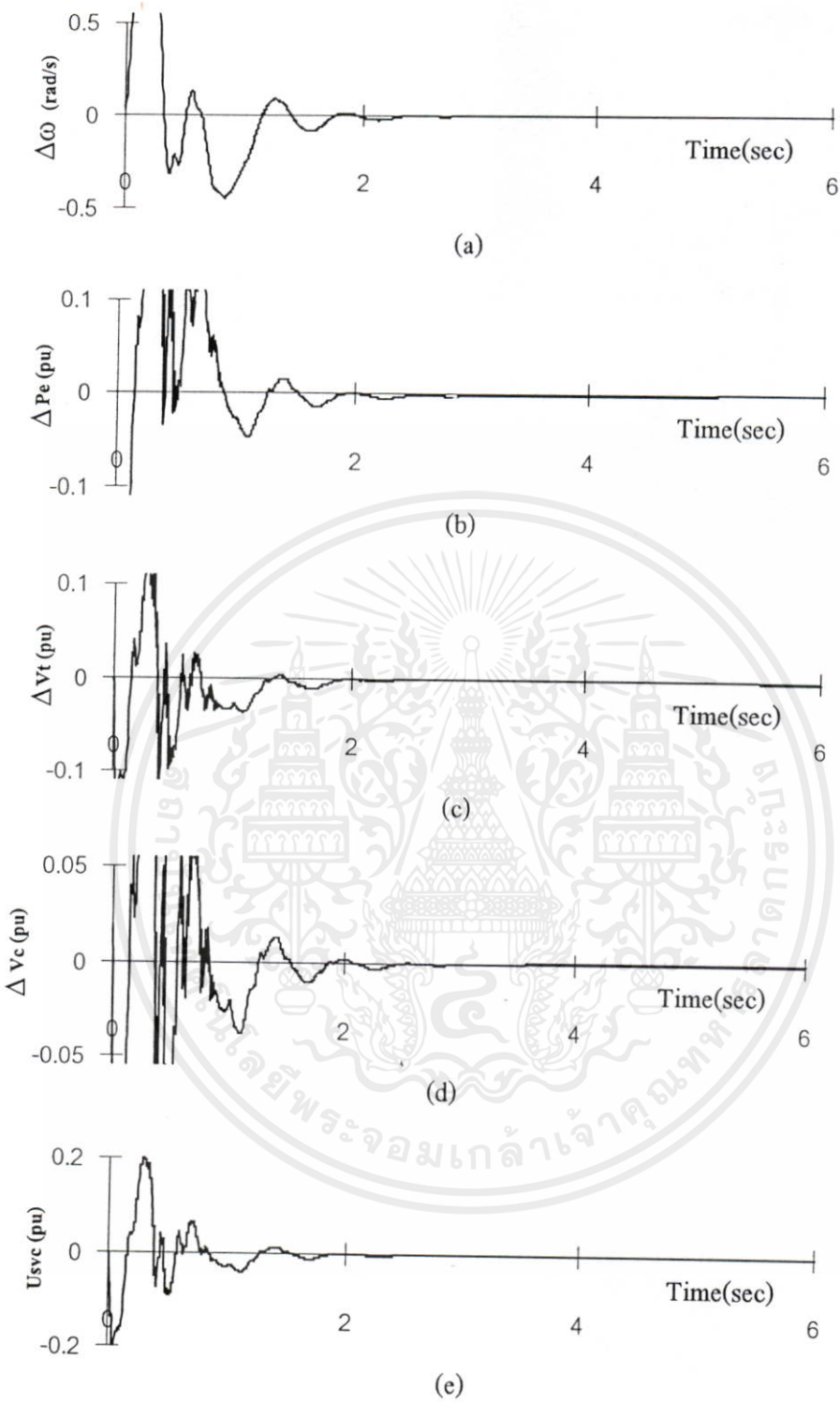
รูปที่ 8.18 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซื่อย่างเดียว ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



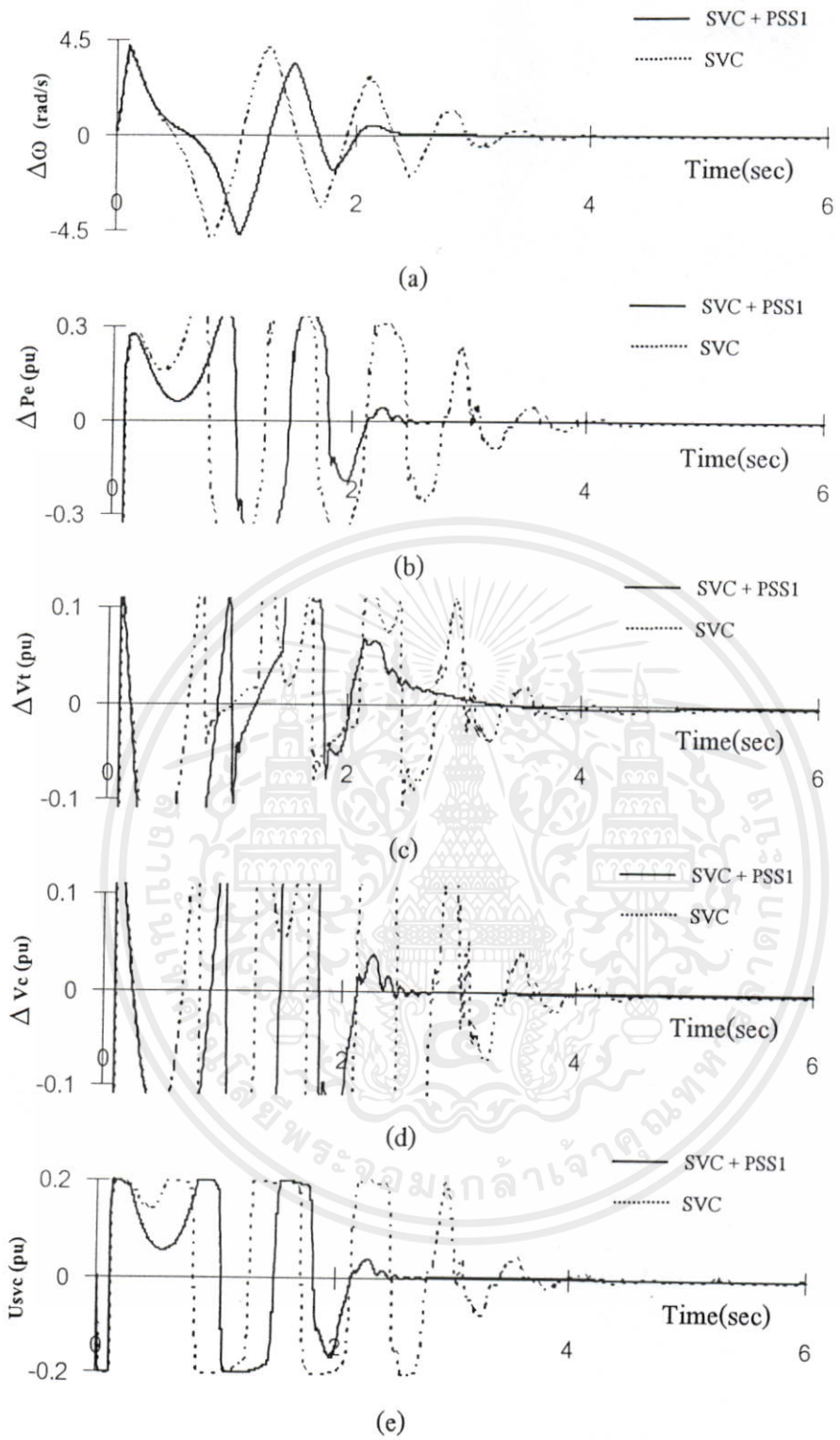
รูปที่ 8.19 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย ว่ากรณีใดๆทั้งสิ้น ต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



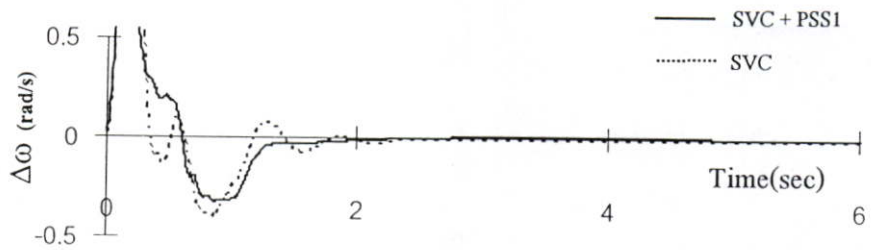
รูปที่ 8.20 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ว่ากรณีใดๆ ที่ได้ทำการใช้งานวิทยานิพนธ์นี้ ไม่ว่าจะเป็นการอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

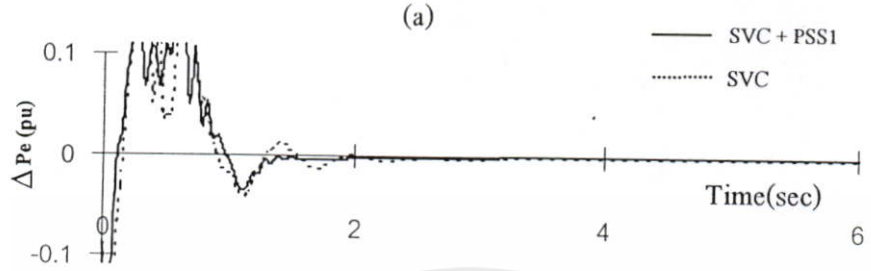


รูปที่ 8.21 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

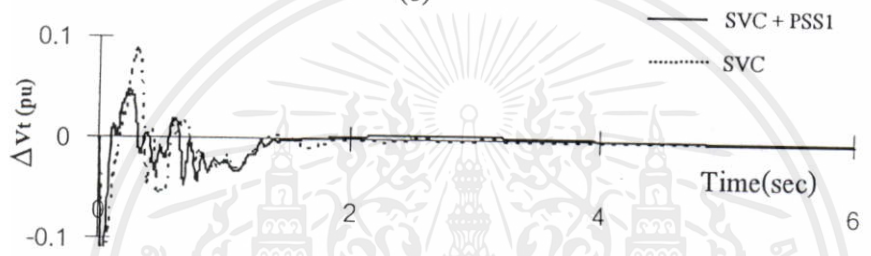
ว่ากรณีใดๆ ... และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(a)

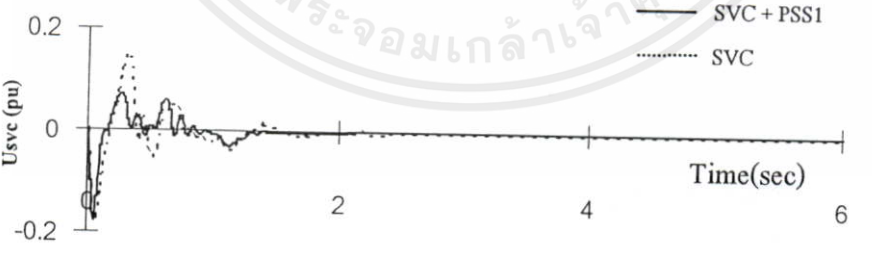


(b)



(c)

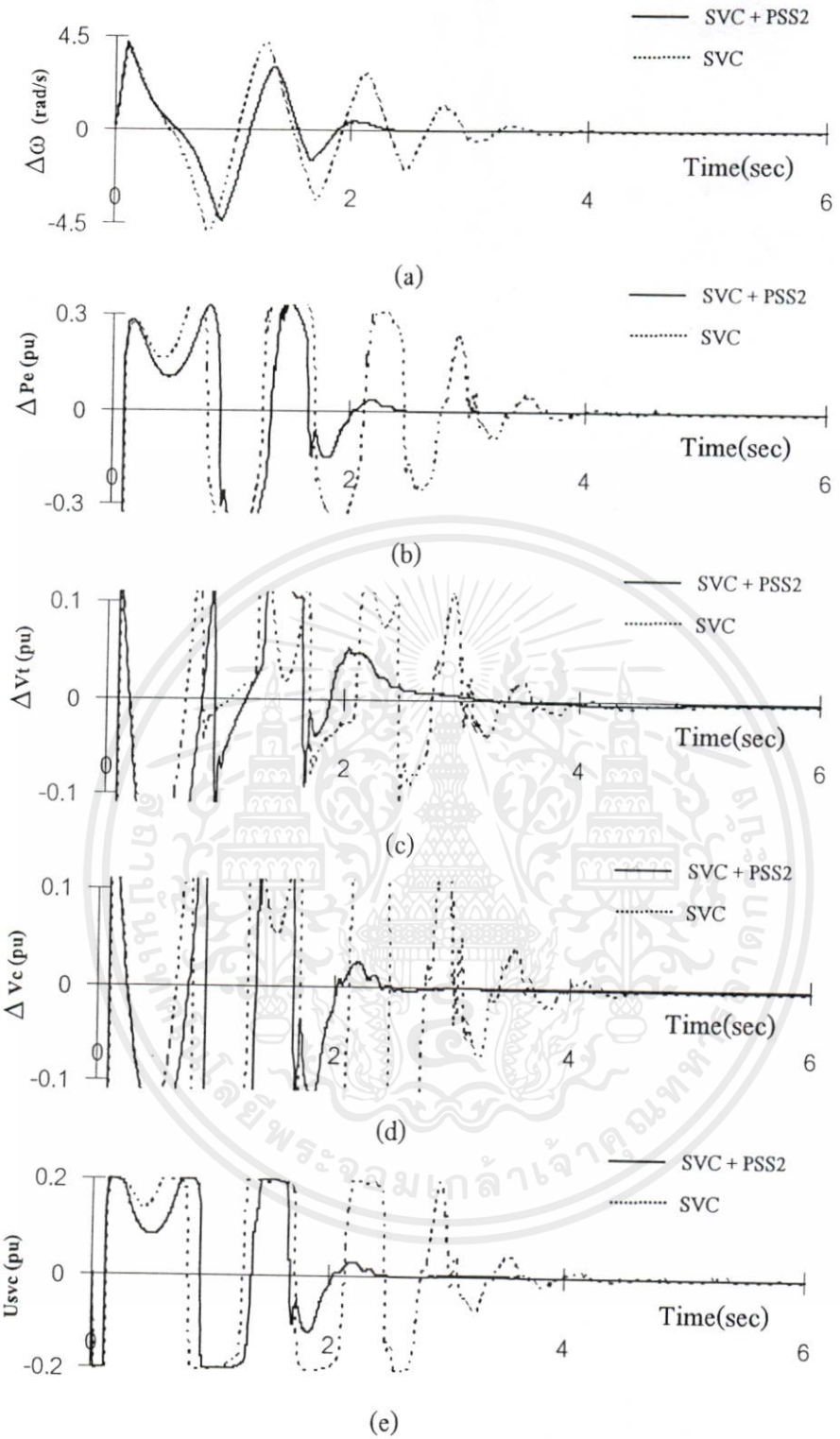
(d)



(e)

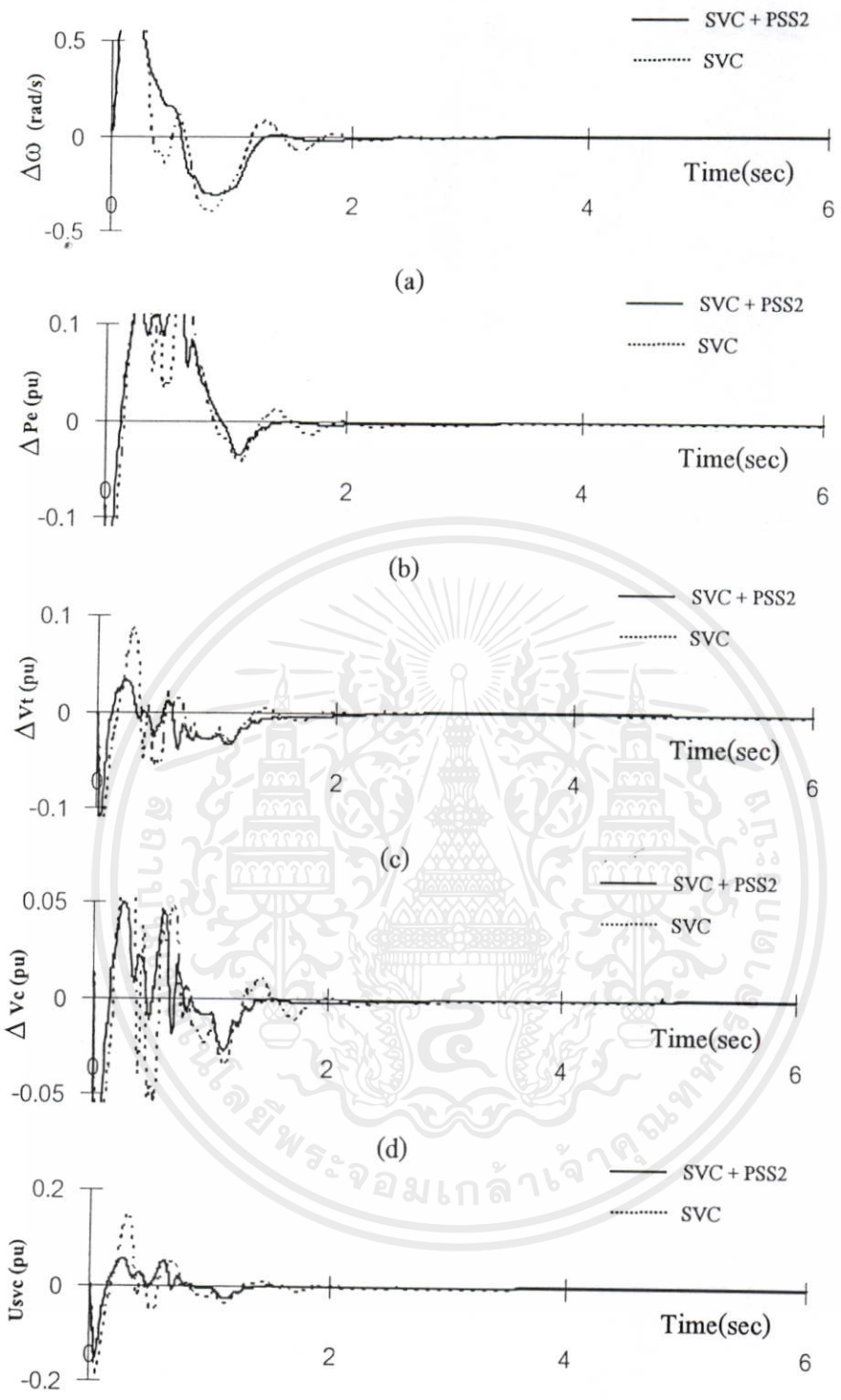
รูปที่ 8.22 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ว่ากรณีใดๆทั้งสิ้น ยกเว้นจะมีการอนุญาตเป็นลายลักษณ์อักษร และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



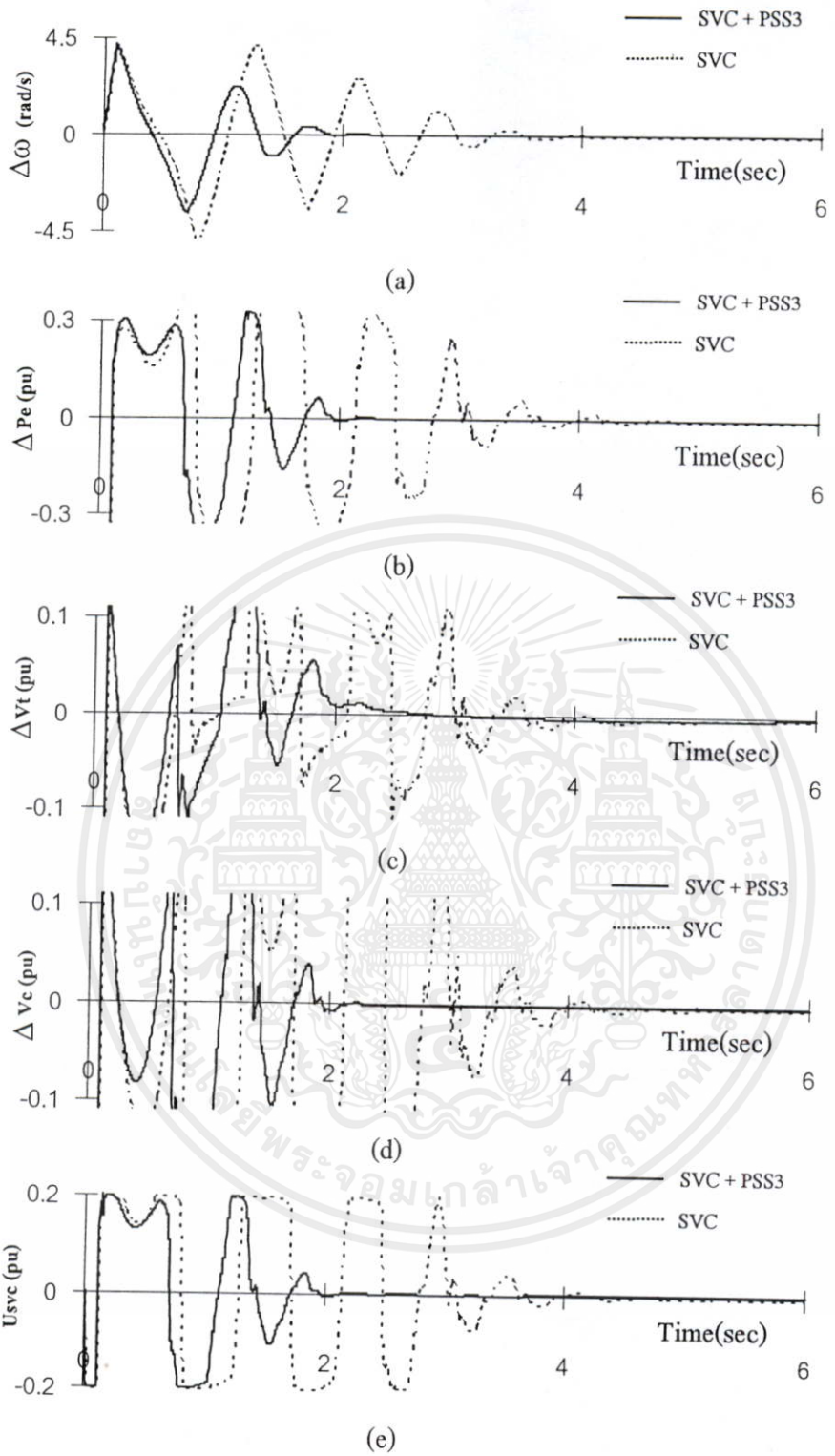
รูปที่ 8.23 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ลิขสิทธิ์เอกสารนี้เป็นของจุฬาลงกรณ์มหาวิทยาลัย ห้ามนำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



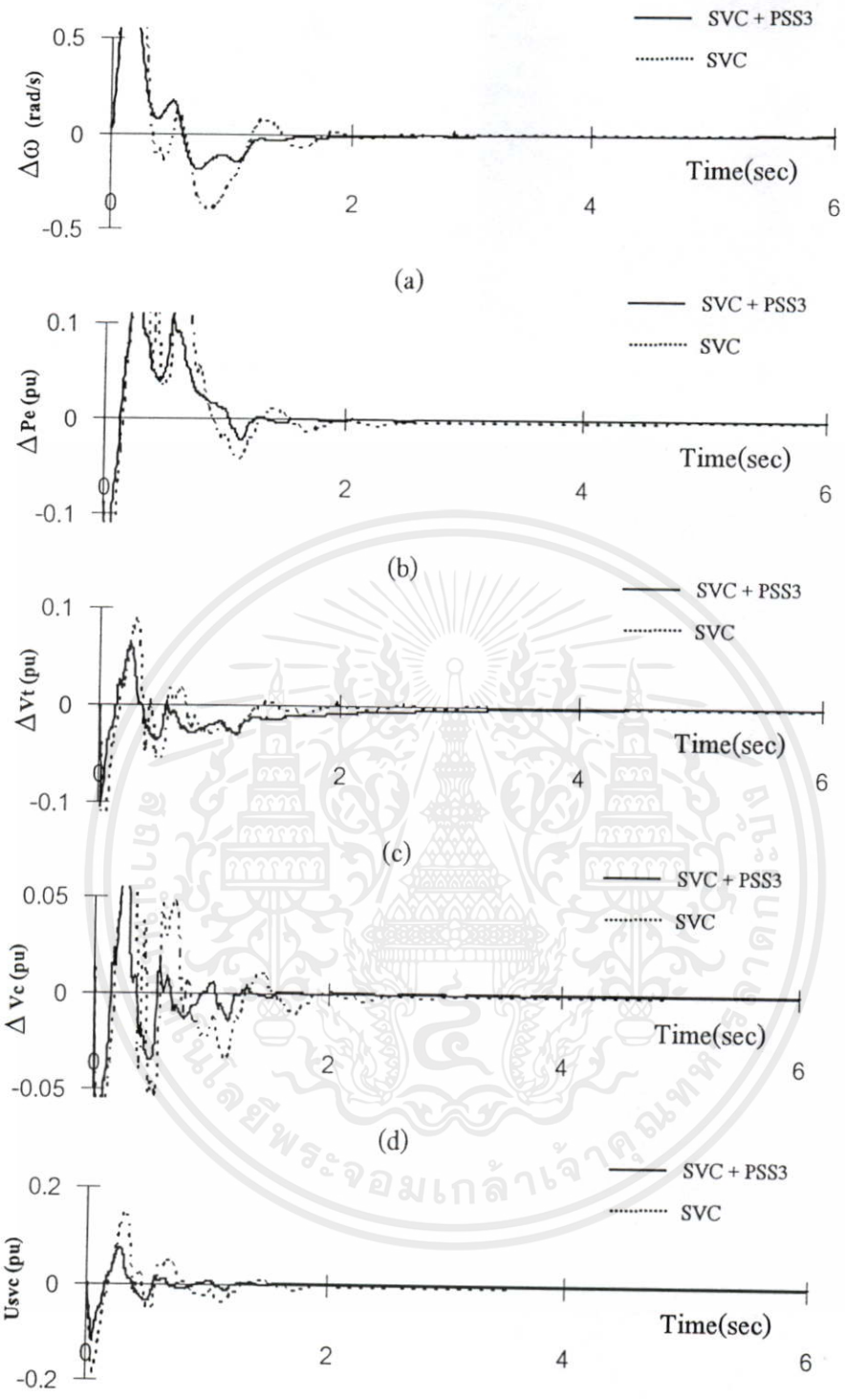
รูปที่ 8.24 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย ว่ากรณีใดๆ ที่ไม่ได้ และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้ ไม่อนุญาตให้ใช้ประโยชน์ด้านการ



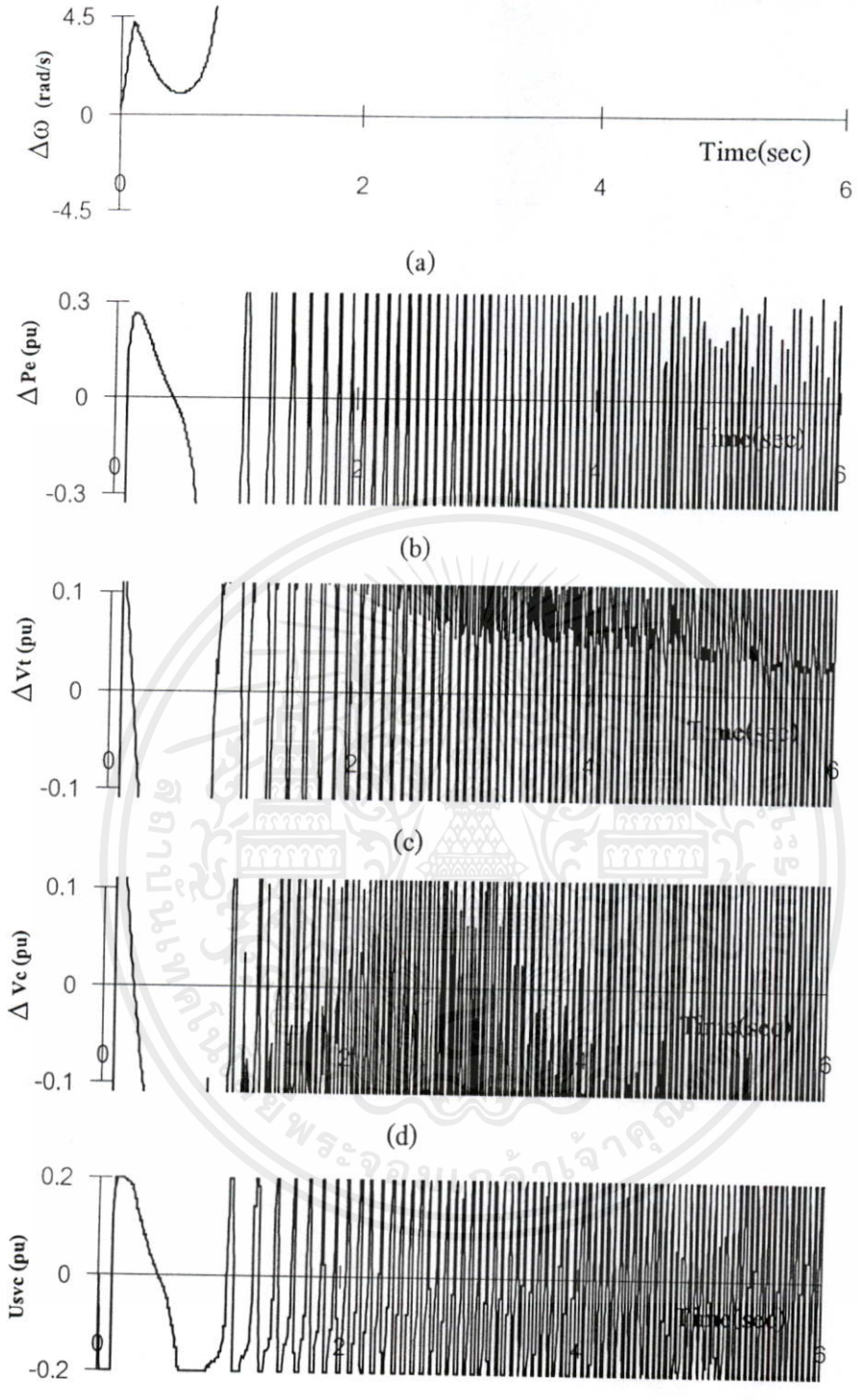
รูปที่ 8.25 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



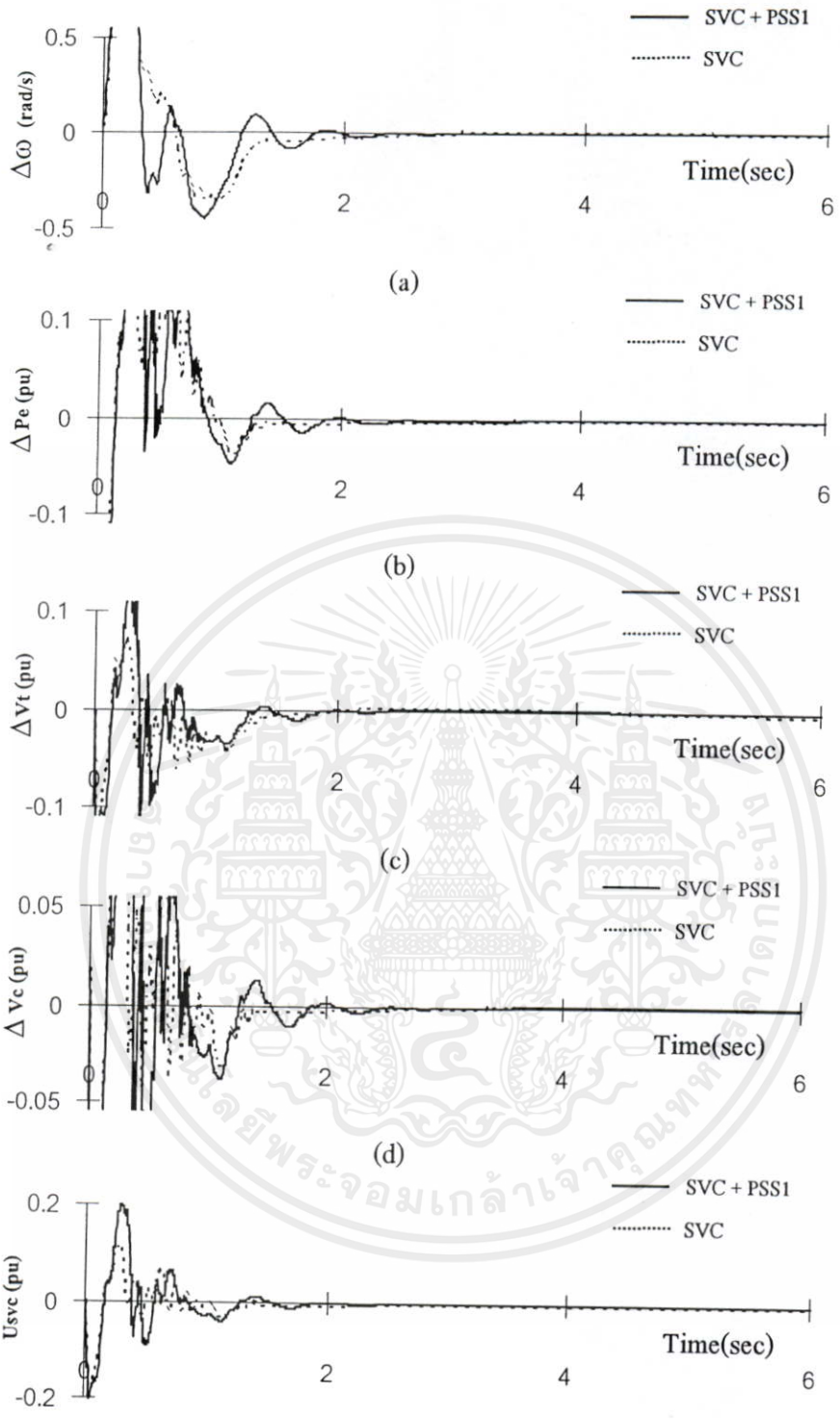
รูปที่ 8.26 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ว่ากรณีใดๆทั้งสิ้น ไม่ว่าการนำไปใช้ประโยชน์ด้านการศึกษา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



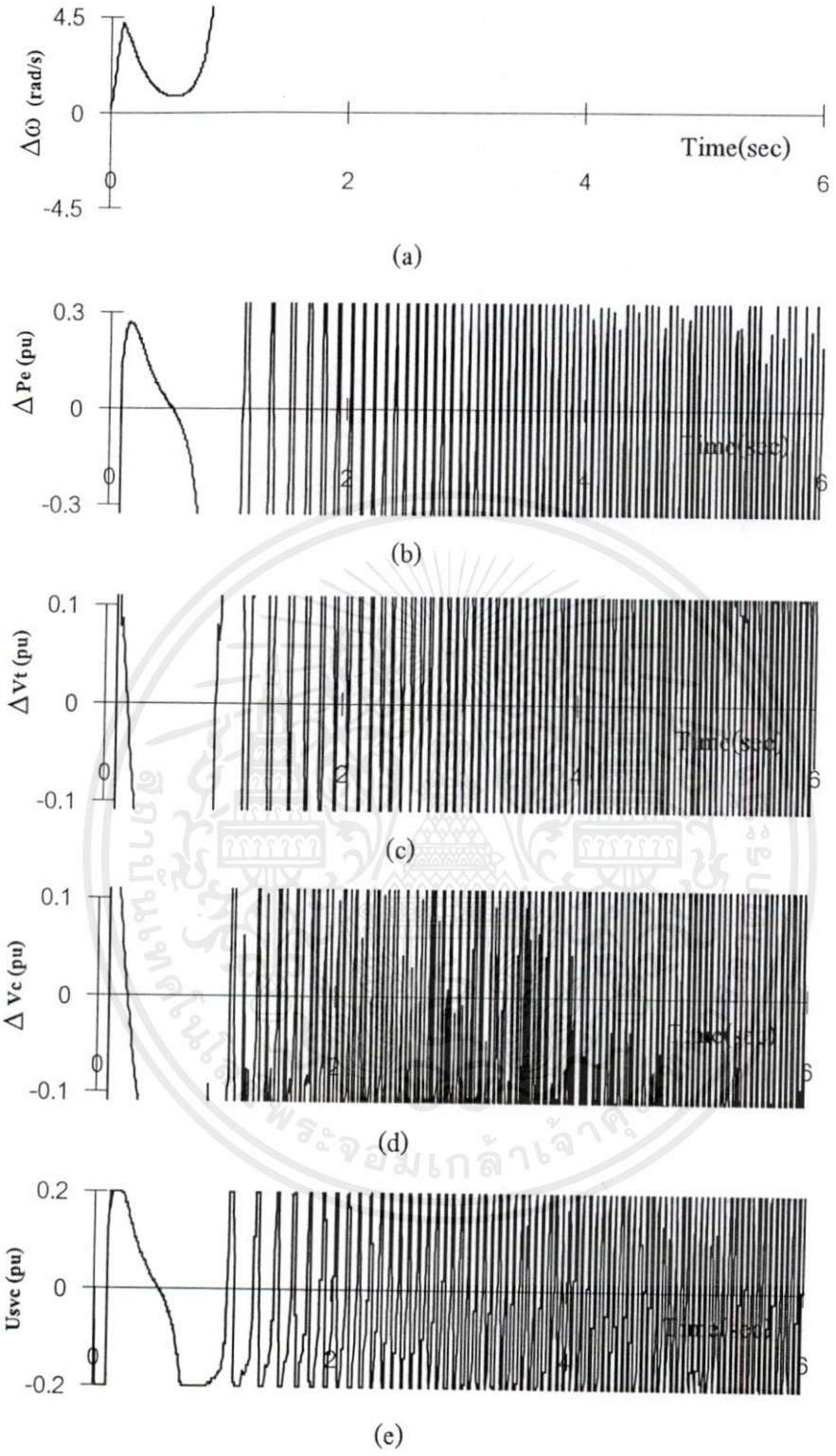
รูปที่ 8.27 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ลิขสิทธิ์มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ว่ากรณีใดๆ ห้ามมิให้ทำซ้ำ ดัดแปลง หรือนำไปใช้ประโยชน์ด้านการค้า และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



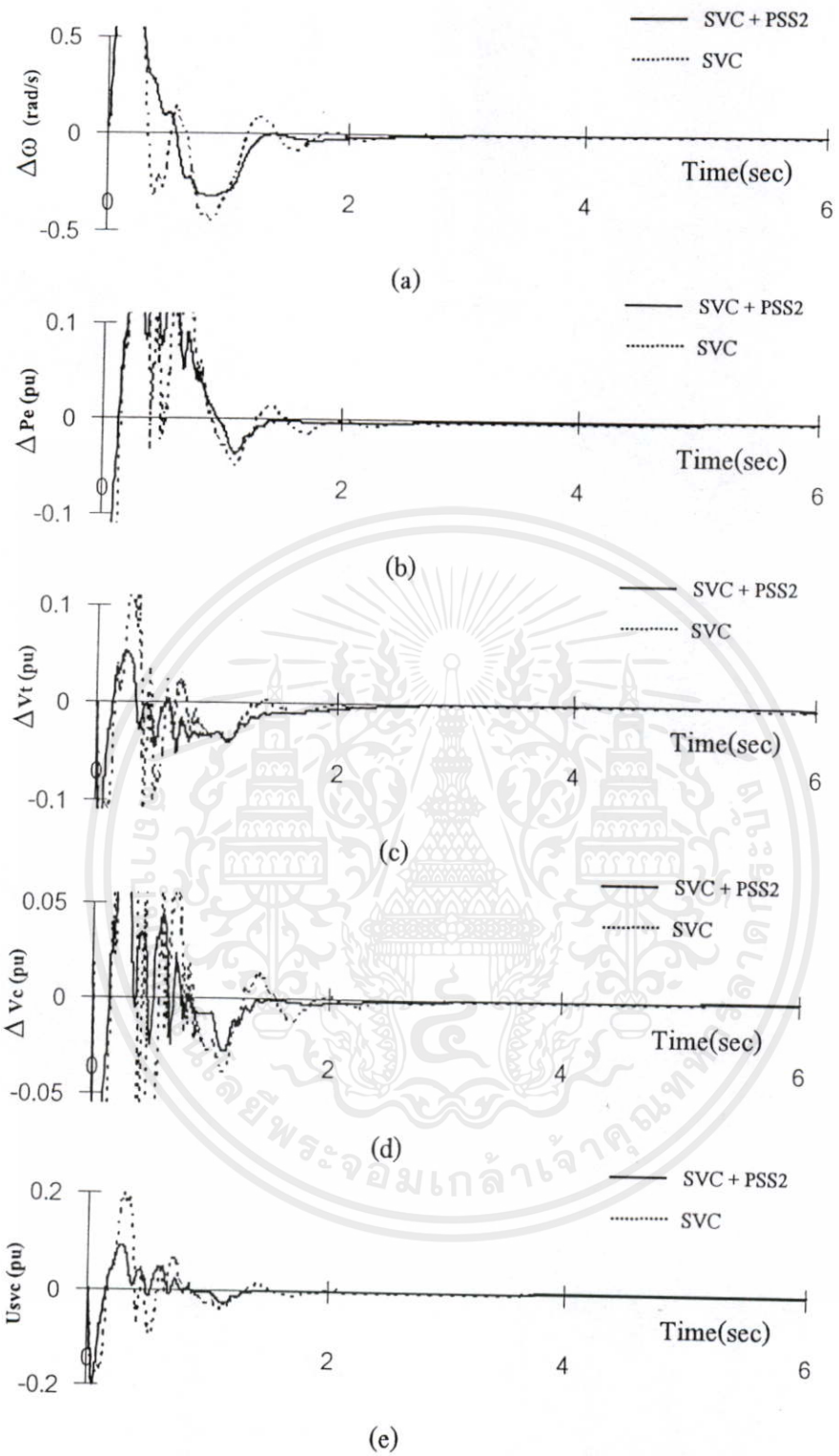
รูปที่ 8.28 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



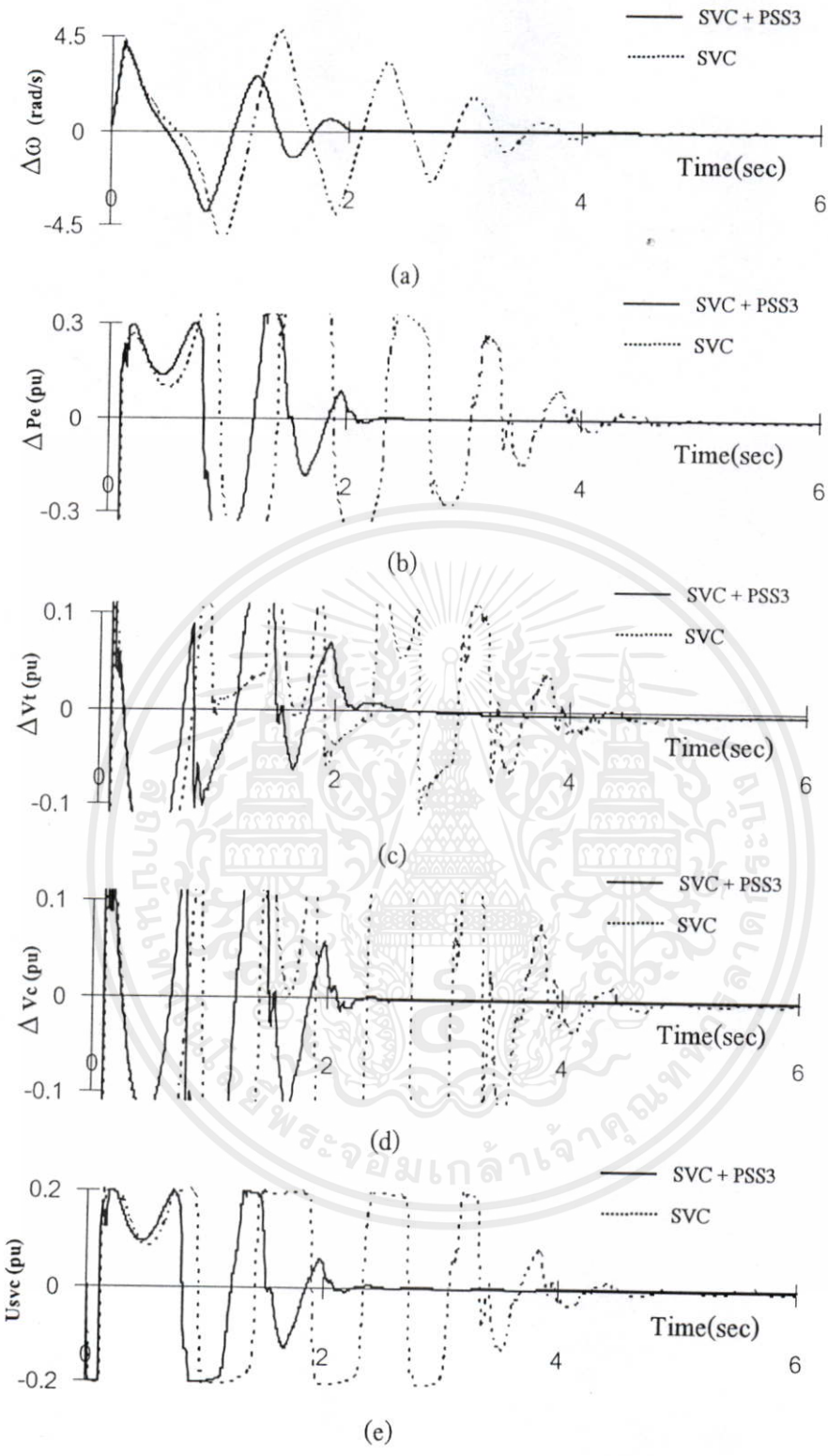
รูปที่ 8.29 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



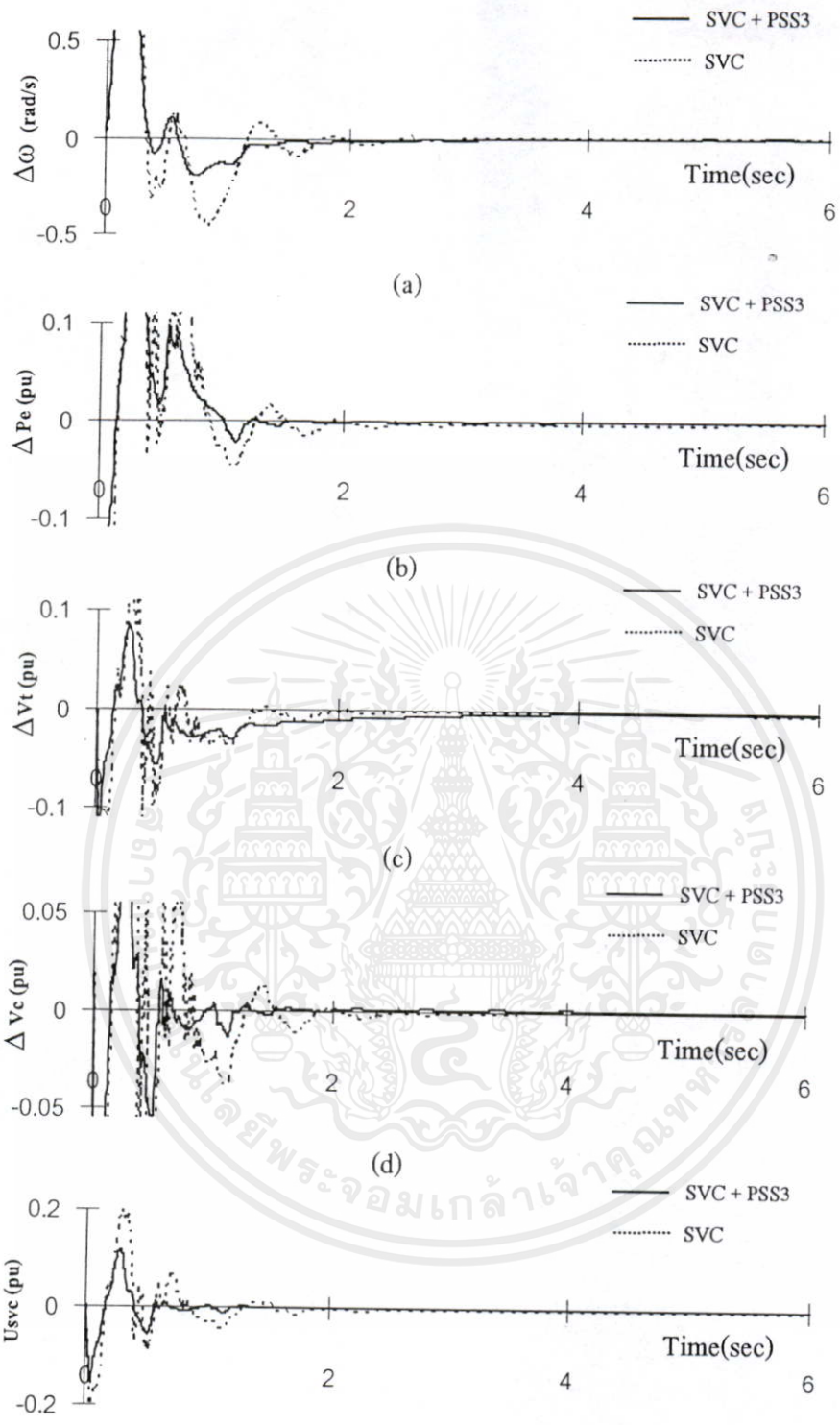
รูปที่ 8.30 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8.31 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8.32 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยวิธีไอเทอร์เรชั่น

สารนี้จัดทำขึ้นเพื่อใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.4.2 ผลการออกแบบด้วยวิธีเจเนติกอัลกอริทึมโดยใช้โปรแกรม GAFCD

ค่าอินพุทเริ่มต้นสำหรับโปรแกรม GAFCD ที่ใช้ในการออกแบบในหัวข้อ 8.4.2 ทั้งหมดนี้จะใช้ค่าตามตารางที่ 8.6

8.4.2.1 กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว

ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว 6 ครั้ง แสดงในตารางที่ 8.7 ผลการออกแบบที่แสดงถึงการพัฒนาคำตอบในแต่ละเจนเนอเรชั่นแสดงในรูปที่ 8.33 ผลตอบสนองโดยใช้ค่าพารามิเตอร์ควบคุมฟัซซี่ที่ให้คำตอบที่ดีที่สุด ต่อการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า แสดงในรูปที่ 8.37 และ 8.38 เมื่อ $X_j = 1.3$ เท่า แสดงในรูปที่ 8.45 และ 8.46

ตารางที่ 8.6 ค่าอินพุทเริ่มต้นโปรแกรม GAFCD

```
Enter the population size ------------> 30
Enter chromosome length (24) ---------> 24
Print chromosome strings? (y/n) ------> n
Enter maximum number of generations --> 50
Enter initial crossover probability ----------> 0.900000
Enter initial mutation probability -----------> 0.010000
Enter Elitist Selection (1:Y | 0:N) --> 1
Enter tournament size for selection --> 2
```

ตารางที่ 8.7 ผลการออกแบบโดยโปรแกรม GAFCD กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว

ครั้งที่	$\alpha 1$	$\alpha 2$	$\alpha 3$	Gen	ฟิทเนส	J
1	0.012	90°	0.24	44	1.1129	2246.3223
2	0.012	89°	0.24	44	1.0982	2276.3669
3	0.012	90°	0.24	19	1.1129	2246.3223
4	0.012	90°	0.24	42	1.1129	2246.3223
5	0.012	90°	0.24	38	1.1129	2246.3223
6	0.012	90°	0.24	41	1.1129	2246.3223

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.4.2.2 กรณีควบคุมแบบผสมผสาน

กรณีนี้จะใช้ SVC ชนิดฟัซซี่ควบคุมร่วมกับ PSS ทั้ง 3 ชนิด โดยผลการออกแบบโดยใช้โปรแกรม GAFCD สำหรับกรณี SVC+PSS1 แสดงในตารางที่ 8.8 กรณี SVC+PSS2 แสดงในตารางที่ 8.9 และกรณี SVC+PSS3 แสดงในตารางที่ 8.10

ผลการออกแบบที่แสดงถึงการพัฒนาคำตอบในแต่ละเจนเนอเรชั่น กรณี SVC+PSS1 แสดงในรูปที่ 8.34 กรณี SVC+PSS2 แสดงในรูปที่ 8.35 และกรณี SVC+PSS3 แสดงในรูปที่ 8.36 ผลตอบสนองโดยใช้ค่าพารามิเตอร์ควบคุมฟัซซี่ที่ให้คำตอบที่ดีที่สุด ต่อการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า กรณี SVC+PSS1 แสดงในรูปที่ 8.39 และ 8.40 เมื่อ $X_j = 1.3$ เท่า แสดงในรูปที่ 8.47 และ 8.48 เมื่อ $X_j = 1.0$ เท่า กรณี SVC+PSS2 แสดงในรูปที่ 8.41 และ 8.42 เมื่อ $X_j = 1.3$ เท่า แสดงในรูปที่ 8.49 และ 8.50 และเมื่อ $X_j = 1.0$ กรณี SVC+PSS3 แสดงในรูปที่ 8.43 และ 8.44 เมื่อ $X_j = 1.3$ เท่า แสดงในรูปที่ 8.51 และ 8.52

ตารางที่ 8.8 ผลการออกแบบโดยโปรแกรม GAFCD กรณี SVC+PSS1

ครั้งที่	$\alpha 1$	$\alpha 2$	$\alpha 3$	Gen	ฟิทเนส	J
1	0.01	90°	0.23	37	1.9727	1267.3309
2	0.01	90°	0.23	24	1.9727	1267.3309
3	0.01	90°	0.23	26	1.9727	1267.3309
4	0.01	90°	0.23	41	1.9727	1267.3309
5	0.01	90°	0.23	22	1.9727	1267.3309
6	0.01	90°	0.23	30	1.9727	1267.3309

ตารางที่ 8.9 ผลการออกแบบโดยโปรแกรม GAFCD กรณี SVC+PSS2

ครั้งที่	$\alpha 1$	$\alpha 2$	$\alpha 3$	Gen	ฟิทเนส	J
1	0.01	85°	0.24	4	2.2942	1089.6946
2	0.01	85°	0.24	40	2.2942	1089.6946
3	0.01	85°	0.24	18	2.2942	1089.6946
4	0.01	85°	0.24	19	2.2942	1089.6946
5	0.01	85°	0.24	48	2.2942	1089.6946
6	0.01	85°	0.24	46	2.2942	1089.6946

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 8.10 ผลการออกแบบโดยโปรแกรม GAFCD กรณี SVC+PSS3

ครั้งที่	$\alpha 1$	$\alpha 2$	$\alpha 3$	Gen	ฟิทเนส	J
1	0.012	89°	0.29	34	3.0838	810.6910
2	0.012	89°	0.29	34	3.0838	810.6910
3	0.012	89°	0.29	38	3.0838	810.6910
4	0.012	89°	0.29	13	3.0838	810.6910
5	0.012	89°	0.29	29	3.0838	810.6910
6	0.012	89°	0.29	24	3.0838	810.6910

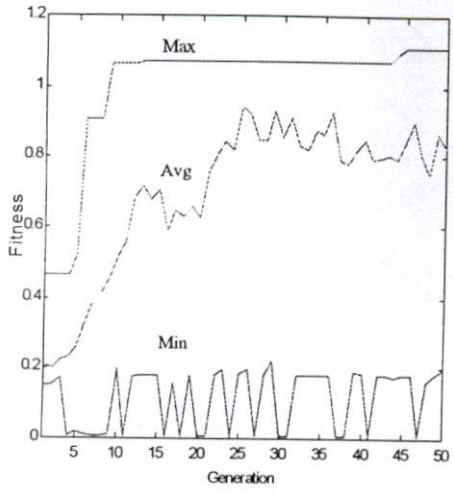
8.4.3 ผลเปรียบเทียบการออกแบบโดยวิธีไอเทอร์เรชั่นกับการออกแบบโดยใช้โปรแกรม GAFCD

ค่าพารามิเตอร์ควบคุมฟัซซี่ได้จากจากการออกแบบโดยวิธีไอเทอร์เรชั่นในหัวข้อ 8.4.1.1 และ 8.4.1.2 และที่ได้จากการออกแบบโดยใช้โปรแกรม GAFCD ในหัวข้อ 8.4.2.1 และ 8.4.2.2 สามารถนำมาเปรียบเทียบได้ดังตารางที่ 8.11 ซึ่งจะพบว่าวิธีออกแบบโดยใช้โปรแกรม GAFCD ให้ผลตอบสนองที่ดีกว่าหรือเท่ากับการออกแบบด้วยวิธีไอเทอร์เรชั่น

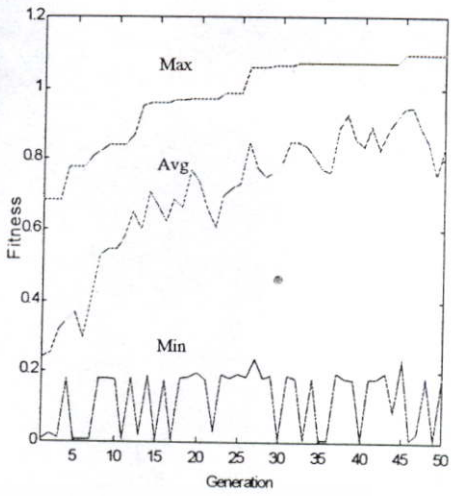
ตารางที่ 8.11 ผลเปรียบเทียบการออกแบบโดยใช้วิธีไอเทอร์เรชั่นกับการออกแบบโดยใช้โปรแกรม GAFCD

วิธีควบคุม	วิธีออกแบบ	α_1	α_2	α_3	J (X_j=1.0)	J (X_j=1.3)
SVC	ไอเทอร์เรชั่น	0.01	90°	0.23	2313.1667	3114.2698
	GA	0.012	90°	0.24	2246.3223	2914.5310
SVC+PSS1	ไอเทอร์เรชั่น	0.01	90°	0.23	1267.3309	387800.3114
	GA	0.01	90°	0.23	1267.3309	387800.3114
SVC+PSS2	ไอเทอร์เรชั่น	0.01	85°	0.24	1089.6945	408544.8784
	GA	0.01	85°	0.24	1089.6945	408544.8784
SVC+PSS3	ไอเทอร์เรชั่น	0.01	90°	0.29	819.8666	984.7628
	GA	0.012	89°	0.29	810.6910	961.5888

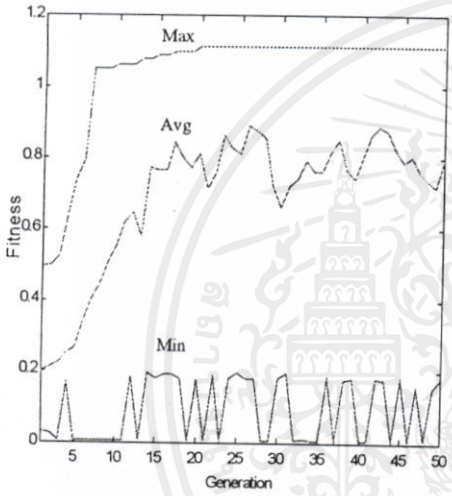
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



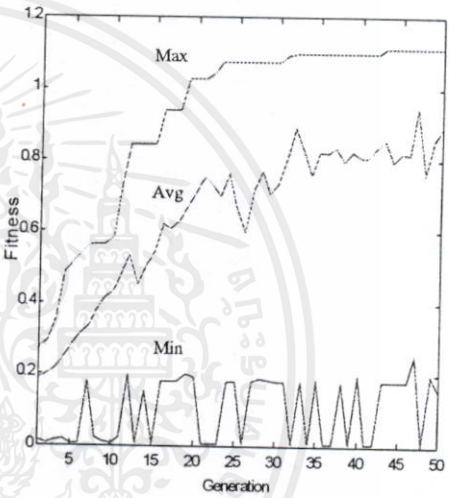
(1)



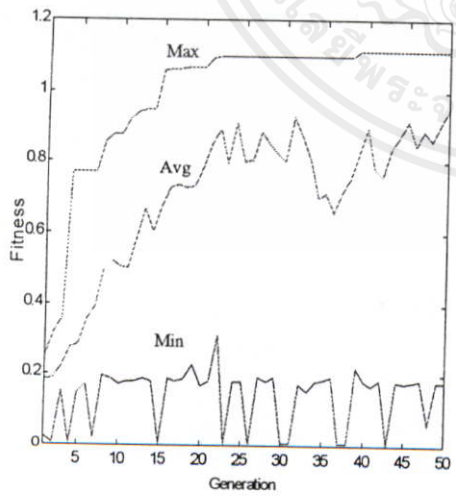
(2)



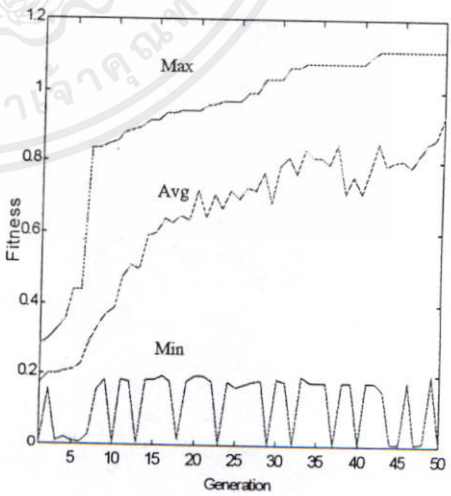
(3)



(4)



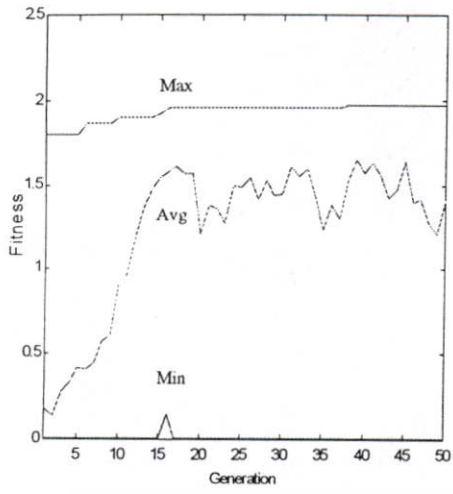
(5)



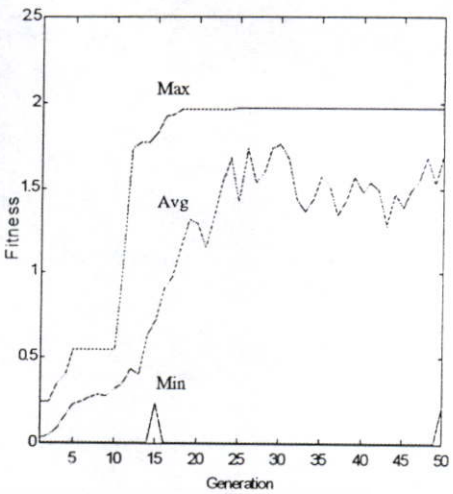
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

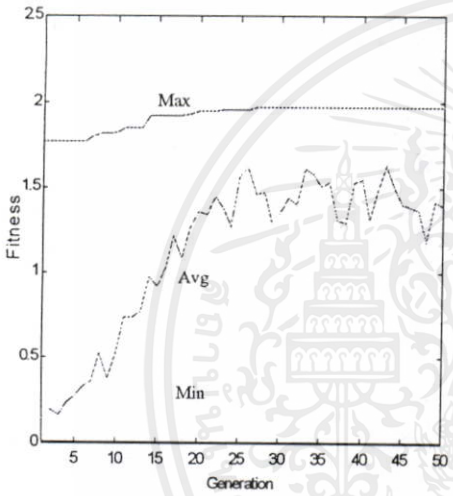
รูปที่ 8.33 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC ชนิดฟัซซี่อย่างเดียว



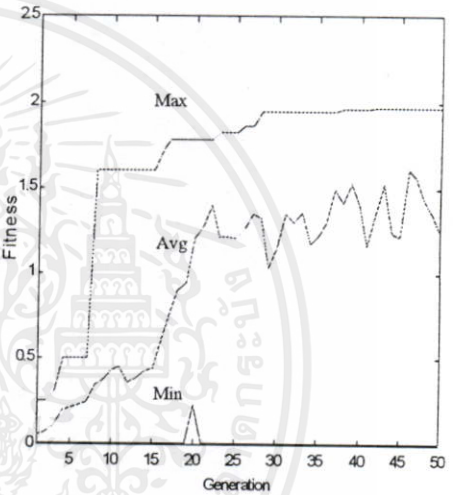
(1)



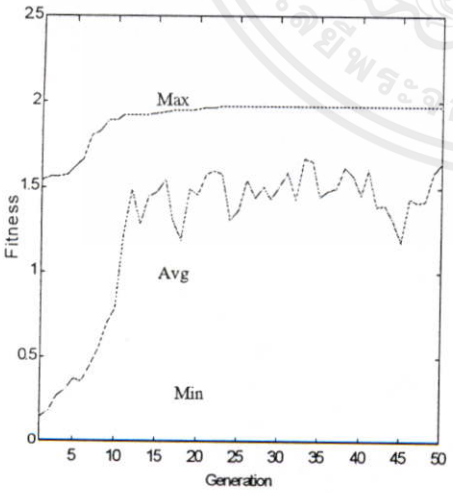
(2)



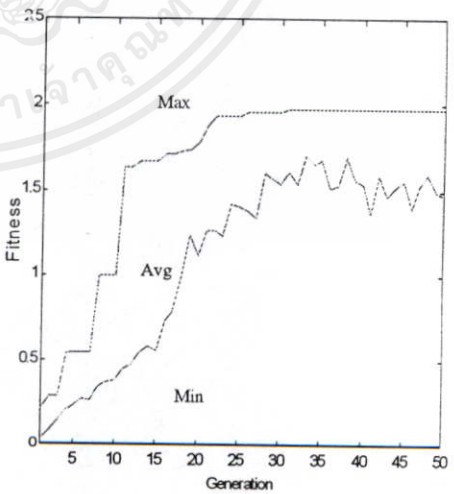
(3)



(4)



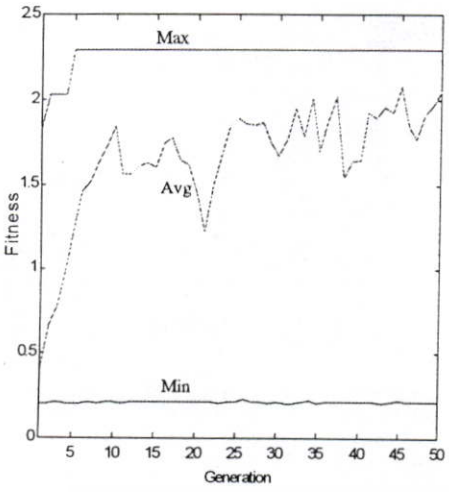
(5)



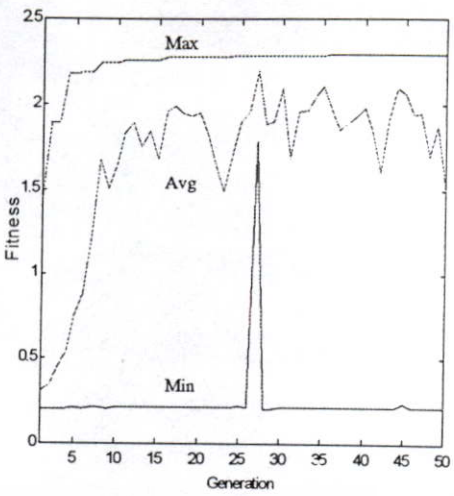
(6)

รูปที่ 8.34 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS1

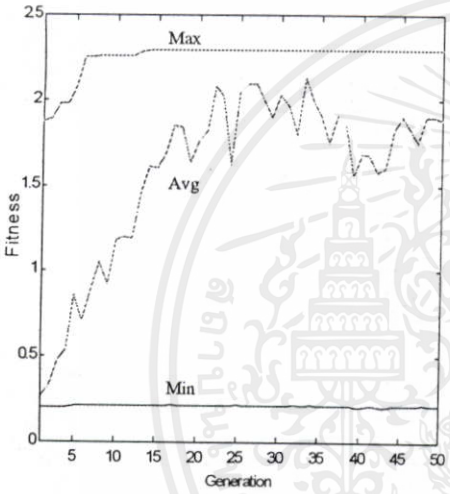
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



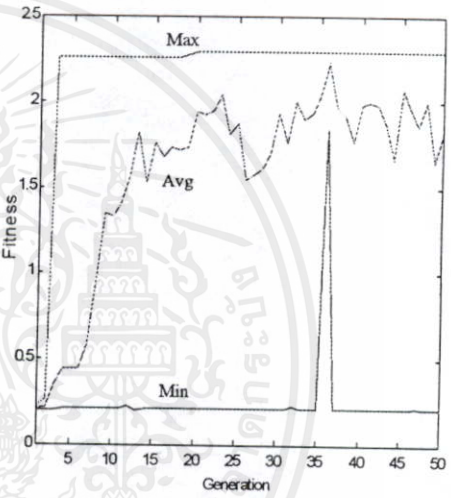
(1)



