

การพัฒนาอัลกอริธึมการค้นหาแบบตามเชิงปรับตัวหลายวิธี

นายจักรกฤษณ์ เกลือบวัง



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

มหาวิทยาลัยเทคโนโลยีสุรนารี

ปีการศึกษา 2553

**DEVELOPMENT OF MULTI-PATH ADAPTIVE
TABU SEARCH ALGORITHMS**

Jukkrit Kluabwang



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

Suranaree University of Technology

Academic Year 2010

การพัฒนาอัลกอริธึมการค้นหาแบบตามเชิงปรับตัวหลายวิถี

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นำวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาคุณวุฒิบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

(ผศ. ดร.อนันต์ อุ่นศิริไธย์)

ประธานกรรมการ

(ศ. น.ท. ดร.สราวุฒิ สุจิตกร)

กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

(รศ. ดร.เดชา พวงดาวเรือง)

กรรมการ

(รศ. ดร.อาทิตย์ ศรีแก้ว)

กรรมการ

(ดร.สุวัฒน์ กลิ่นเขียว)

กรรมการ

(อ. ดร.วุฒิ ด้านกิตติกุล)

รักษาการแทนรองอธิการบดีฝ่ายวิชาการ

(รศ. น.อ. ดร.วรพจน์ ขำพิศ)

คณบดีสำนักวิชาวิศวกรรมศาสตร์

จักรกฤษณ์ เคลือบวัง : การพัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิถี
(DEVELOPMENT OF MULTI-PATH ADAPTIVE TABU SEARCH ALGORITHMS)
อาจารย์ที่ปรึกษา : ศาสตราจารย์ นาวาอากาศโท ดร.สรายุทธ์ สุจิตกร, 273 หน้า.

งานวิจัยนี้นำเสนอแนวทางการพัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิถี หรือ MATS ด้วยโครงสร้างส่วนจัดการหน่วยค้นหาที่ประกอบด้วยสามกลไกที่สำคัญ ได้แก่ กลไกแบ่งปริภูมิการค้นหา ทำหน้าที่คัดสรรคำตอบเริ่มต้นให้แก่หน่วยค้นหาย่อยจากปริภูมิการค้นหาที่แบ่งได้ กลไกลำดับการค้นหา ทำหน้าที่กำกับหน่วยค้นหาให้ดำเนินการค้นหาเชิงลำดับ และกลไกยกเลิกการค้นหา ทำหน้าที่ลดจำนวนหน่วยค้นหาที่ถูกประเมินว่าให้คำตอบด้อยคุณภาพ ออกจากวงรอบการค้นหา การประเมินสมรรถนะของ MATS กับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวนหกชนิดและห้าชนิดให้ผลการทดสอบชัดเจนว่า MATS สามารถทำความเร็วการค้นหาคำตอบกว้างได้ดีกว่า ATS ได้จริง ส่วนอีกหนึ่งชนิด ATS ดีกว่า MATS เนื่องจากลักษณะของปัญหามีความง่าย การวิเคราะห์คุณสมบัติการลู่เข้าหาคำตอบกว้างของ MATS ได้รับการนำเสนอในวิทยานิพนธ์ นอกจากนี้ได้ประยุกต์ใช้งาน MATS กับปัญหาจริงทางด้านวิศวกรรมศาสตร์ ได้แก่ ปัญหาการออกแบบตัวควบคุมสำหรับระบบจำลองยานพาหนะย่อยส่วนแห่งอิลลินอยส์ และ ปัญหาการระบุเอกลักษณ์ระบบพลวัตสำหรับระบบลูกตุ้มนาฬิกาผนวกรวดผลการค้นหาด้วย MATS ให้คำตอบที่มีคุณภาพเป็นที่น่าพอใจ

สาขาวิชาวิศวกรรมไฟฟ้า

ปีการศึกษา 2553

ลายมือชื่อนักศึกษา_____

ลายมือชื่ออาจารย์ที่ปรึกษา_____

ลายมือชื่ออาจารย์ที่ปรึกษาร่วม_____

JUKKRIT KLUABWANG : DEVELOPMENT OF MULTI-PATH
ADAPTIVE TABU SEARCH ALGORITHMS. THESIS ADVISOR :
PROF. WNG. CMD. SARAWUT SUJITJORN, PhD., 273 PP.

DEVELOPMENT OF MULTI-PATH ADAPTIVE TABU SEARCH ALGORITHMS

This doctoral thesis proposes the multi-path adaptive tabu search algorithms or MATS. The proposed algorithms consist of three important mechanisms namely partitioning mechanism assigned to subdivide the entire search space, and choose a proper initial solution for each search unit, sequencing mechanism to keep all search units working sequentially, and discarding mechanism to eliminate the search unit(s) with low-quality solution(s). To assess search performances, MATS has been tested on six surface optimization problems, five results of which show that the MATS is faster than the ATS. On one of the test problems, the ATS is faster than the MATS due to the simplicity of the problem. Convergence analysis of the MATS is presented in this thesis. Two real-world problems, i.e. controller design for the Illinois roadway simulator (IRS) system and identification of the cart plus pendulum (CPP) system, solved by the proposed MATS, are also presented in this thesis. Satisfactory results have been obtained.

School of Electrical Engineering

Academic Year 2010

Student's Signature_____

Advisor's Signature_____

Co-advisor's Signature_____

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงด้วยดี ผู้วิจัยขอกราบขอบพระคุณ บุคคล และกลุ่มบุคคลต่าง ๆ ที่ได้กรุณาให้คำปรึกษา แนะนำ และช่วยเหลืออย่างดียิ่ง ทั้งในด้านวิชาการ และด้านการดำเนินงานวิจัย ดังต่อไปนี้

ศาสตราจารย์ นาวาอากาศโท ดร.สรายุทธ สุจิตจร อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้กรุณา มอบหมายงานวิจัยอันเป็นประโยชน์ ได้อุทิศความรู้ สติปัญญา ความสามารถ ให้คำปรึกษาแนะนำ และให้กำลังใจ ด้วยดีเสมอมา

รองศาสตราจารย์ ดร.เดชา พวงดาวเรือง อาจารย์ที่ปรึกษาร่วม ที่ได้กรุณามอบวิทยานิพนธ์ คุชฌีบัณฑิตให้ศึกษาเป็นแนวทางเริ่มต้น และข้อมูลจากการทดสอบระบบพลวัตไม่เป็นเชิงเส้น เพื่อการระบุเอกลักษณ์ระบบ ตลอดจนคำแนะนำ และกำลังใจมาโดยตลอด

คณาจารย์ สาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารีทุกท่านที่ได้ให้ความรู้ อันทรงคุณค่าจากการเข้าเรียนรายวิชาบัณฑิตศึกษา และทุกท่านที่ได้ให้คำแนะนำทั้งภาควิชาการ และภาคสังคมตลอดระยะเวลาที่ผู้วิจัยได้ใช้ชีวิตอยู่ที่มหาวิทยาลัยเทคโนโลยีสุรนารี แห่งนี้

เพื่อน พี่น้อง ที่บัณฑิตศึกษาสาขาวิชาวิศวกรรมไฟฟ้า ทุกคนที่ได้มีโอกาสสร้างสรรค์ ประสบการณ์และความทรงจำดี ๆ ตลอดมา

ผู้ช่วยศาสตราจารย์ อัจฉรา คลวิทยาคุณ เพื่อนร่วมชีวิต ที่คอยให้คำแนะนำ กำลังใจ และความช่วยเหลือทุก ๆ เรื่อง ด้วยดีเสมอมา

ท้ายนี้ ขอกราบขอบพระคุณบิดา มารดา ที่ให้การอบรมเลี้ยงดู สนับสนุน และส่งเสริม การศึกษาอย่างดียิ่งเสมอมา ขอกราบขอบพระคุณครูบาอาจารย์ทุกท่าน ที่ให้การอบรมสั่งสอน และ ประสิทธิ์ประสาทวิชาความรู้ด้วยความเมตตาปราณีต่อศิษย์ และขอขอบคุณมหาวิทยาลัยเทคโนโลยี ราชมงคลล้านนา เขตพื้นที่ตาก ผู้สนับสนุนหลักอย่างเป็นทางการ สำหรับทุนการศึกษาระดับปริญญาเอกของผู้วิจัยครั้งนี้

จักรกฤษณ์ เคลือบวัง

สารบัญ

หน้า

บทคัดย่อ (ภาษาไทย).....	ก
บทคัดย่อ (ภาษาอังกฤษ).....	ข
กิตติกรรมประกาศ.....	ค
สารบัญ.....	ง
สารบัญตาราง.....	ช
สารบัญรูป.....	ฉ
คำอธิบายสัญลักษณ์และคำย่อ.....	ท

บทที่

1 บทนำ.....	1
1.1 ความสำคัญของปัญหา.....	1
1.2 วัตถุประสงค์ของการวิจัย.....	1
1.3 ขอบเขตของการวิจัย.....	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ.....	2
1.5 ข้อตกลงเบื้องต้นของการวิจัย.....	2
1.6 การจัดรูปแบบของวิทยานิพนธ์.....	2
2 ทัศนัวรรณกรรมที่เกี่ยวข้อง.....	4
2.1 กล่าวนำ.....	4
2.2 อัลกอริธึมการค้นหาแบบตามดั้งเดิมและรูปแบบที่ได้รับการดัดแปลง.....	4
2.2.1 อัลกอริธึมการค้นหาแบบตามเชิงปฏิบัติ (RTS).....	5
2.2.2 อัลกอริธึมการค้นหาแบบตามเชิงขนาน (PTS).....	6
2.2.3 อัลกอริธึมการค้นหาแบบตามเชิงดัดแปลง (MoTS).....	6
2.2.4 อัลกอริธึมการค้นหาแบบตามเชิงความน่าจะเป็น (PrTS).....	7
2.2.5 อัลกอริธึมการค้นหาแบบตามเชิงผสมผสาน (HTS).....	8
2.2.6 อัลกอริธึมการค้นหาแบบตามเชิงปรับตัว (ATS).....	10

สารบัญ (ต่อ)

หน้า

2.3 สรุป.....	11
3 อัลกอริธึมการค้นหาแบบตามเชิงปรับตัว.....	12
3.1 กล่าวนำ.....	12
3.2 การค้นหาแบบตามเชิงปรับตัว.....	12
3.3 ผลการทดสอบและการวิเคราะห์.....	17
3.4 สรุป.....	21
4 อัลกอริธึมการค้นหาแบบตามเชิงปรับตัวหลายวิธี.....	22
4.1 กล่าวนำ.....	22
4.2 การค้นหาแบบตามเชิงปรับตัวหลายวิธี.....	22
4.3 การวิเคราะห์อัลกอริธึม.....	41
4.3.1 การวิเคราะห์ความซับซ้อนและความถูกต้อง.....	41
4.3.2 การวิเคราะห์ประสิทธิภาพเชิงเวลา.....	55
4.4 การพิสูจน์คุณสมบัติการรู้เข้าหาคำตอบวงกว้างของ MATS.....	58
4.5 สรุป.....	63
5 การประเมินสมรรถนะ.....	64
5.1 กล่าวนำ.....	64
5.2 การตั้งค่าพารามิเตอร์สำหรับ MATS	64
5.2.1 กลไกแบ่งปริภูมิการค้นหา (PM).....	64
5.2.2 กลไกลำดับการค้นหา (SM).....	67
5.2.3 กลไกยกเลิกการค้นหา (DM).....	67
5.3 ผลการทดสอบ.....	72
5.4 วิเคราะห์ผลการทดสอบ.....	82
5.5 สรุป.....	94
6 การประยุกต์ใช้งาน MATS กับการออกแบบตัวควบคุม.....	96
6.1 กล่าวนำ.....	96
6.2 ระบบจำลองยานยนต์ยี่ห้อส่วนแห่งอีลีนิยส์.....	96
6.3 การคาดเดาตัวควบคุม PD เข้ากับบล็อก G_{ff}	105

สารบัญ (ต่อ)

หน้า

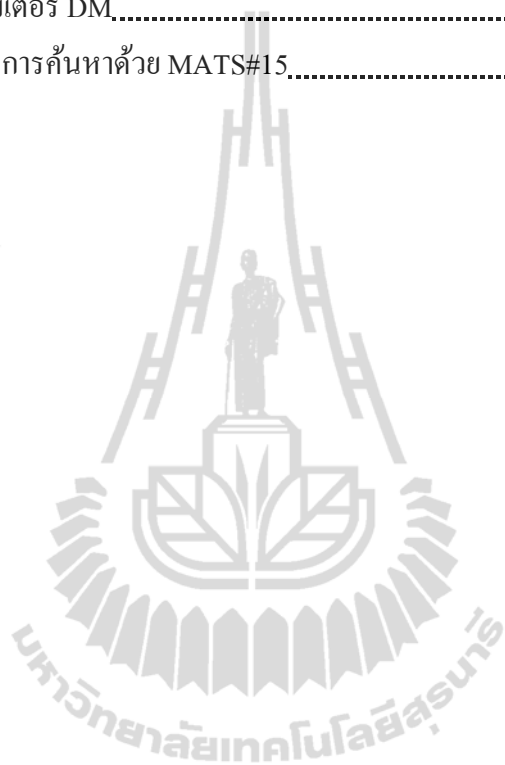
6.4	การปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิมด้วย MATS.....	114
6.5	การปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD ด้วย MATS.....	123
6.6	การวิเคราะห์ผลของการรบกวน.....	132
6.7	สรุป.....	138
7	การประยุกต์ใช้งาน MATS กับการระบุเอกลักษณ์ระบบพลวัต.....	140
7.1	กล่าวนำ.....	140
7.2	ความรู้พื้นฐานการระบุเอกลักษณ์ระบบ.....	140
7.3	ระบบลูกตุ้มนาฬิกาผกผัน.....	147
7.4	การระบุเอกลักษณ์ระบบ CPP.....	152
7.5	การเตรียมข้อมูลจากการทดสอบ.....	153
7.6	การระบุเอกลักษณ์ระบบพลวัตด้วย MATS.....	154
7.7	ผลการค้นหาด้วย MATS แบบ 15 เส้นทางค้นหา.....	162
7.8	สรุป.....	172
8	สรุป.....	173
8.1	ข้อสรุป.....	173
8.2	ข้อเสนอแนะ.....	175
	รายการอ้างอิง.....	178
	ภาคผนวก.....	182
	ภาคผนวก ก. โปรแกรมการค้นหาแบบดาบุงเชิงปรับตัวหลายวิธี.....	182
	ภาคผนวก ข. โปรแกรมการออกแบบตัวควบคุมระบบ IRS ด้วย MATS.....	192
	ภาคผนวก ค. โปรแกรมการระบุเอกลักษณ์ระบบ CPP ด้วย MATS.....	204
	ภาคผนวก ง. รายชื่อบทความวิจัยที่ได้รับการตีพิมพ์เผยแพร่.....	251
	ประวัติผู้เขียน.....	273

สารบัญตาราง

ตารางที่	หน้า
3.1 คุณสมบัติของทั้งหกฟังก์ชันทดสอบ.....	17
3.2 พารามิเตอร์ของ ATS ที่ใช้ทดสอบ.....	19
3.3 ผลการทดสอบสมรรถนะของการค้นหาแบบตามเชิงปรับตัวที่เสนอ.....	20
5.1 พารามิเตอร์ของ MATS ภาค PM ที่ใช้ทดสอบ.....	66
5.2 พารามิเตอร์ของ DM ใน MATS ที่ใช้ทดสอบ.....	68
5.3 จำนวนรอบการค้นหาเฉลี่ยที่พบคำตอบวงกว้างจากทดสอบของ ATS แบบเดี่ยว.....	69
5.4 เวลาการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS.....	73
5.5 จำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS.....	74
5.6 เปอร์เซ็นต์อัตราส่วนความเร็วปรับปรุงของ ATS แบบเดี่ยว และ MATS.....	74
5.7 ประมาณการจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS	75
5.8 เปอร์เซ็นต์การลดจำนวนรอบการค้นหาเฉลี่ยของ MATS ที่สมมูลกับ ATS แบบเดี่ยว.....	75
6.1 พารามิเตอร์ระบบย่อยส่วนพาหนะบนผิวถนน IRS (ของรถจักรยาน เพื่อเป็นตัวอย่าง).....	98
6.2 โพลและซีโรของระบบ $G_{irs}(s)$	103
6.3 คำตอบแทนที่ได้จากพารามิเตอร์ดั้งเดิมและปริภูมิการค้นหาทั้ง 23 มิติ.....	116
6.4 ปริภูมิการค้นหาย่อย	118
6.5 คำตอบที่ได้หลังจากการค้นหาด้วย MATS#23.....	121
6.6 ปริภูมิการค้นหาทั้ง 25 มิติ ของปัญหาการคัดสรรพารามิเตอร์ ตัวควบคุมระบบ IRS	125
6.7 ปริภูมิการค้นหาย่อยทั้ง 25 พารามิเตอร์.....	126
6.8 คำตอบที่ได้หลังการค้นหาด้วย MATS#25.....	129
7.1 คำตอบแทนและปริภูมิการค้นหาของ MATS#15	159
7.2 คำตอบเริ่มต้นและค่า SSE ของ MATS ทั้ง 15 หน่วยค้นหาย่อย.....	161

สารบัญตาราง (ต่อ)

ตารางที่	หน้า
7.3 การตั้งค่าพารามิเตอร์ DM.....	162
7.4 คำตอบที่ได้จากการค้นหาด้วย MATS#15.....	163



สารบัญรูป

รูปที่	หน้า
3.1 อัลกอริทึมการค้นหา.....	13
3.2 เส้นทางการค้นหาและสองกลไกหลักของการค้นหาแบบตาบอด เชิงปรับตัวที่เสนอ.....	16
4.1 ขั้นตอนการทำงานของ MATS.....	24
4.2 ขั้นตอนการทำงานของ PM.....	25
4.3 ตัวอย่างการแบ่งปริภูมิการค้นหาและการตั้งชื่อเส้นทางการค้นหา.....	26
4.4 การแบ่งปริภูมิการค้นหาเป็นส่วนหนึ่งของฟังก์ชันซีเคิลฟอกซ์ไฮล.....	27
4.5 ลักษณะการทำงานของ SM.....	30
4.6 ตัวอย่างการทำงานของ SM ใน MATS แบบ4 เส้นทางการค้นหาที่ยังไม่มี DM	32
4.7 ขั้นตอนการทำงานของ DM.....	36
4.8 ตัวอย่างการทำงานของ SM ใน MATS แบบ4 เส้นทางการค้นหาพร้อม DM.....	37
4.9 การค้นหาของ ATS แบบเดี่ยว จำนวน 4 ครั้งด้วยคำตอบเริ่มต้นที่แตกต่างกัน.....	42
4.10 ค่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ที่ดีที่สุดจะอยู่ระหว่าง กรณีที่ดีที่สุด (n_{min}) กับกรณีแย่ที่สุด (n_{max}).....	45
4.11 การค้นหาของ MATS แบบ 4 หน่วยค้นหา.....	47
4.12 ค่าจำนวนรอบการค้นหาเฉลี่ยของ MATS สั้นกว่าของ ATS แบบเดี่ยว ด้วยอัตราส่วน ρ ทำให้ MATS มีโอกาสพบคำตอบได้รวดเร็วกว่าของ ATS แบบเดี่ยว.....	57
5.1 ตัวอย่าง PM พร้อมการตั้งชื่อหน่วยค้นหา.....	65
5.2 กราฟแท่งอัตราส่วนความเร็วปรับปรุงของ MATS เทียบกับ ATS แบบเดี่ยวบนฟังก์ชัน.....	78
5.3 กราฟหลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา BF.....	83
5.4 กราฟหลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา RF.....	85
5.5 กราฟหลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา SF.....	87
5.6 กราฟหลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ShF.....	89

สารบัญรูป (ต่อ)

รูปที่	หน้า
5.7 กราฟผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ScF1.....	91
5.8 กราฟผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ScF2.....	93
6.1 ภาพแสดงทิศทางเพื่อการวิเคราะห์ของแบบจำลองย่อยส่วนพาหนะ IRS.....	97
6.2 บล็อกไดอะแกรมของแบบจำลองย่อยส่วนรถจักรยาน IRS พร้อม ระบบควบคุม.....	100
6.3 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs}(s)$	105
6.4 ระบบ IRS ที่มีการคาดเดาควบคุม PD หรือ G_{PD} กับบล็อก G_{ff}	106
6.5 ขั้นตอนการประเมินค่าของฟังก์ชันวัตถุประสงค์ในการค้นหา ตัวควบคุม PD.....	109
6.6 (ก) แผนภูมิการทำงานของการค้นหาแบบตามูเชิงปรับตัวหลายวิถี เพื่อการออกแบบตัวควบคุมพีดีสำหรับระบบจำลองรถยนต์แห่งอิลลินอยส์ (ข) แผนภูมิการทำงานของการหาค่าฟังก์ชันวัตถุประสงค์.....	111
6.7 กราฟการผู้เข้าหาผลเฉลยของ MATS ในการค้นหาตัวควบคุมของ PD.....	112
6.8 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs,PD}(s)$	113
6.9 ขั้นตอนการประเมินค่าของฟังก์ชันวัตถุประสงค์ในการค้นหา ตัวควบคุมใหม่.....	115
6.10 (ก) แผนภูมิการทำงานของ MATS แบบ 23 หน่วยค้นหาเพื่อการปรับแต่ง ค่าพารามิเตอร์ใหม่ให้ตัวควบคุมดั้งเดิมสำหรับระบบ IRS (ข) แผนภูมิ การทำงานของการหาค่าฟังก์ชันวัตถุประสงค์.....	117
6.11 กราฟการผู้เข้าหาผลเฉลยของ MATS ในการค้นหาพารามิเตอร์ 23 ตัว.....	122
6.12 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs,23-s}(s)$	122

สารบัญรูป (ต่อ)

รูปที่	หน้า
6.13 (ก) แผนภูมิการทำงานของ MATS แบบ 25 หน่วยค้นหาเพื่อ การปรับแต่งค่าพารามิเตอร์ใหม่ให้ทั้งตัวควบคุมดั้งเดิมและตัวควบคุม ที่ดีสำหรับระบบ IRS (ข) แผนภูมิการทำงานของกราฟค่า ฟังก์ชันวัตถุประสงค์.....	124
6.14 กราฟการลู่เข้าหาผลเฉลยของ MATS กับการค้นหาพารามิเตอร์ 25 ตัวของตัวควบคุม.....	130
6.15 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs,PD\&23-s} (s)$	130
6.16 ภาพรวมผลตอบสนองแบบขั้นบันไดที่ได้ศึกษาทั้งหมด.....	131
6.17 แผนภาพระบบ IRS ดั้งเดิมที่มีการรบกวน v	133
6.18 แผนภาพระบบ IRS ที่มีการปรับแต่งพารามิเตอร์พร้อมตัวควบคุม PD และปรากฏการณ์รบกวน v	133
6.19 สัญญาณอ้างอิง r สัญญาณการรบกวน v และผลตอบสนองของระบบ IRS ดั้งเดิม.....	134
6.20 ภาพขยายเอาต์พุตต่อการรบกวนซิกบวก (ภาพบน) และ การรบกวนซิกลบ (ภาพล่าง).....	135
6.21 ภาพเอาต์พุตของทั้งสี่ระบบต่ออินพุตที่เป็นการรบกวนและ สัญญาณขั้นบันได.....	136
6.22 ภาพขยายเอาต์พุตของทั้งสี่ระบบต่อการรบกวนซิกบวก.....	137
6.23 ภาพขยายเอาต์พุตของทั้งสี่ระบบต่อการรบกวนซิกลบ.....	138
7.1 การระบุเอกลักษณ์ระบบ.....	141
7.2 การพัฒนาแบบจำลองทางคณิตศาสตร์.....	143
7.3 แผนผังลำดับงานระบุเอกลักษณ์ระบบ.....	145
7.4 ระบบ CPP.....	148
7.5 โครงสร้างระบบ CPP ของบริษัท Feedback.....	148
7.6 การเตรียมข้อมูลดิบสำหรับการระบุเอกลักษณ์.....	153
7.7 ขั้นตอนของระเบียบวิธีรู้งู่กฏตาอันดับสี่.....	155

สารบัญรูป (ต่อ)

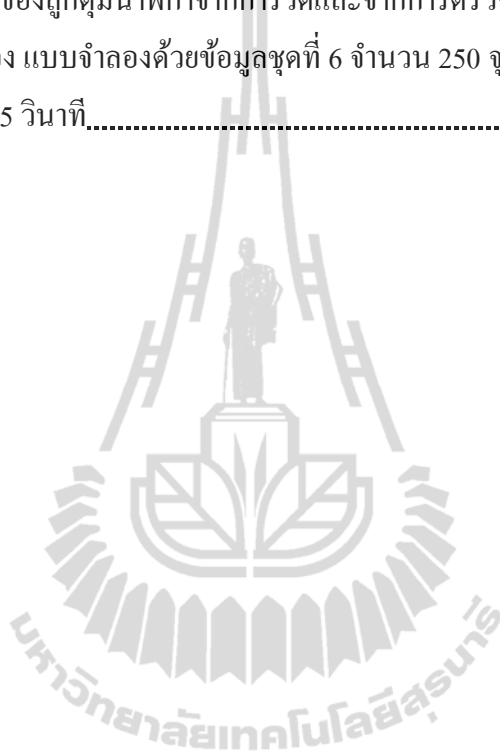
รูปที่	หน้า
7.8	ลักษณะทั่วไปของ k_1 k_2 k_3 k_4 และ h ในระเบียบวิธีรุงเงดตาอันดับสี่.....156
7.9	ขั้นตอนการคำนวณค่า SSE.....157
7.10	(ก) แผนภูมิการทำงานของการค้นหาแบบตาบู่เชิงปรับตัวหลายวิธี เพื่อการระบุเอกลักษณ์ระบบพลวัตสำหรับระบบบรณนกเพนดูลัม (ข) แผนภูมิการทำงานของการหาค่าฟังก์ชันวัตถุประสงค์.....160
7.11	กราฟการลู่เข้าผลเฉลยของ MATS กับปัญหาการระบุเอกลักษณ์ ระบบพลวัต CPP.....162
7.12	ผลการระบุเอกลักษณ์ ที่ค่า $SSE = 3.1891$ 164
7.13	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและจากการระบุเอกลักษณ์ ด้วยข้อมูลชุดที่ 2 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที.....164
7.14	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและจากการระบุเอกลักษณ์ ด้วยข้อมูลชุดที่ 3 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที.....165
7.15	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและจากการระบุเอกลักษณ์ ด้วยข้อมูลชุดที่ 4 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที.....165
7.16	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและจากการระบุเอกลักษณ์ ด้วยข้อมูลชุดที่ 5 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที.....166
7.17	ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากช่วงข้อมูลที่ 1 ของฟังก์ชันแรงเดิม.....168
7.18	ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากช่วงข้อมูลที่ 1 ของฟังก์ชันแรงใหม่.....169
7.19	ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากช่วงข้อมูลที่ 6 ของฟังก์ชันแรงเดิม.....169
7.20	ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากช่วงข้อมูลที่ 6 ของฟังก์ชันแรงใหม่.....170
7.21	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและการตรวจสอบ ความถูกต้องของ แบบจำลองด้วยข้อมูลชุดที่ 1 จำนวน 250 จุด คิดเป็นเวลา 37.5 วินาที.....171

สารบัญรูป (ต่อ)

รูปที่

หน้า

7.22	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและการตรวจสอบ ความถูกต้องของ แบบจำลองด้วยข้อมูลชุดที่ 6 จำนวน 250 จุด คิดเป็นเวลา 37.5 วินาที.....	171
------	---	-----



คำอธิบายสัญลักษณ์และคำย่อ

AC	=	เกณฑ์ปรารถนา (aspiration criteria)
AR	=	กลไกปรับรัศมีการค้นหา (adaptive search radius mechanism)
ATS	=	การค้นหาแบบตามูเชิงปรับตัว (adaptive tabu search algorithms)
BF	=	ฟังก์ชันโบฮาเชฟสกี (Bohachevsky's function)
BT	=	กลไกย้อนรอยการค้นหา (back-tracking mechanism)
CLWs	=	ตัวทำงานรายการตัวเลือก (candidate list workers)
CPP	=	ระบบลูกตุ้มนาฬิกาพลาทอน (cart plus pendulum system)
$Count_{ATS}$	=	จำนวนรอบการค้นหาของ ATS
$Count_{MATS}$	=	จำนวนรอบการค้นหาของ MATS
$Count_{MATSeq}$	=	จำนวนรอบการค้นหาของ MATS ที่ผสมมูลกับของ ATS
$Count_{max}$	=	จำนวนรอบการค้นหาสูงสุดซึ่งเป็นหนึ่งในเกณฑ์ยุติการค้นหา
DM	=	กลไกยกเลิกการค้นหา (discarding mechanism)
GA	=	จินติกส์อัลกอริทึม (genetic algorithms)
GS	=	คำตอบวงกว้าง (global solution)
HDD	=	ฮาร์ดดิสก์ไดรฟ์ (hard disk drive)
HTS	=	การค้นหาแบบตามูเชิงผสมผสาน (hybrid tabu search algorithms)
IRS	=	เครื่องจำลองยานยนต์ย่อบางแห่งอิลลินอยส์ (Illinois roadway simulator)
IS	=	คำตอบเริ่มต้น (initial solution)
M	=	มวลของลูกตุ้มนาฬิกา
M	=	มวลของรถ
MATS	=	การค้นหาแบบตามูเชิงปรับตัวหลายวิถี (multi-path adaptive tabu search algorithms)
MIP	=	ตัวทำงานสำหรับการโปรแกรมเชิงจำนวนเต็มผสม (mixed integer programming)
MoTS	=	การค้นหาแบบตามูเชิงดัดแปลง (modified tabu search algorithms)
MRC	=	ตัวควบคุมเชิงอ้างอิงแบบจำลอง (model reference controller)

คำอธิบายสัญลักษณ์และคำย่อ (ต่อ)

$n_{\min}$	=	จำนวนรอบการค้นหาน้อยที่สุดของ ATS
$n_{\max}$	=	จำนวนรอบการค้นหาที่มากที่สุดของ ATS
OO	=	เทคนิคการหาค่าเหมาะที่สุดแบบออร์ดิแนล (ordinal optimization)
PTS	=	การค้นหาแบบตามูเชิงขนาน (parallel tabu search algorithms)
PrTS	=	การค้นหาแบบตามูเชิงความน่าจะเป็น (probabilistic tabu search algorithms)
PSO	=	เทคนิคการหาค่าเหมาะที่สุดแบบฝูงอนุภาค (particle swarm optimization)
PM	=	กลไกแบ่งปริภูมิการค้นหา (partitioning mechanism)
RAM	=	หน่วยความจำแบบแรม (random access memory)
RF	=	ฟังก์ชันราสทริกิน (Rastrigin's function)
RTS	=	การค้นหาแบบตามูเชิงปฏิกิริยา
S_c	=	คำตอบแกน
ScF	=	ฟังก์ชันชีเฟล (Schwefel's function)
SF	=	ฟังก์ชันชีเคิลฟอกซ์โฮล (Shekel's foxholes function)
ShF	=	ฟังก์ชันชูเบิร์ต (Shubert's function)
SQP	=	ปัญหาการหาค่าเหมาะที่สุดแบบควอดราติกอนุกรม
SM	=	กลไกลำดับการค้นหา (sequencing mechanism)
SSE	=	ผลรวมค่าความคลาดเคลื่อนกำลังสอง (sum square of error)
t	=	เวลาค้นหาเฉลี่ยต่อรอบการค้นหา
t_d	=	ค่าเวลาประวิง (delay time)
t_r	=	ค่าเวลาไต่ระดับ (rise time)
t_s	=	ค่าเวลาเข้าที่ (settling time)
TC	=	เกณฑ์ยุติการค้นหา (termination criteria)
TL	=	รายการตามู (tabu list)
T_s	=	เวลาการค้นหา
TS	=	การค้นหาแบบตามู (tabu search)
TSW	=	ตัวทำงานของการค้นหาแบบตามู (tabu search worker)

คำอธิบายสัญลักษณ์และคำย่อ (ต่อ)

T_w	=	เวลาคอย (waiting time)
UCP	=	ปัญหายูนิตคอมมิตเมนต์ (unit commitment problem)
P.O.	=	เปอร์เซ็นต์การพุ่งเกิน (percent of overshoot)
δ	=	เวลาเปลี่ยนสถานะ
f	=	แรงกระทำต่อรถ
u	=	อินพุต
y	=	ตำแหน่งรถตามแนวนอน
θ	=	ตำแหน่งเชิงมุมของพูลเลย์ที่ติดกับมอเตอร์
ϕ	=	ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกา
G	=	ความเร่งโน้มถ่วง
l	=	ความยาวของก้านลูกตุ้ม
t_{ij}	=	ค่าเปอร์เซ็นต์เวลาไต่ขึ้น
t_{sj}	=	ค่าเปอร์เซ็นต์เวลาเข้าที่

บทที่ 1

บทนำ

1.1 ความสำคัญของปัญหา

ผ่านมาหลายทศวรรษ เทคนิคการหาค่าเหมาะที่สุด เป็นเครื่องมืออันทรงพลังและมีประสิทธิภาพมากในการหาผลเฉลย สำหรับปัญหาซับซ้อน ทางด้านวิศวกรรมศาสตร์และวิทยาศาสตร์ คณะผู้วิจัยสาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารี ได้เล็งเห็นถึงคุณค่าและความสำคัญของเทคนิคดังกล่าว จึงได้เริ่มศึกษาการค้นหาแบบตาบู่ (tabu search : TS) (Glover, 1986) จนกระทั่ง พ.ศ. 2547 ได้กลายเป็นการค้นหาแบบตาบู่เชิงปรับตัว (adaptive tabu search : ATS) (Sujitjorn, S., 2006) ที่มีความสามารถในการค้นหาคำตอบได้รวดเร็วว่าการค้นหาแบบตาบู่ดั้งเดิม โดยยืนยันศักยภาพได้จากบทความและวารสารที่เผยแพร่ระดับชาติและระดับนานาชาติมาแล้ว และเพื่อสานต่อแนวทางการพัฒนาเครื่องมืออันทรงพลังนี้ให้มีสมรรถนะที่สูงขึ้นต่อไป จึงเกิดเป็นหัวข้อวิจัยภายใต้ชื่อ “การพัฒนาอัลกอริธึมการค้นหาแบบตาบู่เชิงปรับตัวหลายวิถี” หรือ ชื่อภาษาอังกฤษ “development of multi-path adaptive tabu search” หรือเรียกโดยย่อว่า “MATS” โดยผู้วิจัยคาดหวังว่าอัลกอริธึมใหม่นี้จะทำความเร็วในการค้นหาคำตอบได้ดีกว่าการค้นหาแบบตาบู่เชิงปรับตัว และเพื่อให้เป็นทางเลือกที่ดีกว่า สำหรับนำไปประยุกต์ใช้ในการแก้ปัญหาที่ซับซ้อนในทางวิศวกรรมศาสตร์ด้านต่าง ๆ ต่อไป

1.2 วัตถุประสงค์ของการวิจัย

- 1.2.1 เพื่อศึกษาแนวทางพัฒนาอัลกอริธึมใหม่บนฐานของอัลกอริธึมการค้นหาแบบตาบู่เชิงปรับตัว
- 1.2.2 เพื่อพัฒนาอัลกอริธึมการค้นหาแบบตาบู่เชิงปรับตัวหลายวิถี
- 1.2.3 เพื่อพิสูจน์การลู่เข้าหาผลเฉลยของการค้นหาแบบตาบู่เชิงปรับตัวหลายวิถี
- 1.2.4 เพื่อประยุกต์ใช้อัลกอริธึมใหม่ที่พัฒนาขึ้นกับปัญหาทางด้านวิศวกรรมศาสตร์

1.3 ขอบเขตของการวิจัย

- 1.3.1 พัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี
- 1.3.2 พิสูจน์การรู้เข้าหาผลเฉลยของอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี
- 1.3.3 ประยุกต์ใช้ในงานออกแบบระบบควบคุม
- 1.3.4 ประยุกต์ใช้ในงานระบุเอกลักษณ์ระบบพลวัต

1.4 ประโยชน์ที่คาดว่าจะได้รับ

- 1.4.1 ได้อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี
- 1.4.2 ได้บทพิสูจน์การรู้เข้าหาผลเฉลยของอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี
- 1.4.3 ได้ประยุกต์ใช้การค้นหาแบบตามูเชิงปรับตัวหลายวิธีกับงานออกแบบระบบควบคุม
- 1.4.4 ได้ประยุกต์ใช้การค้นหาแบบตามูเชิงปรับตัวหลายวิธีกับงานระบุเอกลักษณ์ระบบพลวัต

1.5 ข้อตกลงเบื้องต้นของการวิจัย

- 1.5.1 อัลกอริธึมที่จะพัฒนาขึ้นให้ดำเนินการตามลำดับขั้น (sequential)
- 1.5.2 ปริภูมิการค้นหาสามารถถูกแบ่งเป็นปริภูมิย่อยได้
- 1.5.3 ขอบเขตปริภูมิการค้นหาจะถูกกำหนดขึ้นเพื่อใช้ประโยชน์ในการหาผลเฉลยตั้งแต่เมื่อเริ่มต้นเส้นทางการค้นหาเรื่อยๆ เท่านั้น

1.6 การจัดรูปเล่มของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ประกอบด้วย 8 บท 4 ภาคผนวก ซึ่งมีเนื้อหาโดยสรุปดังต่อไปนี้

บทที่ 1 กล่าวถึงบทนำ ประกอบไปด้วย ความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของงานวิจัย ขอบเขตการวิจัย ประโยชน์ที่คาดว่าจะได้รับ ข้อตกลงเบื้องต้นของการวิจัย และการจัดรูปเล่มของวิทยานิพนธ์

บทที่ 2 กล่าวถึงการปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้องกับการค้นหาแบบตามูเชิงเดิม เพื่อใช้เป็นข้อมูลอ้างอิงและเป็นแนวทางสำหรับดำเนินงานวิจัยต่อไป

บทที่ 3 กล่าวทบทวนถึงแนวคิดและหลักการของอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัว นอกจากนี้ยังดำเนินการทดสอบกับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวน 6 พื้นที่ ได้แก่ ฟังก์ชันโบฮาเชฟสกี ฟังก์ชันราสทริกิน ฟังก์ชันซิคเลฟอกโฮล ฟังก์ชันซูเบิร์ต ฟังก์ชันซีเฟลแบบที่หนึ่งและฟังก์ชันซีเฟลแบบที่สอง ทั้งนี้เพื่อใช้เป็นข้อมูลสำหรับการพัฒนาอัลกอริธึมใหม่ต่อไป

บทที่ 4 กล่าวถึงอัลกอริธึมใหม่ในชื่ออัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี ซึ่งประยุกต์ใช้โครงสร้างการจัดการหน่วยค้นหา อันประกอบด้วยสามกลไกหลัก ได้แก่ กลไกแบ่งปริภูมิการค้นหา กลไกลำดับการค้นหา และกลไกยกเลิกการค้นหา เพื่อบริหารอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวจำนวนตั้งแต่สองหน่วยค้นหาขึ้นไปให้ทำงานร่วมกันได้ดี โดยมีเป้าหมายเพื่อลดเวลาการค้นหารวมลงให้น้อยที่สุด นอกจากนี้ยังกล่าวถึงบทพิสูจน์การลู่เข้าหาคำตอบวงกว้างของ MATS และการวิเคราะห์อัลกอริธึมของ MATS อีกด้วย

บทที่ 5 กล่าวถึงกระบวนการทดสอบสมรรถนะ โดยเลือกทดสอบกับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวน 6 พื้นที่ เช่นเดียวกับบทที่ 3 ผลการทดสอบพบว่าอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธีมีสมรรถนะการค้นหาที่สูงขึ้นกว่าของอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวเดิม อย่างน่าพอใจ

บทที่ 6 กล่าวถึงการประยุกต์ MATS เพื่อการออกแบบตัวควบคุมสำหรับระบบจำลองยานยนต์ยี่ห้อแอ่งอิลลินอยส์ โดยเสนอไว้ 3 แนวทาง ได้แก่ แนวทางที่หนึ่งการคาดเดาตัวควบคุม PD เข้ากับบล็อก G_{ff} แนวทางที่สองการปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิม และแนวทางที่สามการปรับแต่งค่าพารามิเตอร์ใหม่ให้ทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD โดยทั้งสามแนวทางให้ผลเป็นที่น่าพอใจ

บทที่ 7 กล่าวถึงการประยุกต์ MATS เพื่อการระบุเอกลักษณ์ระบบพลวัตของระบบลูกตุ้มนาฬิกาพวงกรรหรือ CPP โดยเสนอฟังก์ชันแรง f ที่ประกอบด้วยพารามิเตอร์จำนวน 15 ตัว แล้วดำเนินการค้นหาด้วย MATS แบบ 15 หน่วยค้นหา ผลการระบุเอกลักษณ์ระบบ CPP จากข้อมูลทดสอบในย่านการทำงานไม่เชิงเส้น ให้ผลเป็นที่น่าพอใจ

บทที่ 8 เป็นบทสรุปของงานวิจัยและข้อเสนอแนะ

ภาคผนวก ก. แสดงรายละเอียดโปรแกรม MATS ที่พัฒนาขึ้นด้วยโปรแกรม MATLAB

ภาคผนวก ข. แสดงรายละเอียดโปรแกรมออกแบบตัวควบคุมระบบ IRS

ภาคผนวก ค. แสดงรายละเอียดโปรแกรมระบุเอกลักษณ์ระบบ CPP

ภาคผนวก ง. แสดงรายชื่อบทความวิจัยที่ได้รับการตีพิมพ์

บทที่ 2

ปรัทัศนัวรรณกรรมและงานวิจัยที่เกี่ยวข้อง

2.1 กล่าวนำ

บทที่ 2 นี้ นำเสนอการปรัทัศนัวรรณกรรมและงานวิจัยที่เกี่ยวข้องกับการค้นหาแบบตามูเชิงปรับตัว โดยสามารถแบ่งออกได้เป็นหกแนวทางได้แก่ การค้นหาแบบตามูเชิงปรับตัว การค้นหาแบบตามูเชิงปฏิกริยา การค้นหาแบบตามูเชิงขนาน การค้นหาแบบตามูเชิงดัดแปลง การค้นหาแบบตามูเชิงความน่าจะเป็นและการค้นหาแบบตามูเชิงผสมผสานโดยใช้ ฐานข้อมูลที่เป็นแหล่งสะสมรายงานวิจัยและวรรณกรรมที่เกี่ยวข้องทางวิศวกรรมศาสตร์และเทคโนโลยี เช่น ฐานข้อมูล IEEE IEICE SpringerLink ScienceDirect และ Google เป็นต้น ผลการปรัทัศนัวรรณกรรมและงานวิจัยที่เกี่ยวข้องมีรายละเอียดดังต่อไปนี้

2.2 อัลกอริธึมการค้นหาแบบตามูดั้งเดิมและรูปแบบที่ได้รับการดัดแปลง

การค้นหาแบบตามู (Tabu search : TS) (Glover, 1986) เป็นวิธีการค้นหาคำตอบที่เหมาะสมที่สุดด้วยคอมพิวเตอร์ ในปัญหาแบบไม่ต่อเนื่องเชิงผสมผสาน การค้นหาแบบตามู อาศัยกระบวนการค้นหาคำตอบค่าใกล้เคียง (neighborhood search) และรายการตามู (Tabu list : TL) ซึ่งจะทำหน้าที่เก็บค่าคำตอบในอดีต คำตอบใน TL สามารถใช้ประโยชน์ในกรณีการค้นหาเกิดการติดลือกโดยคำตอบวงแคบเฉพาะถิ่น (local solution) โดยอาศัยเกณฑ์ปรารธนา (aspiration criteria : AC) ช่วยในการค้นหาคำตอบค่าที่มีคุณภาพ กระบวนการค้นหาจะเคลื่อนที่แบบดิสทอร์บิโนสติก ซึ่งจะถูกลำหนดไว้ว่าเป็นระเบียบแบบแผน โดยทั่วไปจะถูกกำกับไว้ด้วยเงื่อนไขความคงอยู่ต่ำสุด (recency) และเงื่อนไขความซ้ำซาก (frequency) (Glover, 1989; Glover and Laguna, 1997) อัลกอริธึม TS แบบทั่วไปมีรายละเอียดดังต่อไปนี้

ขั้นตอนที่ 1 การเริ่มต้น กำหนด TC TL และ AC จากปรกฏมิกการค้นหาที่กำหนดสร้างคำตอบเริ่มต้น และค่าตอบข้างเคียง

ขั้นตอนที่ 2 การวนรอบ สร้างปรกฏมิกการค้นหาย่อยและคำตอบที่เป็นไปได้ ประเมินค่าฟังก์ชันวัตถุประสงค์ทุกคำตอบที่เป็นไปได้ของค่าข้างเคียง หากคำตอบที่ดีที่สุดปัจจุบันจากค่าใกล้เคียงมีค่าต่ำกว่าให้แทนที่คำตอบที่ดีที่สุดด้วยคำตอบที่ดีที่สุดจากค่าใกล้เคียงดังกล่าวและปรับปรุง TL โดยการบันทึกคำตอบที่ดีที่สุดค่าที่ถูกแทนที่นั้นไว้ มิฉะนั้นให้คงคำตอบที่ดี

ที่สุดเดิมไว้ ถ้ากระบวนการค้นหายังไม่สามารถแทนที่คำตอบที่ดีที่สุดเดิมได้ภายในเวลาที่กำหนด ให้ไปดำเนินการ ขั้นตอนที่ 3 และปรับปรุงตัวนับ

ขั้นตอนที่ 3 การยุติการค้นหา ยุติการค้นหาเมื่อพบว่าเงื่อนไข TC เป็นจริง มิฉะนั้นให้กลับไปดำเนินการ ขั้นตอนที่ 1

ขั้นตอนที่ 4 เหน้ปรารธนา ดำเนินการค้นหาภายใต้เงื่อนไข AC ปรับปรุง TL และตัวนับ แล้วไปดำเนินการ ขั้นตอนที่ 1

ปัจจุบันพบว่ามึรูปแบบอันหลากหลายของ TS (Kluabwang and Sujitjorn, 2009) ดังที่จะทบทวนต่อไปนี้

2.2.1 การค้นหาแบบตามูเชิงปฏิกริยา (reactive tabu search : RTS)

การค้นหาแบบตามูเชิงปฏิกริยา (Battiti and Tecchiolli, 1994) มีการดัดแปลงจากการค้นหาแบบตามูดั้งเดิม อยู่สองประการ ประการที่หนึ่งได้มีการดัดแปลงเทคนิคการค้นหาคำตอบเฉพาะถิ่นที่มีการเคลื่อนที่ ในรอบการคำนวณถัดไปจะเลือกจากคำตอบที่ดีที่สุดของค่าใกล้เคียงของจุดในรอบการค้นหา ปัจจุบันคำตอบที่ดีที่สุดของการค้นหาหมายถึงคำตอบที่ให้ค่าฟังก์ชันวัตถุประสงค์ต่ำที่สุด และประการที่สองเป็นการหลีกเลี่ยงการวนซ้ำเป็นกระบวนการป้องกันการย้อนกลับไปดำเนินการซ้ำส่วนที่เพิ่งจะดำเนินการไป โดยอาศัยรายการตามู ส่วนกลไกปฏิกริยา (reaction mechanism) เป็นกลไกปรับเปลี่ยนความยาวของ TL ให้เหมาะสม กลไกปฏิกริยานี้จะทำงานเมื่อพบว่าค่าฟังก์ชันวัตถุประสงค์ไม่ได้ถูกปรับปรุงมาเป็นเวลานาน ซึ่งเป็นปรากฏการณ์ติดกับดัก (local trap) ดังนั้นกลไกปฏิกริยานี้จึงช่วยการค้นหาแบบตามูเชิงปฏิกริยาให้มีคุณสมบัติการหนีลือกได้เป็นอย่างดี การค้นหาแบบตามูเชิงปฏิกริยา (Wassan, 2006) พัฒนาจาก RTS ของเบตติติและเทคคิโอ ลี (Battiti, R., and Tecchiolli, G., 1994) เฉพาะใน ส่วนกลไกหนีลือก (escape mechanism) โดยการเปลี่ยนจากกลไกการจัดตำแหน่งใหม่แบบสุ่ม (random repositioning) ไปเป็นรูปแบบการแลกเปลี่ยนที่ 2 (2-exchanges scheme) ขณะที่ยังคงรูปแบบการแลกเปลี่ยนที่ 1 (1-exchanges scheme) ไว้อยู่ รูปแบบการแลกเปลี่ยนที่ 1 เป็นการแทนที่หนึ่งในคำตอบของเส้นทางที่ 2 ด้วยหนึ่งในคำตอบของเส้นทางที่ 1 และสลับคำตอบระหว่างเส้นทางที่ 1 กับเส้นทางที่ 2 เส้นทางละคำตอบ ส่วนรูปแบบการแลกเปลี่ยนที่ 2 หมายถึงการแทนที่คำตอบหนึ่งคู่ในของเส้นทางที่ 2 ด้วยคำตอบหนึ่งคู่ในเส้นทางที่ 1 และสลับคำตอบอีกหนึ่งคู่ระหว่างเส้นทางที่ 1 กับเส้นทางที่ 2 ทั้งรูปแบบการแลกเปลี่ยนที่ 1 และ รูปแบบการแลกเปลี่ยนที่ 2 เป็นเทคนิควิธีของกระบวนการค้นหาค่าใกล้เคียง ซึ่งช่วยเสริมกลยุทธ์ความหลากหลาย (diversification strategy) ให้กับกระบวนการค้นหาคำตอบได้

2.2.2 การค้นหาแบบตาบูเชิงขนาน (parallel tabu search, PTS)

การค้นหาแบบตาบูเชิงขนาน (Crainic, T.G., Toulouse, M., and Gendreau, M., 1996) ชนิดนี้ เป็นการขนานการค้นหาแบบตาบูเชิงลำดับ (sequential tabu search) หลายการค้นหาเข้าด้วยกัน มีหน่วยความจำส่วนกลาง ทำหน้าที่ติดต่อสื่อสารระหว่างหน่วยการค้นหาและตรวจจับค่าต่ำสุดที่เป็นคำตอบวงกว้าง ระหว่างที่การค้นหาหน่วยต่าง ๆ กำลังดำเนินการอยู่ การค้นหาแบบตาบูเชิงขนาน (Talbi, E.G., Hafidi, Z., and Geib, J.M., 1998) เป็นการขนานการค้นหาแบบตาบูจำนวนมากกว่าหนึ่งการค้นหาบนแพลตฟอร์มเดียวกัน ให้ทำงานแบบพร้อมกัน โดยแต่ละการค้นหาจะมีกลไกการขนานแบบปรับตัวได้ (adaptive parallelism) ทำหน้าที่ปรับระดับการขนาน (parallelism degree) ให้เหมาะกับโหลดการคำนวณของระบบ การค้นหาแบบตาบูที่นำมาขนานกัน จะทำงานอย่างอิสระโดยไม่ต้องมีการติดต่อสื่อสารกัน และเริ่มต้นด้วยคำตอบแรกที่แตกต่างกัน รวมถึงพารามิเตอร์การค้นหาก็ก่อต่างกันด้วย เช่น ความยาวของ TL และ จำนวนรอบการคำนวณสูงสุด นอกจากนี้ที่กล่าวนี้แล้ว ยังมีการค้นหาแบบตาบูเชิงขนาน (Sait, S.M., Youssef, H., Barada, H.R., and Ahmad, A.Y., 2000) ซึ่งประกอบด้วย 3 ส่วนที่สำคัญ คือ ส่วนที่หนึ่งกระบวนการหลัก (master process) ส่วนที่สองผู้ปฏิบัติการค้นหาแบบตาบู (tabu search workers : TSWs) และ ส่วนที่สามผู้ปฏิบัติการบัญชีรายชื่อ (candidate list workers : CLWs) อัลกอริธึมนี้ทำงานพร้อมกันอยู่สองระดับ ได้แก่ ระดับบนคือระดับกระบวนการค้นหาแบบตาบู ซึ่งกระบวนการหลักจะทำการกำหนดจำนวนหน่วยผู้ปฏิบัติ การค้นหาแบบตาบู TSWs แล้วกำหนดคำตอบเริ่มต้นของแต่ละหน่วยผู้ปฏิบัติการค้นหาแบบตาบูด้วยค่าเดียวกัน ส่วนระดับล่างคือ ระดับการสร้างบัญชีรายชื่อการคำตอบใกล้เคียงเฉพาะถิ่น (local neighborhood search) ระดับนี้หน่วยผู้ปฏิบัติการค้นหาแบบตาบู TSWs ที่ถูกสร้างขึ้นแล้วในระดับบนแต่ละตัวจะอาศัยผู้ปฏิบัติการบัญชีรายชื่อ CLWs จำนวนมาก เพื่อค้นหาคำตอบค่าใกล้เคียงของคำตอบปัจจุบัน โดยจะกำหนดคำตอบเริ่มต้น ช่วงการค้นหาที่ไม่ทับซ้อนกัน และพารามิเตอร์อื่น ๆ ที่จำเป็นให้กับ CLW แต่ละตัว คำตอบที่ดีที่สุดของ TSWs จะถูกคัดเลือกจากตัวประมวลผลหลัก (master) เพื่อกำหนดเป็นค่าเริ่มต้นในรอบการค้นหาถัดไปให้กับแต่ละ TSW ดำเนินการค้นหาต่อไปจนครบรอบการค้นหาที่กำหนดไว้หรือพบคำตอบที่พอใจตามที่ตั้งไว้ อัลกอริธึมนี้ถูกนำไปใช้แก้ปัญหาการจัดวางเซลล์มาตรฐานภายในวงจรรวมขนาดใหญ่มาก (VLSI standard - cell placement problem) ได้ผลเป็นที่น่าพอใจ

2.2.3 การค้นหาแบบตาบูเชิงดัดแปลง (modified tabu search, MoTS)

การค้นหาแบบตาบูเชิงดัดแปลง (Nowicki, E., and Smutnicki, C., 2005) ชนิดนี้เป็นอัลกอริธึมที่พัฒนาขึ้นมาโดยอิงปรากฏการณ์หุบเขาขนาดใหญ่ (big valley phenomenon) ร่วมกับการใช้เทคนิคการเชื่อมเส้นทางใหม่ (path relinking technique) เพื่อสร้างคุณสมบัติการค้นหาคำตอบค่าใกล้เคียงแบบใหม่ อัลกอริธึมนี้ได้ถูกนำไปใช้แก้ปัญหาการจัดตารางการทำงานของ

โรงงาน (job shop scheduling problem) ได้ผลเป็นที่น่าพอใจ การค้นหาแบบตามูเชิงคัดแปลง (Zhang, D., Fu, Z., and Zhang, L.C., 2005) ชนิดนี้ กำหนดพารามิเตอร์ความยาวรายการตามู ด้วยกระบวนการผ่าเหล่า (mutation operation) ของจินตคติอัลกอริทึมซึ่งช่วยเพิ่มความสามารถ การค้นหาคำตอบวงกว้าง และได้มีการประยุกต์ใช้อัลกอริทึมนี้กับการแก้ปัญหาลดค่าต้นทุนเสีย ในระบบจำหน่ายขนาดใหญ่ สามารถทำให้สมรรถนะการค้นหาชนิดนี้ดีกว่าการค้นหาแบบตามู ดั้งเดิมการค้นหาแบบตามูเชิงคัดแปลง (Glover, F., 2006) ชนิดนี้เป็นการพัฒนากระบวนการ บรานซ์ และบาวด์ (branch and bound procedure) ของโกลเวอร์ ที่เสนอไว้ครั้งแรกเมื่อ ค.ศ.1978 (Glover, F., 1978) โดยการแทนที่หน่วยความจำการค้นหาแบบต้นไม้ (tree search memory) ด้วย ข่ายงานหน่วยความจำแบบปรับตัวได้ของการค้นหาแบบตามูและได้ใช้กลยุทธ์ร่วมแบบใหม่ ที่มีความอ่อนตัวมากกว่ากลไกในกระบวนการบรานซ์และบาวด์ นอกจากนี้ยังได้ปรับปรุงกระบวนการ เข้มข้น (intensification procedure) และกระบวนการหลากหลาย (diversification procedure) โดยอิง แนวคิดของการค้นหาแบบกระจาย (scatter search) การวิเคราะห์ความถี่ (frequency analysis) และความแปรปรวนของหน่วยความจำปรับตัวที่เรียกว่า หน่วยความจำที่ฝังแบบจำลอง (model embedded memory) อัลกอริทึมนี้ถูกนำไปใช้แก้ปัญหาคำตอบโปรแกรมจำนวนเต็มผสม (mixed integer programming : MIP) ได้ผลเป็นที่น่าพอใจ

2.2.4 การค้นหาแบบตามูเชิงความน่าจะเป็น (probabilistic tabu search, PrTS)

การค้นหาแบบตามูเชิงความน่าจะเป็น (Xu, J., Chiu, S.Y., and Glover, F., 1997) อัลกอริทึมการค้นหาแบบตามูแบบดั้งเดิมพิจารณาทิศการเคลื่อนตัวของการค้นหา เพียงพิจารณาจากรายการตามูและค่าฟังก์ชันวัตถุประสงค์ต่ำสุดของคำตอบตัวเลือก แต่ PrTS แตกต่างตรงที่ ประยุกต์ใช้ความน่าจะเป็นเข้าร่วมกับการเลือกคำตอบถัดไปด้วย จึงช่วยแก้ไขข้อผิดพลาดของ หน่วยความจำตามูที่มีสัญญาณรบกวน กลยุทธ์รายการคำตอบทางเลือกและกลยุทธ์ความเข้มข้นที่ อาศัยการนำคำตอบที่เหมาะสมกลับมาพิจารณาใหม่ เทคนิคนี้จึงมีชื่อเรียกว่าการคัดเลือกทิศการ เคลื่อนตัวเชิงความน่าจะเป็น (probabilistic move selection) และเป็นที่มาของชื่ออัลกอริทึมนี้นั่นเอง อัลกอริทึมนี้ได้ถูกนำไปแก้ปัญหาคำตอบแบบเคลื่อนย้ายสื่อสารทางไกลได้ผลเป็นที่น่าพอใจ การค้นหาแบบตามูเชิงความน่าจะเป็น (Kochetov, Y.A., and Goncharov, E.N., 2000) เป็น อัลกอริทึมที่ใช้กลไกการสร้างคำตอบค่าใกล้เคียงแบบสุ่มที่มีค่าขีดเริ่มคงตัวนอกจากนี้ ยังประยุกต์กฎเชิงปรับตัวเพื่อจัดการความยาวรายการตามู อัลกอริทึมนี้จะสร้างลูกโซ่มาคอฟแบบไม่ เป็นรายคาบจำนวนจำกัดจากเซตที่เหมาะสมภายในเซตคำตอบที่ได้ คุณสมบัติเช่นนี้ยืนยันการลู่เข้า หาผลเฉลยเชิงเส้นกำกับของคำตอบที่ดีที่สุดที่ได้ ซึ่งจะนำไปสู่การพบคำตอบทั่วไปที่ต้องการ อัลกอริทึมนี้ได้ถูกนำไปใช้แก้ปัญหาคำตอบการจัดวางตำแหน่งเครื่องอำนวยความสะดวกแบบหลายช่วง (multi-stage uncapacitated facility location problem) ได้ผลเป็นที่น่าพอใจ การค้นหาแบบตามู

เชิงความน่าจะเป็น (Ghosh, D., 2003) เป็นอัลกอริทึมที่ใช้ปฏิบัติการก่อนกระบวนการ (preprocessing) สำหรับหาค่าความน่าจะเป็นของเซตค่ายอด ต่าง ๆ ที่สัมพันธ์กับคำตอบค่าเหมาะสม และใช้ค่าความน่าจะเป็นนี้ กำหนดขนาดของคำตอบค่าใกล้เคียงที่เหมาะสม อัลกอริทึมนี้ได้ถูกนำไปแก้ปัญหาลำดับของต้นไม้กระจายตัวน้อยที่สุด (generalized minimum spanning tree problem) ได้ผลเป็นที่น่าพอใจ

2.2.5 การค้นหาแบบตามูเชิงผสมผสาน (hybrid tabu search, HTS)

การค้นหาแบบตามูเชิงผสมผสาน (Ongsakul, W., and Bhasaputra, P., 2002) มีโครงสร้างหลักเป็น การค้นหาแบบตามูธรรมดา และมีอัลกอริทึมการอบอ่อนจำลอง (simulated annealing algorithm) ทำหน้าที่สร้างคำตอบค่าใกล้เคียง (neighborhood solution) ให้กับการค้นหาแบบตามูหลัก นอกจากนี้ยังอาศัยเงื่อนไขการยอมรับเชิงความน่าจะเป็น (probabilistic acceptance criterion) ของอัลกอริทึมการอบอ่อนมาทดแทน ระดับความปรารถนา (aspiration level) ของอัลกอริทึมการค้นหาแบบตามูอีกด้วย รวมทั้งแผนการอบอ่อน (cooling scheduling) และรายชื่อตามูได้รับการประยุกต์ใช้ในอัลกอริทึมนี้ด้วย การค้นหาแบบตามูเชิงผสมผสาน (Gong, D., Zhou, Y., Guo, X., Xiaoping, M, and Ming, Li, 2002) ชนิดนี้ประกอบขึ้นจากอัลกอริทึมหลักสองชนิดคือ จินตนาการอัลกอริทึม และ อัลกอริทึมการค้นหาแบบตามู ปกติแล้วจินตนาการอัลกอริทึมจะประกอบด้วย กระบวนการลดปริภูมิการค้นหาและกระบวนการคงความหลากหลายของวิวัฒนาการในประชากร ซึ่งกระบวนการทั้งสองมีแนวทางที่ขัดแย้งกัน (trade-off) จึงมักก่อให้เกิดปัญหาการหาค่าที่เหมาะสม ดังนั้นเพื่อลดความขัดแย้งดังกล่าวจึงอาศัยการค้นหาแบบตามูมาหาค่าที่เหมาะสม การค้นหาแบบตามูเชิงผสมผสาน (Mori, H., and Tani, H., 2003) ชนิดนี้ เป็นการพัฒนาการค้นหาแบบตามูเชิงขนาน (parallel tabu search : PTS) โดยอาศัย การหาค่าเหมาะที่สุดแบบอันดับ (ordinal optimization : OO) มาช่วยสร้างคำตอบค่าใกล้เคียง (neighborhood) ให้กับ PTS ทำการเลือกในรูปแบบของความน่าจะเป็น การค้นหาแบบตามูเชิงผสมผสาน (Kalinli, A., 2004) ชนิดนี้ประกอบขึ้นจาก อัลกอริทึมการค้นหาแบบตามู และ จินตนาการอัลกอริทึม โดยการค้นหาแบบตามูจำนวนมากว่าหนึ่งกระบวนการทำงานไปพร้อม ๆ กัน แล้วส่งผลการค้นหาไปให้จินตนาการอัลกอริทึมผ่านกระบวนการไขว้ (crossover operation) เพื่อสร้างผลการค้นหาใหม่ จนนำไปสู่การค้นหาสุดท้ายที่ดีที่สุด การค้นหาแบบตามูเชิงผสมผสาน (Swarnkar, R., and Tiwari, M.K., 2004) ชนิดนี้มีโครงสร้างหลักเป็น อัลกอริทึมการอบอ่อนจำลอง และมีการค้นหาแบบตามูทำหน้าที่สร้างหน่วยความจำระยะสั้น สำหรับจดจำคำตอบที่เพิ่งพบในช่วงเวลาหนึ่ง เป็นการป้องกันการติดลู่ค่าต่ำสุดเฉพาะถิ่น แต่ธรรมชาติของการค้นหาแบบตามูเป็นเชิงกำหนด (deterministic) มักประสบกับปัญหาการวนซ้ำ ซึ่งก็สามารถหลีกเลี่ยงปัญหานี้ด้วยคุณสมบัติทางสโทแคสติก (stochastic characteristic) ของอัลกอริทึมการอบอ่อน ดังนั้นเมื่อรวมเอาคุณสมบัติที่ดีของทั้งสองอัลกอริทึมเข้า

ด้วยกัน แล้วนำไปประยุกต์ใช้กับปัญหากระบวนการผลิตแบบอ่อนตัวจึงให้ผลการค้นหาที่ดีอย่างน่าพอใจ การค้นหาแบบตามูเชิงผสมผสาน (Liu, Y., Liu, Y., Wang, L., and Chen, K., 2005) ชนิดนี้ มีการค้นหาแบบตามูเป็นโครงข่ายงานหลักที่มีการเสริมสมรรถนะการหาคำคำตอบด้วยปฏิบัติการเค-มีนส์ (K-means operation) ซึ่งเป็นวิธีการในอัลกอริทึมเค-มีนส์ (K-means algorithm) และมีการเพิ่มศักยภาพการสร้างคำตอบค่าใกล้เคียงให้กับการค้นหาแบบตามู ด้วยปฏิบัติการผ่าเหล่า (mutation operation) ซึ่งเป็นวิธีการในจินตนาการอัลกอริทึม การค้นหาแบบตามูเชิงผสมผสาน (Pedroso, J.P., and Kubo, M., 2005) ชนิดนี้ มีการค้นหาแบบตามูเป็นโครงข่ายงานหลัก ซึ่งคงไว้เฉพาะหน่วยความจำระยะสั้น (short term memory) และผสานอัลกอริทึมตรึงและผ่อนคลาย (relax-and-fix-one-product) เข้ามาเสริมสมรรถนะความหลากหลายและเพื่อช่วยปลดปล่อยการติดกับดัก เมื่อการค้นหาแบบตามูดำเนินการไปจนพบว่าคำตอบไม่ได้รับการปรับปรุงในช่วงระยะเวลาหนึ่งแล้ว จะมีกลไกไปกระตุ้นให้อัลกอริทึมตรึงและผ่อนคลายเริ่มทำงาน หลักการทำงานของอัลกอริทึมตรึงและผ่อนคลายเริ่มต้นด้วยการไปลบรายการตามู ที่บันทึกไว้บางส่วนหรือทั้งหมด หลังจากนั้นจึงเริ่มสร้างคำตอบค่าใกล้เคียงขึ้นมาใหม่เพื่อปรับปรุงโอกาสของการพบคำตอบที่ดีขึ้น แล้วทำการเคลื่อนตัวไปยังคำตอบที่ดีที่สุด เมื่อการค้นหาเริ่มเข้าสู่ภาวะปกติ อัลกอริทึมตรึงและผ่อนคลายหยุดทำงาน การค้นหาจะกลับมาอยู่ในโครงสร้างการค้นหาแบบตามูเช่นเดิม การค้นหาแบบตามูเชิงผสมผสาน (Chen, J-S., Pan, J.C-H., and Wu, C-K., 2006) ชนิดนี้ แตกต่างจาก TS ดังเดิม คือ เมื่อกระบวนการค้นหาไม่พบคำตอบที่ดีขึ้นจากค่าใกล้เคียงในช่วงเวลาหนึ่ง กฎการจ่าย (dispatching rule) ที่เป็นวิธีการศึกษาสำนึก (heuristic) ของปัญหาเฉพาะด้านจะเข้าไปสร้างบริเวณคำตอบใหม่แทน ทำให้ HTS สามารถหนีการติดล็อกและเคลื่อนตัวไปหาคำตอบที่ดีขึ้นได้ พารามิเตอร์ที่สำคัญเจ็ดค่าของอัลกอริทึมนี้ ได้แก่ คำตอบเริ่มต้น การค้นหาคำตอบค่าใกล้เคียง การเลือกทิศการเคลื่อนที่ รายการตามู คำตอบที่ดีที่สุด กฎการหยุด และ วิธีการผสมผสาน วิธีการผสมผสานในอัลกอริทึมนี้จะกระทำกับคำตอบล่าสุดที่ไม่ได้รับการปรับปรุงมานานถึงระยะเวลาหนึ่ง โดยการแบ่งคำตอบนั้นออกเป็นสามส่วนแต่ละส่วนมีจำนวนบิตเป็นไปตามการสุ่ม หลังจากนั้นจึงทำการผสมผสานทำนองเดียวกับกระบวนการจินตนาการ ก็จะได้คำตอบใหม่ที่มีคุณสมบัติหนีการติดล็อกได้ดี การค้นหาแบบตามูเชิงผสมผสาน (Victoire, T.A.A., and Jeyakumar, A.E., 2006) ชนิดนี้ เกิดจากรวมกันของสามอัลกอริทึมได้แก่ การค้นหาแบบตามู การหาค่าเหมาะที่สุดแบบฝูงอนุภาค (particle swarm optimization : PSO) และเทคนิคการโปรแกรมกำลังสองเชิงลำดับ (sequential quadratic programming technique : SQP) เพื่อแก้ปัญหาการวางแผนการผลิตไฟฟ้า (unit commitment problem : UCP) TS ทำหน้าที่แก้ปัญหาโปรแกรมย่อยแบบผสมผสาน ของ UCP โดยสร้างรูปแบบการรบกวนแบบสุ่ม (random perturbation scheme) และสร้างตารางพันธกรณีเริ่มต้น (initial feasible commitment schedule) ส่วน PSO และ SQP ใช้แก้ปัญหาโปรแกรมย่อยแบบไม่เชิงเส้นของ UCP

การค้นหาแบบตามูเชิงผสมผสาน (Tahir, M.A., Bouridane, A., and Kurugollu, F., 2007) ชนิดนี้อาศัยแนวคิดแบบการค้นหาแบบตามูมาช่วยกระบวนการเลือกคุณลักษณะและกระบวนการถ่วงน้ำหนักภายในการคัดแยก K-NN (K-nearest-neighbor classifier) ซึ่งเป็นการคัดแยกที่นิยมใช้ในเทคโนโลยีการตรวจรู้รูปแบบ (pattern recognition)

2.2.6 การค้นหาแบบตามูเชิงปรับตัว (adaptive tabu search : ATS)

การค้นหาแบบตามูเชิงปรับตัว (Kolahan, F., and Liang, M., 1998) ชนิดนี้ได้เสนอกลไกการสร้างคำตอบค่าใกล้เคียงแบบปรับตัว (adaptive neighborhood generation : ANG) ที่สามารถปรับย่านการสลับที่ (transposition range) ให้เหมาะสมกับการปรับปรุงฟังก์ชันวัตถุประสงค์ อยู่เสมอ คุณสมบัติที่แตกต่างจากการค้นหาแบบตามูดั้งเดิมมีอยู่สองประการคือการปรับเปลี่ยนย่านการสลับที่ตามการปรับปรุงฟังก์ชันวัตถุประสงค์ กับ การลดย่านการสลับที่สองด้าน (double-ended transposition range) อัลกอริทึมนี้ได้ถูกนำไปใช้หาค่าเหมาะที่สุดในแบบได้อย่างเสียอย่าง (best trade-off) ระหว่างเป้าหมายกระบวนการผลิตแบบตรงเวลาพอดี (just-in-time production) กับ ภาวะกดดันช่วงเวลาการผลิต (processing time compression) และ ค่าใช้จ่ายที่เพิ่มขึ้น (extension costs) ผลการทดสอบอัตราการรู้เข้าและคุณภาพของคำตอบ เมื่อเปรียบเทียบกับ การสร้างคำตอบค่าใกล้เคียงวิธีอื่น พบว่า ATS ชนิดนี้ให้ผลดีกว่าอย่างเห็นได้ชัด การค้นหาแบบตามูเชิงปรับตัว (Franca, P.M., Sosa, N.M., and Pureza, V., 1999) ประกอบด้วยสองกลไกหลักคือ กลไกการเริ่มต้นคำตอบ (initial solution) ซึ่งใช้การศึกษานิกแบบสร้างเสริม (constructive heuristics) ที่อาศัยทั้งน้ำหนัก (weight) และ ระยะทาง (distance) ในการหาค่าที่เหมาะสม ส่วนกลไกที่สองคือ กลไกการสร้างคำตอบค่าใกล้เคียง (neighborhood generation) อาศัยการศึกษานิกแบบการค้นหาเฉพาะถิ่น (local search heuristics) ซึ่งผนวกเทคนิคการสลับคู่ (pairwise interchange) และ เทคนิคการเติม (insertion) มาเสริมสมรรถนะ อัลกอริทึมนี้ได้้นำไปใช้แก้ปัญหาการจัดกลุ่มลูกค้า (customer clustering problem) ได้ผลการทดสอบเป็นที่น่าพอใจ การค้นหาแบบตามูเชิงปรับตัวฉบับของคณะวิศวกรรมศาสตร์และเทคโนโลยีสุรนารี ได้เสนอสองกลไกหลักเพื่อการปรับตัวหนีการติดกับและการปรับตัวให้มีโอกาสพบคำตอบวงกว้างให้รวดเร็วขึ้น สองกลไกดังกล่าวก็คือกลไกย้อนรอยการค้นหา (back-tracking mechanism) และกลไกปรับรัศมีการค้นหา (adaptive radius mechanism) ถ้าพิจารณาเชิงยุทธศาสตร์ของการค้นหา แล้วการเสริมกลไกทั้งสองเปรียบได้กับการเสริมความแข็งแรงให้ทั้งกลยุทธ์ความหลากหลายและกลยุทธ์ความเข้มข้นให้กับ การค้นหาแบบตามูดั้งเดิม ซึ่งภายหลังจากที่เพิ่มกลไกทั้งสองเข้าไปแล้ว ส่งผลทำให้การค้นหาแบบตามูเชิงปรับตัวใหม่นี้มีสมรรถนะการค้นหาที่ดีขึ้นอย่างน่าพอใจ และยังสามารถประยุกต์ใช้ในการแก้ปัญหาสำหรับงานที่มีความซับซ้อนสูง เช่น ระบบควบคุม ระบบไฟฟ้ากำลัง

และระบบไฟฟ้าแรงสูง เป็นต้น รายละเอียดของอัลกอริธึมการค้นหาแบบตามูเชิงปรับฉบับนี้จะได้รับการทบทวนไว้ในหัวข้อถัดไป

2.3 สรุป

บทที่ 2 นี้ได้นำเสนอการปฏิสน์วรรณกรรมและงานวิจัยที่เกี่ยวข้องกับพัฒนาการของอัลกอริธึมการค้นหาแบบตามู โดยอาศัยฐานข้อมูลจาก IEEE IEICE SpringerLink Science-Direct และ Google การปฏิสน์วรรณกรรมดังกล่าวมาแล้วข้างต้น สามารถแบ่งได้เป็นสองกลุ่มใหญ่ๆ ตามจำนวนตัวประมวลผลที่ใช้งาน ได้แก่ กลุ่มอัลกอริธึมการค้นหาแบบตามูเชิงอนุกรม (sequential tabu search algorithm) ที่มีการใช้งานตัวประมวลผลเพียงตัวเดียว และกลุ่มอัลกอริธึมการค้นหาแบบตามูเชิงขนาน (parallel tabu search algorithm) ซึ่งใช้ตัวประมวลผลมากกว่าหนึ่งในการดำเนินการค้นหาไปพร้อม ๆ กัน เพื่อเพิ่มความเร็วการค้นหาให้สูงขึ้น เทคนิคที่พบการใช้ในทั้งสองกลุ่ม ได้แก่ เทคนิคการสร้างคำตอบเริ่มต้น เทคนิคสร้างคำตอบใกล้เคียง เทคนิคการเคลื่อนที่ เทคนิคการบริหารรายการตามู และ เทคนิคการเริ่มต้นใหม่ เป็นต้น ซึ่งเทคนิคเหล่านี้เป็นเสมือนเครื่องมือในสองกล่องเครื่องมือของอัลกอริธึมการค้นหาแบบตามู นั่นคือ กลยุทธ์ความหลากหลาย (diversification strategy) และกลยุทธ์ความเข้มข้น (intensification strategy) นั่นเอง และการค้นหาแบบตามูเชิงปรับฉบับที่เสนอโดยคณะผู้วิจัยสาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารี ก็จัดอยู่ในกลุ่มการค้นหาแบบตามูเชิงอนุกรม เนื่องจากใช้ตัวประมวลผลเดียวด้วยเช่นกัน

ผลจากการปฏิสน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง จะถูกใช้เป็นข้อมูลอ้างอิง และเป็นแนวทางสำหรับการดำเนินงานวิจัยต่อไป

บทที่ 3

การค้นหาแบบตามูเชิงปรับตัว

3.1 กล่าวนำ

การทบทวนการค้นหาแบบตามูเชิงปรับตัวที่เสนอโดยคณะวิจัยมหาวิทยาลัยเทคโนโลยีสุรนารี ได้แบ่งการนำเสนอออกเป็นสองหัวข้อย่อย ได้แก่ ส่วนที่หนึ่งเป็นการอธิบายอัลกอริธึม และส่วนที่สองเป็นการแสดงผลการทดสอบสมรรถนะพร้อมการวิเคราะห์ ทั้งนี้เพื่อให้เกิดความชัดเจนและความเข้าใจที่ดีขึ้น

3.2 การค้นหาแบบตามูเชิงปรับตัว

การค้นหาแบบตามูเชิงปรับตัวฉบับนี้ ถูกพัฒนาต่อยอดอัลกอริธึมการค้นหาแบบตามูดั้งเดิม โดยเพิ่มกลไกย้อนรอยการค้นหา (back-tracking mechanism) และกลไกปรับรัศมีการค้นหา (adaptive radius mechanism) กลไกย้อนรอยการค้นหาเป็นกระบวนการดึงเอาคำตอบที่เคยพบแล้วและอยู่ในรายการตามู มาเป็นคำตอบเริ่มต้นตัวใหม่แทนหาคำตอบถัดไปด้วยกระบวนการปกติ ทั้งนี้เนื่องจากบางโอกาสกระบวนการค้นหาที่ไม่สามารถหาคำตอบถัดไปที่ดีกว่าเดิมได้ หมายความว่า การค้นหากำลังอยู่ในสถานการณ์ติดกับดัก (entanglement) นั่นเอง คณะผู้วิจัยมหาวิทยาลัยเทคโนโลยีสุรนารีจึงเสนอกลไกย้อนรอยนี้มาเพื่อแก้ปัญหาดังกล่าวโดยตั้งเงื่อนไขการเรียกใช้กลไกย้อนรอยนี้ด้วยจำนวนรอบสูงสุดที่ไม่สามารถหาคำตอบถัดไปที่ดีกว่าได้ ก็ให้มีการเรียกใช้กลไกดังกล่าวเพื่อเปิดโอกาสให้การค้นหาได้เปลี่ยนพื้นที่การค้นหาใหม่ ส่วนกลไกปรับรัศมีการค้นหา ทำหน้าที่ปรับรัศมีการค้นหาให้สั้นลงโดยยังรักษาจำนวนคำตอบใกล้เคียงที่จะถูกสุ่มเลือกภายในรัศมีการค้นหาเพื่อเป็นคำตอบถัดไปไว้เท่าเดิม ซึ่งเปรียบเสมือนการเพิ่มความสามารถของการมองเห็นคำตอบวงกว้างให้ดีขึ้นอีก

การค้นหาแบบตามูที่เสนอนี้ เริ่มต้นการค้นหาด้วยการสุ่มคำตอบแรกภายในปริภูมิการค้นหาแล้วกำหนดให้ เป็นศูนย์กลางเพื่อหาคำตอบถัดไปในระยะรัศมีที่กำหนด คำตอบถัดไปจะเป็นคำตอบที่ให้ค่าการปรับปรุงฟังก์ชันวัตถุประสงค์ที่สุดในจำนวนที่สุ่มได้ การค้นหาดำเนินงานแบบวนซ้ำจนกระทั่งการปรับปรุงฟังก์ชันวัตถุประสงค์ถึงค่าขีดเริ่มที่กำหนดไว้ของกลไกปรับรัศมี รัศมีการค้นหาจะหดสั้นเข้าตามลำดับที่ตั้งไว้เพื่อให้พื้นที่การสุ่มคำตอบถัดไปแคบลง เป็นการเพิ่มโอกาสและเพิ่มความเร็วการพบคำตอบวงแคบเฉพาะถิ่นอีกด้วย

ขั้นตอนโดยสรุปของอัลกอริธึมการค้นหาแบบตาบ

ขั้นตอนที่ 0 (การเริ่มต้น)

สร้างคำตอบเริ่มต้น กำหนดคำตอบข้างเคียง กำหนดคำตอบที่ดีที่สุด AC TC TL และ ตัวนับรอบการค้นหาไว้ที่ค่าเริ่มต้น

ขั้นตอนที่ 1 (การวนรอบ)

สร้างหรือรับค่าปริภูมิการค้นหา สร้างชุดคำตอบที่เป็นไปได้จากคำตอบข้างเคียง คำตอบเริ่มต้นหรือคำตอบก่อนหน้านี้และประเมินค่าวัตถุประสงค์ของคำตอบที่เป็นไปได้เหล่านั้น หากหนึ่งในคำตอบที่เป็นไปได้ให้ค่าวัตถุประสงค์ต่ำกว่าคำตอบที่ดีที่สุดปัจจุบันแล้วให้แทนที่ด้วยคำตอบที่เป็นไปได้นั้นแทน หลังจากนั้นจึงทำการปรับปรุง TL แต่ถ้าหากชุดคำตอบที่เป็นไปได้นั้นให้ค่าวัตถุประสงค์ที่ไม่ดีไปกว่าคำตอบที่ดีที่สุดปัจจุบันก็ให้คงคำตอบที่ดีที่สุดปัจจุบันไว้และบันทึกคำตอบที่เป็นไปได้ที่ดีที่สุดปัจจุบันนั้นไว้ใน TL ด้วย ทั้งนี้เมื่อเวลาการค้นหาหรือจำนวนรอบการค้นหาคำเนินไปพอสมควรแต่กระบวนการค้นหายังไม่สามารถปรับปรุงคำตอบที่ดีที่สุดปัจจุบันได้ให้ไปดำเนินการในขั้นตอนที่ 3 แทน พร้อมทั้งปรับตัวนับจำนวนรอบการค้นหา

ขั้นตอนที่ 2 (เกณฑ์ยุติการค้นหา)

ยุติการค้นหาเมื่อพบคำตอบกว้างหรือพบ TC มิฉะนั้นให้ไปดำเนินการในขั้นตอนที่ 1

ขั้นตอนที่ 3 (เกณฑ์ปรารถนา)

ดำเนินการค้นหาตามหลักของ TS ที่สอดคล้องกับ AC ปรับปรุง TL และตัวนับแล้วไปดำเนินการในขั้นตอนที่ 1

รูปที่ 3.1 อัลกอริธึมการค้นหาแบบตาบดั้งเดิม

ขั้นตอนโดยสรุปของอัลกอริธึมการค้นหาแบบตามเชิงปรับตัว

ขั้นตอนที่ 0 (การเริ่มต้น)

สร้างคำตอบเริ่มต้น กำหนดคำตอบข้างเคียง กำหนดคำตอบที่ดีที่สุด AC AR
BT TC TL และตัวนับรอบการค้นหาไว้ที่ค่าเริ่มต้น

ขั้นตอนที่ 1 (การวนรอบ)

สร้างหรือรับค่าปริภูมิการค้นหา สร้างชุดคำตอบที่เป็นไปได้จากคำตอบข้างเคียง
คำตอบเริ่มต้นหรือคำตอบก่อนหน้านี้และประเมินค่าวัตถุประสงค์ของคำตอบที่
เป็นไปได้เหล่านั้น หากหนึ่งในคำตอบที่เป็นไปได้ให้ค่าวัตถุประสงค์ต่ำกว่า
คำตอบที่ดีที่สุดปัจจุบันแล้วให้แทนที่ด้วยคำตอบที่เป็นไปได้นั้นแทน หลังจากนั้น
จึงทำการปรับปรุง TL แต่ถ้าหากชุดคำตอบที่เป็นไปได้นั้นให้ค่าวัตถุประสงค์ที่ไม่
ดีไปกว่าคำตอบที่ดีที่สุดปัจจุบันก็ให้คงคำตอบที่ดีที่สุดปัจจุบันไว้และบันทึก
คำตอบที่เป็นไปได้ที่ดีที่สุดปัจจุบันนั้นไว้ใน TL ด้วย ทั้งนี้เมื่อเวลาการค้นหาหรือ
จำนวนรอบการค้นหาดำเนินไปพอสมควรแต่กระบวนการค้นหายังไม่สามารถ
ปรับปรุงคำตอบที่ดีที่สุดปัจจุบันได้ให้ไปดำเนินการในขั้นตอนที่ 3 แทน พร้อมทั้ง
ปรับตัวนับจำนวนรอบการค้นหา

ขั้นตอนที่ 2 (เกณฑ์ยุติการค้นหา)

ยุติการค้นหาเมื่อพบคำตอบวงกว้างหรือพบ TC มิฉะนั้นให้ไปดำเนินการใน
ขั้นตอนที่ 1

ขั้นตอนที่ 3 (เกณฑ์ปรารภณา)

ดำเนินการค้นหาตามหลักของอัลกอริธึม ATS ที่สอดคล้องกับ AC หากเกิด
ปรากฏการณ์ติดล็อกให้เรียกใช้ BT หากค่าวัตถุประสงค์ของคำตอบที่ดีที่สุดอยู่ใน
เกณฑ์ที่กำหนดให้เรียกใช้ AR ปรับปรุง TL และตัวนับ แล้วไปดำเนินการใน
ขั้นตอนที่ 1

รูปที่ 3.1 อัลกอริธึมการค้นหาแบบตามเชิงปรับตัวที่เสนอ

ขั้นตอนโดยสรุปของกลไกย้อนรอย (BT)

ขั้นตอนที่ 1 (การรับค่า)

เปิดรับค่าคำตอบที่เผยแพร่ในอดีตจาก TL

ขั้นตอนที่ 2 (กำหนดค่า)

กำหนดให้คำตอบหนึ่งใน TL เป็นคำตอบถัดไปของ ATS

ขั้นตอนที่ 3 (ออกจาก BT)

ออกจาก BT กลับสู่วงรอบการค้นหา ATS

รูปที่ 3.1 อัลกอริธึมการค้นหาแบบกลไกย้อนรอยการค้นหา

ขั้นตอนของกลไกปรับรมีการค้นหา (AR)

ขั้นตอนที่ 1 (การลดรมีการค้นหา)