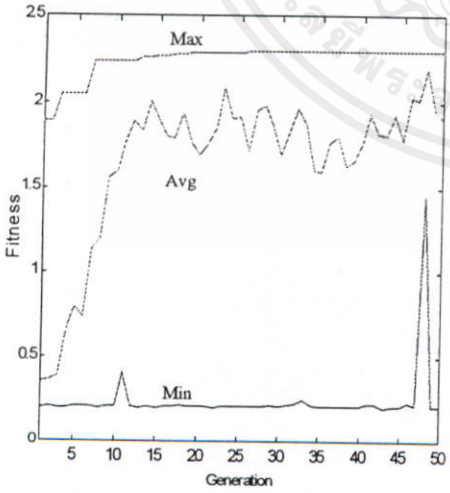
(2)



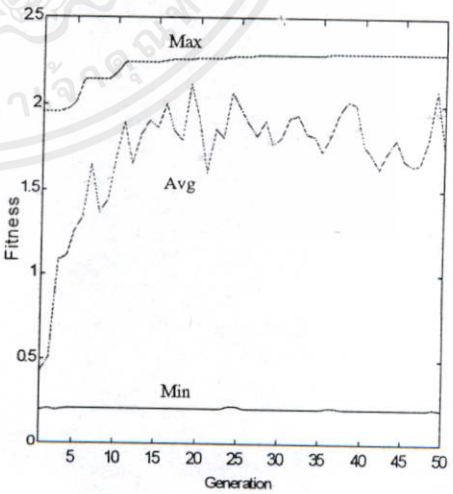
(3)



(4)



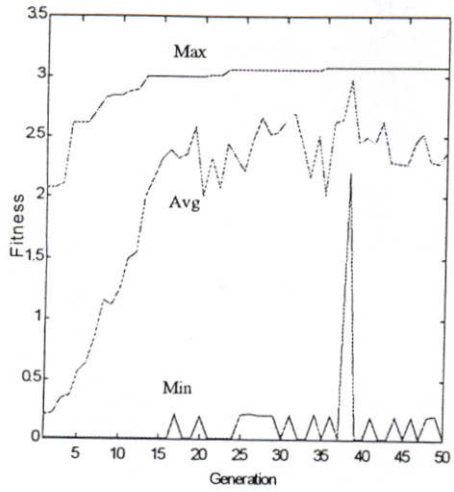
(5)



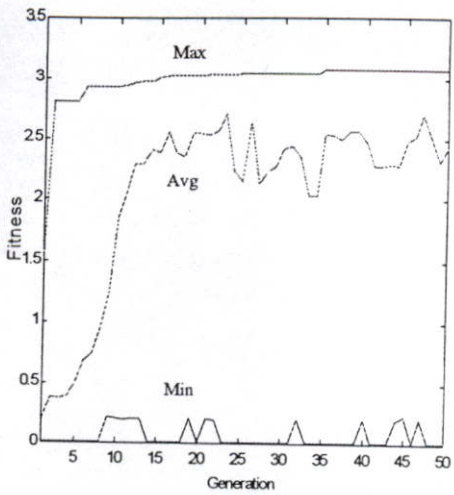
(6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

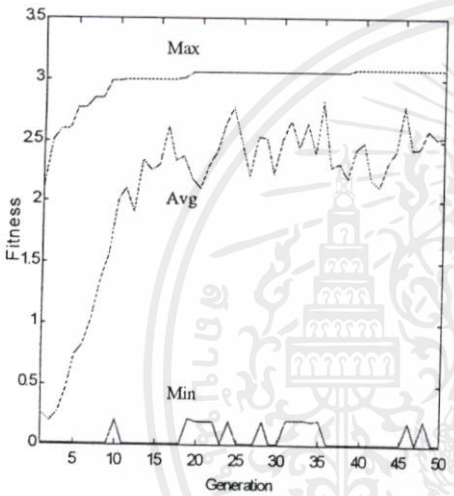
รูปที่ 8.35 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS2



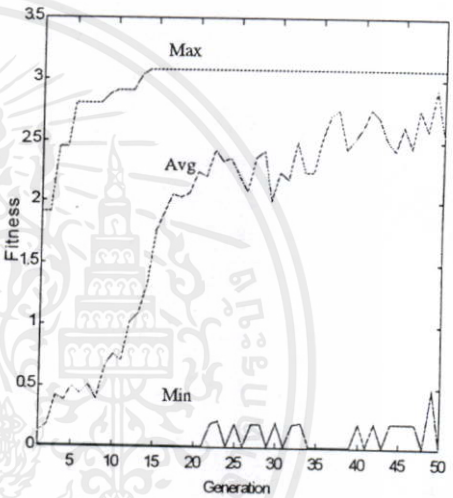
(1)



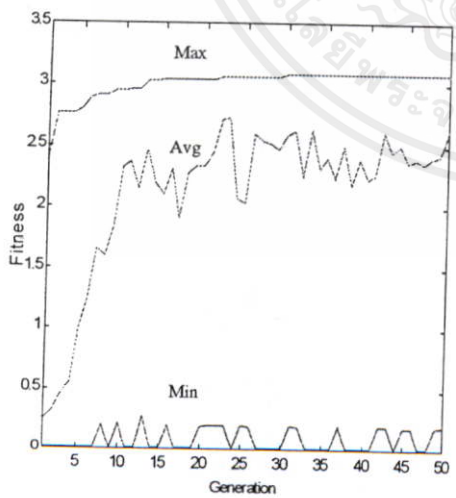
(2)



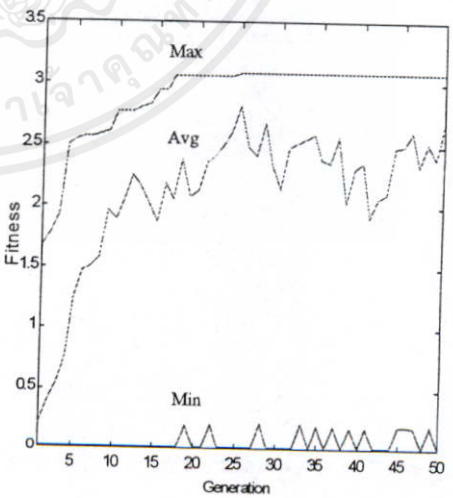
(3)



(4)



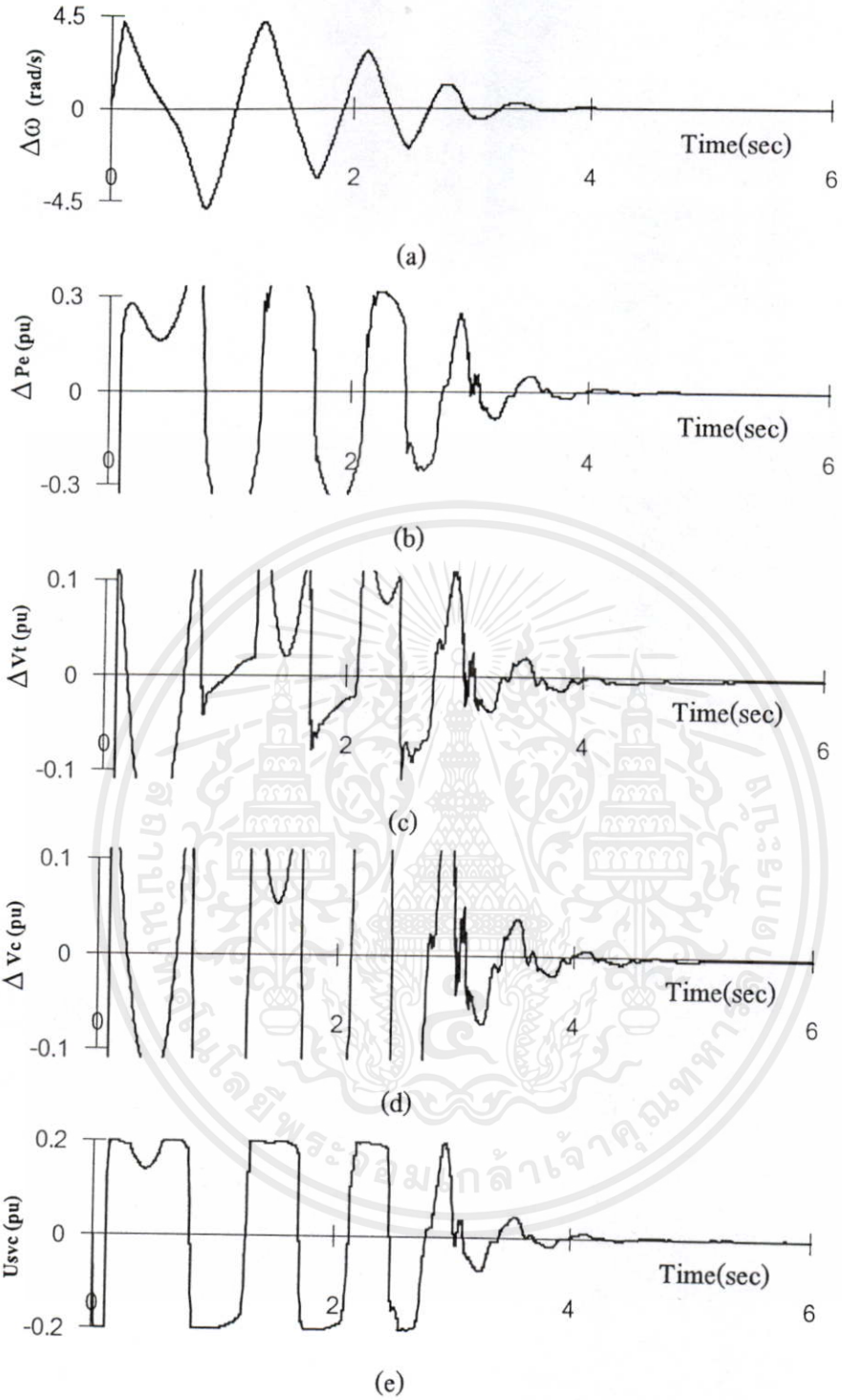
(5)



(6)

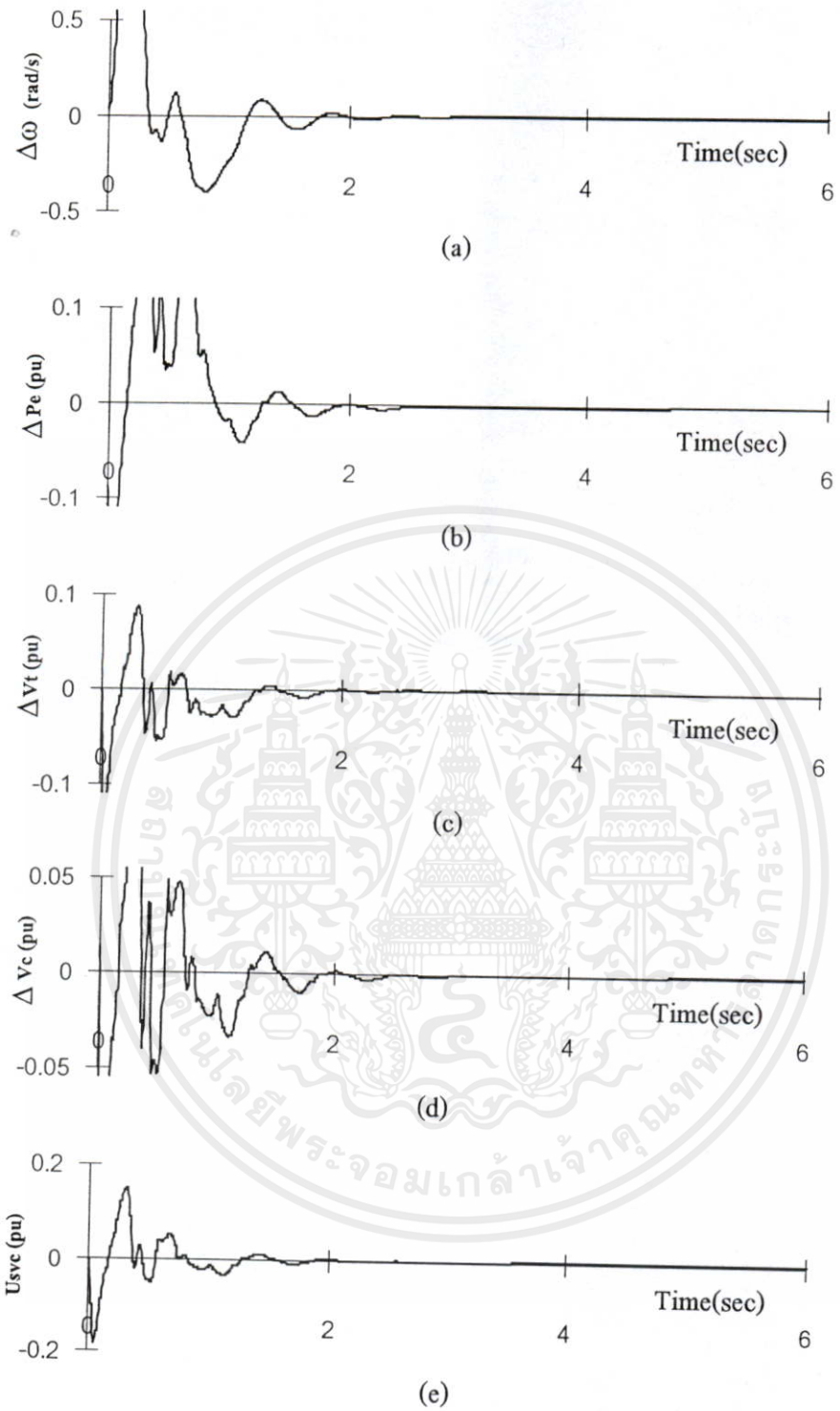
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 8.36 ผลการออกแบบด้วยโปรแกรม GAFCD กรณีควบคุมด้วย SVC+PSS3



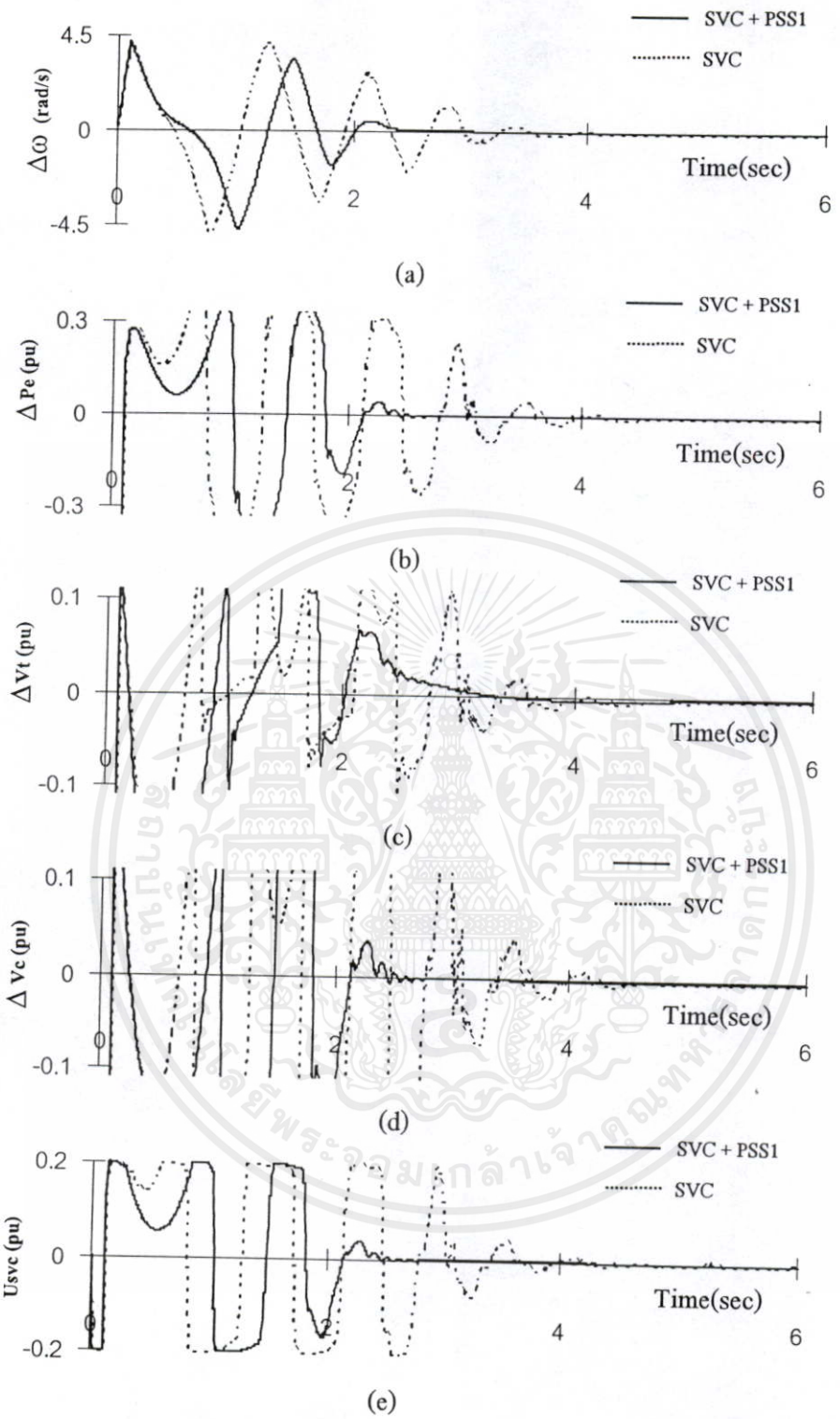
รูปที่ 8.37 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



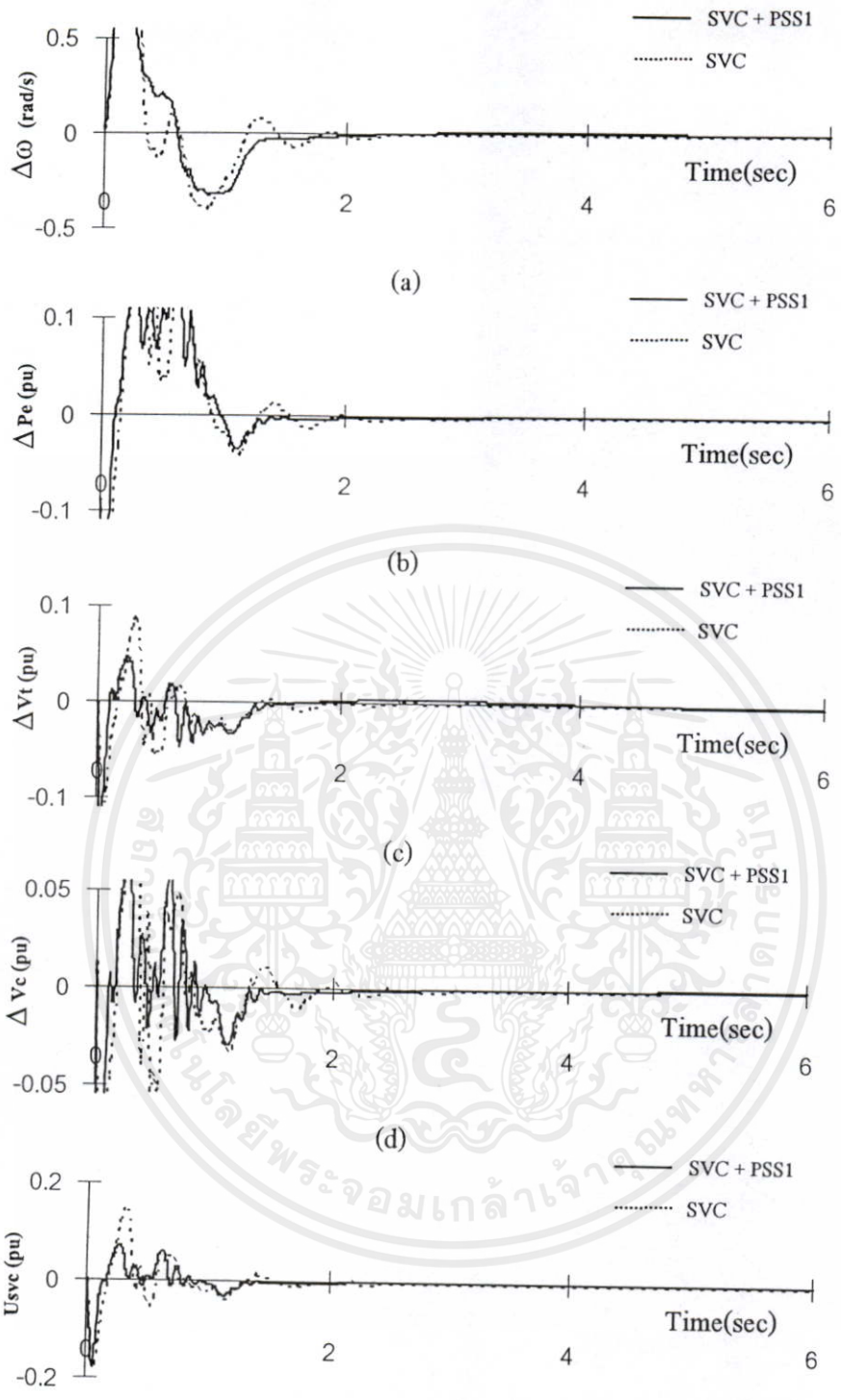
รูปที่ 8.38 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



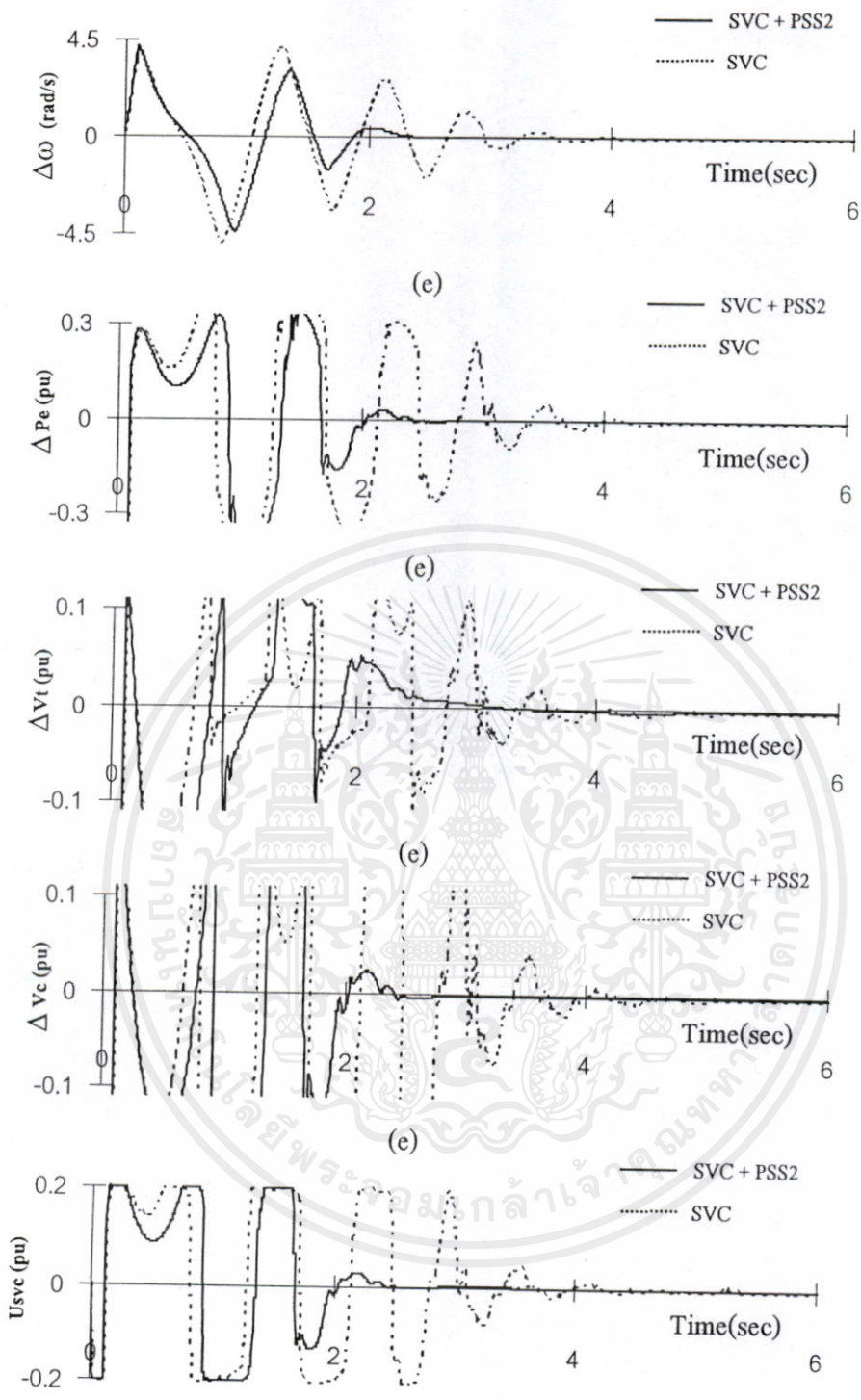
รูปที่ 8.39 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

ลิขสิทธิ์ของเอกสารนี้เป็นของจุฬาลงกรณ์มหาวิทยาลัย ห้ามมิให้ทำซ้ำ ดัดแปลง หรือนำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยไม่ได้รับอนุญาตเป็นลายลักษณ์อักษร และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



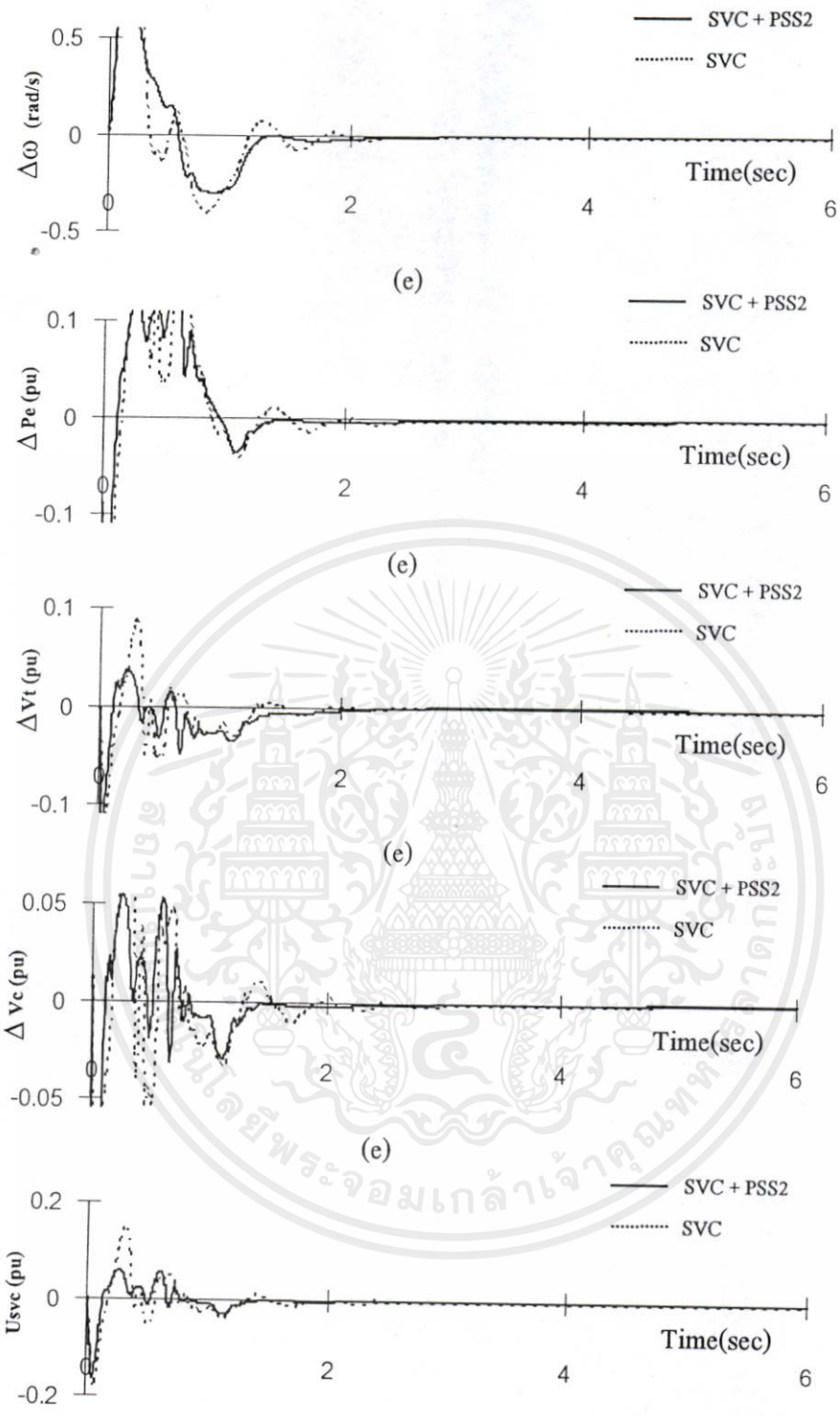
รูปที่ 8.40 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



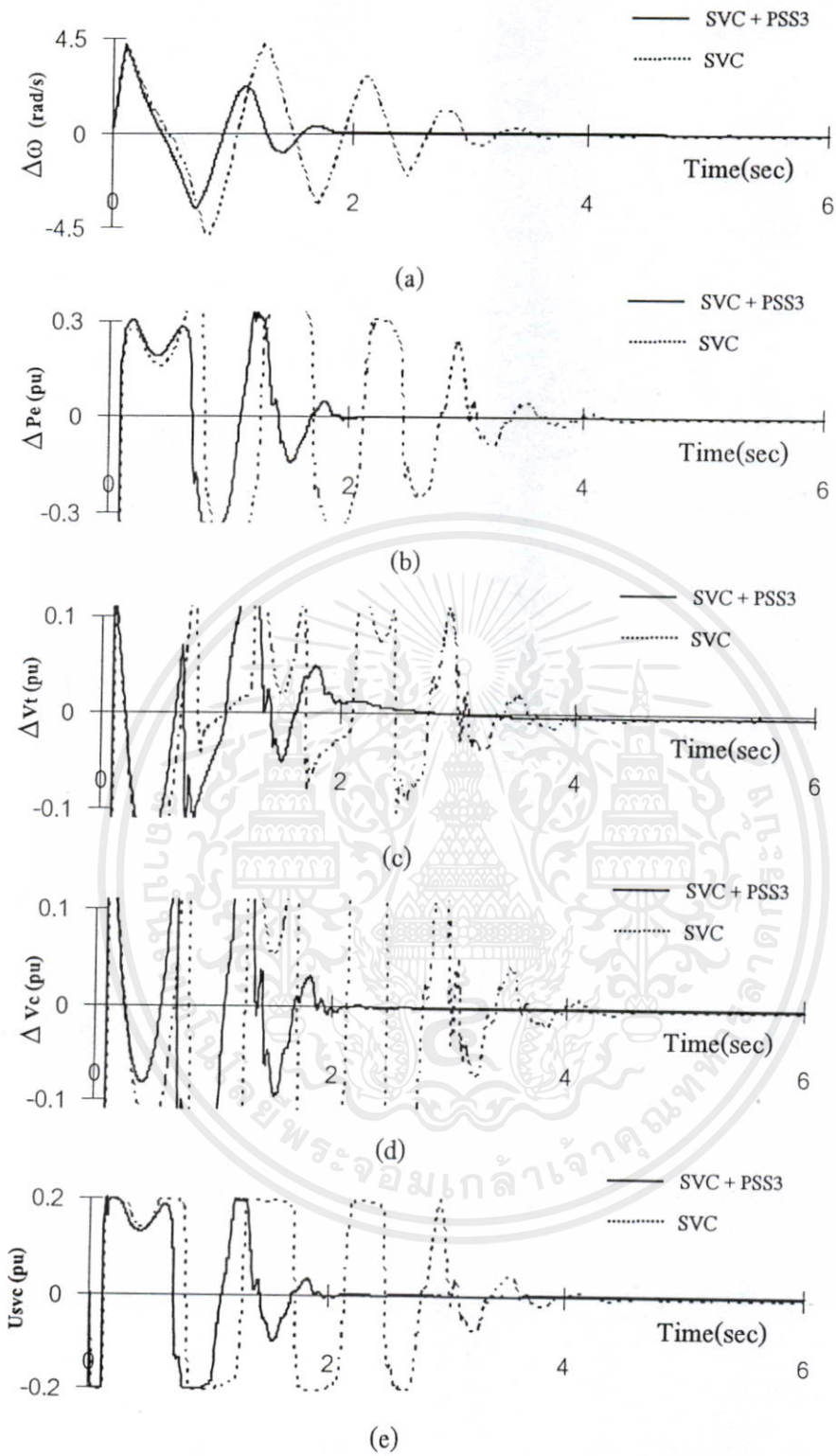
รูปที่ 8.41 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



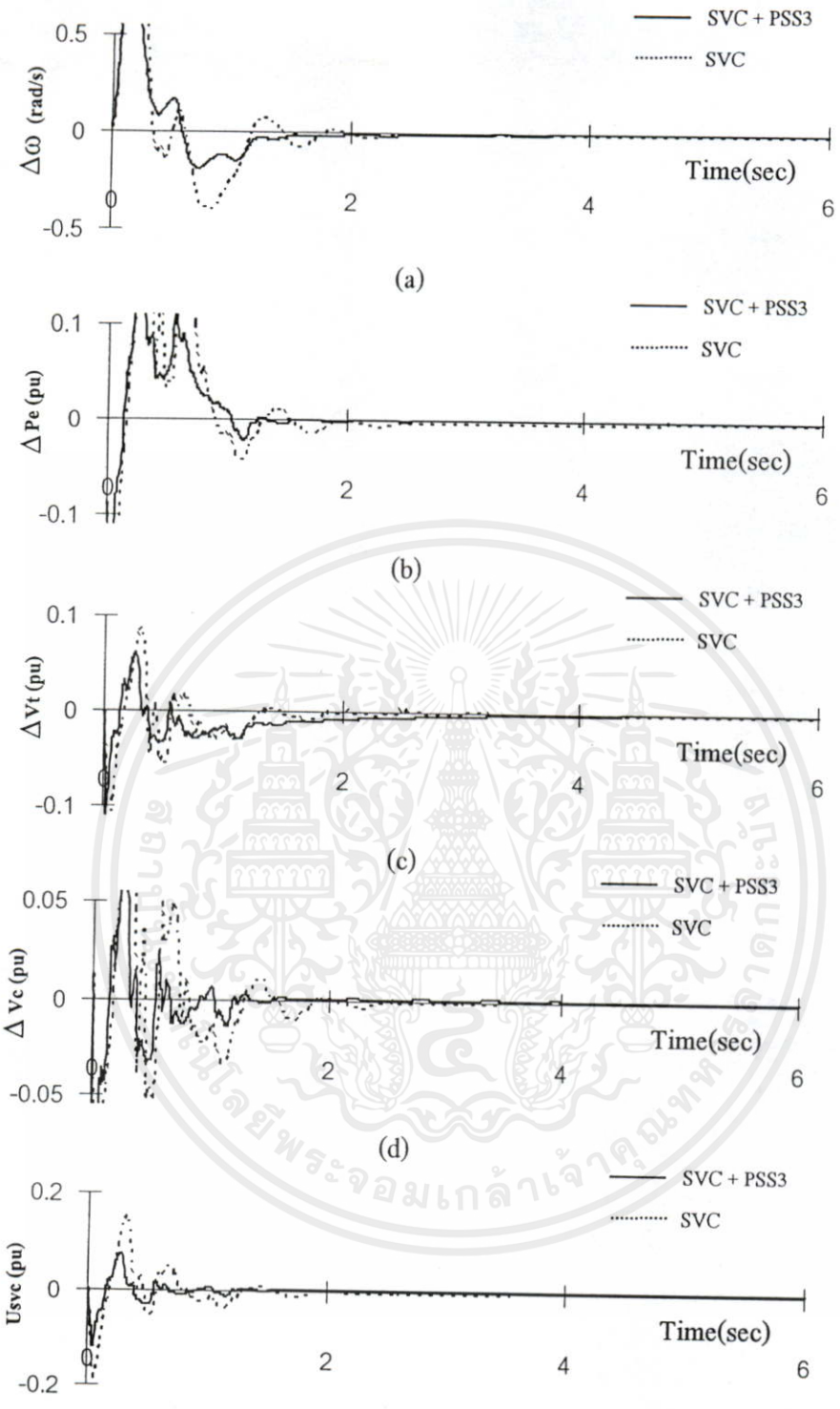
รูปที่ 8.42 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

ว่ากรณีใดๆทั้งสิ้น ไม่ว่าการดัดแปลงนั้นจะเป็นประโยชน์ต่อผู้ใช้งานหรือไม่ก็ตาม และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

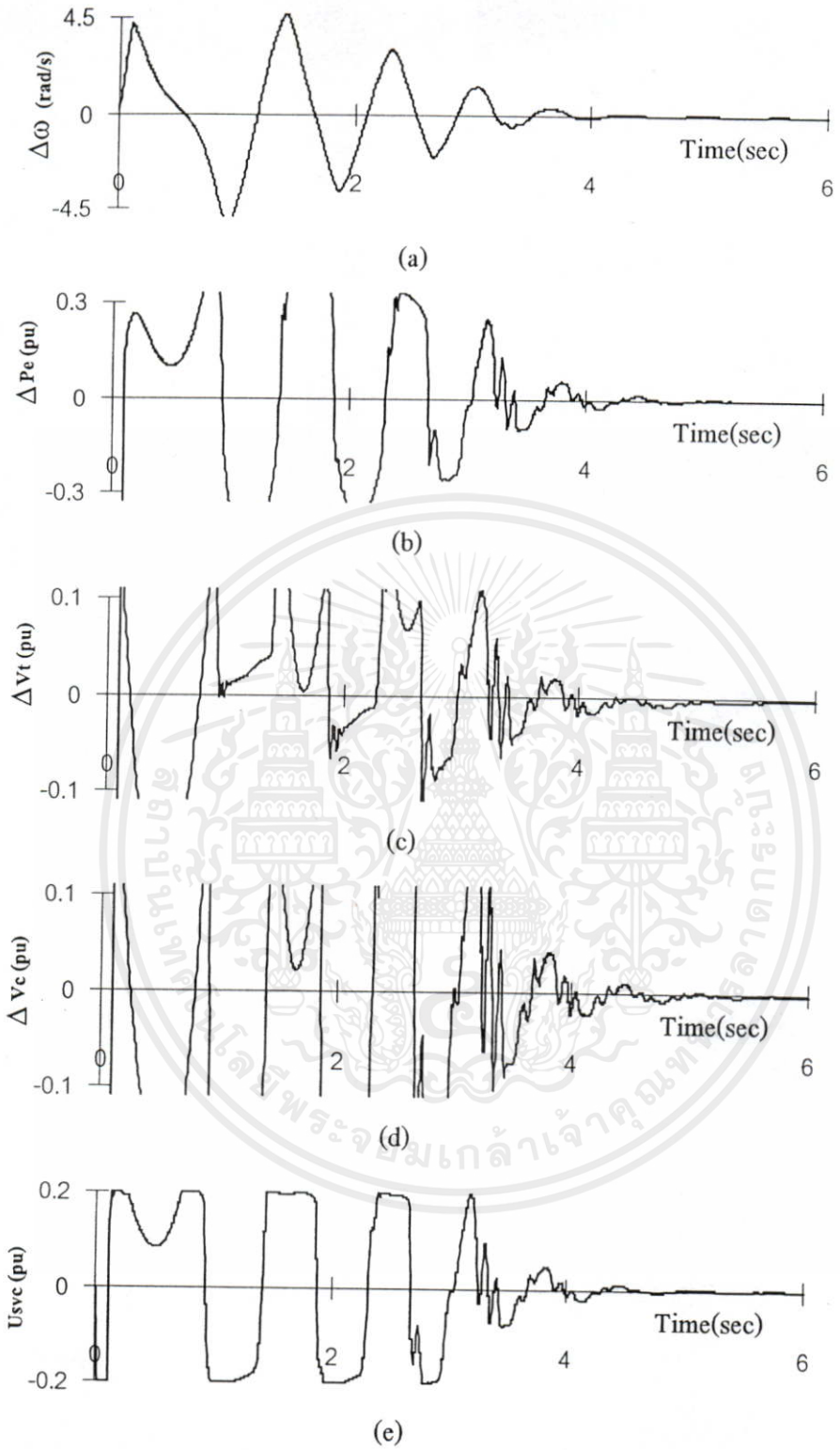


รูปที่ 8.43 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.0$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



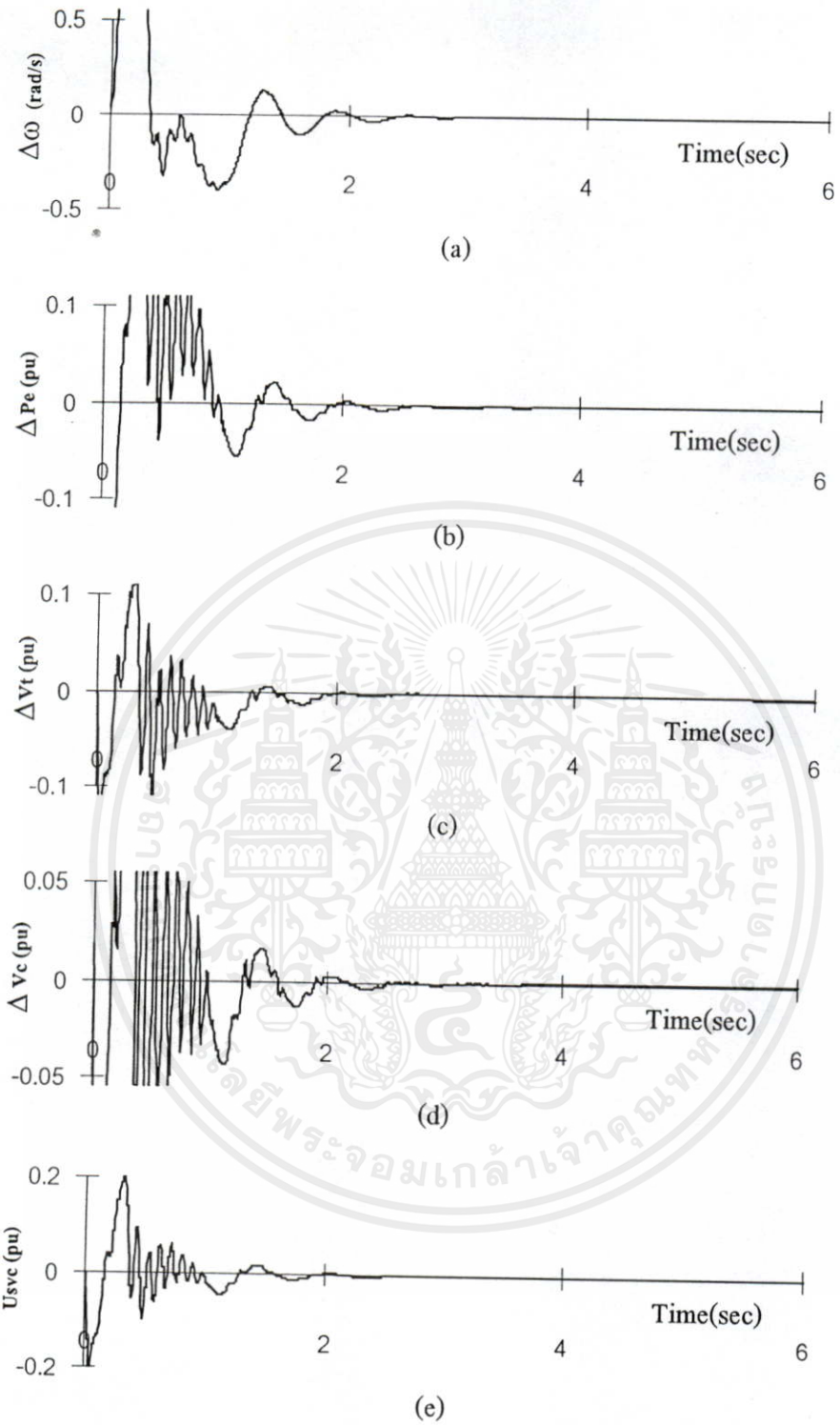
รูปที่ 8.44 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.0 เท่า ออกแบบด้วยโปรแกรม GAFCD



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

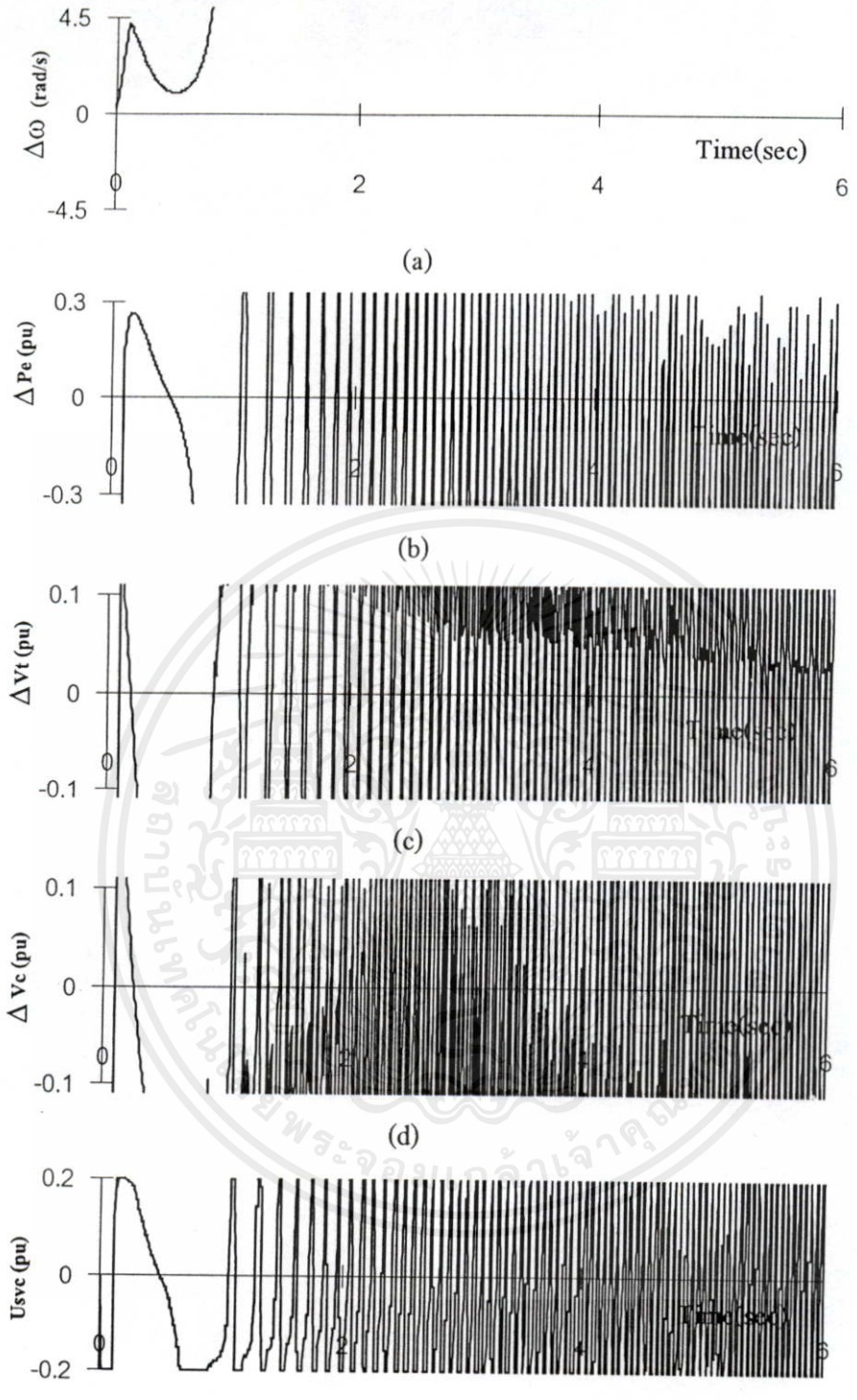
รูปที่ 8.45 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยโปรแกรม GAFCD

ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



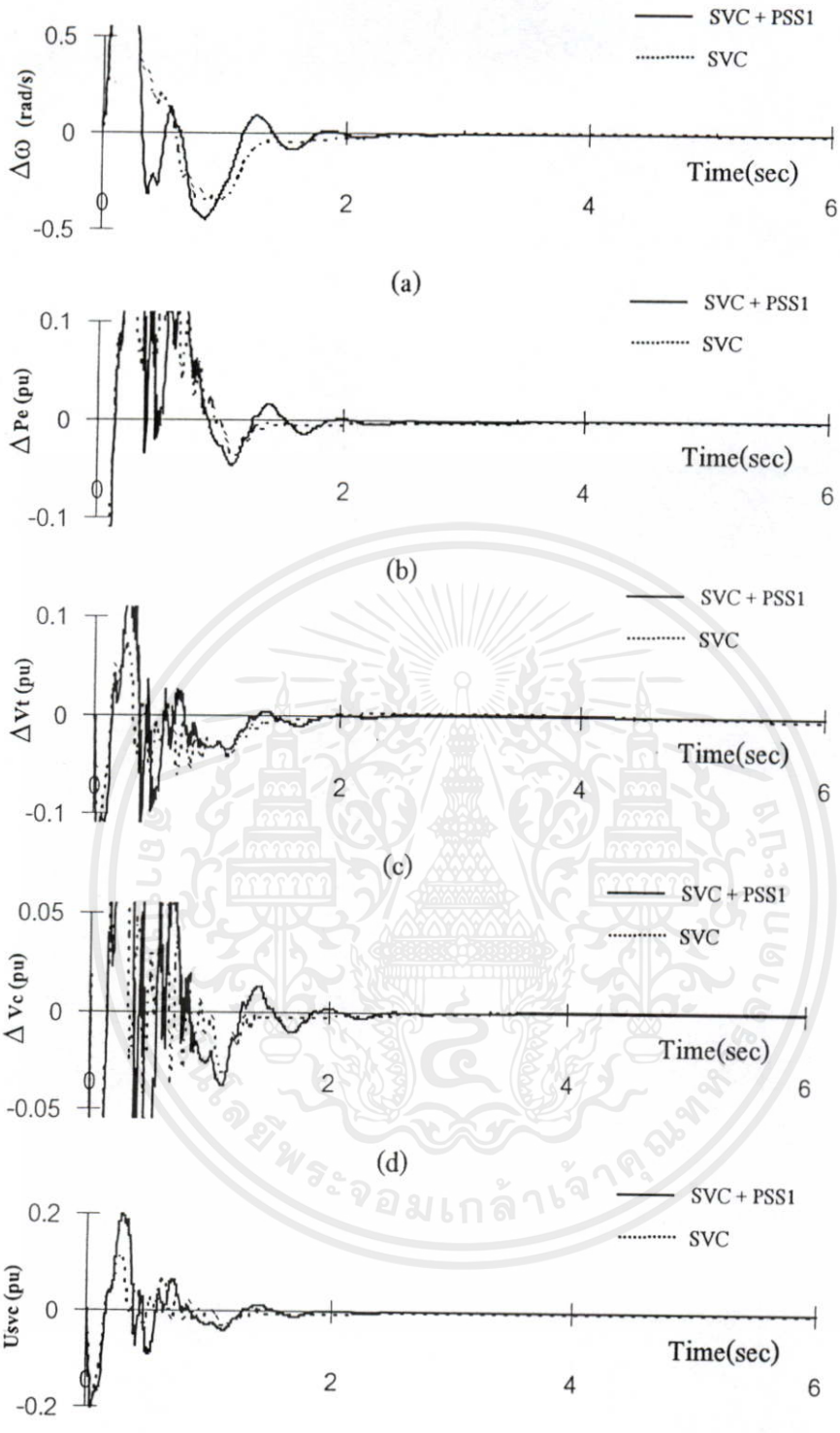
รูปที่ 8.46 ผลตอบสนองกรณีควบคุมด้วย SVC ชนิดฟัซซี่ ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD

ว่ากรณีใดๆ ที่ไม่ได้รับอนุญาตให้ใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ



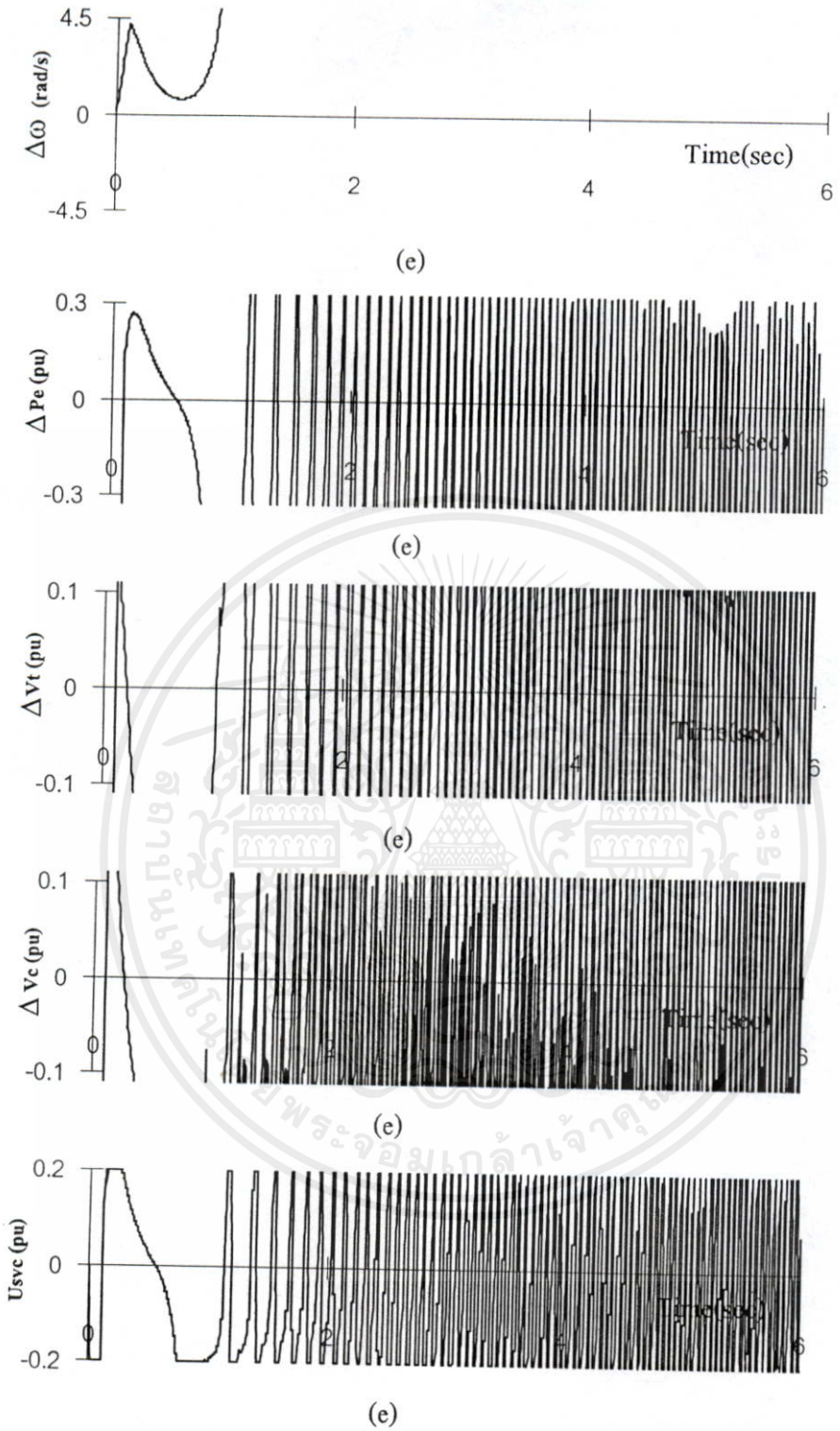
รูปที่ 8.47 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยโปรแกรม GAFCD

ลิขสิทธิ์มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ว่ากรณีใดๆทั้งสิ้น การนำไปใช้ประโยชน์ด้านการศึกษาหรือการค้นคว้าวิจัยต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



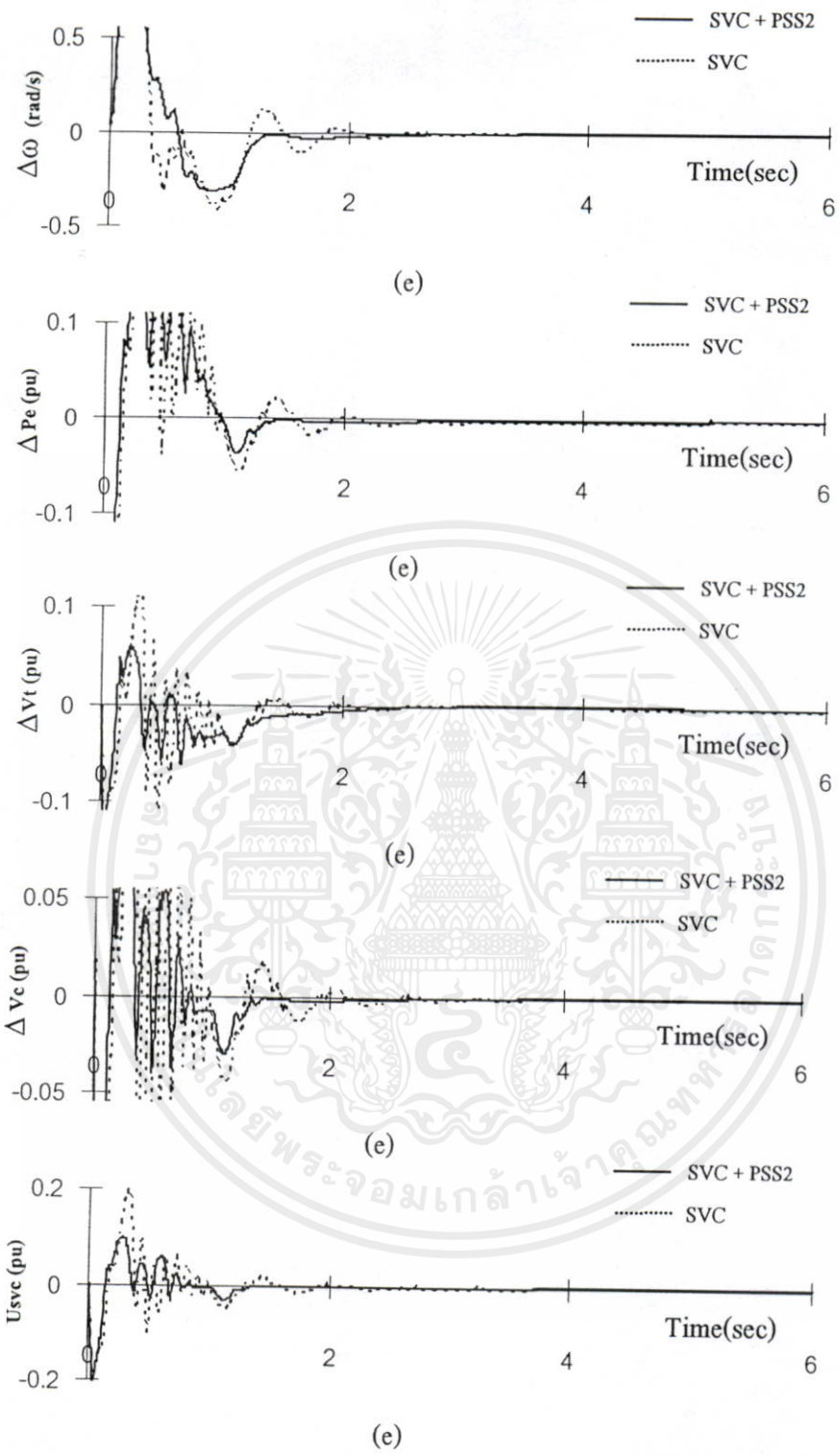
รูปที่ 8.48 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS1 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD

ว่ากรณีใดๆ ที่ได้ทำขึ้นเพื่อใช้ในการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



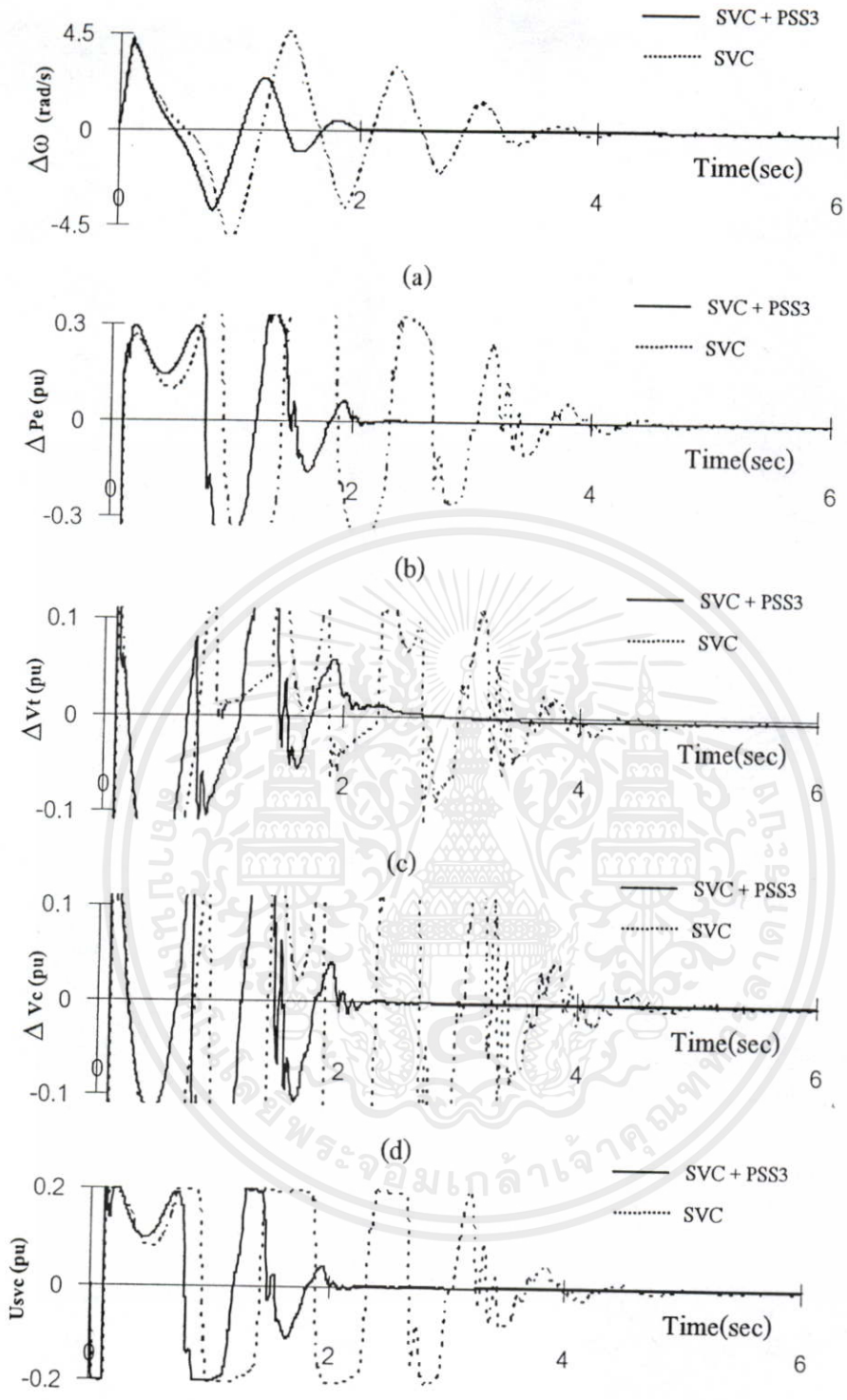
รูปที่ 8.49 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยโปรแกรม GAFCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



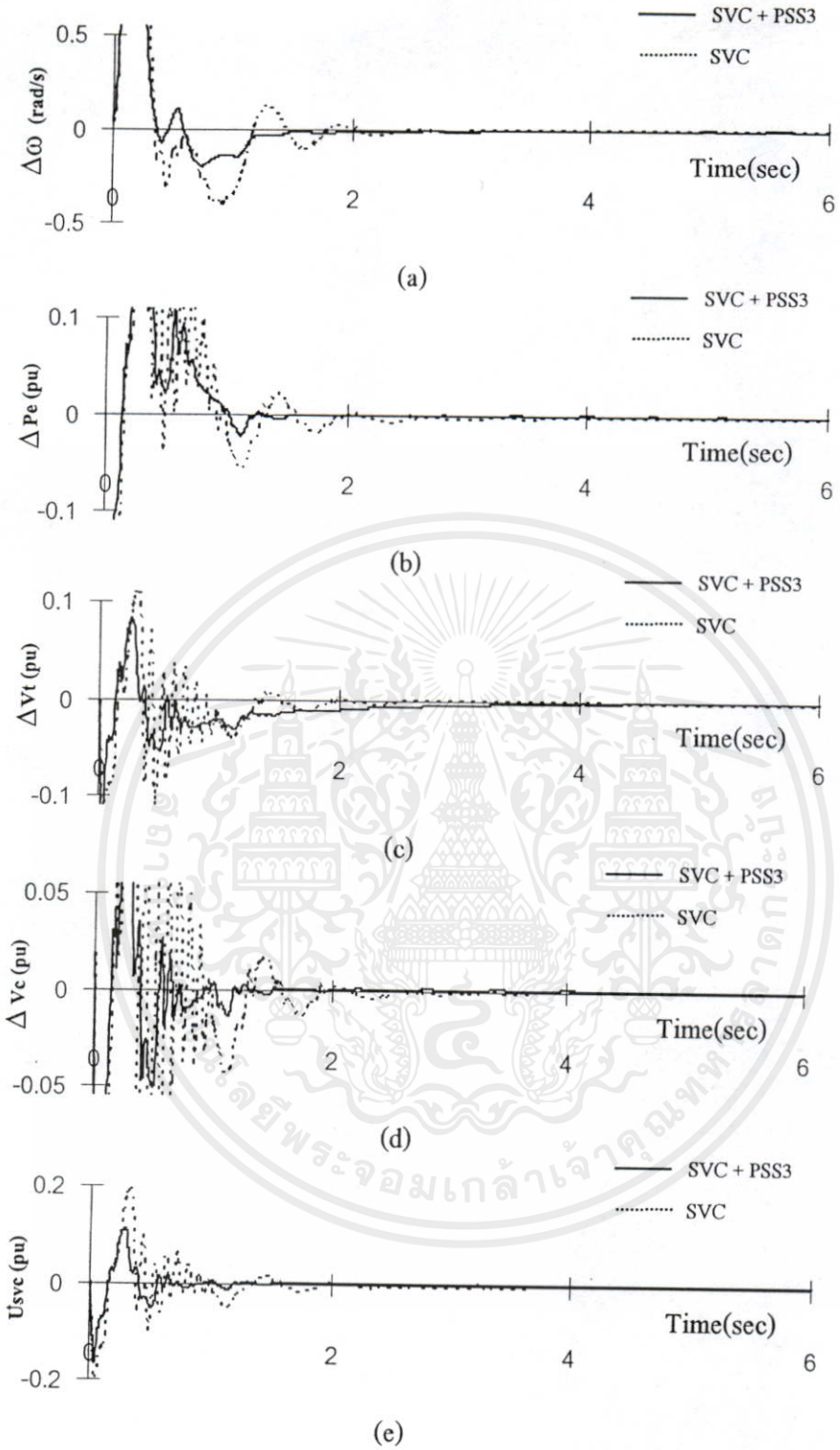
รูปที่ 8.50 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS2 ต่อการรบกวนแบบเล็กน้อย เมื่อ $X_e = 1.3$ เท่า ออกแบบด้วยโปรแกรม GAFCD

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8.51 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบรุนแรง เมื่อ $X_j = 1.3$ เท่า ออกแบบด้วยโปรแกรม GAFCD

ลิขสิทธิ์มหาวิทยาลัยเชียงใหม่
ว่ากรณีใดๆ ทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8.52 ผลตอบสนองกรณีควบคุมด้วย SVC+PSS3 ต่อการรบกวนแบบเล็กน้อย เมื่อ X_j = 1.3 เท่า ออกแบบด้วยโปรแกรม GAFCD