ดำเนินการลดรมีการค้นหาลงตามสัดส่วนที่สอดคล้องกับค่าฟังก์ชันวัตถุประสงค์

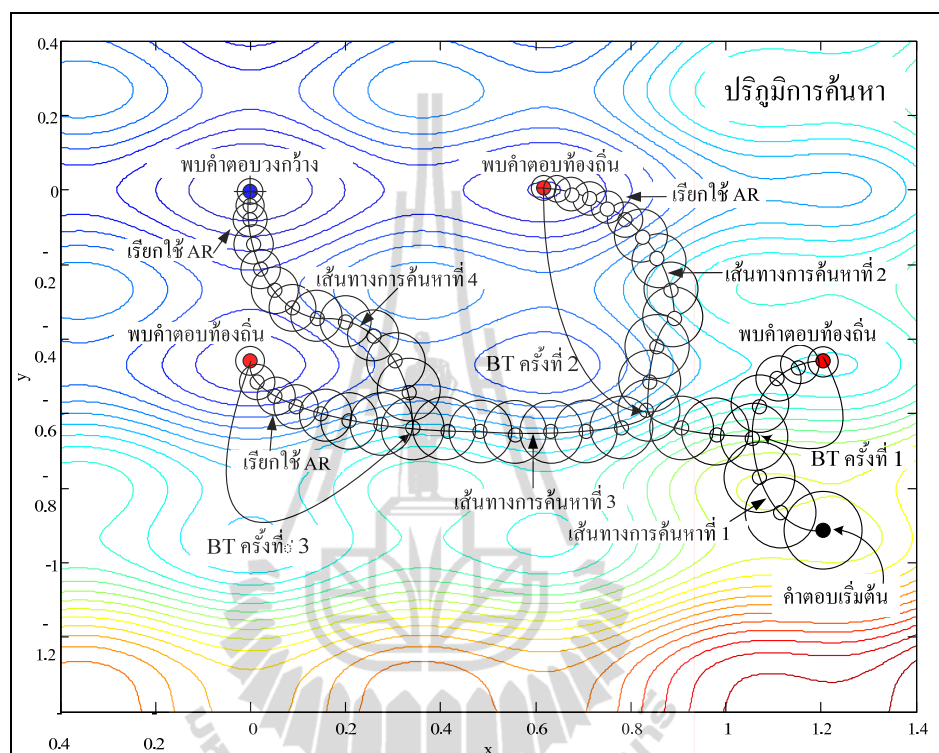
ขั้นตอนที่ 2 (ออกจาก AR)

หากค่าฟังก์ชันวัตถุประสงค์ของคำตอบที่ดีที่สุดปัจจุบันไม่สอดคล้องกับเงื่อนไขการเรียกใช้ AR ให้ใช้รมีการค้นหาเดิมตอนเริ่มต้น แล้วออกจาก AR

รูปที่ 3.1 อัลกอริธึมการค้นหาแบบกลไกปรับรมีการค้นหา

เมื่อการค้นหาเริ่มถึงขีดจำกัดในการปรับปรุงฟังก์ชันวัตถุประสงค์หรือการค้นหาอยู่ใกล้คำตอบวงแคบเฉพาะถิ่นนานพอสมควร กลไกย้อนรอยการค้นหาจะถูกกระตุ้นให้ทำงานอีกเพื่อปลดการติดกับดักท้องถิ่น (local trap) วิธีการที่กลไกย้อนรอยทำให้การค้นหาหลุดจากการติดกับดัก คือการกำหนดคำตอบถัดไป (next solution) โดยเลือกคำตอบหนึ่งในรายการตามมาใช้เป็นคำตอบตั้งต้น ทั้งนี้เป็นคำตอบที่การค้นหาเคยพบมาแล้วในอดีตที่จัดว่าเป็นคำตอบที่มีคุณภาพเพื่อให้เกิดการเปลี่ยนทิศทางการค้นหา และเปลี่ยนตำแหน่งการค้นหาเพื่อหนีล่อลวงไปในตัวด้วย ภายหลังที่การค้นหาหนีล่อลวงเฉพาะถิ่นได้ ก็จะเข้าสู่สภาวะการทำงานปกติ และมุ่งค้นหาคำตอบเฉพาะถิ่นที่ดีขึ้น และจะพบคำตอบเฉพาะถิ่นที่กลายเป็นคำตอบทั่วไปในที่สุด ผู้อ่านสามารถศึกษา

พัฒนาการของการค้นหาแบบตามจนเป็นการค้นหาแบบตามูที่เสนอได้จากอัลกอริธึมดังแสดงในรูปที่ 3.1 ร่วมกับภาพเส้นทางการค้นหาของ ATS ที่เสนอ ดังแสดงในรูปที่ 3.2



รูปที่ 3.2 เส้นทางการค้นหาและสอกลไกหลักของการค้นหาแบบตามูเชิงปรับตัว

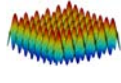
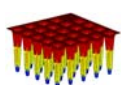
พิจารณารูปที่ 3.2 อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวกล่าวถึง เริ่มต้นการค้นหาด้วยคำตอบเริ่มต้น ที่อยู่ตอนล่างขวาของปฏิภูมิการค้นหา คำตอบที่ได้รับการคัดเลือกให้เป็นคำตอบถัดไปอยู่ทางตอนบนเอียงไปทางซ้ายของคำตอบเดิม เมื่อผ่านการค้นหาไปได้ประมาณสี่รอบได้มีการเรียกใช้กลไกปรับรัศมีการค้นหาหรือ AR ร่วมด้วยสังเกตจากขนาดของรัศมีการค้นหาที่มีแนวโน้มสั้นลงตามลำดับตามเส้นทางการค้นหาที่ 1 (Path#1) จนกระทั่งการค้นหาคำเนินไปจนสุดเส้นทางนี้ก็จะพบคำตอบที่องถิ่นซึ่งตั้งอยู่เหนือคำตอบเริ่มต้น ดังแสดงในรูปที่ 3.2 การวนซ้ำของคำตอบก็จะเกิดขึ้นเนื่องจากค่าฟังก์ชันวัตถุประสงค์ไม่ได้รับการปรับปรุง จนกระทั่งจำนวนการวนซ้ำถึงเกณฑ์ที่กำหนดไว้ กลไกย้อนรอยการค้นหาจะถูกเรียก ซึ่งจากรูปที่ 3.2 ก็จะเป็นการเรียกกลไกนี้เป็นครั้งแรก (BT ครั้งที่ 1) ซึ่งผลการเรียกใช้กลไกย้อนรอยจะได้คำตอบเริ่มต้นใหม่จากหนึ่งคำตอบในรายการตามูที่เคยผ่านมาซึ่ง สิ่งที่เห็นตามรูปที่ 3.2 จะเป็นคำตอบที่อยู่ระหว่างคำตอบเริ่มต้นกับคำตอบเฉพาะถิ่นที่ 1 หลังจากนั้นกระบวนการค้นหาแบบตามูก็ดำเนินไปตามปกติ

ซึ่งได้คำตอบถัดไปเคลื่อนตัวไปทางซ้ายแนวระดับแล้ววกขึ้นทางตอนบนได้ระยะหนึ่ง กลไกปรับ
รัศมีก็ถูกเรียกใช้ตามเกณฑ์อีก จนกระทั่งการค้นหาเดินทางไปพบคำตอบเฉพาะถิ่นที่ 2 และการ
วนซ้ำก็เกิดขึ้นจนต้องเรียกกลไกย้อนรอยเป็นครั้งที่ 2 (BT ครั้งที่ 2) จึงส่งคำตอบเก่าที่มีคุณภาพมา
ให้อยู่ใกล้ ๆ กับคำตอบเริ่มต้น และหลังจากนั้น ATS ก็ดำเนินการค้นหาต่อไปเรื่อย ๆ ตามกลไก
ดังกล่าว จนกระทั่งพบคำตอบท้องถิ่นที่ 3 และในที่สุดคำตอบวงกว้าง (global solution) ถูกพบ
บริเวณซ้ายบนของปริภูมิการค้นหา เกณฑ์ยุติการค้นหา (termination criteria : TC) จึงสั่งหยุดการ
ค้นหา แล้วจึงรายงานคำตอบวงกว้างพร้อมค่าฟังก์ชันวัตถุประสงค์ให้ทราบ ถือเป็นการสิ้นสุด
กระบวนการค้นหา

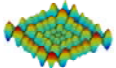
3.3 ผลการทดสอบและการวิเคราะห์

อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวฉบับนี้ได้รับการโปรแกรมด้วยภาษา MATLABTM
เพื่อศึกษาสมรรถนะการค้นหา โดยให้ค้นหาคำตอบวงกว้างของหกฟังก์ชันในปัญหาการหาค่า
เหมาะที่สุดเชิงพื้นที่ ได้แก่ ฟังก์ชันโบฮาเชฟสกี (Bohachevsky function : BF) ฟังก์ชันราสทริกิน
(Rastrigin function : RF) ฟังก์ชันชีเคิล ฟอกซ์ โฮล (Shekel's foxholes function : SF) ฟังก์ชัน
ชูเบอร์ท (Shubert function : ShF) ฟังก์ชันชวีเฟลชนิดที่หนึ่ง (Schwefel function type I : ScF#1)
และ ฟังก์ชันชวีเฟลชนิดที่สอง (Schwefel function type II : ScF#2) รายละเอียดคุณสมบัติของ
ฟังก์ชันทดสอบดังแสดงในตารางที่ 3.1

ตารางที่ 3.1 คุณสมบัติของทั้งหกฟังก์ชันทดสอบ

ชื่อฟังก์ชัน	ลักษณะทั่วไปของฟังก์ชัน	ปริภูมิการ ค้นหา	ภาพร่าง พื้นผิว
Bohachevsky (BF)	$f(x, y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7$ $f_{\min}(0, 0) = 0$	[-2, 2]	
Rastrigin (RF)	$f(x, y) = x^2 + y^2 - 10\cos(2\pi x) - 10\cos(2\pi y) + 20$ $f_{\min}(0, 0) = 0$	[-2, 2]	
Shekel's foxholes (SF)	$f(x_1, x_2) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1}$ where $a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}$ $f_{\min}(-32, -32) = 1$	[-40, 40]	

ตารางที่ 3.1 คุณสมบัติของฟังก์ชันทดสอบ (ต่อ)

ชื่อฟังก์ชัน	ลักษณะทั่วไปของฟังก์ชัน	ปริภูมิการค้นหาค้นหา	ภาพร่างพื้นผิว
Shubert (ShF)	$f(x, y) = \left(\sum_{i=1}^5 i \cos((i+1)x_i + i) \right) \left(\sum_{i=1}^5 i \cos((i+1)y + i) \right)$ $f_{\min}(x_j, y_j) = -186.7309, j \in \{1, 2, \dots, 17, 18\}$	[-10, 10]	
Schwefel I (ScF 1)	$f(x_1, x_2) = 418.9829 \times 2 - \sum_{i=1}^2 \left(x_i \sin \sqrt{ x_i } \right)$	[-500, 500]	
Schwefel II (ScF 2)	$f_{\min}(421, 421) = 0$	[400, 500]	

ฟังก์ชันโบฮาเชฟสกี (Bohachevsky's function) หรือ ฟังก์ชัน BF เป็นฟังก์ชันที่มีตัวแปรสองตัว คือ x กับ y สมการฟังก์ชัน BF ปรากฏคำตอบวงแคบเฉพาะถิ่นจำนวนมากแต่จะมีคำตอบวงกว้างหรือคำตอบ x และ y ที่ทำให้ $f(x, y)$ มีค่าน้อยที่สุดมีอยู่เพียงคำตอบเดียวคือ $x = y = 0$ ที่ให้ค่า $f(x, y) = 0$ ฟังก์ชันที่สองคือ ฟังก์ชันราสทริกิน (Rastrigin's function) หรือ ฟังก์ชัน RF เป็นฟังก์ชันที่มีตัวแปรสองตัวแปรเช่นเดียวกับฟังก์ชัน BF และมีคำตอบวงกว้างอยู่ที่ตำแหน่ง $x = y = 0$ และ $f(x, y) = 0$ แตกต่างจากฟังก์ชัน BF ตรงที่พื้นผิวมีความลึกและชันมากกว่า ฟังก์ชันที่สามได้แก่ ฟังก์ชันชีเคิลฟอกซ์โฮล (Shekel's foxholes function) หรือ ฟังก์ชัน SF เป็นฟังก์ชันที่มีตัวแปรสองตัว คือ x_1 กับ x_2 และมีค่าฟังก์ชัน $f(x_1, x_2)$ ซึ่งพบว่าฟังก์ชัน SF ปรากฏคำตอบวงแคบเฉพาะถิ่นจำนวนมากแต่จะมีคำตอบวงกว้างหรือคำตอบ x_1 และ x_2 ที่ทำให้ $f(x_1, x_2)$ มีค่าน้อยที่สุดมีเพียงคำตอบเดียวคือ $x_1 = x_2 = -32$ ซึ่งจะทำให้ $f(x_1, x_2) = 0.998$ ฟังก์ชันที่สี่ คือ ฟังก์ชันชูเบิร์ด (Shubert's function) หรือ ฟังก์ชัน ShF เป็นฟังก์ชันที่มีตัวแปรสองตัว คือ x กับ y และมีค่าฟังก์ชัน $f(x, y)$ ซึ่งพบว่าฟังก์ชัน ShF มีคำตอบวงกว้างที่มีค่า $f(x, y) = -186.7309$ จำนวนถึง 18 ตัว กระจายทั่วพื้นผิว และปรากฏคำตอบวงแคบเฉพาะถิ่นจำนวนมากเช่นกัน ฟังก์ชันที่ห้า คือ ฟังก์ชันชวีเฟล (Schwefel's function) หรือ ฟังก์ชัน ScF เป็นฟังก์ชันที่มีตัวแปรสองตัว คือ x กับ y และมีค่าฟังก์ชัน $f(x, y)$ ซึ่งพบว่าฟังก์ชัน ScF ปรากฏคำตอบวงแคบเฉพาะถิ่นจำนวนมากแต่จะมีคำตอบวงกว้างหรือคำตอบ x และ y ที่ทำให้ $f(x, y)$ มีค่าน้อยที่สุดมีอยู่เพียงคำตอบเดียวคือ $x = y = 421$ ซึ่งจะทำให้ $f(x, y) = 0$ ในงานวิจัยนี้ต้องการศึกษาผลของขนาดปริภูมิการค้นหาคู่สมรรถนะด้วยจึงทำการแบ่งฟังก์ชัน ชวีเฟลออกเป็นสองชนิดตามขนาดปริภูมิ ดังนี้ ฟังก์ชันชวีเฟล ชนิดที่หนึ่ง ScF1 มีขอบเขตการค้นหาทั้งแกน x และแกน y อยู่ระหว่าง (-500, +500) ส่วนฟังก์ชันชวีเฟล ชนิดที่สอง ScF2 มีขอบเขตการค้นหาทั้งแกน x และแกน y

อยู่ระหว่าง (400 - 500) ถ้าเปรียบเทียบต่อพื้นที่ ฟังก์ชันซีเพลชนิดที่หนึ่งจะใหญ่กว่าชนิดสองถึง 100 เท่า

ตารางที่ 3.2 พารามิเตอร์ของ ATS ที่ใช้ทดสอบ

ฟังก์ชันทดสอบ	พารามิเตอร์ของ ATS									
	ปริภูมิ	รัศมี	จำนวน	BT		AR			TC	
	การค้นหาค้นหา	การค้นหาค้นหา	คำตอบข้างเคียง	Re_{max}	K^{th}	ช่วงที่ 1	ช่วงที่ 2	ช่วงที่ 3	Count max	J_{min}
BF	[2,-2]	0.2 (5%)	30	5	5	$J < 0.1$ $R = 0.002$	$J < 0.001$ $R = 10e-4$	-	10,000	$10e-9$
RF	[2,-2]	0.2 (5%)	30	5	5	$J < 0.1$ $R = 0.002$	$J < 0.001$ $R = 10e-4$	-	10,000	$10e-8$
SF	[40, -40]	0.8 (1%)	30	5	5	$J < 5$ $R = 0.2$	$J < 2$ $R = 0.1$	-	10,000	0.999
ShF	[10, -10]	1 (5%)	30	5	5	$J < 100$ $R = 0.1$	$J < 150$ $R = 0.015$	-	10,000	-186
ScF1	[500, -500]	100 (10%)	20	5	5	$J < 100$ $R = 50$	$J < 10$ $R = 0.01$	$J < 1$ $R = 0.001$	10,000	$10e-4$
ScF2	[500, 400]	1 (1%)	20	5	5	$J < 10$ $R = 0.01$	$J < 1$ $R = 0.001$	-	10,000	$10e-4$

หมายเหตุ : J_{min} คือ ค่าวัตถุประสงค์ขั้นต่ำที่ยอมรับว่าเป็นคำตอบวงกว้าง

รัศมีการค้นหาควรตั้งไว้ไม่เกิน 20 เปอร์เซ็นต์ของปริภูมิการค้นหา เช่นฟังก์ชัน RF กำหนดไว้ 5 เปอร์เซ็นต์ และ ฟังก์ชัน SF กำหนดไว้ 1 เปอร์เซ็นต์ของปริภูมิการค้นหา เป็นต้น จำนวนคำตอบใกล้เคียงของฟังก์ชันแรกได้แก่ ฟังก์ชัน BF ฟังก์ชัน RF ฟังก์ชัน SF และ ฟังก์ชัน ShF เลือกใช้ 30 ค่า ส่วนของสองฟังก์ชันหลังคือฟังก์ชัน ScF1 และ ฟังก์ชัน ScF2 เลือกใช้คำตอบใกล้เคียงจำนวน 20 ค่า ค่าการเข้าคำตอบสูงสุดก่อนการเรียกใช้กลไกย้อนรอยการค้นหาหรือ Re_{max} ตั้งไว้ที่ 5 ครั้งสำหรับทั้งหกฟังก์ชัน และลำดับคำตอบย้อนหลังในรายการตามเมื่อมีการเรียกใช้กลไกย้อนรอยหรือ k^{th} backward selection ตั้งไว้ที่ลำดับที่ 5 ย้อนขึ้นไปจากคำตอบล่าสุด สำหรับทั้งหกฟังก์ชันเช่นกัน ทั้งนี้เป็นไปตามคำแนะนำการตั้งค่าอัลกอริธึมการค้นหาแบบดาวยุชเชิงปรับตัว (Sujitjorn et al., 2006) กลไกปรับรัศมีการค้นหาส่วนใหญ่ใช้แบบสองระดับ ยกเว้นฟังก์ชันเดียวคือฟังก์ชัน ScF1 ต้องใช้ถึงสามระดับเพื่อช่วยให้การค้นหาคำตอบวงกว้างรวดเร็วขึ้น ดังแสดงในตารางที่ 3.2

การตั้งเงื่อนไขยุติการค้นหาหรือ TC กำหนดให้มีค่าใกล้เคียงกับค่าฟังก์ชันวัตถุประสงค์ ค่าตอบวงกว้างของฟังก์ชันนั้น ๆ ตัวอย่างกรณีของฟังก์ชัน BF ณ ค่าตอบวงกว้างจะให้ค่าฟังก์ชันวัตถุประสงค์ประมาณศูนย์จึงตั้ง $TC < 1 \times 10^{-9}$ และของฟังก์ชัน RF ณ ค่าตอบวงกว้างจะให้ค่าฟังก์ชันวัตถุประสงค์ประมาณศูนย์เช่นเดียวกัน แต่เลือกตั้ง $TC < 1 \times 10^{-8}$ เพื่อให้ง่ายต่อการค้นหา และจำนวนรอบการค้นหาเฉลี่ยของ RF ไม่สูงมากนัก

การทดสอบกระทำงานบนเครื่องคอมพิวเตอร์ Intel Celeron (R) ความเร็ว 2.6 จิกกะเฮิรตซ์ ซึ่งมีหน่วยความจำ RAM ขนาด 238 เมกกะไบต์ หน่วยความจำ HDD ขนาด 30 จิกกะไบต์ การทดสอบแต่ละครั้งจะทำการสุ่มเลือกคำตอบเริ่มต้นภายในปริภูมิการค้นหามาหนึ่งคำตอบแล้วปล่อยให้ ATS ทำการค้นหาจบพบเงื่อนไขยุติการค้นหา แล้วบันทึกค่าเวลาการค้นหากับจำนวนรอบการค้นหาที่ใช้ไป ทดสอบจำนวน 50 ครั้งแล้วหาค่าเฉลี่ย ได้ผลการทดสอบเป็นเวลาการค้นหาเฉลี่ย และจำนวนรอบการค้นหาเฉลี่ย แสดงอยู่ในตารางที่ 3.3 ซึ่งจากผลการทดสอบที่ได้รายงานนี้ทั้งหมด ล้วนได้จากเกณฑ์ยุติการค้นหาแบบพหุค่าฟังก์ชันวัตถุประสงค์ ไม่ได้เป็นแบบจำนวนรอบการค้นหาสูงสุด ซึ่งแสดงให้เห็นถึงคุณภาพคำตอบวงกว้างที่คืนนั่นเอง

ตารางที่ 3.3 ผลการทดสอบสมรรถนะของการค้นหาแบบดาบเชิงปรับตัวที่เสนอ

ผลการค้นหา	ฟังก์ชันทดสอบ					
	BF	RF	SF	ShF	ScF1	ScF2
(1) เวลาการค้นหาเฉลี่ย (วินาที)	5.3940	8.2067	2.2837	3.3978	38.1037	0.2522
(2) รอบการค้นหาเฉลี่ย (รอบ)	556.06	759.80	108.22	342.54	336.24	31.24
(1)/(2) (วินาที/รอบ)	0.0097	0.0108	0.0211	0.0099	0.1133	0.0081

แถวที่ 1 ของตารางที่ 3.3 แสดงค่าเวลาการค้นหาเฉลี่ยของฟังก์ชัน BF RF SF ShF ScF1 และ ScF2 ตามลำดับ ส่วนค่าจำนวนรอบการค้นหาเฉลี่ยแสดงอยู่ในแถวที่ 2 พิจารณาผลการทดสอบที่ได้ตามตารางที่ 3.3 พบว่าฟังก์ชัน ScF2 จะใช้เวลาการค้นหาคำตอบวงกว้างน้อยที่สุด คือ 0.2522 วินาที ด้วยรอบการค้นหาเฉลี่ย 31.24 รอบ ถัดขึ้นไปเป็นฟังก์ชัน SF ใช้เวลาการค้นหาคำตอบวงกว้างมากที่สุด คือ 2.2837 วินาที ด้วยรอบการค้นหาเฉลี่ย 108.22 รอบ ลำดับที่ 3 4 และ 5 เป็นฟังก์ชัน ShF ฟังก์ชัน BF และ ฟังก์ชัน RF ตามลำดับ ส่วนลำดับสุดท้ายลำดับที่ 6 คือฟังก์ชัน ScF1 จะใช้เวลาการค้นหาคำตอบวงกว้างมากที่สุด คือ 38.1037 วินาที ด้วยรอบการค้นหาเฉลี่ย 336.24 รอบ

แถวสุดท้ายแถวที่ 3 ของตารางที่ 3.3 เป็นค่าที่ได้จากการคำนวณภายในแต่ละหลัก นั่นคือนำค่าเวลาการค้นหาเฉลี่ยจากแถวที่ 1 หาค่าด้วยค่าจำนวนรอบการค้นหาเฉลี่ยจากแถวที่ 2 ผลลัพธ์ที่ได้จะเป็นเวลาเฉลี่ยที่ใช้ในการค้นหาคำตอบแต่ละรอบการค้นหาของแต่ละฟังก์ชันทดสอบ ความซับซ้อนของคำนวณหรือการประเมินฟังก์ชันวัตถุประสงค์ของแต่ละฟังก์ชันทดสอบ พิจารณาได้จากเวลาที่ตัวประมวลผลใช้คำนวณหรือประเมินแต่ละครั้ง ดังนั้นค่าเวลาเฉลี่ยต่อรอบการค้นหาที่แสดงอยู่ในแถวที่ 3 ของตารางที่ 3.3 นี้ก็บ่งบอกถึงระดับความยากง่ายของฟังก์ชันทดสอบแต่ละแบบได้วิธีหนึ่ง ถ้าเป็นเช่นนั้นย่อมสามารถกล่าวได้ว่าฟังก์ชันที่ใช้เวลาการค้นหาเฉลี่ยต่อรอบการค้นหามากที่สุด ก็คือฟังก์ชันที่มีระดับความยากมากที่สุด สำหรับอัลกอริธึม ATS ถ้าพิจารณาตามแถวที่ 3 ของตารางที่ 3.3 แล้วจะพบว่า ฟังก์ชัน ScF1 มีเวลาการค้นหาเฉลี่ย 0.1133 วินาที/รอบ รองลงไปเป็น ฟังก์ชัน SF ด้วยเวลาการค้นหาเฉลี่ย 0.0211 วินาที/รอบ ถัดไปเป็นฟังก์ชัน RF ด้วยเวลาการค้นหาเฉลี่ย 0.0108 วินาที/รอบ ฟังก์ชัน ShF ด้วยเวลาการค้นหาเฉลี่ย 0.0099 วินาที/รอบ ฟังก์ชัน BF ด้วยเวลาการค้นหาเฉลี่ย 0.0097 วินาที/รอบ และ สุดท้ายฟังก์ชันที่ง่ายที่สุดต่อการค้นหาด้วย ATS ได้แก่ ฟังก์ชัน ScF2 ซึ่งมีเวลาการค้นหาเฉลี่ย 0.0081 วินาที/รอบ นั่นเอง

3.4 สรุป

บทนี้เป็นการทบทวนอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวฉบับมหาวิทยาลัยเทคโนโลยีสุรนารี ที่ปรับปรุงสมรรถนะของอัลกอริธึมการค้นหาแบบตามูดั้งเดิมด้วยกลไกย้อนรอยการค้นหาและกลไกปรับรัศมีการค้นหา และที่สำคัญ ATS ฉบับนี้จะถูกใช้เป็นอัลกอริธึมแกนสำหรับการพัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี โดยรายละเอียดจะกล่าวถึงในหัวข้อถัดไป

บทที่ 4

อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี

4.1 กล่าวนำ

การค้นหาแบบตามูเชิงปรับตัวหลายวิธี (multi-path ATS : MATS) ที่ดำเนินการพัฒนาต่อ ยอดกับ ATS นี้ หมายถึงอัลกอริธึมการค้นหาคำตอบให้กับปัญหาใด ๆ ที่ดำเนินการบนคอมพิวเตอร์ แบบตัวประมวลผลเดี่ยว โดยทำการแบ่งปริภูมิการค้นหาออกเป็นปริภูมิการค้นหาย่อย (sub-search space) ด้วยกลไกแบ่งปริภูมิการค้นหา (partitioning mechanism : PM) ซึ่งมีจำนวนปริภูมิการค้นหาย่อยเท่ากับจำนวนเท่ากับ ATS ย่อย หลังจากนั้น PM จะทำการสุ่มคำตอบภายในแต่ละปริภูมิการค้นหาย่อยเพื่อทำหน้าที่เป็นคำตอบเริ่มต้นให้กับแต่ละ ATS ย่อย หลังจากนั้นจะปล่อยให้ ATS ย่อย เหล่านั้นทำการค้นหาคำตอบอย่างอิสระภายในปริภูมิการค้นหารวมเดียวกันตาม กลไกลำดับการ ค้นหา (sequencing mechanism : SM) แต่ละลำดับทำการค้นหาในช่วงเวลาที่สอดคล้องกับการแบ่ง เวลา (time sharing) เมื่อกระบวนการค้นหาค้นหาไประยะเวลาหนึ่ง กลไกยกเลิกการค้นหา (discarding mechanism : DM) จะทำการยกเลิกการค้นหาบางปริภูมิย่อยที่ฟังก์ชันวัตถุประสงค์ ไม่ได้รับการปรับปรุงมานาน ทั้งนี้คำตอบสุดท้ายก่อนการยกเลิกการค้นหายังคงเก็บรักษาไว้เพื่อ ป้องกันการย้อนกลับมาค้นหาซ้ำอีก (cycling) การลดจำนวนปริภูมิย่อยลง เป็นการเพิ่มโอกาส พบคำตอบวงกว้างในปริภูมิย่อยที่เหลือนมากขึ้น กลไกยกเลิกการค้นหาย่อยจะดำเนินไปจนกระทั่งเหลือ เพียงหนึ่งปริภูมิการค้นหาย่อย เพื่อช่วยเพิ่มความเร็วการค้นหาและเป็นการรักษากลยุทธ์ ความเข้มข้นของการค้นหาแบบตามูเชิงปรับตัวไว้อีกด้วย

4.2 การค้นหาแบบตามูเชิงปรับตัวหลายวิธี (MATS)

MATS เป็นหนึ่งพัฒนาการสืบเนื่องจาก ATS ซึ่งต่อไปจะเรียกว่า ATS แบบเดี่ยว ที่พัฒนาขึ้นโดยคณะวิจัยของมหาวิทยาลัยเทคโนโลยีสุรนารี แนวทางการพัฒนา MATS ดำเนินไป ภายใต้งานวิจัยที่ผู้วิจัยตั้งไว้คือ จะทำอย่างไรให้ อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายหน่วย ค้นหา ดำเนินงานในหลาย ๆ พื้นที่ของปริภูมิการค้นหาโดยมีเป้าหมายให้ใช้เวลาการค้นหาคำตอบ วงกว้างลดลงเมื่อเปรียบเทียบกับ ATS แบบเดี่ยว ภายใต้งานวิจัยว่าต้องดำเนินการได้ด้วยตัว ประมวลผลเดี่ยว ผู้อ่านบางท่านอาจสงสัยถึงเหตุผลที่ต้องตั้งเงื่อนไขดังกล่าว ทั้งนี้เพื่อความสะดวก ในการประยุกต์ใช้อัลกอริธึมนี้กับเครื่องคอมพิวเตอร์พกพาหรือเครื่องคอมพิวเตอร์ตั้งโต๊ะ

แบบตัวประมวลผลเดี่ยว ซึ่งมีการใช้งานกันอยู่แพร่หลายนั่นเอง การตอบโจทย์ข้อนี้คณะวิจัยอาศัยกลยุทธ์ความร่วมมือในการค้นหาคำตอบกว้างของหลาย ๆ หน่วยค้นหา ภายใต้การทำงานของตัวประมวลผลเดี่ยวตามวิธีการแบ่งเวลา (time sharing) ทำให้เกิดจำนวนเส้นทางการค้นหาของ ATS ย่อย ๆ จำนวนมากกว่าหนึ่งเส้นทางการค้นหา ขณะที่ ATS แบบเดี่ยว มีเพียงหนึ่งเส้นทางการค้นหาการทำงานของ ATS ย่อย ของ MATS ทำให้เวลาที่ใช้สร้างเป็นเส้นทางการค้นหานานมากกว่า ATS แบบเดี่ยว จากการที่ต้องคอยเวลาซึ่งกันและกันของ ATS ย่อย แต่ประโยชน์ที่เกิดขึ้นภายใต้ MATS เป็นการเพิ่มความหลากหลายในการสำรวจปริภูมิการค้นหา อันถือเป็นกลยุทธ์ที่สำคัญแบบหนึ่งในการเพิ่มโอกาสการพบคำตอบคุณภาพสูง กลยุทธ์ดังกล่าวมีการเรียกขานว่า diversification strategy หรือ exploration strategy แต่การทำให้ MATS ใช้เวลาการค้นหารวมสั้นกว่า ATS แบบเดี่ยว จำเป็นต้องมีกลยุทธ์ทางการบริหารจัดการที่ดี ดังที่จะได้อธิบายต่อไป

PM ทำหน้าที่แบ่งปริภูมิการค้นหาออกเป็นปริภูมิการค้นหาย่อยแล้วดำเนินการค้นหาคำตอบที่ดีที่สุดในแต่ละปริภูมิย่อยให้ทำหน้าที่เป็นคำตอบเริ่มต้นสำหรับแต่ละ ATS ย่อย ทั้งนี้เพื่อกระจายหน่วยค้นหาให้ทำการสำรวจทั่วทั้งปริภูมิการค้นหา ปริภูมิการค้นหาย่อยเป็นค่าตัวแปรภายใน PM ซึ่งถูกประกาศและถูกยกเลิกการใช้เมื่อสิ้นสุดกลไก PM นั้นหมายความว่าเมื่อ PM ดำเนินการจัดหาคำตอบเริ่มต้นให้กับทุก ATS ย่อย เรียบร้อย ATS ย่อย จะดำเนินการค้นหาต่อไปภายใต้ปริภูมิการค้นหาเดียวกันนั่นเอง

SM ทำหน้าที่ลำดับการค้นหาให้กับแต่ละหน่วยค้นหา ให้การค้นหาทั้งหมดต้องอยู่ภายใต้การประมวลผลเชิงลำดับ และ DM ทำหน้าที่ประเมินโอกาสหรือแนวโน้มที่ ATS ย่อย แต่ละหน่วยจะพบคำตอบคุณภาพสูง และสั่งยกเลิก ATS ย่อย หน่วยใดที่ถูกประเมินว่ามีโอกาสน้อยที่จะพบคำตอบที่มีคุณภาพตามที่ต้องการ

การทำงานของ DM อย่างมีประสิทธิภาพ ซึ่งขึ้นอยู่กับกรอบการออกแบบด้วย จะส่งผลให้เวลาการค้นหารวมไม่ยาวนาน และคาดหวังว่าจะใช้เวลาสั้นกว่า ATS แบบเดี่ยว DM ทำหน้าที่ตัดทอนเส้นทางการค้นหาของ ATS ย่อย ไปเรื่อย ๆ จนเหลือเพียงเส้นทางเดียวที่จะทำการค้นหาต่อไปจนพบคำตอบกว้าง MATS อาจได้รับการเขียนแสดงเป็นขั้นตอนโดยย่อได้ดังรูปที่ 4.1 เกณฑ์ยุติการค้นหา หรือ TC มีสองชนิด ได้แก่ ชนิดที่หนึ่งใช้จำนวนรอบการค้นหาสูงสุด หรือ $Count_{max}$ จะยุติการค้นหาของ ATS ย่อย ทุกตัวในการกำกับของ MATS เมื่อสิ้นสุดการคำนวณในรอบการค้นหา $Count_{max}$ ส่วน TC ชนิดที่สอง คือ คำตอบที่ได้ของ ATS ย่อย ตัวใดตัวหนึ่งอยู่ในเกณฑ์ที่ยอมรับได้ MATS ก็จะสั่ง ATS ย่อย ทุกตัวในการกำกับการยุติการค้นหา TC ชนิดนี้เป็นคุณสมบัติติดตัวของ ATS แบบเดี่ยว อยู่แล้ว และยังคงใช้การได้กับทั้งตัว ATS ย่อย นั้นเองรวมถึง MATS อีกด้วย นั่นหมายความว่า ATS ย่อย ทุกตัวในการกำกับของ MATS สามารถยุติการค้นหาของ MATS ได้ทันทีที่เงื่อนไขถูกพบโดยไม่จำเป็นต้องรอสิ้นสุดรอบการค้นหาของ MATS และไม่จำกัดว่าการยุติการค้นหา

จะต้องเหลือเพียงหน่วยค้นหาเดียว ดังนั้นเกณฑ์ยุติการค้นหา หรือ TC ชนิดที่สอง จึงมีบทบาทสำคัญอย่างยิ่งของ MATS ในการลดเวลาการค้นหาและรักษาคุณภาพคำตอบที่พบไว้ได้ เนื่องจาก ATS ย่อย ทุกตัวสามารถยุติการค้นหาของ MATS ได้ทุกเมื่อที่พบคำตอบคุณภาพตามต้องการนั่นเอง ซึ่งผลการทดสอบที่จะได้กล่าวถึงรายละเอียดในหัวข้อถัดไปนั้น ก็ล้วนได้จาก TC ชนิดที่สองนี้ทั้งนั้น

ขั้นตอนโดยสรุปของอัลกอริทึมการค้นหาแบบตามเชิงปรับตัวหลายวิถี

ขั้นตอนที่ 0 (การเริ่มต้น)

เรียกใช้ PM เพื่อแบ่งปริภูมิการค้นหาออกเป็นปริภูมิการค้นหาย่อยแล้วสร้างคำตอบเริ่มต้นจำนวน N ตัว กำหนดคำตอบข้างเคียง กำหนดคำตอบที่ดีที่สุด AC AR BT TC TL และตัวนับรอบการค้นหาไว้ที่ค่าเริ่มต้นให้กับ ATS ย่อยเหมือนกันทั้ง N ตัว

ขั้นตอนที่ 1 (การวนรอบ)

ATS ย่อยทั้ง N ตัวดำเนินการค้นหาอย่างอิสระภายในปริภูมิการค้นหาเดียวกันและภายใต้การกำกับของ SM ซึ่งจะอนุญาตให้ ATS ย่อยแต่ละตัวดำเนินการค้นหาเพียงหนึ่งรอบการค้นหาเรียงลำดับกัน โดยในวงรอบการค้นหาของ MATS ATS ย่อยทุกตัวเริ่มต้นการวนรอบด้วยการสร้างชุดคำตอบที่เป็นไปได้จากคำตอบข้างเคียงคำตอบเริ่มต้นหรือคำตอบก่อนหน้าและประเมินค่าวัตถุประสงค์ของคำตอบที่เป็นไปได้เหล่านั้น หากหนึ่งในคำตอบที่เป็นไปได้ให้ค่าวัตถุประสงค์ต่ำกว่าคำตอบที่ดีที่สุดปัจจุบันแล้วให้แทนที่ด้วยคำตอบที่เป็นไปได้นั้นแทน หลังจากนั้นจึงทำการปรับปรุง TL แต่ถ้าหากชุดคำตอบที่เป็นไปได้นั้นให้ค่าวัตถุประสงค์ที่ไม่ดีไปกว่าคำตอบที่ดีที่สุดปัจจุบันก็ให้คงคำตอบที่ดีที่สุดปัจจุบันไว้และบันทึกคำตอบที่เป็นไปได้ที่ดีที่สุดปัจจุบันนั้นไว้ใน TL ด้วย ทั้งนี้เมื่อเวลาการค้นหาหรือจำนวนรอบการค้นหาดำเนินไปพอสมควรตามที่ผู้กำหนดจะมีการเรียกใช้ DM แต่กระบวนการค้นหายังไม่สามารถปรับปรุงคำตอบที่ดีที่สุดปัจจุบันได้ให้ไปดำเนินการในขั้นตอนที่ 3 แทน พร้อมทั้งปรับตัวนับจำนวนรอบการค้นหา

ขั้นตอนที่ 2 (เกณฑ์ปรารถนา)

ATS ย่อยแต่ละตัวมีเกณฑ์ปรารถนา BT และ AR ที่ถูกตั้งไว้เหมือนกันอยู่แล้ว แต่เกณฑ์ปรารถนาในระดับส่วนจัด การอย่าง MATS คือ DM ซึ่งทำหน้าที่ลดจำนวนหน่วยค้นหาให้เหลือเพียงหน่วยค้นหาคุณภาพสูงเท่านั้น หลังจากนั้นจึงกลับไปดำเนินในขั้นตอนที่ 1

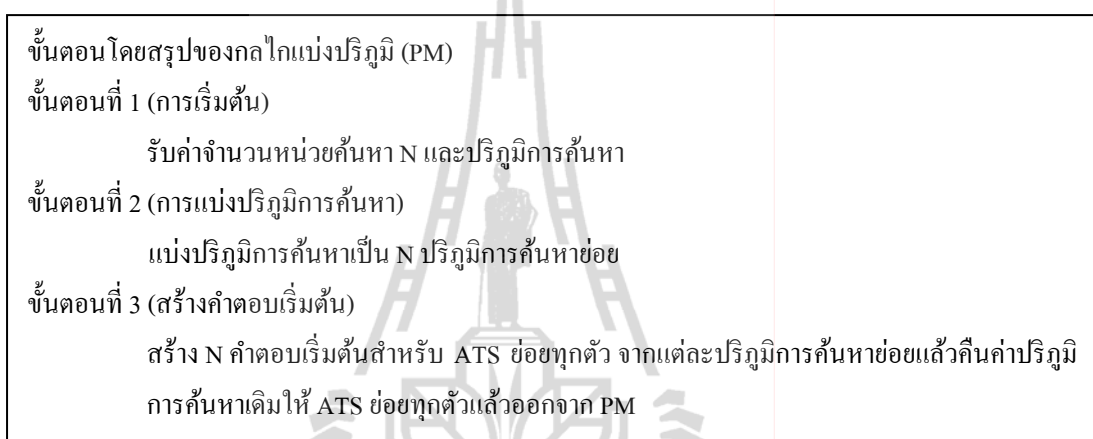
ขั้นตอนที่ 3 (เกณฑ์ยุติการค้นหา)

ATS ย่อยทุกตัวสามารถยุติการค้นหาของ MATS ลงได้เมื่อพบคำตอบวงกว้างหรือพบ TC มิฉะนั้นให้ไปดำเนินในขั้นตอนที่ 1

รูปที่ 4.1 ขั้นตอนการทำงานของ MATS

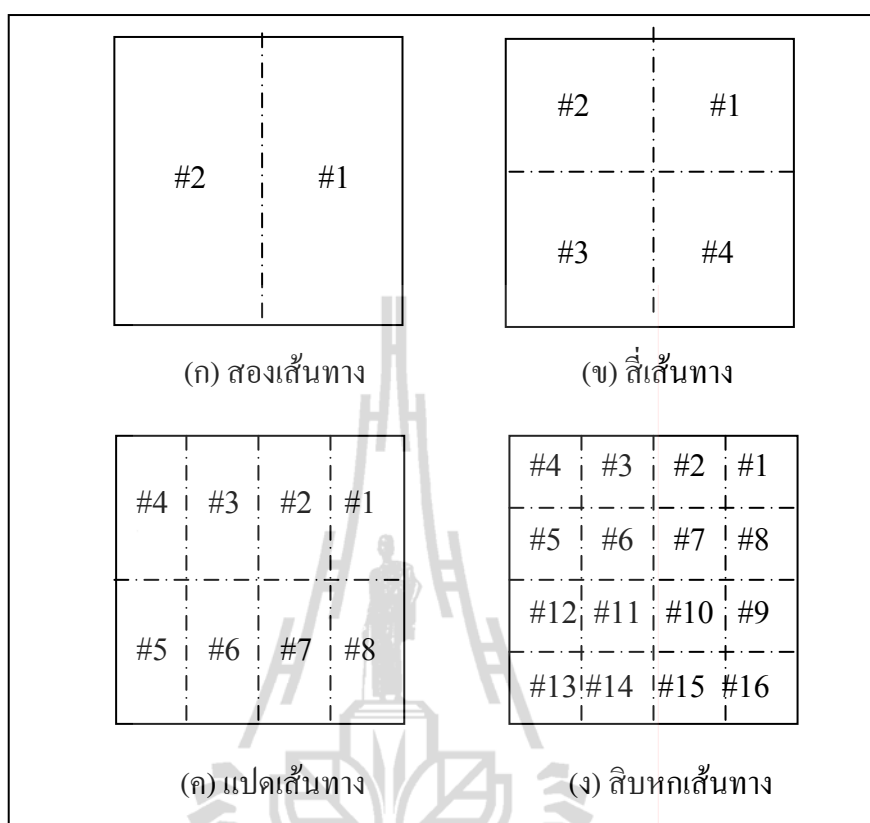
กลไกแบ่งปฏิภูมิการค้นหา (PM)

กลไกแบ่งปฏิภูมิการค้นหา หรือ PM ให้การกระจายคำตอบเริ่มต้นของแต่ละหน่วยค้นหาให้ทั่วถึงทั้งปฏิภูมิการค้นหา เพื่อเพิ่มโอกาสในการพบปฏิภูมีย่อยที่มีคำตอบวงกว้างอยู่ภายใน PM เริ่มต้นด้วยการแบ่งปฏิภูมิการค้นหาทั้งหมดออกเป็นปฏิภูมิการค้นหาย่อย ๆ จากนั้นจึงสร้างคำตอบเริ่มต้นที่ไม่ซ้ำกันให้แต่ละปฏิภูมีย่อย ขั้นตอนการทำงานของ PM แสดงไว้ในรูปที่ 4.2



รูปที่ 4.2 ขั้นตอนการทำงานของ PM

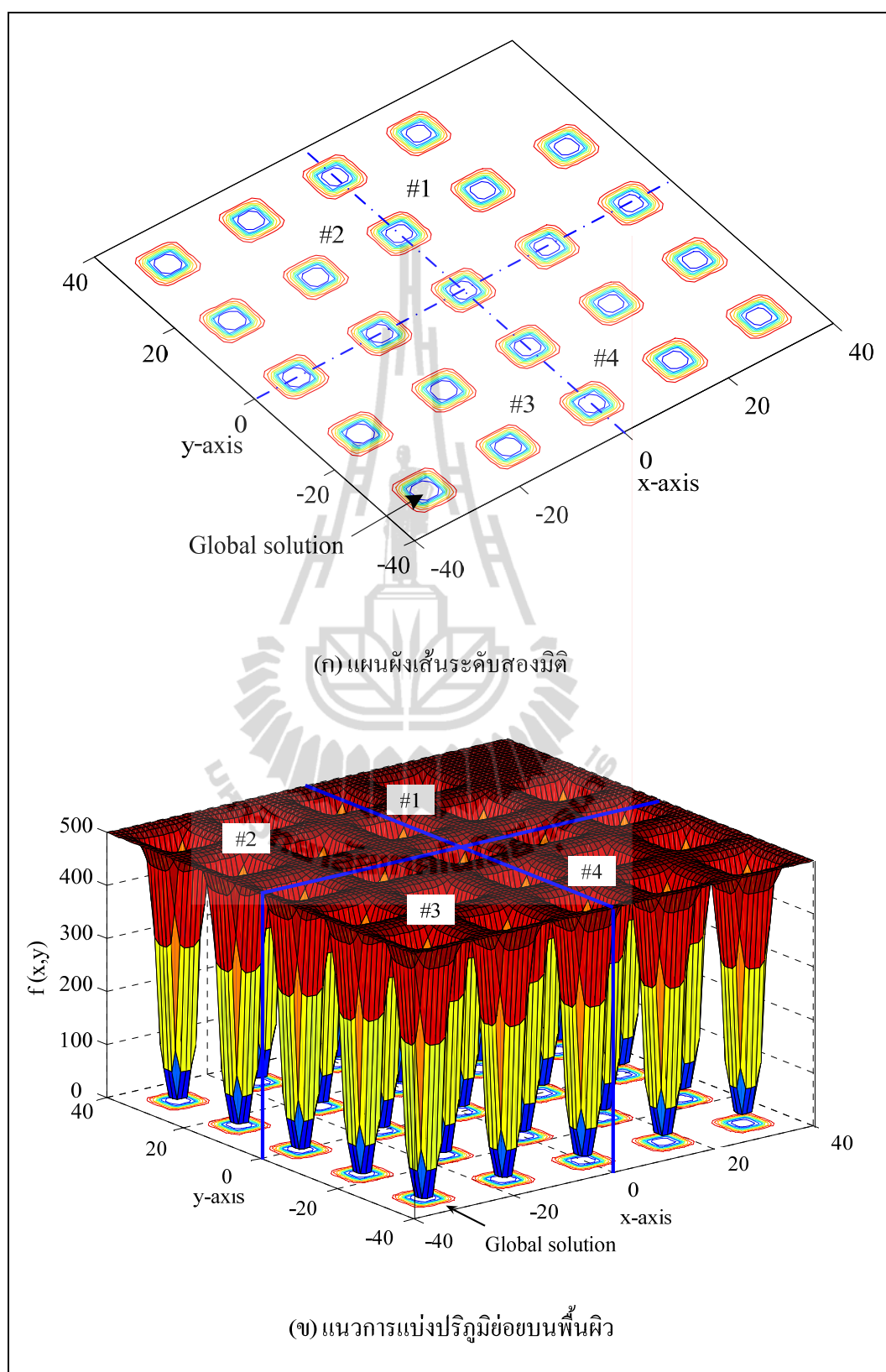
รูปแบบหนึ่งที่เป็นไปได้ของ PM กับปัญหาแบบสองมิติที่ปฏิภูมิการค้นหามีเส้นรอบรูปของปฏิภูมิเป็นสี่เหลี่ยม การแบ่งปฏิภูมิการค้นหาสามารถทำได้โดยใช้เส้นแนว (grid) ทั้งแบบแนวตั้ง (vertical grid) และแบบแนวนอน (horizontal grid) หรือจะใช้สองแบบร่วมกันก็ได้ ดังแสดงในรูปที่ 4.3 โดย รูปที่ 4.3 (ก) แสดงการแบ่งปฏิภูมิการค้นหาวางออกเป็นสองส่วนเท่า ๆ กัน โดยใช้เส้นแนวตั้งแบ่งออกเป็นปฏิภูมิการค้นหาย่อยทางขวาให้ชื่อว่า ปฏิภูมิการค้นหาย่อยที่ 1 หรือ #1 และปฏิภูมิการค้นหาย่อยทางซ้ายมือคือปฏิภูมิการค้นหาย่อยที่ 2 หรือ #2 การแบ่งแบบสองส่วนนี้สามารถทำได้ด้วยวิธีการอื่น เช่น ใช้เส้นแนวนอนแบ่งเป็นส่วนบนกับส่วนล่าง หรือจะใช้เส้นแนวทแยงมุมจากมุมบนซ้ายลงมามุมล่างขวาได้ส่วนบนขวากับส่วนล่างซ้าย เป็นต้น รูปที่ 4.3 (ข) แสดงตัวอย่างการแบ่งปฏิภูมิการค้นหาวางออกเป็นสี่ส่วนเท่า ๆ กัน ด้วยการเพิ่มเติมเส้นแนวนอนให้กับรูปที่ 4.3 (ก) ก็จะได้ปฏิภูมิการค้นหาย่อยจำนวนสี่ส่วนเท่า ๆ กันตามต้องการ หากเพิ่มเส้นแนวตั้งจำนวนสองเส้นทางด้านซ้ายและขวาให้กับ รูปที่ 4.3 (ข) จะได้ปฏิภูมิการค้นหาย่อยจำนวนแปดส่วนเท่า ๆ กัน ดังแสดงในรูปที่ 4.3 (ค) และ รูปที่ 4.3 (ง) แสดงตัวอย่างการแบ่งปฏิภูมิการค้นหาวางออกเป็นสิบหกส่วนเท่า ๆ กัน ตามแบบที่ได้อธิบายข้างต้น



รูปที่ 4.3 ตัวอย่างการแบ่งปริภูมิการค้นหาและการตั้งชื่อเส้นทางค้นหา

อย่างไรก็ตาม วิธีการแบ่งปริภูมิการค้นหาดั้งเดิมออกเป็นปริภูมิย่อย มิได้จำกัดแต่เพียงวิธีที่อธิบาย อาจแบ่งด้วยวิธีการอื่นที่เหมาะสมกับปัญหาได้ ภายหลังการแบ่งปริภูมิการค้นหา ขึ้นต่อไป PM สร้างคำตอบเริ่มต้นด้วยวิธีการสุ่มคำตอบภายในแต่ละปริภูมิย่อยจากนั้นจึงถอดถอนสภาพปริภูมิย่อยออก ดังนั้น ATS ย่อย แต่ละหน่วยจึงมีอิสระในการเคลื่อนตัวตามวิธีการค้นหาไปทั่วปริภูมิดั้งเดิม ATS ย่อยที่ถูกจัดตั้งในปริภูมิย่อยที่มีคำตอบวงกว้าง หรืออาจเป็นปริภูมิย่อยที่อยู่ใกล้ชิดกับคำตอบวงกว้างจะสามารถเข้าถึงคำตอบวงกว้างได้รวดเร็ว อย่างไรก็ตามการสร้างคำตอบเริ่มต้นมิได้จำกัดแต่เพียงการสุ่ม ผู้ใช้สามารถเลือกใช้วิธีศึกษาสำนึก (heuristic) กำหนดคำตอบเริ่มต้นให้เหมาะสมกับปัญหาได้

ภาพสองมิติและสามมิติของการแบ่งปริภูมิการค้นหาคือส่วนหนึ่งของฟังก์ชันชีเคิลฟอกซ์โฮล (Shekel's foxholes function : SF) ดังแสดงในรูปที่ 4.4



รูปที่ 4.4 การแบ่งปริภูมิการค้นหของฟังก์ชันซีเกลฟอกซ์โฮลออกเป็นสี่ส่วน

รูปที่ 4.4 เป็นตัวอย่างการทำงานตามกลไก PM ปริภูมิ SF มีค่าสูงสุดและต่ำสุดของทั้ง x และ y เป็น ± 40 ดังรูปที่ 4.4 (ก) ที่อาจเขียนในรูปพิกัดของปริภูมิการค้นหาว่า [ขอบเขตบนตัวแปรที่ 1 ขอบเขตบนตัวแปรที่ 2 ขอบเขตล่างตัวแปรที่ 1 ขอบเขตล่างตัวแปรที่ 2] นั่นคือ $[40 \ 40; -40 \ -40]$ สมมุติว่า ต้องการแบ่งปริภูมิ SF ออกเป็นสี่ปริภูมิการค้นหาย่อยจึงทำการแบ่งด้วยเส้นแนวนอนและเส้นแนวตั้งอย่างละหนึ่งเส้น อยู่กึ่งกลางของทั้งสองแกน เส้นแบ่งจึงตัดกันเป็นมุมฉากที่จุด $(0,0)$ จึงเกิดเป็นสี่ปริภูมิการค้นหาย่อยที่มีพื้นที่เท่า ๆ กัน ปรากฏบนภาพฉายสองมิติดังรูปที่ 4.4 (ก) และกำหนดชื่อเป็นปริภูมิการค้นหาย่อยที่ 1 (#1) ปริภูมิการค้นหาย่อยที่ 2 (#2) ปริภูมิการค้นหาย่อยที่ 3 (#3) และ ปริภูมิการค้นหาย่อยที่ 4 (#4) ที่มีขอบเขตเป็น $[40 \ 40; 0 \ 0]$ $[0 \ 40; -40 \ 0]$ $[0 \ 0; -40 \ -40]$ และ $[40 \ 0; 0 \ -40]$ สำหรับปริภูมิย่อย #1 #2 #3 และ #4 ตามลำดับ และอาจสังเกตเห็นว่าคำตอบวงกว้างปรากฏอยู่ใน #3 กระบวนการสุ่มเลือกคำตอบเริ่มต้นของ PM กรณีของ #3 มีโอกาสมากที่คำตอบเริ่มต้นนั้นจะเป็นคำตอบที่ใกล้เคียงกับคำตอบวงกว้างหรืออาจเป็นคำตอบวงกว้างเลยก็เป็นไปได้ นั่นหมายความว่ายิ่งเรามีจำนวนหน่วยค้นหามากขึ้น ปริภูมิการค้นหา ก็จะยิ่งถูกแบ่งมากขึ้นตามไปด้วย ทำให้คำตอบวงกว้างมีโอกาสจะถูกคัดเลือกให้เป็นคำตอบเริ่มต้นของปริภูมิการค้นหาย่อยหนึ่งนั้นมากขึ้นตามไปด้วย เท่ากับเป็นผลดีต่อ MATS นั่นเอง ถ้าพิจารณาการแบ่งปริภูมิตามรูปที่ 4.3 โดยสมมุติให้เป็นปริภูมิของ SF ดังรูปที่ 4.4 กรณีที่จำนวนหน่วยค้นหาเป็น 2 พิจารณาที่รูป 4.3 (ก) พบว่าคำตอบวงกว้างอยู่ใน #2 ถ้าจำนวนหน่วยค้นหาเพิ่มเป็น 4 พิจารณาที่รูป 4.3 (ข) พบว่าคำตอบวงกว้างอยู่ใน #3 เช่นเดิมถ้าจำนวนหน่วยค้นหาเพิ่มเป็น 8 พิจารณาที่รูป 4.3 (ค) พบว่าคำตอบวงกว้างอยู่ใน #5 และกรณีสุดท้ายเมื่อจำนวนหน่วยค้นหาเพิ่มเป็น 16 พิจารณาที่รูป 4.3 (ง) พบว่าคำตอบวงกว้างอยู่ใน #13 ถ้าเรียงตามขนาดปริภูมิย่อยที่บรรจุคำตอบวงกว้างไว้จากใหญ่ไปหาเล็กจะเรียงได้ดังนี้ #2 #3 #5 และ #13 ซึ่งของการแบ่งปริภูมิค้นหาแบบ 2 4 8 และ 16 หน่วยค้นหา ตามลำดับ แสดงให้เห็นว่ายิ่งจำนวนหน่วยค้นหามากขึ้น ขนาดของปริภูมิย่อยที่มีคำตอบวงกว้างอยู่ก็ยิ่งเล็กลง ทำให้มีโอกาสได้คำตอบเริ่มต้นเป็นคำตอบวงกว้างหรือค่าใกล้เคียงคำตอบวงกว้างมากยิ่งขึ้นนั่นเอง แต่การเพิ่มจำนวนหน่วยค้นหาให้มากขึ้นไปก็มีขอบเขตจำกัด ตัวอย่างปัญหาการหาเหมาะที่สุดแบบสองมิติที่มีลักษณะปริภูมิค้นหาเป็นรูปสี่เหลี่ยมดังรูปที่ 4.3 จำนวนหน่วยค้นหาสูงสุดที่จะเป็นไปได้สำหรับปัญหานี้จะเป็น ดังสมการที่ (4.1)

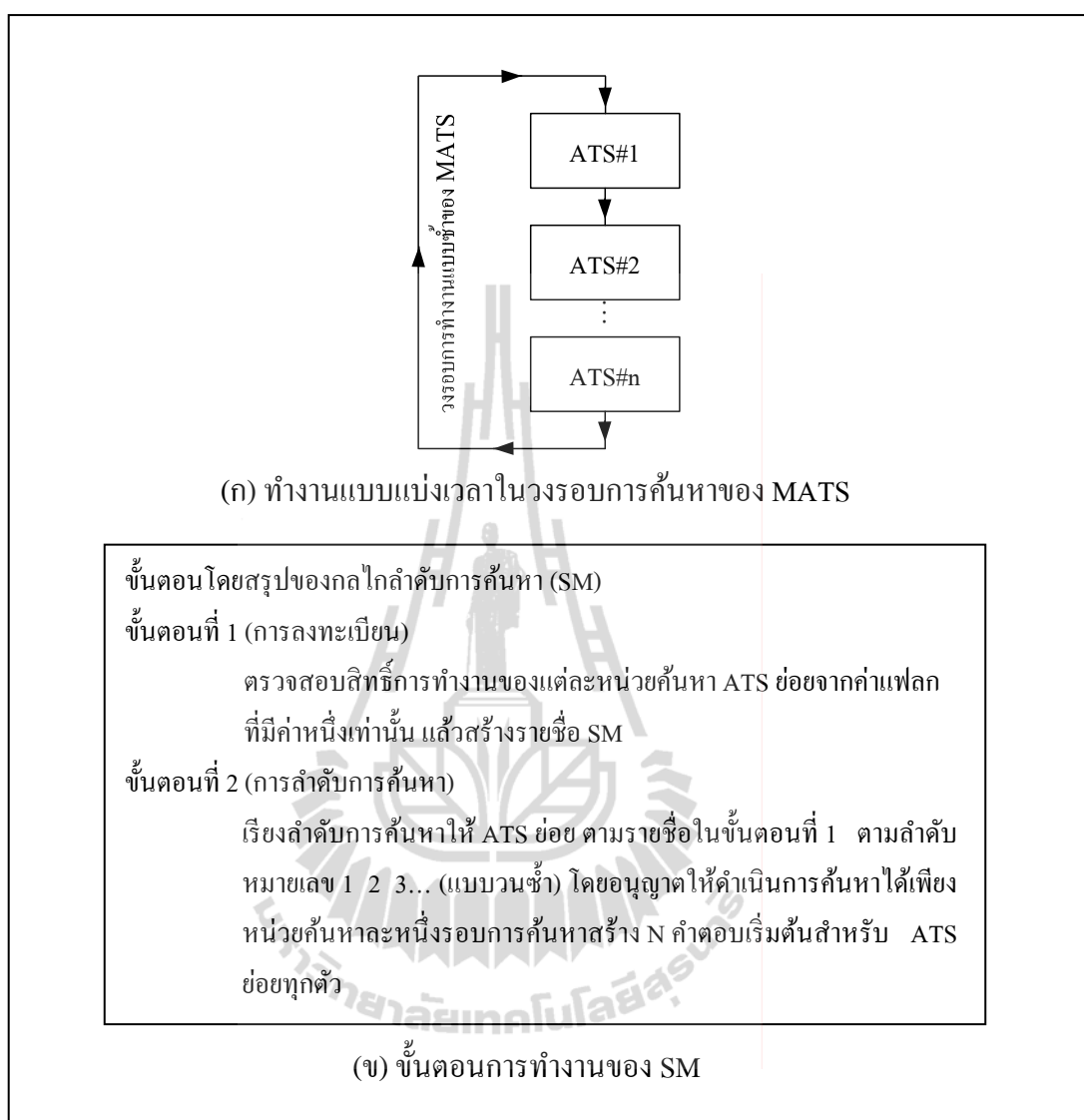
$$N_{\max} = \left(\frac{L}{2R} \right)^2 \quad (4.1)$$

โดยที่ $N_{\max}$ คือจำนวนหน่วยค้นหาสูงสุดที่แบ่งได้ L คือ ความยาวด้านหนึ่งของปริภูมิค้นหาที่เป็นสี่เหลี่ยมจัตุรัส มีหน่วยเป็น เมตร (m) และรัศมีการค้นหา R ของ ATS มีหน่วยเป็นเมตร (m) ผู้อ่าน

พึงระลึกไว้เสมอว่า ATS จะคัดเลือกค่าใกล้เคียงภายในวงกลมรัศมี R โดยมีค่าตอบปัจจุบันทำหน้าที่เป็นจุดศูนย์กลางวงกลม ตัวอย่าง ปฏิภูมิปัญหาหนึ่งเป็นสี่เหลี่ยมจัตุรัสขนาด 30 ซม. x 30 ซม. และ ATS รัศมีการค้นหา $R = 1.5$ ซม. ดังนั้น จะได้ $N_{\max}$ เท่ากับ 100 การเพิ่มจำนวนหน่วยค้นหาที่สูงกว่า $N_{\max}$ ย่อมไม่ก่อให้เกิดประโยชน์ต่อ MATS แต่อย่างใด เพราะ MATS ถือว่าพื้นที่สำรวจในรัศมีการค้นหาเป็นปฏิภูมิการค้นหาย่อยที่เล็กที่สุด หากนำปฏิภูมิการค้นหาย่อยเหล่านี้มาเรียงชิดติดกันจนเต็มปฏิภูมิการค้นหาจะใช้จำนวน $N_{\max}$ ปฏิภูมิการค้นหาย่อย จึงถือว่า ATS ย่อยมีจำนวนมากเพียงพอและกระจายทั่วถึงทั้งปฏิภูมิการค้นหาแล้วนั่นเอง

กลไกลำดับการค้นหา (SM)

SM ทำหน้าที่สร้างลำดับการทำงานของ ATS ย่อยแต่ละเส้นทาง ให้กับตัวประมวลผลดำเนินงาน ตามคำสั่งภายในของ ATS ย่อยนั้น ๆ รูปแบบการจัดสรรเวลาสำหรับการประมวลผลให้กับ ATS ย่อย นั้นมีได้หลายรูปแบบเช่น รูปแบบที่หนึ่งแบบสถิตย์ (static) มีการจัดสรรจำนวนรอบการค้นหาคงที่เท่ากันทุกหน่วยค้นหาตลอดการค้นหาของ MATS และ รูปแบบที่สองแบบพลวัต (dynamic) มีการจัดสรรจำนวนรอบการค้นหาไม่คงที่ขึ้นกับสถานการณ์การค้นหาของแต่ละ ATS ย่อย เป็นต้น สำหรับบทความนี้เลือกศึกษา SM ในรูปแบบที่หนึ่ง ด้วยการตั้งจำนวนรอบการค้นหาสำหรับ ATS ย่อย แบบหนึ่งรอบการค้นหา เปรียบเทียบความแตกต่างกับกรณีของ ATS แบบเดียวที่ไม่จำเป็นต้องใช้ SM เนื่องจากตัวประมวลผลสามารถดำเนินการสร้างลำดับคำตอบให้กับ ATS แบบเดียวได้อย่างต่อเนื่องปราศจากการถูกขัดจังหวะเพื่อให้ไปทำงานอย่างอื่นเนื่องจากมีเพียงหนึ่งเส้นทางค้นหา การค้นหาตามหลักการแบ่งเวลาของ SM แสดงดังรูปที่ 4.5 (ก)

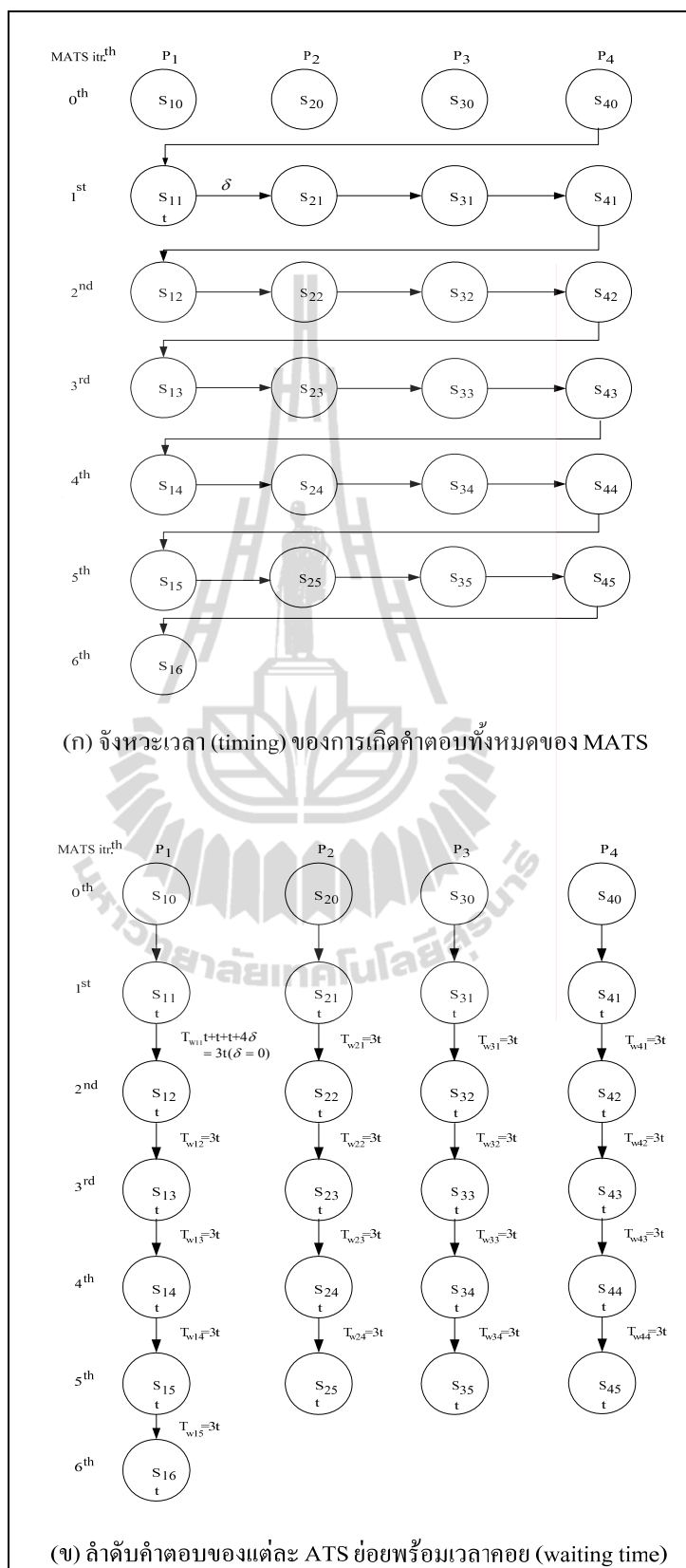


รูปที่ 4.5 ลักษณะการทำงานของ SM

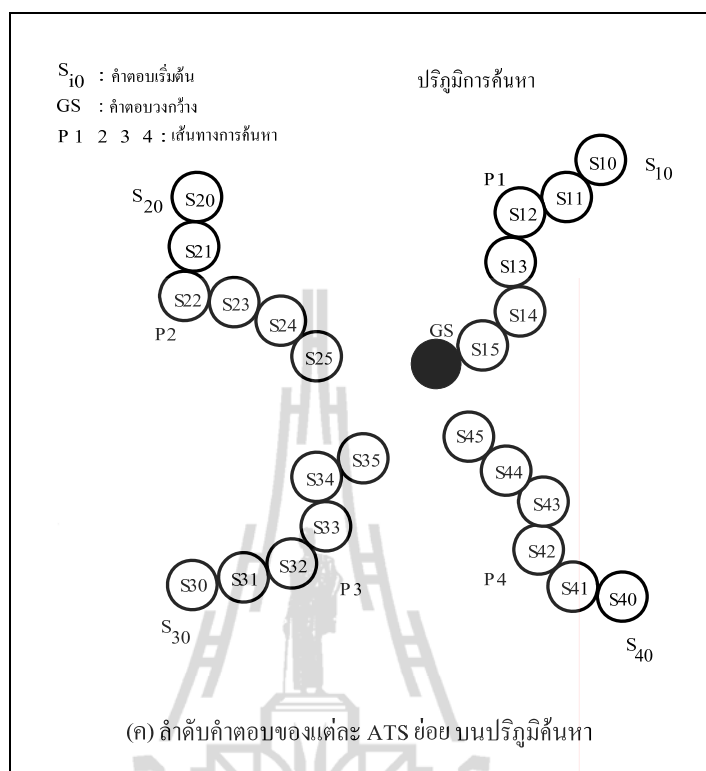
ลักษณะการทำงานของ ATS ย่อยใน MATS เมื่อ ATS#1 ดำเนินการค้นหาเสร็จสิ้นไปหนึ่งรอบการค้นหาแล้ว ก็จะหยุดเพื่อรอคิวการค้นหาในรอบถัดไปช่วงเวลาถัดไป SM สั่งให้ ATS#2 ดำเนินการจนเสร็จกระบวนการค้นหาหนึ่งรอบ และสั่งให้หยุดรอคิวรอบการค้นหาถัดไป ต่อจากนั้นจึงเป็นการทำงานของ ATS#3 ATS#4 จนถึง ATS#n ในลักษณะเช่นนี้ไปเรื่อย ๆ จากนั้นจึงเริ่มต้นที่ ATS#1 ใหม่ ตามด้วย ATS#2 ต่อไปจนถึง ATS #n ตามลำดับจนครบรอบการค้นหาที่ 2 ของ MATS รอบการค้นหาของ MATS จะวนซ้ำเช่นนี้จนกระทั่ง เมื่อมีการเรียกใช้กลไกยกเลิกการค้นหา หรือ DM จะทำให้จำนวนของหน่วยค้นหาในวงจรการค้นหาของ MATS ลดลงทำให้ SM ต้องตรวจสอบจำนวนหน่วยค้นหาที่เหลืออยู่ ต่อจากนั้นจึงเลื่อนลำดับช่วงเวลาการค้นหาของ ATS ย่อย

ที่อยู่ถัดไปขึ้นมาแทนที่หน่วยค้นหาซึ่งถูกยกเลิกการค้นหาจาก DM ไปแล้ว ขั้นตอนการทำงานของ SM สามารถสรุปได้ดังรูปที่ 4.5 (ข)

พิจารณารูปที่ 4.6 (ก) เส้นทางการค้นหาของ ATS#1 หรือ search path #1 (P_1) หมายถึงเส้นที่เชื่อมต่อพิกัดบนปริภูมิการค้นหาของคำตอบที่อยู่ติดกันทั้งหมดของ ATS#1 ใน P_1 จะประกอบด้วยคำตอบที่พบโดย ATS#1 เรียงกันตามลำดับเวลาตั้งแต่เริ่มต้นจนสิ้นสุดการค้นหา ตัวอย่างลำดับคำตอบ ตามรูปที่ 4.6 (ก) ของ P_1 ทำการสร้างลำดับคำตอบภายในวงรอบการค้นหาของ MATS อยู่ทั้งหมด 6 รอบการค้นหา ได้คำตอบเรียงอยู่คอลัมน์ที่ 1 แสดงได้ดังนี้ $\{S_{10}, S_{11}, S_{12}, S_{13}, S_{14}, S_{15}, S_{16}\}$ โดย S_{10} คือคำตอบเริ่มต้น ตามด้วย S_{11} เป็นคำตอบถัดไปซึ่งเกิดขึ้นในรอบการค้นหาที่ 1 ของ MATS แล้วก็เป็น S_{12} ไปเรื่อย ๆ จนกระทั่งได้คำตอบสุดท้ายของ P_1 คือ S_{16} ถ้าพิจารณาจาก รูปที่ 4.6 (ค) ประกอบด้วยก็จะพบว่า S_{16} ก็คือคำตอบวงกว้าง (global solution หรือ GS) ของ MATS ที่ถูกพบโดย ATS#1 ใน P_1 นั่นเอง ลำดับคำตอบของ P_2 พิจารณาคอลัมน์ที่สองเรียงจากบนลงล่าง จะได้ลำดับคำตอบ P_2 คือ $\{S_{20}, S_{21}, S_{22}, S_{23}, S_{24}, S_{25}\}$ มี S_{20} คือคำตอบเริ่มต้นแล้วตามด้วย S_{21} ไปเรื่อย ๆ จนได้ S_{25} เป็นคำตอบสุดท้ายของ P_2 ถ้าพิจารณากลไก SM จากรูปที่ 4.5 (ก) ช่วงเริ่มต้นการค้นหาของ MATS จะมี ATS ย่อย อยู่ครบทั้งสี่ตัว ($n=4$ สำหรับตัวอย่างนี้) เรียงลำดับก่อนหลังจากบนลงล่างโดยเริ่มต้นที่ ATS#1 ตามด้วย ATS#2 ATS#3 และ ATS#4 ตามลำดับ โดยหลังจากที่ PM จัดสรรคำตอบเริ่มต้นให้กับ ATS ย่อย จนครบทุกตัว เรียงลำดับจากซ้ายไปขวาดังแต่ P_1 จนถึง P_4 ทำให้แต่ละเส้นทางค้นหาได้คำตอบต้นทางเรียบร้อยแล้ว ดังรูปที่ 4.6 (ก) แถวที่ 1 หรือวงรอบการค้นหาที่ศูนย์ของ MATS นั่นเอง



รูปที่ 4.6 ตัวอย่างการทำงานของ SM ใน MATS แบบ 4 เส้นทางค้นหาที่ยังไม่มี DM



รูปที่ 4.6 ตัวอย่างการทำงานของ SM ใน MATS แบบ 4 เส้นทางค้นหาที่ยังไม่มี DM (ต่อ)

เมื่อขึ้นวงรอบการค้นหาที่ 1 ของ MATS SM จะอนุญาตให้ ATS#1 ดำเนินการค้นหาจนได้คำตอบมาหนึ่งค่าก็จะสั่งให้ ATS#1 หยุดรอ แล้วจึงค่อยอนุญาตให้ ATS#2 ทำการค้นหาคำตอบของตัวเองอีกหนึ่งรอบ จากนั้นสั่งให้ ATS#2 หยุดรออีก แล้วจึงค่อยอนุญาตให้ ATS#3 ทำการค้นหาคำตอบของตัวเองอีกหนึ่งรอบ จากนั้นสั่งให้ ATS#3 หยุดรออีก แล้วจึงค่อยอนุญาตให้ ATS#4 ทำการค้นหาคำตอบของตัวเองอีกหนึ่งรอบ จากนั้นสั่งให้ ATS#2 หยุดรออีกเป็นอันสิ้นสุดวงรอบการค้นหาที่ 1 ของ MATS เมื่อ MATS ขึ้นวงรอบการค้นหาที่ 2 SM จะวนกลับไปอนุญาตให้ ATS#1 ทำการค้นหาคำตอบถัดไปต่ออีกหนึ่งรอบจนได้ S_{12} แล้วตามด้วย ATS#2 ATS#3 และ ATS#4 ตามลำดับ ก่อนที่วนกลับไปเริ่มต้น ATS#1 เพื่อขึ้นรอบการค้นหาของ MATS ใหม่ เป็นเช่นนี้ ไปจนถึงรอบการค้นหาที่ 6 ATS #1 จึงพบกับคำตอบวงกว้างหรือค่าใกล้เคียงคำตอบวงกว้างซึ่งสอดคล้องกับหนึ่งในเงื่อนไขยุติการค้นหา ถ้าพิจารณากับปฏิภูมิการค้นหาร่วมกับลำดับคำตอบที่แต่ละ ATS ย่อย สร้างขึ้นดังรูปที่ 4.6 (ค) จะเห็นได้ชัดเจนว่า ATS#1 ถึงคำตอบวงกว้าง หรือ GS ก่อน ATS ย่อย ตัวอื่น ๆ และเป็นตัวทำให้ MATS ยุติการค้นหาในรอบการค้นหาที่ 6 นั่นเอง การดำเนินการค้นหาในลักษณะแบ่งเวลาการทำงานนี้จึงเป็นวิธีการพื้นฐานสำหรับระบบที่มีตัวประมวลผลเพียงตัวเดียว

การศึกษาผลของ SM ต่อเวลาการค้นหาของ MATS สามารถพิจารณาจากรูปที่ 4.6 (ก) โดยกำหนดค่าพารามิเตอร์ต่าง ๆ ไว้ดังต่อไปนี้

t (averaged searching time per round) คือเวลาเฉลี่ยที่ ATS ย่อย ใช้ในการค้นหาจนกระทั่งพบคำตอบแต่ละรอบการค้นหา การศึกษานี้ทำการประมาณให้ค่า t เท่ากันทุกครั้งที่ ATS ย่อย ทุก ๆ เส้นทางดำเนินการค้นหา

δ (switching time) คือเวลาที่ตัวประมวลผลใช้ในการโอนย้ายจากการดำเนินงานกับ ATS ย่อย เส้นทางหนึ่ง ไปดำเนินงานกับ ATS ย่อย อีกเส้นทางหนึ่ง การศึกษานี้ประมาณให้ทุกครั้งที่มีการเปลี่ยนหน่วยค้นหา ไม่ว่าจะเป็นจาก ATS#1 ไป ATS#2 หรือจาก ATS#2 ไป ATS#1 หรือจาก ATS#2 ไป ATS#3 หรือ ATS#3 ไป ATS#4 เป็นต้น ก็ตาม ให้ใช้เวลาเท่ากันหมดนั่นคือ δ และยังประมาณให้ค่า δ มีค่าน้อยมากหรือ $\delta \approx 0$ นั่นเอง

T_w (waiting time) คือเวลาคอย หรือ เวลาขัดจังหวะ (interrupt) ของตัวประมวลผลที่ถูกใช้ไปทำงานอื่นก่อนกลับมาค้นหาคำตอบถัดไปให้กับ ATS ย่อยอีก จึงถูกมองว่าเป็นเวลาที่ ATS ย่อย แต่ละตัวใช้รอคอยตัวประมวลผลนั่นเอง ตัวอย่างเวลาคอย สามารถพิจารณาได้ดังรูปที่ 4.6 (ข) การสร้างลำดับคำตอบ P_i ช่วงจาก S_{i1} ไป S_{i2} P_i จำเป็นใช้เวลาคอยตัวประมวลผลที่ไปทำการค้นหาคำตอบ S_{21} S_{31} และ S_{41} ก่อนกลับมาค้นหา S_{i2} คิดเป็นเวลาคอย T_{w11} วินาที

T_s (searching time) คือเวลาการค้นหาทั้งหมดที่ใช้ระหว่างการค้นหาคำตอบวงกว้างหรือค่าใกล้เคียงคำตอบวงกว้างของ MATS ภายใต้การทำงานของ SM

ตัวอย่างความสัมพันธ์ของ T_s กับเวลาคอย T_w เวลาค้นหา t และ เวลาโอนย้าย δ สำหรับกรณีการสร้างลำดับคำตอบ P_i พิจารณารูปที่ 4.6 (ข) จะเป็นไปตามสมการที่ (4.2)

$$\begin{aligned} T_s &= t + T_{w11} + t + T_{w12} + t + T_{w13} + t + T_{w14} + t + T_{w15} + t + 20\delta \\ &= t + 3t + t + 3t + t + 3t + t + 3t + t + 3t + t + 0 \\ &= 21t \end{aligned} \quad (4.2)$$

T_s ของ MATS สำหรับการสร้างลำดับคำตอบของ P_i ตามสมการที่ (4.2) แบ่งได้เป็นสองส่วน ได้แก่ ส่วนของเวลาที่ตัวประมวลผลใช้ทำงานมีจำนวน 6 ครั้ง ๆ ละ t คิดเป็นเวลา $6t$ วินาที กับส่วนของเวลาที่ตัวประมวลผลต้องรอคอย จึงให้ชื่อว่าเวลาคอย (waiting time หรือ T_w) มีทั้งหมด 6 ช่วงเวลาได้แก่ T_{w11} จนถึง T_{w15} ผลรวมของทุกช่วงก็คือ T_w ดังแสดงในตามสมการที่ (4.3) ข้างล่างนี้

$$\begin{aligned}
 T_w &= T_{w11} + T_{w12} + T_{w13} + T_{w14} + T_{w15} \\
 &= 3t + 3t + 3t + 3t + 3t \\
 &= 15t
 \end{aligned}
 \tag{4.3}$$

เมื่อพิจารณาสัดส่วนของเวลาคอยตามสมการที่ (4.3) ต่อเวลาการค้นหาทั้งหมดตามสมการที่ (4.2) จะเป็นไปตามสมการที่ (4.4)

$$T_w / T_s = 15t/21t = 0.7143 \tag{4.4}$$

จากสมการที่ (4.4) พบว่าเวลาคอยที่ใช้ไปในการค้นหาคำตอบของ MATS คิดเป็น 71.43% ของเวลาการค้นหาทั้งหมด ซึ่งถือว่าค่อนข้างมาก แต่สิ่งที่ได้มาก็คือการสำรวจปริภูมิการค้นหาที่ได้หลายพื้นที่มากกว่าของ ATS แบบเดียว ส่งผลให้มีโอกาสพบคำตอบวงกว้างได้มากกว่า ส่วนปัญหาเรื่องของเวลาการค้นหาที่ค่อนข้างมากอันเนื่องจากเวลาคอย ถ้ามีกลไกบางอย่างช่วยแก้ปัญหานี้ ก็จะช่วยให้ประสิทธิภาพรวมของ MATS ดีขึ้น ทางเลือกหนึ่งสำหรับการลดปัญหาเวลาคอยใน MATS ก็คือ กลไกยกเลิกการค้นหา (discarding mechanism : DM) ซึ่งจะได้กล่าวถึงรายละเอียดในหัวข้อต่อไป

กลไกยกเลิกการค้นหา (DM)

DM ช่วยกำจัดบางหน่วยค้นหาที่ถูกประเมินสภาพว่ามีแนวโน้มจะพบคำตอบวงกว้างได้น้อยกว่าหน่วยค้นหาอื่น เช่น บางหน่วยค้นหาอาจเกิดการล็อกโดยคำตอบวงแคบเฉพาะถิ่น ซึ่งจะส่งผลให้การเข้าสู่หาผลเฉลยของเส้นทางอื่นอาจเข้าไปด้วยเป็นต้น ขั้นตอนการทำงานของ DM แสดงได้ในรูปที่ 4.7

ขั้นตอนโดยสรุปของกลไกยกเลิกการค้นหา (DM)

ขั้นตอนที่ 1 (การเริ่มต้น)

รับค่าฟังก์ชันวัตถุประสงค์ของทุกหน่วยค้นหา ATS ย่อยที่ยังอยู่ในวงรอบการค้นหาของ MATS

ขั้นตอนที่ 2 (การจัดอันดับ)

เรียงลำดับหน่วยค้นหาตามค่าฟังก์ชันวัตถุประสงค์จากน้อยไปมากเพื่อจัดกลุ่ม ATS ย่อย

ตามคุณภาพ (ปัญหาการหาค่าต่ำสุด)

ขั้นตอนที่ 3 (การยกเลิกการค้นหา)

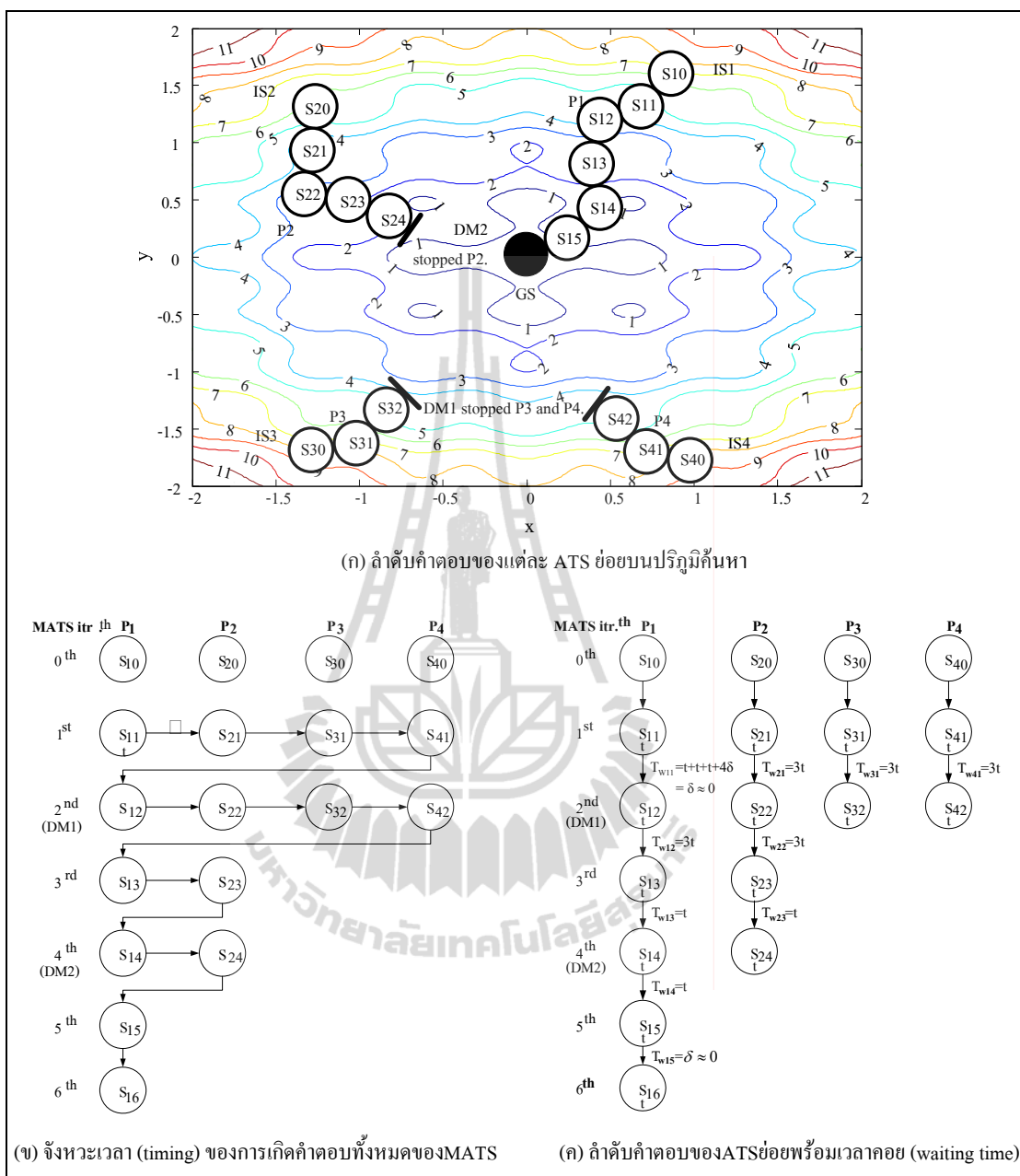
คัดหน่วยค้นหาที่ด้อยคุณภาพออกจากวงรอบการค้นหาด้วยการกำหนดค่าแฟลกเป็นศูนย์ แล้วกลับ