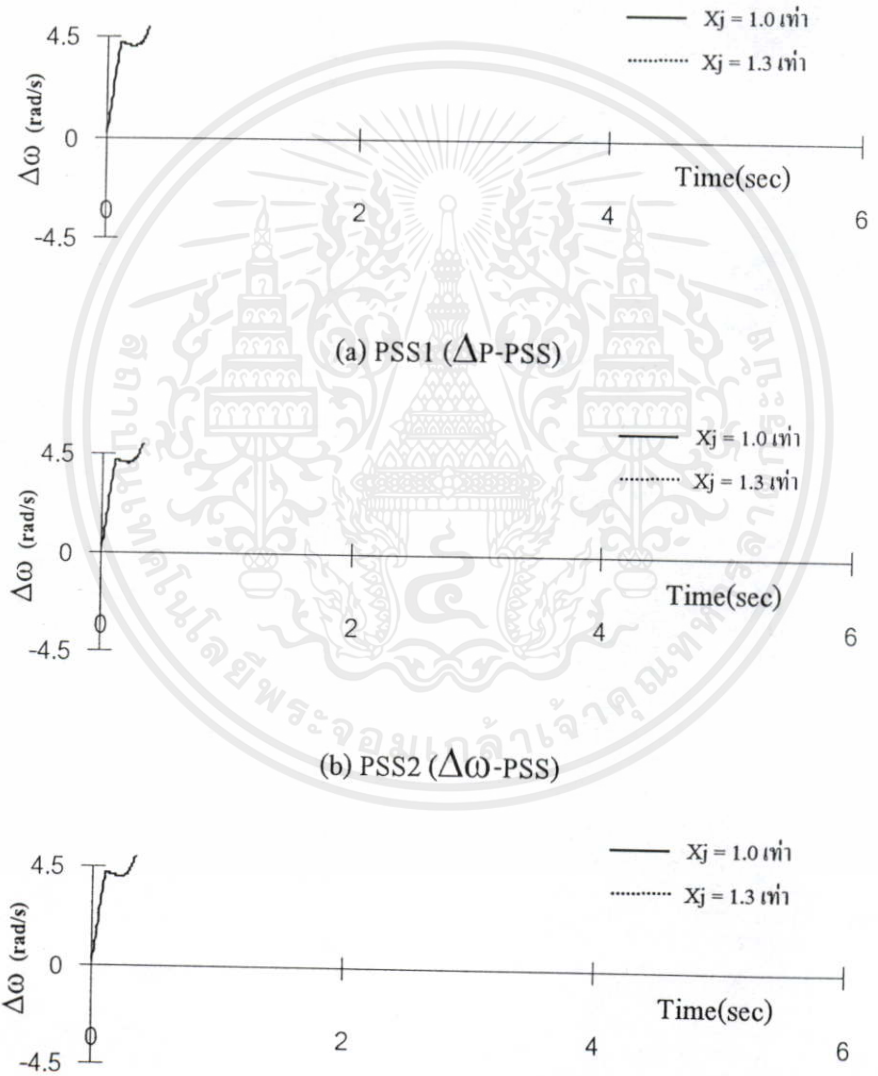
ลิขสิทธิ์มหาวิทยาลัยเชียงใหม่ ... ไปใช้ประโยชน์ด้านการ ... ว่ากรณีใดๆ ... ถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 9
บทสรุปและข้อเสนอแนะ

9.1 บทสรุป

ผลสรุปการทดสอบต่างๆในวิทยานิพนธ์ฉบับนี้ สามารถสรุปเป็นข้อๆได้ดังนี้

1. การควบคุมด้วย PSS แต่เพียงอย่างเดียว ไม่สามารถรักษาเสถียรภาพเมื่อเกิดการรบกวนแบบรุนแรงได้ ดังรูปที่ 9.1



(a) PSS1 (ΔP-PSS)

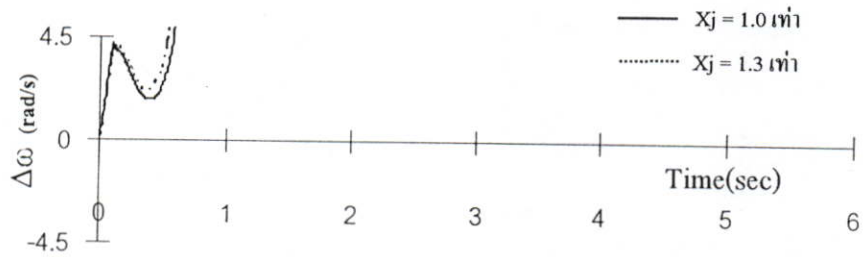
(b) PSS2 ($\Delta\omega$-PSS)

(c) PSS3 ($\Delta\omega$-PSS ชนิดฟัซซี่)

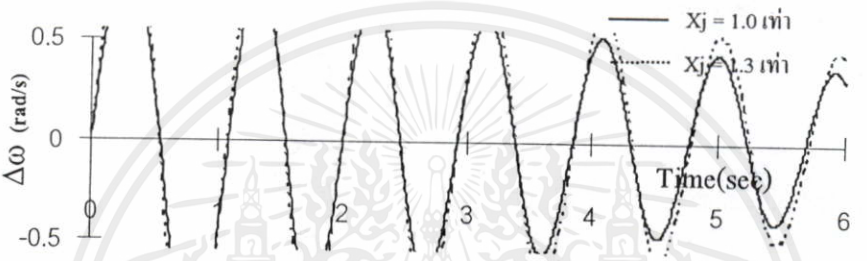
รูปที่ 9.1 เปรียบเทียบผลตอบสนองเมื่อควบคุมด้วย PSS แบบต่างๆภายใต้การรบกวนแบบรุนแรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การใช้ SVC โดยควบคุมจากสัญญาณแรงดันแต่เพียงอย่างเดียว ไม่สามารถสร้างแดมปิ้งในระบบได้มากพอ ดังรูปที่ 9.2 ดังนั้นจึงต้องมีการสร้างสัญญาณควบคุมเสริมเข้ามาควบคุมร่วม



(a) การรบกวนแบบรุนแรง

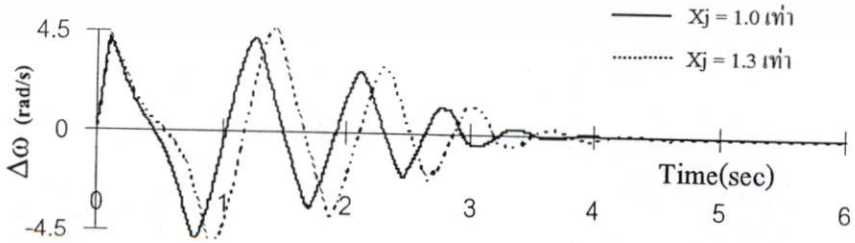


(b) การรบกวนแบบเล็กน้อย

รูปที่ 9.2 ผลตอบสนองเมื่อควบคุมด้วย SVC โดยไม่มีสัญญาณควบคุมเสริม

3. การควบคุมด้วยวิธีฟัซซี่ที่นำเสนอ มีความง่ายและรวดเร็วในการคำนวณสัญญาณควบคุม และสามารถประยุกต์ใช้กับ PSS และ SVC ได้

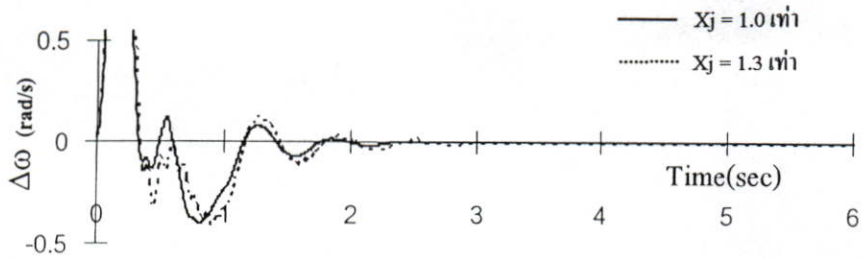
4. การควบคุม SVC ด้วยตัวควบคุมฟัซซี่ที่นำเสนอ ให้ผลตอบสนองที่ดีทั้งการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย และสามารถทำงานได้ในย่านทำงานที่กว้าง ดังแสดงในรูปที่ 9.3



(a) การรบกวนแบบรุนแรง

รูปที่ 9.3 ผลตอบสนองเมื่อควบคุมด้วย SVC ชนิดฟัซซี่

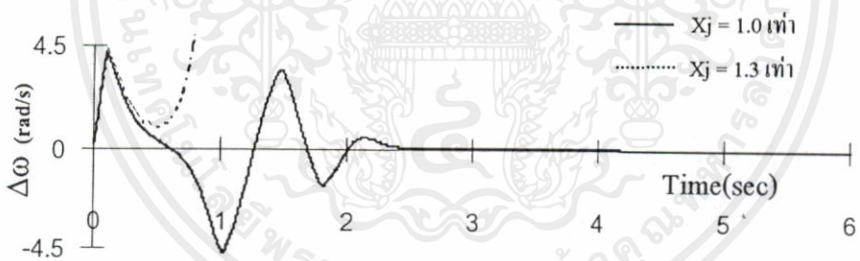
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



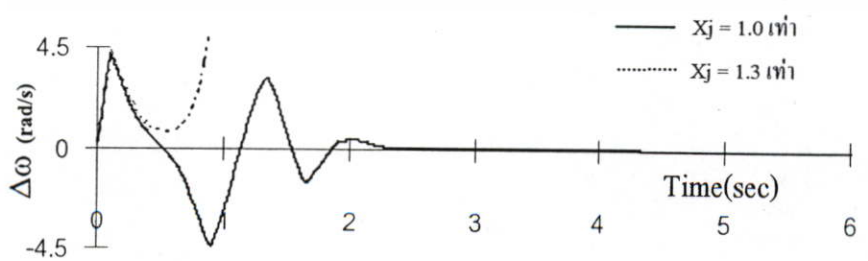
(b) การรบกวนแบบเล็กน้อย

รูปที่ 9.3 (ต่อ) ผลตอบสนองเมื่อควบคุมด้วย SVC ชนิดฟัซซี่

5. การนำ SVC ที่ควบคุมด้วยตัวควบคุมฟัซซี่ที่นำเสนอ เมื่อนำมาใช้ควบคุมร่วมกับ PSS จะต้องมีการพิจารณาอย่างรอบคอบ เพราะอาจทำให้ย่านทำงานที่กว้างลดลงได้ จากผลการทดสอบในบทที่ 8 พบว่า เมื่อใช้ SVC ควบคุมร่วมกับ PSS3 ซึ่งเป็นกรณีที่ให้ผลตอบสนองที่ดีที่สุด ผลตอบสนองที่ได้ยังคงดีเหมือนเดิมทั้งการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย และยังสามารถทำงานได้ในย่านกว้างเหมือนเดิม ผลตอบสนองเปรียบเทียบการควบคุมร่วมทั้ง 3 วิธีแสดงในรูปที่ 9.4



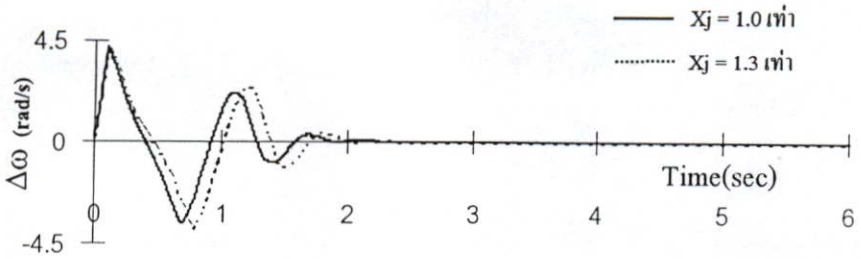
(a) SVC+PSS1



(b) SVC+PSS2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 9.4 ผลตอบสนองเปรียบเทียบการควบคุม SVC ร่วมกับ PSS แบบต่างๆ



(c) SVC+PSS3

รูปที่ 9.4 (ต่อ) ผลตอบสนองเปรียบเทียบการควบคุม SVC ร่วมกับ PSS แบบต่างๆ

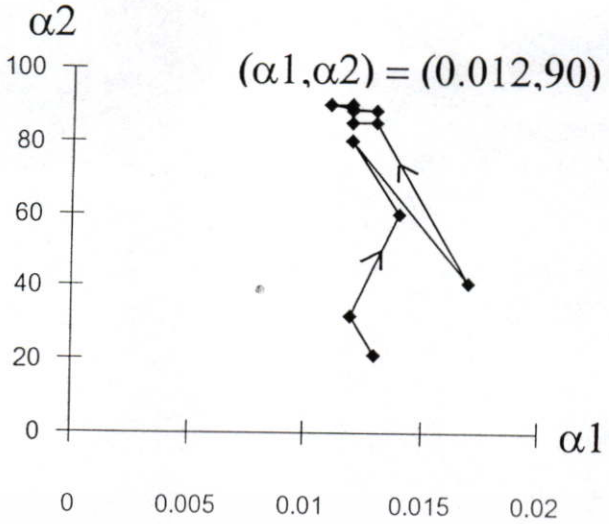
6. การเข้ารหัสโครโมโซมแบบเกรย์แทนไบนารี ช่วยเพิ่มความแม่นยำและความเร็วในการค้นหาคำตอบของ GA รวมทั้งยังแก้ปัญหาการลู่เข้าก่อนกำหนดของ GA ได้เป็นอย่างดี

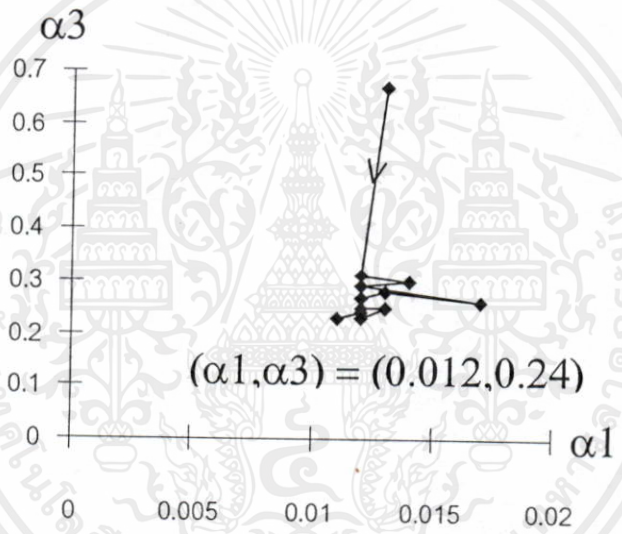
7. การออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีเจเนติกอัลกอริทึ่ม โดยใช้โปรแกรม GAFCD ให้คำตอบที่ดีกว่าและมีความเร็วมากกว่าการออกแบบโดยใช้วิธีไอเทอร์เรชั่น โดยการออกแบบด้วยโปรแกรม GAFCD ใช้เวลาประมาณ 11 นาที และการออกแบบด้วยวิธีไอเทอร์เรชั่นใช้เวลาประมาณ 15,472 นาที* ซึ่งประหยัดเวลาลงประมาณ 99.93 % ตัวอย่างเส้นทางเข้าหาคำตอบเมื่อเปรียบเทียบวิธีเจเนติกอัลกอริทึ่มกับวิธีไอเทอร์เรชั่นแสดงในรูปที่ 9.5 และรูปที่ 9.6 ซึ่งจะแสดงให้เห็นว่าเส้นทางเข้าหาคำตอบของวิธีเจเนติกอัลกอริทึ่มสั้นกว่าวิธีไอเทอร์เรชั่น ทำให้เวลาที่ใช้ของวิธีเจเนติกอัลกอริทึ่มน้อยกว่าวิธีไอเทอร์เรชั่น

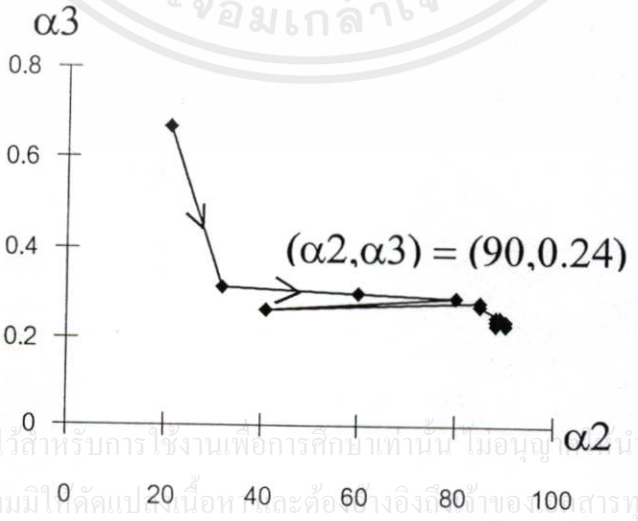
8. การออกแบบตัวควบคุมฟัซซี่ของ SVC ด้วยวิธีเจเนติกอัลกอริทึ่มโดยใช้โปรแกรม GAFCD มีความยืดหยุ่นในการนำไปใช้งานสูง ดังนั้นเมื่อมีการปรับปรุงหรือเปลี่ยนแปลงการควบคุมของ SVC การค้นหาคำตอบด้วยวิธีเจเนติกอัลกอริทึ่มยังคงใช้ได้เหมือนเดิม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

* $\alpha 1$ มีความละเอียด 101 ช่วง $\alpha 2$ เท่ากับ 91 ช่วง $\alpha 3$ เท่ากับ 101 ช่วง เวลาประมวลผลคำตอบ 1 ครั้งประมาณ 1 วินาที ดังนั้นเวลาที่ใช้ทั้งหมด คือ $101 \times 91 \times 101/60 = 15,472$ นาที

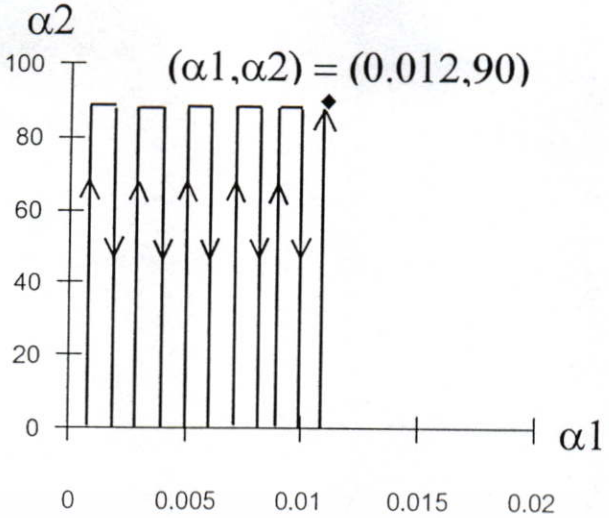


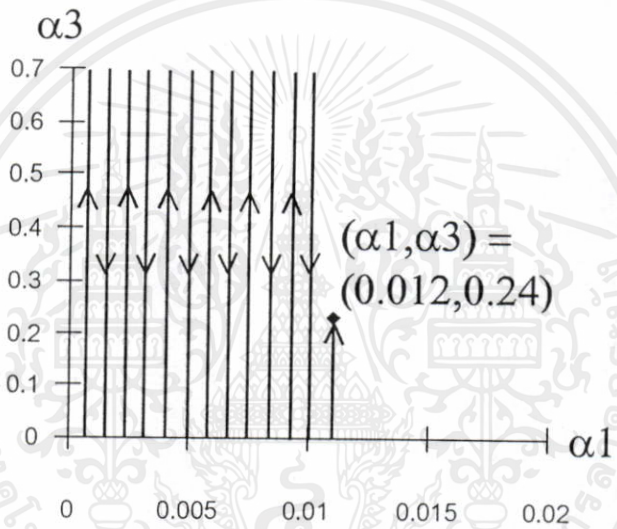


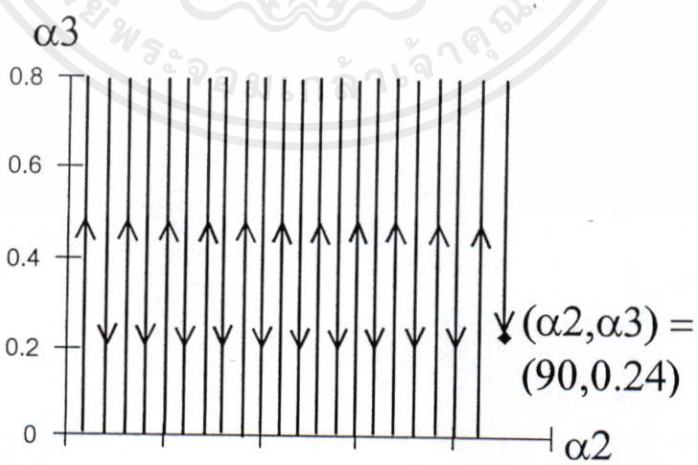


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 9.5 เส้นทางเข้าหาคำตอบของวิธีเจเนติกอัลกอริทึม







เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 9.6 เส้นทางเข้าหาคำตอบของวิธีไอเทอร์เรชั่น

9.2 ข้อเสนอแนะ

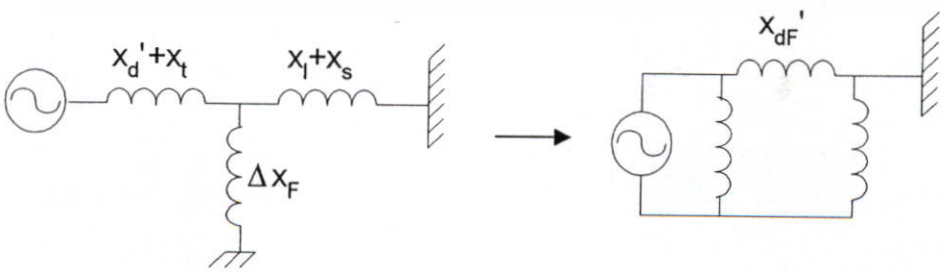
1. วิธีฟัซซี่แบบใช้เฟสเพลนที่นำเสนอนี้ ยังคงมีแกนของเฟสเพลนคงที่ การปรับให้แกนในแนวตั้งและแนวนอนของเฟสเพลนให้เคลื่อนที่ได้อย่างเหมาะสมจะช่วยเพิ่มพื้นที่ในการควบคุมแบบหน่วงให้มากขึ้นได้ ซึ่งจะทำให้ผลตอบสนองเข้าสู่จุดสมดุลได้เร็วขึ้นกว่าเดิม

2. สวิทชิ่งไลน์ SL ที่ใช้สำหรับวิธีฟัซซี่แบบใช้เฟสเพลนที่นำเสนอนี้ สมมุติให้คงที่ที่ 135° ในทิศทวนเข็มนาฬิกา (เป็นมุมที่ดีที่สุดสำหรับวิธีควบคุมในวิทยานิพนธ์นี้ ซึ่งทดสอบโดยให้ SL เป็นยีนส์ตัวหนึ่งในโครโมโซมของ GA) การหาความสัมพันธ์ที่ทำให้ SL มีการเปลี่ยนแปลงอย่างเหมาะสมต่อการรบกวน จะเป็นการเพิ่มพื้นที่การรับหรือจ่ายกำลังไฟฟ้ารีแอกทีฟให้มากขึ้น ซึ่งอาจทำให้ตัวควบคุมทำงานได้ในย่านกว้างกว่าเดิม

3. แบบจำลองที่ใช้ในวิทยานิพนธ์นี้ ประกอบด้วยเครื่องจักรกลไฟฟ้าซิงโครนัสเพียง 1 ตัว ในอนาคตจะต้องมีการพัฒนาให้แบบจำลองประกอบด้วยเครื่องจักกลไฟฟ้าซิงโครนัสหลายตัว แต่ทั้งนี้ต้องคำนึงถึงศักยภาพของเครื่องคอมพิวเตอร์ที่นำมาใช้ในการคำนวณด้วย ซึ่งในที่นี้แนะนำให้มีเครื่องจักรกลไฟฟ้าซิงโครนัส 2 หรือ 3 ตัวก่อน

4. เจเนติกอัลกอริทึ่มจะมีบทบาทในงานวิศวกรรมในอนาคตมากขึ้น การตัดสินใจว่าปัญหาใดจะใช้เจเนติกอัลกอริทึ่มแก้ปัญหา ต้องพิจารณาเงื่อนไขต่างๆประกอบด้วย เพราะปัญหาบางชนิดไม่เหมาะสมกับการใช้เจเนติกอัลกอริทึ่ม

5. ในงานวิจัยนี้ได้พิจารณาเฉพาะฟอล์ทชนิด 3 เฟสลงดิน ซึ่งเป็นการพิจารณเฉพาะโครงข่ายลำดับบวกเท่านั้น ถ้าหากเป็นฟอล์ทชนิดอื่น เช่น 2 เฟสลงดิน, ฟอล์ทระหว่าง 2 เฟส หรือ ฟอล์ท 1 เฟสลงดิน จะต้องพิจารณาค่ารีแอกแตนซ์โครงข่ายลำดับลบ และโครงข่ายลำดับศูนย์ด้วย ซึ่งจะทำให้แบบจำลองต้องมีการปรับเปลี่ยน โดยเพิ่มค่ารีแอกแตนซ์ขนานเข้าไป ณ ตำแหน่งที่เกิดฟอล์ท ดังเช่นรูปที่ 9.7 และค่ารีแอกแตนซ์นี้จะมีค่าเปลี่ยนแปลงไปตามชนิดของการเกิดฟอล์ทตาม ตารางที่ 9.1 [41]



รูปที่ 9.7 การแปลงเป็นวงจรสมมูลย์อย่างง่ายระหว่างเกิดฟอล์ทสำหรับฟอล์ทชนิดต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจัดทำขึ้นเพื่อแจกจ่าย หรือการดัดแปลงเนื้อหาและข้อความ ไม่ว่าส่วนหนึ่งหรือทั้งหมดของเอกสารก่อนการนำไปใช้

ตารางที่ 9.1 ค่ารีแอกแตนซ์ขนานตามชนิดของฟอล์ท

รีแอกแตนซ์ขนาน	ประเภทฟอล์ท			
	3 เฟสลงดิน	2 เฟสลงดิน	ระหว่าง 2 เฟส	1 เฟสลงดิน
ΔX_F	0	$\frac{X_2 X_0}{X_2 + X_0}$	X_2	$X_2 + X_0$

จากรูปที่ 9.7 ซึ่งเป็นการแปลงวงจรสมมูลย์จาก Y เป็น Δ ค่ารีแอกแตนซ์สุทธิเมื่อเกิดฟอล์ทในวงจร Δ สามารถคำนวณได้ตามสมการที่ (9.1)

$$X'_{dF} = X'_d + X_t + X_l + X_s + \frac{(X'_d + X_t)(X_l + X_s)}{\Delta X_F} \tag{9.1}$$

ซึ่งจะพบว่า ข้อมูลที่จะต้องพิจารณาเพิ่มขึ้นมา คือ ค่าของรีแอกแตนซ์ของโครงข่ายลำดับลบและโครงข่ายลำดับศูนย์ ดังนั้นหากมีข้อมูลเหล่านี้เพิ่มเติมการวิเคราะห์รูปแบบฟอล์ทอื่นๆก็สามารถทำได้ รายละเอียดเพิ่มเติมสำหรับเรื่องนี้สามารถดูได้จาก [41]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] A.J.Wood, B.F.Wollenberg. **Power Generation Operation and Control**. New York : John Wiley & Sons. 1996.

[2] A.D.Gerhart, T.Hillesland, J.F.Luini and M.L.Rockfield , Jr. "Power System Stabilizer : Field Testing and Digital Simulation." IEEE Trans. on PAS, Sep/Oct 1971, pp. 343-348.

[3] P.W.Sauer, M.A. Pai. **Power System Dynamics and Stability.** New Jersey : PrenticeHall. 1998.

[4] Marzan A. Iskandar, Masashi Satoh, Yasuyuki Ohmorl, Seiichi Matoba, Takatsugu Okabe and Yoshibumi Mizutani. "On Fuzzy Control Based Static Var Compensator for Power System Stability Control." ANNPS'93, Proc. of the 2nd International Forum on Applications of Neural Networks to Power System, April 1993, pp. 201-205.

[5] IEEE Tutorial Course. **Application of Static Var Systems for System Dynamics Performance.** IEEE 87 THO 187-5-PWR, 1987.

[6] J.A.Erinmez, D.J.Coates, H.Lu. "The Application of Static Var Compensators on the UK Tranmission Network." Proc.IEE conference on DC/AC system, London, 1992.

[7] S.S.Vadhera. **Power System Analysis & Stability.** Delhi : Khanna publishers. 1994.

[8] H.F.Wang, F.J. Swift. "Capability of The Static Var Compensator in Damping Power System Oscillations." IEE Proc.-Gener. Transm.Disturb., vol. 143, No. 4, July 1996. pp. 353-358.

[9] E.Z.Zhou. "Application of Static Var Compensators to Increase Power System Damping." IEE Trans. on Power Systems, vol. 8, No. 2, May 1992. pp. 655-661.

[10] Yuan-Yih Hsu, Chi-Jui WU. "Design of PID Static Var Controllers for the Damping of Subsybchronous Oscillations." IEEE Trans. on Energy Conversion, vol. 3, No. 2, June 1988. pp. 210-216.

[11] Chi-Jui WU, Yuan-Yih Hsu. "Design of Self-Tuning PID Power System Stabilizer for Multimachine Power Systems." IEEE Trans. on Power Systems, vol. 3,No. 3, Aug 1988. pp. 1059-1064.

[12] Chin-Hsing Cheng, Yuan-Yih Hsu. "Damping of Generator Oscillations Using an Adaptive Static Var Compensator." IEEE Trans. on Power Systems, vol. 7,No. 2, May 1992. pp.718-725.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[13] Lance Chambers. **Practical Handbook of Genetic Algorithms Application Volume 1.** New York : CRC Press. 1992.

[14] Scott Robert Ladd. **Genetic Algorithm in C++.** New York : M&T Books. 1996.

[15] K.F.Man, K.S.Tang, S.Kwong and W.A.Halang. **Genetic Algorithms for Control and Signal Processing.** New York : Springer-Verlag. 1997.

[16] J.H.Holland, **Adaption in Natural and Artificial Systems.** New York : MIT Press.1975.

[17] L.Davis, **Handbook of Genetic Algorithms.** Van Nostrand Reinhold. 1991.

[18] J.J.Grefenstette, "Optimization of Control Parameters for Genetic Algorithms." IEEE Trans. on System, Man, and Cybernetics, SMC-16(1),1986. pp. 122-128.

[19] Z.Michalewicz, **Genetic Algorithms + Data Structures = Evolution Program.** 2 nd Ed.,New York : Springer-Verlag. 1994.

[20] P.Kundur . **Power System Stability and Control .** New York : McGraw-Hill. 1994.

[21] O.Berge, K.O.Petterson and S.Sorzdal. "Active cancellation of Transformer Noise : Field Measurments" Applied Acoustics, 1988. pp. 309-320.

[22] D.E.Goldberg, **Genetic Algorithms in Search, Optimization and Machine Learning.** New York : Addison-Wesley. 1989.

[23] J.E.Baker, "Reducing Bias and Inefficiency in the Selection Algorithms." Proc. 2nd Int. Conf. Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987. pp.14-21.

[24] B.L.Miller, D.E.Goldberg. "Genetic Algorithms, Tournament Selection, and the Effects of Noise." IlliGAL Report No. 95006, Department of Computer Science, University of Illinois at Urbana-Champaign, July 1995.

[25] C.K.Oei, D.E.Goldberg and Shan-Jin Chang. "Tournament Selection and the Preservation of Diversity." IlliGAL Report No. 91011, Department of Computer Science, University of Illinois at Urbana-Champaign, Dec 1991.

[26] Darrell Whitley. "A Genetic Algorithm Tutorial." Technical Report CS-93-103 , Department of Computer Science, Colorado State University, Nov 1993.

[27] D.E.Goldberg, K.Deb. "A Comparative Analysis of Selection Schemes used in Genetic Algorithms." Foundations of Genetic Algorithm, Morgan Kaufmann, 1991. pp. 69-93.

[28] G.Syswerda. "Uniform Crossover in Genetic Algorithms." Proc. of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, 1989. pp. 2-9.

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย ... ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

[29] IEEE Special Stability Controls Working Group. "Static Var Compensator Models for Power Flow and Dynamic Performance Simulation", IEEE Trans. on Power Systems, Vol. 9, No. 1, Feb 1994, pp.229-240.

[30] M.Noroozian. "Application of Shunt and Series Compensators for Damping of Electromechaical Oscillations", IEE, 1998, pp.4/1-4/3.

[31] S.E.M.de Oliveira. "Synchronous and Damping Torque Coefficients and Power System Steady-State Stability as Affected by Static Var Compensators", IEEE Trans. on Power Systems, Vol. 9, No. 1, Feb 1994, pp. 109-119.

[32] Y.H.Song, A.T.Johns and R.K.Aggarwal. "Nonlinear Thyristor-Controlled Static Var Compensation", The European Power Electronics Association, 1993, pp. 56-60.

[33] Youüe Ma, Shousun Chen and Baolin Zhang. "A Study on Nonlinear SVC Control for Improving Power System Stability", IEEE TENCON'93, 1993, pp. 166-169.

[34] S.E.M.de Oliveira. "Effect of Excitation Systems and of Power System Stabilisers on Syschronous Generator Damping and Sychronising Torques", IEE Proc., Vol. 136, No. 5, Sep 1989, pp.264-270.

[35] C.S.Indulkar and Baldev Raj. "Application of Fuzzy Controller to Automatic Generation Control", Electric Machines and Power Systems, 1995, pp.209-220.

[36] C.S.Chang, Weihul Fu and Fushuan Wen. "Load Frequency Control Using Genetic Algorithm based Fuzzy Gain Scheduling of PI Controllers", Electric Machines and Power Systems, 1998, pp. 39-52.

[37] Fushuan Wen, C.S.Chang and Weihui Fu. "New Approach to Alarm Processing in Power Systems based on the Set Covering Theory and a Refined Genetic Algorithm", Electric Machines and Power Systems, 1998, pp.53-67.

[38] M.A.Abido and Y.L.Abdel-Magid. "A Genetic-Based Power Systems Stabilizer", Electric Machines and Power Systems, 1998, pp.559-571.

[39] G.N.Taranto and D.M.Falcäo. "A Genetic-Based Control Design for Damping Power System Inter-Area Oscillations", IEEE Proc. of the 35th Conference on Decision and Control, Dec 1996, pp. 4389-4394.

[40] Juan Shi, L.H.Herron and A.Kalam. "A Fuzzy Logic Controller Applied to Power System Stabilizer for a Synchronous Machine Power System", IEEE TENCON'92 , Nov 1992, pp.346-350.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

[41] J.Machowski, J.W.Bialek and J.R.Bumby. **Power System Dynamics and Stability.** Chichester : John Wiley & Sons. 1997.

[42] T.Hiyama. "Integrated Fuzzy Logic Stabilizing Controller for Power Systems", IEEE Proc. of the 35th Conference on Decision and Control, Dec 1996, pp. 2185-2190.

[43] T.Hiyama, M.Kugimiya and H.Satoh. "Advanced PID Type Fuzzy Logic Power System Stabilizer", IEEE Trans. On Energy Conversion, vol.9, No.3, Sep 1994, pp. 514-520.

[44] T.Hiyama, M.Miyazaki and H.Satoh. "A Fuzzy Logic Excitation for Stability Enhancement of Power Systems with Multi-mode Oscillations", IEEE Trans. On Energy Conversion, vol.11, No.2, Sep 1996, pp. 449-454.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

อกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก.

1. สมการที่ใช้คำนวณระบบจำลอง

GEN

$$\frac{d\delta}{dt} = \Delta\omega \tag{1}$$

$$\frac{d\omega}{dt} = \frac{1}{M}(\Delta P_i - D\Delta\omega - \Delta P_e) \tag{2}$$

$$\frac{d\phi f}{dt} = V_f - \frac{X_d \phi f}{X_d' T_{d0}'} - \frac{(x_d - x_d')V_0 \cos\delta}{X_d'} \tag{3}$$

AVR

$$\frac{dV_f}{dt} = \frac{K_A}{T_A}(-\Delta V_t + R_f) - \frac{V_f - V_{f0}}{T_A} \tag{4}$$

$$\frac{dR_f}{dt} = \frac{K_f}{T_f}\frac{dV_f}{dt} - \frac{R_f}{T_f} \tag{5}$$

GOV

$$\frac{dg}{dt} = \frac{1}{T_g}(-\Delta\omega - \sigma g) \tag{6}$$

$$\frac{dP_i}{dt} = \frac{1}{T_t}(g - \Delta P_i) \tag{7}$$

SVC

$$\frac{dV_{cm}}{dt} = \frac{V_c - V_{cm}}{T_v} \tag{8}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\frac{dP_m}{dt} = \frac{1}{T_m}(\Delta P_c - P_m) \tag{9}$$

$$\frac{dB_s}{dt} = \frac{K_s}{T_s}(V_{cm} - B_s) \tag{10}$$

สมการที่ (1)-(10) เมื่อทำให้เป็นเชิงเส้นจะได้

$$\frac{d\Delta\delta}{dt} = \Delta\omega \tag{11}$$

$$\frac{d\Delta\omega}{dt} = \frac{1}{M}(\Delta P_i - D\Delta\omega - \Delta P_e) \tag{12}$$

$$\frac{d\Delta\phi f}{dt} = \Delta V_f - \frac{X_d}{X_d' T_{d0}'}\Delta\phi f - \frac{(x_d - x_d')V_0 \sin\delta}{X_d'}\Delta\delta \tag{13}$$

$$\frac{d\Delta V_f}{dt} = \frac{K_A}{T_A}(-\Delta V_t + \Delta R_f) - \frac{1}{T_A}\Delta V_f \tag{14}$$

$$\frac{d\Delta R_f}{dt} = \frac{K_f}{T_f}\frac{d\Delta V_f}{dt} - \frac{\Delta R_f}{T_f} \tag{15}$$

$$\frac{d\Delta g}{dt} = \frac{1}{T_g}(-\Delta\omega - \sigma\Delta g) \tag{16}$$

$$\frac{d\Delta P_i}{dt} = \frac{1}{T_t}(\Delta g - \Delta P_i) \tag{17}$$

$$\frac{d\Delta V_{cm}}{dt} = \frac{\Delta V_c - \Delta V_{cm}}{T_v} \tag{18}$$

$$\frac{d\Delta P_m}{dt} = \frac{1}{T_m}(\Delta P_c - \Delta P_m) \tag{19}$$

$$\frac{d\Delta B_s}{dt} = \frac{K_s}{T_s}(\Delta V_{cm} - \Delta B_s) \tag{20}$$

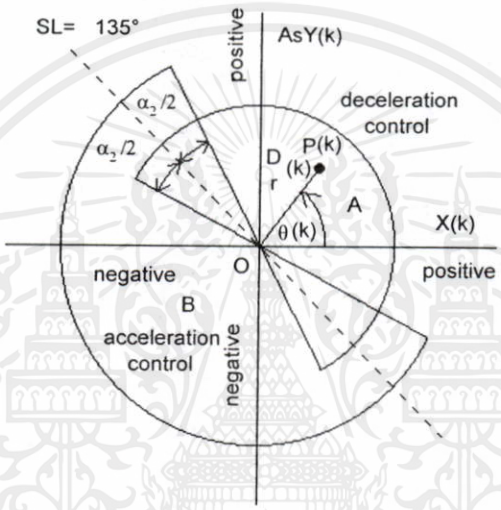
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข.

1. ฟัซซี่ชนิดเฟสเพลน

วิธีฟัซซี่ชนิดเฟสเพลนที่นำมาประยุกต์ใช้กับ SVC ในวิทยานิพนธ์ฉบับนี้ เป็นวิธีที่ดัดแปลงมาจากฟัซซี่ชนิดเฟสเพลนซึ่งใช้กับ PSS ใน [42-44] โดยมีวิธีการดังต่อไปนี้

1. จุด P(k) ใดๆบนเฟสเพลน กำหนดโดยคู่อันดับ [X(k),AsY(k)] ดังรูปที่ 1 โดย As คือ สเกลลิ่งแฟกเตอร์



รูปที่ 1 เฟสเพลน

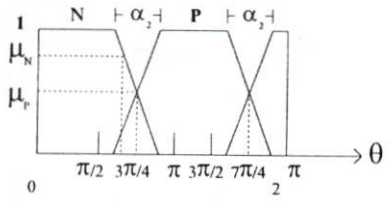
2. จุด P(k) บนเฟสเพลนถูกแปลงความหมายเป็น ขนาด D(k) และมุม θ(k)

$$D(k) = \sqrt{X(k)^2 + (AsY(k))^2} \quad (1)$$

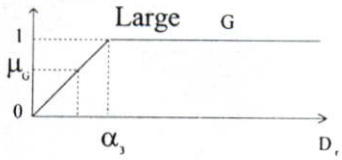
$$\theta\ (k) = \tan^{-1}(AsY(k) / X(k)) \quad (2)$$

3. พื้นที่ส่วน A และ B บนเฟสเพลนในรูปที่ 1 ถูกกำหนดโดยฟังก์ชั่นการเป็นสมาชิกของ θ(k) โดย N (θ(k)) สำหรับการจ่ายกำลังไฟฟ้ารีแอกทีฟ และ P(θ(k)) สำหรับการรับกำลังไฟฟ้ารีแอกทีฟ ฟังก์ชั่นการเป็นสมาชิกของ θ(k) แสดงในรูปที่ 2(a) และฟังก์ชั่นการเป็นสมาชิกของ D(k) แสดงในรูปที่ 2(b)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(a)



(b)

รูปที่ 2 (a) ฟังก์ชั่นการเป็นสมาชิกของθ(k) และ (b) ฟังก์ชั่นการเป็นสมาชิกของ D(k)

4. เขียนกฎการควบคุมของฟัซซี่ ได้ดังต่อไปนี้

กฎข้อที่ 1 : IF (θ(k) คือ N) THEN U_{SVC} (k) = U_C (k)

กฎข้อที่ 2 : IF (θ(k) คือ P) THEN U_{SVC}(k) = U_L(k)

กฎข้อที่ 3 : IF (D(k) คือ LARGE) THEN U_C(k) = U_{max}

โดยที่ U_{SVC}(k) คือ สัญญาณควบคุม SVC

U_C(k) คือ สัญญาณควบคุมเมื่อปรับ SVCให้จ่ายกำลังไฟฟ้ารีแอกทีฟ

U_L(k) คือ สัญญาณควบคุมเมื่อปรับ SVCให้รับกำลังไฟฟ้ารีแอกทีฟ

U_{max} คือ ขนาดสัญญาณควบคุมสูงสุด

5. ใช้การเฉลี่ยน้ำหนัก หาค่าสัญญาณควบคุม U_{SVC} ดังสมการที่ (3)

$$U_{SVC}(k) = \frac{\mu_N U_C(k) + \mu_P U_L(k)}{\mu_N + \mu_P} \tag{3}$$

โดยที่ μ_N คือ ระดับความเป็นสมาชิกของ N

μ_P คือ ระดับความเป็นสมาชิกของ P

แทนค่า U_C=-U_L ในสมการที่ (3) จะได้

$$U_{SVC}(k) = \left(\frac{\mu_N - \mu_P}{\mu_N + \mu_P}\right) U_C(k) \tag{4}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. จากกฎควบคุมฟัซซี่ข้อ 3 จะได้

$$U_C(k) = \mu_G U_{\max} \tag{5}$$

แทนค่า U_C จากสมการที่ (5) ลงในสมการที่ (4) จะได้

$$U_{SVC}(k) = \left(\frac{\mu_N - \mu_P}{\mu_N + \mu_P}\right)\mu_G U_{\max} \tag{6}$$

เนื่องจาก $\mu_N + \mu_P = 1$ ดังนั้นสมการที่ (6) สามารถเขียนใหม่ได้เป็น

$$U_{SVC}(k) = (2\mu_N - 1)\mu_G U_{\max} \tag{7}$$

2. ตัวอย่างการคำนวณฟัซซี่ชนิดเฟสเพลน

1. สมมุติให้สัญญาณอินพุทของตัวควบคุมฟัซซี่ $E_p(k)$ มีค่าเท่ากับ 0.2 pu , $E_p(k-1)$ = 0.1 pu, $\alpha_1 = 0.012$, $\alpha_2 = 90^\circ$, $\alpha_3 = 0.24$, T = 0.01 s และ U_{max} = 0.2 pu
2. คำนวณค่า $\dot{E}_p(k)$ โดย

$$\dot{E}_p(k) = (E_p(k) - E_p(k-1))/T = (0.2\text{-}0.1)/0.01 = 10 \text{ pu/s}$$

3. จากรูปที่ 1 X(k) = $E_p(k)$, As = α_1 , Y(k) = $E_p(k-1)$ จุดP(k) คือ คู่อันดับ (X(k),AsY(k)) = ($E_p(k)$,$\alpha_1 \dot{E}_p(k)$) ดังนั้น

$$D(k) = \sqrt{(0.2)^2 + ((0.012)(10))^2} = 0.233 \text{ pu}$$
$$\theta(k) = \tan^{-1}((0.012)(10)/0.2) = 30.963^\circ$$

4. นำ $D(k)$ และ $\theta(k)$ ไปหาค่าความเป็นสมาชิกจากฟังก์ชั่นสมาชิกในรูปที่ 2(a) และรูปที่ 2(b) จะได้ $\mu_N(k) = 1.0$ และ $\mu_G(k) = 0.233/0.24 = 0.971$
5. คำนวณหา $U_{svc}(k)$ จากสมการที่ (7) จะได้

$$U_{svc}(k) = ((2)(1.0)\text{-}1)(0.971)(0.2) = 0.1942 \text{ pu}$$

เอกสารนี้ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค.

1. โปรแกรม pss1.c

```
/* pss1.c : ΔP-PSS */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\gtool.h" /* graphic utility */
#include"c:\define.h" /* #define */
#define Umpss   0.1  /* max. Upss */
#define Kpss    1     /* gain */
#define T1      0.04  /* T1 time constant */
#define T2      0.05  /* T2 time constant */
static double vd0 , vq0 , id0 , iq0 , del0 , del1 , Q0 ,Q0l ,
              Q0t , fif0 , vf0 , vc0 , v0 , Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /*1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
static double xd = 1.620, /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance  */
              xq = 1.600,  /* q-axis reactance */
              Td0d =  5.600, /* d-axis open circuit time
                                constant */
              M = 0.0191 , /*Inertia constant */
              D = 0.00531 ; /* Damping coeffient */
static double dy1[N],  /* Real power Pe  */
              dy2[N],  /* Terminal voltage Vt */
              dy3[N],  /* SVC  Vc */
              ti,      /* simulation time */
              Ups,Upss, /* Supplementary control signal
*/
              P1, out ;
static double Pd;  /* small disturbance */
static double ZZ;  /* used in AVR limit */
static int  DD1,DD2,DXD1,DXD2; /* GOV limit */
static int  AVRL, AVRL1;  /* AVR limit */
static int  PSS,PSS1; /* PSS limit */
static int  kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double  x[],double  t,int n,double
                dy2[N],double dy3[N],int flag1);
void sys1(double  x[],double  f[],double  y[],double
                dy2[N],double dy3[N],int flag1);
unsigned _stack = 50000;
FILE *fpw;

void   main()
{
 static double  GD;
 static double  x1[NA], y1[4], f[NA];
 static double  dt1[2],dt2[2],dt3[2],dt4[2];
 static int  flag,fri;
 int  stop;
 float  Time;
 do{
 fpw=fopen("Pss1.dat","w"); /* open output file */
 /* Initial value of variables */
 Upss=0;Pd=0.0;ZZ=0.0;
 AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
 XD2=0.0;PSS=PSS1 = 0.0;
 vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;Q0=
 0.0;Q0l=0.0;Q0t=0.0;
 fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
 xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
 for (kpt = 0; kpt < N; kpt++)
 { dy1[kpt] = 0.0;
   dy2[kpt] = 0.0;
   dy3[kpt] = 0.0;
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆที่ไม่ได้รับอนุญาตทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;  /* d-axis reactance */
xdd = 0.250; /* d-axis transient reactance */
xq = 1.600; /*  q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;  /* Inertia constant */
D = 0.00531; /* Damping coeficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
 }
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0; fri=0; kpt=0.0; ti=0.0;
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" <===== ΔP-PSS Control ======>\n\n\n");
printf("              * wait a minute *                \n");
select = 1; delay(500); clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" <===  Simulation Power System  ===>\n");
printf("  Please select Disturbance.\n");
printf("  large:[0],small:[1] --->");
scanf("     %d",&CH);
printf("\n");
if(CH==0)
{ printf("  Select trip time after 3-phase fault.\n");
  printf("              [ 0 - 0.1s ] --->");
  scanf("   %f",&Time);
  (int)KH=Time/0.005;
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
/* draw graph axis */
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0),
          "0.30");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0),
          "0.00");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0),
          "-0.30");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0),
          "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการดัดแปลงทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    w_axiss(0.0, 0.5, N * T, -0.5);
    w_line(0.0, 0, 0);

    g_view(0, 0, 639, 399, -1, 1);
    g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
    g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
    g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
    g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

    g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
    w_view(40.0, 135.0, 440.0, 235.0);
    w_axiss(0.0, 0.1, N * T, -0.1);
    w_line(0.0, 0, 0);

    g_view(0, 0, 639, 399, -1, 1);
    g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
    g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
    g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
    g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0),
          "-0.1");

    g_view(40.0, 5.0, 440.0,365.0,-1,-1);
    w_view(40.0, 265.0, 440.0, 365.0);
    w_axiss(0.0, 0.1, N * T, -0.1);
    w_line(0.0, 0, 0);

    g_view(0, 0, 639, 399, -1, 1);
    g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
    g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
    g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0),
          "ΔVt");
    g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0),
          "-0.1");
}
initial_system();
if(CH==1) Pd = 0.0;  /* small disturbance */
else  Pd = 0.0;
    x_stand(x1); /* initial condition */
    for (kpt = 0;kpt <=N-1;kpt++)
    {  ti = (double)kpt * T;  /*current time */
if(CH==0)   /* large disturbance */
 if(kpt==0) flag = 1;   /* 3-phase fault */
  if(CH==0)
   if(kpt==0+KH)  /* protection trip time */
   { flag = 0;
     xl1 = 2.0*xl1;
   }
  if(CH==1)     /* small disturbance */
   if(kpt>=0)
    if (kpt < 100)
     Pd = -0.1/100*kpt;
          if(CH==1)
           if(kpt>=100)
            if (kpt < 200)
             Pd = -0.2 + 0.1 /100*kpt;
          if(CH==1)
           if(kpt>=200)
            Pd = 0.0;
  AVRL = AVRL1; DD1  = DXD1; DD2  = DXD2; PSS  =
  PSS1;
  runge4(x1, T, NA,dy2,dy3,flag);
  /* A V R Limit */
  if(x1[3] > Efmax - vf0)
  { AVRL1 = 1;
   ZZ = Efmax - vf0;
  }
  else
  if(x1[3] < Efmin - vf0)
  {  AVRL1 = -1;
   ZZ = Efmin - vf0;
  }
  else
  {  AVRL1 = 0;
   ZZ = x1[3];
  }
  Ups = Kpss*out;
  /* PSS Limit */
  if(Ups>Umpss) Ups = Umpss;
  if(Ups<-Umpss) Ups = -Umpss;
  if(Ups>Umpss)
  {  PSS1 = 1;
    Upss = Umpss;
  }
  else
  if (Ups<-Umpss)
  {  PSS1 = -1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจัดทำซ้ำ ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
   Upss = - Umpss;
 }
 else
 { PSS1 = 0.0;
   Upss = Ups;
 }
 /* G O V Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1)  DXD1=1;
 else
 if(GD < -0.1)  DXD1=-1;
 else
 DXD1=0;
 if(x1[5] > (Pmax-(P0-Pd)))  DXD2=1;
 else
 if(x1[5] < -(P0-Pd))  DXD2=-1;
 else
 DXD2=0;
 sys1(x1, f, y1,dy2,dy3,flag);
 d_out(y1);
 /* Plot  Graph */
 dt1[1] = x1[1];
 dt2[1] = dy1[kpt+1];
 dt3[1] = dy2[kpt+1];
 if(kpt==0) fri =0;
 else
 fri=1;
 if(CH==0)
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,4.5,N*T,-4.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri,1);
   fprintf(fpw,"%f %f %f %f
         %f\n",ti,dt1[1],dt2[1],dt3[1],Upss);
   g_view(80.0,140.0, 480.0, 240.0,1,1);
   w_view(80.0,140.0, 480.0, 240.0);
   w_axis(0.0,0.3,N*T,-0.3);
   w_attr(3, 0xffff);
   w_line2((ti), dt2[1], fri, 2);
   g_view(80.0, 270.0, 480.0, 370.0,1,1);
   w_view(80.0, 270.0, 480.0, 370.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt3[1] , fri, 3);
 }
 else
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,0.5,N*T,-0.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri, 1);

 fprintf(fpw,"%f %f %f %f %f
         \n",ti,dt1[1],dt2[1],dt3[1],Upss);

   g_view(80.0, 140.0, 480.0, 240.0,1,1);
   w_view(80.0, 140.0, 480.0, 240.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt2[1], fri, 2);

   g_view(80.0, 270.0, 480.0, 370.0,1,1);
   w_view(80.0, 270.0, 480.0, 370.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt3[1] , fri, 3);
   }
   dt1[0] = dt1[1];
   dt2[0] = dt2[1];
   dt3[0] = dt3[1];
 }
 getch(); closegraph(); clrscr();
 printf("\n\n\n\n\n\n\n\n\n\n");
 printf("      Continous ? (Quit press ESC)");
 stop=getch(); fclose(fpw);
 }
 while(stop!=0x1b);  /* ESC to exit */
 }
 void x_stand(double x[])
 /* initial vaule of variables */
 {
 x[0] = del0;
 x[1] = 0.0;
 x[2] = fif0;
```

เอกสารนี้ใช้ประกอบการเรียนการสอนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double  A, C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = xt+xl1+xl2+xee;
xa = 0;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/*  9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);

/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{
dy1[kpt+1] = y1[0] - P0;
dy2[kpt+1] = y1[1] - Vt0;
dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double f[],double y[],double
        dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double  A1,A2,B1,B2,B3,C1,C2;
double vd,vq, id,iq;
if(flag1==1)
{
A1 = xl1;
A2 = xl2+xee;
B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
B2 = xl1*A1/(3.0*xl1+A1);
B3 = 2.0*xl1*A1/(3.0*xl1+A1);
C1 = xt+B1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
C2 = A2+B3;

xe = (C1*C2+B2*(C1+C2))/B2;
xa = (C1*C2+B2*(C1+C2))/C2;
xa = -1.0/xa;
xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{
xe = xt+xl1+xl2+xee;
xa = 0;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f */
switch(AVRL)
{
case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
break;
case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
break;
case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
break;
}
/* 3: vf  */
switch(PSS)
{
case 1 :
f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x
[4])/Tf+Umpss))/Te;
break;
case -1:
f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
      Umpss))/Te;
break;
case 0 :
f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
      x[4])/Tf+Upss))/Te;
break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{
case 1 :f[5] =  0.1;
break;
case -1:f[5] = -0.1;
break;
case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] / W0;
break;
}
/* 6: h */
switch(DD2)
{
case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
break;
case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
break;
case  0 :f[6] = (-x[6]+x[5])/Tw;
break;
}
/* 7: Vcm */
f[7] = (dy3[kpt] - x[7]) /Tv;
/* 8: Pin */
f[8] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1;
/* 9: PSS */
f[9] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการณ์ใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
f[10] =  (x[9]-x[10])/T2;
f[11] = (T1*((-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1)-x
[11])/T2;
x[12] =x[10]+x[11];
out = x[12];
}
void runge4( double x[],double t,int n,
            double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differential equations */
{
double  k1[NA],  k2[NA],  k3[NA],  k4[NA],  y[3],
        z[NA];
int    i;
sys1(x, k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k1[i];
 sys1(z, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k2[i];
 sys1(z, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t * k3[i];
 sys1(z, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
```

---

2. โปรแกรม pss2.c

```
/* pss2.c : Δω-PSS */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\gtool.h" /* graphic utility */
#include"c:\define.h" /* #define */
#define  Umpss   0.1  /* max. Upss */
#define  Kwpss   13.0  /* gain */
#define  Tww     0.056  /* measuring time constant */
#define  Trr     4.0  /* reset circuit time constant */
#define  T1      0.3  /* T1 time constant */
#define  T2      0.02  /* T2 time constant */
#define  T3      0.2  /* T3 time constant */
#define  T4      0.02  /* T4 time constant */
static double  vd0 , vq0 , id0 , iq0 , del0 , del1 , Q0 ,Q0l ,
            Q0t , fif0 , vf0 , vc0 , v0 , Bs0;
static double  xc,xe,xa,xb,xl,
            xt  = 0.14,   /* Xt */
            xl1 = 0.21,  /*1/2 per cct */
            xl2 = 0.30,  /* 1/2 per cct */
            xee = 0.05;  /* Xj */
static double  xd = 1.620, /* d-axis reactance */
            xdd = 0.250, /* d-axis transient reactance  */
            xq = 1.600,  /* q-axis reactance */
            Td0d =  5.600, /* d-axis open circuit time
                        constant */
            M = 0.0191 , /*Inertia constant */
            D = 0.00531 ; /* Damping coeffient */
static double  dy1[N],  /* Real power Pe  */
            dy2[N],  /* Terminal voltage Vt */
            dy3[N],  /* SVC  Vc */
            ti,      /* simulation time */
            Ups,Upss, /* Supplementary control signal
*/
            P1, out ;
static double  Pd;  /* small disturbance */
static double  ZZ;  /* used in AVR limit */
static int  DD1,DD2,DXD1,DXD2; /* GOV limit */
static int  AVRL, AVRL1;  /* AVR limit */
static int  PSS,PSS1; /* PSS limit */
static int  kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double  x[],double  t,int n,double
            dy2[N],double dy3[N],int flag1);
void sys1(double  x[],double  f[],double  y[],double
          dy2[N],double dy3[N],int flag1);
unsigned _stack = 50000;
FILE *fpw;

void    main()
```

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีที่มีผู้นำไปใช้โดยมิได้รับอนุญาต ทางสถาบันฯ ขอสงวนสิทธิ์ในการดำเนินคดีตามกฎหมายสูงสุด ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
static double GD;
static double x1[NA], y1[4], f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
fpw=fopen("Pss1.dat","w"); /* open output file */
/* Initial value of variables */
Upss=0;Pd=0.0;ZZ=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
XD2=0.0;PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;Q0=
0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
}
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;  /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;  /*  q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;  /* Inertia constant */
D = 0.00531; /* Damping coeficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0; fri=0; kpt=0.0; ti=0.0;
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" <====== ΔP-PSS  C o n t r o l ======>\n\n\n");
printf("              * wait a minute *                \n");
select = 1; delay(500); clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" <===  Simulation Power System  ===>\n");
printf("  Please select Disturbance.\n");
printf("  large:[0],small:[1] --->");
scanf("     %d",&CH);
printf("\n");
if(CH==0)
{ printf("  Select trip time after 3-phase fault.\n");
  printf("              [ 0 - 0.1s ] --->");
  scanf("   %f",&Time);
  (int)KH=Time/0.005;
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
/* draw graph axis */
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0),
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจำหน่าย จ่าย แจก หรือนำไปเผยแพร่ต่อ อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        "0.30");
    g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0),
        "0.00");
    g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
    g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0),
        "-0.30");

    g_view(40.0, 5.0, 440.0,365.0,-1,-1);
    w_view(40.0, 265.0, 440.0,365.0);
    w_axiss(0.0, 0.1, N * T, -0.1);
    w_line(0.0, 0, 0);

    g_view(0, 0, 639, 399, -1, 1);
    g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
    g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
    g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
    g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0),
        "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0),
        "-0.1");
  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0),
        "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0),
        "-0.1");
}
initial_system();
if(CH==1) Pd = 0.0;  /* small disturbance */
else  Pd = 0.0;
    x_stand(x1); /* initial condition */
    for (kpt = 0;kpt <=N-1;kpt++)
    { ti = (double)kpt * T;  /*current time */
if(CH==0)   /* large disturbance */
 if(kpt==0) flag = 1;  /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)  /* protection trip time */
  { flag = 0;
  xl1 = 2.0*xl1;
  }
if(CH==1)     /* small disturbance */
 if(kpt>=0)
  if (kpt < 100)
   Pd = -0.1/100*kpt;
        if(CH==1)
         if(kpt>=100)
          if (kpt < 200)
           Pd = -0.2 + 0.1 /100*kpt;
        if(CH==1)
         if(kpt>=200)
          Pd = 0.0;
AVRL = AVRL1; DD1  = DXD1; DD2  = DXD2; PSS  =
PSS1;
runge4(x1, T, NA,dy2,dy3,flag);
/* A V R Limit */
if(x1[3] > Efmax - vf0)
{ AVRL1 = 1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจำหน่าย จ่ายแจก หรือทำซ้ำ ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 { AVRL1 = -1;
   ZZ = Efmin - vf0;
 }
 else
 { AVRL1 = 0;
   ZZ = x1[3];
 }
 Ups = out;
 /* PSS Limit */
 if(Ups>Umpss) Ups = Umpss;
 if(Ups<-Umpss) Ups = -Umpss;
 if(Ups>Umpss)
 { PSS1 = 1;
   Upss = Umpss;
 }
 else
 if (Ups<-Umpss)
 { PSS1 = -1;
   Upss = - Umpss;
 }
 else
 { PSS1 = 0.0;
   Upss = Ups;
 }
 /* G O V Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1) DXD1=1;
 else
 if(GD < -0.1) DXD1=-1;
 else
 DXD1=0;
 if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
 else
 if(x1[5] < -(P0-Pd)) DXD2=-1;
 else
 DXD2=0;
 sys1(x1, f, y1,dy2,dy3,flag);
 d_out(y1);
 /* Plot Graph */
 dt1[1] = x1[1];
 dt2[1] = dy1[kpt+1];
 dt3[1] = dy2[kpt+1];
 if(kpt==0) fri =0;
 else
 fri=1;
 if(CH==0)
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,4.5,N*T,-4.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri,1);
   fprintf(fpw,"%f %f %f %f
         %f\n",ti,dt1[1],dt2[1],dt3[1],Upss);
   g_view(80.0,140.0, 480.0, 240.0,1,1);
   w_view(80.0,140.0, 480.0, 240.0);
   w_axis(0.0,0.3,N*T,-0.3);
   w_attr(3, 0xffff);
   w_line2((ti), dt2[1], fri, 2);

   g_view(80.0, 270.0, 480.0, 370.0,1,1);
   w_view(80.0, 270.0, 480.0, 370.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt3[1] , fri, 3);
 }
 else
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,0.5,N*T,-0.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri, 1);

 fprintf(fpw,"%f %f %f %f %f
       \n",ti,dt1[1],dt2[1],dt3[1],Upss);

   g_view(80.0, 140.0, 480.0, 240.0,1,1);
   w_view(80.0, 140.0, 480.0, 240.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt2[1], fri, 2);

   g_view(80.0, 270.0, 480.0, 370.0,1,1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    w_view(80.0, 270.0, 480.0, 370.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt3[1] , fri, 3);
    }
    dt1[0] = dt1[1];
    dt2[0] = dt2[1];
    dt3[0] = dt3[1];
    }
  getch(); closegraph(); clrscr();
  printf("\n\n\n\n\n\n\n\n\n\n");
  printf("      Continous ? (Quit press ESC)");
  stop=getch(); fclose(fpw);
  }
  while(stop!=0x1b);  /* ESC to exit */
  }
  void x_stand(double x[])
  /* initial value of variables */
  {
  x[0] = del0;
  x[1] = 0.0;
  x[2] = fif0;
  x[3] = 0.0;
  x[4] = 0.0;
  x[5] = 0.0;
  x[6] = 0.0;
  x[7] = 0.0;
  x[8] = 0.0;
  x[9] = 0.0;
  x[10] = 0.0;
  x[11]= 0.0;
  x[12]= 0.0;
  x[13]= 0.0;
  x[14]= 0.0;
  x[15]= 0.0;
  x[16]= 0.0;
  }
  void initial_system()
  /* initial power system model at normal operation */
  {
  static double A, C;
  /* 1.delta1 */
  del1 = asin(P0*xl/2.0);
  /* 2. xc */
  xc = xl/(4.0*(1.0 - cos(del1)));
  Bs0 = 1.0/xc;
  /* 3.xe,xa,xb */
  xe = xt+xl1+xl2+xee;
  xa = 0;
  /* 4.delta1 */
  del1 = asin(P0*xe);
  /* 5. Q0 */
  Q0l = (1.0-cos(del1))/xe;
  Q0t = xa;
  Q0  = Q0l - Q0t;
  /* 6.vd0 Calculation */
  A   = ( Vt0 * Vt0 / xq + Q0);
  A   = A * A;
  A   = A + P0 * P0;
  vd0 = P0 * Vt0 / sqrt(A);
  /* 7.iq0 Calculation */
  C   = Vt0 * Vt0 + xq * Q0;
  C   = C * C;
  C   = C + xq * xq * P0 * P0;
  iq0 = P0 * Vt0 / sqrt(C);
  /* 8.vq0 Calculation */
  A   = Vt0 * Vt0 - vd0 * vd0;
  vq0 = sqrt(A);
  /*  9.id0 Calculation */
  id0 = (P0 - vq0 * iq0) / vd0;
  /* 10.del0 Calculation */
  del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);

  /* 11.fif0 Calculation */
  fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
  fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
  /* 12.vf0 Calculation */
  vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
  Xdd;
  /* 13.vc0 calculation */
  vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
      +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
  /* 14.v0 Calculation */
  A = iq0 * xe + vd0 - vd0 * xa * xe;
  C = sin (del0);
  v0 = A/C;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การขาย ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
void d_out(double y1[])
/* output deviation */
{
dy1[kpt+1] = y1[0] - P0;
dy2[kpt+1] = y1[1] - Vt0;
dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[][N],double f[],double
        y[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1;
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;
  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = xt+xl1+xl2+xee;
  xa = 0;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
      +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd -
        xdd)*id;
  break;
}
/* 3: vf */
switch(PSS)
{ case 1 : f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
                x[4])/Tf+Umpss))/Te;
  break;
  case -1:
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
      Umpss))/Te;
  break;
  case 0 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
        x[4])/Tf+Upss))/Te;
  break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
            W0;
  break;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
    /* 6: h */
    switch(DD2)
    { case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
      break;
      case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
      break;
      case  0 :f[6] = (-x[6]+x[5])/Tw;
      break;
    }
    /* 7: SVC -> Vcm */
    f[7] = (dy3[kpt] - x[7]) /Tv;
    f[8] = (x[1]-x[8])/Tww;
    f[9] = (Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-x[9])/Trr;
    f[10] = (x[9]-x[10])/T2;
    f[11] = (T1*(Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-
            x[9])/Trr-x[11])/T2;
    out1 = x[10]+x[11];
    f[12] = (out1-x[12])/T4;
    f[13]=(T3*((x[9]-x[10])/T2+(T1*(Trr*(Kwpss/W0)*(x[1]-
    x[8])/Tww-x[9])/Trr-x[11])/T2)-x[13])/T4;
    out = (x[12]+x[13]);
    }
    void runge4( double x[],double t,int n,
                 double dy2[N],double dy3[N],int flag1)
    /* Runge-Kutta4 for solving differential equation */
    {
    double  k1[NA], k2[NA], k3[NA], k4[NA], y[3],
            z[NA];
    int     i;
    sys1(x, k1, y,dy2,dy3,flag1);
    for (i = 0; i < n; i++)
     z[i] = x[i] + t / 2 * k1[i];
     sys1(z, k2, y,dy2,dy3 ,flag1);
    for (i = 0; i < n; i++)
     z[i] = x[i] + t / 2 * k2[i];
     sys1(z, k3, y,dy2,dy3,flag1);
    for (i = 0; i < n; i++)
     z[i] = x[i] + t * k3[i];
     sys1(z, k4, y,dy2,dy3,flag1);
    for (i = 0; i < n; i++)
     x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
    }
```

3. โปรแกรม PSS3.c

```
/* pss1.c : ΔP-PSS */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\define.h"
#include"c:\gtool.h"
#define  Umpssw  0.1  /* max. pss control signal */
#define  Tw      0.056  /* measuring time constant */
#define  Tr      4.0  /* reset circuit time constant */
#define  PI      M_PI  /* PI constant */
static float  sbpw,sapw,sDrpw;  /* alpha1,alpha2,alpha3 */
static double  vd0,vq0,id0,iq0,del0,del1,
               Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double  xc,xe,xa,xb,xl,
               xt  = 0.14,  /* Xt */
               xl1 = 0.21,  /* 1/2 per cct */
               xl2 = 0.30,  /* 1/2 per cct */
               xee = 0.05;  /* Xj */
static double  xd = 1.620,  /* d-axis reactance */
               xdd = 0.250, /* d-axis transient reactance */
               xq = 1.600, /* q-axis reactance */
               Td0d = 5.600, /* d-axis open circuit time
                                constant */
               M = 0.0191, /* Inertia constant */
               D = 0.00531; /*Damping coeffient */
static double  p1[1][N],
               dy1[N],  /* Real power Pe */
               dy2[N],  /* Terminal voltage Vt */
               dy3[N],  /* SVC  Vc*/
               ti,      /* simulation time */
               Uee,P1,
               Wr,Ups,Upss,out22;
static double  Pd;  /* small disturbance */
static double  ZZ;  /* used in AVR limit */
static double  SDEPW,SEEPW,SDkPW,STkPW; /* fuzzy
                                           control */
static int     DD1,DD2,DXD1,DXD2;  /* GOV limit */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
static int    AVRL, AVRL1;  /* AVR limit */
static int    SVCL, SVCL1;  /* SVC limit */
static int    PSS,PSS1;     /* PSS limit */
static int    kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double t,int n,double dy2
[N],double
            dy3[N],int flag1);
void sys1(double x[],double f[],double y[],double dy2
[N],double dy3[N],int flag1);
void svcfuzw(double p1[][N],double Vt,double Pe,double
            x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
unsigned _stack = 50000;
FILE *fpw;

void main()
{
static double GD;
static double x1[NA], y1[4], f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
/*--- Select Disturbance  Large , Small ---*/
fpw=fopen("f1sml.dat","w");
Ups=0;Upss=0.0;Wr=0.0;
Pd=0.0;ZZ=0.0;
SDEP=0.0;SDEPW=0.0;
SEEP=0.0;SEEPW=0.0;
SDkP=0.0;SDkPW=0.0;
STkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;
SVCL=0.0;SVCL1=0.0;PSS=PSS1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{
 dy1[kpt] = 0.0;
 dy2[kpt] = 0.0;
 dy3[kpt] = 0.0;
 p1[0][kpt]=0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620; /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;  /* q-axis reactance */
Td0d = 5.600; /*  d-axis open circuit time constant */
M = 0.0191;  /* Inertia constant */
D = 0.00531;
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{
 x1[kpt] = 0.0;
 f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{
 dt1[kpt] = 0.0;
 dt2[kpt] = 0.0;
 dt3[kpt] = 0.0;
 dt4[kpt] = 0.0;
}
flag=0;
fri=0;
kpt=0.0;
ti=0.0;
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("<=== Δ-PSS based on Fuzzy Control ===>\n\n");
printf("                Enter to continue          \n");
```

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
getch();select=1;delay(500);clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("    <=== Simulation Power System ===>\n");
printf("          Please select Disturbance.\n");
printf("          large:[0],small:[1] --->");
scanf("    %d",&CH);printf("\n");
if(CH==0)
{ printf("        Select trip time after 3-phase fault.\n");
  printf("        [ 0 - 0.1s ] --->");
  scanf("   %f",&Time);
  (int)KH=Time/0.005;
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
sbpw=0.31;sapw=11;sDrpw=0.89;
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.30");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.00");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.30");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.1");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
initial_system();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งนี้ ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(CH==1) Pd = 0.0;     /*small disturbance*/
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <N-1;kpt++)       /* start loop */
{ ti = (double)kpt * T;            /* current time */
  if(select==1)
  { if(CH==0)          /* large disturbance */
          if(kpt==0) flag = 1;   /* 3-phase fault */
    if(CH==0)
          if(kpt==0+KH)  /* trip fault line */
          { flag = 0;  /* fault is cleared */
            xl1 = 2.0*xl1;
          }
    if(CH==1)      /* small disturbance */
     if(kpt>=0)
          if (kpt < 100)
           Pd = -0.1/100*kpt;
    if(CH==1)
     if(kpt>=100)
          if (kpt < 200)
           Pd = -0.2 + 0.1 /100*kpt;
    if(CH==1)
     if(kpt>=200)
          Pd = 0.0;
  }
 AVRL = AVRL1;
 DD1  = DXD1;
 DD2  = DXD2;
 SVCL = SVCL1;
 PSS  = PSS1;
 runge4(x1,T, NA,dy2,dy3,flag);
 /* A V R Limit */
 if(x1[3] > Efmax - vf0)
 { AVRL1 = 1;
   ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 { AVRL1 = -1;
   ZZ = Efmin - vf0;
 }
 else
 { AVRL1 = 0;
   ZZ = x1[3];
 }
 if(kpt%2==0)  /* sampling time for controller = 0.01s */
  svcfuzw(p1,x1[7],dy1[kpt],Wr);
 else p1[0][kpt] = p1[0][kpt-1];
 Ups = p1[0][kpt];
 if(Ups>Umpssw)
 { PSS1=1;
   Upss = Umpssw ;
 }
 else
 if(Ups<-Umpssw)
 { PSS1=-1;
   Upss = -Umpssw;
 }
 else
 { PSS1=0;
   Upss = Ups;
 }
 /* G O V Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1)
  DXD1=1;
 else
 if(GD < -0.1)
  DXD1=-1;
 else
  DXD1=0;
 if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
 else
 if(x1[5] < -(P0-Pd)) DXD2=-1;
 else DXD2=0;
 sys1(x1,f, y1,dy2,dy3,flag);
 d_out(y1);
 /* Plot  Graph */
 dt1[1] = x1[1]; /* speed deviation */
 dt2[1] = dy1[kpt+1]; /* real power deviation */
 dt3[1] = dy2[kpt+1]; /* terminal voltage deviation */
 if(kpt==0)fri =0;
 else fri=1;
 if(CH==0)
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    w_axis(0.0,4.5,N*T,-4.5);
    w_attr(3, 0xffff);
    w_line2((ti), dt1[1], fri,1);
    fprintf(fpw,"%f %f %f %f
    %f\n",ti,dt1[1],dt2[1],dt3[1],Upss);

    g_view(80.0,140.0, 480.0, 240.0,1,1);
    w_view(80.0,140.0, 480.0, 240.0);
    w_axis(0.0,0.3,N*T,-0.3);
    w_attr(3, 0xffff);
    w_line2((ti), dt2[1], fri, 2);

    g_view(80.0, 270.0, 480.0, 370.0,1,1);
    w_view(80.0, 270.0, 480.0, 370.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt3[1], fri, 3);
  }
  else
  { g_view(80.0, 10.0, 480.0, 110.0,1,1);
    w_view(80.0, 10.0, 480.0, 110.0);
    w_axis(0.0,0.5,N*T,-0.5);
    w_attr(3, 0xffff);
    w_line2((ti), dt1[1], fri, 1);
    fprintf(fpw,"%f %f %f %f
    %f\n",ti,dt1[1],dt2[1],dt3[1],Upss);

    g_view(80.0, 140.0, 480.0, 240.0,1,1);
    w_view(80.0, 140.0, 480.0, 240.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt2[1], fri, 2);

    g_view(80.0, 270.0, 480.0, 370.0,1,1);
    w_view(80.0, 270.0, 480.0, 370.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt3[1], fri, 3);
  }
  dt1[0] = dt1[1];
  dt2[0] = dt2[1];
  dt3[0] = dt3[1];
  }

    getch();closegraph();clrscr();
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf("    Continous ? (Quit press ESC)");
    stop=getch();fclose(fpw);
    }
    while(stop!=0x1b); /* ESC to exit*/
    }
    void x_stand(double x[])
    /* initial value of variables */
    {
    x[0] = del0;
    x[1] = 0.0;
    x[2] = fif0;
    x[3] = 0.0;
    x[4] = 0.0;
    x[5] = 0.0;
    x[6] = 0.0;
    x[7] = 0.0;
    x[8] = 0.0;
    x[9] = 0.0;
    x[10] = 0.0;
    x[11]= 0.0;
    x[12]= 0.0;
    x[13]= 0.0;
    x[14]= 0.0;
    x[15]= 0.0;
    x[16]= 0.0;
    }
    void initial_system()
    /* initial power system model at normal operation */
    {
    static double A, C;
    /* 1.delta1 */
    del1 = asin(P0*xl/2.0);
    /* 2. xc */
    xc = xl/(4.0*(1.0 - cos(del1)));
    Bs0 = 1.0/xc;
    /* 3.xe,xa,xb */
    xe = xt+xl1+xl2+xee;
    xa = 0;
    /* 4.delta1 */
    del1 = asin(P0*xe);
    /* 5. Q0 */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output variable */
{
dy1[kpt+1] = y1[0] - P0;
dy2[kpt+1] = y1[1] - Vt0;
dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double f[],double y[],double dy2
[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1;
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;

  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = xt+xl1+xl2+xee;
  xa = 0;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* S V C Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax - x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin - x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3]) - x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
switch(PSS)
{ case 1 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
        x[4])/Tf+Umpssw))/Te;
  break;
  case -1:
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
        Umpssw))/Te;
  break;
  case 0 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
        x[4])/Tf+Upss))/Te;
  break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] = 0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
            W0;
  break;
}
/* 6: h */
switch(DD2)
{ case 1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case 0 :f[6] = (-x[6]+x[5])/Tw;
  break;
}
f[11] = (x[1]-x[11])/Tw;
f[12] = (x[1]-x[11])/Tw - x[12]/Tr;
Wr = x[12];
}
void runge4( double x[],double t,int n,
        double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differential equations */
{
double k1[NA], k2[NA], k3[NA], k4[NA], y[3], z
[NA];
int i;
sys1(x,k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k1[i];
sys1(z,k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k2[i];
sys1(z,k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t * k3[i];
sys1(z,k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcsitaw(double DEW,double EEW,double
        Dkw,double *Tkw)
/* find angle on phase plane */
{
if(DEW>0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI;
if(DEW>0.0 && EEW<0.0)
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW<0.0)
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI+360.0;
}
void svcsimemw(double Tkw,double aw,double
        *Nkw,double SL)
/* find membership function of phase phane angle */
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การขาย หรือดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(Tkw >= 0.0 && Tkw < SL - aw / 2.0)
*Nkw = 1.0;
if(Tkw >= SL - aw/2.0 && Tkw <  SL + aw / 2.0)
*Nkw = ( - Tkw + SL ) / aw + 0.5;
if(Tkw >= SL + aw/2.0 && Tkw < SL +180.0 - aw/2.0)
*Nkw = 0.0;
if(Tkw >= SL + 180.0 - aw / 2.0 && Tkw <  SL + 180.0 +
aw / 2.0)
*Nkw = ( Tkw - ( SL + 180.0 ) ) / aw + 0.5;
if(Tkw >= SL + 180.0 + aw / 2.0  && Tkw <= 360.0)
*Nkw = 1.0;
}
void svcfuzw(double p1[][N],double Vt,double Pe,double
            x1)
/* fuzzy control */
{
static double SNkPW,SGkPW;
SNkPW = 0.0;
SDEPW = sbpw*(x1-SEEPW)/(2.0*T);
SEEPW = x1;
SDkPW = sqrt(SDEPW*SDEPW + x1*x1);
/* Phase plan selection */
svcsitaw(SDEPW,x1,SDkPW,&STkPW);
svcsimemw(STkPW,sapw,&SNkPW,SL3);
/* determine Gain factor */
SGkPW = SDkPW/sDrpw;
if(SGkPW>1.0)
SGkPW=1.0;
/* calculate the supplementary control signal for AVR */
p1[0][kpt] = (2.0*SNkPW-1.0)*Umpssw*SGkPW;
if(p1[0][kpt] < -Umpssw) p1[0][kpt]=-Umpssw;
if(p1[0][kpt] >  Umpssw) p1[0][kpt]= Umpssw;
}
```

---

4. โปรแกรม SVC.c

```
/* svcc : SVC based on fuzzy control */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<time.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\saminar\gtool.h"
#include"c:\define.h"
#define PI M_PI
static float sbp,sap,sDrp; /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /* 1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
static double xd = 1.620,  /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600,  /* q-axis reactance */
              Td0d = 5.600,/* d-axis open circuit time
                              constant */
              M = 0.0191,  /* Inertia constant */
              D = 0.00531; /* Damping coeffient */
static double u1[1][N],   /* Supplementary control signal
                             for SVC */
              dy1[N],     /* Real power Pe */
              dy2[N],     /* Terminal voltage Vt */
              dy3[N],     /* SVC  Vc */
              ti,         /* simulation time */
              Uee,P1,
              out11,out22;
static double Pd; /* small disturbance */
static double ZZ; /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */
static double Jw1; /* performance index J for large
                      disturbance */
static double Jw2; /* performance index J for small
                      disturbance */
static int DD1,DD2,DXD1,DXD2;  /* GOV limit */
static int AVRL, AVRL1;        /* AVR limit */
static int SVCL, SVCL1;        /* SVC limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
```

สารนิพนธ์นี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
void runge4(double x[],double u1[][N],double t,
        int n,double dy2[N],double dy3[N],int flag1);
void sys1(double x[],double u1[][N],double f[],double
        y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[][N],double Vt,double Pe,double
        x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
unsigned _stack = 50000;
FILE *fpw;

void main()
{
static double GD;
static double x1[NA], y1[4], f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
fpw=fopen("nousvcl.dat","w");
Uee=0;Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
XD2=0.0;SVCL=0.0;SVCL1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0][kpt] = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD=0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;
/*--- Select Disturbance  Large , Small ---*/
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("    <======   C o n t r o l   ======>\n");
printf("    Please select control method.\n");
printf("    [1] no control signal(Usvc)\n");
printf("    [2] FUZZY control\n");
printf("    selection is --->");
scanf("    %d",&select);
printf("\n");
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("    <===  Simulation Power System  ===>\n");
printf("    Please select Disturbance.\n");
printf("    large:[0],small:[1] --->");
scanf("    %d",&CH);
printf("\n");
if(CH==0)
{ printf("          Select trip time after 3-phase fault.\n");
  printf("                    [ 0 - 0.1s ] --->");
  scanf(" %f",&Time);
  (int)KH=Time/0.005;
}
if(select==2)
{         clrscr();
```

กสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        printf("\n\n\n\n\n\n\n\n\n\n");
        printf("<==== Turning  Parameter ====>\n");
        printf("                    alpha1 --->");
        scanf("     %f",&sbp);
        printf("\n");
        printf("                    alpha2 --->");
        scanf("     %f",&sap);
        printf("\n");
        printf("                    alpha3 --->");
        scanf("     %f",&sDrp);
        printf("\n");
}

flag = 0;
xl = xt + xl1 + xl2 + xee;
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.0");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.3");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.3");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.0");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.5");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.1");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
```

สารนิพนธ์ฉบับนี้เป็นการรวบรวมข้อมูลเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ในทางการค้า ในกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
initial_system();
if(CH==1) Pd = 0.0;   /* small disturbance */
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <=N-1;kpt++)     /* start loop */
{ ti = (double)kpt * T;              /* current time */
 if(CH==0)             /* large disturbance */
  if(kpt==0) flag = 1;   /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)         /* fault line trip */
  { flag = 0;
        xl1 = 2.0*xl1;
  }
 if(CH==1)      /* small disturbance */
  if(kpt>=0)
   if (kpt < 100)
        Pd = -0.1/100*kpt;
 if(CH==1)
  if(kpt>=100)
   if (kpt < 200)
        Pd = -0.2 + 0.1 /100*kpt;
 if(CH==1)
  if(kpt>=200)
   Pd = 0.0;
if(select==2)
 if(kpt%2==0)  /* sampling time for controller = 0.01s */
  svcfuz(u1,x1[7],dy1[kpt],x1[9]);
 else u1[0][kpt] = u1[0][kpt-1];
AVRL = AVRL1;
DD1  = DXD1;
DD2  = DXD2;
SVCL = SVCL1;
runge4(x1, u1, T, NA,dy2,dy3,flag);
/* A V R Limit */
if(x1[3] > Efmax - vf0)
{ AVRL1 = 1;
  ZZ = Efmax - vf0;
}
else
if(x1[3] < Efmin - vf0)
{ AVRL1 = -1;
  ZZ = Efmin - vf0;
}
else
{ AVRL1 = 0;
  ZZ = x1[3];
}
/* G O V Limit */
GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
if(GD > 0.1) DXD1=1;
else
if(GD < -0.1) DXD1=-1;
else DXD1=0;
if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
else
if(x1[5] < -(P0-Pd)) DXD2=-1;
else DXD2=0;
/* S V C Limit */
if (x1[10] > 1.0 - Bs0)
{ SVCL1 = 1;
 x1[10] = 1.0 - Bs0;
}
else
if (x1[10] < -1.0 - Bs0)
{ SVCL1 = -1;
 x1[10] = -1.0 - Bs0;
}
else SVCL1 = 0;
xc = 1.0 / (x1[10] + Bs0);
sys1(x1, u1, f, y1,dy2,dy3,flag);
d_out(y1);
/* calulation Performance index J */
if(CH==0)
{ Jw1 += fabs(x1[1])* ti+fabs((u1[0][kpt]-u1[0][kpt-
1])/T);
}
if(CH==1)
{ Jw2 += fabs(x1[1])* ti+fabs((u1[0][kpt]-u1[0][kpt-
1])/T);
}
/* Plot  Graph */
dt1[1] = x1[1]; /* speed deviation */
dt2[1] = dy1[kpt+1];  /* real power deviation */
dt3[1] = dy2[kpt+1];  /* terminal voltage deviation */
if(kpt==0) fri =0;
else fri=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(CH==0)
{ g_view(80.0, 10.0, 480.0, 110.0,1,1);
  w_view(80.0, 10.0, 480.0, 110.0);
  w_axis(0.0,4.5,N*T,-4.5);
  w_attr(3, 0xffff);
  w_line2((ti), dt1[1], fri,1);
  fprintf(fpw," %f %f %f %f %f %f
  \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt]);

  g_view(80.0,140.0, 480.0, 240.0,1,1);
  w_view(80.0,140.0, 480.0, 240.0);
  w_axis(0.0,0.3,N*T,-0.3);
  w_attr(3, 0xffff);
  w_line2((ti), dt2[1], fri, 2);

  g_view(80.0, 270.0, 480.0, 370.0,1,1);
  w_view(80.0, 270.0, 480.0, 370.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt3[1], fri, 3);
}
else
{ g_view(80.0, 10.0, 480.0, 110.0,1,1);
  w_view(80.0, 10.0, 480.0, 110.0);
  w_axis(0.0,0.5,N*T,-0.5);
  w_attr(3, 0xffff);
  w_line2((ti), dt1[1], fri, 1);
  fprintf(fpw," %f %f %f %f %f %f
  \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt]);

  g_view(80.0, 140.0, 480.0, 240.0,1,1);
  w_view(80.0, 140.0, 480.0, 240.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt2[1], fri, 2);

  g_view(80.0, 270.0, 480.0, 370.0,1,1);
  w_view(80.0, 270.0, 480.0, 370.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt3[1], fri, 3);
}
dt1[0] = dt1[1];
dt2[0] = dt2[1];
dt3[0] = dt3[1];
}
getch();closegraph();clrscr();
printf("\n\n\n\n\n\n\n\n\n\n\n");
printf("                Continous ? (Quit press ESC)");
stop=getch();
fclose(fpw);
}
while(stop!=0x1b); /* ESC to exit */
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power sytem model at normal operation */
{
static double A, C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
```

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจำหน่าย จ่ายแจก หรือนำมาให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/*  6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/*  9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* outpt deviation */
{  dy1[kpt+1] = y1[0] - P0;
   dy2[kpt+1] = y1[1] - Vt0;
   dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[][N],double f[],double y
[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations of power system model */
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{  A1 = xl1*xc/(xc-xl1);
   A2 = xl2+xee;
   B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
   B2 = xl1*A1/(3.0*xl1+A1);
   B3 = 2.0*xl1*A1/(3.0*xl1+A1);
   C1 = xt+B1;
   C2 = A2+B3;

   xe = (C1*C2+B2*(C1+C2))/B2;
   xa = (C1*C2+B2*(C1+C2))/C2;
   xa = -1.0/xa;
   xb = (C1*C2+B2*(C1+C2))/C1;
  }
if(flag1==0)
{  xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
   xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
   xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
   xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf))/Te;
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] / W0;
  break;
}
/* 6: h */
switch(DD2)
{ case 1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case 0 :f[6] = (-x[6]+x[5])/Tw;
  break;
}
/* 7: SVC : Vcm */
f[7] = (-dy3[kpt] - x[7]) /Tv;
/* 8: SVC : Pin          */
f[8] = ((y[0] - (P0 -Pd )) - x[8]) / TP1;
/* 9: SVC : Pout */
f[9] = ((y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
/* 10:SVC : Bs */
if(select==1)
{ switch(SVCL)
  { case 1 :f[10] = (-x[10]+Ksv * (x[7])) / Tc;
    break;
    case -1 :f[10] = (-x[10]+Ksv * (x[7])) / Tc;
    break;
    case 0 :f[10] = (-x[10]+Ksv * (x[7])) / Tc;
    break;
  }
}
if(select==2)
{ switch(SVCL)
  { case 1 :f[10] = (-x[10]+Ksv * (x[7]+/*u1[0]
[kpt]*/Umfp)) / Tc;
    break;
    case -1 :f[10] = (-x[10]+Ksv * (x[7]-/*u1[0]
[kpt]*/Umfp)) / Tc;
    break;
    case 0 :f[10] = (-x[10]+Ksv * (x[7]+u1[0][kpt])) / Tc;
    break;
  }
}
}
void runge4(double x[],double u1[][N],double t,int n,
                double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differential equations */
{
double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
int i;
sys1(x, u1, k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k1[i];
sys1(z, u1, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k2[i];
sys1(z, u1, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t * k3[i];
```

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sys1(z, u1, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcfuz(double u1[][N],double Vt,double Pe,double
x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0) SGkP=1.0;
/* calculate the SVC supplementary control signal */
u1[0][kpt] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[0][kpt] < -Umfp) u1[0][kpt]=-Umfp;
if(u1[0][kpt] >  Umfp) u1[0][kpt]= Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI+360.0;
}
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

---

5. โปรแกรม svc_pss1.c

```
/* svc_pss1 : SVC+PSS1 control */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\gtool.h"
#include"c:\define.h"
#define Umpss   0.1
#define Kpss    1
#define T1      0.04
#define T2      0.05
#define PI      M_PI
static float sbp,sap,sDrp;  /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /* 1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
static double xd = 1.620,  /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600,  /* q-axis reactance */
              Td0d = 5.600,/* d-axis open circuit time
                              constant */
              M = 0.0191,  /* Inertia constant          */
              D = 0.00531; /* Damping coeffient */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
static double u1[1][N],   /* supplementary control signal
                             for SVC */
              dy1[N],     /* Real power Pe */
              dy2[N],     /* Terminal voltage Vt */
              dy3[N],     /* SVC Vc */
              ti,         /* simulation time */
              Ups,Upss,
              P1,out,out22;
static double Pd; /* small disturbance */
static double ZZ; /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */
static double Jw1;    /* performance index for large
                         disturbance */
static double Jw2;    /* performance index for small
                         disturbabce */
static int DD1,DD2,DXD1,DXD2;   /* GOV limit */
static int AVRL, AVRL1;         /* AVR limit */
static int SVCL, SVCL1;         /* SVC limit */
static int PSS,PSS1;            /* PSS limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double u1[][N],double  t,int
        n,double dy2[N],double dy3[N],int flag1);
void sys1(double  x[],double  u1[][N],double  f[],double
      y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[][N],double Vt,double Pe,double
        x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
unsigned _stack = 50000;
FILE *fpw;

void main()
{
static double GD;
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
fpw=fopen("dpsvcl.dat","w");
Ups=0;Upss=0;Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
XD2=0.0;SVCL=0.0;SVCL1=0.0;
PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0][kpt] = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Dampling coefficient */
GD=0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;
/*--- Select Disturbance  Large , Small ---*/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การขาย การจัดทำซ้ำ การดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("    <======= SVC+PSS1 Control =======>\n\n");
printf("                    Enter to continue \n");
getch();select=1;clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("    <=== Simulation Power System ===>\n");
printf("                    Please select Disturbance.\n");
printf("                    large:[0],small:[1] --->");
scanf("    %d",&CH);
printf("\n");
if(CH==0)
{ printf("    Select trip time after 3-phase fault.\n");
  printf("    [ 0 - 0.1s ] --->");
  scanf("  %f",&Time);
  (int)KH=Time/0.005;
}
if(select==1)
{ clrscr();
  printf("\n\n\n\n\n\n\n\n\n\n");
  printf("  <======= Turning  Parameter =======>\n");
  printf("                    alpha1 --->");
  scanf("  %f",&sbp);
  printf("\n");
  printf("                    alpha2 --->");
  scanf("  %f",&sap);
  printf("\n");
  printf("                    alpha3 --->");
  scanf("  %f",&sDrp);
  printf("\n");
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.30");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.00");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.30");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.1");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
initial_system();
if(CH==1) Pd = 0.0;    /* small disturbance */
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <=N-1;kpt++)     /* start loop */
{ ti = (double)kpt * T;           /* current time */
 if(CH==0)                        /*large disturbance */
  if(kpt==0) flag = 1;   /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)          /* fault line trip */
  { flag = 0;
        xl1 = 2.0*xl1;
  }
 if(CH==1)                /* small disturbance*/
  if(kpt>=0)
   if (kpt < 100)
        Pd = -0.1/100*kpt;
 if(CH==1)
  if(kpt>=100)
   if (kpt < 200)
        Pd = -0.2 + 0.1 /100*kpt;
 if(CH==1)
  if(kpt>=200)
   Pd = 0.0;
if(select==1)if(kpt%2==0)  /* sampling time for controller
                           = 0.01s */
  svcfuz(u1,x1[7],dy1[kpt],x1[15]);
 else u1[0][kpt] = u1[0][kpt-1];
 AVRL = AVRL1;
 DD1  = DXD1;
 DD2  = DXD2;
 SVCL = SVCL1;
 PSS  = PSS1;
 runge4(x1, u1, T, NA,dy2,dy3,flag);
 /* AVR Limit */
 if(x1[3] > Efmax - vf0)
 { AVRL1 = 1;
   ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 { AVRL1 = -1;
   ZZ = Efmin - vf0;
 }
 else
 { AVRL1 = 0;
   ZZ = x1[3];
 }
 Ups = Kpss*out;
 if(Ups>Umpss) Ups = Umpss;
 if(Ups<-Umpss) Ups = -Umpss;
 if(Ups>Umpss)
 { PSS1 = 1;
   Upss = Umpss;
 }
 else
 if (Ups<-Umpss)
 { PSS1 = -1;
   Upss = - Umpss;
 }
 else
 { PSS1 = 0.0;
   Upss = Ups;
 }
 /* GOV Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1) DXD1=1;
 else
 if(GD < -0.1) DXD1=-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
else DXD1=0;
if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
else
if(x1[5] < -(P0-Pd)) DXD2=-1;
else DXD2=0;
/* SVC Limit */
if (x1[13] > 1.0 - Bs0)
{ SVCL1 = 1;
  x1[13] = 1.0 - Bs0;
}
else
if (x1[10] < -1.0 - Bs0)
{ SVCL1 = -1;
  x1[13] = -1.0 - Bs0;
}
else SVCL1 = 0;
xc = 1.0 / (x1[13] + Bs0);
sys1(x1, u1, f, y1,dy2,dy3,flag);
d_out(y1);
/* calulation performance index J */
if(CH==0)
{ Jw1 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
}
if(CH==1)
{ Jw2 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
}
/* Plot Graph */
dt1[1] = x1[1];
dt2[1] = dy1[kpt+1];
dt3[1] = dy2[kpt+1];
if(kpt==0)fri =0;
else fri=1;
if(CH==0)
{ g_view(80.0, 10.0, 480.0, 110.0,1,1);
  w_view(80.0, 10.0, 480.0, 110.0);
  w_axis(0.0,4.5,N*T,-4.5);
  w_attr(3, 0xffff);
  w_line2((ti), dt1[1], fri,1);
  fprintf(fpw," %f %f %f %f %f %f %f
  \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);
  g_view(80.0,140.0, 480.0, 240.0,1,1);
  w_view(80.0,140.0, 480.0, 240.0);
  w_axis(0.0,0.3,N*T,-0.3);
  w_attr(3, 0xffff);
  w_line2((ti), dt2[1], fri, 2);

  g_view(80.0, 270.0, 480.0, 370.0,1,1);
  w_view(80.0, 270.0, 480.0, 370.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt3[1]/*Upss*/, fri, 3);
}
else
{ g_view(80.0, 10.0, 480.0, 110.0,1,1);
  w_view(80.0, 10.0, 480.0, 110.0);
  w_axis(0.0,0.5,N*T,-0.5);
  w_attr(3, 0xffff);
  w_line2((ti), dt1[1], fri, 1);
  fprintf(fpw," %f %f %f %f %f %f %f
  \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);
  g_view(80.0, 140.0, 480.0, 240.0,1,1);
  w_view(80.0, 140.0, 480.0, 240.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt2[1], fri, 2);

  g_view(80.0, 270.0, 480.0, 370.0,1,1);
  w_view(80.0, 270.0, 480.0, 370.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt3[1], fri, 3);
}
dt1[0] = dt1[1];
dt2[0] = dt2[1];
dt3[0] = dt3[1];
}
getch();closegraph();clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" Continous ? (Quit press ESC)");
stop=getch();
fclose(fpw);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
while(stop!=0x1b); /* ESC to exit */
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double A, C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A  = ( Vt0 * Vt0 / xq + Q0);
A  = A * A;
A  = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C  = Vt0 * Vt0 + xq * Q0;
C  = C * C;
C  = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A  = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
  +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{ dy1[kpt+1] = y1[0] - P0;
  dy2[kpt+1] = y1[1] - Vt0;
  dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[][N],double f[],double y
[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
double      vd,vq,id,iq;
if(flag1==1)
{  A1 = xl1*xc/(xc-xl1);
   A2 = xl2+xee;
   B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
   B2 = xl1*A1/(3.0*xl1+A1);
   B3 = 2.0*xl1*A1/(3.0*xl1+A1);
   C1 = xt+B1;
   C2 = A2+B3;

   xe = (C1*C2+B2*(C1+C2))/B2;
   xa = (C1*C2+B2*(C1+C2))/C2;
   xa = -1.0/xa;
   xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{  xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
   xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
   xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
   xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f       */
switch(AVRL)
{  case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
   break;
   case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
   break;
   case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
   break;
}
/* 3: vf */
switch(PSS)
{  case 1 :
   f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
          x[4])/Tf+Umpss))/Te;
   break;
   case -1:
   f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
          Umpss))/Te;
   break;
   case 0 :
   f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
          x[4])/Tf+Upss))/Te;
   break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{  case 1 :f[5] =  0.1;
   break;
   case -1:f[5] = -0.1;
   break;
   case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
                W0;
   break;
}
/* 6: h */
switch(DD2)
{  case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
   break;
   case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
   break;
```

สารนิพนธ์เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการ ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  case  0 :f[6] = (-x[6]+x[5])/Tw;
  break;
 }
 /* 7: SVC : Vcm */
 f[7] = (-dy3[kpt] - x[7]) /Tv;
 /* 8: SVC : Pin          */
 f[8] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1;
 /* 9: SVC : Pout */
 f[9] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
 f[10] =  (x[9]-x[10])/T2;
 f[11] = (T1*((-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1)-x
 [11])/T2;
 x[12] =x[10]+x[11];
 out = x[12];
 f[14] = ((y[0]-(P0-Pd))-x[14])/TP1;
 f[15] = ((y[0]-(P0-Pd))-x[14])/TP1-x[15]/TQ1;
 /* 10:SVC : Bs */
 if(select==1)
 { switch(SVCL)
   { case  1 :f[13] = (-x[13]+Ksv *(x[7]+u1[0][kpt])) / Tc;
     break;
     case -1 :f[13] = (-x[13]+Ksv * (x[7]+u1[0][kpt])) / Tc;
     break;
     case  0 :f[13] = (-x[13]+Ksv * (x[7]+u1[0][kpt])) / Tc;
     break;
   }
 }
 }
 void runge4(double x[],double u1[][N],double t,int n,
             double dy2[N],double dy3[N],int flag1)
 /* Runge-Kutta4 for solving differential equations */
 {
 double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
 int i;
 sys1(x, u1, k1, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k1[i];
 sys1(z, u1, k2, y,dy2,dy3 ,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k2[i];
 sys1(z, u1, k3, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t * k3[i];
 sys1(z, u1, k4, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
 x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
 }
 void svcfuz(double u1[][N],double Vt,double Pe,double
 x1)
 /* fuzzy control */
 {
 static double SNkP,SGkP;
 SNkP = 0.0;
 SDEP = sbp*(x1-SEEP)/(2.0*T);
 SEEP = x1;
 SDkP = sqrt(SDEP*SDEP + x1*x1);
 /* Phase plan selection          */
 svcsita(SDEP,x1,SDkP,&STkP);
 svcsimem(STkP,sap,&SNkP,SL3);
 /* determine Gain factor */
 SGkP = SDkP/sDrp;
 if(SGkP>1.0) SGkP=1.0;
 /* calculate the SVC supplementary control signal */
 u1[0][kpt] = (2.0*SNkP-1.0)*Umfp*SGkP;
 if(u1[0][kpt] < -Umfp) u1[0][kpt]=-Umfp;
 if(u1[0][kpt] >  Umfp) u1[0][kpt]= Umfp;
 }
 void svcsita(double DE,double EE,double Dk,double *Tk)
 /* find angle on phase plane */
 {
 if(DE>0.0 && EE>0.0)
 *Tk= asin(DE/Dk)*180.0/PI;
 if(DE>0.0 && EE<0.0)
 *Tk= -asin(DE/Dk)*180.0/PI+180.0;
 if(DE<0.0 && EE<0.0)
 *Tk= -asin(DE/Dk)*180.0/PI+180.0;
 if(DE<0.0 && EE>0.0)
 *Tk= asin(DE/Dk)*180.0/PI+360.0;
 }
 void svcsimem(double Tk,double a,double *Nk,double
 SL)
 /* fine membership function of phase plane angle */
 {
 if(Tk >= 0.0 && Tk < SL - a / 2.0)
 *Nk = 1.0;
 if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การขาย หรือการใช้งานอื่นๆ ทั้งนี้ ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารนี้ทุกครั้งที่มีการนำไปใช้

```
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

---

6. โปรแกรม svc_pss2.c

```
/* svc_pss2 : SVC+PSS2 control */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\gtool.h"
#include"c:\define.h"
#define Umpss   0.1
#define Tww     0.056
#define Trr     4.0
#define T1      0.3
#define T2      0.02
#define T3      0.2
#define T4      0.02
#define Kwpss   13.0
#define PI      M_PI
static float sbp,sap,sDrp;  /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /* 1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
static double xd = 1.620,  /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600,  /* q-axis reactance */
              Td0d = 5.600, /* d-axis open circuit time
                               constant */
              M = 0.0191,  /* Inertia constant */
              D = 0.00531; /* Damping coeffient        */
 static double u1[1][N],   /* supplementaru control signal
                              for SVC */
              dy1[N],     /* Real power Pe */
              dy2[N],     /* Terminal voltage Vt */
              dy3[N],     /* SVC  Vc */
              ti,         /* simulation time */
              Ups,Upss,
              P1,out,out1,out22;
 static double Pd;  /* small disturbance */
 static double ZZ;  /* used in AVR limit */
 static double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */
 static double Jw1;  /* performance index for large
                        disturbance */
 static double Jw2;  /* performance index for small
                        disturbancc */
 static int DD1,DD2,DXD1,DXD2;    /* GOV limit */
 static int AVRL, AVRL1;          /* AVR limit */
 static int SVCL, SVCL1;          /* SVC limit */
 static int PSS,PSS1;             /* PSS limit */
 static int kpt,CH,KH,select;
 void initial_system(void);
 void x_stand(double x[]);
 void d_out(double y1[]);
 void runge4(double x[],double u1[][N],double t,int
             n,double dy2[N],double dy3[N],int flag1);
 void sys1(double x[],double u1[][N],double f[],double
           y[],double dy2[N],double dy3[N],int flag1);
 void svcfuz(double u1[][N],double Vt,double Pe,double
             x1);
 void svcsita(double DE,double EE,double Dk,double
 *Tk);
 void svcsimem(double Tk,double a,double *Nk,double
 SL);
 unsigned _stack = 50000;
 FILE *fpw;
 void main()
 {
 static double GD;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
fpw=fopen("dwsvclga.dat","w");
Ups=0;Upss=0;Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
XD2=0.0;
SVCL=0.0;SVCL1=0.0;PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0][kpt] = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;
/*--- Select Disturbance  Large , Small ---*/
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("     <======= SVC+PSS2 Control
=======>\n\n");
printf("                    Enter to continue \n");
getch();select=1;clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("     <=== Simulation Power System  ===>\n");
printf("                    Please select Disturbance.\n");
printf("                    large:[0],small:[1] --->");
scanf("     %d",&CH);
printf("\n");
if(CH==0)
{ printf("            Select trip time after 3-phase fault.\n");
  printf("                         [ 0 - 0.1s ] --->");
  scanf("   %f",&Time);
  (int)KH=Time/0.005;
}
if(select==1)
{ clrscr();
  printf("\n\n\n\n\n\n\n\n\n\n");
  printf("   <======= Turning  Parameter =======>\n");
  printf("                    alpha1 --->");
  scanf("   %f",&sbp);
  printf("\n");
  printf("                    alpha2 --->");
  scanf("   %f",&sap);
  printf("\n");
  printf("                    alpha3 --->");
  scanf("   %f",&sDrp);
  printf("\n");
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากำไรเกินควร อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.30");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.00");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.30");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
}
else
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.1");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
 }
 initial_system();
 if(CH==1) Pd = 0.0;   /* small disturbance */
 else Pd = 0.0;
 x_stand(x1);
 for (kpt = 0;kpt <=N-1;kpt++)     /* start loop */
 { ti = (double)kpt * T;           /* current time */
 if(CH==0)                         /* large disturbance */
 if(kpt==0) flag = 1;   /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)                    /* fault line trip */
  { flag = 0;
    xl1 = 2.0*xl1;
  }
 if(CH==1)              /* small disturbance */
  if(kpt>=0)
   if (kpt < 100)
    Pd = -0.1/100*kpt;
 if(CH==1)
  if(kpt>=100)
   if (kpt < 200)
        Pd = -0.2 + 0.1 /100*kpt;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า ทำกรณีไฟล์ดิจิทัล อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if(CH==1)
   if(kpt>=200)
     Pd = 0.0;
 if(select==1)if(kpt%2==0)  /* sampling time for controller
                              =0.01s */
  svcfuz(u1,x1[7],dy1[kpt],x1[15]);
 else u1[0][kpt] = u1[0][kpt-1];
 AVRL = AVRL1;
 DD1  = DXD1;
 DD2  = DXD2;
 SVCL = SVCL1;
 PSS  = PSS1;
 runge4(x1, u1, T, NA,dy2,dy3,flag);
 /* AVR Limit */
 if(x1[3] > Efmax - vf0)
 {  AVRL1 = 1;
    ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 {  AVRL1 = -1;
    ZZ = Efmin - vf0;
 }
 else
 {  AVRL1 = 0;
    ZZ = x1[3];
 }
 Ups = out;
 if(Ups>Umpss) Ups = Umpss;
 if(Ups<-Umpss) Ups = -Umpss;
 if(Ups>Umpss)
 {  PSS1 = 1;
    Upss = Umpss;
 }
 else
 if (Ups<-Umpss)
 {  PSS1 = -1;
    Upss = - Umpss;
 }
 else
 {  PSS1 = 0.0;
    Upss = Ups;
 }
 /* GOV Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1) DXD1=1;
 else
 if(GD < -0.1) DXD1=-1;
 else DXD1=0;
 if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
 else
 if(x1[5] < -(P0-Pd)) DXD2=-1;
 else DXD2=0;
 /* SVC */
 if (x1[16] > 1.0 - Bs0)
 {  SVCL1 = 1;
    x1[16] = 1.0 - Bs0;
 }
 else if (x1[16] < -1.0 - Bs0)
 {  SVCL1 = -1;
    x1[16] = -1.0 - Bs0;
 }
 else SVCL1 = 0;
 xc = 1.0 / (x1[16] + Bs0);
 sys1(x1, u1, f, y1,dy2,dy3,flag);
 d_out(y1);
 /* calulation performance index J */
 if(CH==0)
 {  Jw1 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
 }
 if(CH==1)
 {  Jw2 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
 }
 /* Plot Graph  */
 dt1[1] = x1[1]; /* speed deviation */
 dt2[1] = dy1[kpt+1]; /* real power deviation */
 dt3[1] = dy2[kpt+1]; /* terminal voltage deviation */
 if(kpt==0)fri =0;
 else fri=1;
 if(CH==0)
 {  g_view(80.0, 10.0, 480.0, 110.0,1,1);
    w_view(80.0, 10.0, 480.0, 110.0);
    w_axis(0.0,4.5,N*T,-4.5);
    w_attr(3, 0xffff);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการค้าทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    w_line2((ti), dt1[1], fri,1);
    fprintf(fpw," %f %f %f %f %f %f %f
    \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);

    g_view(80.0,140.0, 480.0, 240.0,1,1);
    w_view(80.0,140.0, 480.0, 240.0);
    w_axis(0.0,0.3,N*T,-0.3);
    w_attr(3, 0xffff);
    w_line2((ti), dt2[1], fri, 2);

    g_view(80.0, 270.0, 480.0, 370.0,1,1);
    w_view(80.0, 270.0, 480.0, 370.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt3[1]/*Upss*/, fri, 3);
  }
  else
  { g_view(80.0, 10.0, 480.0, 110.0,1,1);
    w_view(80.0, 10.0, 480.0, 110.0);
    w_axis(0.0,0.5,N*T,-0.5);
    w_attr(3, 0xffff);
    w_line2((ti), dt1[1], fri, 1);
    fprintf(fpw," %f %f %f %f %f %f %f
    \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);

    g_view(80.0, 140.0, 480.0, 240.0,1,1);
    w_view(80.0, 140.0, 480.0, 240.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt2[1], fri, 2);

    g_view(80.0, 270.0, 480.0, 370.0,1,1);
    w_view(80.0, 270.0, 480.0,370.0);
    w_axis(0.0,0.1,N*T,-0.1);
    w_attr(3, 0xffff);
    w_line2((ti), dt3[1], fri, 3);
  }
  dt1[0] = dt1[1];
  dt2[0] = dt2[1];
  dt3[0] = dt3[1];
  }
  getch();
  closegraph();
  clrscr();
  printf("\n\n\n\n\n\n\n\n\n\n\n");
  printf("                    Continous ? (Quit press ESC)");
  stop=getch();
  fclose(fpw);
  }
  while(stop!=0x1b); /* ESC to exit */
  }
  void x_stand(double x[])
  /* initial value of variables */
  {
  x[0] = del0;
  x[1] = 0.0;
  x[2] = fif0;
  x[3] = 0.0;
  x[4] = 0.0;
  x[5] = 0.0;
  x[6] = 0.0;
  x[7] = 0.0;
  x[8] = 0.0;
  x[9] = 0.0;
  x[10] = 0.0;
  x[11]= 0.0;
  x[12]= 0.0;
  x[13]= 0.0;
  x[14]= 0.0;
  x[15]= 0.0;
  x[16]= 0.0;
  }
  void initial_system()
  /* initial power system model at normal operation */
  {
  static double A,C;
  /* 1.delta1 */
  del1 = asin(P0*xl/2.0);
  /* 2. xc */
  xc = xl/(4.0*(1.0 - cos(del1)));
  Bs0 = 1.0/xc;
  /* 3.xe,xa,xb */
  xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
  (xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{ dy1[kpt+1] = y1[0] - P0;
  dy2[kpt+1] = y1[1] - Vt0;
  dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[][N],double f[],double y
[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
double     vd,vq,id,iq;
if(flag1==1)
{  A1 = xl1*xc/(xc-xl1); /*xl1;*/
   A2 = xl2+xee;
   B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
   B2 = xl1*A1/(3.0*xl1+A1);
   B3 = 2.0*xl1*A1/(3.0*xl1+A1);
   C1 = xt+B1;
   C2 = A2+B3;

   xe = (C1*C2+B2*(C1+C2))/B2;
   xa = (C1*C2+B2*(C1+C2))/C2;
   xa = -1.0/xa;
   xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{  xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
   xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
   xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
   xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การจำหน่าย จ่ายแจก หรือแสวงหากำไร ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
switch(PSS)
{ case 1 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
      x[4])/Tf+Umpss))/Te;
  break;
  case -1:
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
      Umpss))/Te;
  break;
  case 0 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
      x[4])/Tf+Upss))/Te;
  break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
          W0;
  break;
}
/* 6: h */
switch(DD2)
{ case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case  0 :f[6] = (-x[6]+x[5])/Tw;
  break;
}
/* 7: SVC : Vcm */
f[7] = (-dy3[kpt] - x[7]) /Tv;
/* 8: SVC : Pin          */
f[14] = ((y[0] - (P0 -Pd )) - x[14]) / TP1;
/* 9: SVC : Pout */
f[15] = ((y[0] - (P0 -Pd )) - x[14]) / TP1 -x[15]/ TQ1;
/* 10 : PSS */
f[8] = (x[1]-x[8])/Tww;
f[9] = (Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-x[9])/Trr;
f[10] = (x[9]-x[10])/T2;
f[11] = (T1*(Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-x[9])/Trr-
x[11])/T2;
out1 = x[10]+x[11];
f[12] = (out1-x[12])/T4;
f[13]=(T3*((x[9]-x[10])/T2+(T1*(Trr*(Kwpss/W0)*(x[1]-
x[8])/Tww-x[9])/Trr-x[11])/T2)-x[13])/T4;
out = (x[12]+x[13]);
/* 11:SVC : Bs */
if(select==1)
{ switch(SVCL)
  { case  1 :f[16] = (-x[16]+Ksv * (x[7]+Umfp)) / Tc;
    break;
    case -1 :f[16] = (-x[16]+Ksv * (x[7]-Umfp)) / Tc;
    break;
    case  0 :f[16] = (-x[16]+Ksv * (x[7]+u1[0][kpt])) / Tc;
    break;
  }
}
}
void runge4(double x[],double u1[][N],double t,int n,
            double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differential equations */
{
double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
```

เอกสารนี้ในเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int i;
sys1(x, u1, k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k1[i];
sys1(z, u1, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k2[i];
sys1(z, u1, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t * k3[i];
sys1(z, u1, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcfuz(double u1[][N],double Vt,double Pe,double
x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection          */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0) SGkP=1.0;
/* calculate the SVC supplementary control signal */
u1[0][kpt] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[0][kpt] < -Umfp) u1[0][kpt]=-Umfp;
if(u1[0][kpt] >  Umfp) u1[0][kpt]= Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/PI+360.0;
}
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

---

7. โปรแกรม svc_pss3.c

```
/* svc_pss3 : SVC+PSS3 control */
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\gtool.h"
#include"c:\define.h"
#define Umpssw  0.1
#define Tw      0.056
#define Tr      4.0
#define PI      M_PI
static float sbp,sap,sDrp;  /* alpha1,alpha2,alpha3 */
static float sbpw,sapw,sDrpw;
/*alphaw1,alphaw2,alphaw3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การกระทำใดๆ ต่อเอกสารนี้ ไม่ว่าจะเป็นการดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            xt  = 0.14,  /* Xt */
            xl1 = 0.21,  /* 1/2 per cct */
            xl2 = 0.30,  /* 1/2 per cct */
            xee = 0.05;  /* Xj */
static double xd = 1.620,  /* d-axis reactance */
            xdd = 0.250, /* d-axis transient reactance */
            xq = 1.600,  /* q-axis reactance */
            Td0d = 5.600,/* d-axis open circuit time
                            constant          */
            M = 0.0191,  /* Inertia constant */
            D = 0.00531; /* Damping coeffient */
static double u1[1][N],   /* supplementary control signal
                             for SVC */
            p1[1][N],    /* supplementary control signal
                             for PSS */
            dy1[N],      /* Real power Pe */
            dy2[N],      /* Terminal voltage Vt */
            dy3[N],      /* SVC  Vc*/
            ti,          /* simulation time */
            Uee,P1,
            Wr,Ups,Upss,out22;
static double Pd;  /* small disturbance */
static double ZZ;  /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP;   /* fuzzy control
*/
static double SDEPW,SEEPW,SDkPW,STkPW;
static double Jw1;    /* performance index for large
                         disturbance */
static double Jw2;    /* performance index for small
                         disturbance */
static int DD1,DD2,DXD1,DXD2;    /* GOV limit */
static int AVRL, AVRL1;          /* AVR limit */
static int SVCL, SVCL1;          /* SVC limit */
static int PSS,PSS1;             /* PSS limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double u1[][N],double t,int
          n,double dy2[N],double dy3[N],int flag1);
void sys1(double  x[],double  u1[][N],double  f[],double
        y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[][N],double Vt,double Pe,double
                x1);
void svcfuzw(double p1[][N],double Vt,double Pe,double
                x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
unsigned _stack = 50000;
FILE *fpw;

void main()
{
static double GD;
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
do{
fpw=fopen("f1svcsit.dat","w");
Ups=0;Upss=0.0;Wr=0.0;
Pd=0.0;ZZ=0.0;
SDEP=0.0;SDEPW=0.0;SEEP=0.0;SEEPW=0.0;SDkP=0.
0;SDkPW=0.0;STkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;D
XD2=0.0;SVCL=0.0;SVCL1=0.0;
PSS=PSS1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;Q0=
0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0][kpt] = 0.0;
  p1[0][kpt]=0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การผลิตซ้ำ ดัดแปลง เผยแพร่ หรือนำไปจัดแสดงในรูปแบบอื่นๆ และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;
/*--- Select Disturbance  Large , Small ---*/
clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("     <======= SVC+PSS3 Control
=======>\n\n");
printf("                    Enter to continue \n");
getch();select=1;clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf(" <===  Simulation Power System  ===>\n");
printf("     Please select Disturbance.\n");
printf("     large:[0],small:[1] --->");
scanf("     %d",&CH);
printf("\n");
if(CH==0)
{ printf("          Select trip time after 3-phase fault.\n");
  printf("                    [ 0 - 0.1s ] --->");
  scanf("  %f",&Time);
  (int)KH=Time/0.005;
}
if(select==1)
{ clrscr();
  printf("\n\n\n\n\n\n\n\n\n\n");
  printf("  <======= Turning  Parameter =======>\n");
  printf("                    alpha1 --->");
  scanf("%f",&sbp);
  printf("\n");
  printf("                    alpha2 --->");
  scanf("%f",&sap);
  printf("\n");
  printf("                    alpha3 --->");
  scanf("%f",&sDrp);
  printf("\n");
}
flag = 0;
xl = xt + xl1 + xl2 + xee;
sbpw=0.31;sapw=11;sDrpw=0.89;
g_init();
g_cls();
if(CH==0)
{ g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0,440.0, 105.0);
  w_axiss(0.0, 4.5, N * T, -4.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 4.5");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-4.5");

  g_view(40.0, 5.0,440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.3, N * T, -0.3);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.30");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.00");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.30");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0,365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
 }
 else
 { g_view(40.0, 5.0,440.0, 105.0,-1,-1);
  w_view(40.0, 5.0, 440.0, 105.0);
  w_axiss(0.0, 0.5, N * T, -0.5);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((110.0-0.0)*0.0/4.0+4.0), " 0.50");
  g_print(54, (int)((110.0-0.0)*2.0/4.0+2.0), " 0.00");
  g_print(29, (int)((110.0-0.0)*2.0/4.0+0.0), "ΔW");
  g_print(54, (int)((110.0-0.0)*4.0/4.0-2.0), "-0.50");

  g_view(40.0, 5.0, 440.0, 235.0,-1,-1);
  w_view(40.0, 135.0, 440.0, 235.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((250.0-140.0)*0.0/4.0+134.0), " 0.1");
  g_print(54, (int)((250.0-140.0)*2.0/4.0+132.0), " 0.0");
  g_print(29, (int)((250.0-140.0)*2.0/4.0+130.0), "ΔP");
  g_print(54, (int)((250.0-140.0)*4.0/4.0+128.0), "-0.1");

  g_view(40.0, 5.0, 440.0,365.0,-1,-1);
  w_view(40.0, 265.0, 440.0, 365.0);
  w_axiss(0.0, 0.1, N * T, -0.1);
  w_line(0.0, 0, 0);

  g_view(0, 0, 639, 399, -1, 1);
  g_print(54, (int)((380.0-270.0)*0.0/4.0+264.0), " 0.1");
  g_print(54, (int)((380.0-270.0)*2.0/4.0+262.0), " 0.0");
  g_print(29, (int)((380.0-270.0)*2.0/4.0+260.0), "ΔVt");
  g_print(54, (int)((380.0-270.0)*4.0/4.0+258.0), "-0.1");
 }
 initial_system();
 if(CH==1) Pd = 0.0;  /* small disturbance */
 else Pd = 0.0;
 x_stand(x1);

 for (kpt = 0;kpt <=N-1;kpt++)    /* start loop */
 { ti = (double)kpt * T;          /* current time */
 if(CH==0)                        /* large disturbance */
 if(kpt==0) flag = 1;   /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)                   /* fault line trip */
  { flag = 0;
    xl1 = 2.0*xl1;
  }
 if(CH==1)              /* small disturbance */
  if(kpt>=0)
   if (kpt < 100)
    Pd = -0.1/100*kpt;
 if(CH==1)
  if(kpt>=100)
   if (kpt < 200)
    Pd = -0.2 + 0.1 /100*kpt;
 if(CH==1)
  if(kpt>=200)
   Pd = 0.0;
 if(select==1)
  if(kpt%2==0)   /* sampling time for controller =0.01s */
   svcfuz(u1,x1[7],dy1[kpt],x1[9]);
  else u1[0][kpt] = u1[0][kpt-1];
 AVRL = AVRL1;
 DD1  = DXD1;
 DD2  = DXD2;
 SVCL = SVCL1;
 PSS  = PSS1;
 runge4(x1, u1, T, NA,dy2,dy3,flag);
 /* AVR Limite */
 if(x1[3] > Efmax - vf0)
 { AVRL1 = 1;
   ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 { AVRL1 = -1;
   ZZ = Efmin - vf0;
 }
 else
 { AVRL1 = 0;
   ZZ = x1[3];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีต่างๆ อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 }
 if(kpt%2==0)   /* sampling time for controller =0.01s */
  svcfuzw(p1,x1[7],dy1[kpt],Wr);
 else p1[0][kpt] = p1[0][kpt-1];
 Ups = p1[0][kpt];
 if(Ups>Umpssw)
 { PSS1=1;
   Upss = Umpssw ;
 }
 else
 if(Ups<-Umpssw)
 { PSS1=-1;
   Upss = -Umpssw;
 }
 else
 { PSS1=0;
   Upss = Ups;
 }
 /* GOV Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1) DXD1=1;
 else
 if(GD < -0.1) DXD1=-1;
 else DXD1=0;
 if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
 else
 if(x1[5] < -(P0-Pd)) DXD2=-1;
 else DXD2=0;
 /* SVC Limit */
 if (x1[10] > 1.0 - Bs0)
 { SVCL1 = 1;
   x1[10] = 1.0 - Bs0;
 }
 else
 if (x1[10] < -1.0 - Bs0)
 { SVCL1 = -1;
   x1[10] = -1.0 - Bs0;
 }
 else SVCL1 = 0;
 xc = 1.0 / (x1[10] + Bs0);
 sys1(x1, u1, f, y1,dy2,dy3,flag);
 d_out(y1);
 /* calulation performance index J */
 if(CH==0)
 { Jw1 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
 }
 if(CH==1)
 { Jw2 += fabs(x1[1])* ti+fabs((u1[0][kpt]-
          u1[0][kpt-1])/T);
 }
 /* Plot  Graph */
 dt1[1] = x1[1];
 dt2[1] = dy1[kpt+1];
 dt3[1] = dy2[kpt+1];
 if(kpt==0)fri =0;
 else fri=1;
 if(CH==0)
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,4.5,N*T,-4.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri,1);
   fprintf(fpw," %f %f %f %f %f %f %f
   \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);

   g_view(80.0,140.0, 480.0, 240.0,1,1);
   w_view(80.0,140.0, 480.0, 240.0);
   w_axis(0.0,0.3,N*T,-0.3);
   w_attr(3, 0xffff);
   w_line2((ti), dt2[1], fri, 2);

   g_view(80.0, 270.0, 480.0, 370.0,1,1);
   w_view(80.0, 270.0, 480.0, 370.0);
   w_axis(0.0,0.1,N*T,-0.1);
   w_attr(3, 0xffff);
   w_line2((ti), dt3[1], fri, 3);
 }
 else
 { g_view(80.0, 10.0, 480.0, 110.0,1,1);
   w_view(80.0, 10.0, 480.0, 110.0);
   w_axis(0.0,0.5,N*T,-0.5);
   w_attr(3, 0xffff);
   w_line2((ti), dt1[1], fri, 1);
   fprintf(fpw," %f %f %f %f %f %f %f
   \n",ti,dt1[1],dt2[1],dt3[1],dy3[kpt+1],u1[0][kpt],Upss);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจัดจำหน่าย จ่ายแจก หรือนำไปเผยแพร่ต่อ ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  g_view(80.0, 140.0, 480.0, 240.0,1,1);
  w_view(80.0, 140.0, 480.0, 240.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt2[1], fri, 2);

  g_view(80.0, 270.0, 480.0, 370.0,1,1);
  w_view(80.0, 270.0, 480.0, 370.0);
  w_axis(0.0,0.1,N*T,-0.1);
  w_attr(3, 0xffff);
  w_line2((ti), dt3[1], fri, 3);
}
dt1[0] = dt1[1];
dt2[0] = dt2[1];
dt3[0] = dt3[1];
}
getch();closegraph();clrscr();
printf("\n\n\n\n\n\n\n\n\n\n");
printf("                Continous ? (Quit press ESC)");
stop=getch();
fclose(fpw);
}
while(stop!=0x1b); /* ESC to exit */
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double A, C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{
dy1[kpt+1] = y1[0] - P0;
dy2[kpt+1] = y1[1] - Vt0;
dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[][N],double f[],double y
[],double dy2[N],double dy3[N],int flag1)
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1*xc/(xc-xl1);
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;

  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
  xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
switch(PSS)
{ case 1 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
        x[4])/Tf+Umpssw))/Te;
  break;
  case -1:
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
        Umpssw))/Te;
  break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีไฟล์งานเสีย อื่นๆ ต้องการนำไปใช้ต่อ ปรึกษา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  case 0 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
          x[4])/Tf+Upss))/Te;
  break;
 }
 /* 3: vf */
 /* 4: Rf */
 f[4] = (Kf * ZZ -x[4] ) / Tf;
 /* 5: g */
 switch(DD1)
 { case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
              W0;
  break;
 }
 /* 6: h */
 switch(DD2)
 { case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case  0 :f[6] = (-x[6]+x[5])/Tw;
  break;
 }
 /* 7: SVC : Vcm */
 f[7] = (-dy3[kpt] - x[7]) /Tv;
 /* 8: SVC : Pin          */
 f[8] = ((y[0] - (P0 -Pd )) - x[8]) / TP1;
 /* 9: SVC : Pout */
 f[9] = ((y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
 /* PSS */
 f[11] = (x[1]-x[11])/Tw;
 f[12] = (x[1]-x[11])/Tw - x[12]/Tr;
 Wr = x[12];
 /* 10:SVC : Bs */
 if(select==1)
 { switch(SVCL)
  { case  1 :f[10] = (-x[10]+Ksv *(x[7]+u1[0][kpt])) / Tc;
    break;
    case -1 :f[10] = (-x[10]+Ksv * (x[7]+u1[0][kpt])) / Tc;
    break;
    case  0 :f[10] = (-x[10]+Ksv * (x[7]+u1[0][kpt])) / Tc;
    break;
  }
 }
 }
 void runge4( double x[],double u1[][N],double t,int n,
              double dy2[N],double dy3[N],int flag1)
 /* Runge-Kutta4 for solving differntial equations */
 {
 double k1[NA], k2[NA], k3[NA], k4[NA], y[3],  z
 [NA];
 int i;
 sys1(x,u1,k1, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k1[i];
 sys1(z,u1,k2, y,dy2,dy3 ,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k2[i];
 sys1(z,u1,k3, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
  z[i] = x[i] + t * k3[i];
 sys1(z,u1,k4, y,dy2,dy3,flag1);
 for (i = 0; i < n; i++)
 x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
 }
 void svcsita(double DE,double EE,double Dk,double *Tk)
 /* fine angle on phase plane */
 {
 if(DE>0.0 && EE>0.0)
 *Tk= asin(DE/Dk)*180.0/PI;
 if(DE>0.0 && EE<0.0)
 *Tk= -asin(DE/Dk)*180.0/PI+180.0;
 if(DE<0.0 && EE<0.0)
 *Tk= -asin(DE/Dk)*180.0/PI+180.0;
 if(DE<0.0 && EE>0.0)
 *Tk= asin(DE/Dk)*180.0/PI+360.0;
 }
 void svcsimem(double Tk,double a,double *Nk,double
 SL)
 /* find membership function of phase plane angle */
 {
 if(Tk >= 0.0 && Tk < SL - a / 2.0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
void svcfuz(double u1[][N],double Vt,double Pe,double
x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0) SGkP=1.0;
/* calculate the supplementary control signal for SVC */
u1[0][kpt] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[0][kpt] < -Umfp) u1[0][kpt]=-Umfp;
if(u1[0][kpt] >  Umfp) u1[0][kpt]= Umfp;
}
void svcsitaw(double DEW,double EEW,double
Dkw,double *Tkw)
/* fine angle on phase plane */
{
if(DEW>0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI;
if(DEW>0.0 && EEW<0.0)
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW<0.0)
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI+360.0;
}
void svcsimemw(double Tkw,double aw,double
                *Nkw,double SL)
/* find membership function of phase plane angle */
{
if(Tkw >= 0.0 && Tkw < SL - aw / 2.0)
*Nkw = 1.0;
if(Tkw >= SL - aw/2.0 && Tkw <  SL + aw / 2.0)
*Nkw = ( - Tkw + SL ) / aw + 0.5;
if(Tkw >= SL + aw/2.0 && Tkw < SL +180.0 - aw/2.0)
*Nkw = 0.0;
if(Tkw >= SL + 180.0 - aw / 2.0 && Tkw <  SL + 180.0 +
aw / 2.0)
*Nkw = ( Tkw - ( SL + 180.0 ) ) / aw + 0.5;
if(Tkw >= SL + 180.0 + aw / 2.0  && Tkw <= 360.0)
*Nkw = 1.0;
}
void svcfuzw(double p1[][N],double Vt,double Pe,double
        x1) /* fuzzy control */
{
static double SNkPW,SGkPW;
SNkPW = 0.0;
SDEPW = sbpw*(x1-SEEPW)/(2.0*T);
SEEPW = x1;
SDkPW = sqrt(SDEPW*SDEPW + x1*x1);
/* Phase plan selection */
svcsitaw(SDEPW,x1,SDkPW,&STkPW);
svcsimemw(STkPW,sapw,&SNkPW,SL3);
/* determine Gain factor */
SGkPW = SDkPW/sDrpw;
if(SGkPW>1.0)
SGkPW=1.0;
/* calculate the supplementary control signal for AVR */
p1[0][kpt] = (2.0*SNkPW-1.0)*Umpssw*SGkPW;
if(p1[0][kpt] < -Umpssw) p1[0][kpt]=-Umpssw;
if(p1[0][kpt] >  Umpssw) p1[0][kpt]= Umpssw;
}
```

---

8. เฮดเดอร์ไฟล์ define.h

```
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#define Umfp    0.2   /*Max. control signal */
```

สารนิพนธ์นี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของสารนิพนธ์นี้ทุกครั้งที่มีการนำไปใช้

```
#define SL3     135.0 /* switching line */
#define Pmax    1.05 /*    Limit of GOV */
#define Efmax   7.30 /*  Limit of field voltage (max) */
#define Efmin   -7.30 /*  Limit of field voltage (min) */
#define Bsmax   1.0  /* Limit of SVC susceptance(max)*/
#define Bsmin   -1.0  /* Limit of SVC susceptance(min) */
#define Ue 200.0       /*Exciter gain */
#define Te 0.05        /*Exciter time constant */
#define Kf 0.07     /* Hunting parameter */
#define Tf 1.250    /* Time setting */
#define SIGM 0.05 /* Permanent drop */
#define Tg 0.20    /* Gate time conatant */
#define Tw 2.000   /* Water time constant */
#define P0 1.0       /* Real power */
#define V0 1.0       /* Reactive power */
#define Vt0 1.0     /* Terminal voltage */
#define W0 377.0 /* Synchronous speed */
#define Xdd   (( 1.0 - xe*xa ) * xdd + xe )
#define Xq    (( 1.0 - xe*xa ) * xq + xe )
#define Xd    (( 1.0 - xe*xa ) * xd + xe )
#define T   0.005       /* Sampling time */
#define N   1201        /* Number of data */
#define LIMIT  (T*(double)(N-1)) /*Max.num. data*/
#define NA   17         /*Number of differential equations
*/
#define Vc0 1.0         /* SVC */
#define Tc    0.05      /* SVC */
#define Ksv   50.0      /* SVC */
#define Tv    0.01      /* SVC*/
#define TQ1 4.0         /*reset circuit time constant */
#define  TP1 0.01       /*measuring time constant */
```

---

9. เฮดเดอร์ไฟล์ gtool.h

```
/* Last update : Sep. 4 ,1998
   Written by Komsan Hongesombut */
#include <dos.h>
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#define     coright       0
#define     cotop    0
#define     coleft        639
#define     cobottom  399
typedef unsigned  word;
int  w_vx1,w_vy1,w_vx2,w_vy2;
double  w_sx = 0.0;
double  w_sy = 0.0;
double  w_sy2= 0.0;
double  w_dx = 1.0;
double  w_dy = 1.0;
int  w_col = 7;
word w_pat = 0xffff;
int maxcolor;
#ifndef __BCPLUSPLUS__
void   w_view(double, double, double, double);
void   w_axis(double, double, double, double);
void   w_axiss(double, double, double, double);
/* Draw Graph */
void   w_attr(int, word);  /*  set attribute */
void   w_line(double, double, int);
void   w_line2(double, double, int, int);
void g_init (void);     /* initial graphics mode */
void g_print(int x,int y,char *st);
void g_screen(int x1,int x2,int x3,int x4);
/*initial attribute */
void g_cls(void);/* clear viewport */
void t_cls(void);/* clear text mode screen */
void g_line(int x1,int y1,int x2,int y2,int col,int
linestyle,int thickness,word pat);
void g_view(int x1,int x2,int x3,int x4,int x5,int x6);
/* set viewport area */
void w_attr22(int ,word );
void w_line222(double,double,int ,int );
#endif
void g_view(int x1,int x2,int x3,int x4,int x5,int x6)
{
if (x6 == -1) setviewport(x1,x2,x3+100,x4+10,x5);
else setviewport(x1,x2,x3,x4,x5);
}
void g_init (void)       /* initial grahics mode */
{ int driver = DETECT, mode, ErrorCode;
/* Set up screen in graphics mode, virtual coords */
```

สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการณ์ใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  initgraph (&driver, &mode, "");
  if ((ErrorCode = graphresult ()) != grOk)
  { clrscr ();
    gotoxy (5,12);
    printf ("%s",grapherrormsg(ErrorCode));
    delay (3000);
    exit (0);
  }
 maxcolor=getmaxcolor()+1;
}
void g_print (int x,int y,char *st)
{ int oldc;
  oldc = getcolor();
  setcolor(maxcolor-5);
  settextjustify (CENTER_TEXT,CENTER_TEXT);
  settextstyle (2,HORIZ_DIR,6);
  outtextxy (x,y,st);
  setcolor (oldc);
}
void g_cls(void)
{ cleardevice();
}
void g_line(int x1,int y1,int x2,int y2,int col,int
linestyle,int
          thickness,word pat)
{ int   oldc,a,b,c;
  struct linesettingstype *oldlineinfo=0;
  oldc=getcolor();
  setcolor(col);
  getlinesettings(oldlineinfo);
  setlinestyle(linestyle,pat,thickness);
  line(x1,y1,x2,y2);

setlinestyle((*oldlineinfo).linestyle,(*oldlineinfo).upattern
          ,(* oldlineinfo).thickness);
  setcolor(oldc);
}
void   w_view(x1, y1, x2, y2)
double  x1, y1, x2, y2;
{ w_vx1 = (int)x1;
  w_vy1 = (int)y1;
  w_vx2 = (int)x2;
  w_vy2 = (int)y2;
}
void   w_axiss(x1, y1, x2, y2)
double  x1, y1, x2, y2;
{ int  i1,  i2,  i3,  i4;
  i2 = 5;
  g_line(w_vx1, w_vy2, w_vx1, w_vy1, 9, 0, 0,w_pat);
  i1 = (int)((w_vy2+w_vy1)/2);
  g_line(w_vx1, i1, w_vx2, i1, 9, 0, 0,w_pat);
  i3 = w_vx1+i2;
  g_line(w_vx1, w_vy1, i3, w_vy1, 9, 0, 0,w_pat);
  g_line(w_vx1, w_vy2, i3, w_vy2, 9, 0, 0,w_pat);
  i4 = (int)((i1+w_vy1)/2);
  g_line(w_vx1, i4, i3, i4, 9, 0, 0,w_pat);
  i4 = (int)((i1+w_vy2)/2);
  g_line(w_vx1, i4, i3, i4, 9, 0, 0,w_pat);
  i4 = (int)((w_vx2+w_vx1)/2);
  g_line(i4, i1+3, i4, i1-3, 9, 0, 0,w_pat);
  g_line(w_vx2, i1+3, w_vx2, i1-3, 9, 0, 0,w_pat);
  g_print(i4-10, i1+10, "  3.0");
  g_view(0, 0, 639, 399, -1, -1);
  g_print(w_vx2-14, i1+10, "   6.0");
  g_print((int)((i4+w_vx2)/2)-45, i1+25, "     t [s]  ");
  g_view(w_vx1, w_vy1, w_vx2, w_vy2, -1, -1);
  w_sx = x1;
  w_sy = y1;
  w_dx = (x2 - x1) / (w_vx2 - w_vx1);
  w_dy = (y2 - y1) / (w_vy2 - w_vy1);
}
void   w_axis(x1, y1, x2, y2)
double  x1, y1, x2, y2;
{ w_sx = x1;
  w_sy = y1;
  w_dx = (x2 - x1) / (w_vx2 - w_vx1);
  w_dy = (y2 - y1) / (w_vy2 - w_vy1);
}
void   w_attr(col, pat)
int   col;
word pat;
{ w_col = col;
  w_pat = pat;
}
void  w_attr22(col, pat)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจำหน่าย จ่ายแจก และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  int col;
  word pat;
  { w_col = col;
    w_pat = pat;
  }
  void w_line(x, y, flag)
  double x, y;
  int flag;
  { int ix, iy;
    static int prex = 0;
    static int prey = 0;
    ix = (int)(w_vx1 + (x - w_sx) / w_dx);
    iy = (int)(w_vy1 + (y - w_sy) / w_dy);
    if (flag) g_line(prex, prey, ix, iy, w_col, 0, 1, w_pat);
    prex = ix;
    prey = iy;
  }
  void w_line2(xxx, yyy, flag ,no)
  double xxx, yyy;
  int flag ,no;
  { int ix, iy;
    static int prex[15];
    static int prey[15];
    ix = (int)( (xxx - w_sx) / w_dx);
    iy = (int)( (yyy - w_sy) / w_dy);
    if (flag) {g_line(prex[no], prey[no], ix, iy, w_col, 0,
          1,w_pat);}
    prex[no] = ix;
    prey[no] = iy;
  }
  void w_line222(xxx, yyy, flag ,no)
  double xxx, yyy;
  int flag ,no;
  { int ix, iy;
    static int prex[15];
    static int prey[15];
    ix = (int)( (xxx - w_sx) / w_dx);
    iy = (int)( (yyy - w_sy) / w_dy);
    if (flag) {g_line(prex[no], prey[no], ix, iy, w_col, 0,
          1,w_pat);}
    prex[no] = ix;
    prey[no] = iy;
  }
```

```
void t_cls(void)
{ clrscr();
}
void g_screen(int x1,int x2,int x3,int x4)
{ int screen_x,screen_y;
  screen_x= getmaxx();
  screen_y= getmaxy();
  if (!((screen_x>=600) && (screen_y>=400)))
  {outtextxy(10,10,"Can not use with this monitor");
   exit(0);
  }
/* test monitor is 640 X 380 More */
  setcolor(x1);
  setbkcolor(x2);
  settextstyle(x3,x3,x4);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ง.