สู่วงรอบการค้นหาของ MATS

รูปที่ 4.7 ขั้นตอนการทำงานของ DM

MATS เรียกใช้งาน DM ตามเกณฑ์จำนวนรอบการค้นหาที่กำหนดไว้ล่วงหน้า และสามารถเรียกใช้ DM ได้มากกว่าหนึ่งครั้ง การตัดทอนจำนวนหน่วยค้นหาให้น้อยลงเป็นความอ่อนตัวที่ผู้ใช้ปรับแต่งได้ อาจไม่จำเป็นต้องลดลงครั้งหนึ่งของจำนวนเดิมดังที่นำเสนอในที่นี้ เมื่อมีการเรียกใช้ DM ขั้นตอนที่ 0 DM จะนำค่าฟังก์ชันวัตถุประสงค์ของ ATS ย่อยแต่ละตัวที่ยังมีสิทธิ์ทำงานอยู่ในวงรอบการค้นหาของ MATS มาเก็บไว้ โดยสิทธิ์ดังกล่าวถูกกำหนดด้วยค่าสถานะของ flag นั่นคือ ถ้าสถานะของ flag = 1 หมายถึง ATS ย่อย นั้นมีสิทธิ์ทำงานใน MATS ในทางตรงกันข้าม ถ้าสถานะของ flag = 0 หมายถึง ATS ย่อย นั้นหมดสิทธิ์ทำงานใน MATS นั่นเอง ขั้นตอนที่ 1 DM จะทำการเรียงลำดับจากค่าฟังก์ชันวัตถุประสงค์น้อยไปหาค่าฟังก์ชันวัตถุประสงค์มาก (สำหรับปัญหาการหาค่าต่ำสุด) และขั้นตอนที่ 2 ยกเลิกสิทธิ์การทำงานในวงรอบการค้นหาของ MATS ให้กับหน่วยค้นหาที่มีค่าฟังก์ชันวัตถุประสงค์มากที่สุดและมารองลงมาตามลำดับเป็นจำนวนหนึ่ง ซึ่งอาจเลือกการลดจำนวนหน่วยค้นหาลงทีละกึ่งหนึ่งของจำนวนหน่วยค้นหาที่มีอยู่ขณะนี้ก็ได้ ส่วนหน่วยค้นหาที่เหลือยังคงมีสิทธิ์ดำเนินการค้นหาได้ต่อไปในวงรอบการค้นหาของ MATS การเรียกใช้ DM ครั้งที่สอง หรือ DM2 สามารถทำได้หากจำนวนหน่วยค้นหาที่เหลืออยู่มากกว่าหนึ่งเส้นทาง โดยต้องกำหนดค่ารอบการค้นหาที่จะเรียกใช้ DM2 ให้มากกว่าของรอบการเรียกใช้ DM1 นั่นเอง ส่วนจะมากกว่าเท่าไรก็ขึ้นอยู่กับความเหมาะสมของแต่ละปัญหา

DM มี 2 ค่าพารามิเตอร์อิสระให้ผู้ใช้สามารถปรับตั้งได้ ได้แก่ ค่าเมตริกซ์การลดจำนวนหน่วยค้นหาที่เบื้องต้นเลือกรูปแบบการลดจำนวนครั้งละกึ่งหนึ่งของจำนวนก่อนการเรียกใช้ DM



รูปที่ 4.8 ตัวอย่างการทำงานของ SM ใน MATS แบบ 4 เส้นทางค้นหาพร้อม DM

เป็นค่าปกติไว้ก่อน แต่ไม่ได้จำกัดอยู่เพียงรูปแบบนี้รูปแบบเดียวเท่านั้น ผู้ใช้สามารถเลือกรูปแบบการลดจำนวนหน่วยค้นหาได้ตามความเหมาะสมของปัญหา ตัวอย่างต่อไปนี้ เป็นรูปแบบการลดจำนวนหน่วยค้นหาหลังที่ละกิ่งหนึ่งเมื่อเรียกใช้ DM สำหรับการค้นหาคำตอบวงกว้างของฟังก์ชัน BF กรณี MATS สองเส้นทางค้นหา หรือ MATS#2 จะมีการเรียกใช้ DM หนึ่งครั้งในรอบการค้นหาที่ 5 โดยทำการลดจำนวนหน่วยค้นหาจาก 2 ให้เหลือ 1 หน่วยค้นหาที่จะทำการค้นหาต่อไป กรณี

MATS เส้นทางการเดินทาง DM จะถูกเรียกใช้สองครั้งในรอบการค้นหาที่ 5 และ 10 โดยทำการลดจำนวนหน่วยค้นหาจาก 4 เหลือ 2 ในครั้งที่หนึ่ง และเหลือ 1 หน่วยค้นหาในครั้งสุดท้าย กรณี MATS แปลเส้นทาง DM อาจถูกเรียกใช้ทั้งหมดสามครั้ง ได้แก่ รอบการค้นหาที่ 5 จะทำการลดจำนวนหน่วยค้นหาจาก 8 เหลือ 4 ครั้งที่สองในรอบการค้นหาที่ 10 จะทำการลดจำนวนหน่วยค้นหาจาก 4 เหลือ 2 และ ครั้งสุดท้ายคือครั้งที่สามในรอบการค้นหาที่ 15 จะทำการลดจำนวนหน่วยค้นหาจาก 2 เหลือ 1 ตามลำดับ

เพื่อให้เข้าใจการทำงานของ DM ได้ดียิ่งขึ้น ต่อจากนี้ให้พิจารณารูปที่ 4.8 โดยอาศัยสัญลักษณ์และข้อกำหนดที่คล้ายกันกับรูปที่ 4.6 ที่ได้กล่าวถึงไปแล้วก่อนหน้านี้ รูปที่ 4.8 (ก) แสดงเส้นทางการค้นหาทั้งสี่เส้นทางของ MATS บนพื้นผิวของฟังก์ชัน BF ที่มีเส้นระดับ (contour) พร้อมตัวเลขกำกับเพื่อความชัดเจนยิ่งขึ้น เส้นทางค้นหาดังกล่าวเกิดขึ้นจากการวางตัวของคำตอบที่พบในแต่ละรอบการค้นหาของ ATS ย่อย ช่วงเริ่มต้นการค้นหาหรือรอบการค้นหาที่ศูนย์ของ MATS PM จัดสรรคำตอบเริ่มต้นสำหรับ ATS#1 คือ S_{10} มีค่าฟังก์ชันวัตถุประสงค์ประมาณ 7 ของ ATS#2 คือ S_{20} มีค่าประมาณ 8 ของ ATS#3 คือ S_{30} มีค่าประมาณ 8 และของ ATS#4 คือ S_{40} มีค่าประมาณ 8.5 ตามลำดับ พอขึ้นรอบการค้นหาที่ 1 MATS เริ่มต้นการค้นหาด้วยเส้นทาง P_1 จนได้ S_{11} ซึ่งค่าฟังก์ชันวัตถุประสงค์ของ S_{11} หรือ $f(S_{11}) \approx 4.5$ ถัดไป MATS จะทำการค้นหาคำตอบในเส้นทาง P_2 จนได้ S_{21} ซึ่งมีค่า $f(S_{21}) \approx 4$ ถัดจากนี้เป็นการค้นหาในเส้นทาง P_3 ทำให้ได้ S_{31} ด้วยค่า $f(S_{31}) \approx 7$ และครั้งสุดท้ายของรอบการค้นหาที่ 1 เป็นของเส้นทาง P_4 ทำให้ได้ S_{41} ด้วยค่า $f(S_{41}) \approx 7$ ภาพลำดับการสร้างคำตอบเหล่านี้ได้แสดงรายละเอียดไว้แล้วในรูปที่ 4.8 (ข) เมื่อขึ้นรอบการค้นหาที่ 2 MATS ก็ดำเนินการสร้างลำดับคำตอบต่อไปตามปกติจนครบทุกหน่วยค้นหาช่วงท้ายรอบการค้นหาที่ 2 ซึ่ง MATS ได้กำหนดให้มีการเรียกใช้ DM ครั้งแรก หรือ DM1 ตามขั้นตอนของ DM ที่ได้อธิบายไว้ดังรูปที่ 4.6 (ข) ได้ผลการเรียงลำดับค่าฟังก์ชันวัตถุประสงค์จากน้อยไปมากของทุกหน่วยค้นหาดังสมการที่ (4.5)

$$f(S_{22}) \approx 3 < f(S_{12}) \approx 4 < f(S_{42}) \approx 4.5 < f(S_{32}) \approx 5 \quad (4.5)$$

จากสมการที่ (4.5) คุณภาพคำตอบที่ได้พิจารณาจากค่าฟังก์ชันวัตถุประสงค์พบว่า ATS#2 ให้คำตอบคุณภาพดีที่สุด คือ 3 รองลงมาเป็นของ ATS#1 ได้ค่าประมาณ 4 ถัดลงมาเป็นของ ATS#4 ให้ค่าอยู่ที่ 4.5 และคุณภาพแย่สุดเป็นของ ATS#3 ให้ค่าประมาณ 5 และตามรูปแบบการลดจำนวนหน่วยค้นหาที่กำหนดไว้คือลดลงทีละครึ่งหนึ่งของจำนวนที่มีอยู่ ดังนั้นจากจำนวนสี่เส้นทางจะต้องมีเพียงสองเส้นทางเท่านั้นที่อยู่ทำงานต่อไป ส่วนอีกสองเส้นทางต้องถูกยกเลิกการค้นหา และสองหน่วยค้นหานั้นก็คือหน่วยค้นหาที่ 3 และ 4 สังเกตจากรูปที่ 4.8 (ค) รอบการค้นหาที่ 2 เป็นรอบการ

ค้นหาสุดท้ายของ P_3 และ P_4 ทำให้ P_3 ได้ลำดับคำตอบคือ $\{S_{30}, S_{31}, S_{32}\}$ และของ P_4 คือ $\{S_{40}, S_{41}, S_{42}\}$ เมื่อพิจารณารูปที่ 4.8 (ข) การเรียกใช้ DM1 ครั้งนี้ช่วยลดจำนวนหน่วยค้นหาไปสองหน่วย ค้นหาในรอบการค้นหาที่ 2 เมื่อขึ้นรอบการค้นหาที่ 3 จึงเหลือสองหน่วยค้นหาที่สลับกันทำงาน ด้วยตัวประมวลผลทำให้เวลาทยอยลดลงตามไปด้วยนั่นเอง ขณะนี้จำนวนหน่วยค้นหาที่เหลืออยู่สองเส้นทาง หมายความว่าเราสามารถเรียกใช้ DM ได้อีกหนึ่งครั้งนั่นเองเพื่อลดให้เหลือเพียงหน่วยค้นหาเดียว ดังนั้น DM2 จึงถูกกำหนดไว้ที่รอบการค้นหาที่ 4 สอดคล้องกับข้อเสนอแนะการตั้งค่า DM ที่ได้กล่าวไว้ในตอนต้นว่า การเรียกใช้ DM สามารถทำได้มากกว่าหนึ่งครั้งหากจำนวนหน่วยค้นหา ยังคงเหลืออยู่มากกว่าหนึ่งเส้นทาง ดังนั้นเมื่อ MATS ดำเนินการค้นหาคำตอบ S_{14} และ S_{24} ครบแล้ว ตอนท้ายรอบการค้นหาที่ 4 DM2 จะถูกเรียกใช้ โดยเริ่มต้นด้วยการประเมินคุณภาพคำตอบแล้วทำการเรียงลำดับจากน้อยไปมากเหมือนเช่น DM1 ถ้าพิจารณารูปที่ 4.8 (ก) เมื่ออ่านค่าฟังก์ชันวัตถุประสงค์จะได้ผลตามสมการที่ (4.6)

$$f(S_{14}) \approx 1.5 < f(S_{24}) \approx 2 \quad (4.6)$$

จากสมการที่ (4.6) เห็นได้อย่างชัดเจนว่าค่าฟังก์ชันวัตถุประสงค์ของ ATS#2 สูงกว่าของ ATS#1 และถ้าพิจารณาตามเกณฑ์ของ DM ที่ตั้งไว้จึงต้องยกเลิกการค้นหาของ ATS#2 แล้วปล่อยให้ ATS#1 ทำการค้นหาต่อไปจนกระทั่งสิ้นสุดรอบการค้นหาที่ 6 เสร็จสิ้นไขว่คว้าการค้นหาถูกพบเพราะได้คำตอบที่ให้ค่าฟังก์ชันวัตถุประสงค์ใกล้เคียงคำตอบกว้าง ตามที่ตัวอย่างนี้เลือกใช้พื้นที่ผิวฟังก์ชัน BF ซึ่งทราบแน่ชัดแล้วว่ามีคำตอบกว้างอยู่ที่พิกัด (0,0) ด้วยค่าฟังก์ชันเท่ากับศูนย์ MATS จึงหยุดการค้นหาด้วยเหตุผลดังกล่าว การอธิบายข้างต้นเป็นการอธิบายร่วมกับการยกตัวอย่างเชิงตัวเลขพร้อมภาพประกอบจากพื้นที่ผิวฟังก์ชันจริงที่ใช้ทดสอบมุ่งเน้นให้ผู้อ่านเข้าใจการทำงานของ DM และเห็นภาพกระบวนการประเมินคุณภาพหน่วยค้นหาด้วยเกณฑ์คุณภาพคำตอบขณะเรียกใช้ DM ให้ชัดเจนนั่นเอง

การศึกษารูปของ DM ต่อเวลาการค้นหาของ MATS พิจารณาจากรูปที่ 4.8 (ข) และรูปที่ 4.8 (ค) สามารถสร้างความสัมพันธ์ทางพีชคณิตได้ในทำนองเดียวกับสมการที่ (4.2) โดยอาศัยความสัมพันธ์ของ T_s กับเวลาคอย T_w เวลาค้นหา t และ เวลาโอนย้าย δ สำหรับกรณีการสร้างลำดับคำตอบ P_1 จึงแสดงได้ดังสมการที่ (4.7)

$$\begin{aligned} T_s &= t + T_{w11} + t + T_{w12} + t + T_{w13} + t + T_{w14} + t + T_{w15} + t + 12\delta \\ &= t + 3t + t + 3t + t + t + t + t + \delta + t + 0 \\ &= 14t \end{aligned} \quad (4.7)$$

เวลาคอย T_w ของ MATS หาได้ตามสมการที่ (4.8) ข้างล่างนี้

$$\begin{aligned} T_w &= T_{w11} + T_{w12} + T_{w13} + T_{w14} + T_{w15} \\ &= 3t + 3t + t + t + \delta \\ &= 8t \end{aligned} \quad (4.8)$$

เมื่อพิจารณาสัดส่วนของเวลาคอยตามสมการที่ (4.8) ต่อเวลาการค้นหาทั้งหมดตามสมการที่ (4.7) จะเป็นไปตามสมการที่ (4.9)

$$T_w / T_s = 8t/14t = 0.5714 \quad (4.9)$$

จากสมการที่ (4.9) พบว่าเวลาคอยที่ใช้ไปในการค้นหาคำตอบของ MATS คิดเป็น 57.14% ของเวลาการค้นหาทั้งหมด ซึ่งลดลงจากกรณีไม่มี DM ดังสมการที่ (4.4) ถึง 14.29% ถ้าพิจารณาที่เวลาการค้นหาทั้งหมดของ MATS สำหรับกรณีไม่มี DM แสดงดังสมการที่ (4.2) ใช้เวลาไป $T_{s_no_DM} = 21t$ ภายหลังที่เพิ่ม DM ให้กับ MATS ใช้เวลาการค้นหาลดลง แสดงดังสมการที่ (4.7) ด้วยค่า $T_{s_DM} = 14t$ พบว่า DM ช่วยลดเวลาการค้นหาลงไป 7t หรือคิดเป็น 33.33% ของเวลาการค้นหาทั้งหมดของ MATS ที่ลดลงไป จากตัวอย่างที่ได้อธิบายไว้ข้างต้นนี้คงพอทำให้ผู้อ่านเห็นบทบาทและความสำคัญของ DM ต่อการพัฒนา MATS ให้มีประสิทธิภาพการค้นหาที่ดีขึ้นได้มากขึ้น

ถึงตอนนี้เราสามารถประมวลภาพรวมของ MATS กับทั้งสามกลไกได้ดังนี้ PM ช่วยกระจายคำตอบเริ่มต้นสำหรับหน่วยค้นหาให้ทั่วถึงทั้งปริภูมิการค้นหา ทั้งนี้เพื่อลดโอกาสเกิดปรากฏการณ์ “ตกสำรวจ” ในบางพื้นที่แล้วจะส่งผลให้ไม่พบคำตอบวงกว้างที่ต้องการหรือต้องใช้เวลาการค้นหายาวนาน เมื่อเทียบกับกลยุทธ์ของ TS แล้ว PM จัดอยู่ในกลยุทธ์ความหลากหลายนั่นเอง ส่วน SM ช่วยทำให้ MATS เป็นอัลกอริธึมการค้นหาแบบอนุกรมแต่คิดในเชิงขนานและสามารถทำงานได้ทั้งบนตัวประมวลผลเดี่ยวและหลายตัวประมวลผล แม้ว่าจะทำให้เกิดเวลาคอยส่งผลให้เวลาการค้นหามากขึ้นบ้างแต่ก็ยังมีส่วนกลไกช่วยชดเชยได้ กลไกดังกล่าวคือ DM ที่จะคอยคัดเลือกให้เหลือเพียงหน่วยค้นหาที่จำเป็นในวงรอบการค้นหาของ MATS เมื่อเทียบกับกลยุทธ์ของ TS แล้ว DM จัดอยู่ในกลยุทธ์ความเข้มข้นโดยยังคำนึงถึงคุณสมบัติความหลากหลายไม่ให้อยู่เสียไปมากนัก แต่เมื่อลองเปรียบเทียบกับจินตિકอัลกอริธึม genetic algorithms (GAs) แล้ว DM จะมีความคล้ายคลึงกับกระบวนการคัดเลือก (selection) ของ GAs ที่จะคัดเลือกผู้สืบสายพันธุ์ที่มีความทนทานและความแข็งแรงเท่านั้นเพื่อความอยู่รอดในธรรมชาติได้นั่นเอง

อัลกอริธึม MATS ที่ได้พัฒนาขึ้นนี้ ได้รับการตีพิมพ์เผยแพร่ใน Kluabwang et al., 2008 หัวข้อต่อไปเป็นการวิเคราะห์อัลกอริธึมซึ่งจะพิจารณา MATS อย่างเป็นแบบแผนในมุมมองต่าง ๆ ให้เห็นข้อเด่นและข้อด้อยของ MATS อย่างชัดเจน

4.3 การวิเคราะห์อัลกอริธึม

หัวข้อนี้นำเสนอการวิเคราะห์อัลกอริธึม MATS ไว้สองแนวทางได้แก่ การวิเคราะห์ความซับซ้อนและการวิเคราะห์ความถูกต้อง โดยจะเปรียบเทียบกับอัลกอริธึม ATS แบบเดียว การวิเคราะห์ความซับซ้อนเชิงเวลากรณีเฉลี่ย (average-case time complexity analysis) ถูกเลือกสำหรับวิเคราะห์ ATS กับ MATS แล้วนำมาเปรียบเทียบกัน โดยเวลาการค้นหาค่าจะถูกพิจารณาผ่านจำนวนรอบการค้นหาเฉลี่ยสำหรับ ATS แบบเดียว และจำนวนรอบเฉลี่ยที่ใช้เพื่อค้นหาของ ATS ย่อยที่พบคำตอบวงกว้างสำหรับ MATS การวิเคราะห์ห้วงหลักการทางสถิติสำหรับการทดสอบทั่วไปที่ว่าจำนวนครั้งการทดสอบ (trial) ที่มากพอรวมกับการสุ่มกลุ่มตัวอย่าง (sample) ที่ทั่วถึงกลุ่มประชากร (population) ผลการวิเคราะห์นั้นจะมีความน่าเชื่อถือ การนำเสนอแบ่งออกเป็นสองหัวข้อย่อย ได้แก่ หัวข้อที่หนึ่งการวิเคราะห์ความซับซ้อนและความถูกต้องของ ATS แบบเดียว และ MATS ตามลำดับ ส่วนหัวข้อที่สองการวิเคราะห์ประสิทธิภาพ ซึ่งมีรายละเอียดดังต่อไปนี้

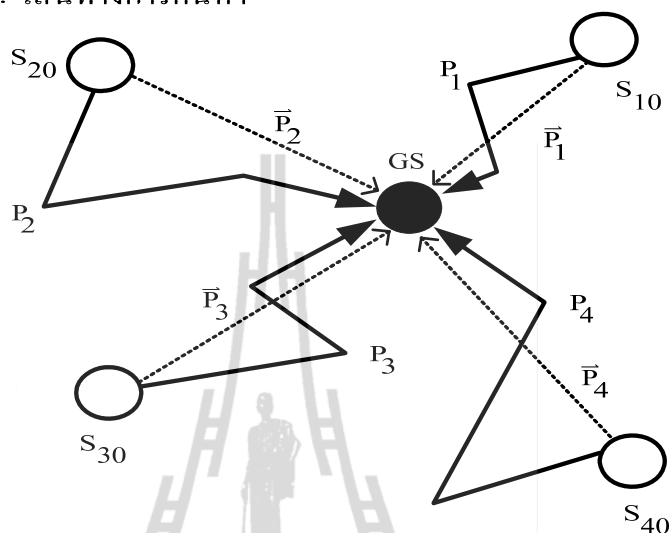
4.3.1 การวิเคราะห์ความซับซ้อนและความถูกต้อง

การนำเสนอเริ่มต้นด้วยตัวอย่างการค้นหา ATS แบบเดียว จำนวนสี่ครั้ง โดยสมมติให้เริ่มต้นด้วยคำตอบที่แตกต่างกัน ดังแสดงในรูปที่ 4.9 สำหรับปัญหาการหาค่าเหมาะที่สุดแบบสองมิติ เพื่อใช้วิเคราะห์เวลาการค้นหาคำตอบของ ATS แบบเดียว

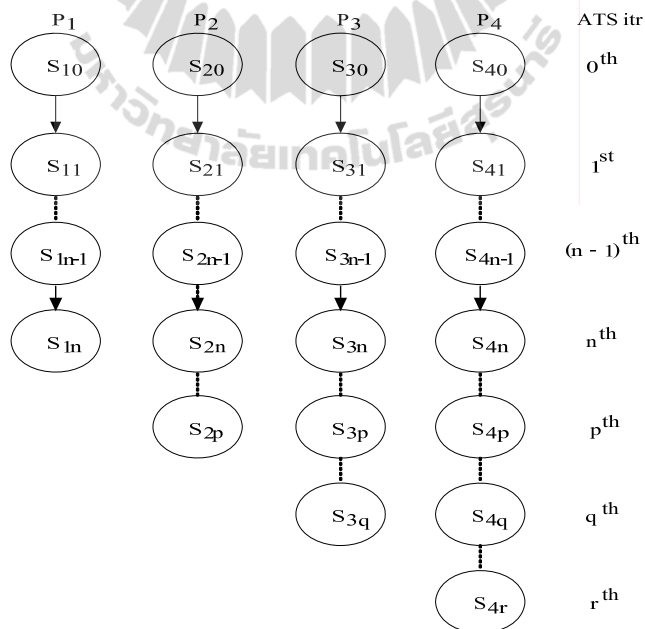
พิจารณารูปที่ 4.9 แสดงตัวอย่างการทดสอบของ ATS แบบเดียว จำนวน 4 ครั้ง (trials) ด้วยคำตอบเริ่มต้นที่แตกต่างกันกระจายทั่วทั้งปริภูมิการค้นหา แสดงดังรูปที่ 4.9 (ก) การทดสอบครั้งที่ 1 ATS แบบเดียว เริ่มการค้นหาด้วยคำตอบเริ่มต้น S_{10} หลังจากนั้น ATS แบบเดียว ก็ทำการค้นหาคำตอบถัดไปเรื่อย ๆ ตามเส้นทางการค้นหา P_1 จนพบคำตอบวงกว้าง P_1 ของการทดสอบ ATS แบบเดียว ครั้งที่ 1 เกิดจากการเรียงตัวของคำตอบที่ได้จากการค้นหาทั้งหมด n รอบการค้นหา ดังรูปที่ 4.9 (ข) การทดสอบ ATS แบบเดียว ครั้งที่ 2 เริ่มการค้นหาด้วยคำตอบเริ่มต้น S_{20} ซึ่งอยู่ทางด้านบนซ้ายของปริภูมิการค้นหา ตามรูปที่ 4.9 (ก) หรือ S_{20} ตามรูปที่ 4.9 (ข) หลังจากนั้น ATS แบบเดียว ก็เดินตามเส้นทางการค้นหา P_2 จนพบ

S_{i0} : คำตอบเริ่มต้น
 GS : คำตอบวงกว้าง
 $P_{1,...,4}$: เส้นทางการค้นหา

ปฏิกิริยาการค้นหา



(ก) เส้นทางค้นหาของ ATS แบบเดี่ยว ทั้งสี่เส้นทางบนปฏิกิริยาการค้นหา



(ข) ลำดับคำตอบของ ATS แบบเดี่ยว ทั้งสี่เส้นทางตั้งแต่เริ่มต้นจนสิ้นสุดการค้นหา

รูปที่ 4.9 การค้นหาของ ATS แบบเดี่ยว จำนวน 4 ครั้งด้วยคำตอบเริ่มต้นที่แตกต่างกัน

คำตอบวงกว้าง ลำดับคำตอบของ P_2 ใช้ไป p รอบการค้นหา ดังรูปที่ 4.9 (ข) ซึ่งมากกว่าการทดสอบครั้งที่ 1 มาถึงการทดสอบ ATS แบบเดี่ยว ครั้งที่ 3 ด้วยคำตอบเริ่มต้น S_{30} ซึ่งอยู่ทางด้านล่างซ้ายของปริภูมิการค้นหา ตามรูปที่ 4.9 (ก) หลังจากนั้น ATS แบบเดี่ยว ก็เดินตามเส้นทางการค้นหา P_3 จนพบคำตอบวงกว้าง ลำดับคำตอบของ P_3 ใช้ไป q รอบการค้นหา ดังรูปที่ 4.9 (ข) ซึ่งมากกว่าการทดสอบครั้งที่ 1 และ ครั้งที่ 2 แล้วก็มาถึงการทดสอบ ATS แบบเดี่ยว ครั้งที่ 4 เริ่มต้นด้วย S_{40} ซึ่งอยู่ทางด้านล่างขวาของปริภูมิการค้นหา ตามรูปที่ 4.9 (ก) หลังจากนั้น ATS แบบเดี่ยว ก็ดำเนินการค้นหาตามเส้นทาง P_4 จนพบคำตอบวงกว้าง ด้วยจำนวนรอบการค้นหาร r ดังรูปที่ 4.9 (ข) ซึ่งมากกว่าการทดสอบ ATS แบบเดี่ยว ครั้งที่ 3 ทั้งนี้สามารถสรุปความสัมพันธ์ของจำนวนรอบการค้นหาที่ได้ดังรูปที่ 4.9 (ข) เป็นตามสมการที่ (4.10)

$$n < p < q < r \quad (4.10)$$

สำหรับปัญหาการหาค่าเหมาะที่สุดแบบสองมิติ ระยะขจัด (displacement) จากคำตอบเริ่มต้นไปหาคำตอบวงกว้าง ถูกกำหนดให้เป็นค่า $|\bullet|$ เช่น ระยะขจัด $\overline{P}_1$ จาก S_{10} ไปหา GS มีขนาดเท่ากับ $|\overline{P}_1|$ และการขจัด $\overline{P}_4$ จาก S_{40} ไป GS คือ $|\overline{P}_4|$ แสดงดังรูปที่ 4.9 (ก) มักพบว่ามีความสัมพันธ์กับจำนวนรอบการค้นหา (iteration) แบบตรงไปตรงมา หมายความว่า การทดสอบ ATS แบบเดี่ยว ครั้งที่ i มีคำตอบเริ่มต้นซึ่งอยู่ใกล้คำตอบวงกว้างมักใช้จำนวนรอบการค้นหาน้อยกว่ากรณีที่ i ใช้คำตอบเริ่มต้นห่างคำตอบวงกว้างมากกว่านั่นเอง หากพิจารณาจากรูปที่ 4.9 (ก) จะพบความสัมพันธ์ระยะขจัด ตามรูปที่ 4.9 (ก) เป็นไปตามสมการที่ (4.11)

$$|\overline{P}_1| < |\overline{P}_2| < |\overline{P}_3| < |\overline{P}_4| \quad (4.11)$$

ซึ่งสอดคล้องกับความสัมพันธ์ของจำนวนรอบการค้นหาที่ใช้ไปในแต่ละครั้ง การทดสอบของ ATS แบบเดี่ยว ดังสมการที่ (4.10) นั่นเอง

การหาค่าเฉลี่ยจำนวนรอบการค้นหาที่ใช้หาคำตอบวงกว้างของ ATS แบบเดี่ยว จากการทดสอบจำนวน 4 ครั้ง สามารถคำนวณได้ตามสูตรในสมการที่ (4.12)

$$Count_{ATS} = \frac{\sum_{i=1}^k n_i}{k} \quad (4.12)$$

เมื่อ k คือจำนวนครั้งของการทดสอบ ATS แบบเดี่ยว และ n_i คือจำนวนรอบการค้นหาก็ใช้ในการค้นหาคำตอบวงกว้างครั้งที่ i จะได้ค่า $Count_{ATS}$ แสดงดังสมการที่ (4.13)

$$Count_{ATS} = \frac{(n+p+q+r)}{4} \quad (4.13)$$

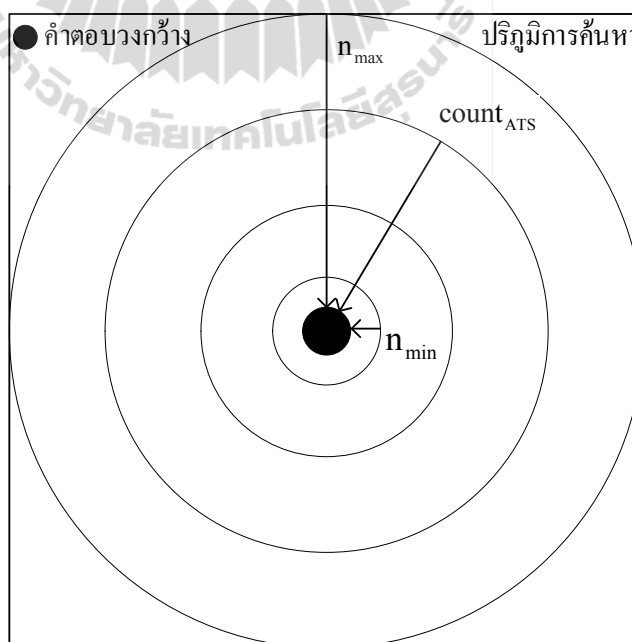
หากกำหนดให้ n_{min} คือจำนวนรอบการค้นหาคำน้อยที่สุดที่ ATS แบบเดี่ยวใช้ในการค้นหาคำตอบวงกว้างที่ได้จากคำตอบเริ่มต้น S_{min} สามารถจัด S_{min} เป็นปัญหาสำหรับการวิเคราะห์อัลกอริทึมเชิงความซับซ้อนของการใช้เวลาแบบกรณีดีที่สุด (best-case time complexity analysis) ได้ และทางตรงกันข้าม ถ้า n_{max} คือจำนวนรอบการค้นหามากที่สุดที่ ATS แบบเดี่ยวใช้ในการค้นหาคำตอบวงกว้างที่ได้จากคำตอบเริ่มต้น S_{max} ก็สามารถจัด S_{max} เป็นปัญหาสำหรับการวิเคราะห์อัลกอริทึมเชิงความซับซ้อนของการใช้เวลาค้นหาแบบกรณีแย่ที่สุด (worst-case time complexity analysis) ได้ แต่บทความนี้สนใจศึกษาปัญหาสำหรับการวิเคราะห์อัลกอริทึมเชิงซับซ้อนของการใช้เวลาค้นหาแบบกรณีเฉลี่ย (average-case time complexity analysis) จึงต้องหาค่าเฉลี่ยจำนวนรอบการค้นหาคำตอบเริ่มต้นที่แตกต่างกันทั่วทั้งปริภูมิการค้นหาพิจารณาจากรูปที่ 4.9 (ข) ได้ค่า $n_{min} = n$ มาจากเส้นทางการค้นหา P_1 ของการทดสอบครั้งที่ 1 และ $n_{max} = r$ จากเส้นทางการค้นหา P_4 ของการทดสอบครั้งที่ 4 จะได้ค่าเฉลี่ยจำนวนรอบการค้นหาคำตอบวงกว้างของ ATS แบบเดี่ยว หรือ $Count_{ATS}$ ของ ATS แบบเดี่ยว มีค่าระหว่าง n_{min} กับ n_{max} การปรับแต่งค่าพารามิเตอร์ของ ATS แบบเดี่ยว อย่างเหมาะสมจะทำให้ได้ค่า $Count_{ATS}$ ใกล้เคียงกับค่าของ n_{min} มากกว่าค่าของ n_{max} นั้นหมายความว่า ATS แบบเดี่ยว ใช้เวลาการค้นหาคำตอบวงกว้างน้อยกว่าค่ากึ่งกลางระหว่าง n_{min} กับ n_{max} ทำให้ ATS แบบเดี่ยวมีประสิทธิภาพการค้นหาที่ด้นั้นเอง ทางสถิติหากข้อมูลของปัญหาที่วิเคราะห์มีการกระจายตัวแบบปกติ ค่า $Count_{ATS}$ ที่ได้จากการวิเคราะห์ตัวอย่างที่มากเพียงพอ ย่อมลู่เข้าหาค่าคงที่ค่าหนึ่ง แม้ว่าจะเพิ่มจำนวนตัวอย่างให้เข้าใกล้จำนวนประชากรก็ตาม แสดงดังสมการที่ (4.14)

$$\lim_{k \rightarrow \infty} \frac{\sum_{i=1}^k n_i}{k} = Count_{ATS} \quad (4.14)$$

นั่นหมายความว่าคำตอบที่ได้จากจำนวนตัวอย่างที่จำกัด (finite sample) แต่ให้คำตอบค่าใกล้เคียงกับคำตอบที่ได้จากจำนวนตัวอย่างมากพอ ๆ กับจำนวนประชากรหรืออนันต์ ย่อมเป็นคำตอบของ

ปัญหาได้ดีด้วยจำนวนครั้งการประเมินคำตอบที่น้อยกว่าประเมินคำตอบของประชากรทั้งหมด ในทางปฏิบัติเมื่อพยายามปรับแต่งค่าพารามิเตอร์ของ ATS แบบเดี่ยว ให้เหมาะกับปัญหาที่สุดแล้ว จึงทำการทดสอบกับกลุ่มตัวอย่างที่ดีพอและด้วยจำนวนครั้งที่มากพอ จะได้ค่า $Count_{ATS}$ ที่คงที่ และน่าเชื่อถือ แสดงดังสมการที่ (4.14) และสามารถแสดงได้ดังรูปที่ 4.10 ซึ่งค่า $Count_{ATS}$ จะอยู่ ระหว่าง n_{min} และ n_{max} หลังจากนั้น $Count_{ATS}$ จะถูกใช้เป็นค่าอ้างอิงสำหรับการปรับแต่ง ค่าพารามิเตอร์ของ MATS เพื่อให้ได้เวลาการค้นหาคำตอบวงกว้างที่สั้นกว่าของ ATS แบบเดี่ยว ต่อไป

การวิเคราะห์ความถูกต้องของคำตอบวงกว้างที่ได้จาก ATS แบบเดี่ยว พิจารณา จากรูปที่ 4.9 (ก) แสดงการทดสอบ ATS แบบเดี่ยว จำนวน 4 ครั้ง โดยเริ่มต้นการค้นหาแต่ละครั้ง ด้วยคำตอบที่แตกต่างกันจากส่วนต่าง ๆ ของปริภูมิการค้นหา แต่ทุกเส้นทางการค้นหาล้วนพยายาม สร้างเส้นทางการค้นหามุ่งสู่คำตอบวงกว้างจน ATS แบบเดี่ยว บรรลุเกณฑ์คุณภาพคำตอบที่ต้องการให้ได้ในที่สุดทุกเส้นทาง ดังนั้นการทดสอบนี้จึงแสดงให้เห็นได้เป็นอย่างดีว่า ATS แบบเดี่ยว สามารถให้ความถูกต้องของคำตอบได้อย่างน่าเชื่อถือภายใต้ค่าพารามิเตอร์ ของ ATS แบบเดี่ยว และการทดสอบที่เหมาะสม



รูปที่ 4.10 ค่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ที่ดีที่สุดจะอยู่ระหว่าง
กรณีดีที่สุด (n_{min}) กับกรณีแย่ที่สุด (n_{max})

ค่าเวลาการค้นหาของ ATS แบบเดี่ยวสามารถคำนวณได้จากค่า $Count_{ATS}$ ดังสมการที่ (4.15)

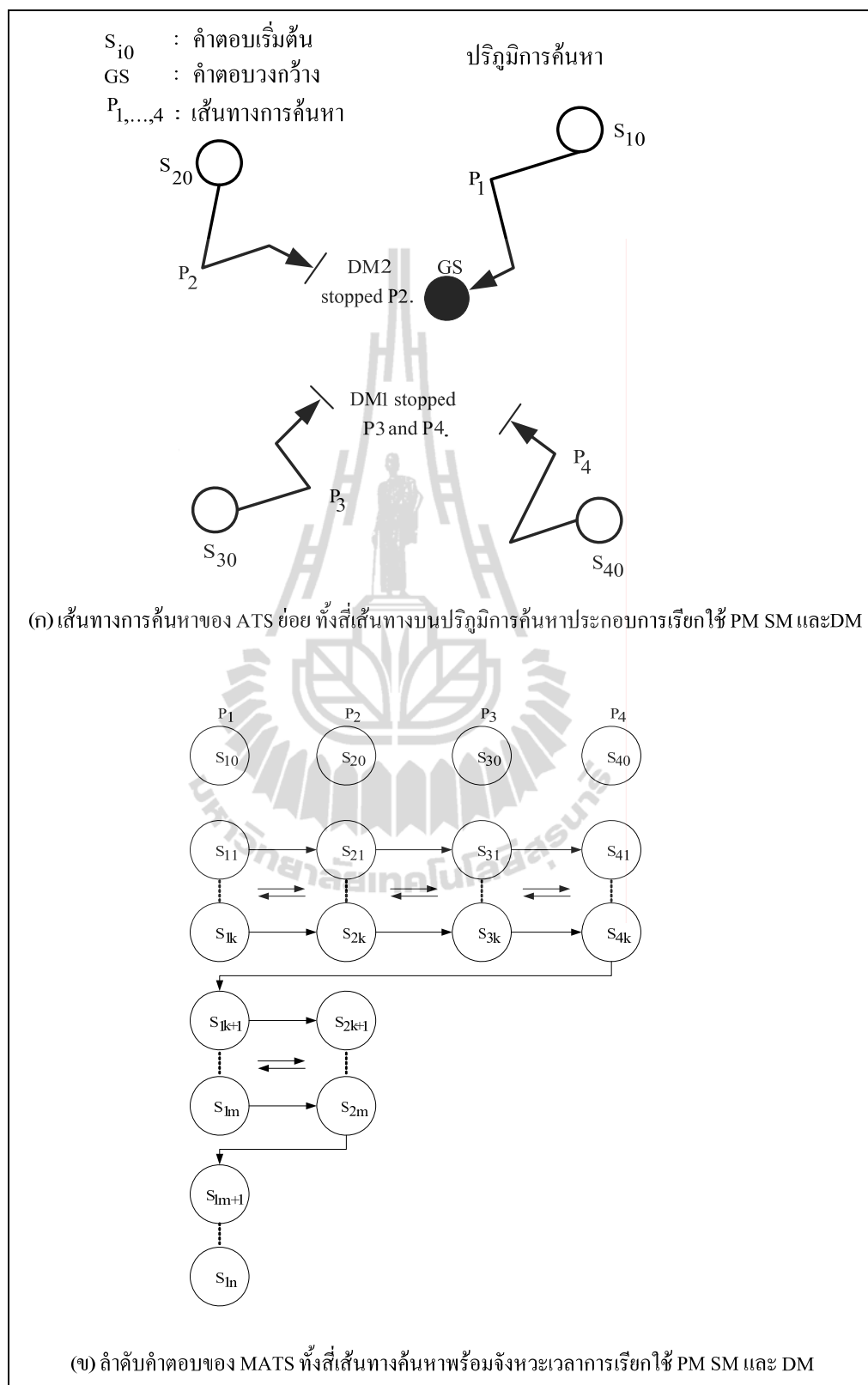
$$T_{S,ATS} = Count_{ATS} \times t \quad (4.15)$$

เมื่อ	$T_{S,ATS}$	คือ เวลาที่ ATS แบบเดี่ยว ใช้เพื่อการค้นหาคำตอบ (จากประมาณการ)
	$Count_{ATS}$	คือ จำนวนรอบการค้นหาเฉลี่ยที่ใช้ในการค้นหาของ ATS แบบเดี่ยว
	t	คือ เวลาเฉลี่ยที่ ATS แบบเดี่ยวใช้ในการค้นหาคำตอบใหม่แต่ละรอบ (ปริมาณสเกลาร์)