1. โปรแกรมย่อย jsvc.c

```
/* jsvc.c : performance_index in case SVC
   Last update  Sep 5 , 1998
   Written by Komsan Hongesombut      */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\define.h"
#define PI M_PI
static float sbp,sap,sDrp; /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0I,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /* 1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
static double xd = 1.620,  /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600,  /* q-axis reactance */
              Td0d = 5.600,/* d-axis open circuit time
                              constant        */
              M = 0.0191,  /* Inertia constant */
              D = 0.00531; /* Damping coeffient */
static double u1[2],   /* Supplementary control signal for
SVC */
              dy1[N],    /* Real power Pe */
              dy2[N],    /* Terminal voltage Vt */
              dy3[N],    /* SVC  Vc */
              ti,        /* simulation time */
              Uee,P1,
              out11,out22;
static double Pd; /* small disturbance */
static double ZZ; /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP; /* fuzzy control */
static double Jw1; /* performance index J for large
                      disturbance */
static double Jw2; /* performance index J for small
                      disturbance */
static int DD1,DD2,DXD1,DXD2;    /* GOV limit */
static int AVRL, AVRL1;          /* AVR limit */
static int SVCL, SVCL1;          /* SVC limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double  x[],double u1[],double  t,int n,double
            dy2[N],double dy3[N],int flag1);
void sys1(double  x[],double u1[],double  f[],double
          y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[],double Vt,double Pe,double x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
double performance_index(double alpha1,double
                         alpha2,double alpha3);
unsigned _stack = 50000;
FILE *fpw;

double performance_index(double alpha1,double
                         alpha2,double alpha3)
/* calculate performance index value of power system
model both large and small disturbance                */
{
static double GD;
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
Uee=0;Pd=0.0;ZZ=0.0;SDEP=0.0;SEEP=0.0;SDkP=0.0;
STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0I=0.0;Q0t=0.0;
```

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าด้วยลิขสิทธิ์ และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0] = 0.0;
  u1[1] = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;
select=1;
Time=0.1;
(int)KH=Time/0.005;
sbp=alpha1;sap=alpha2;sDrp=alpha3;
for(CH=0;CH<=1;CH++){
Uee=0;Pd=0.0;ZZ=0.0;SDEP=0.0;SEEP=0.0;SDkP=0.0;
STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0]  = 0.0;
  u1[1]  = 0.0;
}
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;flag = 0;
xl = xt + xl1 + xl2 + xee;
initial_system();
if(CH==1) Pd = 0.0;   /* small disturbance */
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <=N-1;kpt++)     /* start loop */
{ ti = (double)kpt * T;           /* current time */
if(CH==0)                         /* large disturbance */
 if(kpt==0) flag = 1;   /* 3-phase fault */
```

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่าการดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(CH==0)
 if(kpt==0+KH)                    /* fault line trip */
 { flag = 0;
   xl1 = 2.0*xl1;
 }
if(CH==1)                         /* small disturbance */
 if(kpt>=0)
  if (kpt < 100)
   Pd = -0.1/100*kpt;
if(CH==1)
 if(kpt>=100)
  if (kpt < 200)
   Pd = -0.2 + 0.1 /100*kpt;
if(CH==1)
 if(kpt>=200)
  Pd = 0.0;
if(select==1)if(kpt%2==0)  /* sampling time for controller
                              = 0.01s */
 svcfuz(u1,x1[7],dy1[kpt],x1[9]);
else u1[1] = u1[0];
AVRL = AVRL1;
DD1  = DXD1;
DD2  = DXD2;
SVCL = SVCL1;
runge4(x1, u1, T, NA,dy2,dy3,flag);
/* AVR Limit */
if(x1[3] > Efmax - vf0)
{ AVRL1 = 1;
  ZZ = Efmax - vf0;
}
else
if(x1[3] < Efmin - vf0)
{ AVRL1 = -1;
  ZZ = Efmin - vf0;
}
else
{ AVRL1 = 0;
  ZZ = x1[3];
}
/* GOV Limit */
GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
if(GD > 0.1) DXD1=1;
else
if(GD < -0.1) DXD1=-1;
else DXD1=0;
if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
else
if(x1[5] < -(P0-Pd)) DXD2=-1;
else DXD2=0;
/* SVC Limit */
if (x1[10] > 1.0 - Bs0)
{ SVCL1 = 1;
  x1[10] = 1.0 - Bs0;
}
else
if (x1[10] < -1.0 - Bs0)
{ SVCL1 = -1;
  x1[10] = -1.0 - Bs0;
}
else SVCL1 = 0;
xc = 1.0 / (x1[10] + Bs0);
sys1(x1, u1, f, y1,dy2,dy3,flag);
d_out(y1);
/* calulation performance index J */
if(CH==0)
{ Jw1 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
if(CH==1)
{ Jw2 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
u1[0]= u1[1];
}
}
return(Jw1+0.1*Jw2);
}
void x_stand(double x[])
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power sytem model at normal operation */
{
static double A, C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/*  6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/*  9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* outpt deviation */
{  dy1[kpt+1] = y1[0] - P0;
   dy2[kpt+1] = y1[1] - Vt0;
   dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[],double f[],double
        y[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations of power system model */
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{  A1 = xl1*xc/(xc-xl1);
   A2 = xl2+xee;
   B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
   B2 = xl1*A1/(3.0*xl1+A1);
   B3 = 2.0*xl1*A1/(3.0*xl1+A1);
   C1 = xt+B1;
   C2 = A2+B3;
   xe = (C1*C2+B2*(C1+C2))/B2;
   xa = (C1*C2+B2*(C1+C2))/C2;
   xa = -1.0/xa;
   xb = (C1*C2+B2*(C1+C2))/C1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การดัดแปลงส่วนใดส่วนหนึ่ง หรือทั้งหมดให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
if(flag1==0)
{ xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
  xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f       */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf))/Te;
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
                W0;
  break;
}
/* 6: h */
switch(DD2)
{ case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case  0 :f[6] = (-x[6]+x[5])/Tw;
  break;
}
/* 7: SVC : Vcm */
f[7] = (-dy3[kpt] - x[7]) /Tv;
/* 8: SVC : Pin          */
f[8] = ((y[0] - (P0 -Pd )) - x[8]) / TP1;
/* 9: SVC : Pout */
f[9] = ((y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
/* 10:SVC : Bs */
if(select==1)
{ switch(SVCL)
  { case  1 :f[10] = (-x[10]+Ksv * (x[7]+Umfp)) / Tc;
    break;
    case -1 :f[10] = (-x[10]+Ksv * (x[7]-Umfp)) / Tc;
    break;
    case  0 :f[10] = (-x[10]+Ksv * (x[7]+u1[1])) / Tc;
    break;
  }
}
}
void runge4(double x[],double u1[],double t,int n,
            double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differential equations */
{
double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
int i;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sys1(x, u1, k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k1[i];
sys1(z, u1, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k2[i];
sys1(z, u1, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t * k3[i];
sys1(z, u1, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcfuz(double u1[],double Vt,double Pe,double x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0) SGkP=1.0;
/* calculate the SVC supplementary control signal */
u1[1] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[1] < -Umfp) u1[1] =-Umfp;
if(u1[1] >  Umfp) u1[1] = Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI+360.0;
}
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

---

2. โปรแกรมย่อย jsvcpss1.c

```
/* jsvcpss1.c : performance_index in case SVC+PSS1
   Last update Sep 5 ,1998
   Written by Komsan Hongesombut                    */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\define.h"
#define Umpss   0.1
#define Kpss    1
#define T1      0.04
#define T2      0.05
#define PI      M_PI
static float sbp,sap,sDrp;  /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14,  /* Xt */
              xl1 = 0.21,  /* 1/2 per cct */
              xl2 = 0.30,  /* 1/2 per cct */
              xee = 0.05;  /* Xj */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
static double xd = 1.620,  /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600,  /* q-axis reactance */
             Td0d = 5.600, /* d-axis open circuit time
                              constant */
              M = 0.0191,  /* Inertia constant */
              D = 0.00531; /* Damping coeffient */
static double u1[2],    /* supplementary control signal for
                           SVC */
              dy1[N],    /* Real power Pe */
              dy2[N],    /* Terminal voltage Vt */
              dy3[N],    /* SVC Vc */
              ti,        /* simulation time */
              Ups,Upss,
              P1,out,out22;
static double Pd; /* small disturbance */
static double ZZ; /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */
static double Jw1;     /* performance index for large
                          disturbance */
static double Jw2;     /* performance index for small
                          disturbabce */
static int DD1,DD2,DXD1,DXD2;  /* GOV limit */
static int AVRL, AVRL1;        /* AVR limit */
static int SVCL, SVCL1;        /* SVC limit */
static int PSS,PSS1;           /* PSS limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double u1[],double  t,int n,double
            dy2[N],double dy3[N],int flag1);
void sys1(double x[],double u1[],double  f[],double
          y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[],double Vt,double Pe,double x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
double performance_index(double alpha1,double
                         alpha2,double alpha3);
unsigned _stack = 50000;
FILE *fpw;
```

```
double performance_index(double alpha1,double
                         alpha2,double alpha3)
/* calculate performance index of power system model
  both large and small disturbance                  */
{
static double GD;
static double x1[NA], y1[4], f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
Jw1=Jw2=0.0;
flag=0;fri=0;kpt=0.0;ti=0.0;
select=1; /* for fuzzy control type */
Time=0.1;
(int)KH=Time/0.005;
sbp=alpha1;sap=alpha2;sDrp=alpha3;
for(CH=0;CH<=1;CH++){
/*--- Select Disturbance  Large , Small ---*/
Ups=0;Upss=0;Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0]  = 0.0;
  u1[1]  = 0.0;
}
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;flag = 0;
xl = xt + xl1 + xl2 + xee;
initial_system();
if(CH==1) Pd = 0.0;    /* small disturbance */
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <=N-1;kpt++)     /* start loop */
{ ti = (double)kpt * T;           /* current time */
if(CH==0)                         /* large disturbance */
if(kpt==0) flag = 1;   /* 3-phase fault */
if(CH==0)
 if(kpt==0+KH)                    /* fault line trip */
 { flag = 0;
   xl1 = 2.0*xl1;
 }
if(CH==1)                 /* small disturbance */
 if(kpt>=0)
  if (kpt < 100)
   Pd = -0.1/100*kpt;
if(CH==1)
 if(kpt>=100)
  if (kpt < 200)
   Pd = -0.2 + 0.1 /100*kpt;
if(CH==1)
 if(kpt>=200)
  Pd = 0.0;
if(select==1)
 if(kpt%2==0) /* sampling time for controller =0.01s */
   svcfuz(u1,x1[7],dy1[kpt],x1[15]);
  else u1[1] = u1[0];
 AVRL = AVRL1;
 DD1  = DXD1;
 DD2  = DXD2;
 SVCL = SVCL1;
 PSS  = PSS1;
 runge4(x1, u1, T, NA,dy2,dy3,flag);
 /* AVR Limit */
 if(x1[3] > Efmax - vf0)
 { AVRL1 = 1;
   ZZ = Efmax - vf0;
 }
 else
 if(x1[3] < Efmin - vf0)
 { AVRL1 = -1;
   ZZ = Efmin - vf0;
 }
 else
 { AVRL1 = 0;
   ZZ = x1[3];
 }
 Ups = Kpss*out;
 if(Ups>Umpss) Ups = Umpss;
 if(Ups<-Umpss) Ups = -Umpss;
 if(Ups>Umpss)
 { PSS1 = 1;
   Upss = Umpss;
 }
 else
 if (Ups<-Umpss)
 { PSS1 = -1;
   Upss = - Umpss;
 }
 else
 { PSS1 = 0.0;
   Upss = Ups;
 }
 /* GOV Limit */
 GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
 if(GD > 0.1) DXD1=1;
 else
 if(GD < -0.1) DXD1=-1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
else DXD1=0;
if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
else
if(x1[5] < -(P0-Pd)) DXD2=-1;
else DXD2=0;
/* SVC Limit */
if (x1[13] > 1.0 - Bs0)
{ SVCL1 = 1;
  x1[13] = 1.0 - Bs0;
}
else
if (x1[10] < -1.0 - Bs0)
{ SVCL1 = -1;
  x1[13] = -1.0 - Bs0;
}
else SVCL1 = 0;
xc = 1.0 / (x1[13] + Bs0);
sys1(x1, u1, f, y1,dy2,dy3,flag);
d_out(y1);
/* calulation performance index J */
if(CH==0)
{ Jw1 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
if(CH==1)
{ Jw2 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
u1[0] = u1[1];
}
}
return(Jw1+0.1*Jw2);
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double A,C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การจัดจำหน่าย จ่ายแจก ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{ dy1[kpt+1] = y1[0] - P0;
  dy2[kpt+1] = y1[1] - Vt0;
  dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[],double f[],double
        y[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
double    vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1*xc/(xc-xl1);
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;
  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
  xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
switch(PSS)
{ case 1 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        x[4])/Tf+Umpss))/Te;
    break;
    case -1:
    f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
        Umpss))/Te;
    break;
    case 0 :
    f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
        x[4])/Tf+Upss))/Te;
    break;
  }
  /* 4: Rf */
  f[4] = (Kf * ZZ -x[4] ) / Tf;
  /* 5: g */
  switch(DD1)
  { case 1 :f[5] =  0.1;
    break;
    case -1:f[5] = -0.1;
    break;
    case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
                W0;
    break;
  }
  /* 6: h */
  switch(DD2)
  { case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
    break;
    case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
    break;
    case  0 :f[6] = (-x[6]+x[5])/Tw;
    break;
  }
  /* 7: SVC : Vcm */
  f[7] = (-dy3[kpt] - x[7]) /Tv;
  /* 8: SVC : Pin          */
  f[8] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1;
  /* 9: SVC : Pout */
  f[9] = (-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
  f[10] =  (x[9]-x[10])/T2;
  f[11] = (T1*((-(y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1)-x
  [11])/T2;
  x[12] =x[10]+x[11];
  out = x[12];
  f[14] = ((y[0]-(P0-Pd))-x[14])/TP1;
  f[15] = ((y[0]-(P0-Pd))-x[14])/TP1-x[15]/TQ1;
  /* 10:SVC : Bs */
  if(select==1)
  { switch(SVCL)
    { case  1 :f[13] = (-x[13]+Ksv * (x[7]+u1[1])) / Tc;
      break;
      case -1 :f[13] = (-x[13]+Ksv * (x[7]+u1[1])) / Tc;
      break;
      case  0 :f[13] = (-x[13]+Ksv * (x[7]+u1[1])) / Tc;
      break;
    }
  }
  }
  void runge4(double x[],double u1[],double t,int n,
              double dy2[N],double dy3[N],int flag1)
  /* Runge-Kutta4 for solving differential equations */
  {
  double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
  int i;
  sys1(x, u1, k1, y,dy2,dy3,flag1);
  for (i = 0; i < n; i++)
   z[i] = x[i] + t / 2 * k1[i];
  sys1(z, u1, k2, y,dy2,dy3 ,flag1);
  for (i = 0; i < n; i++)
   z[i] = x[i] + t / 2 * k2[i];
  sys1(z, u1, k3, y,dy2,dy3,flag1);
  for (i = 0; i < n; i++)
   z[i] = x[i] + t * k3[i];
  sys1(z, u1, k4, y,dy2,dy3,flag1);
  for (i = 0; i < n; i++)
  x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
  }
  void svcfuz(double u1[],double Vt,double Pe,double x1)
  /* fuzzy control */
  {
  static double SNkP,SGkP;
  SNkP = 0.0;
  SDEP = sbp*(x1-SEEP)/(2.0*T);
  SEEP = x1;
  SDkP = sqrt(SDEP*SDEP + x1*x1);
  /* Phase plan selection          */
  svcsita(SDEP,x1,SDkP,&STkP);
```

สารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0)
SGkP=1.0;
/* calculate the SVC supplementary control signal */
u1[1] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[1] < -Umfp) u1[1] =-Umfp;
if(u1[1] >  Umfp) u1[1] = Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI+360.0;
}
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

3. โปรแกรมย่อย jsvcpss2.c

```
/* jsvcpss2.c : performance_index in case SVC+PSS2
   Last update Sep 5 ,1998
   Written by Komsan Hongesombut                */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\define.h"
#define Umpss   0.1
#define Tww     0.056
#define Trr     4.0
#define T1      0.3
#define T2      0.02
#define T3      0.2
#define T4      0.02
#define Kwpss   13.0
#define PI      M_PI
static float sbp,sap,sDrp; /* alpha1,alpha2,alpha3 */
static double vd0,vq0,id0,iq0,del0,del1,
              Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
static double xc,xe,xa,xb,xl,
              xt  = 0.14, /* Xt */
              xl1 = 0.21, /* 1/2 per cct */
              xl2 = 0.30, /* 1/2 per cct */
              xee = 0.05; /* Xj */
static double xd = 1.620, /* d-axis reactance */
              xdd = 0.250, /* d-axis transient reactance */
              xq = 1.600, /* q-axis reactance */
              Td0d = 5.600, /* d-axis open circuit time
                               constant */
              M = 0.0191,  /* Inertia constant          */
              D = 0.00531; /* Damping coeffient */
static double u1[2],   /* supplementary control signal for
                          SVC */
              dy1[N],    /* Real power Pe */
              dy2[N],    /* Terminal voltage Vt */
              dy3[N],    /* SVC Vc */
              ti,        /* simulation time */
              Ups,Upss,
              P1,out,out1;
static double Pd; /* small disturbance */
static double ZZ; /* used in AVR limit */
static double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
static double Jw1;      /* performance index for large
                           disturbance */
static double Jw2;      /* performance index for small
                           disturbabce */
static int DD1,DD2,DXD1,DXD2;   /* GOV limit */
static int AVRL, AVRL1;         /* AVR limit */
static int SVCL, SVCL1;                 /* SVC limit */
static int PSS,PSS1;            /* PSS limit */
static int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double u1[],double  t,int n,double
     dy2[N],double dy3[N],int flag1);
void sys1(double x[],double u1[],double  f[],double
     y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[],double Vt,double Pe,double x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
double performance_index(double alpha1,double
                         alpha2,double alpha3);
unsigned _stack = 50000;
FILE *fpw;

double performance_index(double alpha1,double
                         alpha2,double alpha3)
/*  calculate performance index of power system model
    both large and small disturbance                  */
{
static double GD;
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
Ups=0;Upss=0;Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;
del0=0.0;del1=0.0;Q0=0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{  dy1[kpt] = 0.0;
   dy2[kpt] = 0.0;
   dy3[kpt] = 0.0;
   u1[0]    = 0.0;
   u1[1]    = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{  x1[kpt] = 0.0;
   f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{  dt1[kpt] = 0.0;
   dt2[kpt] = 0.0;
   dt3[kpt] = 0.0;
   dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;select=1;
Time=0.1;
(int)KH=Time/0.005;
sbp=alpha1;sap=alpha2;sDrp=alpha3;
for(CH=0;CH<=1;CH++){
Ups=0;Upss=0;
Pd=0.0;ZZ=0.0;
SDEP=0.0;SEEP=0.0;SDkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
PSS=PSS1 = 0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0]  = 0.0;
  u1[1]  = 0.0;
}
xt  = 0.14;xl1 = 0.21;xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD=0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;flag = 0;
xl = xt + xl1 + xl2 + xee;
initial_system();
if(CH==1) Pd = 0.0;    /* small disturbance */
else Pd = 0.0;
x_stand(x1);
for (kpt = 0;kpt <=N-1;kpt++)   /* start loop */
{ ti = (double)kpt * T;          /* current time */
if(CH==0)                        /* large disturbance */
 if(kpt==0) flag = 1;   /* 3-phase fault */
if(CH==0)
 if(kpt==0+KH)                   /* fault line trip */
 { flag = 0;
   xl1 = 2.0*xl1;
 }
if(CH==1)                        /* small disturbance */
 if(kpt>=0)
  if (kpt < 100)
    Pd = -0.1/100*kpt;
if(CH==1)
 if(kpt>=100)
  if (kpt < 200)
    Pd = -0.2 + 0.1 /100*kpt;
if(CH==1)
 if(kpt>=200)
  Pd = 0.0;
if(select==1)if(kpt%2==0)  /* sampling time for controller
                              = 0.01s */
 svcfuz(u1,x1[7],dy1[kpt],x1[15]);
else u1[1] = u1[0];
AVRL = AVRL1;
DD1  = DXD1;
DD2  = DXD2;
SVCL = SVCL1;
PSS  = PSS1;
runge4(x1, u1, T, NA,dy2,dy3,flag);
/* AVR Limit */
if(x1[3] > Efmax - vf0)
{ AVRL1 = 1;
  ZZ = Efmax - vf0;
}
else
if(x1[3] < Efmin - vf0)
{ AVRL1 = -1;
  ZZ = Efmin - vf0;
}
else
{ AVRL1 = 0;
  ZZ = x1[3];
}
Ups = out;
if(Ups>Umpss) Ups = Umpss;
if(Ups<-Umpss) Ups = -Umpss;
if(Ups>Umpss)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{ PSS1 = 1;
  Upss = Umpss;
}
else
if (Ups<-Umpss)
{ PSS1 = -1;
  Upss = - Umpss;
}
else
{ PSS1 = 0.0;
  Upss = Ups;
}
/* GOV Limit */
GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
if(GD > 0.1) DXD1=1;
else
if(GD < -0.1) DXD1=-1;
else DXD1=0;
if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
else
if(x1[5] < -(P0-Pd)) DXD2=-1;
else DXD2=0;
/* SVC Limit */
if (x1[16] > 1.0 - Bs0)
{ SVCL1 = 1;
  x1[16] = 1.0 - Bs0;
}
else
if (x1[16] < -1.0 - Bs0)
{ SVCL1 = -1;
  x1[16] = -1.0 - Bs0;
}
else SVCL1 = 0;
xc = 1.0 / (x1[16] + Bs0);
sys1(x1, u1, f, y1,dy2,dy3,flag);
d_out(y1);
/* calulation performance index J */
if(CH==0)
{ Jw1 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
if(CH==1)
{ Jw2 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
}
u1[0] = u1[1];
}
}
return(Jw1+0.1*Jw2);
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double A,C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
    +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{ dy1[kpt+1] = y1[0] - P0;
  dy2[kpt+1] = y1[1] - Vt0;
  dy3[kpt+1] = y1[3] - vc0;
}
void sys1(double x[],double u1[],double f[],double
        y[],double dy2[N],double dy3[N],int flag1)
{
double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1*xc/(xc-xl1);
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;

  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
  xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
    +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  f[0] = x[1];
  /* 1: w */
  f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
  /* 2: f       */
  switch(AVRL)
  { case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
    break;
    case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
    break;
    case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
    break;
  }
  /* 3: vf */
  switch(PSS)
  { case 1 :
    f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
          x[4])/Tf+Umpss))/Te;
    break;
    case -1:
    f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
          Umpss))/Te;
    break;
    case 0 :
    f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
          x[4])/Tf+Upss))/Te;
    break;
  }
  /* 4: Rf */
  f[4] = (Kf * ZZ -x[4] ) / Tf;
  /* 5: g */
  switch(DD1)
  { case 1 :f[5] =  0.1;
    break;
    case -1:f[5] = -0.1;
    break;
    case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] /
                W0;
    break;
  }
  /* 6: h */
  switch(DD2)
  { case  1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
    break;
    case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
    break;
    case  0 :f[6] = (-x[6]+x[5])/Tw;
    break;
  }
  /* 7: SVC : Vcm */
  f[7] = (-dy3[kpt] - x[7]) /Tv;
  /* 8: SVC : Pin          */
  f[14] = ((y[0] - (P0 -Pd )) - x[14]) / TP1;
  /* 9: SVC : Pout */
  f[15] = ((y[0] - (P0 -Pd )) - x[14]) / TP1 -x[15]/ TQ1;
  /* PSS */
  f[8] = (x[1]-x[8])/Tww;
  f[9] = (Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-x[9])/Trr;
  f[10] = (x[9]-x[10])/T2;
  f[11] = (T1*(Trr*(Kwpss/W0)*(x[1]-x[8])/Tww-x[9])/Trr-
  x[11])/T2;
  out1 = x[10]+x[11];
  f[12] = (out1-x[12])/T4;
  f[13]=(T3*((x[9]-x[10])/T2+(T1*(Trr*(Kwpss/W0)*(x[1]-
  x[8])/Tww-x[9])/Trr-x[11])/T2)-x[13])/T4;
  out = (x[12]+x[13]);
  /* 10:SVC : Bs */
  if(select==1)
  { switch(SVCL)
    { case  1 :f[16] = (-x[16]+Ksv * (x[7]+Umfp)) / Tc;
      break;
      case -1 :f[16] = (-x[16]+Ksv * (x[7]-Umfp)) / Tc;
      break;
      case  0 :f[16] = (-x[16]+Ksv * (x[7]+u1[1])) / Tc;
      break;
    }
  }
  }
  void runge4(double x[],double u1[],double t,int n,
              double dy2[N],double dy3[N],int flag1)
  /* Runge-Kutta4 for solving differntial equations */
  {
  double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
  int i;
  sys1(x, u1, k1, y,dy2,dy3,flag1);
  for (i = 0; i < n; i++)
   z[i] = x[i] + t / 2 * k1[i];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sys1(z, u1, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t / 2 * k2[i];
sys1(z, u1, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
  z[i] = x[i] + t * k3[i];
sys1(z, u1, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcfuz(double u1[],double Vt,double Pe,double x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection          */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0)
SGkP=1.0;
/* calculate the SVC supplementary control signal  */
u1[1] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[1] < -Umfp) u1[1] =-Umfp;
if(u1[1] >  Umfp) u1[1] = Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/M_PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/D:)*180.0/M_PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/M_PI+360.0;
}
```

```
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
```

---

4. โปรแกรมย่อย jsvcpss3.c

```
/* jsvcpss3.c : performance_index in case SVC+PSS3
   Last update Sep 5 ,1998
   Written by Komsan Hongesombut                */
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include"c:\define.h"
#define Umpssw  0.1
#define Tw      0.056
#define Tr      4.0
#define PI      M_PI
float sbp,sap,sDrp;  /* alpha1,alpha2,alpha3 */
float sbpw,sapw,sDrpw; /* alphaw1,alphaw2,alphaw3 */
double vd0,vq0,id0,iq0,del0,del1,
       Q0,Q0l,Q0t,fif0,vf0,vc0,v0,Bs0;
double xc,xe,xa,xb,xl,
       xt  = 0.14,  /* Xt */
       xl1 = 0.21,  /* 1/2 per cct */
       xl2 = 0.30,  /* 1/2 per cct */
       xee = 0.05;  /* Xj */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารนี้ทุกครั้งที่มีการนำไปใช้

```
double xd = 1.620,  /* d-axis reactance */
       xdd = 0.250, /* d-axis transient reactance */
       xq = 1.600,  /* q-axis reactance */
       Td0d = 5.600,/* d-axis open circuit time constant */
       M = 0.0191,  /* Inertia constant */
       D = 0.00531; /* Damping coeffient */
double u1[2],          /* supplementary control signal for
                          SVC */
       p1[2],  /* supplementary control signal for PSS */
       dy1[N],         /* Real power Pe */
       dy2[N],         /* Terminal voltage Vt */
       dy3[N],         /* SVC Vc */
       ti,             /* simulation time */
          Uee,P1,
          Wr,Ups,Upss,out22;
double Pd;  /* small disturbance */
double ZZ;  /* used in AVR limit */
double SDEP,SEEP,SDkP,STkP;  /* fuzzy control */


double SDEPW,SEEPW,SDkPW,STkPW;
double Jw1;  /* performance index  for large disturbance
*/
double Jw2;  /* performance index  for small disturbance
*/
int DD1,DD2,DXD1,DXD2;    /* GOV limit */
int AVRL, AVRL1;          /* AVR limit */
int SVCL, SVCL1;          /* SVC limit */
int PSS,PSS1;
int kpt,CH,KH,select;
void initial_system(void);
void x_stand(double x[]);
void d_out(double y1[]);
void runge4(double x[],double u1[],double t,int n,double
          dy2[N],double dy3[N],int flag1);
void sys1(double x[],double  u1[],double  f[],double
        y[],double dy2[N],double dy3[N],int flag1);
void svcfuz(double u1[],double Vt,double Pe,double x1);
void svcfuzw(double p1[],double Vt,double Pe,double x1);
void svcsita(double DE,double EE,double Dk,double
*Tk);
void svcsimem(double Tk,double a,double *Nk,double
SL);
```

```
double performance_index(double alpha1,double
                         alpha2,double alpha3);
unsigned _stack = 50000;
FILE *fp;
double performance_index(double alpha1,double
                         alpha2,double alpha3)
/* calculate performance index of power system model
  both large and small disturbance                  */
{
static double GD;
static double x1[NA],y1[4],f[NA];
static double dt1[2],dt2[2],dt3[2],dt4[2];
static int flag,fri;
int stop;
float Time;
Ups=0;Upss=0.0;Wr=0.0;
Pd=0.0;ZZ=0.0;
SDEP=0.0;SDEPW=0.0;SEEP=0.0;SEEPW=0.0;
SDkP=0.0;SDkPW=0.0;STkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;
DD1=0.0;DD2=0.0;DXD1=0.0;DXD2=0.0;SVCL=0.0;
SVCL1=0.0;
PSS=PSS1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;
fif0=0.0;vf0=0.0;vc0=0.0;v0=0.0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
 dy2[kpt] = 0.0;
 dy3[kpt] = 0.0;
 u1[0]   = 0.0;
 u1[1]   = 0.0;
 p1[0]   = 0.0;
 p1[1]   = 0.0;
}
Jw1=Jw2=0.0;
xt  = 0.14;
xl1 = 0.21;
xl2 = 0.30;
xee = 0.05;
xd = 1.620;   /* d-axis reactance */
xdd = 0.250;  /* d-axis transient reactance */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
xq = 1.600;   /* q-axis reactance */
Td0d = 5.600; /* d-axis open circuit time constant */
M = 0.0191;   /* Inertia constant */
D = 0.00531;  /* Damping coefficient */
GD = 0.0;
for (kpt = 0; kpt < NA; kpt++)
{ x1[kpt] = 0.0;
  f[kpt] = 0.0;
}
for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
for (kpt = 0; kpt < 2; kpt++)
{ dt1[kpt] = 0.0;
  dt2[kpt] = 0.0;
  dt3[kpt] = 0.0;
  dt4[kpt] = 0.0;
}
flag=0;fri=0;kpt=0.0;ti=0.0;select = 1;
if(select==1)
{ sbp = alpha1;
  sap = alpha2;
  sDrp = alpha3;
  sbpw=0.31;
  sapw=11.0;
  sDrpw=0.89;
}
for(CH=0;CH<=1;CH++) {
Ups=0;Upss=0.0;Wr=0.0;
Pd=0.0;ZZ=0.0;
SDEP=0.0;SDEPW=0.0;SEEP=0.0;SEEPW=0.0;
SDkP=0.0;SDkPW=0.0;STkP=0.0;STkP=0.0;
AVRL=0.0;AVRL1=0.0;DD1=0.0;DD2=0.0;DXD1=0.0;
DXD2=0.0;SVCL=0.0;SVCL1=0.0;
PSS=PSS1=0.0;
vd0=0.0;vq0=0.0;id0=0.0;iq0=0.0;del0=0.0;del1=0.0;
Q0=0.0;Q0l=0.0;Q0t=0.0;fif0=0.0;vf0=0.0;vc0=0.0;v0=0.
0;Bs0=0.0;
xc=0.0;xe=0.0;xa=0.0;xb=0.0;xl=0.0;
for (kpt = 0; kpt < N; kpt++)
{ dy1[kpt] = 0.0;
  dy2[kpt] = 0.0;
  dy3[kpt] = 0.0;
  u1[0]  = 0.0;
  u1[1]  = 0.0;
  p1[0]  = 0.0;
  p1[1]  = 0.0;
 }
 xt  = 0.14;
 xl1 = 0.21;
 xl2 = 0.30;
 xee = 0.05;
 xd = 1.620;   /* d-axis reactance */
 xdd = 0.250;  /* d-axis transient reactance */
 xq = 1.600;   /* q-axis reactance */
 Td0d = 5.600; /* d-axis open circuit time constant */
 M = 0.0191;   /* Inertia constant */
 D = 0.00531;
 GD = 0.0;
 for (kpt = 0; kpt < NA; kpt++)
 { x1[kpt] = 0.0;
   f[kpt] = 0.0;
 }
 for (kpt = 0; kpt < 4; kpt++) y1[kpt] = 0.0;
 for (kpt = 0; kpt < 2; kpt++)
 { dt1[kpt] = 0.0;
   dt2[kpt] = 0.0;
   dt3[kpt] = 0.0;
   dt4[kpt] = 0.0;
 }
 flag=0;fri=0;kpt=0.0;ti=0.0;
 if(CH==0)
 { Time = 0.1;
   (int)KH=Time/0.005;
 }
 flag = 0;
 xl = xt + xl1 + xl2 + xee;
 initial_system();
 if(CH==1) Pd = 0.0;  /* small disturbance */
 else Pd = 0.0;
 x_stand(x1);
 for (kpt = 0;kpt <N-1;kpt++)      /* start loop */
 { ti = (double)kpt * T;           /* current time */
 if(CH==0)                         /* large
disturbance */
  if(kpt==0) flag = 1;   /* 3-phase fault */
 if(CH==0)
  if(kpt==0+KH)                    /* fault line trip */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    { flag = 0;
      xl1 = 2.0*xl1;
    }
  if(CH==1)                    /* small disturbance */
   if(kpt>=0)
    if (kpt < 100)
     Pd = -0.1/100*kpt;
  if(CH==1)
   if(kpt>=100)
    if (kpt < 200)
     Pd = -0.2 + 0.1 /100*kpt;
  if(CH==1)
   if(kpt>=200)
    Pd = 0.0;
  if(select==1)if(kpt%2==0)  /* sampling time for controller
  = 0.01s */
   svcfuz(u1,x1[7],dy1[kpt],x1[9]);
  else u1[1] = u1[0];
  AVRL = AVRL1;
  DD1  = DXD1;
  DD2  = DXD2;
  SVCL = SVCL1;
  PSS  = PSS1;
  runge4(x1, u1, T, NA,dy2,dy3,flag);
  /* AVR Limit */
  if(x1[3] > Efmax - vf0)
  { AVRL1 = 1;
    ZZ = Efmax - vf0;
  }
  else
  if(x1[3] < Efmin - vf0)
  { AVRL1 = -1;
    ZZ = Efmin - vf0;
  }
  else
  { AVRL1 = 0;
    ZZ = x1[3];
  }
  if(kpt%2==0)  /* sampling time for controller = 0.01s */
   svcfuzw(p1,x1[7],dy1[kpt],Wr);
  else p1[1] = p1[0];
  Ups = p1[1];
  if(Ups>Umpssw)
    { PSS1=1;
      Upss = Umpssw ;
    }
    else
    if(Ups<-Umpssw)
    { PSS1=-1;
      Upss = -Umpssw;
    }
    else
    { PSS1=0;
      Upss = Ups;
    }
    /* GOV Limit */
    GD=(-x1[1]/W0 - SIGM*x1[5]) /Tg;
    if(GD > 0.1) DXD1=1;
    else
    if(GD < -0.1) DXD1=-1;
    else DXD1=0;
    if(x1[5] > (Pmax-(P0-Pd))) DXD2=1;
    else
    if(x1[5] < -(P0-Pd)) DXD2=-1;
    else DXD2=0;
    /* SVC Limit */
    if (x1[10] > 1.0 - Bs0)
    { SVCL1 = 1;
      x1[10] = 1.0 - Bs0;
    }
    else
    if (x1[10] < -1.0 - Bs0)
    { SVCL1 = -1;
      x1[10] = -1.0 - Bs0;
    }
    else SVCL1 = 0;
    xc = 1.0 / (x1[10] + Bs0);
    sys1(x1, u1, f, y1,dy2,dy3,flag);
    d_out(y1);
    /* calulation performance index J */
    if(CH==0)
    { Jw1 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
    }
    if(CH==1)
    { Jw2 += fabs(x1[1])* ti+fabs((u1[1]-u1[0])/T);
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
u1[0] = u1[1];
p1[0] = p1[1];
}
}
return(Jw1+0.1*Jw2);
}
void x_stand(double x[])
/* initial value of variables */
{
x[0] = del0;
x[1] = 0.0;
x[2] = fif0;
x[3] = 0.0;
x[4] = 0.0;
x[5] = 0.0;
x[6] = 0.0;
x[7] = 0.0;
x[8] = 0.0;
x[9] = 0.0;
x[10] = 0.0;
x[11]= 0.0;
x[12]= 0.0;
x[13]= 0.0;
x[14]= 0.0;
x[15]= 0.0;
x[16]= 0.0;
}
void initial_system()
/* initial power system model at normal operation */
{
static double A,C;
/* 1.delta1 */
del1 = asin(P0*xl/2.0);
/* 2. xc */
xc = xl/(4.0*(1.0 - cos(del1)));
Bs0 = 1.0/xc;
/* 3.xe,xa,xb */
xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/(xt+xl1);
xa = 1.0/xa;
/* 4.delta1 */
del1 = asin(P0*xe);
/* 5. Q0 */
Q0l = (1.0-cos(del1))/xe;
Q0t = xa;
Q0  = Q0l - Q0t;
/* 6.vd0 Calculation */
A   = ( Vt0 * Vt0 / xq + Q0);
A   = A * A;
A   = A + P0 * P0;
vd0 = P0 * Vt0 / sqrt(A);
/* 7.iq0 Calculation */
C   = Vt0 * Vt0 + xq * Q0;
C   = C * C;
C   = C + xq * xq * P0 * P0;
iq0 = P0 * Vt0 / sqrt(C);
/* 8.vq0 Calculation */
A   = Vt0 * Vt0 - vd0 * vd0;
vq0 = sqrt(A);
/* 9.id0 Calculation */
id0 = (P0 - vq0 * iq0) / vd0;
/* 10.del0 Calculation */
del0 = asin((iq0*xe+vd0-vd0*xa*xe)/V0);
/* 11.fif0 Calculation */
fif0 = ( xdd * V0 * cos(del0)/Xdd ) ;
fif0 = ( vq0 - fif0 ) * Xdd * Td0d / xe ;
/* 12.vf0 Calculation */
vf0 = (Xd*fif0) / (Xdd*Td0d) - (xd-xdd)*V0*cos(del0) /
Xdd;
/* 13.vc0 calculation */
vc0 = sqrt((vd0+(xt+xl1)*iq0)*(vd0+(xt+xl1)*iq0)
   +(vq0-(xt+xl1)*id0)*(vq0-(xt+xl1)*id0));
/* 14.v0 Calculation */
A = iq0 * xe + vd0 - vd0 * xa * xe;
C = sin (del0);
v0 = A/C;
}
void d_out(double y1[])
/* output deviation */
{ dy1[kpt+1] = y1[0] - P0;
 dy2[kpt+1] = y1[1] - Vt0;
 dy3[kpt+1] = y1[3] - vc0;
}
```

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ทำซ้ำ ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void sys1(double x[],double u1[],double f[],double y
[],double dy2[N],double dy3[N],int flag1)
/* calculate system equations */
{
static double A1,A2,B1,B2,B3,C1,C2;
double vd,vq,id,iq;
if(flag1==1)
{ A1 = xl1*xc/(xc-xl1);
  A2 = xl2+xee;
  B1 = 2.0*xl1*xl1 / (3.0*xl1+A1);
  B2 = xl1*A1/(3.0*xl1+A1);
  B3 = 2.0*xl1*A1/(3.0*xl1+A1);
  C1 = xt+B1;
  C2 = A2+B3;

  xe = (C1*C2+B2*(C1+C2))/B2;
  xa = (C1*C2+B2*(C1+C2))/C2;
  xa = -1.0/xa;
  xb = (C1*C2+B2*(C1+C2))/C1;
}
if(flag1==0)
{ xe = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/xc;
  xa = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xl2+xee);
  xb = (xc*(xt+xl1+xl2+xee)-(xt+xl1)*(xl2+xee))/
(xt+xl1);
  xa = 1.0/xa;
}
/* d-axis voltage calculation */
vd = xq * V0 * sin(x[0]) / Xq;
/* q-axis voltage calculation */
vq = (xe*x[2])/(Xdd*Td0d)+xdd*V0*cos(x[0])/Xdd;
/* d-axis current calculation */
id = x[2]*(1.0-xe*xa)/(Xdd*Td0d)-V0*cos(x[0])/Xdd;
/* q-axis current calculation */
iq = V0 * sin(x[0]) / Xq;
/* Real Power calculation : Pe */
y[0] = vd * id + vq * iq;
/* Terminal Voltage calculation : Vt */
y[1] = sqrt(vd * vd + vq * vq);
/* Reactive Power calculation : Qe */
y[2] = vq * id - vd * iq;
/* SVC Terminal Voltage calculation : Vc */
y[3] = sqrt((vd+(xt+xl1)*iq)*(vd+(xt+xl1)*iq)
     +(vq-(xt+xl1)*id)*(vq-(xt+xl1)*id));
/* delta: torque angle */
f[0] = x[1];
/* 1: w */
f[1] = 1.0 / M * (x[6] - D * x[1] - (y[0]-(P0-Pd)));
/* 2: f      */
switch(AVRL)
{ case 1 :f[2] = Efmax -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case -1:f[2] = Efmin -  x[2] / Td0d - (xd - xdd)*id;
  break;
  case 0 :f[2] = (vf0 + x[3])  -  x[2] / Td0d - (xd - xdd)*id;
  break;
}
/* 3: vf */
switch(PSS)
{ case 1 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
       x[4])/Tf+Umpssw))/Te;
  break;
  case -1:
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-x[4])/Tf-
       Umpssw))/Te;
  break;
  case 0 :
  f[3] = (- x[3] + Ue * (- dy2[kpt] -(Kf*ZZ-
       x[4])/Tf+Upss))/Te;
  break;
}
/* 4: Rf */
f[4] = (Kf * ZZ -x[4] ) / Tf;
/* 5: g */
switch(DD1)
{ case 1 :f[5] =  0.1;
  break;
  case -1:f[5] = -0.1;
  break;
  case 0 :f[5] = -SIGM * x[5] / Tg + 1.0 / Tg * - x[1] / W0;
  break;
}
/* 6: h */
switch(DD2)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{ case 1 :f[6] = (-x[6]+(Pmax-(P0-Pd)))/Tw;
  break;
  case -1 :f[6] = (-x[6]-(P0-Pd))/Tw;
  break;
  case 0 :f[6] = (-x[6]+x[5])/Tw;
  break;
 }
 /* 7: SVC : Vcm */
 f[7] = (-dy3[kpt] - x[7]) /Tv;
 /* 8: SVC : Pin          */
 f[8] = ((y[0] - (P0 -Pd )) - x[8]) / TP1;
 /* 9: SVC : Pout */
 f[9] = ((y[0] - (P0 -Pd )) - x[8]) / TP1 -x[9]/ TQ1;
 /* PSS */
 f[11] = (x[1]-x[11])/Tw;
 f[12] = (x[1]-x[11])/Tw - x[12]/Tr;
 Wr = x[12];
 /* 10:SVC : Bs */
 if(select==1)
 { switch(SVCL)
  { case 1 :f[10] = (-x[10]+Ksv * (x[7]+u1[1])) / Tc;
   break;
   case -1 :f[10] = (-x[10]+Ksv * (x[7]+u1[1])) / Tc;
   break;
   case 0 :f[10] = (-x[10]+Ksv * (x[7]+u1[1])) / Tc;
   break;
  }
 }
}
void runge4(double x[],double u1[],double t,int n,
            double dy2[N],double dy3[N],int flag1)
/* Runge-Kutta4 for solving differntial equations */
{
double k1[NA],k2[NA],k3[NA],k4[NA],y[3],z[NA];
int i;
sys1(x, u1, k1, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k1[i];
sys1(z, u1, k2, y,dy2,dy3 ,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t / 2 * k2[i];
sys1(z, u1, k3, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
 z[i] = x[i] + t * k3[i];
sys1(z, u1, k4, y,dy2,dy3,flag1);
for (i = 0; i < n; i++)
x[i] = x[i] + t * (k1[i] + 2 * k2[i] + 2 * k3[i] + k4[i]) / 6;
}
void svcfuz(double u1[],double Vt,double Pe,double x1)
/* fuzzy control */
{
static double SNkP,SGkP;
SNkP = 0.0;
SDEP = sbp*(x1-SEEP)/(2.0*T);
SEEP = x1;
SDkP = sqrt(SDEP*SDEP + x1*x1);
/* Phase plan selection */
svcsita(SDEP,x1,SDkP,&STkP);
svcsimem(STkP,sap,&SNkP,SL3);
/* determine Gain factor */
SGkP = SDkP/sDrp;
if(SGkP>1.0) SGkP=1.0;
/* calculate the SVC supplementary control signal */
u1[1] = (2.0*SNkP-1.0)*Umfp*SGkP;
if(u1[1] < -Umfp) u1[1] =-Umfp;
if(u1[1] >  Umfp) u1[1] = Umfp;
}
void svcsita(double DE,double EE,double Dk,double *Tk)
/* find angle on phase plane */
{
if(DE>0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/PI;
if(DE>0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/PI+180.0;
if(DE<0.0 && EE<0.0)
*Tk= -asin(DE/Dk)*180.0/PI+180.0;
if(DE<0.0 && EE>0.0)
*Tk= asin(DE/Dk)*180.0/PI+360.0;
}
void svcsitaw(double DEW,double EEW,double
              Dkw,double *Tkw)
/* find angle on phase plane */
{
if(DEW>0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI;
if(DEW>0.0 && EEW<0.0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW<0.0)
*Tkw= -asin(DEW/Dkw)*180.0/PI+180.0;
if(DEW<0.0 && EEW>0.0)
*Tkw= asin(DEW/Dkw)*180.0/PI+360.0;
}
void svcsimemw(double Tkw,double aw,double
*Nkw,double SL)
/* find membership function of phase plane angle */
{
if(Tkw >= 0.0 && Tkw < SL - aw / 2.0)
*Nkw = 1.0;
if(Tkw >= SL - aw/2.0 && Tkw <  SL + aw / 2.0)
*Nkw = ( - Tkw + SL ) / aw + 0.5;
if(Tkw >= SL + aw/2.0 && Tkw < SL +180.0 - aw/2.0)
*Nkw = 0.0;
if(Tkw >= SL + 180.0 - aw / 2.0 && Tkw <  SL + 180.0 +
aw / 2.0)
*Nkw = ( Tkw - ( SL + 180.0 ) ) / aw + 0.5;
if(Tkw >= SL + 180.0 + aw / 2.0  && Tkw <= 360.0)
*Nkw = 1.0;
}
void svcsimem(double Tk,double a,double *Nk,double
SL)
/* fine membership function of phase plane angle */
{
if(Tk >= 0.0 && Tk < SL - a / 2.0)
*Nk = 1.0;
if(Tk >= SL - a/2.0 && Tk <  SL + a / 2.0)
*Nk = ( - Tk + SL ) / a + 0.5;
if(Tk >= SL + a/2.0 && Tk < SL +180.0 - a/2.0)
*Nk = 0.0;
if(Tk >= SL + 180.0 - a / 2.0 && Tk <  SL + 180.0 + a /
2.0)
*Nk = ( Tk - ( SL + 180.0 ) ) / a + 0.5;
if(Tk >= SL + 180.0 + a / 2.0  && Tk <= 360.0)
*Nk = 1.0;
}
void svcfuzw(double p1[],double Vt,double Pe,double x1)
/* fuzzy control */
{
static double SNkPW,SGkPW;
SNkPW = 0.0;
SDEPW = sbpw*(x1-SEEPW)/(2.0*T);
SEEPW = x1;
SDkPW = sqrt(SDEPW*SDEPW + x1*x1);
/* Phase plan selection */
svcsitaw(SDEPW,x1,SDkPW,&STkPW);
svcsimemw(STkPW,sapw,&SNkPW,SL3);
/* determine Gain factor */
SGkPW = SDkPW/sDrpw;
if(SGkPW>1.0) SGkPW=1.0;
/* calculate the supplementary control signal  for AVR */
p1[1] = (2.0*SNkPW-1.0)*Umpssw*SGkPW;
if(p1[1] < -Umpssw) p1[0] =-Umpssw;
if(p1[1] >  Umpssw) p1[0] = Umpssw;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก จ.

1. โปรแกรม tfuncb.c

```
/* tfuncb.c : test function used binary code
   Last update Sep 4 , 1998
   Written by Komsan Hongesombut        */
#include<stdio.h>
#include <stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include<time.h>
#include<dos.h>
#include"c:\object.c"
#include"c:\rand3.c"
#define LINELENGTH 80     /* width of printout */
#define BITS_PER_BYTE 8   /* number of bits per byte
on this machine */
#define UINTSIZE (BITS_PER_BYTE*sizeof(unsigned))
/* # of bits in unsigned */
void app_data();
void app_free();
void app_malloc();
void objfunc(struct individual *critter);
void generation();
void initialize();
void initdata();
void initpop();
void initreport();
void initmalloc();
void freeall();
void nomemory(char *string);
int flip(float prob);
void initrandomnormaldeviate();
double noise(double mu ,double sigma);
void randomized();
double randomnormaldeviate();
float randomperc();
int rnd(int low,int high);
float rndreal(float lo ,float hi);
void preselect_no_scaling();
void prescale(double umax,double uavg,double
umin,double *a,double *b);
double scale(double u,double a,double b);
void scalepop(double max,double avg,double min,struct
individual *pop);
void preselect_expo_scaling();
void preselect_window_scaling();
void select_memory();
void select_free();
void ts_preselected();
int selected();
int ts_selected();
void ts_reset();
void statistics(struct individual *pop);
void copyright();
void repchar (FILE *outfp,char *ch,int repcount);
void skip(FILE *outfp,int skipcount);
int ithruj2int(int i,int j,unsigned *from);
void mutation(unsigned *child);
int crossover (unsigned *parent1,unsigned *parent2,
                unsigned *child1 ,unsigned *child2);
void uniform_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2);
void two_point_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2);
double start,finish;
void report();
void writepop();
void writechrom(unsigned *chrom);
void decode_for_best(struct individual *critter);
void writechrom_to_files_bestfit(unsigned *chrom);
void print_to_file_decode_for_best(struct individual
*critter1);
FILE *outfp, *infp;
FILE *fpga;
/* Global structures and variables */
struct individual
{
  unsigned *chrom;    /* chromosome string for the
individual */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    double   fitness;   /* fitness of the individual */
    int   xsite;     /* crossover site at mating */
    int   parent[2]; /* who the parents of offspring were */
    int  *utility;  /* utility field can be used as pointer to a
*/
    /* dynamically allocated, application-specific data
structure */
};
struct bestever
{
    unsigned *chrom;    /* chromosome string for the best-
ever individual */
    double   fitness;   /* fitness of the best-ever individual */
    int     generation; /* generation which produced it */
};
struct individual *oldpop; /* last generation of individuals
*/
struct individual *newpop; /* next generation of
individuals */
struct individual *masku;
static struct bestever bestfit; /* fittest individual so far */
double sumfitness;      /* summed fitness for entire
population */
double max;             /* maximum fitness of population */
double avg;             /* average fitness of population */
double min;             /* minumum fitness of population */
float  pcross,pcross0;  /* probability of crossover */
float  pmutation,pmutation0; /* probability of mutation */
int   numfiles;           /* number of open files */
int   popsize;            /* population size */
int   lchrom;          /* length of the chromosome per
individual */
int   chromsize;    /* number of bytes needed to store
lchrom string */
int   gen;          /* current generation number */
int   maxgen;       /* maximum generation number */
int   run;          /* current run number */
int   maxruns;      /* maximum number of runs to make */
int   printstrings; /* flag to print chromosome strings
(default on) */
double   nmutation; /* number of mutations  */
double   ncross;    /* number of crossovers  */
static int field_size, vec_size; /* derive from app2.c */
/* variables are declared static so that they cannot conflict
with names of   */
/* other global variables in other files.  See K&R, p 80, for
scope of static */
static double oldrand[55]; /*Array of 55 random numbers
*/
static int jrand;         /* current random number */
static double rndx2;      /* used with random normal
deviate */
static int rndcalcflag;/* used with random normal deviate
*/
struct  time t;
int hour1,hour2,
    min1,min2,
    sec1,sec2,
    sechund1,sechund2;
time_t tt;
int select_function;
int select_scaling;
int select_crossover;
double a,b;
int elitist ;
static int   *tourneylist, tourneypos, tourneysize;
/* for tournament selection */

void main()
{
    struct individual *temp;
    FILE  *fopen();
    fpga=fopen("tf4b5.dat","w");
    printstrings=1;
    numfiles = 0;
    infp = stdin;
    outfp = stdout;
    copyright();
    if(numfiles == 0)
            fprintf(outfp," Number of GA runs to be
performed-> ");
    fscanf(infp,"%d",&maxruns);
    for(run=1; run<=maxruns; run++)
    {       /* Set things up */
            initialize();
            /*start = time(NULL);*/
```

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย เพื่อการศึกษาเท่านั้น ห้ามมิให้นำไปใช้ประโยชน์ด้านการค้า กรณีอื่นๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            gettime(&t);
            hour1 = t.ti_hour;
            min1  = t.ti_min;
            sec1  = t.ti_sec;
            sechund1 = t.ti_hund;
            for(gen=0; gen<maxgen; gen++)
            {
              if(gen%1==0)
              {
               fprintf(outfp,"\nRUN %d of %d:
  GENERATION %d->%d\n",

  run,maxruns,gen,maxgen);
              }
              /* create a new generation */
              generation();
              /* compute fitness statistics on new
  populations */
              statistics(newpop);
              /* report results for new generation */
              if(gen%1==0)
              { report(); }
              /* advance the generation */
              temp = oldpop;
              oldpop = newpop;
              newpop = temp;
            }
            gettime(&t);
            hour2 = t.ti_hour;
            min2  = t.ti_min;
            sec2  = t.ti_sec;
            sechund2 = t.ti_hund;
            fprintf(fpga,"\n");
            fprintf(fpga,"Global Best Individual so far,
  Generation %d:\n",
                        bestfit.generation);
            print_to_file_decode_for_best(&(bestfit));
            decode_for_best((&bestfit));
            objfunc((&bestfit));
            fprintf(fpga,"performance index = %10.15f: ",
  (bestfit.fitness));
            fprintf(fpga,"\nThe start process time is:
  %2d:%02d:%02d.%02d\n",
                        hour1,min1,sec1,sechund1);
                        printf("The stop process time is:
  %2d:%02d:%02d.%02d\n",
                        t.ti_hour, t.ti_min, t.ti_sec, t.ti_hund);
                        fprintf(fpga,"The stop process time is:
  %2d:%02d:%02d.%02d\n",
                        t.ti_hour, t.ti_min, t.ti_sec, t.ti_hund);
                        freeall(); fclose(fpga);
            }
  }
  void app_data()
  /* application dependent data input, called by init_data() */
  /* ask your input questions here, and put output in global
  variables */
  /* In this example application, the utility pointer of each
  individual is  */
  /* assigned a vector of integers that will be used to store
  the interpreted */
  /* chromosome. */
  {  int size = UINTSIZE;
    if(lchrom < UINTSIZE) size = lchrom;
    /* user must specify length of concatenated integers in
  the chromosome. */
     fprintf(outfp," Enter field size (must be less than %d) -
  >", size);
     fscanf(infp,"%d",&field_size);
     vec_size = lchrom/field_size;
     if((((float)lchrom/(float)field_size)-vec_size) > 0.0)
  vec_size++;
  }
  void app_free()
  /* This routine should free any memory allocated */
  /* in the application-dependent routines, called by freeall()
  */
  {  int i;
     for(i = 0; i < popsize; i++)
     {        free(newpop[i].utility);
              free(oldpop[i].utility);
     }
  }
  void app_malloc()
  /* application dependent malloc() calls, called by
  initmalloc() */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{  unsigned nbytes;
   int i;
   nbytes = vec_size * sizeof(int);
   for(i = 0; i < popsize; i++)
   {        if((newpop[i].utility = (int *) malloc(nbytes))
== NULL)
               nomemory("newpop utility");
            if((oldpop[i].utility = (int *) malloc(nbytes)) ==
NULL)
               nomemory("oldpop utility");
   }
}
void app_report()
/* Application-dependent report, called by report() */
{  int i, j;
   /* Print vector interpretation of each chromosome. */
   for(i = 0; i < popsize; i++)
   {        fprintf(outfp,"oldpop %d = ", i);
            for(j = 0; j < vec_size; j++)
               fprintf(outfp, "%d ", oldpop[i].utility[j]);
            fprintf(outfp,"\n");
   }
}
void preselect_no_scaling()
/* prselect in no scaling case */
{  int j;
   sumfitness = 0;
   for(j = 0; j < popsize; j++) sumfitness += oldpop
[j].fitness;
}
void prescale(double umax,double uavg,double
umin,double *a,double *b)
/* used for linear scaling */
{ double fmultiple = 2.0;
 double delta;
 fmultiple = 2.0;
 if(umin>((fmultiple*uavg)-umax)/(fmultiple-1)) /* non
negative test */
 { /* normal scaling */
  delta = umax-uavg;
  *a = ((fmultiple-1.0)*uavg)/delta;
  *b = uavg*(umax-fmultiple*uavg)/delta;
 }
 else
 { /* scale as much as possible */
  delta = uavg-umin;
  *a = uavg/delta;
  *b = -umin*uavg/delta;
 }
}
double scale(double u,double a,double b)
/* used for linear scaling */
{  return(fabs((a*u) + b));
}
void scalepop(double max,double avg,double min,struct
individual *pop)
/* used for linear scaling */
{ int j;
 double a=0.0,b=0.0;
 prescale(max,avg,min,&a,&b);
 sumfitness=0.0;
 for(j=0;j<popsize;j++) {
  pop[j].fitness = scale(pop[j].fitness,a,b);
  sumfitness+=pop[j].fitness;
 }
}
void preselect_window_scaling()
/* used for window scaling */
{  int j;
 sumfitness = 0;
 for(j = 0; j < popsize; j++)
 { oldpop[j].fitness -= min;
  sumfitness += oldpop[j].fitness;
 }
}
void preselect_expo_scaling()
/* used for exponential scaling */
{  int j;
  sumfitness = 0;
  for(j = 0; j < popsize; j++)
  { oldpop[j].fitness -= min;
   oldpop[j].fitness += 1.0;
   oldpop[j].fitness *= oldpop[j].fitness;
   sumfitness += oldpop[j].fitness;
  }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int selected()
/* roulette-wheel selection */
{  float sum, pick;
   int i;
   pick = randomperc()*sumfitness;
   sum = 0;
   if(sumfitness != 0)
   {        for(i = 0; (sum < pick) && (i < popsize); i++)
               sum += oldpop[i].fitness;
   }
   else
            i = rnd(1,popsize);
   return(i-1);
}
void generation()
{ int mate1, mate2, jcross, j = 0;
 int k;
 /* perform any preselection actions necessary before
generation */
 if(select_scaling == 1) /* no scaling */
 { preselect_no_scaling();
 }
 if(select_scaling == 2) /* linear scaling */
 { prescale(max,avg,min,&a,&b);
   scalepop(max,avg,min,oldpop);
 }
 if(select_scaling == 3)
 { preselect_window_scaling();
 }
 if(select_scaling == 4)
 { preselect_expo_scaling();
 }
 if(select_scaling == 5)
 { ts_preselected();
 }
 pcross = pcross0; pmutation = pmutation0;
 /* select, crossover, and mutation */
 do
 {
   /* pick a pair of mates */
   if(select_scaling == 5)
   {
            mate1=ts_selected();
            mate2=ts_selected();
   }
   else
   {
            mate1=selected();
            mate2=selected();
   }
   /* Crossover and mutation */
   if(select_crossover==1)
   {
            jcross = crossover(oldpop[mate1].chrom,
oldpop[mate2].chrom,
                                    newpop[j].chrom,
newpop[j+1].chrom);
   }
   else
   if(select_crossover==2)
   {
            two_point_crossover(oldpop[mate1].chrom,
oldpop[mate2].chrom,
                                    newpop[j].chrom,
newpop[j+1].chrom);
   }
   else
   if(select_crossover==3)
   {
    uniform_crossover(oldpop[mate1].chrom, oldpop
[mate2].chrom,
                                    newpop[j].chrom,
newpop[j+1].chrom);
   }
   mutation(newpop[j].chrom);
   mutation(newpop[j+1].chrom);
   pcross -= (float)((pcross0-0.6)/maxgen);
   pmutation += (float)((0.1-pmutation0)/maxgen);
   /* Decode string, evaluate fitness, & record */
   /* parentage date on both children */
   objfunc(&(newpop[j]));
   newpop[j].parent[0] = mate1+1;
   newpop[j].xsite = jcross;
   newpop[j].parent[1] = mate2+1;
   objfunc(&(newpop[j+1]));
   newpop[j+1].parent[0] = mate1+1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    newpop[j+1].xsite = jcross;
    newpop[j+1].parent[1] = mate2+1;
    /* Increment population index */
    j = j + 2;
   }
  while(j < (popsize-1));
  statistics(newpop);
  if(elitist == 1)
  {
    for(k = 0; k < chromsize; k++)
        {
          newpop[0].chrom[k] = bestfit.chrom[k];
        }
    objfunc(&(newpop[0]));
  }
}
void initialize()
/* Initialization Coordinator */
{  /* get basic problem values from input file */
   initdata();
/* define chromosome size in terms of machine bytes, ie
*/
 /* length of chromosome in bits (lchrom)/(bits-per-byte)
*/ /* chromsize must be known for malloc() of chrom
pointer */
   chromsize = (lchrom/UINTSIZE);
   if(lchrom%UINTSIZE) chromsize++;
   /* malloc space for global data structures */
   initmalloc();
   /* initialize random number generator */
   randomized();
   /* initialize global counters/values */
   nmutation = 0;
   ncross = 0;
   bestfit.fitness = 0.0;
   bestfit.generation = 0;
   /* initialize the populations and report statistics */
   initpop();
   statistics(oldpop);
   initreport();
}
void initdata()
/* data inquiry and setup */
{  char answer[2];
   if(numfiles == 0)
   {       fprintf(outfp,"\n ------- GA Data Entry and
Initialization -------\n");
           fprintf(outfp," Select one funtion to optimize :
\n");
           fprintf(outfp," 1. f(x,y) = x^2 + 2y^2 - 0.3cos
(3px) - 0.4cos(4py) + 0.7 \n");
           fprintf(outfp," 2. f(x,y) = x^2 + 2y^2 - 0.3[cos
(3px)cos(4py)] + 0.3 \n");
           fprintf(outfp," 3. f(x,y) = x^2 + 2y^2 - 0.3[cos
(3px) + cos(4py)] + 0.3 \n");
           fprintf(outfp," 4. f(x,y) = 1/((x+0.5)^2 + 2(y-
0.5)^2 - 0.3cos(3px) - 0.4cos(4py) + 0.8) \n");
           fprintf(outfp," Select ----> ");
   }
   fscanf(infp,"%d",&select_function);
   if(numfiles == 0)
   {
           fprintf(outfp," Enter the population size --------
---> ");
   }
   fscanf(infp,"%d", &popsize);
   if((popsize%2) != 0)
   {
           fprintf(outfp, "Sorry! only even population
sizes are allowed. \n Incrementing popsize by one.\n");
           popsize++;
   };
   if(numfiles == 0)
           fprintf(outfp," Enter chromosome length (56) -
-------> ");
   fscanf(infp,"%d", &lchrom);
   if(numfiles == 0)
           fprintf(outfp," Print chromosome strings? (y/n)
------> ");
   fscanf(infp,"%s",answer);
   if(strncmp(answer,"n",1) == 0) printstrings = 0;
   if(numfiles == 0)
   {       fprintf(outfp," Enter maximum number of
generations --> ");
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            fprintf(fpga," Enter maximum number of
  generations --> ");
    }
    fscanf(infp,"%d", &maxgen);
    fprintf(fpga,"%d \n",maxgen);
    if(numfiles == 0)
    {  fprintf(outfp," Select one crossover method :\n");
            fprintf(outfp," 1. One-point crossover \n");
            fprintf(outfp," 2. Two-point crossover \n");
            fprintf(outfp," 3. Uniform crossover \n");
            fprintf(outfp," Select ----> ");
            fprintf(fpga," Select one crossover method
  :\n");
            fprintf(fpga," 1. One-point crossover \n");
            fprintf(fpga," 2. Two-point crossover \n");
            fprintf(fpga," 3. Uniform crossover \n");
            fprintf(fpga," Select ----> ");
    }
            fscanf(infp,"%d",&select_crossover);
            fprintf(fpga,"%d \n",select_crossover);
    if(numfiles == 0)
    {       fprintf(outfp," Enter initial crossover
  probability ----------> ");
            fprintf(fpga," Enter initial crossover probability
  ----------> ");
    fscanf(infp,"%f", &pcross0);
    fprintf(fpga,"%f \n",pcross0);
    }
    if(numfiles == 0)
    {       fprintf(outfp," Enter initial mutation probability
  ----------> ");
            fprintf(fpga," Enter initial mutation probability
  ----------> ");
    fscanf(infp,"%f", &pmutation0);
    fprintf(fpga,"%f \n",pmutation0);
    }
    if(numfiles == 0)
    {       fprintf(outfp," Select one scaling method : \n");
            fprintf(outfp," 1. No scaling \n");
            fprintf(outfp," 2. Linear scaling method \n");
            fprintf(outfp," 3. Window scaling method \n");
            fprintf(outfp," 4. Exponential scaling method
  \n");
            fprintf(outfp," 5. Tournament selection method
  \n");
            fprintf(outfp," Select ----> ");
            fprintf(fpga," Select one scaling method : \n");
            fprintf(fpga," 1. No scaling \n");
            fprintf(fpga," 2. Linear scaling method \n");
            fprintf(fpga," 3. Window scaling method \n");
            fprintf(fpga," 4. Exponential scaling method
  \n");
            fprintf(fpga," 5. Tournament selection method
  \n");
            fprintf(fpga," Select ----> ");
     }
    fscanf(infp,"%d",&select_scaling);
    fprintf(fpga,"%d \n",select_scaling);
    if(numfiles == 0)
    {       fprintf(outfp," Enter Elitist Selection (1:Y | 0:N)
  --> ");
            fprintf(fpga," Enter Elitist Selection (1:Y | 0:N)
  --> ");
    fscanf(infp,"%d",&elitist);
    fprintf(fpga,"%d \n",elitist);
    }
    app_data();
  }
  void initpop()
  /* Initialize a population at random */
  {  int j, j1, k, stop;
    unsigned mask = 1;
    for(j = 0; j < popsize; j++)
    {
        for(k = 0; k < chromsize; k++)
        {
          oldpop[j].chrom[k] = 0;
          if(k == (chromsize-1))
                stop = lchrom - (k*UINTSIZE);
          else
                stop = UINTSIZE;
          /* A fair coin toss */
          for(j1 = 1; j1 <= stop; j1++)
          { oldpop[j].chrom[k] = oldpop[j].chrom[k]
  <<1;
            if(flip(0.5))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    oldpop[j].chrom[k] = oldpop
[j].chrom[k]|mask;
                }
            }
            oldpop[j].parent[0] = 0; /* Initialize parent info.
*/
            oldpop[j].parent[1] = 0;
            oldpop[j].xsite = 0;
            objfunc(&(oldpop[j])); /* Evaluate initial
fitness */
    }
}
void initreport()
/* Initial report */
{  void  skip();
   skip(outfp,1);
   fprintf(outfp," GA Parameters\n");
   fprintf(outfp," ----------------------------------------
\n");
   fprintf(outfp," Total Population size          =
%d\n",popsize);
   fprintf(outfp," Chromosome length (lchrom)     =
%d\n",lchrom);
   fprintf(outfp," Maximum # of generations (maxgen) =
%d\n",maxgen);
   fprintf(outfp," Initial crossover probability (pcross)    =
%f\n", pcross);
   fprintf(outfp," Initial mutation  probability (pmutation)
= %f\n", pmutation);
   skip(outfp,1);