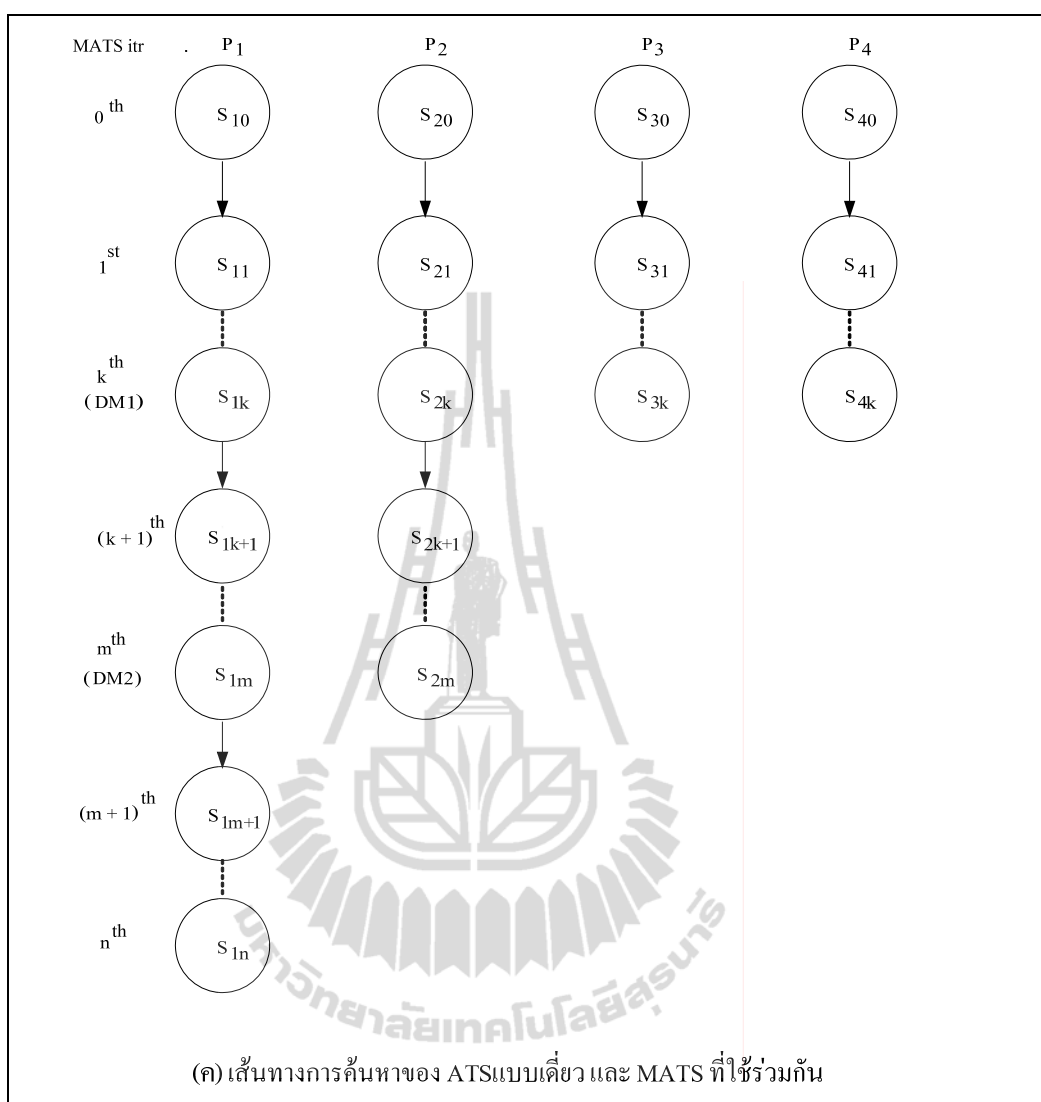
ต่อไปเป็นตัวอย่างการหาค่าเวลาการค้นหาเฉลี่ยของ MATS แบบ 4 เส้นทางค้นหา สำหรับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่

MATS แบบ 4 เส้นทางค้นหา ถูกยกเป็นตัวอย่างในการค้นหาคำตอบวงกว้าง สำหรับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่สองมิติ แสดงดังรูปที่ 4.11





รูปที่ 4.11 การค้นหาของ MATS แบบ 4 หน่วยค้นหา



รูปที่ 4.11 การค้นหาของ MATS แบบ 4 หน่วยค้นหา (ต่อ)

หากพิจารณารูปที่ 4.11 (ก) ที่แสดงเส้นทางการค้นหาของ MATS ทั้งสี่เส้นทาง ได้แก่ P_1 , P_2 , P_3 และ P_4 บนปริภูมิการค้นหา ซึ่งรูปแบบการวางตัวของคำตอบเริ่มต้นและเส้นทางการค้นหาจะคล้ายคลึงกับการทดลอง ATS แบบเดี่ยว จำนวนสี่ครั้งด้วยคำตอบเริ่มต้นที่แตกต่างกัน ดังแสดงในรูปที่ 4.9 (ก) ที่ได้อธิบายไปแล้วในหัวข้อก่อนหน้านี้ เจตนของการนำเสนอนี้ต้องการเปรียบเทียบให้ผู้อ่านเห็นถึงความเหมือนและความแตกต่างระหว่าง ATS แบบเดี่ยว กับ MATS ได้ชัดเจนขึ้นนั่นเอง หากนำ ATS แบบเดี่ยว มาทำหน้าที่เป็น ATS ย่อย ให้กับ MATS ด้วยการกำหนดคำตอบเริ่มต้นอย่างเหมาะสมให้กับ ATS ย่อยเหล่านั้น แล้วปล่อยให้ ATS ย่อย ดำเนินการค้นหาลงบนเส้นทางการค้นหาของตนเองอย่างอิสระตามจังหวะเวลาที่ได้รับการจัดสรรจาก

ตัวประมวลผลเดี่ยวทั้งนี้หากยังไม่ได้ใส่ DM และ TC ให้กับ MATS จนกระทั่ง ATS ย่อยทุกหน่วย ค้นหาสิ้นสุดการค้นหา ลำดับคำตอบที่ได้จาก ATS ย่อย แต่ละเส้นทางของ MATS นี้ จะคล้ายกับ ลำดับคำตอบที่ได้ในแต่ละครั้งทดสอบของ ATS แบบเดี่ยว ดังรูปที่ 4.9 (ข) นั่นเอง เพื่อชี้ให้เห็นว่า เมื่อนำ ATS แบบเดี่ยว ซึ่งมีคุณสมบัติการรู้เข้าหาคำตอบวงกว้างอยู่แล้วมาทำหน้าที่เป็น ATS ย่อย ใน MATS ก็ยังคงมีคุณสมบัติการรู้เข้าหาคำตอบวงกว้างอยู่เหมือนเดิม แต่พอเพิ่ม DM เข้าไปให้กับ MATS จะทำให้ ATS ย่อยบางเส้นทางต้องหยุดการค้นหาก่อนพบคำตอบวงกว้าง โดยที่คุณสมบัติ การรู้เข้าหาคำตอบวงกว้างของ ATS ย่อย (Sujitjorn et al., 2006) ที่เหลืออยู่ในวงรอบการค้นหาของ MATS ก็ยังคงมีอยู่เหมือนเดิม ส่วนการเพิ่ม TC ให้กับ MATS จะมีผลทำให้ ATS ย่อยทุกตัวที่ยังคง ทำงานในวงรอบการค้นหาของ MATS ถูกบังคับให้ยุติการค้นหาของ MATS ได้ทุกหน่วยค้นหาโดย ไม่จำเป็นต้องรอให้ครบแต่ละรอบการค้นหาหรือไม่ต้องรอให้มีการเรียกใช้ DM จนครบก่อน ซึ่ง TC ประเภทนี้สนใจที่คุณภาพคำตอบที่ ATS ย่อยหามาได้ หากได้คำตอบที่พอใจก็จะยุติการ ค้นหา MATS ลงนั่นเอง จากรูปที่ 4.11 (ก) แสดงเส้นทางการค้นหาของ MATS แบบสี่เส้นทาง โดยเริ่มต้นการค้นหาจากคำตอบเริ่มต้นจากทั้งสี่มุมของปริภูมิการค้นหา ระหว่างการค้นหาได้มีการ เรียกใช้ DM สองครั้ง ได้แก่ DM1 และ DM2 ตามลำดับ ทำให้ P_2 , P_3 และ P_4 ต้องหยุดการค้นหา เหลือเพียง P_1 ทำการค้นหาต่อไปจนพบคำตอบวงกว้าง และ MATS ยุติการค้นหาในที่สุด การสร้าง ลำดับคำตอบดูได้จากรูปที่ 4.11 (ก) ตั้งแต่เมื่อ PM จัดสรรคำตอบเริ่มต้นให้กับทุกหน่วยค้นหา เรียบร้อยแล้วในวงรอบการค้นหาที่ศูนย์ของ MATS พอขึ้นวงรอบการค้นหาที่ 1 ทุกหน่วยค้นหา ก็เรียงลำดับกันสร้างลำดับคำตอบของตนตามจังหวะของตัวประมวลผลที่วนมาถึงเข้าไปเข้ามา จนกระทั่งถึงรอบการค้นหาที่ k^{th} MATS ทำการเรียกใช้ DM1 เพื่อลดจำนวนหน่วยค้นหาจาก 4 ให้เหลือครึ่งหนึ่งคือ 2 ผลการประเมินของ DM1 มีผลทำให้ P_3 และ P_4 ต้องหยุดการค้นหา ได้ที่เหลือ ถูกยกเลิกการค้นหา ถัดจากนั้น ATS#1 และ ATS#2 ก็ทำงานใน MATS ต่อไปจนกระทั่ง การค้นหา ดำเนินมาถึงรอบการค้นหาที่ m^{th} MATS ก็เรียกใช้ DM2 ซึ่งเป็นการเรียกใช้ DM ครั้งที่สอง และเป็น ครั้งสุดท้ายเนื่องจากจะเหลือหน่วยค้นเพียงหน่วยค้นหาเดียวแล้ว ผลการประเมินของ DM2 ทำให้ P_2 ต้องหยุดการค้นหา เหลือไว้เพียง P_1 ให้ดำเนินการค้นหาต่อไปจนกระทั่งรอบการค้นหาที่ n^{th} ATS#1 ก็พบคำตอบวงกว้าง ทำให้ MATS ยุติการค้นหาลง ภาพลำดับคำตอบของทุกเส้นทางค้นหา ของ MATS แสดงดังรูปที่ 4.11 (ก) มีลำดับคำตอบของ P_1 เป็นหน่วยค้นหาที่พบคำตอบวงกว้าง จากที่อธิบายไว้แล้วข้างต้นว่าทุกหน่วยค้นหาไม่ว่าจะเป็น P_1 , P_2 , P_3 หรือ P_4 สามารถพบคำตอบ วงกว้างได้ทั้งสิ้น การยกเลิกหน่วยค้นหาของ DM และ การยุติการค้นหา MATS ได้โดย ATS ย่อย ทุกตัวไม่ว่าเวลาใดของ TC ล้วนมุ่งหมายให้เกิดการใช้เวลาค้นหาคำตอบวงกว้างของ MATS ให้น้อยที่สุด แต่ก็มีได้ทำให้ ATS ย่อย สูญเสียการมีอยู่ของคุณสมบัติการรู้เข้าหาคำตอบวงกว้างแต่ อย่างใด ดังคำอธิบายข้างต้น จึงสามารถสรุปได้ว่า หนึ่งใน ATS ย่อย ที่พบคำตอบวงกว้างและยุติ

การค้นหาคำของ MATS ลง ยังคงให้คุณภาพคำตอบกว้างที่มีความถูกต้องอยู่ในเกณฑ์ที่ยอมรับได้ เช่นเดียวกับ ATS แบบเดี่ยว ที่ได้รับการพิสูจน์ไว้เป็นอย่างดีแล้วใน (Sujitjorn et al., 2006)

DM มี 2 ค่าพารามิเตอร์อิสระให้ผู้ใช้สามารถปรับตั้งได้ ได้แก่ ค่าเมตริกซ์การลดจำนวนหน่วยค้นหาที่เบื้องต้นเลือกรูปแบบการลดจำนวนแบบครั้งละกึ่งหนึ่งของจำนวนก่อนการเรียกใช้ DM เป็นรูปแบบการลดจำนวนหน่วยค้นหาตัวอย่างหนึ่ง และ ค่าเมตริกซ์การเรียกใช้ DM ซึ่งทั้งสองค่าจะสัมพันธ์กับจำนวนครั้งของการประเมินคำตอบ ATS ย่อยตลอดการค้นหาคำของ MATS ดังสมการที่ (4.16)

$$T_{S,MATS} = n \times e \times d \times t \quad (4.16)$$

เมื่อ	$T_{S,MATS}$	คือ เวลาที่ MATS ใช้เพื่อการค้นหาคำตอบ (จากประมาณการ)
n		คือ เมตริกซ์จำนวนหน่วยค้นหาที่ลดลงของ DM มีลักษณะเป็นเมตริกซ์แถวและได้จากการกำหนดของผู้ใช้
e		คือ เมตริกซ์เรียกใช้ DM มีลักษณะเป็นเมตริกซ์ทแยงมุมและเป็นค่าจากการทดสอบ
d		คือ เมตริกซ์จำนวนรอบการค้นหาที่เรียกใช้ DM มีลักษณะเป็นเมตริกซ์หลักและเป็นค่าจากการกำหนดของผู้ใช้
t		คือ เวลาเฉลี่ยที่ ATS ย่อย ใช้ในแต่ละรอบการค้นหาจนกระทั่งได้คำตอบใหม่หนึ่งค่า (ปริมาณสเกลาร์)

กำหนดให้ N คือจำนวนหน่วยค้นหาเริ่มต้นทั้งหมด และเป็นจำนวนเต็มบวก $N = 2, 3, \dots, k$ เมื่อ k คือจำนวนครั้งที่มีการเรียกใช้ DM ทั้งหมด จะได้ $n = [n_0 \ n_1 \ n_2 \ \dots \ n_{k-1} \ n_k]$

เมื่อ	n_0	คือ จำนวนหน่วยค้นหาที่เหลืออยู่ก่อน MATS ยุติการค้นหา
	n_1	คือ จำนวนหน่วยค้นหาที่ลดลงเมื่อมีการเรียก DM ครั้งที่ 1
	n_2	คือ จำนวนหน่วยค้นหาที่ลดลงเมื่อมีการเรียก DM ครั้งที่ 2
	n_i	คือ จำนวนหน่วยค้นหาที่ลดลงเมื่อมีการเรียก DM ครั้งที่ i

เมื่อ $n_i \in \{1, 2, \dots, N-1\}$, ($i = 1, \dots, k$) และ $k < N$ ภายใต้เงื่อนไข $N = \sum_{i=0}^k n_i$

$$e = \begin{bmatrix} e_0 & 0 & \cdots & 0 & 0 \\ 0 & e_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & e_{k-1} & 0 \\ 0 & 0 & \cdots & 0 & e_k \end{bmatrix}, e_0 = 1 \text{ และ } e_i \in \{0,1\}$$

นั่นคือ DM ใดถูกเรียกใช้ e นั้นจะมีสถานะเป็น 1 มิฉะนั้นก็จะมีสถานะเป็น 0

เมื่อ e_0 คือ สถานะของ ATS ย่อย ที่เหลืออยู่ก่อน MATS ยุติการค้นหา $e_0 = 1$ เสมอ

e_1 คือ สถานะของ ATS ย่อย ที่ลดลงเมื่อมีการเรียก DM ครั้งที่ 1

e_2 คือ สถานะของ ATS ย่อย ที่ลดลงเมื่อมีการเรียก DM ครั้งที่ 2

e_i คือ สถานะของ ATS ย่อย ที่ลดลงเมื่อมีการเรียก DM ครั้งที่ i

และ $d = [d_0 \ d_1 \ d_2 \ \cdots \ d_{k-1} \ d_k]^T$ โดยที่ $d_0 = \text{Count}_{\text{MATS}}$ คือจำนวนรอบการค้นหาของ MATS ที่ใช้
หาคำตอบวงกว้างภายใต้เงื่อนไข $0 < d_1 < d_2 \cdots < d_{k-1} < d_k < \text{Count}_{\text{ATS},av}$

$\text{Count}_{\text{ATS},av}$ คือค่าเฉลี่ยจำนวนรอบการค้นหาที่ ATS ใช้หาคำตอบวงกว้างแต่ละ
ปัญหาซึ่งเป็นแนวทางในการตั้งค่าเรียกใช้ DM พิจารณาได้จากค่าตัวอย่างในตารางที่ 3

ตัวอย่างที่ 1 การตั้งค่าการเรียกใช้ DM สำหรับกรณีที่มีหน่วยค้นหาทั้งหมด 4 ตัว
นั่นคือ $N = 4$ ถ้าใช้รูปแบบการลดจำนวนหน่วยค้นหาตามปกติจะเรียกใช้ DM จำนวน 2 ครั้ง
จากจำนวนหน่วยค้นหา 4 ให้เหลือ 1 หน่วยค้นหา เมตริกซ์ n จะได้สมาชิกมีค่าดังนี้ $n_1 = 2$ (ลดลง
ครั้งหนึ่งจาก 4 เมื่อเรียกใช้ DM1) $n_2 = 1$ (ลดลงครั้งหนึ่งจาก 2 เมื่อเรียกใช้ DM2) และ
 $n_0 = N - n_1 - n_2 = 4 - 2 - 1 = 1$ (จำนวนหน่วยที่เหลืออยู่ก่อน MATS ยุติการค้นหาคือหนึ่งหน่วย
ค้นหา) เมื่อตรวจสอบเงื่อนไข $N = \sum_{i=0}^k n_i$ พบว่า $4 = 1 + 2 + 1$ ยังคงเป็นจริง ส่วนเมตริกซ์ d จะได้
สมาชิกมีค่าดังนี้ $d_0 = \text{Count}_{\text{MATS}}$ ในที่นี้กำหนดให้มีการเรียกใช้ DM ครั้งที่ 1 และ 2 ในรอบการ
ค้นหาที่ 10 และ 20 ตามลำดับ จึงได้ค่า $d_1 = 10$ และ $d_2 = 20$ นั่นเอง หากการทดสอบพบว่าการ
เรียกใช้ DM ครบทั้งสองครั้ง จะมีความสัมพันธ์แสดง $T_{S,\text{MATS}}$ ดังสมการที่ (4.17)

$$\begin{aligned} T_{S,\text{MATS}} &= [1 \ 2 \ 1] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \text{Count}_{\text{MATS}} \\ 10 \\ 20 \end{bmatrix} t \\ &= (\text{Count}_{\text{MATS}} + 2 \times 10 + 1 \times 20) t \\ &= (\text{Count}_{\text{MATS}} + 40) t \end{aligned} \quad (4.17)$$

ค่า $n_0 = N - n_1 - n_2 = 4 - 2 - 1 = 1$ เนื่องจากเหลือเพียงหนึ่งหน่วยค้นหา ก่อน MATS ยุติการค้นหา สมมติให้ MATS ดำเนินการค้นหาต่อไปจนพบคำตอบวงกว้างที่รอบการค้นหาที่ 35 จะได้ $Count_{MATS} = 35$ เมื่อแทนค่าลงสมการที่ (4.17) จะได้ $T_{S,MATS} = (35 + 40)t = 75t$ หมายความว่า MATS ใช้เวลาการค้นหาไป $75t$ นั่นเอง

ตัวอย่างที่ 2 กรณีที่มีการเรียกใช้ DM เพียง 1 ครั้งลดจำนวนหน่วยค้นหาจาก 4 เหลือ 2 แล้วพบ GS สมการที่ (4.18) จึงแสดงความสัมพันธ์ของค่า $T_{S,MATS}$

$$\begin{aligned} T_{S,MATS} &= [2 \ 2 \ 0] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} Count_{MATS} \\ 10 \\ 20 \end{bmatrix} t \\ &= (2 \times Count_{MATS} + 2 \times 10)t \\ &= (2 \times 15 + 20)t = 50t \end{aligned} \quad (4.18)$$

ค่า $n_0 = N - n_1 = 4 - 2 = 2$ เนื่องจากเหลือสองหน่วยค้นหา ก่อนที่ MATS จะยุติการค้นหา ภายใต้การประมาณว่าในรอบการค้นหา $Count_{MATS}$ ก่อน MATS ยุติการค้นหาได้ค้นหาครบทุกเส้นทางแล้ว และ $e_2 = e_3 = 0$ แต่ $e_1 = 1$ ตามสมการที่ (4.18) เป็นกรณีสมมติให้มีการพบคำตอบวงกว้างก่อนการเรียกใช้ DM2 ซึ่งตั้งไว้ที่รอบการค้นหาที่ 20 ถ้าสมมติให้ $Count_{MATS} = 15$ แล้วแทนค่าลงในสมการที่ (4.18) จะได้ $T_{S,MATS} = (30 + 20)t = 50t$ หมายความว่า MATS ใช้เวลาการค้นหาไป $50t$ นั่นเอง เปรียบเทียบกับกรณีก่อนหน้า แม้ว่า DM ถูกเรียกไม่ครบสองครั้งแต่สามารถทำเวลาการค้นหาได้น้อยกว่าถึง $25t$ ทั้งนี้ขึ้นอยู่กับความเหมาะสมของแต่ละปัญหา

ตัวอย่างที่ 3 การตั้งค่าการเรียกใช้ DM สำหรับ MATS ที่มีหน่วยค้นหาทั้งหมด 8 หน่วยค้นหา นั่นคือ $N = 8$ ถ้าใช้รูปแบบการลดจำนวนหน่วยค้นหาตามปกติจะเรียกใช้ DM จำนวน 3 ครั้ง จากจำนวนหน่วยค้นหา 8 ให้เหลือ 1 หน่วยค้นหา เมตริกซ์ n จะได้สมาชิกมีค่าดังนี้ $n_0 = 1$ และ $n_1 = 4$ (ลดลงครึ่งหนึ่งจาก 8 เมื่อเรียกใช้ DM1) $n_2 = 2$ (ลดลงครึ่งหนึ่งจาก 4 เมื่อเรียกใช้ DM2) และ $n_3 = 1$ (ลดลงครึ่งหนึ่งจาก 2 เมื่อเรียกใช้ DM3) เมื่อตรวจสอบเงื่อนไข $N = \sum_{i=0}^k n_i$ พบว่า $8 = 1 + 4 + 2 + 1$ ยังคงเป็นจริง ส่วนเมตริกซ์ d จะได้สมาชิกมีค่าดังนี้ $d_0 = Count_{MATS}$ ในที่นี้กำหนดให้มีการเรียกใช้ DM ครั้งที่ 1 2 และ 3 ในรอบการค้นหาที่ 10 20 และ 30 เรียงตามลำดับ จึงได้ค่า $d_1 = 10$, $d_2 = 20$ และ $d_3 = 30$ นั่นเอง หากการทดสอบพบว่าการเรียกใช้ DM ครบทั้งสามครั้ง จะได้ดังสมการที่ (4.19)

$$\begin{aligned}
T_{S,MATS} &= [1 \ 4 \ 2 \ 1] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Count_{MATS} \\ 10 \\ 20 \\ 30 \end{bmatrix} t \\
&= (Count_{MATS} + 4 \times 10 + 2 \times 20 + 1 \times 30)t \\
&= (Count_{MATS} + 110)t
\end{aligned} \tag{4.19}$$

ค่า $n_0 = N - n_1 - n_2 - n_3 = 8 - 4 - 2 - 1 = 1$ เนื่องจากเหลือเพียงหนึ่งหน่วยค้นหา ก่อน MATS ยุติการค้นหา จากสมการที่ (4.19) สมมติให้ MATS ดำเนินการค้นหาต่อไปจนพบคำตอบวงกว้างที่รอบการค้นหาที่ 35 จะได้ $Count_{MATS} = 35$ เมื่อแทนค่าลงในสมการที่ (4.18) จะได้ $T_{S,MATS} = (35 + 110)t = 145t$ หมายความว่า MATS ใช้เวลาการค้นหาไป $145t$ นั่นเอง

ตัวอย่างที่ 4 สำหรับ MATS ที่มีหน่วยค้นหาทั้งหมด 8 หน่วยค้นหา และมีการเรียกใช้ DM เพียง 1 ครั้ง ลดจำนวนหน่วยค้นหาจาก 8 เหลือ 4 หน่วยค้นหา แล้วพบ GS ได้ดังสมการที่ (4.20)

$$\begin{aligned}
T_{S,MATS} &= [4 \ 4 \ 0 \ 0] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} Count_{MATS} \\ 10 \\ 20 \\ 30 \end{bmatrix} t \\
&= (4 \times Count_{MATS} + 4 \times 10)t \\
&= (4Count_{MATS} + 40)t
\end{aligned} \tag{4.20}$$

ค่า $n_0 = N - n_1 = 8 - 4 = 4$ เนื่องจากเหลือสี่หน่วยค้นหา ก่อน MATS ยุติค้นหานั่นเอง ภายใต้การประมาณว่าในรอบการค้นหา $Count_{MATS}$ ก่อน MATS ยุติการค้นหาได้ค้นหาครบทุกเส้นทางแล้ว และ $e_2 = e_3 = 0$ แต่ $e_1 = 1$ จากสมการที่ (4.19) เป็นกรณีที่มีการพบคำตอบวงกว้างก่อนการเรียกใช้ DM2 ซึ่งตั้งไว้ที่รอบการค้นหาที่ 20 ถ้าสมมติให้ $Count_{MATS} = 15$ แล้วแทนค่าลงในสมการที่ (4.20) จะได้ $T_{S,MATS} = (4 \times 15 + 40)t = 100t$ หมายความว่า MATS ใช้เวลาการค้นหาไป $100t$ นั่นเอง

ตัวอย่างที่ 5 สำหรับ MATS ที่มีหน่วยค้นหาทั้งหมด 8 หน่วยค้นหา และมีการเรียกใช้ DM เพียง 2 ครั้ง ลดจำนวนหน่วยค้นหาจาก 8 เหลือ 2 หน่วยค้นหา แล้วพบ GS จะได้ดังสมการที่ (4.21)

$$\begin{aligned}
T_{S,MATS} &= [2 \ 4 \ 2 \ 0] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} Count_{MATS} \\ 10 \\ 20 \\ 30 \end{bmatrix} t \\
&= (2 \times Count_{MATS} + 4 \times 10 + 2 \times 20)t \\
&= (2Count_{MATS} + 80)t
\end{aligned} \tag{4.21}$$

ค่า $n_0 = N - n_1 - n_2 = 8 - 4 - 2 = 2$ เนื่องจากเหลือสองหน่วยค้นหา ก่อน MATS ยุติค้นหานั่นเอง ภายใต้การประมาณว่าในรอบการค้นหา $Count_{MATS}$ ก่อน MATS ยุติการค้นหาได้ค้นหาครบทุกเส้นทางแล้ว และ $e_3 = 0$ แต่ $e_1 = e_2 = 1$ จากสมการที่ (4.21) เป็นกรณีที่มีการพบคำตอบวงกว้าง ก่อนการเรียกใช้ DM3 ซึ่งตั้งไว้ที่รอบการค้นหาที่ 30 ถ้าสมมติให้ $Count_{MATS} = 25$ แล้วแทนค่าลงในสมการที่ (4.21) จะได้ $T_{S,MATS} = (2 \times 25 + 80)t = 130t$ หมายความว่า MATS ใช้เวลาการค้นหาไป $130t$ นั่นเอง

เปรียบเทียบเวลาการค้นหาสำหรับกรณี MATS แบบ 8 เส้นทางค้นหา จากตัวอย่างที่ 3 ถึง ตัวอย่างที่ 5 พบว่ากรณีที่มีการเรียกใช้ DM เพียงหนึ่งครั้งใช้เวลาน้อยที่สุด รองลงมาเป็นกรณีเรียกใช้ DM สองครั้ง และช้าที่สุดเป็นของการเรียกใช้ DM ครบทั้งสามครั้ง ในทางปฏิบัติจริง ต้องมีการปรับแต่งค่าพารามิเตอร์ของ MATS โดยเฉพาะเมตริกซ์ d เพื่อให้ได้สมรรถนะการค้นหาที่ดีขึ้นกับลักษณะของแต่ละปัญหา เช่น การเปลี่ยนเมตริกซ์ d ในตัวอย่างที่ 3 ตามสมการที่ (4.19) ให้เป็น $d = [Count_{MATS} \ 5 \ 10 \ 15]^T$ แล้วคำนวณใหม่จะได้ $T_{S,MATS} = (Count_{MATS} + 4 \times 5 + 2 \times 10 + 1 \times 15)t$ หากสมมติให้ $Count_{MATS} = 35$ เหมือนเดิมจะได้ $T_{S,MATS} = (35 + 4 \times 5 + 2 \times 10 + 1 \times 15)t = 90t$ ซึ่งลดลงจากเดิม $145t$ เหลือ $90t$ เป็นการชี้ให้เห็นว่าการกำหนดเมตริกซ์ d ที่เหมาะสมย่อมทำให้ผลการค้นหาของ MATS ดีขึ้นได้ ทำนองเดียวกันกับตัวอย่างที่ 4 ตามสมการที่ (4.20) หากแทนที่เมตริกซ์ d ด้วย $d = [Count_{MATS} \ 5 \ 10 \ 15]^T$ สามารถคำนวณค่าใหม่ได้ $T_{S,MATS} = (4 \times Count_{MATS} + 20)t$ หากแทน $Count_{MATS} = 15$ เช่นเดิมจะได้ $T_{S,MATS} = (4 \times 15 + 20)t = 80t$ ซึ่งลดลงจากเดิม $100t$ เหลือ $80t$ และสุดท้าย ตัวอย่างที่ 5 ที่มีการเรียกใช้ DM2 หากแทนที่เมตริกซ์ d ดังกล่าวลงในสมการที่ (4.21) จะได้ $T_{S,MATS} = (2 \times Count_{MATS} + 40)t$ และแทนค่า $Count_{MATS} = 25$ ลงไป เช่นเดิมจะได้ $T_{S,MATS} = (2 \times 25 + 40)t = 90t$ ซึ่งลดลงจาก $130t$ เหลือ $90t$

การวิเคราะห์ MATS ในมุมมองด้านความถูกต้องของคำตอบวงกว้างที่หาได้นั้น อยู่บนความเชื่อที่ว่า ATS ย่อย ภายในกำกับ ของ MATS ทุกเส้นทางมีคุณสมบัติการรู้เข้าหาคำตอบวงกว้างไม่ว่าคำตอบเริ่มต้นจะอยู่ ณ ตำแหน่งใดของปริภูมิการค้นหา เช่นเดียวกับ ATS แบบเดี่ยว

ซึ่งได้กล่าวถึงไปแล้วในหัวข้อก่อนหน้านี้และยืนยันด้วยการพิสูจน์อย่างเป็นแบบแผนใน (Sujitjorn et al., 2006) ผู้อ่านที่สนใจรายละเอียดเกี่ยวกับบทพิสูจน์การหาคำตอบวงกว้างของ ATS แบบเดี่ยว สามารถอ่านเพิ่มเติมได้จากเอกสารอ้างอิงดังกล่าว ทั้งนี้ด้วยเหตุผลที่ว่ากลไกภายใน MATS มิได้กระทบต่อกระบวนการสร้างลำดับคำตอบในแง่ของคุณภาพคำตอบที่ได้สำหรับ ATS ย่อยแต่ละเส้นทาง มีผลเพียงการให้สิทธิ์และระดับสิทธิ์การค้นหาคำตอบของแต่ละ ATS ย่อย เท่านั้นเอง

4.3.2 การวิเคราะห์ประสิทธิภาพเชิงเวลา

จากหัวข้อก่อนหน้านี้เรื่องความถูกต้องของอัลกอริทึมได้แสดงให้เห็นด้วยตัวอย่างแล้วว่าคำตอบที่ได้จากทั้งของ ATS แบบเดี่ยว และ MATS มีความถูกต้องน่าเชื่อถือ ส่วนการวิเคราะห์ความซับซ้อนเชิงเวลาที่ใช้สำหรับการค้นหาของ MATS ก็ได้แสดงด้วยตัวอย่างให้เห็นถึงความสามารถในการลดเวลาการค้นหาด้วยการปรับแต่งพารามิเตอร์ของ DM ส่วนการวิเคราะห์ความซับซ้อนเชิงเวลาการค้นหาของ ATS แบบเดี่ยว ก็ได้แสดงให้เห็นแล้วว่าค่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ที่ผ่านการปรับแต่งค่าพารามิเตอร์ของ ATS แบบเดี่ยว อย่างดีแล้วจะให้ค่า $Count_{ATS}$ ที่คงที่ค่าหนึ่งในแต่ละปัญหาซึ่งสอดคล้องกับสมการที่ (4.14) และรูปที่ 4.10

หัวข้อนี้เป็นการพิจารณาประสิทธิภาพเชิงเวลาของ MATS เทียบกับ ATS แบบเดี่ยว เพื่อใช้เป็นตัววัดสมรรถนะของ MATS ที่พัฒนาต่อเนื่องมาจาก ATS แบบเดี่ยว ให้ชัดเจนยิ่งขึ้น การดำเนินการเริ่มต้นด้วยการแปลง $Count_{MATS}$ ให้เป็นจำนวนรอบการค้นหาของ MATS สมมูลกับ ATS แบบเดี่ยว หรือ $Count_{MATSeq}$ ด้วยสมการที่ (4.22) ซึ่งใช้แนวคิดเดียวกันกับสมการที่ (4.16) เพราะเหตุว่าค่า $Count_{MATS}$ แสดงจำนวนรอบการค้นหาของ MATS ซึ่งซ่อนจำนวนรอบการค้นหาของ ATS ย่อย หลายหน่วยค้นหาไว้ภายในหาคำนวณจำนวนรอบการค้นหาของ ATS ย่อย ต่าง ๆ มาเรียงต่อกันเราจะได้ $Count_{MATSeq}$ ที่เทียบเท่าจำนวนรอบการค้นหาของ ATS ย่อยทั้งหมดของ MATS ซึ่งอยู่ในฐานเวลาเดียวกันกับ $Count_{ATS}$ จึงสามารถเปรียบเทียบกันได้อย่างสมเหตุสมผล

$$Count_{MATSeq} = n_0 \cdot Count_{MATS} + \sum_{i=1}^k n_i \cdot d_i, \quad 1 < k \leq N \quad \text{และ} \quad N = \sum_{i=0}^k n_i \quad (4.22)$$

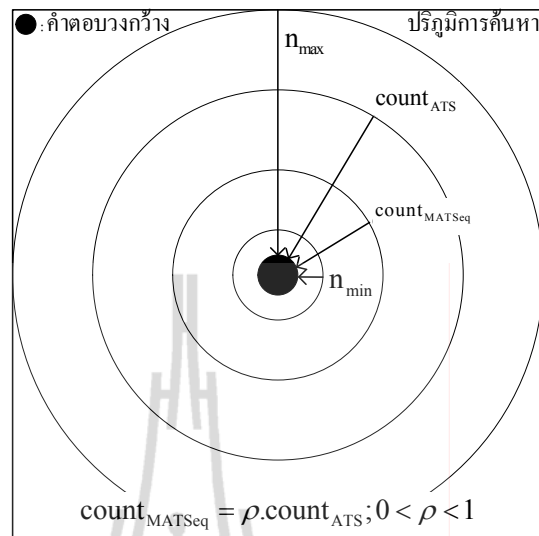
ถ้านิยามให้ประสิทธิภาพเชิงเวลาการค้นหาของ MATS เทียบกับ ATS แบบเดี่ยว คือสัดส่วนเวลาการค้นหาของ ATS แบบเดี่ยว ต่อเวลาการค้นหาของ MATS แสดงดังสมการที่ (4.23)

$$\eta_{MATS/ATS} = \frac{T_{S,ATS}}{T_{S,MATS}} = \frac{t \cdot Count_{ATS}}{t \cdot Count_{MATSeq}} \quad (4.23)$$

เมื่อ $\eta_{MATS/ATS}$ คือประสิทธิภาพเชิงเวลาการค้นหาของ MATS เทียบกับ ATS แบบเดี่ยว $T_{S,ATS}$ คือเวลาการค้นหาค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ $T_{S,MATS}$ คือเวลาการค้นหาค้นหาเฉลี่ยของ MATS และ $Count_{MATSeq}$ คือจำนวนรอบการค้นหาค้นหาเฉลี่ยของ MATS ที่สมมูลกับรอบการค้นหาค้นหาเฉลี่ยของ ATS แบบเดี่ยว จากสมการที่ (4.23) วิธีการหนึ่งที่จะทำให้ประสิทธิภาพของ MATS ดีกว่าของ ATS แบบเดี่ยว ได้นั้น คือการทำให้ค่า $Count_{MATSeq}$ มีค่าน้อยกว่า $Count_{ATS}$ ซึ่งขึ้นกับการปรับตั้งของ n_i และ d_i อย่างเหมาะสมนั่นเอง

การปรับแต่งค่าเมตริกซ์ d เพื่อให้ได้สมรรถนะการค้นหาของ MATS ที่ดีขึ้น ดังตัวอย่างที่ได้กล่าวไว้แล้วข้างต้น สามารถอาศัยข้อมูลพารามิเตอร์ของ ATS แบบเดี่ยว ที่ผ่านการปรับแต่งจนได้สมรรถนะการค้นหาของ ATS แบบเดี่ยวที่ดีที่สุดมาเป็นค่าพารามิเตอร์ของ ATS ย่อยใน MATS เท่ากับว่า MATS ที่มี ATS ย่อยสมรรถนะดีร่วมกับการปรับแต่งพารามิเตอร์ของ MATS เอง ย่อมมีโอกาสให้สมรรถนะการค้นหาที่ดีกว่าของ ATS แบบเดี่ยวได้ โดยเฉพาะการปรับแต่งพารามิเตอร์ของ DM อย่างเหมาะสมจะทำให้ได้เส้นทางการค้นหาที่มีระยะขจัดใกล้คำตอบวงกว้างเหลืออยู่ในวงรอบการค้นหาค้นหาของ MATS เป็นเหตุทำให้ $Count_{MATS}$ น้อยกว่า $Count_{ATS}$ ได้นั้นหมายความว่าหากดำเนินการต่อไปอย่างเหมาะสมก็สามารถทำให้ค่า $Count_{MATSeq}$ น้อยกว่า $Count_{ATS}$ ได้ในที่สุด

นอกจากนี้ DM ที่ทำให้จำนวนหน่วยค้นหาลดลงจะช่วยลดเวลาคอยของหน่วยค้นหาที่อยู่ใกล้คำตอบวงกว้างทำให้ $T_{S,MATS}$ น้อยกว่า $T_{S,ATS}$ ในที่สุดการนำเสนอให้เห็นโอกาสที่ MATS จะมีสมรรถนะการค้นหาที่ดีกว่า ATS แบบเดี่ยว สามารถเปรียบเทียบกันด้วยจำนวนรอบการค้นหาค้นหาของ MATS ที่สั้นลงกว่าของ ATS แบบเดี่ยว แสดงดังรูปที่ 4.12 ด้วยสัดส่วน ρ ซึ่งมีค่ามากกว่าศูนย์แต่น้อยกว่าหนึ่ง อันเนื่องจาก PM และ DM ที่ได้นำเสนอไปแล้วข้างต้น เป็นเหตุผลที่ทำให้ MATS สามารถค้นหาคำตอบได้รวดเร็วกว่า ATS แบบเดี่ยวในบางปัญหา



รูปที่ 4.12 ค่าจำนวนรอบการค้นหาลดลงของ MATS สั้นกว่าของ ATS แบบเดี่ยว
ด้วยอัตราส่วน ρ ทำให้ MATS มีโอกาสพบคำตอบ
ได้รวดเร็วกว่าของ ATS แบบเดี่ยว

เมื่อแทนค่าความสัมพันธ์ $Count_{MATSeq} = \rho \cdot Count_{ATS}$ จากรูปที่ 4.12 ลงใน
สมการที่ (4.23) จะได้ค่าประสิทธิภาพการค้นหาดังสมการที่ (4.24)

$$\eta_{MATS/ATS} = \frac{t \cdot Count_{ATS}}{t \cdot Count_{MATSeq}} = \frac{t \cdot Count_{ATS}}{t \cdot \rho \cdot Count_{ATS}} = \frac{1}{\rho}, \quad 0 < \rho < 1 \quad (4.24)$$

MATS จะมีประสิทธิภาพการค้นหาสูงกว่าของ ATS แบบเดี่ยว เมื่อคัดสรร
พารามิเตอร์ของ DM อย่างเหมาะสมจนทำให้ $Count_{MATSeq}$ น้อยกว่า $Count_{ATS}$ ด้วยสัดส่วน ρ
แสดงดังสมการที่ (4.24) การประเมินประสิทธิภาพการค้นหานี้ จำเป็นต้องพึงพาการทดลองบน
คอมพิวเตอร์กับปัญหาที่ซับซ้อนเป็นจำนวนครั้งที่มากพอ เนื้อหารายละเอียดจึงได้นำเสนอเป็น
บทหนึ่งที่ว่าด้วยเรื่องสมรรถนะการค้นหา ดังปรากฏในบทที่ 5 ถัดไปจากบทนี้

4.4 การพิสูจน์การลู่เข้าหาผลเฉลยของอัลกอริธึมการค้นหา

แบบตามเชิงปรับตัวหลายวิถี

การพิสูจน์การลู่เข้าหาผลเฉลยของอัลกอริธึมการค้นหาแบบตามเชิงปรับตัวหลายวิถี หรือ MATS อาศัยแนวทางการพิสูจน์ว่า ATS ย่อยทุกตัว ภายใน MATS ยังคงมีคุณสมบัติการลู่เข้าหาคำตอบวงกว้างเช่นเดียวกับ ATS แบบเดี่ยว ซึ่งได้รับการพิสูจน์คุณสมบัติการลู่เข้าหาผลเฉลยไว้แล้วเป็นอย่างดี ปรากฏอยู่ใน เดชา พวงดาวเรือง (2547) และเมื่อกระบวนการค้นหาของ MATS ดำเนินไปได้ช่วงเวลาหนึ่ง ATS ย่อยหน่วยหนึ่งที่เป็นองค์ประกอบภายในของ MATS จะเป็นผู้พบคำตอบวงกว้างที่เป็นไปตามเงื่อนไขยุติการค้นหาของ MATS ตรรกะทางคณิตศาสตร์บริสุทธิ์ร่วมกับตรรกะเชิงศึกษาสำนึก เพื่อพิสูจน์ว่า MATS มีคุณสมบัติการลู่เข้าหาคำตอบวงกว้างของหน่วยค้นหาแกน (search core) แต่สามารถทำให้มีอัตราการลู่เข้าหาผลเฉลยที่รวดเร็วกว่าของหน่วยค้นหาแกน ได้ด้วยการปรับเปลี่ยนพารามิเตอร์ของกลไกภายในส่วนจัดการหน่วยค้นหาอย่างเหมาะสม ภายใต้ นิยามและทฤษฎีบท ดังรายละเอียดต่อไปนี้

4.4.1 นิยาม

- กำหนดให้ PM คือกลไกแบ่งปริภูมิค้นหาทั้งหมดออกเป็นส่วน ๆ เท่ากับจำนวนหน่วยค้นหา เพื่อจะได้หาคำตอบที่ดีที่สุดในแต่ละปริภูมิย่อยที่ได้มาทำหน้าที่เป็นคำตอบเริ่มต้นให้กับแต่ละหน่วยค้นหา ดังรายละเอียดต่อไปนี้

Ω entire search space

Ω_l $l = 1, 2, \dots, m$ sub-search spaces

$$\bigcap_{l=1, \dots, m} \Omega_l = \phi (\text{null set})$$

$$\bigcup_{l=1, \dots, m} \Omega_l = \Omega$$

S_{lo} initial solutions of Ω_l

P_l $l = 1, 2, \dots, m$ search paths

P_l starts with S_{lo}

- กำหนดให้ SM คือกลไกลำดับการค้นหาให้ ATS ย่อยทำงานเชิงลำดับ
- กำหนดให้ DM คือกลไกยกเลิกการค้นหา

ทฤษฎีบท ก. สำหรับปัญหาการหาค่าเหมาะที่สุดใด ๆ ที่ ATS แบบเดี่ยวมีคุณสมบัติการลู่เข้าหาคำตอบวงกว้างแล้ว MATS ที่ประกอบด้วย ATS ย่อยซึ่งใช้ค่าพารามิเตอร์แบบเดียวกันกับ ATS แบบเดี่ยว ย่อมมีคุณสมบัติการลู่เข้าหาคำตอบวงกว้างสำหรับปัญหาการหาค่าเหมาะที่สุดนั้นด้วย

พิสูจน์ สมมติให้ปัญหาการหาค่าเหมาะที่สุดปัญหาหนึ่งเป็นปัญหาการหาค่าต่ำสุด ในปัญหานี้ ATS แบบเดียวมีคุณสมบัติการลู่เข้าหาคำตอบวงกว้าง ไม่ว่าจะเริ่มต้นการค้นหาจากที่ใดในปริภูมิการค้นหาดังปรากฏใน Sujitjon et al., 2006 และสามารถแสดงคุณสมบัติการลู่เข้าหาคำตอบวงกว้างของ ATS แบบเดียว ได้ดังสมการที่ (4.25)

$$\lim_{k \rightarrow l < \infty} \{ f(x_{i=1, \dots, n}^k) - f(x^*) \} \leq \varepsilon, \quad \forall x^0 \in \Omega \quad (4.25)$$

เมื่อ	Ω	คือ ปริภูมิการค้นหา
	k	คือ ลำดับที่รอบการค้นหา
	$f(\cdot)$	คือ ฟังก์ชันวัตถุประสงค์ โดยที่ $f: \mathbb{R}^n \rightarrow \mathbb{R}$
	n	คือ จำนวนมิติของคำตอบของฟังก์ชัน
	x^0	คือ คำตอบเริ่มต้นของ ATS แบบเดียว
	$x_{i=1, \dots, n}^k$	คือ คำตอบจำนวน n มิติ ณ รอบการค้นหาที่ k
	x^*	คือ คำตอบที่ให้ค่าฟังก์ชันวัตถุประสงค์น้อยที่สุด
	ε	คือ ค่าความคลาดเคลื่อนระหว่างค่าฟังก์ชันวัตถุประสงค์ ณ รอบการค้นหาที่ k กับค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดโดยทั่วไปค่า ε จะมีค่าเท่ากับศูนย์แต่ทางปฏิบัติกำหนดให้มีค่าน้อย ๆ เช่น 10^{-9} - 10^{-6} เป็นต้น

MATS จะมีคุณสมบัติการลู่เข้าหาคำตอบวงกว้างด้วย เนื่องจาก MATS เริ่มต้นด้วยการเรียกใช้กลไกแบ่งปริภูมิหรือ PM เพื่อทำหน้าที่แบ่งปริภูมิการค้นหารวมออกเป็นปริภูมิการค้นหาย่อย เพื่อคัดเลือกคำตอบที่ดีในแต่ละปริภูมิย่อย มาเป็นคำตอบเริ่มต้นให้กับแต่ละ ATS ย่อย เมื่อแต่ละ ATS ย่อยได้คำตอบเริ่มต้นเรียบร้อยแล้วจะเข้าสู่ช่วงรอบการค้นหาหลักอย่างอิสระแล้วคืนค่าปริภูมิการค้นหารวมกลับมาให้ทุกหน่วยค้นหา นั่นคือในบริบทของ MATS ภายใต้กระบวนการงาน PM อาจกล่าวได้ว่า ทุก ATS ย่อยสามารถเข้าถึงได้ทุกจุดของปริภูมิการค้นหารวม และสามารถแสดงคุณสมบัติการลู่เข้าหาคำตอบวงกว้างของ MATS ได้ดังสมการที่ (4.26)

$$\lim_{k \rightarrow l < \infty} \left\{ \begin{aligned} & \left[f_1(x_{1,i=1, \dots, n}^k) - f(x^*) \right] \vee \left[f_2(x_{2,i=1, \dots, n}^k) - f(x^*) \right] \cdots \\ & \vee \left[f_m(x_{m,i=1, \dots, n}^k) - f(x^*) \right] \end{aligned} \right\} \leq \varepsilon$$

$$\forall x_j^0 \in \Omega \quad (4.26)$$

เมื่อ	Ω	คือ ปริภูมิการค้นหา
$\vee$		คือ ตัวดำเนินการเชิงตรรกะ แบบหรือ (OR)
k		คือ ลำดับที่รอบการค้นหา
n		คือ จำนวนมิติของคำตอบของฟังก์ชัน
f_1		คือ ฟังก์ชันวัตถุประสงค์ของ ATSย่อยที่ 1 โดยที่ $f_1 : \mathbb{R}^n \rightarrow \mathbb{R}$
f_2		คือ ฟังก์ชันวัตถุประสงค์ของ ATSย่อยที่ 2 โดยที่ $f_2 : \mathbb{R}^n \rightarrow \mathbb{R}$
f_m		คือ ฟังก์ชันวัตถุประสงค์ของ ATSย่อยที่ m โดยที่ $f_m : \mathbb{R}^n \rightarrow \mathbb{R}$
x_j^0		คือ คำตอบเริ่มต้นจำนวน n มิติ ของ ATSย่อยที่ j
$x_{1,i=1,\dots,n}^k$		คือ คำตอบจำนวน n มิติ ณ รอบการค้นหาที่ k ของ ATSย่อยที่ 1
$x_{2,i=1,\dots,n}^k$		คือ คำตอบจำนวน n มิติ ณ รอบการค้นหาที่ k ของ ATSย่อยที่ 2
$x_{m,i=1,\dots,n}^k$		คือ คำตอบจำนวน n มิติ ณ รอบการค้นหาที่ k ของ ATSย่อยที่ m
x^*		คือ คำตอบที่ให้ค่าฟังก์ชันวัตถุประสงค์น้อยที่สุด
ε		คือ ค่าความคลาดเคลื่อนระหว่างค่าฟังก์ชันวัตถุประสงค์ ณ รอบการค้นหาที่ n กับค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดโดยทั่วไปค่า ε จะมีค่าเท่ากับศูนย์ แต่ทางปฏิบัติกำหนดให้มีค่าน้อย ๆ เช่น $10^{-9} - 10^{-6}$ เป็นต้น

เกณฑ์ยุติการค้นหาที่ถูกประยุกต์ในกระบวนการค้นหาตามสมการที่ (4.26) มีอยู่ 2 เกณฑ์ ที่การค้นหายุติด้วยเกณฑ์หนึ่งเกณฑ์ใดของ 2 เกณฑ์ ดังกล่าวได้แก่ (1) ค่า ε อยู่ในเกณฑ์ที่ยอมรับได้ ไม่ว่าจะเป็นของ f_1 หรือ f_2 หรือ f_m ใด ๆ ของ MATS (2) ค่า k ถึงขอบเขตบนที่ตั้งไว้ เช่น จำนวนรอบการค้นหาสูงสุด $k_{\max} = 10^4 \approx \infty$

ตามปกติ ATS แบบเดี่ยว เมื่อทำการค้นหาเพียงหน่วยเดียวจะมีความต่อเนื่องในการขึ้นรอบการค้นหาใหม่เมื่อเสร็จสิ้นรอบการค้นหาก่อนหน้า แต่เมื่อกระทำการภายใต้การกำกับของ MATS จะถูกขัดจังหวะ (interrupt) และสถานะทางการค้นหาของ ATS ย่อย จะถูกจดจำไว้ในหน่วยความจำ ซึ่งมีความหมายว่าต้องรอให้ ATS ย่อยหน่วยอื่นในวงรอบการค้นหาของ MATS ดำเนินการเสร็จสิ้นในรอบการค้นหานี้ด้วย ซึ่งกระบวนการที่ต้องรอนี้ เป็นไปตามกลไกลำดับการค้นหา หรือ SM เมื่อกระบวนการค้นหาคำเนินการมาได้ระยะเวลาหนึ่งจะมีการตรวจสอบสิทธิ์ของการอยู่ในวงรอบการค้นหาหลักด้วยการเรียกใช้กลไกยกเลิกการค้นหา หรือ DM ทำหน้าที่ประเมินความเหมาะสมให้กับแต่ละ ATS ย่อย พร้อมทั้งทำการยกเลิกสิทธิ์การอยู่ในวงรอบการค้นหาลึกของ ATS ย่อย ที่ถูกประเมินว่ามีสมรรถนะต่ำ ATS ย่อยจำนวนหนึ่งที่ไม่ถูกตัดสิทธิ์การทำงานจะได้รับการคืนสิทธิ์ดำเนินการค้นหาต่อจากสถานะเดิมที่ได้ถูกจดจำไว้ทุกประการและจะทำงานบนปริภูมิการค้นหาเดิมที่ไม่ถูกเปลี่ยนแปลงเมื่อครั้งถูกขัดจังหวะ ผลของ DM จะทำให้

จำนวนของ ATS ย่อย ในวงรอบการค้นหาลักษณะลดลง ดังนั้น MATS จึงมิได้เข้าไปมีผลกระทบต่อ การมีอยู่ของคุณสมบัติการลู่เข้าหาคำตอบวงกว้าง อาจพิจารณาได้ว่าเมื่อ MATS เริ่มต้นปฏิบัติงาน กระบวนการของ MATS อาจแทนด้วยความสัมพันธ์ (4.26) ที่ถูกดัดแปลงเป็น

$$\lim_{k \rightarrow q < l < \infty} \left\{ \begin{array}{l} [f_1(x_{1,i=1,\dots,n}^k) - f(x^*)] \vee [f_2(x_{2,i=1,\dots,n}^k) - f(x^*)] \dots \\ \vee [f_m(x_{m,i=1,\dots,n}^k) - f(x^*)] \end{array} \right\} \leq \delta > \varepsilon$$

$$\forall x_j^0 \in \Omega \text{ เมื่อ } j \in (\mathbb{N} - \{0,1\}) \quad (4.27)$$

มีความหมายว่าเมื่อ $k \rightarrow q < l$, $l < k_{\max}$ กระบวนการ DM ทำงาน โดย MATS ยังไม่พบคำตอบวงกว้าง เมื่อกระบวนการเกิดขึ้นซ้ำเป็นจำนวนครั้งไฟไนต์ ความสัมพันธ์ (4.27) จึงลดรูปเป็น (4.28) ซึ่งให้ความหมายทำนองเดียวกันกับ (4.25) นั่นคือมี ATS ย่อยหน่วยเดียว ที่คงเหลือในกระบวนการจนค้นพบคำตอบวงกว้าง

$$\lim_{k \rightarrow l < \infty} \{ f_p(x_{p,i=1,\dots,n}^k) - f(x^*) \} \leq \varepsilon, \forall x_j^0 \in \Omega, 1 \leq p \leq m \quad (4.28)$$

สามารถกล่าวได้อีกนัยหนึ่งว่า ถ้าทุก ATS ย่อยภายใน MATS มีคุณสมบัติการ ลู่เข้าหาคำตอบวงกว้างแล้ว หนึ่งใน ATS ย่อย ที่ MATS ดำเนินงานภายใต้กระบวนการที่มี DM เลือกให้ทำหน้าที่ค้นหาคำตอบวงกว้างต่อไปจนเสร็จสิ้นกระบวนการค้นหา ย่อมคงความมี คุณสมบัติการลู่เข้าหาคำตอบวงกว้างนั้นอยู่ต่อไป ดังนั้นจึงสรุปได้ว่า MATS ย่อมมีคุณสมบัติการ ลู่เข้าหาคำตอบวงกว้างด้วยตามคุณสมบัติของทุก ๆ ATS การพิสูจน์เสร็จสิ้น

ทฤษฎีบท ข. การคัดสรรพารามิเตอร์ของ MATS อย่างเหมาะสม สามารถทำ ให้ MATS ค้นพบคำตอบวงกว้างได้ภายในเวลาที่สั้นกว่าการค้นหาโดย ATS แบบเดี่ยว

พิสูจน์ กำหนดให้ค่าเฉลี่ยเวลาการค้นหาของ ATS แบบเดี่ยว และของ MATS แทนด้วย T_{ATS} และ T_{MATS} ตามลำดับ แสดงความสัมพันธ์ตามสมการที่ (4.29) และ (4.30) ข้างล่างนี้

$$\lambda^* = T_{ATS} = Count_{ATS} \cdot Time(x) \quad (4.29)$$

$$\lambda = T_{MATS} = Count_{MATS} \cdot Time(x) \quad (4.30)$$

โดยที่ $Time(x)$ คือเวลาเฉลี่ยที่ใช้ในการหาคำตอบแต่ละรอบการค้นหา ซึ่งกำหนดให้มีความเท่ากัน ทั้งของ ATS แบบเดี่ยว และของATS ย่อยภายใน MATS ส่วน $Count_{ATS}$ และ $Count_{MATS}^*$ คือค่าเฉลี่ยจำนวนรอบการค้นหาที่ใช้ในการค้นหาคำตอบวงกว้างของ ATS แบบเดี่ยวและของ MATS ตามลำดับ ส่วนค่า $Count_{MATS}$ คือค่าจำนวนรอบการค้นหาของ MATS ที่สมมูลกับของ ATS แบบเดี่ยว สามารถคำนวณได้ตามสมการที่ (4.31)

$$Count_{MATS} = n \times e \times d \quad (4.31)$$

- โดยที่ n คือ เมตริกซ์จำนวนหน่วยค้นหาแกน เมื่อ $n = [n_0 \ n_1 \ n_2 \ \dots \ n_{k-1} \ n_k]$, k คือจำนวน DM สูงสุดที่ตั้งไว้ $k \in \mathbb{N}^+$ สำหรับ MATS ถ้า $k = 0$ หมายถึง ATS นั้นเอง) ปกติ $n_0 = 1$ และ $N = \sum_{i=0}^k n_i$ เมื่อ N คือจำนวน ATS ย่อยทั้งหมด
- n_1 = จำนวนหน่วยค้นหาแกนที่ถูกยกเลิกหลังจากการเรียกใช้ DM1
- n_2 = จำนวนหน่วยค้นหาแกนที่ถูกยกเลิกหลังจากการเรียกใช้ DM2
- n_k = จำนวนหน่วยค้นหาแกนที่ถูกยกเลิกหลังจากการเรียกใช้ DMk
- e คือ เมตริกซ์แสดงการเรียกใช้ DM ต่าง ๆ (*enable matrix*) มีลักษณะเป็นเมตริกซ์ทแยงมุม โดยที่สมาชิก $e \in \{0,1\}$ จะมีค่าเป็น 1 เมื่อมีการเรียกใช้ DM นั้นเท่านั้น $e = [e_0 \ e_1 \ e_2 \ \dots \ e_{k-1} \ e_k]$ และ ปกติ $e_0 = 1$
- d คือ เมตริกซ์รอบการค้นหาที่มีการเรียกใช้ DM ต่าง ๆ มีลักษณะเป็นเมตริกคอมลัมน์ $d = [d_0 \ d_1 \ d_2 \ \dots \ d_{k-1} \ d_k]^T$ ปกติ $d_0 = Count_{MATS}^*$

ตัวอย่าง กรณี $N = 16$ การคำนวณเมื่อมีการเรียกใช้ DM ทั้งหมดคือ 3 ครั้ง ลดจำนวน ATS ย่อยจาก 16 เหลือ 1

$$Count_{MATS} = [1 \ 8 \ 4 \ 2 \ 1] \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Count_{MATS}^* \\ 10 \\ 20 \\ 30 \\ 40 \end{bmatrix} \quad (4.32)$$

$$= Count_{MATS}^* + 8 \times 10 + 4 \times 20 + 2 \times 30 + 1 \times 40 = Count_{MATS}^* + 260$$

กรณีที่มีการเรียกใช้ DM เพียง 1 ครั้งลดจำนวน ATS ย่อยจาก 16 เหลือ 8 แล้วพบคำตอบวงกว้าง ซึ่งมี $e_1 = 1, e_2 = e_3 = e_4 = 0$ ดังนั้น

$$\begin{aligned} \text{Count}_{\text{MATS}} &= [8 \ 8 \ 0 \ 0 \ 0] \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \text{Count}_{\text{MATS}}^* \\ 10 \\ 20 \\ 30 \\ 40 \end{bmatrix} \\ &= 8 \times \text{Count}_{\text{MATS}}^* + 8 \times 10 = 8\text{Count}_{\text{MATS}}^* + 80 \end{aligned} \quad (4.33)$$

ดังนั้นจึงสามารถสรุปได้ว่าการคัดเลือกเมตริกซ์ n และ d อย่างเหมาะสมภายใต้เงื่อนไข $\text{Count}_{\text{MATS}}^* < \frac{180}{7}$ จะทำให้ $\lambda < \lambda^*$ ซึ่งหมายความว่า MATS สามารถหาคำตอบวงกว้างและพบคำตอบนั้นได้รวดเร็วกว่า ATS แบบเดียว การพิสูจน์เสร็จสิ้น

4.5 สรุป

บทนี้นำเสนอรายละเอียดของ MATS โดยแบ่งออกเป็น 3 หัวข้อ ได้แก่ หัวข้อที่หนึ่งอธิบายถึงหลักการของ MATS ตลอดจนสามองค์ประกอบของ MATS คือกลไกแบ่งปริภูมิการค้นหา (PM) กลไกลำดับการค้นหา (SM) และ กลไกยกเลิกการค้นหา (DM) ส่วนหัวข้อที่สองเป็นการวิเคราะห์อัลกอริทึม MATS และหัวข้อที่สามเสนอบทพิสูจน์คุณสมบัติการหาคำตอบวงกว้างของ MATS ซึ่งประกอบด้วย 2 ทฤษฎีบท ที่จะแสดงให้เห็นว่า MATS มีคุณสมบัติการหาคำตอบวงกว้างและมีศักยภาพมากพอที่จะดำเนินการค้นหาได้รวดเร็วกว่า ATS แบบเดียว หากได้รับการปรับตั้งค่าพารามิเตอร์อย่างเหมาะสม บทถัดไปจะกล่าวถึงการทดสอบสมรรถนะของ MATS กับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวน 6 ปัญหา โดยเปรียบเทียบกับผลของ ATS แบบเดียว

บทที่ 5

การประเมินสมรรถนะ

5.1 กล่าวนำ

การวิเคราะห์สมรรถนะการค้นหาคำตอบของการค้นหาแบบตาบอดเชิงปรับตัวหลายวิธีดำเนินการทดสอบกับปัญหาการหาเหมาะที่สุดเชิงฟังก์ชัน จำนวนหกฟังก์ชัน ได้แก่ ฟังก์ชันโบฮาเชฟสกี (Bohachevsky's function : BF) ฟังก์ชันราสตริกิน (Rastrigin's function : RF) ฟังก์ชันชีเคิลฟอกซ์โฮล (Shekel's foxholes function : SF) ฟังก์ชันชูเบิร์ท (Shubert's function : ShF) ฟังก์ชันชวีเฟลแบบที่หนึ่ง (Schwefel's function type I : ScF1) และ ฟังก์ชันชวีเฟลแบบที่สอง (Schwefel function type II : ScF2) การค้นหาคำตอบกว้างของทั้งปัญหาที่มีคำตอบกว้างเดียวและหลายคำตอบกว้าง ซึ่งมีคำตอบเฉพาะถิ่นจำนวนมากเป็นอุปสรรคสำหรับการค้นหารายละเอียดคุณสมบัติของฟังก์ชันทดสอบแสดงไว้แล้วในบทที่ 3 ตารางที่ 3.3 ส่วนพารามิเตอร์ของอัลกอริธึมการค้นหาแบบตาบอดเชิงปรับตัวทั้งแบบเส้นทางเดียว (ATS) และแบบหลายเส้นทางการค้นหา (MATS) จะใช้การตั้งค่าไว้เหมือนกันเพื่อความเสมอภาคในการเปรียบเทียบสมรรถนะระหว่าง ATS แบบเดียว กับ MATS ซึ่งรายละเอียดการตั้งค่าพารามิเตอร์อธิบายไว้ในหัวข้อ 5.2 ส่วนหัวข้อ 5.3 กล่าวถึงผลการทดสอบ หัวข้อ 5.4 เป็นการวิเคราะห์ผลการทดสอบพร้อมแสดงกราฟการลู่เข้าหาผลเฉลย และ หัวข้อ 5.5 เป็นสรุป

5.2 การตั้งค่าพารามิเตอร์สำหรับ MATS

พารามิเตอร์ของ MATS หมายถึง พารามิเตอร์ต่าง ๆ ของกลไกภายในของ MATS ได้แก่ PM SM และ DM วิชานิพนธ์ฉบับนี้เน้นศึกษาผลของพารามิเตอร์ของ PM และ DM ต่อสมรรถนะการค้นหาคำตอบของ MATS ด้วยพารามิเตอร์ที่แสดงไว้ในตารางที่ 5.1 และ 5.2 ตามลำดับ ส่วน SM คงค่าจำนวนรอบการค้นหาของ ATS ย่อย ไว้ที่หนึ่งรอบตลอดการค้นหาของ MATS

5.2.1 กลไกแบ่งปริภูมิการค้นหา (PM)

การตั้งค่าพารามิเตอร์ของ PM สำหรับปัญหาการหาเหมาะที่สุดเชิงฟังก์ชันที่ใช้ทดสอบนี้ อิงความเท่ากันของขนาดพื้นที่เชิงเรขาคณิตเป็นสำคัญ ซึ่งสามารถดำเนินการได้สะดวก ร่วมกับแบบขาไปซ้ายและแบบบนลงล่างประกอบกันเป็นต้น ตัวอย่างการแบ่งปริภูมิการค้นหาพร้อมวิธีการตั้งชื่อหน่วยค้นหา สำหรับปัญหาแบบสองมิติรูปสี่เหลี่ยม กรณีสี่เหลี่ยม 2 4 8 และ 16 หน่วยค้นหา ดังแสดงในรูปที่ 4.3 ส่วนกรณีของ 32 และ 64 หน่วยค้นหา ดังแสดงในรูปที่ 5.1 ปริภูมิ

การค้นหาย่อยที่แบ่งได้จาก PM ดังแสดงในตารางที่ 5.1 ฟังก์ชัน BF และ ฟังก์ชัน RF มีปฏิสัมพันธ์ที่เหมือนกันจึงให้ผลการแบ่งปฏิสัมพันธ์ค้นหาของ PM เหมือนกันด้วย ส่วนอีกสี่ฟังก์ชันทดสอบที่เหลือมีปฏิสัมพันธ์ค้นหาที่แตกต่างกันจึงให้ผลของ PM แตกต่างกันไปด้วย ตัวอย่างผลการแบ่งปฏิสัมพันธ์ค้นหาของ RF กรณี MATS แบบสองเส้นทางการค้นหา หรือ MATS#2 ด้วยปฏิสัมพันธ์ค้นหา $[2 \ 2; -2 \ -2]$ นั้นได้ ปฏิสัมพันธ์ค้นหาย่อยที่ 1 หรือ #1 คือ $[2 \ 2; 0 \ -2]$ ซึ่งหมายความว่า มีขอบเขตของปฏิสัมพันธ์ค้นหาตามแนวแกน x อยู่ในช่วง $[0, 2]$ และมีขอบเขตของปฏิสัมพันธ์ค้นหาตามแนวแกน y อยู่ในช่วง $[-2, 2]$ ส่วนของปฏิสัมพันธ์ค้นหาย่อยที่ 2 หรือ #2 คือ $[0 \ 2; -2 \ -2]$ มีความหมายทำนองเดียวกับปฏิสัมพันธ์ค้นหาย่อยที่ 1 ก็คือขอบเขตของปฏิสัมพันธ์ค้นหาตามแนวแกน x อยู่ในช่วง $[-2, 0]$ และมีขอบเขตของปฏิสัมพันธ์ค้นหาตามแนวแกน y อยู่ในช่วง $[-2, 2]$ ดังนั้น เป็นต้น สำหรับกรณีของ ScF1 กับ ScF2 พื้นผิวถูกสร้างขึ้นด้วยฟังก์ชันเดียวกัน แต่มีความแตกต่างกันที่ขนาดของปฏิสัมพันธ์ค้นหาโดยของ ScF1 จะมีขนาดใหญ่กว่าของ ScF2 ถึง 100 เท่า ปฏิสัมพันธ์ค้นหาของ ScF1 คือ $[500 \ 500; -500 \ -500]$ ส่วนของ ScF2 คือ $[500 \ 500; 400 \ 400]$ การเลือก ScF1 และ ScF2 มาเป็นฟังก์ชันทดสอบก็เพื่อศึกษาผลของขนาดปฏิสัมพันธ์ค้นหาต่อสมรรถนะการค้นหา รวมถึงธรรมชาติของพื้นผิวของฟังก์ชันต่อความสามารถในการค้นหาด้วย เมื่อได้ปฏิสัมพันธ์ค้นหาย่อยเรียบร้อยแล้วจะเข้าสู่กระบวนการคัดสรรตัวแทนคำตอบจากปฏิสัมพันธ์ย่อยสำหรับงานวิจัยนี้ใช้

#8	#7	#6	#5	#4	#3	#2	#1
#9	#10	#11	#12	#13	#14	#15	#16
#24	#23	#22	#21	#20	#19	#18	#17
#25	#26	#27	#28	#29	#30	#31	#32

(ก) 32 เส้นทาง

#8	#7	#6	#5	#4	#3	#2	#1
#9	#10	#11	#12	#13	#14	#15	#16
#24	#23	#22	#21	#20	#19	#18	#17
#25	#26	#27	#28	#29	#30	#31	#32
#40	#39	#38	#37	#36	#35	#34	#33
#41	#42	#43	#44	#45	#46	#47	#48
#56	#55	#54	#53	#52	#51	#50	#49
#57	#58	#59	#60	#61	#62	#63	#64

(ข) 64 เส้นทาง

รูปที่ 5.1 ตัวอย่าง PM พร้อมการตั้งชื่อหน่วยค้นหา

5.2.2 กลไกลำดับการค้นหา (SM)

พารามิเตอร์ภายใน SM ของ MATS คือ จำนวนรอบที่ ATS ย่อย แต่ละหน่วยค้นหา จะดำเนินการได้ในแต่ละรอบการค้นหาของ MATS การปรับตั้งสามารถแบ่งตามลักษณะ การเปลี่ยนแปลงได้สองแบบ ได้แก่ แบบที่หนึ่งแบบสถิต คือจำนวนรอบของ ATS ย่อย คงที่ตลอด การค้นหาของ MATS ซึ่งเป็นรูปแบบที่เลือกมาใช้งาน MATS เบื้องต้นโดยกำหนดให้จำนวนรอบ การค้นหาของ ATS ย่อย แต่ละหน่วยค้นหาทำงานคนละหนึ่งรอบในแต่ละรอบการค้นหาของ MATS จนกระทั่ง MATS ยุติการค้นหา และแบบที่สองแบบพลวัต คือจำนวนรอบของ ATS ย่อย สามารถเปลี่ยนแปลงได้ตามสถานะการณ์ของ ATS ย่อย แต่ละหน่วยค้นหา แบบที่สองนี้ยังอยู่ใน ขั้นตอนการศึกษาวิจัยจึงมีได้นำเสนอไว้ในที่นี้

5.2.3 กลไกยกเลิกการค้นหา (DM)

พารามิเตอร์ DM ของ MATS มีหน้าที่ลดจำนวนหน่วยค้นหา ATS ย่อย ในกำกับ ของ MATS ให้เหลือน้อยที่สุดแต่ยังพยายามรักษาโอกาสการค้นหาคำตอบวงกว้างให้พบได้ในเวลา รวดเร็วที่สุด วิธีการลดจำนวนหน่วยค้นหาคำเนินการ โดยการยกเลิกการทำงานของบาง ATS ย่อย ที่ถูกประเมินแล้วว่าประสิทธิภาพการค้นหาคำตอบวงกว้างต่ำกว่าออกจากวงจรรอบการค้นหาของ MATS สำหรับเกณฑ์ที่ใช้ในการประเมินประสิทธิภาพการค้นหาของหน่วยค้นหา เบื้องต้นงานวิจัย ฉบับนี้พิจารณาที่ค่าฟังก์ชันวัตถุประสงค์ที่ดีที่สุดของแต่ละหน่วยค้นหา ATS ย่อย ค้นพบ ณ รอบการ ค้นหาที่มีการเรียกใช้ DM จึงเป็นวิธีการประเมินคุณภาพล่าสุด (recency) จำนวนครั้งการเรียกใช้ DM จะขึ้นกับจำนวนหน่วยค้นหา ATS ย่อย ในกำกับของ MATS และรูปแบบการลดจำนวนหน่วย ค้นหา สามารถแบ่งได้เป็นสองลักษณะใหญ่ ๆ ตามอัตราการลดลง ได้แก่ รูปแบบการลดจำนวน หน่วยค้นหาอย่างค่อยเป็นค่อยไป นั่นคือจำนวนหน่วยค้นหาตั้งแต่เริ่มต้นจนเหลือเพียงหนึ่งค้นหา เดียวจะมีการเรียกใช้ DM มากกว่าหนึ่งครั้งยกเว้นกรณีของ MATS#2 ยังจัดอยู่ในรูปแบบการลด จำนวนหน่วยค้นหาอย่างค่อยเป็นค่อยไปด้วย เนื่องจากมีเพียงสองหน่วยค้นหาจึงเรียกใช้ DM เพียงครั้งเดียวเพื่อจะลดจำนวนหน่วยค้นหาให้เหลือหนึ่งค้นหา ตัวอย่างของรูปแบบนี้ตามตารางที่ 5.2 สำหรับฟังก์ชัน BF ได้แก่ MATS#2 #4 #8 #16 และ #32 ฟังก์ชัน RF ได้แก่ MATS#2 #8 #16 และ #32 ฟังก์ชัน SF ได้แก่ MATS#2 #4 #8 และ #16 ฟังก์ชัน ShF ได้แก่ MATS#2 ฟังก์ชัน ScF1 ได้แก่ MATS#2 #4 และ #8 และ ฟังก์ชัน ScF2 ได้แก่ MATS#2 กรณีพิเศษของรูปแบบ การลดจำนวนหน่วยค้นหาอย่างค่อยเป็นค่อยไปคือมีการเรียกใช้ DM มากกว่าหนึ่งครั้งและจำนวน หน่วยค้นหาที่ลดลงก็ไม่จำเป็นต้องเป็นครั้งหนึ่งรวมทั้งจำนวนหน่วยค้นหาที่เหลือสุดท้ายก็ไม่ จำเป็นต้องเหลือหนึ่งหน่วยค้นหาที่ได้อย่างกรณีของ MATS#8 และ #16 ที่ทำการค้นหามบนฟังก์ชัน ScF2 เป็นต้น ส่วนอีกหนึ่งรูปแบบได้แก่รูปแบบการลดจำนวนหน่วยค้นหาอย่างทันทีทันใด ซึ่งม การเรียก DM เพียงครั้งเดียวแล้วลดจำนวนหน่วยค้นหาที่มีทั้งหมดตั้งแต่สามหน่วยค้นหาขึ้นไปให้

เหลือเพียงหน่วยค้นหาเดียว ตัวอย่างตามตารางที่ 5.2 ที่จัดอยู่ในกลุ่มที่มีการลดจำนวนหน่วยค้นหาอย่างทันทีทันใดนี้ สำหรับฟังก์ชัน BF ได้แก่ MATS#64 ฟังก์ชัน RF ได้แก่ MATS#4 และ #64 ฟังก์ชัน SF ได้แก่ MATS#32 และ #64 ฟังก์ชัน ShF ได้แก่ MATS#4 และ #64 ฟังก์ชัน ScF1 ได้แก่ MATS#32 และ #64 และ ฟังก์ชัน ScF2 ได้แก่ MATS#4 #8 #16 #32 และ #64 กรณียกเว้นของรูปแบบการลดจำนวนหน่วยค้นหาอย่างทันทีทันใด แต่จำนวนหน่วยค้นหาที่เหลือสุดท้ายมีมากกว่าหนึ่งหน่วยค้นหาอย่างกรณีของ MATS#8 MATS#16 MATS#32 และ #64 ที่ทำการค้นหามันฟังก์ชัน ShF เป็นต้น จึงมีข้อสังเกตว่า ปัญหาที่ใช้วิธีค้นหาคำเนินการหาผลเฉลยนั้น มีความไม่เป็นเชิงเส้นและมีความซับซ้อนมาก การจัดตั้งพารามิเตอร์ต่าง ๆ สำหรับการค้นหามีประสิทธิภาพจึงต้องมีความรู้ความเข้าใจในตัวปัญหาในระดับหนึ่ง ซึ่งอาจต้องการกระบวนการค้นหาขั้นต้น (pre - search หรือ trial search) เพื่อศึกษาปัญหา

ตารางที่ 5.2 พารามิเตอร์ของ DM ใน MATS ที่ใช้ทดสอบ

ฟังก์ชันทดสอบ		รอบการค้นหาที่เรียกใช้ DM จำนวนหน่วยค้นหาที่ถูกยกเลิกและจำนวนหน่วยค้นหาที่เหลือหลังเรียกใช้ DM																	
		MATS																	
		#2		#4		#8			#16				#32					#64	
	1 st	1 st	2 nd	1 st	2 nd	3 rd	1 st	2 nd	3 rd	4 th	1 st	2 nd	3 rd	4 th	5 th	1 st	2 nd	3 rd	
BF	รอบการค้นหาที่	5	5	10	5	10	15	5	10	15	20	5	10	15	20	25	1	3	5
	จำนวนที่ถูกยกเลิก	1	2	1	4	2	1	8	4	2	1	16	8	4	2	1	32	16	15
	จำนวนที่เหลือ	1	2	1	4	2	1	8	4	2	1	16	8	4	2	1	32	16	1
RF	รอบการค้นหาที่	5	50	-	2	4	6	2	4	6	8	2	4	6	8	10	1	-	-
	จำนวนที่ถูกยกเลิก	1	3	-	4	2	1	8	4	2	1	16	8	4	2	1	63	-	-
	จำนวนที่เหลือ	1	1	-	4	2	1	8	4	2	1	16	8	4	2	1	1	-	-
SF	รอบการค้นหาที่	5	2	4	2	4	6	2	4	6	8	1	-	-	-	-	1	-	-
	จำนวนที่ถูกยกเลิก	1	2	1	4	2	1	8	4	2	1	31	-	-	-	-	63	-	-
	จำนวนที่เหลือ	1	2	1	4	2	1	8	4	2	1	1	-	-	-	-	1	-	-
ShF	รอบการค้นหาที่	1	1	-	1	3	-	3	5	-	-	1	-	-	-	-	1	-	-
	จำนวนที่ถูกยกเลิก	1	3	-	2	2	-	8	4	-	-	28	-	-	-	-	60	-	-
	จำนวนที่เหลือ	1	1	-	6	4	-	8	4	-	-	4	-	-	-	-	4	-	-
ScF1	รอบการค้นหาที่	5	1	3	1	3	5	1	3	5	-	1	-	-	-	-	1	-	-
	จำนวนที่ถูกยกเลิก	1	2	1	4	2	1	8	4	3	-	31	-	-	-	-	63	-	-
	จำนวนที่เหลือ	1	2	1	4	2	1	8	4	1	-	1	-	-	-	-	1	-	-
ScF2	รอบการค้นหาที่	1	1	-	1	-	-	1	-	-	-	1	-	-	-	-	1	-	-
	จำนวนที่ถูกยกเลิก	1	3	-	7	-	-	15	-	-	-	31	-	-	-	-	63	-	-
	จำนวนที่เหลือ	1	1	-	1	-	-	1	-	-	-	1	-	-	-	-	1	-	-

ตารางที่ 5.3 จำนวนรอบการค้นหาเฉลี่ยที่พบคำตอบวงกว้างจากทดสอบของ ATS แบบเดี่ยว

	ฟังก์ชันทดสอบ					
	BF	RF	SF	ShF	ScF1	ScF2
จำนวนรอบการค้นหาเฉลี่ย ของ ATSแบบเดี่ยว (50 ครั้ง)	556.06	759.80	108.22	342.54	336.24	31.24

พิจารณาตารางที่ 5.2 ฟังก์ชัน BF แสดงด้วยข้อมูลแถวที่ 1 - 3 โดยแถวที่ 1 คือรอบการคำนวณที่มีการเรียกใช้ DM แถวที่ 2 คือจำนวนหน่วยค้นหาที่ถูกยกเลิกการค้นหาเมื่อมีการเรียกใช้ DM และแถวที่ 3 คือจำนวนหน่วยค้นหาที่เหลือหลังการเรียกใช้ DM สำหรับ MATS#2 มีการเรียกใช้ DM เพียงหนึ่งครั้ง แสดงด้วยข้อมูลในคอลัมน์ที่ 1 มีการเรียกใช้ DM ในรอบการค้นหาที่ 5 โดยยกเลิกการค้นหาไปหนึ่งหน่วยค้นหา MATS#2 จึงเหลือเพียงหนึ่งหน่วยค้นหาหลังการเรียกใช้ DM ครั้งนี้ สำหรับ MATS#4 แสดงด้วยข้อมูลในคอลัมน์ที่ 2 - 3 เนื่องจากมีการเรียกใช้ DM จำนวนสองครั้ง คือรอบการค้นหาที่ 5 และ 10 ตามลำดับ โดย DM ครั้งที่หนึ่งทำการยกเลิกการค้นหาไปสองหน่วยค้นหา เหลืออีกสองหน่วยค้นหา พอมีการเรียกใช้ DM ครั้งที่สองจึงยกเลิกการค้นหาไปอีกหนึ่งหน่วยค้นหา ทำให้ MATS#4 เหลือเพียงหนึ่งหน่วยค้นหาหลังการเรียกใช้ DM ทั้งสองครั้ง ส่วน MATS#8 แสดงด้วยข้อมูลในคอลัมน์ที่ 4 - 6 เนื่องจากมีการเรียกใช้ DM จำนวนสามครั้ง คือรอบการค้นหาที่ 5 10 และ 15 ตามลำดับ โดย DM ครั้งที่หนึ่งทำการยกเลิกการค้นหาไปสี่หน่วยค้นหา เหลืออีกสี่หน่วยค้นหา การเรียกใช้ DM ครั้งที่สองทำการยกเลิกการค้นหาไปอีกสองหน่วยค้นหา ทำให้ MATS#8 เหลือเพียงสองหน่วยค้นหา และหลังการเรียกใช้ DM ครั้งที่สามซึ่งเป็นครั้งสุดท้ายทำการลดลงหนึ่งหน่วยค้นหาจึงเหลือเพียงหนึ่งหน่วยค้นหาใน MATS#8 ที่จะทำการค้นหาต่อไปจนเงื่อนไขยุติการค้นหาถูกพบ ผู้อ่านสามารถพิจารณาข้อมูลของ MATS#16 MATS#32 และ MATS#64 สำหรับฟังก์ชัน BF ได้ทำนองเดียวกันนี้

ฟังก์ชัน RF แสดงด้วยข้อมูลแถวที่ 4 - 6 สำหรับ MATS#2 มีการเรียกใช้ DM เพียงหนึ่งครั้งเช่นเดียวกับของฟังก์ชัน BF แสดงด้วยข้อมูลในคอลัมน์ที่ 1 เช่นกัน มีการเรียกใช้ DM ในรอบการค้นหาที่ 5 โดยยกเลิกการค้นหาไปหนึ่งหน่วยค้นหา MATS#2 จึงเหลือเพียงหนึ่งหน่วยค้นหาหลังการเรียกใช้ DM ครั้งนี้ สำหรับ MATS#4 จากข้อมูลในคอลัมน์ที่ 2 เนื่องจากมีการเรียกใช้ DM เพียงหนึ่งครั้งในรอบการค้นหาที่ 50 โดย DM ทำการยกเลิกการค้นหาไปสามหน่วยค้นหาทำให้ MATS#4 เหลือเพียงหนึ่งหน่วยค้นหาหลังการเรียกใช้ DM หนึ่งครั้ง ส่วน MATS#8 จากข้อมูลในคอลัมน์ที่ 4 - 6 เนื่องจากมีการเรียกใช้ DM จำนวนสามครั้ง คือรอบการค้นหา

ที่ 2 4 และ 6 ตามลำดับ โดย DM ครั้งที่หนึ่งทำการยกเลิกการค้นหาไปสี่หน่วยค้นหา เหลืออีกสี่หน่วยค้นหา การเรียกใช้ DM ครั้งที่สองทำการยกเลิกการค้นหาไปอีกสองหน่วยค้นหา ทำให้ MATS#8 เหลือเพียงสองหน่วยค้นหา และหลังการเรียกใช้ DM ครั้งที่สามซึ่งเป็นครั้งสุดท้ายทำการลดลงหนึ่งหน่วยค้นหาจึงเหลือเพียงหนึ่งหน่วยค้นหาใน MATS#8 ที่จะทำการค้นหาต่อไป จนเงื่อนไขยุติการค้นหาถูกพบ ข้อมูลส่วนอื่น ๆ ที่เหลือของตารางที่ 5.2 สามารถพิจารณาได้ทำนองเดียวกันนี้

รูปแบบการลดจำนวนหน่วยค้นหาแบบค่อยเป็นค่อยไปตามตารางที่ 5.2 สามารถอธิบายแนวทางการตั้งค่ารอบการเรียกใช้ DM ได้จากตัวอย่างดังต่อไปนี้ กรณีของ MATS#2 ที่ทำการค้นหาคำตอบวงกว้างของฟังก์ชัน BF ตามตารางที่ 5.2 เป็นแบบสองหน่วยค้นหาและต้องมีการเรียกใช้ DM จำนวนหนึ่งครั้ง จึงกำหนดให้มีการเรียกใช้ DM ในรอบการค้นหาที่ 5 เนื่องจากผลการทดสอบด้วย ATS แบบเดี่ยว ตามตารางที่ 5.3 ให้ค่าเฉลี่ยประมาณ 556 รอบการค้นหา จึงจะพบคำตอบวงกว้าง ประกอบกับ MATS มีสองหน่วยค้นหาสลับกันทำงาน จึงควรตั้งค่ารอบการเรียกใช้ DM ให้ต่ำกว่าครึ่งหนึ่งของค่าเฉลี่ยรอบการค้นหาที่ได้จาก ATS แบบเดี่ยว เพื่อจะได้มีโอกาสทำความเร็วการค้นหาคำตอบวงกว้างได้ดีกว่า ATS แบบเดียวนั่นเอง อย่างไรก็ตามการตั้งค่ารอบการเรียกใช้ DM ยิ่งต่ำกว่าค่าเฉลี่ยรอบการค้นหาของ ATS แบบเดี่ยว ยังมีโอกาสใช้เวลาการค้นหาบ่อยก็จริงตามความรู้สึกทั่วไป ซึ่งความเป็นจริงแล้วจะขึ้นกับลักษณะเฉพาะของแต่ละปัญหาด้วย ในที่สุดก็ควรต้องมีการปรับตั้งค่าดังกล่าวให้เหมาะสมอีกครั้งด้วยการลองผิดลองถูกกับแต่ละปัญหาอีกทีหนึ่ง