   fprintf(fpga," GA Parameters\n");
   fprintf(fpga," ----------------------------------------
\n");
   fprintf(fpga," Total Population size          =
%d\n",popsize);
   fprintf(fpga," Chromosome length (lchrom)     =
%d\n",lchrom);
   fprintf(fpga," Maximum # of generations (maxgen) =
%d\n",maxgen);
   fprintf(fpga," Initial crossover probability (pcross)    =
%f\n", pcross);
   fprintf(fpga," Initial mutation  probability (pmutation) =
%f\n", pmutation);
   fprintf(fpga," ----------------------------------------
\n\n");
   fprintf(fpga," min      max      avg      Σfitness
best.fitness    chromosome \n\n");
   fflush(outfp);
}
void initmalloc()
/* memory allocation of space for global data structures */
{ unsigned nbytes;
 int j;
 /* memory for old and new populations of individuals */
 nbytes = popsize*sizeof(struct individual);
 if((oldpop = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("oldpop");
 if((newpop = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("newpop");
 if((masku = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("masku");
 /* memory for chromosome strings in populations */
 nbytes = chromsize*sizeof(unsigned);
 for(j = 0; j < popsize; j++)
 { if((oldpop[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
      nomemory("oldpop chromosomes");

   if((newpop[j].chrom = (unsigned *) malloc(nbytes))
== NULL)
      nomemory("newpop chromosomes");
   if((masku[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
      nomemory("masku chromosomes");
 }
 if((bestfit.chrom = (unsigned *) malloc(nbytes)) ==
NULL)
  nomemory("bestfit chromosome");
 if(select_scaling == 5)
 { select_memory();
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    /* call to application-specific malloc() routines   */
    /* can be used to malloc memory for utility pointer */
    app_malloc();
  }
  void freeall()
  /* A routine to free all the space dynamically allocated in
  initspace() */
  { int i;
    for(i = 0; i < popsize; i++)
     { free(oldpop[i].chrom);
       free(newpop[i].chrom);
       free(masku[i].chrom);
     }
    free(oldpop);
    free(newpop);
    free(masku);
    free(bestfit.chrom);
    select_free();
    app_free();
  }
  void nomemory(char *string)
  { fprintf(stderr,"malloc: out of memory making
  %s!!\n",string);
    exit(-1);
  }
  int flip(float prob)
  /* Flip a biased coin - true if heads */
  { if(randomperc() <= prob)
              return(1);
    else
              return(0);
  }
  void initrandomnormaldeviate()
  /* initialization routine for randomnormaldeviate */
  {  rndcalcflag = 1;
  }
  double noise(double mu ,double sigma)
  /* normal noise with specified mean & std dev: mu &
  sigma */
  { double randomnormaldeviate();
    return((randomnormaldeviate()*sigma) + mu);
  }
  void randomized()
  /* Get seed number for random and start it up */
  { sran2((unsigned int) time(&tt));
  }
  double randomnormaldeviate()
  /* random normal deviate after ACM algorithm 267 / Box-
  Muller Method */
  {  double t, rndx1;
     if(rndcalcflag)
     {       rndx1 = sqrt(- 2.0*log((double) randomperc
  ()));
             t = 6.2831853072 * (double) randomperc();
             rndx2 = rndx1 * sin(t);
             rndcalcflag = 0;
             return(rndx1 * cos(t));
     }
     else
     {       rndcalcflag = 1;
             return(rndx2);
     }
  }
  float randomperc()
  /* Fetch a single random number between 0.0 and 1.0  */
  { return(ran2());
  }
  int rnd(int low,int high)
  /* Pick a random integer between low and high */
  {  int i;
     if(low >= high)
             i = low;
     else
     {       i = (randomperc() * (high - low + 1)) + low;
             if(i > high) i = high;
     }
     return(i);
  }
  float rndreal(float lo ,float hi)
  /* real random number between specified limits */
  {  return((randomperc() * (hi - lo)) + lo);
  }
  void statistics(struct individual *pop)
  /* Calculate population statistics */
  {  int i, j;
     sumfitness = 0.0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    min = pop[0].fitness;
    max = pop[0].fitness;
    /* Loop for max, min, sumfitness */
    for(j = 0; j < popsize; j++)
    {
            sumfitness = sumfitness + pop[j].fitness;     /*
  Accumulate */
            if(pop[j].fitness > max) max = pop[j].fitness; /*
  New maximum */
            if(pop[j].fitness < min) min = pop[j].fitness; /*
  New minimum */
            /* new global best-fit individual */
            if(pop[j].fitness > bestfit.fitness)
             { for(i = 0; i < chromsize; i++)
               bestfit.chrom[i]     = pop[j].chrom[i];
              bestfit.fitness    = pop[j].fitness;
              bestfit.generation = gen;
             }
    }
    /* Calculate average */
    avg = sumfitness/popsize;
}
void copyright()
{  int iskip;
   int ll = 59;
   iskip = (LINELENGTH - ll)/2;
   skip(outfp,1);
   repchar(outfp," ",iskip); repchar(outfp,"-",ll); skip
(outfp,1);
   repchar(outfp," ",iskip-1);
   fprintf(outfp,"|      GA-C (v1.1) - A  Genetic Algorithm
|\n");
   repchar(outfp," ",iskip);
   repchar(outfp,"-",ll); skip(outfp,2);
}
void repchar (FILE *outfp,char *ch,int repcount)
/* Repeatedly write a character to stdout */
{  int j;
   for (j = 1; j <= repcount; j++) fprintf(outfp,"%s", ch);
}
void skip(FILE *outfp,int skipcount)
/* Skip skipcount lines */
{  int j;
   for (j = 1; j <= skipcount; j++) fprintf(outfp,"\n");
}
int ithruj2int(int i,int j,unsigned *from)
/* interpret bits i thru j of a individual as an integer    */
/* j MUST BE greater than or equal to i AND j-i <
UINTSIZE-1  */
/* from is a chromosome, represented as an array of
unsigneds */
{  unsigned mask, temp;
   int bound_flag;
   int iisin, jisin;
   int i1, j1, out;
   if(j < i)
   {       fprintf(stderr,"Error in ithruj2int: j < i\n");
           exit(-1);
   }
   if(j-i+1 > UINTSIZE)
   {       fprintf(stderr,"Error in ithruj2int: j-i+1 >
UINTSIZE\n");
           exit(-1);
   }
   iisin = i/UINTSIZE;
   jisin = j/UINTSIZE;

   i1 = i - (iisin*UINTSIZE);
   j1 = j - (jisin*UINTSIZE);

          if(i1 == 0)
          {  iisin = iisin-1;
             i1 = i - (iisin*UINTSIZE);
          };

          if(j1 == 0)
          {  jisin = jisin-1;
             j1 = j - (jisin*UINTSIZE);
          };
/* check if bits fall across a word boundary */
if(iisin == jisin)
          bound_flag = 0;
else
          bound_flag = 1;
if(bound_flag == 0)
{         mask = 1;
```

สารนิพนธ์นี้เป็นลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            mask = (mask<<(j1-i1+1))-1;
            mask = mask<<(i1-1);
            out = (from[iisin]&mask)>>(i1-1);
            return(out);
    }
    else
    {       mask = 1;
            mask = (mask<<j1)-1;
            temp = from[jisin]&mask;
            mask = 1;
            mask = (mask<<(UINTSIZE-i1+1))-1;
            mask = mask<<(i1-1);
            out = ((from[iisin]&mask)>>(i1-1)) | (temp
<<(UINTSIZE-i1+1));
            return(out);
    }
}
void mutation(unsigned *child)
/* Mutate an allele w/ pmutation, count # of mutations */
{  int j, k, stop;
   unsigned mask, temp = 1;
   for(k = 0; k < chromsize; k++)
   {       mask = 0;
           if(k == (chromsize-1))
             stop = lchrom - (k*UINTSIZE);
           else
             stop = UINTSIZE;
           for(j = 0; j < stop; j++)
           {
             if(flip(pmutation))
             {
                    mask = mask|(temp<<j);
                    nmutation++;
             }
           }
           child[k] = child[k]^mask;
   }
}
int crossover (unsigned *parent1,unsigned *parent2,
               unsigned *child1 ,unsigned *child2)
/* Cross 2 parent strings, place in 2 child strings */
{  int j, jcross, k;
   unsigned mask, temp;
   /* Do crossover with probability pcross */
   if(flip(pcross))
   {       jcross = rnd(1 ,(lchrom - 1));/* Cross between 1
and l-1 */
           ncross++;
           for(k = 1; k <= chromsize; k++)
           {
             if(jcross >= (k*UINTSIZE))
             {
                    child1[k-1] = parent1[k-1];
                    child2[k-1] = parent2[k-1];
             }
             else if((jcross < (k*UINTSIZE)) && (jcross
> ((k-1)*UINTSIZE)))
             {
                    mask = 1;
                    for(j = 1; j <= (jcross-1-((k-
1)*UINTSIZE)); j++)
                    {
                      temp = 1;
                      mask = mask<<1;
                      mask = mask|temp;
                    }
                    child1[k-1] =
(parent1[k-1]&mask)|(parent2[k-1]&(~mask));
                    child2[k-1] =
(parent1[k-1]&(~mask))|(parent2[k-1]&mask);
             }
             else
             {
                    child1[k-1] = parent2[k-1];
                    child2[k-1] = parent1[k-1];
             }
           }
   }
   else
   {
         for(k = 0; k < chromsize; k++)
         {  child1[k] = parent1[k];
            child2[k] = parent2[k];
         }
         jcross = 0;
   }
```

กสารนี้เป็นลิขสิทธิ์ของ [illegible] เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    return(jcross);
  }
  void report()
  /* Write the population report */
  {  repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
    if(printstrings == 1)
    {       repchar(outfp," ",((LINELENGTH-17)/2));
            fprintf(outfp,"Population Report\n");
            fprintf(outfp, "Generation %3d", gen);
            repchar(outfp," ",(LINELENGTH-28));
            fprintf(outfp, "Generation %3d\n", (gen+1));
            fprintf(outfp,"num  string ");
            repchar(outfp," ",lchrom-5);
            fprintf(outfp,"fitness   parents xsite  ");
            fprintf(outfp,"string ");
            repchar(outfp," ",lchrom-5);
            fprintf(outfp,"fitness\n");
            repchar(outfp,"-",LINELENGTH);
            skip(outfp,1);
            writepop();
            repchar(outfp,"-",LINELENGTH);
            skip(outfp,1);
    }
    /* write the summary statistics in global mode  */
    fprintf(outfp,"Generation %d Accumulated Statistics:
\n",gen);
    fprintf(outfp,"Total Crossovers = %f, Total Mutations =
%f\n",ncross,nmutation);
    fprintf(outfp,"min = %10.9f  max = %10.9f  avg =
%10.9f  sum = %10.9f\n",min,max,avg,sumfitness);
    fprintf(fpga," %10.9f  %10.9f  %10.9f  %10.9f
%10.15f  ",min,max,avg,sumfitness,bestfit.fitness);
    fprintf(outfp,"Global Best Individual so far, Generation
%d:\n",
                        bestfit.generation);
    fprintf(outfp,"Fitness = %f: ", bestfit.fitness);
    writechrom((&bestfit)->chrom);fprintf(fpga,"\n");
    skip(outfp,1);
    decode_for_best((&bestfit));
    objfunc((&bestfit));
    fprintf(outfp,"performance index = %10.15f: ",
(bestfit.fitness));
    skip(outfp,1);
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
  }
  void writepop()
  {  struct individual *pind;
    int j;
    for(j=0; j<=popsize-1; j++)
    {       fprintf(outfp,"%3d)  ",j+1);
            /* Old string */
            pind = &(oldpop[j]);
            writechrom(pind->chrom);
            fprintf(outfp,"  %8f | ", pind->fitness);
            /* New string */
            pind = &(newpop[j]);
            fprintf(outfp,"(%2d,%2d)  %2d  ",
            pind->parent[0], pind->parent[1], pind->xsite);
            writechrom(pind->chrom);
            fprintf(outfp,"  %8f\n", pind->fitness);
    }
  }
  void writechrom(unsigned *chrom)
  /* Write a chromosome as a string of ones and zeroes
  */
  {  int j, k, stop;
    unsigned mask=1,tmp;
    static int bit_count=1;
    unsigned mask1;
    int m;
    bit_count=1;
    for(k = chromsize-1; k>=0; k--)
    {       tmp = chrom[k];
            if(k == (chromsize-1))
              { stop = lchrom - (k*UINTSIZE);m=0x1
<<(stop-1);}
            else
              { stop = UINTSIZE;m=0x1<<(stop-1);}
            mask1=m;
            for(j = 0; j < stop; j++)
            {  if(tmp&mask1)
                    fprintf(outfp,"1");
               else
                    fprintf(outfp,"0");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            mask1>>=1;
          }
      }
  }
void objfunc(struct individual *critter)
/* objective function */
{  unsigned mask=0x1;  /* mask for current bit */
   unsigned bitpos;  /* current bit position */
   unsigned bitpos2=0,bitpos3=0;
   unsigned tp;
   double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
   int j, k, stop;
   int n = 10;
   double xx,yy;
   xx=0.0; yy=0.0;
   bitpow2=0; bitpow3=0;
   critter->fitness = 0.0;
   /* loop thru number of bytes holding chromosome */
   for(k = 0; k <= chromsize-1; k++)
   {       if(k == (chromsize-1))
             {stop = lchrom-(k*UINTSIZE);}
           else
             {stop = UINTSIZE;}
           /* loop thru bits in current byte */
           tp = critter->chrom[k];/*tp=oldpop[0].chrom
[0]; */
           for(j = 0; j < stop; j++)
           {  bitpos = j + UINTSIZE*k;/*printf(" %d
",bitpos); */
              /* test for current bit 0 or 1 */
              if((bitpos>=0)&&(bitpos<=27))
              { if((tp&mask) == 1)
                {       yy += pow(2.0,(double) bitpos);
                }
              }
              if((bitpos>=28)&&(bitpos<=55))
              { if((tp&mask) == 1)
                {       xx +=pow(2.0,(double)(bitpos-28));
                }
              }
           tp = tp>>=1;
         }
  }
    xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
    yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
    critter->fitness =
performance_index(xx,yy,select_function);
}
void decode_for_best(struct individual *critter1)
/* objective function */
{  unsigned mask=0x1;  /* mask for current bit */
   unsigned bitpos;  /* current bit position */
   unsigned bitpos2=0,bitpos3=0;
   unsigned tp;
   double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
   int j, k, stop;
   int n = 10;
   double xx,yy;
   xx=0.0; yy=0.0;
   bitpow2=0; bitpow3=0;
   critter1->fitness = 0.0;
   /* loop thru number of bytes holding chromosome */
   for(k = 0; k <= chromsize-1; k++)
   {
      if(k == (chromsize-1))
        {stop = lchrom-(k*UINTSIZE);}
      else
        {stop = UINTSIZE;}
      /* loop thru bits in current byte */
      tp = critter1->chrom[k];/*tp=oldpop[0].chrom
[0]; */
      for(j = 0; j < stop; j++)
      {  bitpos = j + UINTSIZE*k;/*printf(" %d
",bitpos); */
        /* test for current bit 0 or 1 */
        if((bitpos>=0)&&(bitpos<=27))
        { if((tp&mask) == 1)
          { yy += pow(2.0,(double) bitpos);
          }
        }
        if((bitpos>=28)&&(bitpos<=55))
        { if((tp&mask) == 1)
          { xx +=pow(2.0,(double)(bitpos-28));
          }
        }
      tp = tp>>=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
        }
        xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
        yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
        critter1->fitness =
    performance_index(xx,yy,select_function);
        fprintf(outfp,"x = %10.9f   y = %10.9f   \n",xx,yy);
    }
    void writechrom_to_files_bestfit(unsigned *chrom)
    /* Write a chromosome as a string of ones and zeroes
    */
    {   int j, k, stop;
        unsigned mask=1,tmp;
        static int bit_count=1;
        unsigned mask1;
        int m;
        bit_count=1;
        for(k = chromsize-1; k>=0; k--)
        {       tmp = chrom[k];
                if(k == (chromsize-1))
                  { stop = lchrom - (k*UINTSIZE);m=0x1
    <<(stop-1);}
                else
                  { stop = UINTSIZE;m=0x1<<(stop-1);}
                mask1=m;
                for(j = 0; j < stop; j++)
                {
                  if(tmp&mask1)
                          fprintf(fpga,"1");
                  else
                          fprintf(fpga,"0");
                  mask1>>=1;
                }
        }
    }
    void print_to_file_decode_for_best(struct individual
    *critter1)
    /* objective function */
    {   unsigned mask=0x1;  /* mask for current bit */
        unsigned bitpos;  /* current bit position */
        unsigned bitpos2=0,bitpos3=0;
        unsigned tp;
        double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
        int j, k, stop;
        int n = 10;
        double xx,yy;
        xx=0.0; yy=0.0;
        bitpow2=0; bitpow3=0;
        critter1->fitness = 0.0;
        /* loop thru number of bytes holding chromosome */
        for(k = 0; k <= chromsize-1; k++)
        {
                if(k == (chromsize-1))
                  {stop = lchrom-(k*UINTSIZE);}
                else
                  {stop = UINTSIZE;}
                /* loop thru bits in current byte */
                tp = critter1->chrom[k];/*tp=oldpop[0].chrom
    [0]; */
                for(j = 0; j < stop; j++)
                {
                  bitpos = j + UINTSIZE*k;/*printf(" %d
    ",bitpos); */
                  /* test for current bit 0 or 1 */
                  if((bitpos>=0)&&(bitpos<=27))
                  {
                  if((tp&mask) == 1)
                  {
                          yy += pow(2.0,(double) bitpos);
                  }
                  }
                  if((bitpos>=28)&&(bitpos<=55))
                  {
                  if((tp&mask) == 1)
                  {
                          xx +=pow(2.0,(double)(bitpos-28));
                  }
                  }
                  tp = tp>>=1;
                }
        }
        xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
        yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
        critter1->fitness =
    performance_index(xx,yy,select_function);
        fprintf(fpga,"x = %10.15f   y = %10.15f \n",xx,yy);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
void uniform_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2)
{ int i, j, b,l, which_way, data_width;
 int j1, stop;
 unsigned mask1;
 int cp1,cp2;
 int m;
 int k;
 struct individual *idp;
 int fraction;
 if (flip(pcross))
 {
 ncross++;
 for(k = 0; k < chromsize; k++)
        {
          masku[1].chrom[k] = 0;
          if(k == (chromsize-1))
                stop = lchrom - (k*UINTSIZE);
          else
                stop = UINTSIZE;

          /* A fair coin toss */
          for(j1 = 1; j1 <= stop; j1++)
          {
            masku[1].chrom[k] = masku[1].chrom[k]
<<1;
            if (flip(0.5))/*((float)rand()/(float)
RAND_MAX < 0.5)*/
                masku[1].chrom[k] = masku
[1].chrom[k]|0x1;
          }
        }
 k=(chromsize-1);
 fraction = lchrom-((chromsize-1)*16);
 m=0x1 << (fraction-1);
 mask1=m;
 child1[k]=0x0;
 child2[k]=0x0;
 for(i=1;i<=fraction;i++)
 { cp1=(mask1&(parent1[k]))>>(fraction-i);
  cp2=(mask1&(parent2[k]))>>(fraction-i);
  if ((masku[1].chrom[k]&mask1)>>(fraction-i))
  { child1[k]<<=1; child1[k]=(child1[k]|cp2);
   child2[k]<<=1; child2[k]=(child2[k]|cp1);
  }
  else { child1[k]<<=1; child1[k]=(child1[k]|cp1);
         child2[k]<<=1; child2[k]=(child2[k]|cp2);
         }
  mask1>>=1;
 }
 l=(k-1);
 for(k=l;k>=0;k--)
 {
 m=0x1 << 15;
 mask1=m;
 child1[k]=0x0;
 child2[k]=0x0;
 for(i=1;i<=16;i++)
 { cp1=(mask1&(parent1[k]))>>(16-i);
  cp2=(mask1&(parent2[k]))>>(16-i);
  if ((masku[1].chrom[k]&mask1)>>(16-i))
  { child1[k]<<=1; child1[k]=(child1[k]|cp2);
   child2[k]<<=1; child2[k]=(child2[k]|cp1);
  }
  else { child1[k]<<=1; child1[k]=(child1[k]|cp1);
         child2[k]<<=1; child2[k]=(child2[k]|cp2);
         }
  mask1>>=1;
 }
 }
 printf(" CH1 : ");
 }
else
{
 for(k=0;k<chromsize; k++)
 {
  child1[k] = parent1[k];
  child2[k] = parent2[k];
 }
}
}
void two_point_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2)
{ int i, j, b,l, which_way, data_width;
 int j1,max,min,stop,endtop,enddown;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจำหน่าย จ่ายแจก หรือทำซ้ำ ไม่ให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 unsigned  mask1;
 int m,k,p1,p2;
 struct individual *idp;
 struct individual *childpop;
 unsigned mask_record[100];
 if(flip(pcross))
 { ncross++;
 /* create 2-point mask */
 p1=rnd(1,lchrom-1); p2=rnd(1,lchrom-1); /* pick 2 point
*/
 if (p1>p2) { max=p1; min=p2;}  /* p1 locate up and p2
locate down */
 else
 if (p2>p1) { max=p2; min=p1;}  /* p2 locate up and p1
locate down */
 else {max=min=p1=p2;}        /* same point */
 if (max!=min)             /* swap chromosome for mask */
 {
 for(i=1;i<=min;i++)
  mask_record[i]=1;
 for(i=min+1;i<=max;i++)
  mask_record[i]=0;
 for(i=max+1;i<=lchrom;i++)
  mask_record[i]=1;
 }
 else                    /* swap chromosome in case
                          crossover site is the same */
 {
 for(i=1;i<=max;i++)
  mask_record[i]=0;
 for(i=max+1;i<=lchrom;i++)
  mask_record[i]=1;
 }
 endtop=0; enddown=1;
 for(k = 0; k < chromsize; k++)
 {
 masku[1].chrom[k] =0x0;
 if(k == (chromsize-1))
 stop = lchrom - (k*UINTSIZE);
 else
 stop = UINTSIZE;
 endtop+=stop;
 for(j1=endtop;j1>=enddown;j1--)
  {
  masku[1].chrom[k] = masku[1].chrom[k]<<1;
  if(mask_record[j1])
  masku[1].chrom[k] = masku[1].chrom[k]|0x1;
  }
  enddown+=16;
  }
  for(k=0;k<=(chromsize-1);k++)
  {
  child1[k]=((masku[1].chrom[k] & parent1[k]) | parent2
[k]&~(masku[1].chrom[k]));
  child2[k]=((masku[1].chrom[k] & parent2[k]) | parent1
[k]&~(masku[1].chrom[k]));
  }
  }
  else
  {
   for(k=0;k<chromsize; k++)
   {
    child1[k] = parent1[k];
    child2[k] = parent2[k];
   }
  }
 }
 void select_memory()
 /* used for tournament selection */
 { unsigned nbytes;
  int j;
  nbytes = popsize*sizeof(int);
  if((tourneylist = (int *) malloc(nbytes)) == NULL)
   nomemory("tourneylist");
  if(numfiles == 0)
   fprintf(outfp," Enter tournament size for selection --> ");
  fscanf(infp,"%d", &tourneysize);
  if(tourneysize > popsize)
   {
    fprintf(outfp,"FATAL: Tournament size (%d) >
popsize (%d)\n",
          tourneysize,popsize);
   exit(-1);
  }
 }
 void select_free()
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ค้า การจัดจำหน่าย อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* used for tournament selection */
{ free(tourneylist);
};
void ts_preselected()
/* used for tournament selection */
{  ts_reset();
   tourneypos = 0;
}
int ts_selected()
/* used for tournament selection */
{  int pick, winner, i;
   /* If remaining members not enough for a tournament,
then reset list */
   if((popsize - tourneypos) < tourneysize)
   {        ts_reset();
            tourneypos = 0;
   }
   /* Select tourneysize structures at random and conduct a
tournament */
   winner=tourneylist[tourneypos];
   for(i=1; i<tourneysize; i++)
   {        pick=tourneylist[i+tourneypos];
            if(oldpop[pick].fitness >
oldpop[winner].fitness) winner=pick;
   }
   /* Update tourneypos */
   tourneypos += tourneysize;
   return(winner);
}

void ts_reset()
/* Shuffles the tourneylist at random */
{  int i, rand1, rand2, temp;
   for(i=0; i<popsize; i++) tourneylist[i] = i;
   for(i=0; i < popsize; i++)
   {        rand1=rnd(i,popsize-1);
            rand2=rnd(i,popsize-1);
            temp = tourneylist[rand1];
            tourneylist[rand1]=tourneylist[rand2];
            tourneylist[rand2]=temp;
   }
}
```

2. โปรแกรม tfuncg.c

```
/* tfuncg.c : test function used Gray Code
   Last update Sep 4 , 1998
   Written by Komsan Hongesombut      */
#include<stdio.h>
#include <stdlib.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include<time.h>
#include<dos.h>
#include"c:\object.c"
#include"c:\eecon98\rand3.c"
#define LINELENGTH 80      /* width of printout */
#define BITS_PER_BYTE 8    /* number of bits per byte
on this machine */
#define UINTSIZE (BITS_PER_BYTE*sizeof(unsigned))
/* # of bits in unsigned */
void app_data();
void app_free();
void app_malloc();
void objfunc(struct individual *critter);
void generation();
void initialize();
void initdata();
void initpop();
void initreport();
void initmalloc();
void freeall();
void nomemory(char *string);
int flip(float prob);
void initrandomnormaldeviate();
double noise(double mu ,double sigma);
void randomized();
double randomnormaldeviate();
float randomperc();
int rnd(int low,int high);
float rndreal(float lo ,float hi);
void preselect_no_scaling();
void prescale(double umax,double uavg,double
umin,double *a,double *b);
double scale(double u,double a,double b);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void scalepop(double max,double avg,double min,struct
individual *pop);
void preselect_expo_scaling();
void preselect_window_scaling();
void select_memory();
void select_free();
void ts_preselected();
int selected();
int ts_selected();
void ts_reset();
void statistics(struct individual *pop);
void copyright();
void repchar (FILE *outfp,char *ch,int repcount);
void skip(FILE *outfp,int skipcount);
int ithruj2int(int i,int j,unsigned *from);
void mutation(unsigned *child);
int crossover (unsigned *parent1,unsigned *parent2,
                unsigned *child1 ,unsigned *child2);
void uniform_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2);
void two_point_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2);
double start,finish;
void report();
void writepop();
void writechrom(unsigned *chrom);
void decode_for_best(struct individual *critter);
void writechrom_to_files_bestfit(unsigned *chrom);
void print_to_file_decode_for_best(struct individual
*critter1);
FILE *outfp, *infp;
FILE *fpga;
/* Global structures and variables */
struct individual
{
  unsigned *chrom;  /* chromosome string for the
individual */
  double  fitness; /* fitness of the individual */
  int   xsite;  /* crossover site at mating */
  int   parent[2]; /* who the parents of offspring were */
  int   *utility;  /* utility field can be used as pointer to a
*/
  /* dynamically allocated, application-specific data
structure */
};
struct bestever
{
  unsigned *chrom;  /* chromosome string for the best-
ever individual */
  double  fitness; /* fitness of the best-ever individual */
  int    generation; /* generation which produced it */
};

struct individual *oldpop; /* last generation of individuals
*/
struct individual *newpop; /* next generation of
individuals */
struct individual *masku;  /* mask */
static struct bestever bestfit; /* fittest individual so far */
double sumfitness;   /* summed fitness for entire
population */
double max;        /* maximum fitness of population */
double avg;        /* average fitness of population */
double min;        /* minumum fitness of population */
float  pcross,pcross0; /* probability of crossover */
float  pmutation,pmutation0;/* probability of mutation */
int   numfiles;      /* number of open files */
int   popsize;       /* population size */
int   lchrom;        /* length of the chromosome per
individual */
int   chromsize;   /* number of bytes needed to store
lchrom string */
int   gen;       /* current generation number */
int   maxgen;     /* maximum generation number */
int   run;       /* current run number */
int   maxruns;    /* maximum number of runs to make */
int   printstrings; /* flag to print chromosome strings
(default on) */
double   nmutation; /* number of mutations  */
double   ncross;   /* number of crossovers  */
static int field_size, vec_size; /* derive from app2.c */
/* variables are declared static so that they cannot conflict
with names of  */
/* other global variables in other files.  See K&R, p 80, for
scope of static */
```

```
static double oldrand[55]; /* Array of 55 random numbers
*/
static int jrand;        /* current random number */
static double rndx2;     /* used with random normal
deviate */
static int rndcalcflag; /* used with random normal deviate
*/
struct  time t;
int hour1,hour2,
   min1,min2,
   sec1,sec2,
   sechund1,sechund2;
time_t tt;
int select_function;
int select_scaling;
int select_crossover;
double a,b;
int elitist ; /* keep best chromosome */
static int *tourneylist,tourneypos,tourneysize;
/* for tournament selection */
void main()
{
  struct individual *temp;
  FILE  *fopen();
  fpga=fopen("tfgssl6.dat","w");
  printstrings=1;
  numfiles=0;
  infp = stdin;
  outfp = stdout;
  copyright();

  if(numfiles == 0)
        fprintf(outfp," Number of GA runs to be
performed-> ");
  fscanf(infp,"%d",&maxruns);
  for(run=1; run<=maxruns; run++)
  { /* Set things up */
        initialize();
        gettime(&t);
        hour1 = t.ti_hour;
        min1  = t.ti_min;
        sec1  = t.ti_sec;
        sechund1 = t.ti_hund;
        for(gen=0; gen<maxgen; gen++)
        {
          if(gen%1==0)
          {
           fprintf(outfp,"\nRUN %d of %d:
GENERATION %d->%d\n",

run,maxruns,gen,maxgen);
          }
          /* create a new generation */
          generation();
          /* compute fitness statistics on new
populations */
          statistics(newpop);
          /* report results for new generation */
          if(gen%1==0)
          { report(); }
          /* advance the generation */
          temp = oldpop;
          oldpop = newpop;
          newpop = temp;
        }
        gettime(&t);
        hour2 = t.ti_hour;
        min2  = t.ti_min;
        sec2  = t.ti_sec;
        sechund2 = t.ti_hund;

        fprintf(fpga,"\n");
        fprintf(fpga,"Global Best Individual so far,
Generation %d:\n",
                bestfit.generation);
        print_to_file_decode_for_best(&(bestfit));
        objfunc((&bestfit));
        fprintf(fpga,"performance index = %10.15f: ",
(bestfit.fitness));
        fprintf(fpga,"\nThe start process time is:
%2d:%02d:%02d.%02d\n",
        hour1,min1,sec1,sechund1);
        printf("The stop process time is:
%2d:%02d:%02d.%02d\n",
        t.ti_hour, t.ti_min, t.ti_sec, t.ti_hund);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            fprintf(fpga,"The stop process time is:
%2d:%02d:%02d.%02d\n",
            t.ti_hour, t.ti_min, t.ti_sec, t.ti_hund);
            freeall(); fclose(fpga);
    }
}
void app_data()
/* application dependent data input, called by init_data() */
/* ask your input questions here, and put output in global
variables */
/* In this example application, the utility pointer of each
individual is  */
/* assigned a vector of integers that will be used to store
the interpreted */
/* chromosome. */
{  int size = UINTSIZE;
   if(lchrom < UINTSIZE) size = lchrom;
   /* user must specify length of concatenated integers in
the chromosome. */
   fprintf(outfp," Enter field size (must be less than %d) -
>", size);
   fscanf(infp,"%d",&field_size);
   vec_size = lchrom/field_size;
   if((((float)lchrom/(float)field_size)-vec_size) > 0.0)
vec_size++;
}
void app_free()
/* This routine should free any memory allocated */
/* in the application-dependent routines, called by freeall()
*/
{  int i;
   for(i = 0; i < popsize; i++)
   {
            free(newpop[i].utility);
            free(oldpop[i].utility);
   }
}
void app_malloc()
/* application dependent malloc() calls, called by
initmalloc() */
{  unsigned nbytes;
   int i;
   nbytes = vec_size * sizeof(int);
   for(i = 0; i < popsize; i++)
   {
            if((newpop[i].utility = (int *) malloc(nbytes))
== NULL)
               nomemory("newpop utility");
            if((oldpop[i].utility = (int *) malloc(nbytes)) ==
NULL)
               nomemory("oldpop utility");
   }
}
void app_report()
/* Application-dependent report, called by report() */
{
   int i, j;
   /* Print vector interpretation of ech chromosome. */
   for(i = 0; i < popsize; i++)
   {
      fprintf(outfp,"oldpop %d = ", i);
      for(j = 0; j < vec_size; j++)
        fprintf(outfp, "%d ", oldpop[i].utility[j]);
      fprintf(outfp,"\n");
   }
}
void preselect_no_scaling()
/* preselect in case of no scaling */
{  int j;
   sumfitness = 0;
   for(j = 0; j < popsize; j++) sumfitness += oldpop
[j].fitness;
}
void prescale(double umax,double uavg,double
umin,double *a,double *b)
/* in case of linear scaling */
{ double fmultiple = 2.0;
 double delta;
  fmultiple = 2.0;
  if(umin>((fmultiple*uavg)-umax)/(fmultiple-1)) /* non
negative test */
  { /* normal scaling */
   delta = umax-uavg;
   *a = ((fmultiple-1.0)*uavg)/delta;
   *b = uavg*(umax-fmultiple*uavg)/delta;
  }
```

สารนิพนธ์นี้เป็นลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย ใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีอื่นๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    else
    { /* scale as much as possible */
      delta = uavg-umin;
      *a = uavg/delta;
      *b = -umin*uavg/delta;
    }
  }
  double scale(double u,double a,double b)
  /* used for linear scaling */
  {
    return(fabs((a*u) + b));
  }
  void scalepop(double max,double avg,double min,struct
  individual *pop)
  /* used for linear scaling */
  { int j;
    double a=0.0,b=0.0;
    prescale(max,avg,min,&a,&b);
    sumfitness=0.0;
    for(j=0;j<popsize;j++) {
      pop[j].fitness = scale(pop[j].fitness,a,b);
      sumfitness+=pop[j].fitness;
    }
  }
  void preselect_window_scaling()
  /* used for window scaling */
  {  int j;
    sumfitness = 0;
    for(j = 0; j < popsize; j++)
    { oldpop[j].fitness -= min;
      sumfitness += oldpop[j].fitness;
    }
  }
  void preselect_expo_scaling()
  /* used for exponential scaling */
  {  int j;
    sumfitness = 0;
    for(j = 0; j < popsize; j++)
    { oldpop[j].fitness -= min;
      oldpop[j].fitness += 1.0;
      oldpop[j].fitness *= oldpop[j].fitness;
      sumfitness += oldpop[j].fitness;
    }
  }
  int selected()
  /* roulette-wheel selection */
  {  float sum, pick;
    int i;
    pick = randomperc()*sumfitness;
    sum = 0;
    if(sumfitness != 0)
    {
          for(i = 0; (sum < pick) && (i < popsize); i++)
            sum += oldpop[i].fitness;
    }
    else
          i = rnd(1,popsize);
    return(i-1);
  }
  void generation()
  {
    int mate1, mate2, jcross, j = 0;
    int k;
    /* perform any preselection actions necessary before
  generation */
    if(select_scaling == 1) /* no scaling */
    {
      preselect_no_scaling();
    }
    if(select_scaling == 2) /* linear scaling */
    {  prescale(max,avg,min,&a,&b);
      scalepop(max,avg,min,oldpop);
    }
    if(select_scaling == 3)
    {
      preselect_window_scaling();
    }
    if(select_scaling == 4)
    {
      preselect_expo_scaling();
    }
    if(select_scaling == 5)
    {
    ts_preselected();
    }
    pcross = pcross0; pmutation = pmutation0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* select, crossover, and mutation */
do
 {
  /* pick a pair of mates */
  if(select_scaling == 5)
  {
        mate1=ts_selected();
        mate2=ts_selected();
  }
  else
  {
        mate1=selected();
        mate2=selected();
  }
  /* Crossover and mutation */
  if(select_crossover==1)
  {
       jcross = crossover(oldpop[mate1].chrom,
oldpop[mate2].chrom,newpop[j].chrom, newpop
[j+1].chrom);
  }
  else
  if(select_crossover==2)
  {
       two_point_crossover(oldpop[mate1].chrom,
oldpop[mate2].chrom,
                           newpop[j].chrom,
newpop[j+1].chrom);
  }
  else
  if(select_crossover==3)
  {
   uniform_crossover(oldpop[mate1].chrom, oldpop
[mate2].chrom,
                           newpop[j].chrom,
newpop[j+1].chrom);
  }
  mutation(newpop[j].chrom);
  mutation(newpop[j+1].chrom);
  pcross -= (float)((pcross0-0.6)/maxgen);
  pmutation += (float)((0.1-pmutation0)/maxgen);
  /* Decode string, evaluate fitness, & record */
  /* parentage date on both children */
   objfunc(&(newpop[j]));
   newpop[j].parent[0] = mate1+1;
   newpop[j].xsite = jcross;
   newpop[j].parent[1] = mate2+1;
   objfunc(&(newpop[j+1]));
   newpop[j+1].parent[0] = mate1+1;
   newpop[j+1].xsite = jcross;
   newpop[j+1].parent[1] = mate2+1;
   /* Increment population index */
   j = j + 2;
 }
 while(j < (popsize-1));
 statistics(newpop);
 if(elitist == 1)
 {
   for(k = 0; k < chromsize; k++)
     {
       newpop[0].chrom[k] = bestfit.chrom[k];
     }
   objfunc(&(newpop[0]));
 }
}
void initialize()
/* Initialization Coordinator */
{  /* get basic problem values from input file */
  initdata();
/* define chromosome size in terms of machine bytes, ie
*/
/* length of chromosome in bits (lchrom)/(bits-per-byte) */
/* chromsize must be known for malloc() of chrom pointer
*/
  chromsize = (lchrom/UINTSIZE);
  if(lchrom%UINTSIZE) chromsize++;
  /* malloc space for global data structures */
  initmalloc();
  /* initialize random number generator */
  randomized();
  /* initialize global counters/values */
  nmutation = 0;
  ncross = 0;
  bestfit.fitness = 0.0;
  bestfit.generation = 0;
  /* initialize the populations and report statistics */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    initpop();
    statistics(oldpop);
    initreport();
  }
  void initdata()
  /* data inquiry and setup */
  {  char  answer[2];
    if(numfiles == 0)
    {       fprintf(outfp,"\n ------- GA Data Entry and
  Initialization -------\n");
            fprintf(outfp," Select one funtion to optimize :
  \n");
            fprintf(outfp," 1. f(x,y) = x^2 + 2y^2 - 0.3cos
  (3px) - 0.4cos(4py) + 0.7 \n");
            fprintf(outfp," 2. f(x,y) = x^2 + 2y^2 - 0.3[cos
  (3px)cos(4py)] + 0.3 \n");
            fprintf(outfp," 3. f(x,y) = x^2 + 2y^2 - 0.3[cos
  (3px) + cos(4py)] + 0.3 \n");
            fprintf(outfp," 4. f(x,y) = 1/((x+0.5)^2 + 2(y-
  0.5)^2 - 0.3cos(3px) - 0.4cos(4py) + 0.8) \n");
            fprintf(outfp," Select ----> ");
    }
    fscanf(infp,"%d",&select_function);
    if(numfiles == 0)
    {
            fprintf(outfp," Enter the population size --------
  ---> ");
    }
    fscanf(infp,"%d", &popsize);

    if((popsize%2) != 0)
     {
            fprintf(outfp, "Sorry! only even population
  sizes are allowed. \n Incrementing popsize by one.\n");
            popsize++;
     };
    if(numfiles == 0)
            fprintf(outfp," Enter chromosome length (56) -
  -------> ");
    fscanf(infp,"%d", &lchrom);
    if(numfiles == 0)
            fprintf(outfp," Print chromosome strings? (y/n)
  ------> ");
    fscanf(infp,"%s",answer);
    if(strncmp(answer,"n",1) == 0) printstrings = 0;
    if(numfiles == 0)
    {       fprintf(outfp," Enter maximum number of
  generations --> ");
            fprintf(fpga," Enter maximum number of
  generations --> ");
    }
    fscanf(infp,"%d", &maxgen);
    fprintf(fpga,"%d \n",maxgen);
    if(numfiles == 0)
    {  fprintf(outfp," Select one crossover method :\n");
            fprintf(outfp," 1. One-point crossover \n");
            fprintf(outfp," 2. Two-point crossover \n");
            fprintf(outfp," 3. Uniform crossover \n");
            fprintf(outfp," Select ----> ");
            fprintf(fpga," Select one crossover method
  :\n");
            fprintf(fpga," 1. One-point crossover \n");
            fprintf(fpga," 2. Two-point crossover \n");
            fprintf(fpga," 3. Uniform crossover \n");
            fprintf(fpga," Select ----> ");
    }
            fscanf(infp,"%d",&select_crossover);
            fprintf(fpga,"%d \n",select_crossover);
    if(numfiles == 0)
    {       fprintf(outfp," Enter initial crossover
  probability ----------> ");
            fprintf(fpga," Enter initial crossover probability
  ----------> ");
    fscanf(infp,"%f", &pcross0);
    fprintf(fpga,"%f \n",pcross0);
    }

    if(numfiles == 0)
    {       fprintf(outfp," Enter initial mutation probability
  ----------> ");
            fprintf(fpga," Enter initial mutation probability
  ----------> ");
    fscanf(infp,"%f", &pmutation0);
    fprintf(fpga,"%f \n",pmutation0);
    }
    if(numfiles == 0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    {
            fprintf(outfp," Select one scaling method : \n");
            fprintf(outfp," 1. No scaling \n");
            fprintf(outfp," 2. Linear scaling method \n");
            fprintf(outfp," 3. Window scaling method \n");
            fprintf(outfp," 4. Exponential scaling method
 \n");
            fprintf(outfp," 5. Tournament selection method
 \n");
            fprintf(outfp," Select ----> ");
            fprintf(fpga," Select one scaling method : \n");
            fprintf(fpga," 1. No scaling \n");
            fprintf(fpga," 2. Linear scaling method \n");
            fprintf(fpga," 3. Window scaling method \n");
            fprintf(fpga," 4. Exponential scaling method
 \n");
            fprintf(fpga," 5. Tournament selection method
 \n");
            fprintf(fpga," Select ----> ");
    }
    fscanf(infp,"%d",&select_scaling);
    fprintf(fpga,"%d \n",select_scaling);
    if(numfiles == 0)
    {       fprintf(outfp," Enter Elitist Selection (1:Y | 0:N)
--> ");
            fprintf(fpga," Enter Elitist Selection (1:Y | 0:N)
--> ");
    fscanf(infp,"%d",&elitist);
    fprintf(fpga,"%d \n",elitist);
    }
    app_data();
}
void initpop()
/* Initialize a population at random */
{  int j, j1, k, stop;
   unsigned mask = 1;
   for(j = 0; j < popsize; j++)
   {
           for(k = 0; k < chromsize; k++)
           {
            oldpop[j].chrom[k] = 0;
            if(k == (chromsize-1))
                    stop = lchrom - (k*UINTSIZE);
            else
                    stop = UINTSIZE;
            /* A fair coin toss */
            for(j1 = 1; j1 <= stop; j1++)
            {
              oldpop[j].chrom[k] = oldpop[j].chrom[k]
<<1;
              if(flip(0.5))
                      oldpop[j].chrom[k] = oldpop
[j].chrom[k]|mask;
            }
           }
           oldpop[j].parent[0] = 0; /* Initialize parent info.
*/
           oldpop[j].parent[1] = 0;
           oldpop[j].xsite = 0;
           objfunc(&(oldpop[j])); /* Evaluate initial
fitness */
    }
}
void initreport()
/* Initial report */
{  void  skip();
   skip(outfp,1);
   fprintf(outfp," GA Parameters\n");
   fprintf(outfp," ----------------------------------------
\n");
   fprintf(outfp," Total Population size          =
%d\n",popsize);
   fprintf(outfp," Chromosome length (lchrom)     =
%d\n",lchrom);
   fprintf(outfp," Maximum # of generations (maxgen) =
%d\n",maxgen);
   fprintf(outfp," Initial crossover probability (pcross)   =
%f\n", pcross0);
   fprintf(outfp," Initial mutation  probability (pmutation)
= %f\n", pmutation0);
   skip(outfp,1);
   fprintf(fpga," GA Parameters\n");
   fprintf(fpga," ----------------------------------------
\n");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    fprintf(fpga," Total Population size          =
%d\n",popsize);
    fprintf(fpga," Chromosome length (lchrom)      =
%d\n",lchrom);
    fprintf(fpga," Maximum # of generations (maxgen) =
%d\n",maxgen);
    fprintf(fpga," Initial crossover probability (pcross)   =
%f\n", pcross0);
    fprintf(fpga," Initial mutation  probability (pmutation) =
%f\n", pmutation0);
    fprintf(fpga," ----------------------------------------------
\n\n");
    fprintf(fpga," min      max      avg      fitness
best.fitness     chromosome \n\n");

    fflush(outfp);
}
void initmalloc()
/* memory allocation of space for global data structures */
{ unsigned nbytes;
 int j;
 /* memory for old and new populations of individuals */
 nbytes = popsize*sizeof(struct individual);
 if((oldpop = (struct individual *) malloc(nbytes)) ==
NULL)
   nomemory("oldpop");
 if((newpop = (struct individual *) malloc(nbytes)) ==
NULL)
   nomemory("newpop");
 if((masku = (struct individual *) malloc(nbytes)) ==
NULL)
   nomemory("masku");
 /* memory for chromosome strings in populations */
 nbytes = chromsize*sizeof(unsigned);
 for(j = 0; j < popsize; j++)
  {
   if((oldpop[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
        nomemory("oldpop chromosomes");
   if((newpop[j].chrom = (unsigned *) malloc(nbytes))
== NULL)
        nomemory("newpop chromosomes");
   if((masku[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
        nomemory("masku chromosomes");
  }
 if((bestfit.chrom = (unsigned *) malloc(nbytes)) ==
NULL)
  nomemory("bestfit chromosome");
 if(select_scaling == 5)
 {
  select_memory();
 }
 /* call to application-specific malloc() routines  */
 /* can be used to malloc memory for utility pointer */
 app_malloc();
}
void freeall()
/* A routine to free all the space dynamically allocated in
initspace() */
{ int i;
 for(i = 0; i < popsize; i++)
  {
   free(oldpop[i].chrom);
   free(newpop[i].chrom);
   free(masku[i].chrom);
  }
 free(oldpop);
 free(newpop);
 free(masku);
 free(bestfit.chrom);
 select_free();
 app_free();
}
void nomemory(char *string)
{
 fprintf(stderr,"malloc: out of memory making
%s!!\n",string);
 exit(-1);
}
int flip(float prob)
/* Flip a biased coin - true if heads */
{ if(randomperc() <= prob)
        return(1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    else
            return(0);
  }
  void initrandomnormaldeviate()
  /* initialization routine for randomnormaldeviate */
  { rndcalcflag = 1;
  }
  double noise(double mu ,double sigma)
  /* normal noise with specified mean & std dev: mu &
  sigma */
  /*double mu, sigma;*/
  { double randomnormaldeviate();
    return((randomnormaldeviate()*sigma) + mu);
  }
  void randomized()
  /* Get seed number for random and start it up */
  { sran2((unsigned int) time(&tt));
  }
  double randomnormaldeviate()
  /* random normal deviate after ACM algorithm 267 / Box-
  Muller Method */
  { double t, rndx1;
    if(rndcalcflag)
    {       rndx1 = sqrt(- 2.0*log((double) randomperc
  ()));
            t = 6.2831853072 * (double) randomperc();
            rndx2 = rndx1 * sin(t);
            rndcalcflag = 0;
            return(rndx1 * cos(t));
    }
    else
    {       rndcalcflag = 1;
            return(rndx2);
    }
  }
  float randomperc()
  /* Fetch a single random number between 0.0 and 1.0 */
  { return(ran2());
  }
  int rnd(int low,int high)
  /* Pick a random integer between low and high */
  { int i;
    if(low >= high)
            i = low;
    else
    {       i = (randomperc() * (high - low + 1)) + low;
            if(i > high) i = high;
    }
    return(i);
  }
  float rndreal(float lo ,float hi)
  /* real random number between specified limits */
  {
    return((randomperc() * (hi - lo)) + lo);
  }
  void statistics(struct individual *pop)
  /* Calculate population statistics */
  { int i, j;
    sumfitness = 0.0;
    min = pop[0].fitness;
    max = pop[0].fitness;
    /* Loop for max, min, sumfitness */
    for(j = 0; j < popsize; j++)
    { sumfitness = sumfitness + pop[j].fitness; /*
  Accumulate */
            if(pop[j].fitness > max) max = pop[j].fitness; /*
  New maximum */
            if(pop[j].fitness < min) min = pop[j].fitness; /*
  New minimum */
            /* new global best-fit individual */
            if(pop[j].fitness > bestfit.fitness)
            { for(i = 0; i < chromsize; i++)
              bestfit.chrom[i]    = pop[j].chrom[i];
              bestfit.fitness     = pop[j].fitness;
              bestfit.generation = gen;
            }
    }
    /* Calculate average */
    avg = sumfitness/popsize;
  }
  void copyright()
  { int iskip;
    int ll = 59;
    iskip = (LINELENGTH - ll)/2;
    skip(outfp,1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    repchar(outfp," ",iskip); repchar(outfp,"-",ll); skip
  (outfp,1);
    repchar(outfp," ",iskip-1);
    fprintf(outfp,"|      GA-C (v1.1) - A Simple Genetic
  Algorithm        |\n");
    repchar(outfp," ",iskip);
    repchar(outfp,"-",ll); skip(outfp,2);
  }
  void repchar (FILE *outfp,char *ch,int repcount)
  /* Repeatedly write a character to stdout */
  {  int j;
    for (j = 1; j <= repcount; j++) fprintf(outfp,"%s", ch);
  }
  void skip(FILE *outfp,int skipcount)
  /* Skip skipcount lines */
  {  int j;
    for (j = 1; j <= skipcount; j++) fprintf(outfp,"\n");
  }
  int ithruj2int(int i,int j,unsigned *from)
  /* interpret bits i thru j of a individual as an integer    */
  /* j MUST BE greater than or equal to i AND j-i <
  UINTSIZE-1  */
  /* from is a chromosome, represented as an array of
  unsigneds */
  {  unsigned mask, temp;
    int bound_flag;
    int iisin, jisin;
    int i1, j1, out;
    if(j < i)
    {
          fprintf(stderr,"Error in ithruj2int: j < i\n");
          exit(-1);
    }
    if(j-i+1 > UINTSIZE)
    {
          fprintf(stderr,"Error in ithruj2int: j-i+1 >
UINTSIZE\n");
          exit(-1);
    }
    iisin = i/UINTSIZE;
    jisin = j/UINTSIZE;
    i1 = i - (iisin*UINTSIZE);
    j1 = j - (jisin*UINTSIZE);
          if(i1 == 0)
          {  iisin = iisin-1;
            i1 = i - (iisin*UINTSIZE);
          };
          if(j1 == 0)
          {  jisin = jisin-1;
            j1 = j - (jisin*UINTSIZE);
          };
    /* check if bits fall across a word boundary */
    if(iisin == jisin)
          bound_flag = 0;
    else
          bound_flag = 1;
    if(bound_flag == 0)
    {     mask = 1;
          mask = (mask<<(j1-i1+1))-1;
          mask = mask<<(i1-1);
          out = (from[iisin]&mask)>>(i1-1);
          return(out);
    }
    else
    {     mask = 1;
          mask = (mask<<j1)-1;
          temp = from[jisin]&mask;
          mask = 1;
          mask = (mask<<(UINTSIZE-i1+1))-1;
          mask = mask<<(i1-1);
          out = ((from[iisin]&mask)>>(i1-1)) | (temp
<<(UINTSIZE-i1+1));
          return(out);
    }
  }
  void mutation(unsigned *child)
  /* Mutate an allele w/ pmutation, count # of mutations */
  {  int j, k, stop;
    unsigned mask, temp = 1;
    for(k = 0; k < chromsize; k++)
    {     mask = 0;
          if(k == (chromsize-1))
            stop = lchrom - (k*UINTSIZE);
          else
            stop = UINTSIZE;
          for(j = 0; j < stop; j++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
              if(flip(pmutation))
              {
                    mask = mask|(temp<<j);
                    nmutation++;
              }
            }
            child[k] = child[k]^mask;
    }
  }
  int crossover (unsigned *parent1,unsigned *parent2,
                 unsigned *child1 ,unsigned *child2)
  /* Cross 2 parent strings, place in 2 child strings */
  {  int j, jcross, k;
     unsigned mask, temp;
     /* Do crossover with probability pcross */
     if(flip(pcross))
     {       jcross = rnd(1 ,(lchrom - 1));/* Cross between 1
and l-1 */
            ncross++;
            for(k = 1; k <= chromsize; k++)
            {
              if(jcross >= (k*UINTSIZE))
              {
                    child1[k-1] = parent1[k-1];
                    child2[k-1] = parent2[k-1];
              }
              else if((jcross < (k*UINTSIZE)) && (jcross
> ((k-1)*UINTSIZE)))
              {
                    mask = 1;
                    for(j = 1; j <= (jcross-1-((k-
1)*UINTSIZE)); j++)
                    {
                      temp = 1;
                      mask = mask<<1;
                      mask = mask|temp;
                    }
                    child1[k-1] =
(parent1[k-1]&mask)|(parent2[k-1]&(~mask));
                    child2[k-1] =
(parent1[k-1]&(~mask))|(parent2[k-1]&mask);
              }
              else
              {
                    child1[k-1] = parent2[k-1];
                    child2[k-1] = parent1[k-1];
              }
            }
     }
     else
     {
            for(k = 0; k < chromsize; k++)
            {
              child1[k] = parent1[k];
              child2[k] = parent2[k];
            }
            jcross = 0;
     }
     return(jcross);
  }
  void report()
  /* Write the population report */
  {  repchar(outfp,"-",LINELENGTH);
     skip(outfp,1);
     if(printstrings == 1)
     {       repchar(outfp," ",((LINELENGTH-17)/2));
             fprintf(outfp,"Population Report\n");
             fprintf(outfp, "Generation %3d", gen);
             repchar(outfp," ",(LINELENGTH-28));
             fprintf(outfp, "Generation %3d\n", (gen+1));
             fprintf(outfp,"num   string ");
             repchar(outfp," ",lchrom-5);
             fprintf(outfp,"fitness    parents xsite  ");
             fprintf(outfp,"string ");
             repchar(outfp," ",lchrom-5);
             fprintf(outfp,"fitness\n");
             repchar(outfp,"-",LINELENGTH);
             skip(outfp,1);
             writepop();
             repchar(outfp,"-",LINELENGTH);
             skip(outfp,1);
     }
     /* write the summary statistics in global mode  */
     fprintf(outfp,"Generation %d Accumulated Statistics:
\n", gen);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    fprintf(outfp,"Total Crossovers = %f, Total Mutations =
  %f\n",ncross,nmutation);
    fprintf(outfp,"min = %10.9f  max = %10.9f  avg =
  %10.9f  sum = %10.9f\n",min,max,avg,sumfitness);
    fprintf(fpga," %10.9f  %10.9f  %10.9f  %10.9f
  %10.15f  ",min,max,avg,sumfitness,bestfit.fitness);
    fprintf(outfp,"Global Best Individual so far, Generation
  %d:\n",bestfit.generation);
    fprintf(outfp,"Fitness = %f: ", bestfit.fitness);
    writechrom((&bestfit)->chrom);fprintf(fpga,"\n");
    skip(outfp,1);
    decode_for_best((&bestfit));
    objfunc((&bestfit));
    fprintf(outfp,"performance index = %10.15f: ",
  (bestfit.fitness));
    skip(outfp,1);
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
  }
  void writepop()
  {  struct individual *pind;
    int j;
    for(j=0; j<=popsize-1; j++)
    {       fprintf(outfp,"%3d)  ",j+1);
            /* Old string */
            pind = &(oldpop[j]);
            writechrom(pind->chrom);
            fprintf(outfp,"  %8f | ", pind->fitness);
            /* New string */
            pind = &(newpop[j]);
            fprintf(outfp,"(%2d,%2d)  %2d  ",
            pind->parent[0], pind->parent[1], pind->xsite);
            writechrom(pind->chrom);
            fprintf(outfp,"  %8f\n", pind->fitness);
    }
  }
  void writechrom(unsigned *chrom)
  /* Write a chromosome as a string of ones and zeroes
  */
  {  int j, k, stop;
    unsigned mask=1,tmp;
    static int bit_count=1;
    unsigned mask1;
    int m;
    bit_count=1;
    for(k = chromsize-1; k>=0; k--)
    {
            tmp = chrom[k];
            if(k == (chromsize-1))
              { stop = lchrom - (k*UINTSIZE);m=0x1
  <<(stop-1);}
            else
              { stop = UINTSIZE;m=0x1<<(stop-1);}
            mask1=m;
            for(j = 0; j < stop; j++)
            {
              if(tmp&mask1)
                    fprintf(outfp,"1");
              else
                    fprintf(outfp,"0");
              mask1>>=1;
            }
    }
  }
  void objfunc(struct individual *critter)
  /* objective function  */
  {  unsigned mask=0x1;  /* mask for current bit */
    int bitpos;  /* current bit position */
    unsigned bitpos2=0,bitpos3=0;
    unsigned tp;
    double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
    int i,j, k, stop;
    int n = 10;
    double xx,yy;
    static int gray[56],bcd[56];
    xx=0.0; yy=0.0;
    bitpow2=0; bitpow3=0;
    critter->fitness = 0.0;
    /* loop thru number of bytes holding chromosome */
    for(k = 0; k <= chromsize-1; k++)
          {
          if(k == (chromsize-1))
                {stop = lchrom-(k*UINTSIZE);}
          else
                {stop = UINTSIZE;}
          /* loop thru bits in current byte */
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
tp = critter->chrom[k];
for(j = 0; j < stop; j++)
{       bitpos = j + UINTSIZE*k;
        /* test for current bit 0 or 1 */
        if((bitpos>=0)&&(bitpos<=27))
        {
                if((tp&mask) == 1)
                {
                 gray[(int)bitpos]=1;
                }
                else
                gray[(int)bitpos]=0;
        }
        if((bitpos>=28)&&(bitpos<=55))
        {
                if((tp&mask) == 1)
                {
                gray[(int)bitpos]=1;
                }
                else
                gray[(int)bitpos]=0;
        }
       tp = tp>>=1;
}
}
bcd[27]=gray[27];
for(i=26;i>=0;i--)
bcd[i]=bcd[i+1]^gray[i];

bcd[55]=gray[55];
for(i=54;i>=28;i--)
bcd[i]=bcd[i+1]^gray[i];

for(i=0;i<=27;i++)
{
xx += bcd[i]*pow(2.0,(double)i);
}
for(i=28;i<=55;i++)
{
yy += bcd[i]*pow(2.0,(double)(i-28));
}
xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
  critter->fitness =
performance_index(xx,yy,select_function);
}
void decode_for_best(struct individual *critter1)
{  unsigned mask=0x1;  /* mask for current bit */
   int bitpos;  /* current bit position */
   unsigned bitpos2=0,bitpos3=0;
   unsigned tp;
   double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
   int i,j, k, stop;
   int n = 10;
   double xx,yy;
   static int gray[56],bcd[56];
   xx=0.0; yy=0.0;
   bitpow2=0; bitpow3=0;
   critter1->fitness = 0.0;
   /* loop thru number of bytes holding chromosome */
   for(k = 0; k <= chromsize-1; k++)
     {
     if(k == (chromsize-1))
          {stop = lchrom-(k*UINTSIZE);}
     else
          {stop = UINTSIZE;}
     /* loop thru bits in current byte */
     tp = critter1->chrom[k];
     for(j = 0; j < stop; j++)
     {
          bitpos = j + UINTSIZE*k;
          /* test for current bit 0 or 1 */
          if((bitpos>=0)&&(bitpos<=27))
          {
                if((tp&mask) == 1)
                {
                 gray[(int)bitpos]=1;
                }
                else
                gray[(int)bitpos]=0;
          }
          if((bitpos>=28)&&(bitpos<=55))
          {
                if((tp&mask) == 1)
                {
                 gray[(int)bitpos]=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    }
                    else
                    gray[(int)bitpos]=0;
                }
                tp = tp>>=1;
            }
            }
            bcd[27]=gray[27];
            for(i=26;i>=0;i--)
            bcd[i]=bcd[i+1]^gray[i];

            bcd[55]=gray[55];
            for(i=54;i>=28;i--)
            bcd[i]=bcd[i+1]^gray[i];

            for(i=0;i<=27;i++)
            {
            xx += bcd[i]*pow(2.0,(double)i);
            }
            for(i=28;i<=55;i++)
            {
            yy += bcd[i]*pow(2.0,(double)(i-28));
            }
   xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
   yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
   critter1->fitness =
performance_index(xx,yy,select_function);
   fprintf(outfp,"x = %10.12e   y = %10.12e  \n",xx,yy);
}
void writechrom_to_files_bestfit(unsigned *chrom)
/* Write a chromosome as a string of ones and zeroes  */
{  int j, k, stop;
   unsigned mask=1,tmp;
   static int bit_count=1;
   unsigned mask1;
   int m;
   bit_count=1;
   for(k = chromsize-1; k>=0; k--)
   {
        tmp = chrom[k];
        if(k == (chromsize-1))
          { stop = lchrom - (k*UINTSIZE);m=0x1
<<(stop-1);}
        else
          { stop = UINTSIZE;m=0x1<<(stop-1);}
        mask1=m;
        for(j = 0; j < stop; j++)
        {
          if(tmp&mask1)
                fprintf(fpga,"1");
          else
                fprintf(fpga,"0");
          mask1>>=1;
        }
   }
}
void print_to_file_decode_for_best(struct individual
*critter1)
{  unsigned mask=0x1;  /* mask for current bit */
   int bitpos;  /* current bit position */
   unsigned bitpos2=0,bitpos3=0;
   unsigned tp;
   double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
   int i,j, k, stop;
   int n = 10;
   double xx,yy;
   static int gray[56],bcd[56];
   xx=0.0; yy=0.0;
   bitpow2=0; bitpow3=0;
   critter1->fitness = 0.0;
   /* loop thru number of bytes holding chromosome */
   for(k = 0; k <= chromsize-1; k++)
        {
      if(k == (chromsize-1))
                {stop = lchrom-(k*UINTSIZE);}
      else
                {stop = UINTSIZE;}
      /* loop thru bits in current byte */
      tp = critter1->chrom[k];
      for(j = 0; j < stop; j++)
      {
        bitpos = j + UINTSIZE*k;
        /* test for current bit 0 or 1 */
        if((bitpos>=0)&&(bitpos<=27))
        {
                if((tp&mask) == 1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                {
                 gray[(int)bitpos]=1;
                }
                else
                gray[(int)bitpos]=0;
            }
            if((bitpos>=28)&&(bitpos<=55))
            {
                if((tp&mask) == 1)
                {
                 gray[(int)bitpos]=1;
                }
                else
                gray[(int)bitpos]=0;
            }
            tp = tp>>=1;
        }
        }
        bcd[27]=gray[27];
        for(i=26;i>=0;i--)
        bcd[i]=bcd[i+1]^gray[i];

        bcd[55]=gray[55];
        for(i=54;i>=28;i--)
        bcd[i]=bcd[i+1]^gray[i];

        for(i=0;i<=27;i++)
        {
        xx += bcd[i]*pow(2.0,(double)i);
        }
        for(i=28;i<=55;i++)
        {
        yy += bcd[i]*pow(2.0,(double)(i-28));
        }
    xx = -1.0 + (xx*(1-(-1))/(pow(2.0,28.0)-1)) ;
    yy = -1.0 + (yy*(1-(-1))/(pow(2.0,28.0)-1)) ;
    critter1->fitness =
performance_index(xx,yy,select_function);
    fprintf(fpga,"x = %10.12e   y = %10.12e \n",xx,yy);
}
void uniform_crossover(unsigned *parent1,unsigned
*parent2
                                    ,unsigned *child1 ,unsigned
*child2)
{ int i, j, b,l, which_way, data_width;
 int j1, stop;
 unsigned  mask1;
 int cp1,cp2;
 int m;
 int k;
 struct individual *idp;
 int fraction;
 if (flip(pcross))
 {
 ncross++;
 for(k = 0; k < chromsize; k++)
        {
         masku[1].chrom[k] = 0;
         if(k == (chromsize-1))
                stop = lchrom - (k*UINTSIZE);
         else
                stop = UINTSIZE;

         /* A fair coin toss */
         for(j1 = 1; j1 <= stop; j1++)
         {
          masku[1].chrom[k] = masku[1].chrom[k]
<<1;
           if (flip(0.5))/*((float)rand()/(float)
RAND_MAX < 0.5)*/
                masku[1].chrom[k] = masku
[1].chrom[k]|0x1;
          }
        }
 k=(chromsize-1);
 fraction = lchrom-((chromsize-1)*16);
 m=0x1 << (fraction-1);
 mask1=m;
 child1[k]=0x0;
 child2[k]=0x0;
 for(i=1;i<=fraction;i++)
 { cp1=(mask1&(parent1[k]))>>(fraction-i);
  cp2=(mask1&(parent2[k]))>>(fraction-i);
  if ((masku[1].chrom[k]&mask1)>>(fraction-i))
  { child1[k]<<=1; child1[k]=(child1[k]|cp2);
```

สารนิพนธ์นี้เป็นลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

```
    child2[k]<<=1; child2[k]=(child2[k]|cp1);

   }
   else  { child1[k]<<=1; child1[k]=(child1[k]|cp1);
           child2[k]<<=1; child2[k]=(child2[k]|cp2);
           }
  mask1>>=1;
  }
  l=(k-1);
  for(k=l;k>=0;k--)
  {
  m=0x1 << 15;
  mask1=m;
  child1[k]=0x0;
  child2[k]=0x0;
  for(i=1;i<=16;i++)
  { cp1=(mask1&(parent1[k]))>>(16-i);
   cp2=(mask1&(parent2[k]))>>(16-i);
   if ((masku[1].chrom[k]&mask1)>>(16-i))
   { child1[k]<<=1; child1[k]=(child1[k]|cp2);
     child2[k]<<=1; child2[k]=(child2[k]|cp1);

   }
   else  { child1[k]<<=1; child1[k]=(child1[k]|cp1);
           child2[k]<<=1; child2[k]=(child2[k]|cp2);
           }
  mask1>>=1;
  }
  }
  }
 else
 {
  for(k=0;k<chromsize; k++)
  {
   child1[k] = parent1[k];
   child2[k] = parent2[k];
  }
 }
}
void two_point_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2)
{ int i, j, b,l, which_way, data_width;
 int  j1,max,min,stop,endtop,enddown;
 unsigned  mask1;
 int m,k,p1,p2;
 struct individual *idp;
 struct individual *childpop;
 unsigned mask_record[100];
 if(flip(pcross))
 { ncross++;
 /* create 2-point mask */
 p1=rnd(1,lchrom-1); p2=rnd(1,lchrom-1); /* pick 2 point
*/
 if (p1>p2) { max=p1; min=p2;}   /* p1 locate up and p2
locate down */
 else
 if (p2>p1) { max=p2; min=p1;}   /* p2 locate up and p1
locate down */
 else {max=min=p1=p2;}         /* same point */
 if (max!=min)                 /* swap chromosome for mask
*/
 {
 for(i=1;i<=min;i++)
  mask_record[i]=1;
 for(i=min+1;i<=max;i++)
  mask_record[i]=0;
 for(i=max+1;i<=lchrom;i++)
  mask_record[i]=1;
 }
 else                          /* swap chromosome in case
                                                    crossover
site is the same */
 {
 for(i=1;i<=max;i++)
  mask_record[i]=0;
 for(i=max+1;i<=lchrom;i++)
  mask_record[i]=1;
 }
 endtop=0; enddown=1;
 for(k = 0; k < chromsize; k++)
 {
 masku[1].chrom[k] =0x0;                    }
 if(k == (chromsize-1))
 stop = lchrom - (k*UINTSIZE);
 else
 stop = UINTSIZE;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  endtop+=stop;
  for(j1=endtop;j1>=enddown;j1--)
  {
  masku[1].chrom[k] = masku[1].chrom[k]<<1;
  if(mask_record[j1])
  masku[1].chrom[k] = masku[1].chrom[k]|0x1;

  }
  enddown+=16;
  }

  for(k=0;k<=(chromsize-1);k++)
  {
  child1[k]=((masku[1].chrom[k] & parent1[k]) | parent2
[k]&~(masku[1].chrom[k]));
  child2[k]=((masku[1].chrom[k] & parent2[k]) | parent1
[k]&~(masku[1].chrom[k]));
  }
  }
  else
  {
   for(k=0;k<chromsize; k++)
   {
    child1[k] = parent1[k];
    child2[k] = parent2[k];
   }
  }
}
void select_memory()
/* used for tournament selection */
{ unsigned nbytes;
 int j;
 nbytes = popsize*sizeof(int);
 if((tourneylist = (int *) malloc(nbytes)) == NULL)
  nomemory("tourneylist");
 if(numfiles == 0)
  fprintf(outfp," Enter tournament size for selection --> ");
 fscanf(infp,"%d", &tourneysize);
 fprintf(fpga," Enter tournament size for selection --> %d
\n",tourneysize );
 if(tourneysize > popsize)
  {
   fprintf(outfp,"FATAL: Tournament size (%d) >
popsize (%d)\n",
            tourneysize,popsize);
   exit(-1);
  }
}
void select_free()
{ free(tourneylist);
};
void ts_preselected()
{  ts_reset();
  tourneypos = 0;
}
int ts_selected()
{  int pick, winner, i;
  /* If remaining members not enough for a tournament,
then reset list */
  if((popsize - tourneypos) < tourneysize)
  {      ts_reset();
         tourneypos = 0;
  }
  /* Select tourneysize structures at random and conduct a
tournament */
  winner=tourneylist[tourneypos];
  for(i=1; i<tourneysize; i++)
  {      pick=tourneylist[i+tourneypos];
         if(oldpop[pick].fitness >
oldpop[winner].fitness) winner=pick;
  }
  /* Update tourneypos */
  tourneypos += tourneysize;
  return(winner);
}
void ts_reset()
/* Shuffles the tourneylist at random */
{  int i, rand1, rand2, temp;
  for(i=0; i<popsize; i++) tourneylist[i] = i;
  for(i=0; i < popsize; i++)
  {  rand1=rnd(i,popsize-1);
         rand2=rnd(i,popsize-1);
         temp = tourneylist[rand1];
         tourneylist[rand1]=tourneylist[rand2];
         tourneylist[rand2]=temp;
```

สารนิพนธ์นี้เป็นลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีอื่นๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    }
}
```

---

3. โปรแกรมย่อย object.c

```
/* object.c : objective function for test function
 Last update Sep 4 , 1998
 Written by Komsan Hongesombut  */
#include<math.h>
#include<conio.h>
double performance_index(double xx,double yy,int
        select_function);
double performance_index(double xx,double yy,int
        select_function)
{
 double function_value;
 switch(select_function) {
 case 1 : function_value = 1.0-((pow(xx,2.0)) + (2.0*pow
(yy,2.0)) - (0.3*cos(3.0*M_PI*xx)) - (0.4*cos
(4.0*M_PI*yy)) + 0.7);break;
 case 2 : function_value = 1.0-((pow(xx,2.0)) + (2.0*pow
(yy,2.0)) - (0.3*(cos(3.0*M_PI*xx)*cos(4.0*M_PI*yy)))
+ 0.3) ;break;
 case 3 : function_value = 1.0-((pow(xx,2.0)) + (2.0*pow
(yy,2.0)) - (0.3*(cos(3.0*M_PI*xx) + cos(4.0*M_PI*yy)))
+ 0.3);break;
 case 4 : function_value = 1.0/( (pow((xx+0.5),2.0))+
(2.0*pow((yy-0.5),2.0))-(0.3*cos(3.0*M_PI*xx))-(0.4*cos
(4.0*M_PI*yy))+0.8);break;
 }
 return(function_value);
}
```

---

4. โปรแกรมย่อย rand3.c

```
/* rand3.c : random method
 Last update Sep 4 , 1998
 Written by Komsan Hongesombut  */
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
/* The ran2 pseudo-random number generator.  It is *not*
 linear congruential. */
 #define MBIG 1000000000
 #define MSEED 161803398
 #define MZ 0
 #define FAC (1.0/MBIG)
 void sran2(unsigned int seed);
 float ran2();
 static int inext,inextp;
 static long ma[56];
 void sran2(unsigned int seed)
 {
  long idum = seed;
  long mj,mk;
  int i,ii,k;
  mj=MSEED-idum;
  mj %= MBIG;
  ma[55]=mj;
  mk=1;
  for (i=1;i<=54;i++) {
   ii=(21*i) % 55;
   ma[ii]=mk;
   mk=mj-mk;
   if (mk < MZ) mk += MBIG;
   mj=ma[ii];
  }
  for (k=1;k<=4;k++)
   for (i=1;i<=55;i++) {
    ma[i] -= ma[1+(i+30) % 55];
    if (ma[i] < MZ) ma[i] += MBIG;
   }
  inext=0;
  inextp=31;
 }
 float ran2()
 {
  long mj;
  int i,ii,k;
  if (++inext == 56) inext=1;
  if (++inextp == 56) inextp=1;
 mj=ma[inext]-ma[inextp];
 if (mj < MZ) mj += MBIG;
 ma[inext]=mj;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  return mj*FAC;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ฉ.

1. โปรแกรม it_m.c

```
/* it_m.c : iteration method for designing fuzzy controller
   of SVC
   Last update Sep 4 , 1998
   Written by Komsan Hongesombut                    */
#include<stdio.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include<time.h>
#include<dos.h>
#include<stdlib.h>
#include"c:\jsvc.c"
/* jsvc.c : SVC control
   jsvcpss1.c : SVC+PSS1 control
   jsvcpss2.c : SVC+PSS2 control
   jsvcpss3.c : SVC+PSS3 control   */
#include"c:\rand3.c"
/* call for random method */
double  alpha1,alpha2,alpha3;
double  alpha1b,alpha2b,alpha3b;
int     loop;
static double  J,J0,Jmin;
FILE *fp;

main()
{
clrscr();
fp = fopen("record.dat","w");
loop=1;Jmin=1000000;
printf(" LOOP   α1     α2     α3        J        Jmin \n");
for(alpha1=0.008;alpha1<=0.011;alpha1+=0.001)
 for(alpha2=89;alpha2<=90;alpha2+=1.0)
  for(alpha3=0.2;alpha3<=0.3;alpha3+=0.01)
   { J = performance_index(alpha1,alpha2,alpha3);
          if(J<=Jmin)
          { Jmin = J;
            alpha1b = alpha1;
            alpha2b = alpha2;
            alpha3b = alpha3;
            fprintf(fp," Best: %f %f %f
%f\n",alpha1b,alpha2b,alpha3b,Jmin);
          }
          printf(" %d %f %f %f %f  %f
\n",loop,alpha1,alpha2,alpha3,J,Jmin);
          printf(" Best : %f %f %f
\n",alpha1b,alpha2b,alpha3b);
          loop++;
    }
fclose(fp);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ช.

1. โปรแกรม GAFCD.c

```
/* GAFCD.c : Genetic Aided Fuzzy Controller Design
  (Genetic Algorithm method for designing fuzzy
controller
  of SVC
  Last updat Sep 4 , 1998
  Written by Komsan Hongesombut                    */
#include<stdio.h>
#include<conio.h>
#include<float.h>
#include<math.h>
#include<time.h>
#include<dos.h>
#include<stdlib.h>
#include"c:\jsvc.c"
/* jsvc.c : SVC control
  jsvcpss1.c : SVC+PSS1 control
  jsvcpss2.c : SVC+PSS2 control
  jsvcpss3.c : SVC+PSS3 control */
#include"c:\eecon98\rand3.c"
/* call for random method */
#define LINELENGTH 80     /* width of printout */
#define BITS_PER_BYTE 8   /* number of bits per byte
on this machine */
#define UINTSIZE (BITS_PER_BYTE*sizeof(unsigned))
/* # of bits in unsigned */

void app_data();
void app_free();
void app_malloc();
void objfunc(struct individual *critter);
void generation();
void initialize();
void initdata();
void initpop();
void initreport();
void initmalloc();
void freeall();
void nomemory(char *string);
int flip(float prob);
void initrandomnormaldeviate();
double noise(double mu ,double sigma);
void randomized();
double randomnormaldeviate();
float randomperc();
int rnd(int low,int high);
float rndreal(float lo ,float hi);
void preselect();
int selected();
void select_memory();
void select_free();
void reset();
void statistics(struct individual *pop);
void copyright();
void repchar (FILE *outfp,char *ch,int repcount);
void skip(FILE *outfp,int skipcount);
int ithruj2int(int i,int j,unsigned *from);
void mutation(unsigned *child);
void uniform_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2);
void report();
void writepop();
void writechrom(unsigned *chrom);
void decode_for_best(struct individual *critter);
void writechrom_to_files_bestfit(unsigned *chrom);
void print_to_file_decode_for_best(struct individual
*critter1);
FILE *outfp, *infp;
FILE *fpga;
/* Global structures and variables */
struct individual
{
  unsigned *chrom;   /* chromosome string for the
individual */
  double  fitness;  /* fitness of the individual */
  int     xsite;    /* crossover site at mating(use for 1-point
crossover) */
  int     parent[2]; /* who the parents of offspring were */
  int   *utility; /* utility field can be used as pointer to a
*/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* dynamically allocated, application-specific data
structure */
};
struct bestever
{
   unsigned *chrom;   /* chromosome string for the best-
ever individual */
   double  fitness;   /* fitness of the best-ever individual */
   int    generation; /* generation which produced it */
};

struct individual *oldpop; /* last generation of individuals
*/
struct individual *newpop;  /* next generation of
individuals */
struct individual *masku;   /* mask for uniform & 2-point
crossover */
struct bestever bestfit;    /* fittest individual so far */
double sumfitness;          /* summed fitness for entire
population */
double max;                 /* maximum fitness of population
*/
double avg;                 /* average fitness of population */
double min;                 /* minumum fitness of population
*/
float pcross,pcross0;       /* probability of crossover */
float pmutation,pmutation0; /* probability of mutation  */
int   numfiles;             /* number of open files */
int   popsize;              /* population size */
int   lchrom;               /* length of the chromosome per
individual */
int   chromsize;            /* number of bytes needed to store
lchrom string */
int   gen;                  /* current generation number */
int   maxgen;               /* maximum generation number */
int   run;                  /* current run number */
int   maxruns;              /* maximum number of runs to
make */
int   printstrings;         /* flag to print chromosome strings
(default on) */
double nmutation;           /* number of mutations  */
double ncross;              /* number of crossovers  */
static int field_size, vec_size; /* field and vector size */
static int  *tourneylist, tourneypos, tourneysize; /* for
tournament selection */
/* variables are declared static so that they cannot conflict
with names of   */
/* other global variables in other files.  See K&R, p 80, for
scope of static */
static double oldrand[55];   /* Array of 55 random
numbers */
static int jrand;            /* current random number */
static double rndx2;         /* used with random normal
deviate */
static int rndcalcflag;      /* used with random normal
deviate */
double start,finish;         /* used for clock */
struct  time t;              /* used for clock */
int hour1,hour2,
   min1,min2,
   sec1,sec2,
   sechund1,sechund2;
time_t tt;                   /* used for initial seed */
int elitist ;                /* used for elitist selection */
void main()
{
   struct individual *temp;
   FILE  *fopen();
   clrscr();
   fpga=fopen("record.dat","w");
   printstrings=1;
   numfiles = 0;
   infp = stdin;
   outfp = stdout;
   copyright();
   if(numfiles == 0)
         fprintf(outfp," Number of GA runs to be
performed-> ");
   fscanf(infp,"%d",&maxruns);
   for(run=1; run<=maxruns; run++)
   {     /* Set things up */
         initialize();
         gettime(&t);    /* get start time */
         hour1 = t.ti_hour;
         min1  = t.ti_min;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        sec1  = t.ti_sec;
        sechund1 = t.ti_hund;
        for(gen=0; gen<maxgen; gen++)
        {
          fprintf(outfp,"\nRUN %d of %d:
GENERATION %d->%d\n",

run,maxruns,gen,maxgen);
          generation();    /* create a new generation */
          statistics(newpop); /* compute fitness
statistics on new populations */
          report();        /* report results for new
generation */
          temp = oldpop;  /* advance the generation */
          oldpop = newpop;
          newpop = temp;
        }
        gettime(&t);       /* get finish time */
        hour2 = t.ti_hour;
        min2  = t.ti_min;
        sec2  = t.ti_sec;
        sechund2 = t.ti_hund;
        printf("\nThe start process time is:
%2d:%02d:%02d.%02d\n",
        hour1,min1,sec1,sechund1);
        fprintf(fpga,"\n");
        fprintf(fpga,"Global Best Individual so far,
Generation %d:\n",
                bestfit.generation);
        print_to_file_decode_for_best(&(bestfit));
        fprintf(fpga,"performance index = %f: ",
(1/bestfit.fitness)*2500);
        fprintf(fpga,"\nThe start process time is:
%2d:%02d:%02d.%02d\n",hour1,min1,sec1,sechund1);
        printf("The stop process time is:
%2d:%02d:%02d.%02d\n",t.ti_hour, t.ti_min, t.ti_sec,
t.ti_hund);
        fprintf(fpga,"The stop process time is:
%2d:%02d:%02d.%02d\n",t.ti_hour, t.ti_min, t.ti_sec,
t.ti_hund);
        freeall(); fclose(fpga);
  }
}

void app_data()
/* application dependent data input, called by init_data() */
/* ask your input questions here, and put output in global
variables */
/* In this example application, the utility pointer of each
individual is  */
/* assigned a vector of integers that will be used to store
the interpreted */
/* chromosome. */
{
  int size = UINTSIZE;
  if(lchrom < UINTSIZE) size = lchrom;
  /* user must specify length of concatenated integers in
the chromosome. */
  fprintf(outfp," Enter field size (must be less than %d) -
>", size);
  fscanf(infp,"%d",&field_size);
  vec_size = lchrom/field_size;
  if((((float)lchrom/(float)field_size)-vec_size) > 0.0)
vec_size++;
}
void app_free()
/* This routine should free any memory allocated */
/* in the application-dependent routines, called by freeall()
*/
{
  int i;
  for(i = 0; i < popsize; i++)
  {
        free(newpop[i].utility);
        free(oldpop[i].utility);
  }
}

void app_malloc()
/* application dependent malloc() calls, called by
initmalloc() */
{
  unsigned nbytes;
  int i;
  nbytes = vec_size * sizeof(int);
  for(i = 0; i < popsize; i++)
  {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if((newpop[i].utility = (int *) malloc(nbytes))
 == NULL)
                nomemory("newpop utility");
            if((oldpop[i].utility = (int *) malloc(nbytes)) ==
 NULL)
                nomemory("oldpop utility");
    }
}

void app_report()
/* Application-dependent report, called by report() */
{
    int i, j;
    /* Print vector interpretation of ech chromosome. */
    for(i = 0; i < popsize; i++)
    {
            fprintf(outfp,"oldpop %d = ", i);
            for(j = 0; j < vec_size; j++)
               fprintf(outfp, "%d ", oldpop[i].utility[j]);
            fprintf(outfp,"\n");
    }
}
void generation()
{
 int mate1, mate2, jcross, j = 0;
 int k;
 /* perform any preselection actions necessary before
generation */
 preselect();
 pcross = pcross0;
 pmutation = pmutation0;
 /* select, crossover, and mutation */
 do
  {
    /* pick a pair of mates */
    mate1=selected();
    mate2=selected();
    /* Crossover and mutation */
    uniform_crossover(oldpop[mate1].chrom, oldpop
[mate2].chrom,newpop[j].chrom, newpop[j+1].chrom);
    mutation(newpop[j].chrom);
    mutation(newpop[j+1].chrom);
      pcross -= (float)((pcross0 - 0.6)/maxgen);      /*
decrease crossver probability*/
      pmutation += (float)((0.1 - pmutation0)/maxgen); /*
increase mutation probability*/
      /* Decode string, evaluate fitness, & record */
      /* parentage date on both children */
      objfunc(&(newpop[j]));
      newpop[j].parent[0] = mate1+1;
      newpop[j].xsite = jcross;
      newpop[j].parent[1] = mate2+1;
      objfunc(&(newpop[j+1]));
      newpop[j+1].parent[0] = mate1+1;
      newpop[j+1].xsite = jcross;
      newpop[j+1].parent[1] = mate2+1;
      /* Increment population index */
      j = j + 2;
     }
    while(j < (popsize-1));
    statistics(newpop);
    /* perform elitist selection */
    if(elitist==1)
    {
      for(k = 0; k < chromsize; k++)
           {
              newpop[0].chrom[k] = bestfit.chrom[k];
           }
      objfunc(&(newpop[0]));
    }
}
void initialize()
{
  initdata();
 /* get basic problem values from input file */
 /* define chromosome size in terms of machine bytes, ie
*/
 /* length of chromosome in bits (lchrom)/(bits-per-byte)
*/
 /* chromsize must be known for malloc() of chrom pointer
*/
  chromsize = (lchrom/UINTSIZE);
  if(lchrom%UINTSIZE) chromsize++;
  initmalloc(); /* malloc space for global data structures */
```

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    randomized(); /* initialize random number generator */
    /* initialize global counters/values */
    nmutation = 0;
    ncross = 0;
    bestfit.fitness = 0.0;
    bestfit.generation = 0;
    /* initialize the populations and report statistics */
    initpop();
    statistics(oldpop);
    initreport();
}

void initdata()
/* data inquiry and setup */
{
    char  answer[2];
    if(numfiles == 0)
    {
            fprintf(outfp,"\n ------- GA Data Entry and
Initialization -------\n");
            fprintf(outfp," Enter the population size --------
---> ");
            fprintf(fpga,"\n ------- GA Data Entry and
Initialization -------\n");

    }
    fscanf(infp,"%d", &popsize);
    fprintf(fpga," Enter the population size ----------> %d
\n",popsize);
    if((popsize%2) != 0)
    {
            fprintf(outfp, "Sorry! only even population
sizes are allowed. \n Incrementing popsize by one.\n");
            popsize++;
    };
    if(numfiles == 0)
            fprintf(outfp," Enter chromosome length (24) --
-------> ");
    fscanf(infp,"%d", &lchrom);
            fprintf(fpga," Enter chromosome length (24) ---
------> %d \n",lchrom);
    if(numfiles == 0)
            fprintf(outfp," Print chromosome strings? (y/n)
------> ");
    fscanf(infp,"%s",answer);
    fprintf(fpga," Print chromosome strings? (y/n) ------> %s
\n",answer);
    if(strncmp(answer,"n",1) == 0) printstrings = 0;
    if(numfiles == 0)
            fprintf(outfp," Enter maximum number of
generations --> ");
    fscanf(infp,"%d", &maxgen);
    fprintf(fpga," Enter maximum number of generations --
> %d \n",maxgen);
    if(numfiles == 0)
            fprintf(outfp," Enter initial crossover
probability ----------> ");
    fscanf(infp,"%f", &pcross0);
    fprintf(fpga," Enter initial crossover probability ----------
> %f \n",pcross0);
    if(numfiles == 0)
            fprintf(outfp," Enter initial mutation probability
----------> ");
    fscanf(infp,"%f", &pmutation0);
    fprintf(fpga," Enter initial mutation probability ----------
> %f \n",pmutation0);
    if(numfiles == 0)
            fprintf(outfp," Enter Elitist Selection (1:Y | 0:N)
--> ");
    fscanf(infp,"%d",&elitist);
    fprintf(fpga," Enter Elitist Selection (1:Y | 0:N) --> %d
\n",elitist);
    app_data(); /* any application-dependent global input */
}
void initpop()
/* Initialize a population at random */
{
    int j, j1, k, stop;
    unsigned mask = 1;
    for(j = 0; j < popsize; j++)
    {
            for(k = 0; k < chromsize; k++)
            {
                oldpop[j].chrom[k] = 0;
                if(k == (chromsize-1))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                stop = lchrom - (k*UINTSIZE);
            else
                stop = UINTSIZE;
            /* A fair coin toss */
            for(j1 = 1; j1 <= stop; j1++)
            {
              oldpop[j].chrom[k] = oldpop[j].chrom[k]
<<1;
              if(flip(0.5))
                oldpop[j].chrom[k] = oldpop
[j].chrom[k]|mask;
            }
          }
          oldpop[j].parent[0] = 0; /* Initialize parent info.
*/
          oldpop[j].parent[1] = 0;
          oldpop[j].xsite = 0;
          objfunc(&(oldpop[j])); /* Evaluate initial
fitness */
    }
}
void initreport()
/* Initial report */
{
  void skip();
  skip(outfp,1);
  fprintf(outfp," GA Parameters\n");
  fprintf(outfp," ----------------------------------------
\n");
  fprintf(outfp," Total Population size          =
%d\n",popsize);
  fprintf(outfp," Chromosome length (lchrom)     =
%d\n",lchrom);
  fprintf(outfp," Maximum # of generations (maxgen) =
%d\n",maxgen);
  fprintf(outfp," Crossover probability (pcross)   = %f\n",
pcross0);
  fprintf(outfp," Mutation probability (pmutation) =
%f\n", pmutation0);
  skip(outfp,1);

  fprintf(fpga," GA Parameters\n");
  fprintf(fpga," ----------------------------------------
\n");
  fprintf(fpga," Total Population size          =
%d\n",popsize);
  fprintf(fpga," Chromosome length (lchrom)     =
%d\n",lchrom);
  fprintf(fpga," Maximum # of generations (maxgen) =
%d\n",maxgen);
  fprintf(fpga," Crossover probability (pcross)   = %f\n",
pcross0);
  fprintf(fpga," Mutation probability (pmutation) =
%f\n", pmutation0);
  fprintf(fpga," ----------------------------------------
\n\n");
  fprintf(fpga," min    max    avg    Σfitness
best.fitness   chromosome \n\n");
  fflush(outfp);
}
void initmalloc()
/* memory allocation of space for global data structures */
{
 unsigned nbytes;
 int j;
 /* memory for old and new populations of individuals */
 nbytes = popsize*sizeof(struct individual);
 if((oldpop = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("oldpop");
 if((newpop = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("newpop");
 if((masku = (struct individual *) malloc(nbytes)) ==
NULL)
  nomemory("masku");
 /* memory for chromosome strings in populations */
 nbytes = chromsize*sizeof(unsigned);
 for(j = 0; j < popsize; j++)
 {
  if((oldpop[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
        nomemory("oldpop chromosomes");
  if((newpop[j].chrom = (unsigned *) malloc(nbytes))
== NULL)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            nomemory("newpop chromosomes");
    if((masku[j].chrom = (unsigned *) malloc(nbytes)) ==
NULL)
            nomemory("masku chromosomes");
    }
  if((bestfit.chrom = (unsigned *) malloc(nbytes)) ==
NULL)
    nomemory("bestfit chromosome");
  select_memory();
  /* allocate any auxiliary memory for selection */
  /* call to application-specific malloc() routines  */
  /* can be used to malloc memory for utility pointer */
  app_malloc();
}
void freeall()
/* A routine to free all the space dynamically allocated in
initspace() */
{
 int i;
 for(i = 0; i < popsize; i++)
  {
   free(oldpop[i].chrom);
   free(newpop[i].chrom);
   free(masku[i].chrom);
  }
 free(oldpop);
 free(newpop);
 free(masku);
 free(bestfit.chrom);
 select_free(); /* free any auxiliary memory needed for
selection */
 /* call to application-specific free() routines  */
 /* can be used to free memory for utility pointer */
 app_free();
}
void nomemory(char *string)
{
 fprintf(stderr,"malloc: out of memory making
%s!!\n",string);
 exit(-1);
}
int flip(float prob)
/* Flip a biased coin - true if heads */
{
  float randomperc();
  if(randomperc() <= prob)
        return(1);
  else
        return(0);
}

void initrandomnormaldeviate()
/* initialization routine for randomnormaldeviate */
{
  rndcalcflag = 1;
}

double noise(double mu ,double sigma)
/* normal noise with specified mean & std dev: mu &
sigma */
{
  double randomnormaldeviate();
  return((randomnormaldeviate()*sigma) + mu);
}
void randomized()
/* Get seed number for random and start it up */
{  unsigned int ss;
  ss = (unsigned int) time(&tt); /* seed by CPU's clock */
/* direct seed by replacing ss value with the following */
/* ss = 592558108 for c:\jsvc.c */
/* ss = 510443102 for c:\jsvcpss1.c */
/* ss = 594592699 for c:\jsvcpss2.c */
/* ss = 517785338 for c:\jsvcpss3.c */
/* when you use ss value according to above , your results
  will like the results in this thesis (see chapter 8)  */
  sran2((unsigned int)ss);
}

double randomnormaldeviate()
/* random normal deviate after ACM algorithm 267 / Box-
Muller Method */
{
 double t, rndx1;
 if(rndcalcflag)
 {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            rndx1 = sqrt(- 2.0*log((double) randomperc
()));
            t = 6.2831853072 * (double) randomperc();
            rndx2 = rndx1 * sin(t);
            rndcalcflag = 0;
            return(rndx1 * cos(t));
    }
    else
    {
            rndcalcflag = 1;
            return(rndx2);
    }
}

float randomperc()
/* Fetch a single random number between 0.0 and 1.0 */
{
  return((float)ran2());
}

int rnd(int low,int high)
/* Pick a random integer between low and high */
{
   int i;
   if(low >= high)
            i = low;
   else
   {
            i = (randomperc() * (high - low + 1)) + low;
            if(i > high) i = high;
   }
   return(i);
}

float rndreal(float lo ,float hi)
/* real random number between specified limits */
{
   return((randomperc() * (hi - lo)) + lo);
}
void statistics(struct individual *pop)
/* Calculate population statistics */
{
   int i, j;
   sumfitness = 0.0;
   min = pop[0].fitness;
   max = pop[0].fitness;
   /* Loop for max, min, sumfitness */
   for(j = 0; j < popsize; j++)
   {
            sumfitness = sumfitness + pop[j].fitness;
  /* Accumulate */
            if(pop[j].fitness > max) max = pop[j].fitness;
  /* New maximum */
            if(pop[j].fitness < min) min = pop[j].fitness;
  /* New minimum */
            /* new global best-fit individual */
            if(pop[j].fitness > bestfit.fitness)
             {
              for(i = 0; i < chromsize; i++)
               bestfit.chrom[i]    = pop[j].chrom[i];

              bestfit.fitness    = pop[j].fitness;
              bestfit.generation = gen;
             }
   }
   /* Calculate average */
   avg = sumfitness/popsize;
}
void copyright()
{
   int iskip;
   int ll = 59;
   iskip = (LINELENGTH - ll)/2;
   skip(outfp,1);
   repchar(outfp," ",iskip); repchar(outfp,"-",ll); skip
(outfp,1);
   repchar(outfp," ",iskip-1);
   fprintf(outfp,"|       GA Aided Fuzzy Controller Design
- v1.0       |\n");
   repchar(outfp," ",iskip);
   repchar(outfp,"-",ll); skip(outfp,2);
}
void repchar (FILE *outfp,char *ch,int repcount)
/* Repeatedly write a character to stdout */
{
   int j;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    for (j = 1; j <= repcount; j++) fprintf(outfp,"%s", ch);
}
void skip(FILE *outfp,int skipcount)  /* Skip skipcount
lines */
{
   int j;
   for (j = 1; j <= skipcount; j++) fprintf(outfp,"\n");
}

int ithruj2int(int i,int j,unsigned *from)
/* interpret bits i thru j of a individual as an integer     */
/* j MUST BE greater than or equal to i AND j-i <
UINTSIZE-1  */
/* from is a chromosome, represented as an array of
unsigneds */
{
   unsigned mask, temp;
   int bound_flag;
   int iisin, jisin;
   int i1, j1, out;
   if(j < i)
   {
            fprintf(stderr,"Error in ithruj2int: j < i\n");
            exit(-1);
   }
   if(j-i+1 > UINTSIZE)
   {
            fprintf(stderr,"Error in ithruj2int: j-i+1 >
UINTSIZE\n");
            exit(-1);
   }
   iisin = i/UINTSIZE;
   jisin = j/UINTSIZE;

   i1 = i - (iisin*UINTSIZE);
   j1 = j - (jisin*UINTSIZE);

            if(i1 == 0)
            {
                     iisin = iisin-1;
                     i1 = i - (iisin*UINTSIZE);
            };
            if(j1 == 0)
            {
                     jisin = jisin-1;
                     j1 = j - (jisin*UINTSIZE);
            };
   /* check if bits fall across a word boundary */
   if(iisin == jisin)
            bound_flag = 0;
   else
            bound_flag = 1;
   if(bound_flag == 0)
   {
            mask = 1;
            mask = (mask<<(j1-i1+1))-1;
            mask = mask<<(i1-1);
            out = (from[iisin]&mask)>>(i1-1);
            return(out);
   }
   else
   {
            mask = 1;
            mask = (mask<<j1)-1;
            temp = from[jisin]&mask;
            mask = 1;
            mask = (mask<<(UINTSIZE-i1+1))-1;
            mask = mask<<(i1-1);
            out = ((from[iisin]&mask)>>(i1-1)) | (temp
<<(UINTSIZE-i1+1));
            return(out);
   }
}
void mutation(unsigned *child)
/* Mutate an allele w/ pmutation, count # of mutations */
{
   int j, k, stop;
   unsigned mask, temp = 1;
   for(k = 0; k < chromsize; k++)
   {
            mask = 0;
            if(k == (chromsize-1))
               stop = lchrom - (k*UINTSIZE);
            else
               stop = UINTSIZE;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            for(j = 0; j < stop; j++)
            {
              if(flip(pmutation))
              {
                    mask = mask|(temp<<j);
                    nmutation++;
              }
            }
            child[k] = child[k]^mask;
    }
}
void uniform_crossover(unsigned *parent1,unsigned
*parent2,unsigned *child1 ,unsigned *child2)
{
 int i, j, b,l, which_way, data_width;
 int  j1, stop;
 unsigned  mask1;
 int cp1,cp2;
 int m;
 int k;
 struct individual *idp;
 int fraction;
 if (flip(pcross))
 {
 ncross++;
 for(k = 0; k < chromsize; k++)
          {
            masku[1].chrom[k] = 0;
            if(k == (chromsize-1))
                    stop = lchrom - (k*UINTSIZE);
            else
                    stop = UINTSIZE;
            /* A fair coin toss */
            for(j1 = 1; j1 <= stop; j1++)
            {
              masku[1].chrom[k] = masku[1].chrom[k]
<<1;
              if(flip(0.5))
                    masku[1].chrom[k] = masku
[1].chrom[k]|0x1;
            }
          }
  k=(chromsize-1);
  fraction = lchrom-((chromsize-1)*16);
  /* start fraction cross over loop */
 m=0x1 << (fraction-1);
 mask1=m;
 child1[k]=0x0;
 child2[k]=0x0;
 for(i=1;i<=fraction;i++)
 { cp1=(mask1&(parent1[k]))>>(fraction-i);
  cp2=(mask1&(parent2[k]))>>(fraction-i);
  if ((masku[1].chrom[k]&mask1)>>(fraction-i))
  { child1[k]<<=1; child1[k]=(child1[k]|cp2);
   child2[k]<<=1; child2[k]=(child2[k]|cp1);
  }
  else  { child1[k]<<=1; child1[k]=(child1[k]|cp1);
          child2[k]<<=1; child2[k]=(child2[k]|cp2);
          }
  mask1>>=1;
  }
  l=(k-1);
  for(k=l;k>=0;k--)
  {
  m=0x1 << 15;
  mask1=m;
  child1[k]=0x0;
  child2[k]=0x0;
  for(i=1;i<=16;i++)
  { cp1=(mask1&(parent1[k]))>>(16-i);
  cp2=(mask1&(parent2[k]))>>(16-i);
  if ((masku[1].chrom[k]&mask1)>>(16-i))
  { child1[k]<<=1; child1[k]=(child1[k]|cp2);
   child2[k]<<=1; child2[k]=(child2[k]|cp1);
  }
  else  { child1[k]<<=1; child1[k]=(child1[k]|cp1);
          child2[k]<<=1; child2[k]=(child2[k]|cp2);
          }
mask1>>=1;
}
}
}
else
{
 for(k=0;k<chromsize; k++)
 {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      child1[k] = parent1[k];
      child2[k] = parent2[k];
    }
   }
  }

  void report()
  /* Write the population report */
  {
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
    if(printstrings == 1)
    {
          repchar(outfp," ",((LINELENGTH-17)/2));
          fprintf(outfp,"Population Report\n");
          fprintf(outfp, "Generation %3d", gen);
          repchar(outfp," ",(LINELENGTH-28));
          fprintf(outfp, "Generation %3d\n", (gen+1));
          fprintf(outfp,"num   string ");
          repchar(outfp," ",lchrom-5);
          fprintf(outfp,"fitness    parents xsite  ");
          fprintf(outfp,"string ");
          repchar(outfp," ",lchrom-5);
          fprintf(outfp,"fitness\n");
          repchar(outfp,"-",LINELENGTH);
          skip(outfp,1);

          writepop();
          repchar(outfp,"-",LINELENGTH);
          skip(outfp,1);
    }
    /* write the summary statistics in global mode  */
    fprintf(outfp,"Generation %d Accumulated Statistics:
\n",gen);
    fprintf(outfp,"Total Crossovers = %f, Total Mutations =
%f\n",ncross,nmutation);
    fprintf(outfp,"min = %f   max = %f   avg = %f   sum =
%f\n",min,max,avg,sumfitness);
    fprintf(fpga," %f  %f  %f  %f  %f
",min,max,avg,sumfitness,bestfit.fitness);
    fprintf(outfp,"Global Best Individual so far, Generation
%d:\n",bestfit.generation);
    writechrom_to_files_bestfit((&bestfit)->chrom);
    fprintf(outfp,"Fitness = %f: ", bestfit.fitness);
    writechrom((&bestfit)->chrom);fprintf(fpga,"\n");
    skip(outfp,1);
    decode_for_best((&bestfit));
    objfunc((&bestfit));
    fprintf(outfp,"performance index = %f: ",
(1/bestfit.fitness)*2500);
    skip(outfp,1);
    repchar(outfp,"-",LINELENGTH);
    skip(outfp,1);
  }
  void writepop()
  {
    struct individual *pind;
    int j;
    for(j=0; j<=popsize-1; j++)
    {
          fprintf(outfp,"%3d)  ",j+1);
          /* Old string */
          pind = &(oldpop[j]);
          writechrom(pind->chrom);
          fprintf(outfp,"  %8f | ", pind->fitness);
          /* New string */
          pind = &(newpop[j]);
          fprintf(outfp,"(%2d,%2d)  %2d   ",
          pind->parent[0], pind->parent[1], pind->xsite);
          writechrom(pind->chrom);
          fprintf(outfp,"  %8f\n", pind->fitness);
    }
  }
  void writechrom(unsigned *chrom)
  /* Write a chromosome as a string of ones and zeroes
  */
  {
    int j, k, stop;
    unsigned mask=1,tmp;
    static int bit_count=1;
    unsigned mask1;
    int m;
    printf(" ");
    bit_count=1;
    for(k = chromsize-1; k>=0; k--)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            tmp = chrom[k];
            if(k == (chromsize-1))
              { stop = lchrom - (k*UINTSIZE);m=0x1
<<(stop-1);}
            else
              { stop = UINTSIZE;m=0x1<<(stop-1);}
            mask1=m;
            for(j = 0; j < stop; j++)
            {
              if(tmp&mask1)
                    fprintf(outfp,"1");
              else
                    fprintf(outfp,"0");
              mask1>>=1;
            }
    }
}
void objfunc(struct individual *critter)
{
   unsigned mask=0x1;  /* mask for current bit */
   int bitpos;    /* current bit position */
   unsigned bitpos2=0,bitpos3=0;
   unsigned tp;
   double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
   int i,j, k, stop;
   int n = 10;
   double alpha1,alpha2,alpha3,SWL;
   static int gray[24],bcd[24];
   alpha1=0.0; alpha2=0.0; alpha3=0.0;SWL=0.0;
   bitpow2=0; bitpow3=0;
   critter->fitness = 0.0;
   /* loop thru number of bytes holding chromosome */
   for(k = 0; k <= chromsize-1; k++)
   {
          if(k == (chromsize-1))
            {stop = lchrom-(k*UINTSIZE);}
          else
            {stop = UINTSIZE;}
          /* loop thru bits in current byte */
          tp = critter->chrom[k];
          for(j = 0; j < stop; j++)
          {
            bitpos = j + UINTSIZE*k;
            /* test for current bit 0 or 1 */
            if((bitpos>=0)&&(bitpos<=6))
            {
             if((tp&mask) == 1)
             {
                    gray[(int)bitpos]=1;
             }
             else
             gray[(int)bitpos]=0;
            }
            if((bitpos>=7)&&(bitpos<=13))
            {
             if((tp&mask) == 1)
              { gray[(int)bitpos]=1;
              }
              else
             gray[(int)bitpos]=0;
            }
            if((bitpos>=14)&&(bitpos<=23))
            {
             if((tp&mask)==1)
              { gray[(int)bitpos]=1;
              }
              else
              gray[(int)bitpos]=0;
             }
             tp = tp>>=1;
          }
   }
   bcd[6]=gray[6];
   for(i=5;i>=0;i--)
   bcd[i]=bcd[i+1]^gray[i];

   bcd[13]=gray[13];
   for(i=12;i>=7;i--)
   bcd[i]=bcd[i+1]^gray[i];

   bcd[23]=gray[23];
   for(i=22;i>=14;i--)
   bcd[i]=bcd[i+1]^gray[i];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    for(i=0;i<=6;i++)
    {
    alpha3 += bcd[i]*pow(2.0,(double)i);
    }
    for(i=7;i<=13;i++)
    {
    alpha2 += bcd[i]*pow(2.0,(double)(i-7));
    }
    for(i=14;i<=23;i++)
    {
    alpha1 += bcd[i]*pow(2.0,(double)(i-14));
    }

    alpha1 =  (double)((int)(alpha1*1000/1023))/1000;
    alpha2 =  (double)((int)(alpha2*90/127));
    alpha3 =  (double)((int)(alpha3*100/127))/100;
    critter->fitness = (2500/(performance_index
(alpha1,alpha2,alpha3)));
}

void decode_for_best(struct individual *critter1)
{
    unsigned mask=0x1;  /* mask for current bit */
    int bitpos;    /* current bit position */
    unsigned bitpos2=0,bitpos3=0;
    unsigned tp;
    double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
    int i,j, k, stop;
    int n = 10;
    double alpha1,alpha2,alpha3,SWL;
    static int gray[24],bcd[24];

    alpha1=0.0; alpha2=0.0; alpha3=0.0;SWL=0.0;
    bitpow2=0; bitpow3=0;
    critter1->fitness = 0.0;
    /* loop thru number of bytes holding chromosome */
    for(k = 0; k <= chromsize-1; k++)
    {
        if(k == (chromsize-1))
            {stop = lchrom-(k*UINTSIZE);}
        else
            {stop = UINTSIZE;}
        /* loop thru bits in current byte */
        tp = critter1->chrom[k];
        for(j = 0; j < stop; j++)
        {
          bitpos = j + UINTSIZE*k;
          /* test for current bit 0 or 1 */
          if((bitpos>=0)&&(bitpos<=6))
          {
           if((tp&mask) == 1)
           { gray[(int)bitpos]=1;
           }
           else
           gray[(int)bitpos]=0;
          }
          if((bitpos>=7)&&(bitpos<=13))
          {
           if((tp&mask) == 1)
           {  gray[(int)bitpos]=1;
           }
           else
           gray[(int)bitpos]=0;
          }
          if((bitpos>=14)&&(bitpos<=23))
          {
           if((tp&mask)==1)
           {  gray[(int)bitpos]=1;
           }
           else
           gray[(int)bitpos]=0;
          }
          tp = tp>>=1;
        }
    }

    bcd[6]=gray[6];
    for(i=5;i>=0;i--)
    bcd[i]=bcd[i+1]^gray[i];

    bcd[13]=gray[13];
    for(i=12;i>=7;i--)
    bcd[i]=bcd[i+1]^gray[i];

    bcd[23]=gray[23];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    for(i=22;i>=14;i--)
    bcd[i]=bcd[i+1]^gray[i];



    for(i=0;i<=6;i++)
    {
    alpha3 += bcd[i]*pow(2.0,(double)i);
    }
    for(i=7;i<=13;i++)
    {
    alpha2 += bcd[i]*pow(2.0,(double)(i-7));
    }
    for(i=14;i<=23;i++)
    {
    alpha1 += bcd[i]*pow(2.0,(double)(i-14));
    }


    alpha1 =  (double)((int)(alpha1*1000/1023))/1000;
    alpha2 =  (double)((int)(alpha2*90/127));
    alpha3 =  (double)((int)(alpha3*100/127))/100;
    critter1->fitness = (2500/(performance_index
(alpha1,alpha2,alpha3)));
    fprintf(outfp,"t1 = %f  t2 = %f  t3 = %f
\n",alpha1,alpha2,alpha3);


}


void writechrom_to_files_bestfit(unsigned *chrom)
/* Write a chromosome as a string of ones and zeroes */
{
  int j, k, stop;
  unsigned mask=1,tmp;
  static int bit_count=1;
  unsigned mask1;
  int m;
  bit_count=1;
  for(k = chromsize-1; k>=0; k--)
  {
        tmp = chrom[k];
        if(k == (chromsize-1))
          { stop = lchrom - (k*UINTSIZE);m=0x1
<<(stop-1);}
        else
          { stop = UINTSIZE;m=0x1<<(stop-1);}
        mask1=m;
        for(j = 0; j < stop; j++)
        {
          if(tmp&mask1)
                fprintf(fpga,"1");
          else
                fprintf(fpga,"0");
          mask1>>=1;
        }
  }
}

void print_to_file_decode_for_best(struct individual
*critter1)
{
  unsigned mask=0x1;  /* mask for current bit */
  int bitpos;   /* current bit position */
  unsigned bitpos2=0,bitpos3=0;
  unsigned tp;
  double pow(), bitpow1=0, coef,bitpow2=0,bitpow3=0;
  int i,j, k, stop;
  int n = 10;
  double alpha1,alpha2,alpha3,SWL;
  static int gray[24],bcd[24];

  alpha1=0.0; alpha2=0.0; alpha3=0.0;SWL=0.0;
  bitpow2=0; bitpow3=0;
  critter1->fitness = 0.0;
  /* loop thru number of bytes holding chromosome */
  for(k = 0; k <= chromsize-1; k++)
  {
        if(k == (chromsize-1))
          {stop = lchrom-(k*UINTSIZE);}
        else
          {stop = UINTSIZE;}
        /* loop thru bits in current byte */
        tp = critter1->chrom[k];
        for(j = 0; j < stop; j++)
        {
          bitpos = j + UINTSIZE*k;
          /* test for current bit 0 or 1 */
          if((bitpos>=0)&&(bitpos<=6))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
         if((tp&mask) == 1)
                { gray[(int)bitpos]=1;
                }
                else
         gray[(int)bitpos]=0;

        }
        if((bitpos>=7)&&(bitpos<=13))
        {
         if((tp&mask) == 1)
          { gray[(int)bitpos]=1;
          }
          else
         gray[(int)bitpos]=0;

        }
        if((bitpos>=14)&&(bitpos<=23))
        {
         if((tp&mask)==1)
          { gray[(int)bitpos]=1;
          }
          else
          gray[(int)bitpos]=0;
        }
        tp = tp>>=1;
      }
}
/* transform gray code to bcd */
bcd[6]=gray[6];
for(i=5;i>=0;i--)
bcd[i]=bcd[i+1]^gray[i];

bcd[13]=gray[13];
for(i=12;i>=7;i--)
bcd[i]=bcd[i+1]^gray[i];

bcd[23]=gray[23];
for(i=22;i>=14;i--)
bcd[i]=bcd[i+1]^gray[i];

/* calculate bcd to decimal number */
for(i=0;i<=6;i++)
        {
        alpha3 += bcd[i]*pow(2.0,(double)i);
        }
        for(i=7;i<=13;i++)
        {
        alpha2 += bcd[i]*pow(2.0,(double)(i-7));
        }
        for(i=14;i<=23;i++)
        {
        alpha1 += bcd[i]*pow(2.0,(double)(i-14));
        }
        alpha1 = (double)((int)(alpha1*1000/1023))/1000;
        alpha2 = (double)((int)(alpha2*90/127));
        alpha3 = (double)((int)(alpha3*100/127))/100;
        critter1->fitness = (2500/(performance_index
(alpha1,alpha2,alpha3)));
        fprintf(fpga,"t1 = %f  t2 = %f  t3 = %f
\n",alpha1,alpha2,alpha3);
}
void select_memory()
/* used for tournament selection */
{
 unsigned nbytes;
 int j;
 nbytes = popsize*sizeof(int);
 if((tourneylist = (int *) malloc(nbytes)) == NULL)
  nomemory("tourneylist");
 if(numfiles == 0)
  fprintf(outfp," Enter tournament size for selection --> ");
 fscanf(infp,"%d", &tourneysize);
 fprintf(fpga," Enter tournament size for selection --> %d
\n",tourneysize);
 if(tourneysize > popsize)
  {
   fprintf(outfp,"FATAL: Tournament size (%d) >
popsize (%d)\n",
                tourneysize,popsize);
   exit(-1);
  }
}
void select_free()
/* used for tournament selection */
{
```

เอกสารนี้มีลิขสิทธิ์สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า กรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  free(tourneylist);
 };

 void preselect()
 /* used for tournament selection */
 {
   reset();
   tourneypos = 0;
 }

 int selected()
 /* selection */
 {
   int pick, winner, i;
   /* If remaining members not enough for a tournament,
 then reset list */
   if((popsize - tourneypos) < tourneysize)
   {
           reset();
           tourneypos = 0;
   }
   /* Select tourneysize structures at random and conduct a
 tournament */
   winner=tourneylist[tourneypos];
   for(i=1; i<tourneysize; i++)
   {
           pick=tourneylist[i+tourneypos];
           if(oldpop[pick].fitness >
oldpop[winner].fitness) winner=pick;
   }
   /* Update tourneypos */
   tourneypos += tourneysize;
   return(winner);
}

void reset()
/* Shuffles the tourneylist at random */
{
  int i, rand1, rand2, temp;

  for(i=0; i<popsize; i++) tourneylist[i] = i;

  for(i=0; i < popsize; i++)
  {
          rand1=rnd(i,popsize-1);
          rand2=rnd(i,popsize-1);
          temp = tourneylist[rand1];
          tourneylist[rand1]=tourneylist[rand2];
          tourneylist[rand2]=temp;
  }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ซ.

1. ตัวอย่างการหาค่า α_1, α_2, α_3 โดยใช้เจเนติกอัลกอริทึ่ม กรณีควบคุมด้วย SVC ชนิดฟัซซี่

ขั้นตอนต่างๆในการคำนวณด้วยมือ เป็นดังต่อไปนี้

1. สมมุติให้มีการกำหนดค่าเริ่มต้นโปรแกรมดังนี้

lchrom = 24 , popsize = 30, max gen = 50, P_{c0} = 0.9, P_{m0} = 0.01 และ tournament size = 2

2. สร้างประชากรเริ่มต้นโดยวิธีสุ่มโดยมีจำนวนเท่ากับ popsize คือ 30 เช่น

1: 000010011101101110101101

2: 000101100100001110101111

⋮ ⋮

30: 000000100100100011111101

3. ถอดรหัสโครโมโซมแต่ละตัวโดยมีขั้นตอนตามตัวอย่างข้างล่างนี้

3.1 แปลงรหัสเกรย์ไปเป็นรหัสไบนารี

30: 000000100100100011111101 (เกรย์)

↓

30: 000000111000111101010110 (ไบนารี)

3.2 แยก α_1, α_2 และ α_3 จากโครโมโซมที่เป็นไบนารีในหัวข้อ 3.1 โดยความยาวของ α_1 = 10 บิต α_2 = 7 บิต และ α_3 = 7 บิต เรียงลำดับจากซ้ายไปขวา เช่น โครโมโซมที่เป็นไบนารีในตัวอย่างหัวข้อ 3.1 จะได้ α_1 = 0000001110 = 14 , α_2 = 0011110 = 30 , α_3 = 1010110 = 86

3.3 หาค่า α_1, α_2 และ α_3 เช่น จากข้อมูลในตัวอย่างหัวข้อ 3.2 จะได้

$$\alpha_1 = (\text{int}(14)(10^3)/2^{10})/10^3 = 0.014$$

$$\alpha_2 = \text{int}(30)(90\text{-}0)/2^7 = 21^\circ$$

$$\alpha_3 = (\text{int}(86)(10^2)/2^7)/10^2 = 0.67$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. แทนค่า α_1 , α_2 ,α_3 ในโปรแกรมย่อย performance_index เช่น α_1 = 0.014 , α_2 = 21° และ α_3 = 0.67 จะได้ค่าดัชนีทำงาน J เท่ากับ 7757.663

5. นำค่า J ไปคำนวณหาค่าฟิทเนสของโครโมโซมนั้นจาก F = 2,500/J เช่น J = 7757.663 จะได้ F = 2,500/7757.663 = 0.322262

6. คำนวณซ้ำตั้งแต่ข้อ 3 ถึง ข้อ 5 จนกระทั่งครบ 30 รอบเท่ากับจำนวน popsize ตัวอย่างผลการรันในเจนเนอเรชั่นที่ 0 มีลักษณะดังต่อไปนี้

GEN0:

1: 000010011101101110101101 , F1 =0.20478

2: 000101100100001110101111 , F2 =0.18836

⋮ ⋮ ⋮

30: 000000100100100011111101 , F30 =0.32262

คำนวณค่าข้อมูลทางสถิติของ GEN0 ได้ดังนี้ min = 0.09287 , max = 0.32262 , avg = 0.18690 , $P_c = 0.9$, $P_m = 0.01$

7. เก็บคำตอบที่ดีที่สุดไว้ 1 ตัวใน เช่น ใน GEN0 คือ โครโมโซมที่ 30: 000000100100100011111101 ซึ่งมีค่าฟิทเนสเท่ากับ 0.32262 หรือ $\alpha_1 = 0.009$, $\alpha_2 = 12^\circ$, $\alpha_3 = 0.98$

8. สร้างประชากรใหม่โดยใช้ข้อมูลจากเจนเนอเรชั่นก่อนหน้านั่น ดังนี้

8.1 เลือก P1 และ P2 จากโครโมโซมทั้ง 30 ตัวเพื่อสร้าง C1 และ C2 โดยใช้วิธีเลือกแบบทัวร์นาเม้นท์ แล้วทำการครอสโอเวอร์แบบยูนิฟอร์ม เช่น ตัวอย่างข้างล่างเป็นการสร้างประชากรใหม่ใน GEN1 จากข้อมูลประชากรใน GEN0

GEN0:

1: 000010011101101110101101 → P2 C1: 000100000101001110101111

2: 000101100100001110101111 → P1 C2: 000011111100101110101101

⋮ ⋮ ⋮

30: 000000100100100011111101

8.2 ทำมิวเทชั่น โดยสุ่มค่าตัวเลขในช่วง[0,1] ถ้าค่าที่สุ่มได้มีค่ามากกว่า P_m ก็จะทำการมิวเทชั่น ณ ตำแหน่งนั้นๆ เช่น จากตัวอย่างโครโมโซมในหัวข้อ 8.1 ถ้าตำแหน่งที่ 5 จากซ้ายมือของ C1 สุ่มได้ตัวเลข 0.45 โดย $P_m = 0.01$ แสดงว่า ตำแหน่งที่ 5 ของ C1 จะเปลี่ยนจาก 0 เป็น 1 จากวิธีดังกล่าวสมมุติว่าเกิดการมิวเทชั่นในตำแหน่งที่ 5 ของ C1 และตำแหน่งที่ 10 ของ C2 ผลลัพธ์ที่ได้จะเป็นดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

C1: 000110000101001110101111

C2: 000011111000101110101101

9. ทำซ้ำหัวข้อ 8.1 และ 8.2 จนได้ประชากรใหม่ทั้ง 30 ตัว ซึ่งจะเป็นประชากรใหม่ในเจนเนอเรชั่นถัดไปจากนั้นทำการหาค่าฟิทเนสของโครโมโซมทุกตัว เช่น ตัวอย่างข้อมูลที่ได้ของ GEN1 จะมีลักษณะดังรูปข้างล่าง

GEN1:

1: 000110000101001110101111 , F1 = 0.20863
2: 000011111000101110101101 , F2 = 0.23715
⋮ ⋮ ⋮
30: 000000101010000000010001 , F30 = 0.29815

คำนวณค่าข้อมูลทางสถิติของ GEN1 ได้ดังนี้ min = 0.18191 , max = 0.529815 , avg = 0.26551 , $P_c = 0.894$, $P_m = 0.0118$

10. ทำซ้ำขั้นตอนที่ 3 ถึงขั้นตอนที่ 9 จนกระทั่งเท่ากับค่า max gen ที่ตั้งไว้คือ 50 ตัวอย่าง ข้อมูลสำหรับ GEN15 และ GEN42 แสดงดังต่อไปนี้

GEN15:

1: 000110000101001110101111 , F1 = 0.20863
2: 000011111000101110101101 , F2 = 0.23715
⋮ ⋮ ⋮
30: 000000101110000010110000 , F30 = 0.56631

คำนวณค่าข้อมูลทางสถิติของ GEN15 ได้ดังนี้ min = 0.18191 , max = 0.56631 , avg = 0.81373 , $P_c = 0.810$, $P_m = 0.037$

GEN42:

1: 000011111000101110101101 , F1 = 0.23715
2: 000110000101001110101111 , F2 = 0.20863
⋮ ⋮ ⋮
30: 000000101100000000010000 , F30 = 1.11282

คำนวณค่าข้อมูลทางสถิติของ GEN42 ได้ดังนี้ min = 0.18191 , max = 1.11282 , avg = 0.87524 , $P_c = 0.648$, $P_m = 0.0856$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ฌ.

ผลงานวิชาการที่ได้รับการตีพิมพ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วยตัวชดเชยกำลังไฟฟ้าแบบสถิตย์โดยใช้วิธีฟัซซี่

Improving Power System Stability Using Static Var Compensator Based on Fuzzy Control

คมสันต์ หงษ์สมบัติ มณฑล ลีลาจินดาไกรฤกษ์

ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ถนนฉลองกรุง เขตลาดกระบัง กรุงเทพฯ 10520

โทร. (02) 3266052-101 EXT. 619 E-mail: s0061034@kmitl.ac.th , komsan.h@usa.net ICQ: 9760887

บทคัดย่อ

บทความนี้เสนอวิธีการประยุกต์ใช้ทฤษฎีฟัซซี่มาใช้ในการควบคุมตัวชดเชยกำลังไฟฟ้าแบบสถิตย์(SVC)เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง โดยใช้แบบจำลองซึ่งประกอบด้วยเครื่องกำเนิดไฟฟ้าซิงโครนัส1ตัวต่ออยู่กับบัสอนันต์ (Infinite Bus) ผ่านสายส่งวงจรคู่และมีตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ติดตั้งอยู่จุดกึ่งกลางของสายส่ง สัญญาณส่วนเบี่ยงเบนและอัตราการเปลี่ยนแปลงของกำลังไฟฟ้า ณ จุดที่ติดตั้งตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ จะถูกเลือกเป็นอินพุทสำหรับตัวควบคุมฟัซซี่ เพื่อใช้ในการคำนวณสัญญาณควบคุมเสริมของ SVC โดยอาศัยข้อมูลซึ่งได้รับจากเฟสเพลนและฟังก์ชั่นการเป็นสมาชิก ผลการจำลองด้วยคอมพิวเตอร์แสดงให้เห็นถึงประสิทธิภาพของวิธีการที่ได้นำเสนอ รวมถึงการประยุกต์ใช้วิธีที่นำเสนอร่วมกับตัวควบคุมเสถียรภาพระบบไฟฟ้ากำลัง(Power System Stabilizer : PSS)ชนิดต่างๆเช่น ΔP-PSS , Δω-PSS และ Δω-PSS ชนิดควบคุมด้วยฟัซซี่

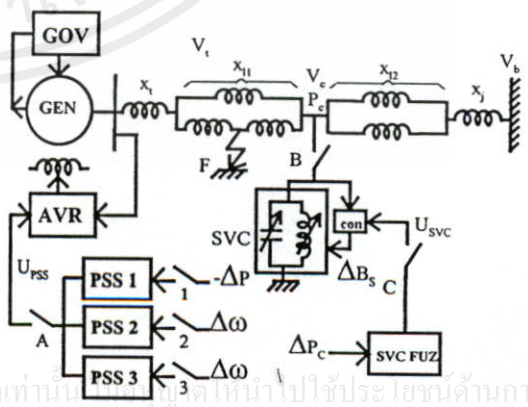
Abstract

In this paper, fuzzy theory has been applied to a static var compensator (SVC) for improving power system stability.The system considered is a synchronous generator connected to an infinite bus through a double circuit transmission line and SVC is installed at the midpoint. In order to maintain the good damping characteristic, electric power deviation at the location of SVC and its change are chosen as input signals at each sampling time for SVC supplementary control signal calculation using fuzzy membership functions which utilize the information from phase plane. All simulations demonstrate the effectiveness of this proposed control method and show the application of this proposed control method operated together with serveral types of power system stabilizers (PSS) ,such as ΔP-PSS,Δω-PSS and Δω-PSS predicated on fuzzy control.

1. บทนำ

ระบบไฟฟ้ากำลังในปัจจุบันมีขนาดใหญ่และมีความซับซ้อนมากขึ้น ทำให้กำลังถูกควบคุมในภาวะที่ทำให้เสถียรภาพระบบลดลง ปัญหานี้จะรุนแรงมากขึ้นอีกในอนาคตหากไม่ได้รับการแก้ไข การเพิ่มเสถียรภาพระบบไฟฟ้ากำลังโดยใช้ AVR ควบคุมร่วมกับ PSS เป็นวิธีการที่นิยมใช้กัน แต่วิธีนี้ค่อนข้างไม่มีประสิทธิภาพนักเมื่อนำมาใช้กับระบบไฟฟ้ากำลังชนิดสายส่งระยะไกล ปัจจุบันมีการนำ SVC มาใช้ในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง แต่ด้วยวิธีการควบคุมแบบเดิมที่สมมุติให้ตัวแปรต่างๆในระบบคงที่ ทำให้วิธีนี้ไม่สามารถนำมาใช้ในการควบคุมในภาวะการทำงานที่กว้างได้ [1] และถึงแม้จะมีการเสนอ SVC ชนิดปรับตัวเองได้ (self-tunning SVC) [2] ที่ให้ผลตอบสนองระบบที่ดีกว่าเดิมแต่วิธีการควบคุมชนิดนี้จะเสียเวลาอย่างมากไปกับการคำนวณระบบสมการที่ซับซ้อน เพื่อขจัดปัญหานี้ บทความนี้จึงเสนอวิธีการควบคุม SVC ด้วยวิธีฟัซซี่ที่อาศัยข้อมูลจากเฟสเพลน ด้วยวิธีการที่นำเสนอนี้จะไม่มีการคำนวณสมการที่ซับซ้อน ทำให้การคำนวณสัญญาณควบคุม SVC เป็นไปอย่างรวดเร็ว ผลการจำลองต่างๆในบทความนี้ให้เห็นถึงประสิทธิภาพของวิธีที่นำเสนอ รวมถึงการประยุกต์ใช้วิธีที่เสนอนี้ร่วมกับ PSS ชนิดต่างๆ เช่น ΔP-PSS , Δω-PSS และ Δω-PSS ชนิดฟัซซี่

2. ระบบจำลอง

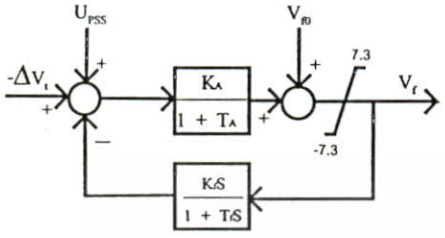


รูปที่ 1 ระบบจำลอง

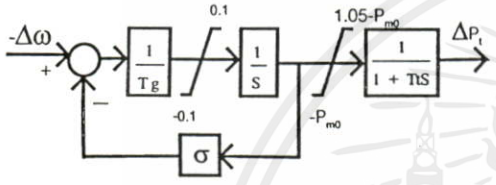
ระบบจำลองที่ใช้ในบทความนี้แสดงไว้ในรูปที่ 1 บล็อกไดอะแกรมของAVR,GOVและSVCแสดงไว้ในรูปที่ 2(a) ,2(b)และ2(c)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

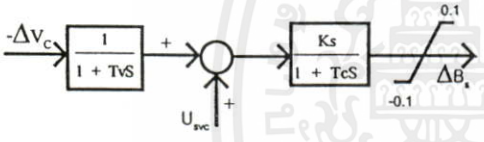
ตามลำดับ จากรูปที่ 1 สวิทช์ 1 ,2 และ 3 สำหรับการควบคุม PSS แบบ ΔP-PSS , Δω-PSS และ Δω-PSS ชนิดฟัซซี่ โดยมีบล็อกไดอะแกรมแสดงไว้ในรูปที่3(a),3(b)และ3(c)บล็อก SVC FUZ เป็นการควบคุมSVC ด้วยวิธีฟัซซี่ตามวิธีที่นำเสนอในบทความนี้ ซึ่งแสดงไว้ในรูปที่ 4



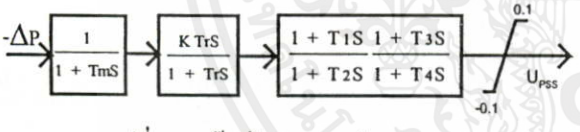
รูปที่ 2(a) บล็อกไดอะแกรม AVR



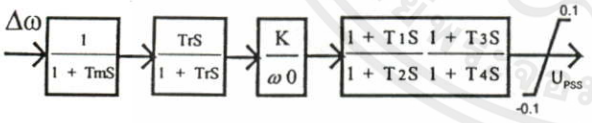
รูปที่ 2(b) บล็อกไดอะแกรม GOV



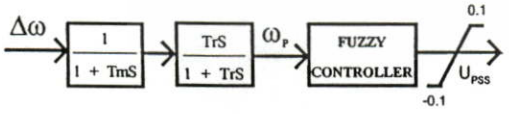
รูปที่ 2(c) บล็อกไดอะแกรม SVC



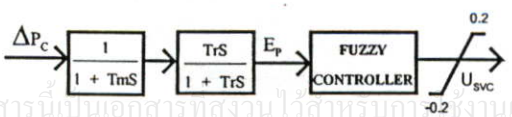
รูปที่ 3(a) บล็อกไดอะแกรม ΔP-PSS



รูปที่ 3(b) บล็อกไดอะแกรม Δω-PSS



รูปที่ 3(c) บล็อกไดอะแกรม Δω-PSS ชนิดควบคุมด้วยฟัซซี่



รูปที่ 4 บล็อกไดอะแกรมควบคุม SVC ด้วยวิธีฟัซซี่

3. การควบคุม SVC ด้วยวิธีฟัซซี่

รูปที่ 4 คือบล็อกไดอะแกรมวิธีการควบคุม SVC ด้วยวิธีฟัซซี่ ภายในบล็อกนี้ $E^{\bullet}_{p}$ คำนวณจากสมการที่ 1

$$\dot{E}_P(k) = \frac{E_P(k) - E_P(k-1)}{T} \qquad (1)$$

k = 1,2,3.....,N เมื่อ T คือเวลาสุ่ม , N คือจำนวนข้อมูล

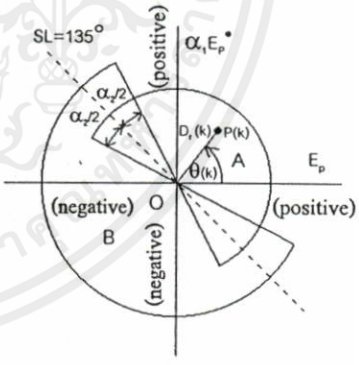
$E_p(k)$และ$\alpha_1 E^{\bullet}_{p}(k)$ เป็นตัวกำหนดจุด P(k) บนเฟสเพลนในรูปที่ 5 จุด O เป็นจุดที่ระบบสมดุล ดังนั้นการทำงานของการควบคุมวิธีนี้จะพยายามเลื่อนตำแหน่งจุด P(k) ให้เข้าสู่จุดศูนย์กลาง O ให้เร็วที่สุดเท่าที่จะทำได้ ข้อมูลต่างๆที่ได้จากเฟสเพลนแสดงไว้ในสมการที่ 2 , 3 และ 4

$$P(k) = D_r(k) \angle \theta(k) \qquad (2)$$

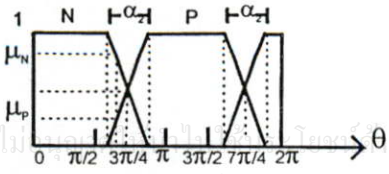
$$D_r(k) = \sqrt{Ep(k)^2 + (\alpha_1 \dot{E}_P(k))^2} \qquad (3)$$

$$\theta(k) = tan^{-1}(\alpha_1 \dot{E}_P(k) / E_P(k)) \qquad (4)$$

พื้นที่บนเฟสเพลนถูกออกเป็น 2 ส่วนคือส่วน A และ B โดยที่พื้นที่ส่วน A เป็นส่วนที่ SVC จ่ายกำลังไฟฟ้ารีแอกทีฟ พื้นที่ส่วน B เป็นส่วนที่ SVC รับกำลังไฟฟ้ารีแอกทีฟ α_2 เป็นมุมทับซ้อนของพื้นที่ทั้งสองส่วน SL คือสวิทช์ซึ่งลายน์(Switching Line)ซึ่งทำมุมทวนเข็มนาฬิกา 135° พื้นที่ทั้งสองนี้ถูกกำหนดโดยฟังก์ชั่นการเป็นสมาชิกของθ คือ $\mu_N(\theta),\mu_p(\theta)$ดังแสดงในรูปที่ 6

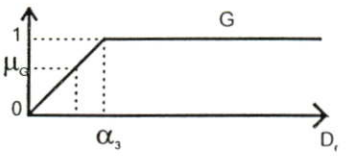


รูปที่ 5 เฟสเพลน

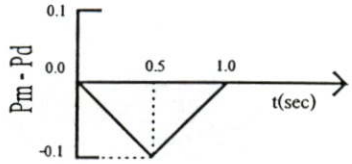


รูปที่ 6 ฟังก์ชั่นการเป็นสมาชิกของθ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7 ฟังก์ชั่นการเป็นสมาชิกของ D_r



รูปที่ 8 การรบกวนแบบเล็กน้อย

จากฟังก์ชั่นการเป็นสมาชิกในรูปที่6และ7 สัญญาณควบคุม U_{svc} สามารถคำนวณได้จากสมการที่ 5

$$U_{SVC}(k) = \frac{\mu_{N(k)} - \mu_{P(k)}}{\mu_{N(k)} + \mu_{P(k)}} \mu_{G(k)} U_{max}$$
$$= (2\mu_{N(k)} - 1)\mu_{G(k)} U_{max} \quad (5)$$

4. การปรับตัวแปรควบคุมฟัซซี่

การที่จะหาตัวแปรควบคุมฟัซซี่ α_1,α_2 และ α_3 จะอาศัยสมการดัชนีการทำงานในสมการที่ 6

$$J_i = \sum_{k=1}^{N} \left(|\Delta\omega(k)| t_k + |\dot{U(k)}| \right) \quad (6)$$

โดยที่ i = 1 คือ การรบกวนแบบรุนแรง

i = 2 คือ การรบกวนแบบเล็กน้อย

และ

$$\dot{U(k)} = \frac{U_{svc}(k) - U_{svc}(k-1)}{T}$$

การรบกวนต่างๆที่ใช้ทดสอบนิยามไว้ดังต่อไปนี้

(1) การรบกวนแบบรุนแรง คือ การเกิดฟอล์ทชนิด 3 เฟส ในตำแหน่ง F ของระบบจำลองในรูปที่ 1 และฟอล์ทนี้ถูกขจัดหลังจาก 0.1 วินาที ผ่านไป

(2) การรบกวนแบบเล็กน้อย คือ การเปลี่ยนแปลงของกำลังอินพุทเครื่องกำเนิดไฟฟ้า ดังแสดงในรูปที่ 8

ในแต่ละรอบการจำลองบนคอมพิวเตอร์ค่าดัชนีการทำงานจะถูกคำนวณโดยการพิจารณาทั้งการรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย ค่า α1,α2 และ α3 สามารถหาได้จากการทำให้สมการ J_{SUM} มีค่าต่ำสุด

$$J_{sum} = J_1 + J_2 \quad (7)$$

5. ผลการจำลอง

ในการจำลองจะใช้ค่าคงที่ต่างๆตามตารางที่ 1 และพารามิเตอร์ควบคุมฟัซซี่ตามตารางที่ 2

ตารางที่1 ค่าคงที่ระบบจำลอง

GEN: T'_{do}=5.6(s), M=7.2(s), D=2.0(pu), X'_d=0.25(pu), X_d=1.62(pu), X_q=1.6(pu), V_t=1.0(pu), V_b=1.0(pu), ω_o=377(rad/s), P_o=1.0(pu)

สายส่ง: X_l=0.14(pu), X_j=0.05(pu), X_{l1}=0.21(pu), X_{l2}=0.30(pu)

AVR: K_A=200, T_A=0.05(s), K_f=0.07, T_f=1.25(s)

GOV: σ=0.05, T_g=0.02(s), T_t=2.0(s)

SVC: K_S=50, T_C=0.05(s), T_V=0.01(s), V_C=1.0(pu)

SVC FUZ: T_m=0.01(s), T_r=4.0(s), T=0.01(s), N=1201

PSS1: K=1.0, T_m=0.01(s), T_r=4.0(s), T_1=0.04(s), T_2=0.05(s), T_3=T_4=0.0(s)

PSS2 : K=13, ω_0=377(rad/s), T_m=0.01(s), T_r=4.0(s), T_1=0.3(s), T_2=0.02(s), T_3=0.2(s), T_4=0.02(s)

PSS3: T_m=0.056(s), T_r=4.0(s)

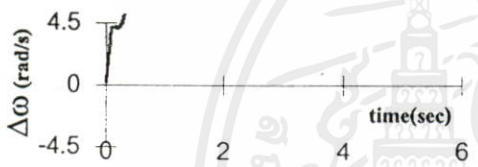
ตารางที่ 2 พารามิเตอร์ควบคุมฟัซซี่

วิธีควบคุม	α_1	α_2	α_3	$\alpha_{\omega 1}$	$\alpha_{\omega 2}$	$\alpha_{\omega 3}$
PSS1+SVC	0.01	90°	0.23	-	-	-
PSS2+SVC	0.08	90°	0.25	-	-	-
PSS3+SVC	0.01	90°	0.23	0.31	11°	0.89

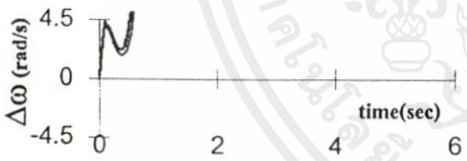
เสถียรภาพของระบบจะพิจารณาจากผลตอบสนองการเปลี่ยนแปลงความเร็วเชิงมุมเครื่องกำเนิดไฟฟ้าภายใต้การรบกวนแบบรุนแรง ทั้งกรณี X_j = 1.0 เท่าและ 1.3 เท่า ผลการจำลองทั้ง 3 กรณีสามารถสรุปได้ดังตารางที่3

ตารางที่ 3 ผลการจำลอง

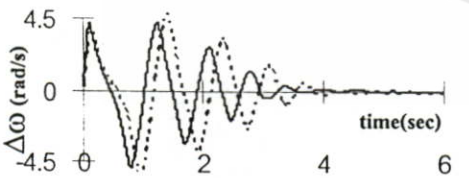
กรณี	วิธีควบคุม	X_j = 1.0 เท่า		X_j = 1.3 เท่า		รูปที่
		เสถียร	ไม่เสถียร	เสถียร	ไม่เสถียร	
1	PSS1		√		√	9
	PSS2		√		√	9
	PSS3		√		√	9
2	SVC (no Usvc)		√		√	10(a)
	SVC	√		√		10(b)
3	PSS1+SVC	√			√	11(a)
	PSS2+SVC	√			√	11(b)
	PSS3+SVC	√		√		11(c)



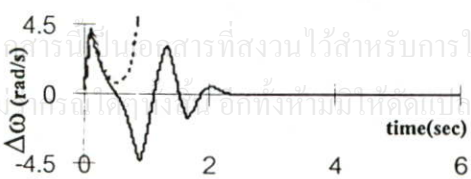
รูปที่ 9 ควบคุมด้วย PSS



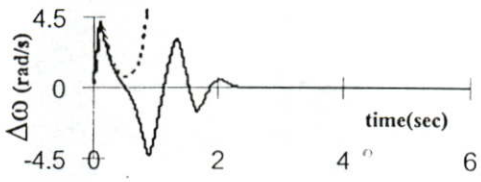
รูปที่ 10(a) ควบคุม SVC โดยไม่มี U_{SVC}



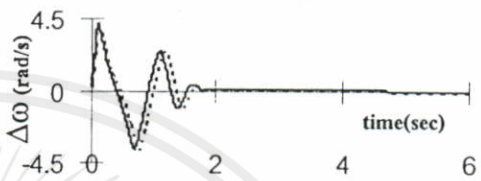
รูปที่ 10(b) ควบคุมSVC ด้วยฟัซซี่



รูปที่ 11(a) ควบคุมSVC ด้วยฟัซซี่ ร่วมกับ ΔP-PSS



รูปที่ 11(b) ควบคุมSVC ด้วยฟัซซี่ ร่วมกับ Δω-PSS



รูปที่ 11(c) ควบคุมSVC ด้วยฟัซซี่ ร่วมกับ Δω-PSS ชนิดฟัซซี่

6. สรุป

วิธีการควบคุม SVC โดยใช้วิธีฟัซซี่ตามวิธีที่ได้นำเสนอนี้ สามารถปรับปรุงผลการตอบสนองทางไดนามิกของระบบไฟฟ้ากำลังได้เป็นอย่างดี และยังมีความไวต่อการเปลี่ยนแปลงของตัวแปรในระบบน้อย นอกจากนี้การใช้วิธีการที่นำเสนอร่วมกับ PSS ชนิดต่างๆ ผลตอบสนองระบบ ณ ที่ค่าออกแบบก็จะดีขึ้นเป็นอย่างมาก แต่ตัวควบคุมก็จะมีความไวต่อการเปลี่ยนแปลงตัวแปรในระบบมากทำให้เสถียรภาพระบบลดลงได้ แต่เมื่อใช้ Δω-PSSชนิดควบคุมด้วยฟัซซี่ทำงานร่วมกับวิธีที่นำเสนอ ผลตอบสนองระบบทั้งใน2กรณีก็จะดีเช่นเดิม ดังนั้นการที่จะประยุกต์ใช้SVCที่ควบคุมด้วยวิธีการที่นำเสนอนี้ร่วมกับ PSS ชนิดต่างๆจะต้องมีการพิจารณาอย่างรอบคอบ มิฉะนั้นอาจทำให้เสถียรภาพของระบบลดลง

กิตติกรรมประกาศ

ผู้เขียนขอขอบคุณสำหรับข้อแนะนำต่างๆที่มีประโยชน์ต่องานวิจัยชิ้นนี้จาก อาจารย์มณฑล ลีลาจินดาไกรฤกษ์ อาจารย์ที่ปรึกษา และ Prof.Yoshibumi Mizutani อาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยโตไก กำลังใจที่ทำให้ผู้เขียนสามารถทำผลงานชิ้นนี้สำเร็จด้วยดี จาก คุณสุมิตรา ศักดิ์ศิรินุกูล และคุณอินทิยา เลิศสุทธิรัศมีวงศ์

เอกสารอ้างอิง

[1] Yao-Nan Yu, Khien Vongsuriya and Leonard N. Wedman , " Application of an Optimal Control Theory to a Power System " ,IEEE Trans.PAS,vol.9,1970

[2] Chi-Jui Wu and Yuan-Yih Hso, "Design of Self-Tuning PID Power System Stabilizer for Multimachine Power Systems",IEEE Trans.Power Systems,Vol3,1988

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

เจเนติกอัลกอริทึ่มสำหรับช่วยออกแบบตัวควบคุมฟัซซี่ของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง

Genetic Algorithm Aided Fuzzy Controller Design of SVC for Improving Power System Stability

คมสันต์ หงษ์สมบัติ มณฑล ลีลาจินดาไกรฤกษ์
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

บทความนี้เสนอวิธีการใหม่ในการออกแบบตัวควบคุมฟัซซี่ของ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง โดยใช้เจเนติกอัลกอริทึ่ม(Genetic Algorithm) แบบจำลองที่ใช้ในบทความนี้ประกอบด้วยเครื่องจักรกลไฟฟ้าซิงโครนัส 1 ตัวต่อกับบัสอนันต์(Infinite Bus) ผ่านสายส่งวงจรคู่และมี SVC ติดตั้งอยู่จุดกึ่งกลางของสายส่ง ตัวควบคุม SVC ออกแบบโดยใช้หลักการการควบคุมแบบฟัซซี่ซึ่งอาศัยข้อมูลที่ได้รับจากเฟสเพลน สัญญาณการเปลี่ยนแปลงและอัตราการเปลี่ยนแปลงกำลังไฟฟ้า ณ จุดที่ติดตั้ง SVC ถูกเลือกเป็นอินพุทสำหรับตัวควบคุมฟัซซี่ โดยที่ตัวแปรควบคุมฟัซซี่จะใช้เจเนติกอัลกอริทึ่มในการค้นหาค่าตัวแปรที่ให้ผลตอบสนองไดนามิกของระบบที่ดีที่สุดหรือใกล้เคียง ผลการจำลองแสดงให้เห็นถึงประสิทธิภาพของวิธีการใหม่ที่นำเสนอ โดยทำการเปรียบเทียบกับผลที่ได้จากการออกแบบโดยใช้วิธีการแบบเก่า รวมทั้งได้แสดงผลการจำลองโดยใช้วิธีการใหม่ที่นำเสนอทำงานร่วมกับตัวควบคุมเสถียรภาพระบบไฟฟ้ากำลัง (PSS) ชนิดต่างๆ

Abstract

This paper presents a novel approach to design SVC fuzzy controller for improving power system stability using Genetic Algorithm(GA). System modeling considered in this paper comprises a synchronous machine connected to an infinite bus through double circuit transmission line and SVC is installed at the mid-point. SVC controller is designed by fuzzy control method and utilizing information from phase plane. Electric power deviation and its change at the location of SVC are chosen as input signals to SVC fuzzy controller and search the optimal fuzzy control parameters to obtain the optimal or near optimal dynamic system performance by using GA. Simulation results demonstrate the effectiveness of this novel proposed control both in case SVC only and in case SVC operated together with serveral types of power system stabilizers(PSS) as well as compare the results of designing by conventional approach with GA approach.

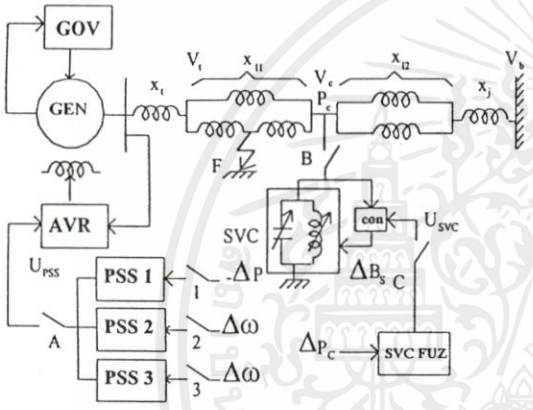
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. บทนำ

จากบทความก่อน [1] พบว่า การควบคุมด้วย PSS อย่างเดียวไม่สามารถทำให้ผลตอบสนองระบบเข้าสู่เสถียรภาพได้ ดังนั้นจึงได้เสนอการปรับปรุงเสถียรภาพโดยใช้ SVC ซึ่งควบคุมด้วยวิธีฟัซซี่ ผลตอบสนองที่ได้สามารถทำให้ระบบเข้าสู่เสถียรภาพได้ แต่วิธีการออกแบบ

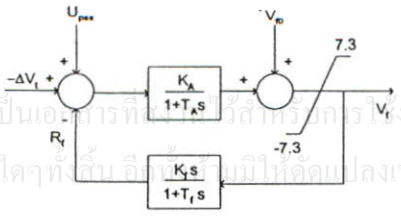
วิธีเดิม จะใช้วิธีไอเทอร์เรชั่น (iteration method) ซึ่งใช้เวลาในการออกแบบประมาณ 130 ชั่วโมง บนเครื่อง PENTIUM 166 MHz บทความนี้จึงได้เสนอวิธีใหม่ในการออกแบบตัวควบคุมฟัซซี่ โดยใช้เจเนติกอัลกอริทึมซึ่งมีการปรับปรุงเจเนติก [2-4] จากการทดสอบพบว่า วิธีใหม่นี้สามารถลดเวลาในการออกแบบลงเหลือ 11 นาที โดยให้ผลตอบสนองที่ดีกว่าหรือเท่ากับการออกแบบด้วยวิธีไอเทอร์เรชั่น

2. ระบบจำลอง

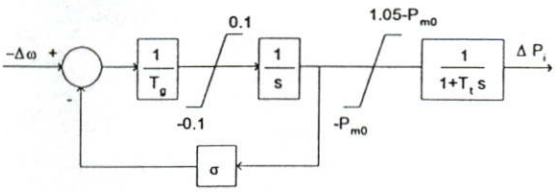


รูปที่ 1 ระบบจำลอง

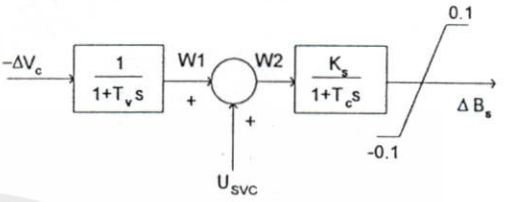
ระบบจำลองที่ใช้ในบทความนี้แสดงไว้ในรูปที่ 1 บล๊อกไดอะแกรมของAVR,GOVและSVCแสดงไว้ในรูปที่ 2(a) ,2(b)และ2(c) ตามลำดับ จากรูปที่ 1 สวิทช์ 1 ,2 และ 3 สำหรับการควบคุม PSS แบบ ΔP-PSS , $\Delta\omega$-PSS และ $\Delta\omega$-PSS ชนิดฟัซซี่ โดยมีบล๊อกไดอะแกรมแสดงไว้ในรูปที่3(a),3(b)และ3(c)บล๊อก SVC FUZ เป็นการควบคุมSVC ด้วยวิธีฟัซซี่ตามวิธีที่นำเสนอในบทความนี้ ซึ่งแสดงไว้ในรูปที่ 4



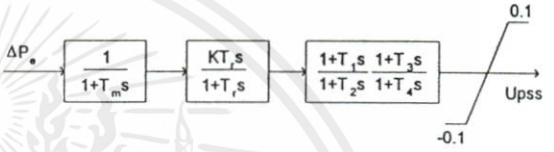
รูปที่ 2(a) บล๊อกไดอะแกรม AVR



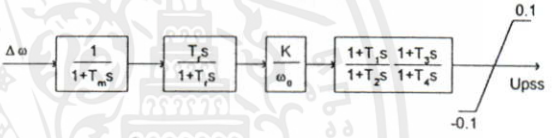
รูปที่ 2(b) บล๊อกไดอะแกรม GOV



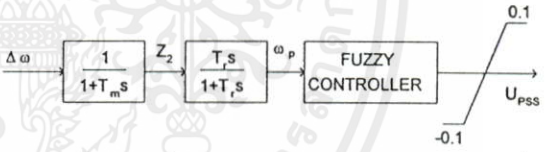
รูปที่ 2(c) บล๊อกไดอะแกรม SVC



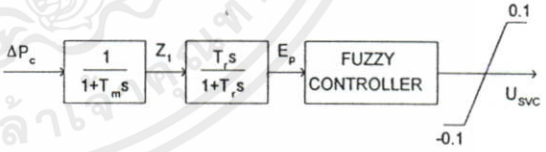
รูปที่ 3(a) บล๊อกไดอะแกรม ΔP-PSS



รูปที่ 3(b) บล๊อกไดอะแกรม $\Delta\omega$-PSS



รูปที่ 3(c) บล๊อกไดอะแกรม $\Delta\omega$-PSS ชนิดฟัซซี่



รูปที่ 4 บล๊อกไดอะแกรมควบคุม SVC ด้วยวิธีฟัซซี่

3. ตัวควบคุมฟัซซี่

รูปที่ 4 คือบล๊อกไดอะแกรมวิธีการควบคุม SVC ด้วยวิธีฟัซซี่ ภายในบล๊อกนี้ $E^{\bullet}_{P}$ คำนวณจากสมการที่ 1

$$E_P^{\bullet}(k) = \frac{E_P(k) - E_P(k-1)}{T} \qquad (1)$$

k = 1,2,3.....,N

T คือเวลาสุ่ม , N คือจำนวนข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

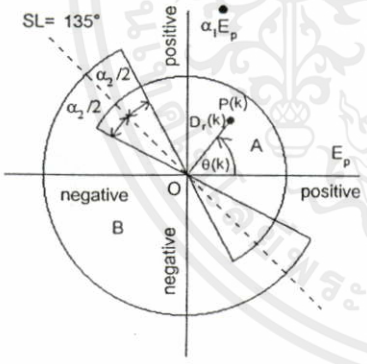
$E_p(k)$และ$\alpha_1 E^{\bullet}_p(k)$ เป็นตัวกำหนดจุด P(k) บนเฟสเพลน ในรูปที่ 5 จุด O เป็นจุดที่ระบบสมดุล ดังนั้นการทำงานของการควบคุมวิธีนี้จะพยายามเลื่อนตำแหน่งจุด P(k) ให้เข้าสู่จุดศูนย์กลาง O ให้เร็วที่สุดเท่าที่จะทำได้ ข้อมูลต่างๆ ที่ได้จากเฟสเพลนแสดงไว้ในสมการที่ 2 , 3 และ 4

$$P(k) = D_r(k) \angle \theta(k) \quad (2)$$

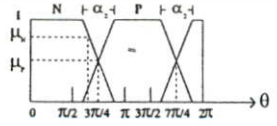
$$D_r(k) = \sqrt{Ep(k)^2 + (\alpha_1 \dot{E}_P(k))^2} \quad (3)$$

$$\theta(k) = tan^{-1}(\alpha_1 \dot{E}_P(k) / E_P(k)) \quad (4)$$

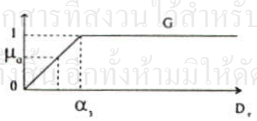
พื้นที่บนเฟสเพลนถูกออกเป็น 2 ส่วนคือส่วนAและB โดยที่พื้นที่ส่วน A เป็นส่วนที่ SVC จ่ายกำลังไฟฟ้ารีแอกทีฟ พื้นที่ส่วน B เป็นส่วนที่ SVC รับกำลังไฟฟ้ารีแอกทีฟ α_2 เป็นมุมทับซ้อนของพื้นที่ทั้งสองส่วน SL คือสวิทชิ่งลายน์ (Switching Line)ซึ่งทำมุมทวนเข็มนาฬิกา 135° พื้นที่ทั้งสองนี้ถูกกำหนดโดยฟังก์ชั่นการเป็นสมาชิกของ θ คือ $\mu_N(\theta), \mu_P(\theta)$ ดังแสดงในรูปที่ 6



รูปที่ 5 เฟสเพลน



รูปที่ 6 ฟังก์ชั่นการเป็นสมาชิกของθ



รูปที่ 7 ฟังก์ชั่นการเป็นสมาชิกของ D_r

จากฟังก์ชั่นการเป็นสมาชิกในรูปที่ 6และ7 สัญญาณควบคุม U_{SVC} สามารถคำนวณได้จากสมการที่ 5

$$U_{SVC}(k) = \frac{\mu_{N(k)} - \mu_{P(k)}}{\mu_{N(k)} + \mu_{P(k)}} \mu_{G(k)U\,max}$$

$$= (2\mu_{N(k)} - 1)\mu_{G(k)U\,max} \quad (5)$$

4. การใช้ GA ค้นหาพารามิเตอร์ควบคุมฟัซซี่

4.1 การเข้ารหัส

พารามิเตอร์ควบคุมฟัซซี่ α_1 , α_2 และ α_3 จะแทนด้วยโครโมโซมซึ่งเข้ารหัสสตริงแบบเกรย์ (Gray code) โดยความยาวยีนส์แต่ละตัวกำหนดโดยสมการที่ 6

$$L \geq \frac{\ln\left[\frac{\alpha\,\max - \alpha\,\min}{\text{step size}}\right]}{\ln 2} \quad (6)$$

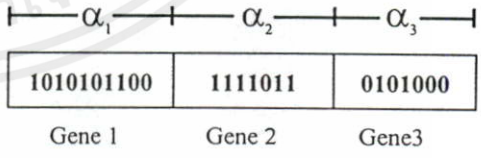
เมื่อ L คือ ความยาวสตริง

α_{max} คือ ค่าขอบเขตบน

α_{min} คือ ค่าขอบเขตล่าง

step size คือ ความละเอียดต่อช่วง เช่น 0.01

สำหรับบทความนี้ใช้ความละเอียดต่อช่วง $\alpha_1 = 0.001$, $\alpha_2 = 1.0$, $\alpha_3 = 0.01$ ความยาวสตริงของแต่ละยีนส์จึงเป็น 10,7,7 ตามลำดับ ดังนั้นโครโมโซมจึงมีความยาวเท่ากับ 24 บิต ดังแสดงในรูปที่ 8



รูปที่ 8 โครโมโซมพารามิเตอร์ควบคุมฟัซซี่

4.2 สมการดัชนีการทำงานและฟังก์ชั่นฟิทเนส

สมการดัชนีการทำงานแสดงในสมการที่ 7 และฟังก์ชั่นฟิทเนสแสดงในสมการที่ 8

$$Min\ J = \sum_{i=1}^{2}(\beta - 1)i + (2 - \beta)\left(\sum_{k=1}^{N}\left(|\Delta\omega(k)|\, t_k + |\dot{v}(k)|\right)\right) \quad (7)$$

$$F = \rho / \left(\sum_{i=1}^{2}(\beta - 1)i + (2 - \beta)\left(\sum_{k=1}^{N}\left(|\Delta\omega(k)|t_k + |\dot{v}(k)|\right)\right)\right) \quad (8)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดย i = 1 คือ การรบกวนแบบรุนแรง

i = 2 คือ การรบกวนแบบเล็กน้อย

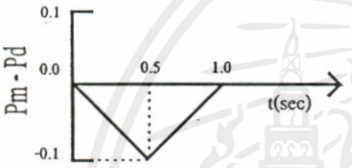
ρ คือ แฟกเตอร์ปรับสเกล β คือ ค่าคงที่ช่วง [0,1] และ

$$\dot{U}(k) = \frac{U_{svc}(k) - U_{svc}(k-1)}{T}$$

การรบกวนต่างๆที่ใช้ทดสอบนิยามไว้ดังต่อไปนี้

(1) การรบกวนแบบรุนแรง คือ การเกิดฟอล์ทชนิด 3 เฟส ในตำแหน่ง F ของระบบจำลองในรูปที่ 1 และฟอล์ทนี้ถูกขจัดหลังจาก 0.1 วินาทีผ่านไป

(2) การรบกวนแบบเล็กน้อย คือ การเปลี่ยนแปลงของกำลังอินพุทเครื่องกำเนิดไฟฟ้า ดังแสดงในรูปที่ 9



รูปที่ 9 การรบกวนแบบเล็กน้อย

4.3 การถอดรหัส

การถอดรหัสโครโมโซมจะแปลงรหัสเกรย์เป็นรหัสไบนารีก่อนแล้วใช้สมการที่ 9 และ 10 แปลงรหัสไบนารีเป็นเลขฐานสิบ โดยแบ่งเป็น 2 กรณี

1.กรณีการเพิ่มแต่ละขั้นเป็นทศนิยม เช่น 0.001

$$\alpha_i = (\text{double})[(\text{int})(X_i(10^P)/2^L)]/10^P \qquad (9)$$

2. กรณีการเพิ่มแต่ละขั้นเป็นจำนวนเต็ม เช่น 1.0

$$\alpha_i = (\text{double})[(\text{int})(X_i(\alpha_{i\,max}-\alpha_{i\,min})/2^L)] \qquad (10)$$

สมการที่ 9 และ 10 จะต้องอยู่ภายใต้เงื่อนไข $2^L \geq 10^P$

โดยที่ i = 1,2,3

X_i คือ ค่าฐานสิบที่ถอดรหัสจากเลขไบนารี่

P คือ ตำแหน่งความละเอียดทศนิยม

4.4 ตัวดำเนินการทางเจเนติกและวิธีการคัดเลือก

ในบทความนี้จะใช้การครอสโอเวอร์แบบยูนิฟอร์มและการมิวเทชั่นแบบบิต โดยมีการปรับปรุงให้ความน่าจะเป็นในการครอสโอเวอร์ลดลงอย่างเป็นเชิงเส้นจาก 0.9 เหลือ 0.6 และความน่าจะเป็นในการมิวเทชั่นเพิ่มขึ้นอย่างเป็นเชิงเส้นจาก 0.01 เป็น 0.1 ซึ่งเป็นไปตามสมการที่ 11 และ 12

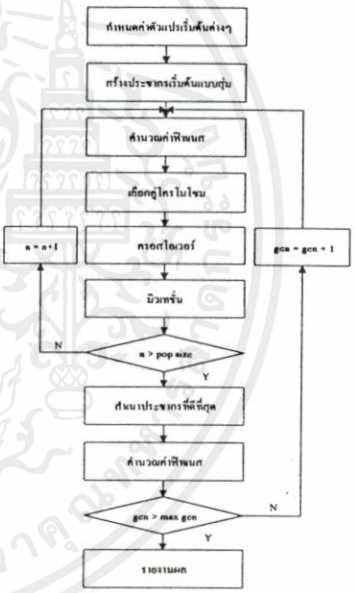
$$Pc(t) = Pc(t-1) - \left[Pc(0) - 0.6\right] / Max\,gen \qquad (11)$$

$$Pm(t) = Pm(t-1) + \left[0.1 - Pm(0)\right] / Max\,gen \qquad (12)$$

สำหรับวิธีการคัดเลือกจะใช้วิธีทัวร์นาเมนท์ โดยใช้ขนาดของทัวร์นาเมนท์ (q) เท่ากับ 2

4.5 ขั้นตอนการใช้ GA หาค่าพารามิเตอร์ควบคุมฟัซซี่

ขั้นตอนต่างๆในการใช้ GA ค้นหาพารามิเตอร์ควบคุมฟัซซี่ของ SVC แสดงดังโฟลชาร์จรูปที่ 10



รูปที่ 10 โฟลชาร์จขั้นตอนการออกแบบด้วย GA

5. ผลการทดสอบโดยจำลองด้วยคอมพิวเตอร์

ข้อมูลที่เป็นค่าคงที่ของระบบจำลองแสดงตารางที่ 1 ผลการทดสอบจะเปรียบเทียบผลที่ได้จากการออกแบบด้วยวิธีไอเทอร์เรชั่นและวิธี GA โดยจะสมมุติให้เกิดเหตุการณ์การรบกวนแบบรุนแรง วิธีที่ดีที่สุดจะให้ค่าดัชนีทำงาน J มีค่าต่ำสุด ผลการทดสอบกรณีต่างๆแสดงในตารางที่ 2 โดย GA ใช้ขนาดประชากรเท่ากับ 30 , จำนวนเจนเนอเรชั่นสูงสุด 50, q = 2 ,ρ = 2500 , β = 0.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารต้นฉบับทุกครั้งที่มีการนำไปใช้

ตารางที่ 1 ค่าคงที่ระบบจำลอง

GEN: T'_{do}=5.6(s),M=7.2(s),D=2.0(pu),X'_d=0.25(pu), X_d=1.62(pu),X_q=1.6(pu),V_t=1.0(pu),V_b=1.0(pu), ω_o=377(rad/s), P_o=1.0(pu)

สายส่ง: X_t=0.14(pu),X_j=0.05(pu),X_{l1}=0.21(pu), X_{l2}=0.30(pu)

AVR: K_A=200,T_A=0.05(s),K_f=0.07,T_f=1.25(s)

GOV: σ=0.05,T_g=0.02(s),T_t=2.0(s)

SVC: K_S=50, T_C=0.05(s),T_V=0.01(s),V_C=1.0(pu)

SVC FUZ: T_m=0.01(s), T_r=4.0(s),T=0.01(s),N=1201

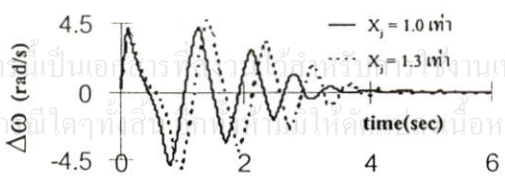
PSS1:K=1.0,T_m=0.01(s),T_r=4.0(s),T_1=0.04(s),T_2=0.05(s) ,T_3=T_4=0.0(s)

PSS2:K=13,ω_0=377(rad/s),T_m=0.01(s),T_r=4.0(s), T_1=0.3(s), T_2=0.02(s),T_3=0.2(s),T_4=0.02(s)

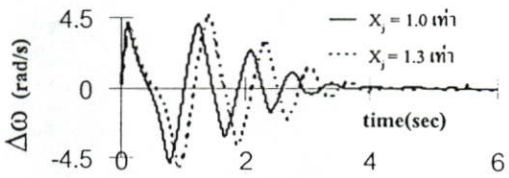
PSS3: T_m=0.056(s),T_r=4.0(s)

การปรับ X_j เป็น 1.3 เท่า เป็นการทดสอบความไวของตัวควบคุมต่อการเปลี่ยนแปลงอิมพีแดนซ์ในโครงข่าย รูปที่ 11 แสดงผลตอบสนองระบบก่อนมีการปรับปรุงเสถียรภาพ โดยจะใช้การควบคุมด้วย PSS อย่างเดียว

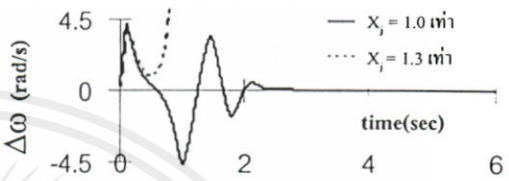
รูปที่ 11 เมื่อควบคุมด้วย PSS อย่างเดียว



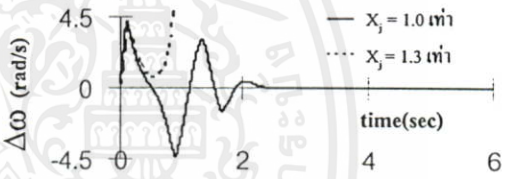
รูปที่ 12(a) SVC ออกแบบด้วยวิธีไอเทอร์เรชั่น



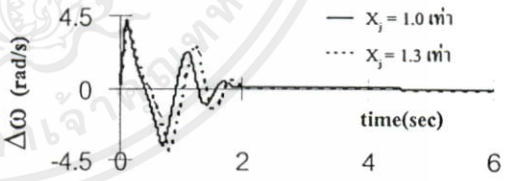
รูปที่ 12(b) SVC ออกแบบด้วยวิธี GA



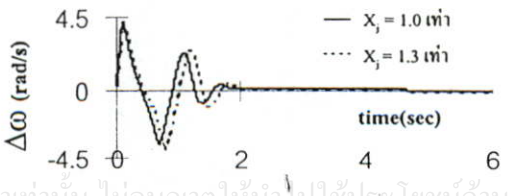
รูปที่ 13 PSS1+SVC ออกแบบวิธีไอเทอร์เรชั่นและ GA



รูปที่ 14 PSS2+SVC ออกแบบวิธีไอเทอร์เรชั่นและ GA



รูปที่ 15(a) PSS3+SVC ออกแบบวิธีไอเทอร์เรชั่น



รูปที่ 15(b) PSS3+SVC ออกแบบด้วยวิธี GA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

ตารางที่ 2 ผลการทดสอบโดยเปรียบเทียบระหว่างวิธีไอเทอร์เรชั่นและวิธี GA

กรณี	วิธีควบคุม	วิธีออกแบบ	α_1	α_2	α_3	J (1.0X$_s$)	J (1.3X$_s$)	รูปที่
1	SVC	ไอเทอร์เรชั่น	0.01	90°	0.23	2313.1667	3114.2698	12(a)
		GA	0.012	90°	0.24	2246.3223	2914.5310	12(b)
2	PSS1+SVC	ไอเทอร์เรชั่น	0.01	90°	0.23	1267.3309	387800.3114	13
		GA	0.01	90°	0.23	1267.3309	387800.3114	13
3	PSS2+SVC	ไอเทอร์เรชั่น	0.01	85°	0.24	1089.6945	408544.8784	14
		GA	0.01	85°	0.24	1089.6945	408544.8784	14
4	PSS3+SVC	ไอเทอร์เรชั่น	0.01	90°	0.29	819.8666	984.7628	15(a)
		GA	0.012	89°	0.29	810.6910	961.5888	15(b)

หมายเหตุ PSS3 ใช้ $\alpha_{\omega_1} = 0.31$, $\alpha_{\omega_2} = 11^\circ$, $\alpha_{\omega_3} = 0.89$

6. สรุป

จากผลการทดลองจะพบว่าวิธีการออกแบบตัวควบคุมฟัซซี่ของ SVC เพื่อนำปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังโดยใช้ GA นี้ให้ผลที่ดีกว่าหรือเท่ากับวิธีการออกแบบด้วยวิธีไอเทอร์เรชั่น แต่จุดเด่นของวิธีการออกแบบโดยใช้ GA คือความเร็วในการค้นหาค่าพารามิเตอร์ที่ดีที่สุดสำหรับการควบคุมฟัซซี่ ซึ่งจากจุดนี้เอง วิธีการที่นำเสนอนี้จะเป็นเครื่องมือในการพัฒนาเทคนิคการควบคุม SVC ที่ให้ผลตอบสนองที่ดีกว่าเดิมต่อไป

กิตติกรรมประกาศ

งานวิจัยชิ้นนี้สำเร็จได้ด้วยดีจากคำแนะนำที่มีประโยชน์ของ รศ.มณฑล ลีลาจินดาไกรฤกษ์ ขอขอบคุณ คุณ สุมิตรา ศักดิ์ศิรินุกูล ที่เป็นแรงใจของผู้เขียนเสมอมา

เอกสารอ้างอิง

[1] คมสันต์ หงษ์สมบัติ และ มณฑล ลีลาจินดาไกรฤกษ์, "การปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วยตัวชดเชยกำลังไฟฟ้าแบบสถิตย์โดยใช้วิธีฟัซซี่", การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 21, 2541. หน้า 397-400.

[2] D.E.Goldberg, "Genetic Algorithm in Search , Optimization and Machine Learning", Addison-Wesley, New York,1989.

[3] K.F.Man, K.S.Tang, S.Kwong and W.A.Halang, "Genetic Algorithms for Control and Signal Processing", Springer, New York, 1997.

[4] S.Mahfoud, "An Analysis of Boltzman Tournament Selection", IlliGAL Report 91007, University of Illinois at Urbana-Champaign, IL, 1991.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การปรับค่าพารามิเตอร์ควบคุมให้เหมาะสมสูงสุดของตัวควบคุม $\Delta\omega$-PSS และตัวควบคุมฟัซซี่ของ SVC สำหรับการควบคุมเสถียรภาพระบบไฟฟ้ากำลัง

Optimum Parameter Tuning of $\Delta\omega$-PSS and SVC based on Fuzzy Controller for Power System Stability Control

คมสันต์ หงษ์สมบัติ มณฑล ลีลาจินดาไกรฤกษ์
ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทนำ

บทความนี้เสนอวิธีการหนึ่งในการปรับค่าพารามิเตอร์ควบคุมของระบบควบคุมเสถียรภาพระบบไฟฟ้ากำลังแบบผสมผสานของ $\Delta\omega$-PSS และตัวควบคุมชนิดฟัซซี่ของตัวชดเชยกำลังไฟฟ้าแบบสถิตย์ (SVC) เพื่อให้ได้ผลตอบสนองไดนามิกที่ดีที่สุด โดยใช้เจเนติกอัลกอริทึม(GA) ปรากฏการณ์ต่างๆของเสถียรภาพเครื่องกำเนิดไฟฟ้าซิงโครนัสภาย ใต้การรบกวนแบบรุนแรงและการรบกวนแบบเล็กน้อย จะถูกจำลองโดยใช้แบบจำลองเครื่องกำเนิดไฟฟ้าซิงโครนัสเดี่ยวต่อกับอินฟินิทบัสผ่านสายส่งวงจรคู่ ผลการจำลองแสดงให้เห็นถึงประสิทธิภาพของวิธีการใหม่ที่นำเสนอ นอกจากนี้ยังพิจารณาผลของความไวต่อการเปลี่ยนแปลงของตัวควบคุมเมื่อทำการปรับขีดจำกัดเสถียรภาพ

Abstract

This paper discribes a technique for tuning parameter of combination power system stability controller, $\Delta\omega$-PSS operated together with SVC based on fuzzy control, in order to obtain the optimum dynamic response by using Genetic Algorithm (GA). The phenomena of stability of synchronous generator under small and large disturbance are simulated by using a single machine connected to an infinite bus through a double circuit transmission line. These simulation results demonstrate the effectiveness of this proposed method for parameter tuning and consider the sensitivity of this controller when stability limit is adjusted.

1. บทนำ

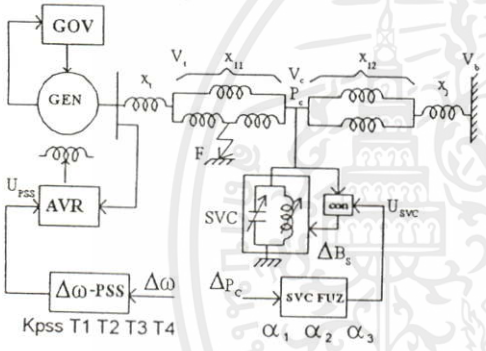
การรักษาเสถียรภาพของระบบไฟฟ้ากำลังจะประกอบด้วยตัวควบคุมชนิดต่างๆมากมาย ซึ่งอาจเป็นทั้งตัวควบคุมแบบเกนคงที่(Fixed gain) หรือตัวควบคุมชนิดฟัซซี่ การที่จะหาพารามิเตอร์ควบคุมที่ให้ผลตอบสนองดีที่สุดนั้นจึงเป็นยาก เนื่องจากไม่สามารถใช้การคำนวณด้วยสมการคณิตศาสตร์ ตัวอย่างการศึกษาเสถียรภาพของเครื่องกำเนิดไฟฟ้าซิงโครนัสสำหรับบทความนี้ จะใช้แบบจำลองเครื่องกำเนิดไฟฟ้าซิงโครนัสเดี่ยวต่อกับอินฟินิทบัส โดยตัวควบคุมแบบเกนคงที่ คือ $\Delta\omega$-PSS ซึ่งทำหน้าที่สร้างทอร์กหน่วงเสริมเข้าไปในระบบ [2] ส่วนตัวควบคุมฟัซซี่จะเป็นตัวสร้างสัญญาณควบคุมเสริมเพิ่มเข้าไปในลูปควบคุมแรงดันของ SVC เพื่อสร้างทอร์กหน่วงในระบบให้สูงขึ้น [1] วิธีเดิมในการตั้งค่าพารามิเตอร์ควบคุม จะเริ่มจากการคำนวณค่าพารามิเตอร์ควบคุมของ $\Delta\omega$-PSS ที่เหมาะสมโดยตัดผลของ SVC ออกจากระบบ จากนั้นทำการติดตั้ง SVC แล้วปรับแต่งพารามิเตอร์ควบคุมฟัซซี่จนได้ผลตอบสนองระบบที่ดีที่สุด ซึ่งการใช้วิธีนี้อาจไม่ได้ผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

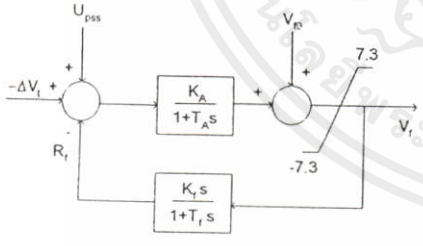
ตอบสนองที่ดีที่สุดจริงๆ บทความนี้จึงเสนอเทคนิคใหม่ในการปรับแต่งพารามิเตอร์ควบคุมของตัวควบคุมทั้งสองชนิดไปพร้อมกันในลักษณะขนาน โดยใช้เจเนติกอัลกอริทึม [3,4] เพื่อให้ได้ผลตอบสนองระบบที่ดีขึ้นกว่าเดิม ผลการทดสอบด้วยการจำลองบนคอมพิวเตอร์แสดงให้เห็นถึงประสิทธิภาพของวิธีการใหม่ที่นำเสนอ ซึ่งจะเป็นแนวทางในการพัฒนาเทคนิคการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังให้ดีต่อไปในอนาคต

2. ระบบจำลอง

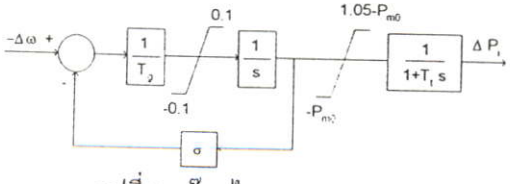
รูปที่1 แสดงแบบจำลองระบบ บล๊อกไดอะแกรมของ AVR ,GOV, SVC, $\Delta\omega$-PSS และ SVC FUZ แสดงในรูปที่ 2 ,3,4,5 และ 6 ตามลำดับ



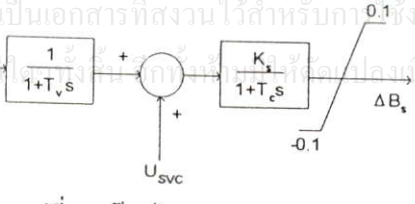
รูปที่ 1 แบบจำลองระบบ



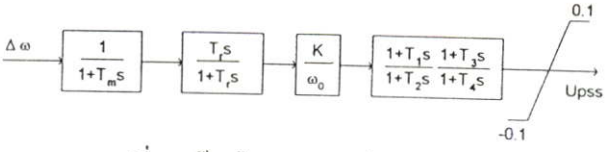
รูปที่ 2 บล๊อกไดอะแกรม AVR



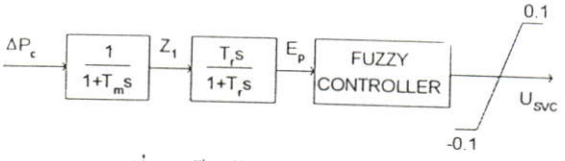
รูปที่ 3 บล๊อกไดอะแกรม GOV



รูปที่ 4 บล๊อกไดอะแกรม SVC



รูปที่ 5 บล๊อกไดอะแกรม $\Delta\omega$-PSS



รูปที่ 6 บล๊อกไดอะแกรม SVC FUZ

3. ตัวควบคุมฟัซซี่

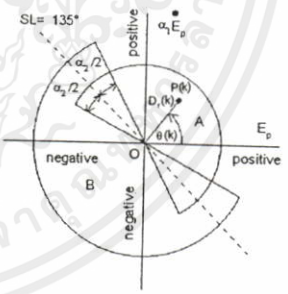
ตัวควบคุมฟัซซี่ (SVC FUZ) แสดงในรูปที่ 6 โดยมีสัญญาณ E_p เป็นอินพุท จากนั้นคำนวณหา E_p ตามสมการที่ 1

$$\dot{E}_p(k) = \frac{E_p(k) - E_p(k-1)}{T} \qquad (1)$$

k = 1,2,3.....,N เมื่อ T คือเวลาสุ่ม

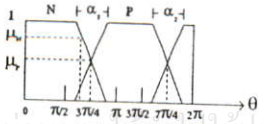
N คือจำนวนข้อมูล

$E_p(k)$ และ $\alpha_1 \dot{E}_p(k)$ เป็นตัวกำหนดจุด P(k) บนเฟสเพลน ในรูปที่ 7

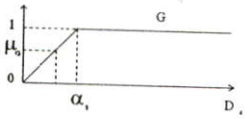


รูปที่ 7 เฟสเพลน

ข้อมูลจากเฟสเพลนจะเป็นตัวกำหนดค่าฟังก์ชั่นการเป็นสมาชิกตามรูปที่ 8 และ 9



รูปที่ 8 ฟังก์ชั่นการเป็นสมาชิกของθ



รูปที่ 9 ฟังก์ชั่นการเป็นสมาชิกของθ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากฟังก์ชั่นการเป็นสมาชิกทั้งในรูปที่ 8,9 สัญญาณควบคุมเสริม U_{SVC} คำนวณได้จากสมการที่ 2

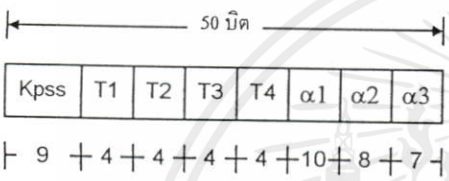
$$U_{SVC}(k) = \frac{\mu_{N(k)} - \mu_{P(k)}}{\mu_{N(k)} + \mu_{P(k)}} \mu_{G(k)U\,max}$$

$$= (2\mu_{N(k)} - 1)\mu_{G(k)U\,max} \qquad (2)$$

4. การใช้ GA ปรับแต่งพารามิเตอร์ควบคุม

4.1 โครงสร้างโครโมโซม

บทความนี้ใช้โครโมโซมยาว 50 บิต เข้ารหัสตัวแปร 8 ตัวด้วยรหัสแบบเกรย์ (Gray Code) [3] โดยมีโครงสร้างโครโมโซมดังกล่าวดังรูปที่ 10



รูปที่ 10 โครงสร้างโครโมโซม

ขอบเขตของ $\alpha 1$ = 0-1.0, $\alpha 2 = 0^{\circ}$-180°, $\alpha 3$ = 0-1.0, T1 = T3 = 0.2-1.5, T2 = T4 =0.02-0.15

4.2 สมการดัชนีการทำงานและฟังก์ชั่นฟิทเนส

สมการดัชนีการทำงานแสดงในสมการที่ 3 และฟังก์ชั่นฟิทเนสแสดงในสมการที่ 4

$$Min\,J = \sum_{i=1}^{2}(\beta-1)i + (2-\beta)\left(\sum_{k=1}^{N}\left(\left|\Delta\omega(k)\right| t_k + \left|\dot{U(k)}\right|\right)\right) \qquad (3)$$

$$F = \rho \Big/ \left(\sum_{i=1}^{2}(\beta-1)i + (2-\beta)\left(\sum_{k=1}^{N}\left(\left|\Delta\omega(k)\right| t_k + \left|\dot{U(k)}\right|\right)\right)\right) \qquad (4)$$

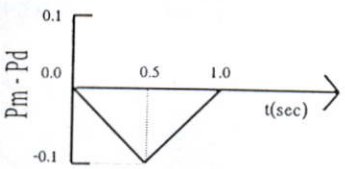
โดยที่ i = 1,2 คือ การรบกวนแบบรุนแรงและเล็กน้อยตามลำดับ ρ คือ แฟกเตอร์ปรับสเกล βคือ ค่าคงที่ช่วง [0,1] และ

$$\dot{U}(k) = \frac{U_{SVC}(k) - U_{SVC}(k-1)}{T}$$

การรบกวนต่างๆที่ใช้ทดสอบนิยามไว้ดังต่อไปนี้

(1) การรบกวนแบบรุนแรง คือ การเกิดฟอล์ทชนิด 3 เฟส ในตำแหน่ง F ของระบบจำลองในรูปที่ 1 และฟอล์ทนี้ถูกขจัดหลังจาก 0.1 วินาทีผ่านไป

(2) การรบกวนแบบเล็กน้อย คือ การเปลี่ยนแปลงของกำลังอินพุทเครื่องกำเนิดไฟฟ้า ดังแสดงในรูปที่ 11



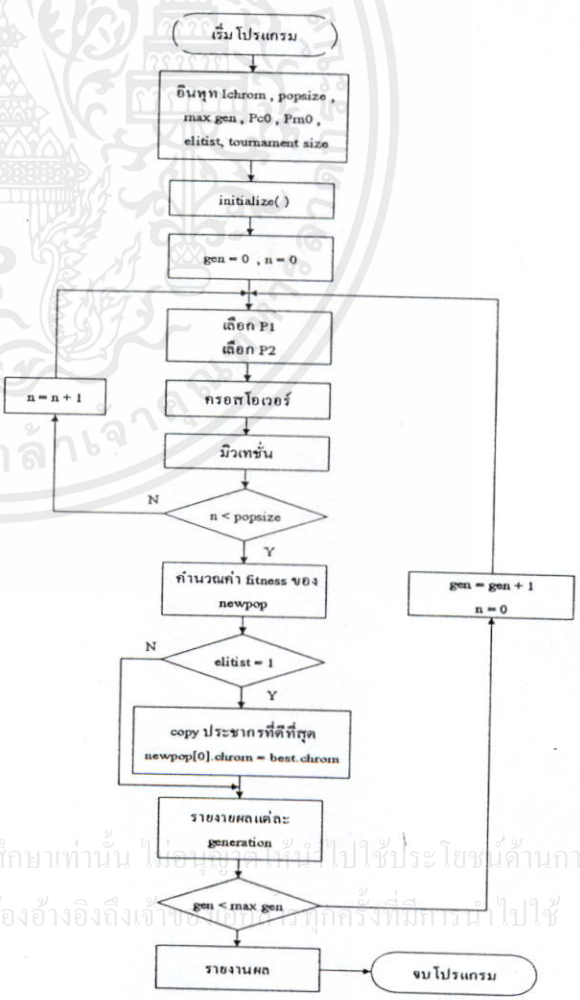
รูปที่ 11 การรบกวนแบบเล็กน้อย

4.2 การปรับปรุงเจเนติก

เพื่อให้โปรแกรมที่เขียนขึ้นมีประสิทธิภาพจึงได้มีการปรับปรุงเจเนติก โดยใช้วิธีการครอสโอเวอร์แบบยูนิฟอร์ม [3] การคัดเลือกแบบทัวร์นาเมนท์ [4] การปรับความน่าจะเป็นในการครอสโอเวอร์และมิวเทชั่นเพิ่มขึ้นและลดลงตามเจนเนอเรชั่น การเก็บรักษาโครโมโซมที่ดีที่สุดไว้ 1 ตัว (Elitist) ในทุกเจนเนอเรชั่น

4.3 การทำงานของโปรแกรม GA

การปรับแต่งพารามิเตอร์ควบคุมด้วยโปรแกรม GA แสดงตามโฟลชาจรูปที่ 12



รูปที่ 12 ขั้นตอนการทำงานของโปรแกรม GA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. ผลการทดสอบ

ในการทดสอบจะใช้ค่าคงที่ต่างๆของระบบจำลองตามตารางที่ 1 โดย GA ใช้ขนาดประชากรเท่ากับ 30, จำนวนเจนเนอเรชั่นสูงสุด 50, P_c = 0.9-0.6, P_m = 0.01-0.1, q = 2,ρ = 2500, β = 0.1

ตารางที่ 1 ค่าคงที่ระบบจำลอง

GEN: T'_{do}=5.6(s),M=7.2(s),D=2.0(pu),X'_d=0.25(pu), X_d=1.62(pu),X_q=1.6(pu),V_t=1.0(pu),V_b=1.0(pu), ω_o=377 (rad/s), P_o=1.0(pu)

สายส่ง: X_t=0.14(pu),X_j=0.05(pu),X_{l1}=0.21(pu), X_{l2}=0.30 (pu)

AVR: K_A=200,T_A=0.05(s),K_f=0.07,T_f=1.25(s)

GOV: σ=0.05,T_g=0.02(s),T_t=2.0(s)

SVC: K_S=50, T_C=0.05(s),T_V=0.01(s),V_C=1.0(pu)

SVC FUZ: T_m=0.01(s), T_r=4.0(s),T=0.01(s),N=1201

ตารางที่ 2 ผลทดลองเปรียบเทียบวิธีเก่ากับวิธีที่นำเสนอ

พารามิเตอร์	ค่า (ใหม่)	พารามิเตอร์	ค่า (เก่า)
α1	0.006	α1	0.012
α2	93°	α2	90°
α3	0.51	α3	0.24
Kpss	13.1	Kpss	13
T1	0.3	T1	0.3
T2	0.1	T2	0.02
T3	0.5	T3	0.2
T4	0.06	T4	0.02
J(1.0)new	841.64	J(1.0)old	1089.74
J(1.246)new	917.64	J(1.246)old	2840.22

ผลทดสอบเปรียบเทียบค่าพารามิเตอร์ควบคุมทั้ง 2 วิธี แสดงในตารางที่ 2 รูปที่ 13 แสดงผลตอบสนองต่างๆของระบบเมื่อ X_j = 1.0 เท่า โดย (a)-(e) ภายใต้การรบกวนแบบรุนแรง และ (f)-(j) ภายใต้การรบกวนแบบเล็กน้อย ผลของความไวของตัวควบคุมเมื่อปรับขีดจำกัดเสถียรภาพลดลง ทำโดยปรับ X_j เป็น 1.246 เท่า ดังแสดงในรูปที่ 14 โดย (a)-(e) ภายใต้การรบกวนแบบรุนแรง และ รูป (f)-(j) ภายใต้การรบกวนแบบเล็กน้อย

6. สรุป

การปรับพารามิเตอร์ควบคุมด้วยวิธีการที่นำเสนอสามารถใช้กับระบบที่มีตัวควบคุมหลายประเภท โดยจะทำการปรับพารามิเตอร์ควบคุมไปพร้อมกันทั้งหมด เพื่อให้ได้ผลตอบสนองสุดท้ายดีที่สุด วิธีที่นำเสนอได้ประยุกต์ใช้ในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วย SVC กับ $\Delta\omega$-PSS ซึ่งพบว่า วิธีที่นำเสนอนี้สามารถปรับปรุงผลตอบสนองได้ดีขึ้นและความไวของตัวควบคุมเมื่อปรับขีดจำกัดเสถียรภาพมีค่าลดลง ทำให้ตัวควบคุมสามารถทำงานได้ในย่านกว้างมากขึ้นโดยการทำงานยังคงดีเหมือนเดิม วิธีที่นำเสนอนี้จะเป็นแนวทางในการปรับปรุงเทคนิคในการปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังที่ดีขึ้นต่อไปในอนาคต

7. กิตติกรรมประกาศ

งานวิจัยชิ้นนี้สำเร็จได้ด้วยดีจากคำแนะนำที่มีประโยชน์ของ รศ.มณฑล ลีลาจินดาไกรฤกษ์ ขอขอบคุณ คุณ สุมิตรา ศักดิ์ศิรินุกูล ที่เป็นกำลังใจของผู้เขียนเสมอมา

8. เอกสารอ้างอิง

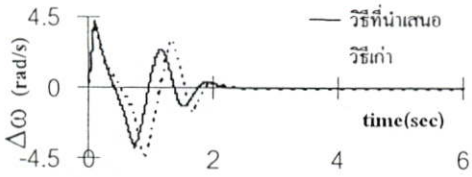
[1] คมสันต์ หงษ์สมบัติ และ มณฑล ลีลาจินดาไกรฤกษ์, "การปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วยตัวชดเชยกำลังไฟฟ้าแบบสถิตย์โดยใช้วิธีฟัซซี่", การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 21, 2541, หน้า 397-400.

[2] P.W.Sauer and M.A.Pai, "Power System Dynamics and Stability", Prentice Hall, New Jersey, 1998.

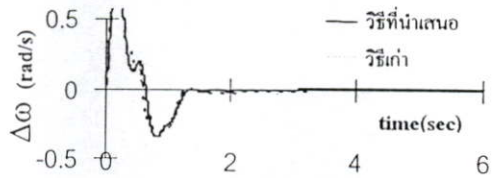
[3] K.F.Man, K.S.Tang, S.Kwong and W.A.Halang, "Genetic Algorithms for Control and Signal Processing", Springer, New York, 1997.

[4] S.Mahfoud, "An Analysis of Boltzman Tournament Selection", IlliGAL Report 91007, University of Illinois at Urbana-Champaign, IL,1991.

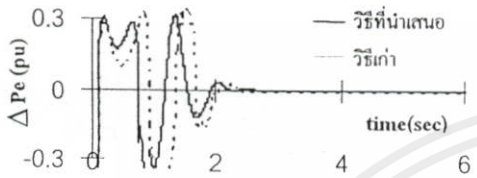
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้



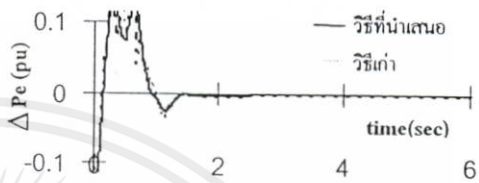
(a) ผลตอบสนอง $\Delta\omega$ (การรบกวนแบบรุนแรง)



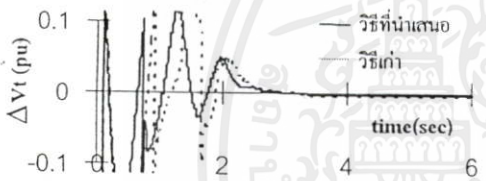
(f) ผลตอบสนอง $\Delta\omega$ (การรบกวนแบบเล็กน้อย)



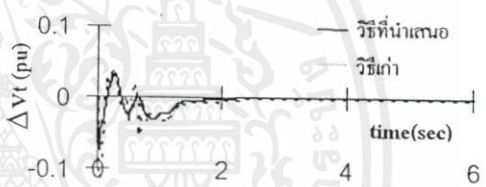
(b) ผลตอบสนอง ΔPe (การรบกวนแบบรุนแรง)



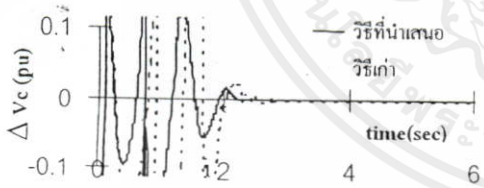
(g) ผลตอบสนอง ΔPe (การรบกวนแบบเล็กน้อย)



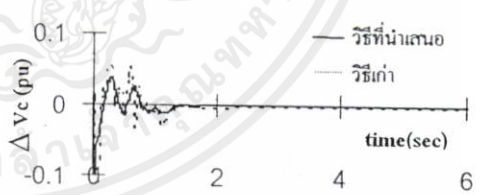
(c) ผลตอบสนอง ΔVt (การรบกวนแบบรุนแรง)



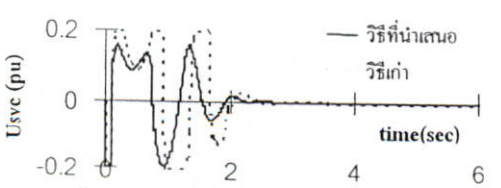
(h) ผลตอบสนอง ΔVt (การรบกวนแบบเล็กน้อย)



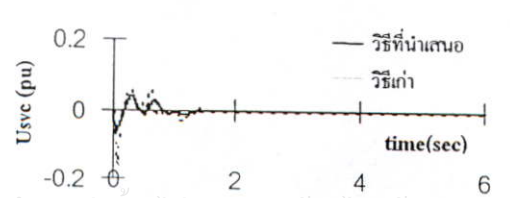
(d) ผลตอบสนอง ΔVc (การรบกวนแบบรุนแรง)



(i) ผลตอบสนอง ΔVc (การรบกวนแบบเล็กน้อย)



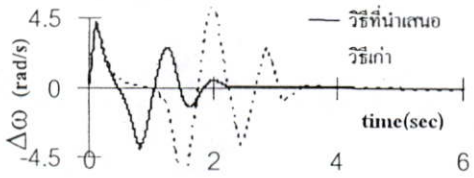
(e) ผลตอบสนอง Usvc (การรบกวนแบบรุนแรง)



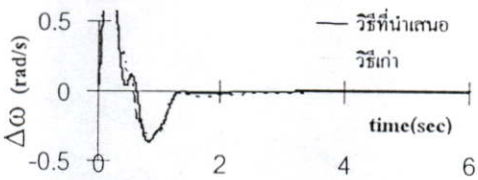
(j) ผลตอบสนอง Usvc (การรบกวนแบบเล็กน้อย)

รูปที่ 13 ผลตอบสนองต่างๆเมื่อ Xj = 1.0 เท่า

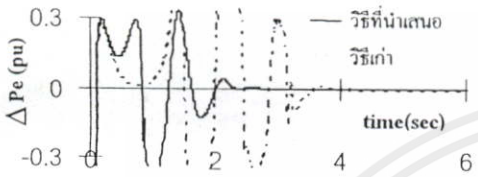
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



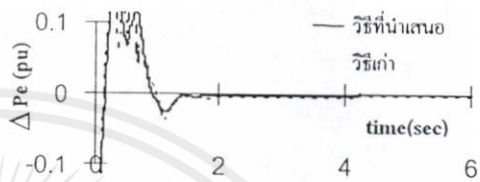
(a) ผลตอบสนอง $\Delta\omega$ (การรบกวนแบบรุนแรง)



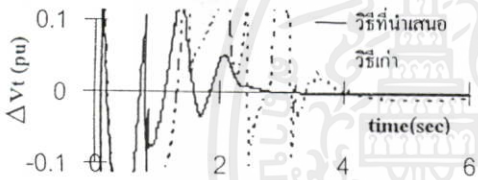
(f) ผลตอบสนอง $\Delta\omega$ (การรบกวนแบบเล็กน้อย)



(b) ผลตอบสนอง ΔPe (การรบกวนแบบรุนแรง)

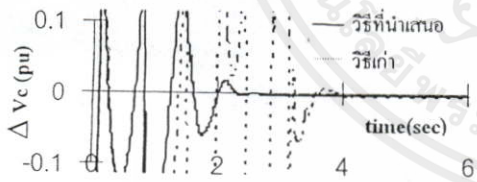


(g) ผลตอบสนอง ΔPe (การรบกวนแบบเล็กน้อย)

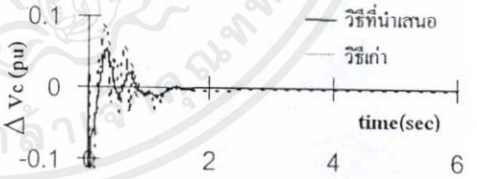


(c) ผลตอบสนอง ΔVt (การรบกวนแบบรุนแรง)

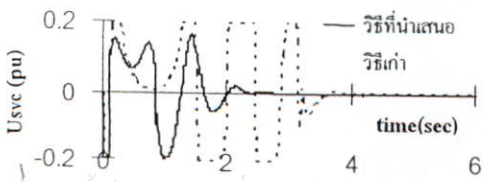
(h) ผลตอบสนอง ΔVt (การรบกวนแบบเล็กน้อย)



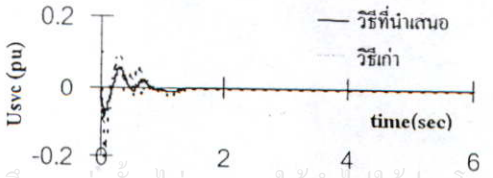
(d) ผลตอบสนอง ΔVc (การรบกวนแบบรุนแรง)



(i) ผลตอบสนอง ΔVc(การรบกวนแบบเล็กน้อย)



(e) ผลตอบสนอง Usvc (การรบกวนแบบรุนแรง)



(j) ผลตอบสนอง Usvc (การรบกวนแบบเล็กน้อย)

รูปที่ 14 ผลตอบสนองต่างๆเมื่อ Xj = 1.246 เท่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประวัติผู้เขียน

นายคมสันต์ หงษ์สมบัติ เกิดเมื่อวันที่ 18 กุมภาพันธ์ 2519 ที่จังหวัดศรีสะเกษ สำเร็จการศึกษาวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมไฟฟ้ากำลัง (เกียรตินิยมอันดับ 1) จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2539 โดยมีผลการเรียนเป็นอันดับ 1 ของภาควิชา โครงงานในระดับปริญญาตรี คือ เครื่องชุบผิวแข็งโลหะโดยใช้หลักการเหนี่ยวนำความถี่สูง

ปี 2539 เข้าทำงานในตำแหน่งวิศวกรไฟฟ้า ส่วนซ่อมบำรุง โรงงานปูนซิเมนต์เขาวง จังหวัดสระบุรี บริษัทปูนซิเมนต์ไทย จำกัด มหาชน ระหว่างที่ทำงานอยู่นั้น ได้พัฒนาโปรแกรมการจัดสรรเวลาเดินเครื่องจักรโรงงานปูนซิเมนต์ เพื่อลดค่าไฟฟ้า และศึกษาความคุ้มทุนโดยเปรียบเทียบค่าใช้จ่ายในการเสียค่าไฟฟ้าตามระบบ TOD และ TOU

ในปีเดียวกันนี้ ได้เข้าศึกษาต่อระดับปริญญาโท สาขาวิชาวิศวกรรมไฟฟ้า ณ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง มีผลงานวิชาการตีพิมพ์ทางด้านวิศวกรรมไฟฟ้าระหว่างศึกษาปริญญาโท ในวารสารประชุมวิชาการวิศวกรรมไฟฟ้า ครั้งที่ 21 จำนวน 1 เรื่อง และในวารสารวิศวะลาดกระบัง จำนวน 2 เรื่อง

ผลงานวิชาการ

1. คมสันต์ หงษ์สมบัติ, มณฑลลีลาจินดาไกรฤกษ์. "การปรับปรุงเสถียรภาพระบบไฟฟ้ากำลังด้วยตัวชดเชยกำลังไฟฟ้าแบบสถิตย์โดยใช้วิธีฟัซซี่." การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 21, 2541.
2. คมสันต์ หงษ์สมบัติ, มณฑล ลีลาจินดาไกรฤกษ์. "เจเนติกอัลกอริทึ่มสำหรับช่วยออกแบบตัวควบคุมฟัซซี่ของ SVC เพื่อปรับปรุงเสถียรภาพระบบไฟฟ้ากำลัง." วิศวสารลาดกระบัง, ปีที่16, ฉบับที่ 1, มีนาคม 2542.
3. คมสันต์ หงษ์สมบัติ, มณฑล ลีลาจินดาไกรฤกษ์. "การปรับค่าพารามิเตอร์ควบคุมให้เหมาะสมสูงสุดของตัวควบคุม $\Delta\omega$-PSS และตัวควบคุมฟัซซี่ของ SVC สำหรับควบคุมเสถียรภาพระบบไฟฟ้ากำลัง." วิศวสารลาดกระบัง, ปีที่ 16, ฉบับที่ 1, มีนาคม 2542.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้