ตัวอย่างถัดไป เป็นการตั้งค่าการเรียกใช้ MATS#4 ซึ่งมีสี่หน่วยค้นหาหรือสี่เส้นทางการค้นหา นั่นเอง ตามตารางที่ 5.2 จะมีการเรียกใช้ DM จำนวนสองครั้ง เพื่อลดจำนวนหน่วยค้นหาจากสี่ให้เหลือสองในการเรียกใช้ DM ครั้งที่หนึ่ง และลดจำนวนหน่วยค้นหาที่เหลือจากสองหน่วยค้นหาเป็นหนึ่งหน่วยค้นหาในการเรียกใช้ DM ครั้งที่สองซึ่งเป็นครั้งสุดท้าย อาศัยข้อมูลจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว เช่นเดียวกับการตั้งค่าของ MATS#2 ที่ได้นำเสนอไปแล้วข้างต้น กรณีของ MATS#4 มีสี่เส้นทางจึงควรตั้งรอบการเรียกใช้ DM ทั้งสองครั้งที่ทำให้ผลรวมรอบการค้นหาทั้งหมดน้อยกว่าค่าเฉลี่ยรอบการค้นหาของ ATS แบบเดี่ยว นั่นคือประมาณ 556 รอบการค้นหา แนวทางการตั้งค่ารอบการเรียกใช้ DM สามารถสรุปได้ตามสมการที่ (5.1) ข้างล่างนี้

$$\sum_{i=1}^n dm_i \times n_i < average\ search\ round_{ATS} \quad (5.1)$$

เมื่อ dm_i คือรอบการค้นหามีการเรียกใช้ DM ครั้งที่ i n_i คือจำนวนหน่วยค้นหาก่อนการเรียกใช้ DM ครั้งที่ i และ $average\ search\ round_{ATS}$ คือค่าเฉลี่ยรอบการค้นหาที่พบคำตอบวงกว้างของ ATS แบบเดี่ยว ซึ่งเป็นค่าที่ได้จากการทดสอบดังแสดงในตารางที่ 5.3 กรณีของ MATS#2 ที่ค้นหาค้น ฟังก์ชัน BF อ้างอิงข้อมูลในตารางที่ 5.2 มีการเรียกใช้ DM หนึ่งครั้ง ณ รอบการค้นหาที่ 5 จึงได้ ค่า $dm_1 = 5$ $n_1 = 2$ และ $average\ search\ round_{ATS} = 556$ และเมื่อแทนค่าเหล่านี้ลงในสมการที่ (5.1) ยังคงทำให้อสมการที่ (5.1) เป็นจริง ดังแสดงในสมการที่ (5.2) ข้างล่างนี้

$$5 \times 2 < 556 \quad (5.2)$$

กรณีของ MATS#4 ที่ค้นหาค้นฟังก์ชัน BF อ้างอิงข้อมูลในตารางที่ 5.2 มีการเรียกใช้ DM สองครั้ง ณ รอบการค้นหาที่ 5 และ รอบการค้นหาที่ 10 ตามลำดับ ดังนั้น $dm_1 = 5$ $dm_2 = 10$ $average\ search\ round_{ATS} = 556$ $n_1 = 4$ และ $n_2 = 2$ เมื่อแทนค่าเหล่านี้ลงในสมการที่ (5.1) ยังคงทำให้อสมการที่ (5.1) เป็นจริง ดังแสดงในสมการที่ (5.3) ข้างล่างนี้

$$5 \times 4 + 10 \times 2 < 556 \quad (5.3)$$

กรณีอื่น ๆ ของ MATS ตามตารางที่ 5.2 เบื้องต้นได้ปฏิบัติตามข้อเสนอแนะการตั้งค่า การเรียกใช้ DM ทำนองเดียวกันกับสองตัวอย่างดังกล่าวข้างต้น ร่วมกับข้อมูลจากการทดสอบของ ATS แบบเดี่ยวตามตารางที่ 5.3 ประกอบ ยกเว้นกรณีที่ไม่สามารถปรับปรุงสมรรถนะให้ ดีกว่า ATS แบบเดี่ยวได้ จึงจำเป็นต้องหารูปแบบการตั้งค่าการเรียกใช้ DM แบบอื่นหรืออาจต้อง เปลี่ยนรูปแบบการลดจำนวนหน่วยค้นหาเป็นแบบอื่น อย่างเช่นรูปแบบการลดจำนวนหน่วยค้นหา แบบทันทีทันใดทั้งนี้เพื่อให้การะกัการปรับปรุงสมรรถนะการค้นหาที่เหนือกว่า ATS แบบเดี่ยว บรรลุได้

การตั้งค่าการเรียกใช้ DM สำหรับรูปแบบการลดจำนวนหน่วยค้นหาแบบ ทันทีทันใดตามตารางที่ 5.2 นั้น กำหนดไว้ที่รอบการค้นหาที่ 1 เพื่อเป็นตัวอย่าง การลดจำนวน หน่วยค้นหาตามรูปแบบนี้ ผู้วิจัยเสนอไว้เพื่อเป็นทางเลือกสำหรับบางปัญหาที่มีความยุ่งยาก ในการตั้งค่าพารามิเตอร์สำหรับรูปแบบปกติที่มีการลดจำนวนหน่วยค้นหาอย่างค่อยเป็นค่อยไป เพื่อให้ได้สมรรถนะการค้นหาที่ดีกว่าหน่วยค้นหาเดี่ยว หรือ ATS แบบเดี่ยว

5.3 ผลการทดสอบ

โปรแกรมการค้นหาแบบตามเชิงปรับตัวหลายวิธีฉบับนี้ พัฒนาขึ้นด้วยภาษา MATLAB™ ดำเนินการทดสอบ กับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นผิวทั้งหกฟังก์ชัน บนเครื่องคอมพิวเตอร์ Pentium 4 ความเร็ว 2.6 จิกกะเฮิร์ตซ์ ซึ่งมีหน่วยความจำ RAM ขนาด 248 เมกะไบต์ หน่วยความจำ HDD ขนาด 30 จิกกะไบต์ จำนวน 50 ครั้งแล้วหาค่าเฉลี่ยได้ผลการทดสอบเป็นผลการค้นหาเฉลี่ย และ จำนวนรอบการค้นหาเฉลี่ย แสดงอยู่ในตารางที่ 5.4 และ 5.5 ตามลำดับ ส่วนตารางที่ 5.6 เป็นการนำเสนอข้อมูลจากตารางที่ 5.4 ในรูปแบบเปอร์เซ็นต์การลดลงของเวลาการค้นหา และตารางที่ 5.7 เป็นการนำเสนอข้อมูลจากตารางที่ 5.5 ในรูปแบบจำนวนรอบการค้นหาเฉลี่ยของ MATS สมมูลกับ ATS แบบเดี่ยว เพื่อยืนยันความถูกต้องของผลเวลาการค้นหาเฉลี่ยที่ได้

ค่าเวลาการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS แต่ละค่าที่แสดงในตารางที่ 5.4 ได้จากการทดสอบ 50 ครั้งแล้วหาค่าเฉลี่ย ค่าของ ATS แบบเดี่ยวในคอลัมน์ที่สองใช้เป็นค่าอ้างอิงสำหรับการปรับตั้งค่าพารามิเตอร์ของ MATS จนกว่าจะได้ค่าที่พอใจ นั่นคือค่าเวลาการค้นหาเฉลี่ยของ MATS ควรต้องต่ำกว่าของ ATS แบบเดี่ยวนั่นเอง สมาชิกของ MATS ที่สร้างขึ้นสำหรับงานวิจัยนี้มีอยู่ด้วย 6 ชนิด ได้แก่ MATS#2 MATS#4 MATS#8 MATS#16 MATS#32 และ MATS#64 ซึ่งตัวเลขหลังสัญลักษณ์ # หมายถึง จำนวนหน่วยค้นหาหรือจำนวนเส้นทางการค้นหาของ MATS ตัวอย่างเช่น MATS#2 ก็คือ MATS แบบสองหน่วยค้นหาหรือแบบสองเส้นทางการค้นหา เป็นต้น ค่าเวลาการค้นหาเฉลี่ยของ MATS ที่ได้เกือบทุกชนิดในทุกฟังก์ชัน ตามตารางที่ 5.4 พบว่าให้ค่าเวลาการค้นหาเฉลี่ยต่ำกว่าของ ATS แบบเดี่ยวของแต่ละฟังก์ชัน ยกเว้นกรณีของฟังก์ชัน ScF2 ที่ทำการค้นหาด้วย MATS#4 MATS#8 MATS#16 MATS#32 และ MATS#64 ใช้เวลาค้นหาค่าตอบวงกว้างเฉลี่ยมากกว่าของ ATS แบบเดี่ยว จึงเป็นกรณีตัวอย่างให้เห็น MATS บางชนิดหากการตั้งค่าพารามิเตอร์ไม่เหมาะสมก็ไม่สามารถให้ผลการค้นหาที่ดีขึ้นกว่า ATS แบบเดี่ยว ซึ่งเป็นหน่วยค้นแกนได้ ตัวอย่างการอ่านค่าเวลาการค้นหาเฉลี่ยจากตารางที่ 5.4 สำหรับกรณีของฟังก์ชัน RF เวลาการค้นหาค่าตอบวงกว้างเฉลี่ยของ ATS แบบเดี่ยว อยู่ที่ 8.2067 วินาที ส่วนของ MATS#2 #4 #8 #16 #32 และ #64 ใช้เวลาการค้นหาค่าตอบวงกว้างเฉลี่ยอยู่ที่ 6.5863 6.9925 7.2175 4.5316 5.0078 และ 7.3221 วินาที ตามลำดับ ซึ่งทุกตัวล้วนน้อยกว่าของ ATS แบบเดี่ยว โดยมีของ MATS#16 ใช้เวลาการค้นหาน้อยที่สุดจึงถือว่าหาค่าตอบวงกว้างของ RF ได้รวดเร็วที่สุด หากมีการคัดเลือกตัวแทน MATS เพื่อทำการค้นหาค่าตอบวงกว้างสำหรับฟังก์ชัน RF หรือ ฟังก์ชันที่มีลักษณะพื้นผิวคล้ายคลึงกับฟังก์ชัน RF จึงแนะนำให้เลือก MATS#16 เพื่อทำการค้นหาจะเหมาะสมที่สุด ตัวอย่างที่สองเป็นกรณีของฟังก์ชัน ScF1 เวลาการค้นหาค่าตอบวงกว้างเฉลี่ยของ ATS แบบเดี่ยว อยู่ที่ 38.1037 วินาที ซึ่งมากกว่ากรณีของฟังก์ชัน RF เกือบห้าเท่า ทั้งนี้เนื่องจากปริภูมิการค้นหาที่มีขนาดใหญ่ถึง ± 500 รวมถึงความยุ่งยากและอุปสรรคของการค้นหา

คำตอบวงกว้างที่มีมากกว่า จึงทำให้ต้องใช้เวลาการค้นหาคำตอบวงกว้างนานขึ้นส่วนเวลาการค้นหาคำตอบวงกว้างของ MATS ทั้งหกชนิด เรียงตามลำดับตั้ง #2 จนถึง #64 ใช้ไป 30.3475 34.6200 28.1975 33.7100 34.8431 และ 29.2327 วินาที ตามลำดับ กรณีนี้คล้ายคลึงกับกรณีแรกที่กำลังกล่าวถึงที่สมาชิกทุกตัวของ MATS ใช้เวลาการค้นหาน้อยกว่าของ ATS แบบเดี่ยว ส่วนกรณีที่ MATS ใช้เวลาการค้นหามากกว่าของ ATS แบบเดี่ยว หรือ ช้ากว่า ATS แบบเดียวนั้น เกิดขึ้นกับ MATS ที่ทดสอบกับปัญหาเพียงฟังก์ชันเดียวจากหกฟังก์ชัน นั่นคือฟังก์ชัน ScF2 โดยเกิดกับ MATS ตั้งแต่ #4 จนถึง #64 เวลาการค้นหาเฉลี่ยของ ATS แบบเดี่ยว อยู่ที่ 0.2522 วินาที แต่ของ MATS#4 ใช้เวลาการค้นหาเฉลี่ยอยู่ที่ 0.2609 วินาที MATS#8 ใช้เวลาการค้นหาเฉลี่ยอยู่ที่ 0.3009 วินาที MATS#16 ใช้เวลาการค้นหาเฉลี่ยอยู่ที่ 0.3213 วินาที MATS#32 ใช้เวลาการค้นหาเฉลี่ยอยู่ที่ 0.4350 วินาที และ MATS#64 ใช้เวลาการค้นหาเฉลี่ยมากถึง 1.5179 วินาที

ตารางที่ 5.4 เวลาการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS

ฟังก์ชัน ทดสอบ	ค่าเวลาการค้นหาเฉลี่ยจากการทดสอบ 50 ครั้ง (วินาที)						
	ATS 1-path	MATS					
		# 2	# 4	# 8	# 16	# 32	# 64
BF	5.3940	5.0516	4.8901	2.3435	2.3309	4.1309	3.3029
RF	8.2067	6.5863	6.9925	7.2175	4.5316	5.0078	7.3221
SF	2.2837	1.8818	1.7697	1.8134	1.3918	1.7944	1.8431
ShF	3.3978	2.9819	3.1369	2.4038	2.6099	2.2640	2.9433
ScF1	38.1037	30.3475	34.6200	28.1975	33.7100	34.8431	29.2327
ScF2	0.2522	0.2419	0.2609	0.3009	0.3213	0.4350	1.5179

ตารางที่ 5.5 จำนวนรอบการค้นหาค่าเฉลี่ยของ ATS แบบเดี่ยว และ MATS

ฟังก์ชัน ทดสอบ	ค่าจำนวนรอบการค้นหาค่าเฉลี่ยจากการทดสอบ 50 ครั้ง (รอบ)						
	ATS	MATS					
	1-path	# 2	# 4	# 8	# 16	# 32	# 64
BF	556.06	530.20	507.94	186.82	142.70	202.34	173.76
RF	759.80	685.40	603.78	713.26	327.32	313.48	200.32
SF	108.22	79.84	72.96	64.62	30.62	54.04	36.16
ShF	342.54	291.56	282.16	75.68	73.02	61.68	70.12
ScF1	336.24	304.88	294.44	241.56	268.22	263.18	131.10
ScF2	31.24	30.02	29.20	26.48	24.26	21.58	19.96

ตารางที่ 5.6 เปอร์เซ็นต์การลดลงของเวลาการค้นหาค่าเฉลี่ยของ MATS เทียบกับ ATS แบบเดี่ยว

ฟังก์ชัน ทดสอบ	ค่าเปอร์เซ็นต์การลดลงของเวลาการค้นหาค่าเฉลี่ยจากการทดสอบ 50 ครั้ง (%)						
	ATS	MATS					
	1-path	# 2	# 4	# 8	# 16	# 32	# 64
BF	0.00	6.35	9.34	56.55	56.79	23.42	38.77
RF	0.00	19.74	14.80	12.05	44.78	38.98	10.78
SF	0.00	17.60	22.51	20.59	39.06	21.43	19.29
ShF	0.00	12.24	7.68	29.25	23.19	33.37	13.38
ScF1	0.00	20.36	9.14	26.00	11.53	8.56	23.28
ScF2	0.00	4.08	-3.45	-19.31	-27.40	-72.48	-501.86

ตารางที่ 5.7 ประมาณการจำนวนรอบการค้นหาค่าเฉลี่ยของ ATS แบบเดี่ยว และ MATS

ฟังก์ชัน ทดสอบ	ค่าจำนวนรอบการค้นหาค่าเฉลี่ยของ MATS ที่สมมูลกับของ ATS แบบเดี่ยว (รอบ)						
	ATS 1-path	MATS					
		# 2	# 4	# 8	# 16	# 32	# 64
BF	556.06	535.20	527.94	241.82	272.70	487.34	328.76
RF	759.80	690.40	753.78	735.26	379.32	427.48	263.32
SF	108.22	84.84	80.96	86.62	82.62	85.04	99.16
ShF	342.54	292.56	285.16	310.72	336.08	274.72	340.48
ScF1	336.24	309.88	299.44	256.56	303.22	295.18	194.10
ScF2	31.24	31.02	32.20	33.48	39.26	52.58	82.96

ตารางที่ 5.8 เปอร์เซ็นต์การลดลงของจำนวนรอบการค้นหาค่าเฉลี่ยของ MATS เทียบกับ ATS แบบเดี่ยว

ฟังก์ชัน ทดสอบ	ค่าเปอร์เซ็นต์การลดลงของจำนวนรอบการค้นหาค่าเฉลี่ยที่สมมูลกับของ ATS (%)						
	ATS 1-path	MATS					
		# 2	# 4	# 8	# 16	# 32	# 64
BF	0.00	3.75	5.06	56.51	50.96	12.36	40.88
RF	0.00	9.13	0.79	3.23	50.08	43.74	65.34
SF	0.00	21.60	25.19	19.96	23.66	21.42	8.37
ShF	0.00	14.59	16.75	9.29	1.89	19.80	0.60
ScF1	0.00	7.84	10.94	23.70	9.82	12.21	42.27
ScF2	0.00	0.70	-3.07	-7.17	-25.67	-68.31	-165.56

สาเหตุที่ทำให้เวลาการค้นหาคำตอบวงกว้างเฉลี่ยของ MATS#64 แตกต่างจากของ ATS แบบเดี่ยวมากดังกล่าว ถ้าพิจารณาจากภาพพื้นผิวของ ScF2 ในบทที่ 3 ตารางที่ 3. 1 ประกอบ จะพบว่าพื้นผิวไม่มีหลุมหรือแอ่งของคำตอบวงแคบเฉพาะถิ่นปรากฏให้เป็นอุปสรรคต่อการค้นหาทั้งของ ATS แบบเดี่ยว และ ของ MATS รวมถึงปริภูมิการค้นหาก็มีขนาดเล็กลงจาก ScF1 ถึง 100 เท่า จึงเป็นสาเหตุให้ ATS แบบเดี่ยว ใช้เวลาน้อยมากในการค้นหาคำตอบวงกว้าง ส่วน MATS นั้นต้องเสียเวลาแบ่งปริภูมิการค้นหาตามกลไก PM และยังต้องเสียเวลาอีกส่วนหนึ่งไปกับเวลาคอยตามกลไก SM ประกอบกับจำนวนหน่วยค้นหาที่มีมากถึง 64 หน่วยค้นหาของ MATS#64 จึงต้องใช้เวลาการค้นหามากดังกล่าวมาแล้วนั่นเอง

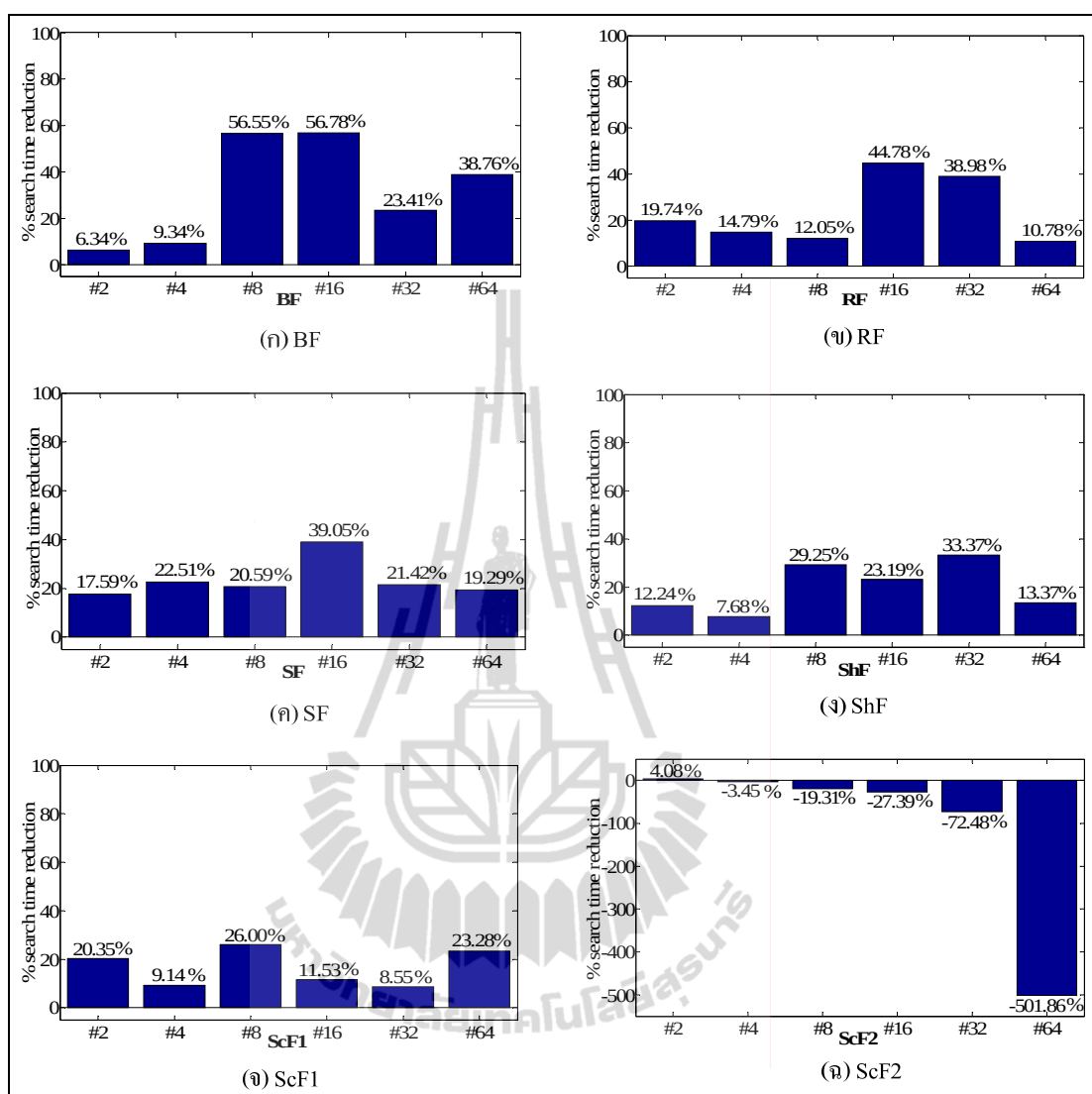
ตารางที่ 5.5 แสดงจำนวนรอบการค้นหาเฉลี่ยจากการทดสอบของ ATS แบบเดี่ยว และของ MATS ทั้งนี้เพื่อใช้ตรวจสอบความถูกต้องของค่าเวลาการค้นหาที่ MATS ใช้ไปด้วย นอกจากนี้จำนวนรอบการค้นหาเฉลี่ยจากการทดสอบของ ATS แบบเดี่ยว ในคอลัมน์ที่ 1 ยังถูกใช้เป็นแนวทางการปรับตั้งค่าพารามิเตอร์การเรียกใช้ DM ของ MATS ด้วยเป้าหมายที่จะลดจำนวนรอบการค้นหาของ MATS เพื่อการลดเวลาการค้นหาของ MATS ให้น้อยกว่าของ ATS แบบเดี่ยว ในที่สุดจากตารางที่ 5.5 พบว่า MATS ทุกชนิดและทุกฟังก์ชันใช้จำนวนรอบการค้นหาน้อยกว่าของ ATS แบบเดี่ยว แต่ไม่ได้หมายความว่า MATS ทุกตัวจะทำการค้นหาคำตอบวงกว้างได้เร็วกว่า ATS แบบเดี่ยว เช่นกรณีของ ฟังก์ชัน SchF#2 ATS หนึ่งในเส้นทาง ตามข้อมูลในตารางที่ 5.5 การค้นหาใช้จำนวนรอบการค้นหาเฉลี่ยที่ 31.24 รอบ ซึ่ง MATS#4 MATS#8 MATS#16 MATS#32 และ MATS#64 ใช้จำนวนรอบการค้นหาเฉลี่ยที่ 29.20 26.48 24.26 21.58 และ 19.96 รอบตามลำดับ ถ้าคิดแบบทั่วไปโดยไม่คำนึงถึงธรรมชาติของ MATS ตามที่ได้อธิบายแล้ว ก็จะเข้าใจว่า MATS ทั้งสามชนิดนี้ทำการค้นหาได้เร็วกว่า ATS แบบเดี่ยว เนื่องจากจำนวนรอบน้อยกว่า แต่ความจริงเมื่อพิจารณาจากค่าเวลาค้นหาเฉลี่ยที่ได้ในการตารางที่ 5.4 พบว่า MATS ทั้งห้าตัวใช้เวลาการค้นหามากกว่าที่ ATS แบบเดี่ยวใช้นี้ เป็นเพราะจำนวนหน่วยค้นหาของ MATS มีมากเกินไป จะสังเกตเห็นว่า ATS แบบเดี่ยว หนึ่งหน่วยค้นหาใช้จำนวนรอบไป ประมาณ 32 รอบ ดังนั้นเพื่อเอาชนะ ATS แบบเดี่ยวให้ได้ MATS#4 ต้องหาคำตอบวงกว้างให้พบภายใน 8 รอบการค้นหา ส่วน MATS#8 ต้องไม่เกิน 4 รอบการค้นหา ส่วน MATS#16 ไม่ควรใช้รอบการค้นหาเกินสองรอบต้องหาคำตอบวงกว้างให้พบ มิฉะนั้นจะไม่สามารถทำเวลาได้ดีกว่า ATS แบบเดี่ยวได้เลย ส่วนของ MATS#32 ยิ่งยากกว่าของ MATS#16 เพราะต้องใช้เพียงหนึ่งรอบการค้นหาเพื่อเอาชนะ ATS แบบเดี่ยว ส่วนของ MATS#64 ยิ่งเป็นไปได้ที่จะชนะ ATS แบบเดี่ยว เนื่องจากหนึ่งวงรอบการค้นหาของ MATS#64 ก็ใช้เวลามากกว่าของ ATS แบบเดี่ยวไปแล้วนั่นเอง จากเหตุผลดังอธิบายข้างต้น สามารถสรุปเป็นแนวทางการเลือกใช้งาน MATS หรือ อัลกอริธึมการค้นหาอื่นใดสำหรับแก้ปัญหาการหาค่าเหมาะที่สุดต่าง ๆ ควรเลือกอัลกอริธึมให้เหมาะสมกับลักษณะของปัญหา เพื่อให้เกิดประสิทธิภาพการทำงานสูงสุด หมายความว่าถ้าหากปัญหามีความง่ายควรเลือกใช้ อัลกอริธึมที่ไม่ซับซ้อนที่สามารถให้คำตอบได้ถูกต้องและรวดเร็วจะเหมาะสมกว่า แต่หากปัญหาที่มีความยุ่งยากซับซ้อนการเลือกใช้อัลกอริธึมอย่างง่ายอาจไม่สามารถให้คำตอบได้ถูกต้องและใช้เวลาการค้นหาคำตอบนานก็จำเป็นต้องเลือกใช้อัลกอริธึมที่มีประสิทธิภาพสูงขึ้นซึ่งจะมีความซับซ้อนของอัลกอริธึมมากขึ้นตามไปด้วยจึงจะเหมาะสมกว่านั่นเอง

“ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหา” หรือ “% search time reduction” ดังแสดงในตารางที่ 5.6 ได้จากการแปลงข้อมูลจากตารางที่ 5.4 ด้วยสมการที่ (5.4)

$$\% \text{ search time reduction} = \frac{(\text{average search time of ATS} - \text{average search time of MATS})}{\text{average search time of ATS}} \times 100 \quad (5.4)$$

เมื่อ average search time of ATS คือเวลาการค้นหเฉลี่ยของ ATS และ average search time of MATS คือเวลาการค้นหเฉลี่ยของ MATS ทำให้เห็นชัดเจนว่าสมรรถนะการค้นหของ MATS ดีขึ้นหรือแย่ลงกว่าของ ATS แบบเดี่ยวอย่างไร หากสมรรถนะการค้นหของ MATS ดีกว่าของ ATS แบบเดี่ยว จะให้เปอร์เซ็นต์การลดลงของเวลาค้นหาเป็นบวก ในทางตรงกันข้าม หากสมรรถนะการค้นหของ ATS แบบเดี่ยวดีกว่าของ MATS จะให้เปอร์เซ็นต์การลดลงของเวลาค้นหาเป็นลบ จากตารางที่ 5.6 คอลัมน์ที่ 2 ซึ่งเป็นผลของ ATS แบบเดี่ยว พบว่าทั้งหกฟังก์ชันจะมีค่าเท่ากับศูนย์เหมือนกันหมด ทั้งนี้เนื่องจากการคำนวณตามสมการที่ (5.4) โดยแทน averaged search time of MATS สำหรับกรณี MATS หนึ่งเส้นทางซึ่งจะมีค่าเท่ากับ average search time of ATS นั่นเอง ทำให้ได้ค่าเป็นศูนย์ ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาเท่ากับศูนย์นี้ยังทำหน้าที่เป็นเส้นแบ่งเขตระหว่างสมรรถนะที่ดีขึ้นและสมรรถนะที่แย่ลงของ MATS เทียบกับของ ATS นั่นเอง

ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาของ MATS เกือบทุกชนิดตั้งแต่แบบสองหน่วยค้นหจนกระทั่งหกสิบสี่หน่วยค้นหในทั้งหกฟังก์ชันล้วนมีค่าเป็นบวกมากกว่าหนึ่งทั้งนั้นแสดงว่า MATS ส่วนใหญ่มีสมรรถนะการค้นหที่ดีกว่า ATS แบบเดี่ยว ตัวอย่างเช่น กรณีของฟังก์ชัน SF MATS#16 สามารถให้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาได้เป็นบวกและสูงที่สุดในกลุ่ม MATS และ ATS แบบเดี่ยว ที่ทำการค้นหาคำตอบวงกว้างบนฟังก์ชัน SF ด้วยค่า 39.06% และ กรณีของฟังก์ชัน ShF ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาที่น้อยที่สุดในกลุ่มของ MATS เป็นของ MATS#4 มีค่า 7.68% ซึ่งก็ยังคงมีค่าเป็นบวกซึ่งแสดงถึงสมรรถนะการค้นหของ MATS ที่สูงกว่าของ ATS เป็นต้น ยกเว้นกรณีของฟังก์ชัน ScF2 ที่ MATS#4 MATS#8 MATS#16 #32 และ #64 ให้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาได้ติดลบ จึงมีความหมายว่าสมรรถนะการค้นหของ MATS ที่ด้อยกว่า ATS แบบเดี่ยว โดยค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาของ MATS#4 #8 #16 #32 และ #64 ที่ได้คือ -3.45% -19.31% -27.40% -72.48% และ -501.86% ตามลำดับ เพื่อให้เห็นภาพเปอร์เซ็นต์การลดลงของเวลาค้นหาของ MATS เทียบกับ ATS ชัดเจนขึ้นจึงนำเสนอในรูปแบบกราฟแท่ง แสดงดังรูปที่ 5.2



รูปที่ 5.2 กราฟแท่งเปอร์เซ็นต์การลดลงของเวลาค้นหาของ MATS เทียบกับ
ATS แบบเดี่ยวนฟังก์ชัน

จำนวนรอบการค้นหาละเลี้ยวของ MATS สมมูลกับ ATSแบบเดี่ยว แสดงดังตารางที่ 5.7
ได้จากการคำนวณตามสมการที่ (5.5) โดยอาศัยของมูลการเรียกใช้ DM จากตารางที่ 5.2 ประกอบ

$$\text{MATS search rounds equivalent to ATS} = \sum_{i=1}^n dm_i \times n_i \quad (5.5)$$

โดยที่ *MATS search rounds equivalent to ATS* หมายถึง จำนวนรอบการค้นหาของ MATS ที่สมมูลกับ ATS dm_i คือรอบการค้นหาที่มีการเรียกใช้ DM ครั้งที่ i n_i คือจำนวนหน่วยค้นหาก่อนการเรียกใช้ DM ครั้งที่ i ตัวอย่างการแปลงค่าสำหรับตารางที่ 5.7 กรณีของปัญหา BF ดำเนินการค้นหาด้วย MATS#4 โดยอาศัยข้อมูลจากตารางที่ 5.5 พบว่าใช้จำนวนรอบการค้นหาเฉลี่ย 507.94 รอบ ร่วมกับข้อมูล จากตารางที่ 5.2 ว่ามีการเรียกใช้ DM จำนวนสองครั้งในรอบการค้นหาที่ 5 และ 10 ตามลำดับ โดยทำครั้งแรกทำการลดจำนวนหน่วยค้นหาลงสองหน่วยค้นหา และครั้งที่สองลดอีกหนึ่งหน่วยค้นหาให้เหลือเพียงหนึ่งค้นหาในที่สุดจนกระทั่ง พบคำตอบวงกว้าง เมื่อแทนค่าลงในสมการที่ (5.5) จะได้ดังสมการที่ (5.6) จำนวนรอบการค้นหาเฉลี่ยสมมูลของ MATS#2 มีค่า 527.94 รอบ ซึ่งน้อยกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ซึ่งมีค่า 556.06 รอบ ดังนั้นเมื่อพิจารณาเวลาการค้นหาเฉลี่ยของ MATS#2 ที่ใช้ ไป 4.8901 วินาที ซึ่งน้อยกว่าของ ATS แบบเดี่ยว ซึ่งใช้ไป 5.3940 วินาที

$$MATS \#2 \text{ search rounds equivalent to ATS} = 507.94 \times 1 + 5 \times 2 + 10 \times 1 = 527.94 \quad (5.6)$$

กรณีของปัญหา SF ดำเนินการค้นหาด้วย MATS#64 โดยอาศัยข้อมูลจากตารางที่ 5.5 ใช้จำนวนรอบการค้นหาเฉลี่ย 36.16 รอบ ร่วมกับข้อมูลจากตารางที่ 5.2 ว่ามีการเรียกใช้ DM จำนวนหนึ่งครั้งในรอบการค้นหาที่ 1 แล้วทำการลดจำนวนหน่วยค้นหาลง 63 หน่วยค้นหา เมื่อแทนค่าลงในสมการที่ (5.5) จะได้ดังสมการที่ (5.7) จำนวนรอบการค้นหาเฉลี่ยสมมูลของ MATS#64 มีค่า 99.16 รอบ ซึ่งน้อยกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ซึ่งมีค่า 108.22 รอบ ดังนั้นเมื่อพิจารณาเวลาการค้นหาเฉลี่ยของ MATS#64 ที่ใช้ ไป 1.8431 วินาที ซึ่งน้อยกว่าของ ATS แบบเดี่ยว ซึ่งใช้ไป 2.2837 วินาที จึงมีความสมเหตุสมผล แม้ว่าจะลดจำนวนรอบการค้นหาเฉลี่ยสมมูลลงมาจากของ ATS แบบเดี่ยว ได้ไม่มากนักก็ตาม

$$MATS \#64 \text{ search rounds equivalent to ATS} = 36.16 \times 1 + 1 \times 63 = 99.16 \quad (5.7)$$

กรณีของปัญหา ShF ดำเนินการค้นหาด้วย MATS#32 โดยอาศัยข้อมูลจากตารางในตารางที่ 5.5 ใช้จำนวนรอบการค้นหาเฉลี่ย 61.68 รอบ ร่วมกับข้อมูลจากตารางที่ 5.2 ว่ามีการเรียกใช้ DM จำนวนหนึ่งครั้งในรอบการค้นหาที่ 1 แล้วทำการลดจำนวนหน่วยค้นหาลง 28 หน่วยค้นหา เหลือจำนวนหน่วยค้นหาในวงรอบการค้นหาของ MATS อีก 4 หน่วยค้นหา เมื่อแทนค่าลงในสมการที่ (5.5) จะได้ดังสมการที่ (5.8) จำนวนรอบการค้นหาเฉลี่ยสมมูลของ MATS#32 มีค่า 274.72 รอบ ซึ่งน้อยกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ซึ่งมีค่า 342.54 รอบ ดังนั้นเมื่อพิจารณา

เวลาการค้นหาเฉลี่ยของ MATS#32 ที่ใช้ไป 2.2640 วินาที ซึ่งน้อยกว่าของ ATS แบบเดี่ยว ซึ่งใช้ไป 3.3978 วินาที จึงมีความสมเหตุสมผล สิ่งที่น่าสังเกตสำหรับปัญหานี้คือ จำนวนหน่วยค้นหาคงเหลือก่อนการพบคำตอบวงกว้างไม่ใช่หนึ่งเหมือนกรณีทั่วไป เนื่องจากเมื่อได้พยายามปรับแต่งค่า DM แล้วพบว่าหากใช้ n_0 ที่มากกว่าหนึ่งจะให้ผลการค้นหาที่ดีขึ้น ทั้งนี้อาจเกิดจากธรรมชาติของพื้นผิว Sbf ประกอบด้วยคำตอบวงกว้างถึง 18 ค่า ดังนั้นการมีเส้นทางค้นหาคงเหลือในวงรอบการค้นหา มากกว่าหนึ่งเส้นทางเท่ากับเปิดโอกาสให้พบคำตอบวงกว้างตัวใดตัวหนึ่งได้รวดเร็วขึ้นกว่าการมีเพียงหนึ่งเส้นทางการค้นหาก็เป็นได้

$$\text{MATS \#32 search rounds equivalent to ATS} = 61.68 \times 4 + 1 \times 28 = 274.72 \quad (5.8)$$

กรณีของปัญหา ScF2 ดำเนินการค้นหาด้วย MATS#2 โดยอาศัยข้อมูลจากตารางในตารางที่ 5.5 ใช้จำนวนรอบการค้นหาเฉลี่ย 30.02 รอบ ร่วมกับข้อมูลจากตารางที่ 5.2 ว่ามีการเรียกใช้ DM จำนวนหนึ่งครั้งในรอบการค้นหาที่ 1 แล้วทำการลดจำนวนหน่วยค้นหาลง 1 หน่วยค้นหา เมื่อแทนค่าลงในสมการที่ (5.5) จะได้ดังสมการที่ (5.9) จำนวนรอบการค้นหาเฉลี่ยสมมูลของ MATS#8 มีค่า 31.02 รอบ ซึ่งน้อยกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ซึ่งมีค่า 31.24 รอบ เมื่อพิจารณาเวลาการค้นหาเฉลี่ยของ MATS#8 ที่ใช้ไป 0.2419 วินาที พบว่าน้อยกว่าของ ATS แบบเดี่ยว ซึ่งใช้ไป 0.2522 วินาที จึงเป็นเรื่องที่สมเหตุสมผลกันระหว่างจำนวนรอบการค้นหาสมมูลกับเวลาการค้นหาเฉลี่ย

$$\text{MATS \#64 search rounds equivalent to ATS} = 30.02 \times 1 + 1 \times 1 = 31.02 \quad (5.9)$$

กรณีของปัญหา ScF2 ดำเนินการค้นหาด้วย MATS#8 โดยอาศัยข้อมูลจากตารางในตารางที่ 5.5 ใช้จำนวนรอบการค้นหาเฉลี่ย 26.48 รอบ ร่วมกับข้อมูลจากตารางที่ 5.2 ว่ามีการเรียกใช้ DM จำนวนหนึ่งครั้งในรอบการค้นหาที่ 1 แล้วทำการลดจำนวนหน่วยค้นหาลง 7 หน่วยค้นหา เมื่อแทนค่าลงในสมการที่ (5.5) จะได้ดังสมการที่ (5.10) จำนวนรอบการค้นหาเฉลี่ยสมมูลของ MATS#8 มีค่า 33.48 รอบ ซึ่งมากกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว ซึ่งมีค่า 31.24 รอบ เมื่อพิจารณาเวลาการค้นหาเฉลี่ยของ MATS#8 ที่ใช้ไป 0.3009 วินาที จึงพบว่ามากกว่าของ ATS แบบเดี่ยว ซึ่งใช้ไป 0.2522 วินาที จึงเป็นเรื่องที่สมเหตุสมผลกันระหว่างจำนวนรอบการค้นหาสมมูลกับเวลาการค้นหาเฉลี่ย เมื่อพิจารณาจากลักษณะพื้นผิวของ ScF2 พบว่ามีเรียบและลาดเอียงเข้าสู่คำตอบวงกว้างในทุกทิศทาง ทำให้มีความง่ายต่อการค้นหาจึงเป็นข้อได้เปรียบสำหรับอัลกอริทึมที่ไม่มีความซับซ้อนในการประมวลผลที่จะใช้เวลาเพียงเล็กน้อยในการค้นหาก็พบคำตอบวงกว้างได้

โดยง่ายด้าย ดังนั้นปัญหา ScF2 โดยการค้นหาของ MATS#8 จึงไม่สามารถทำเวลาการค้นหาได้น้อยกว่า ATS แบบเดี่ยวได้ ด้วยเหตุผลและการอธิบายดังกล่าวข้างต้น

$$MATS \#64 \text{ search rounds equivalent to ATS} = 26.48 \times 1 + 1 \times 7 = 33.48 \quad (5.10)$$

ผลการทดสอบสมรรถนะของ MATS เทียบกับของ ATS แบบเดี่ยว ด้วยปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่ทั่วทั้งฟังก์ชันโดยพิจารณาจากค่าเปอร์เซ็นต์อัตราส่วนความเร็วปรับปรุงดังแสดงในรูปที่ 5.2 พบว่าห้าฟังก์ชันแรกได้แก่ BF RF SF ShF และ ScF1 MATS ทั้งหกชนิดตั้งแต่แบบสองหน่วยค้นหาจนถึงหกสิบสี่หน่วยค้นหาสามารถทำความเร็วการค้นหาตอบวงกว้างได้ดีกว่า ATS แบบเดี่ยว ส่วนฟังก์ชันสุดท้ายได้แก่ ScF2 มีเพียง MATS แบบสองหน่วยค้นหาเท่านั้นที่สามารถทำความเร็วการค้นหาได้ดีกว่า ATS แบบเดี่ยว ส่วนอีกห้าชนิดที่เหลือได้แก่ MATS#4 #8 #16 #32 และ #64 ไม่สามารถทำความเร็วให้ดีกว่าของ ATS แบบเดี่ยวได้ จึงแสดงให้เห็นว่า MATS มีความเร็วเฉลี่ยในการค้นหาคำตอบต่ำกว่า ATS แบบเดียวนั่นเอง

กรณีของ BF MATS#16 สามารถให้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาได้สูงสุดถึง 56.79% เท่าเทียบกับ ATS แบบเดี่ยว และที่น้อยที่สุดเป็นของ MATS#2 ได้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาเพียง 6.35% ใกล้เคียงกับของ ATS แบบเดี่ยว กรณีของ RF เปอร์เซ็นต์การลดลงของเวลาค้นหาสูงสุด 44.78% เป็นของ MATS#16 และการลดลงของเวลาค้นหาน้อยที่สุด 10.78% เป็นของ MATS#64 กรณีของ SF ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาสูงสุดยังคงเป็นของ MATS#16 ทำได้ 39.06% และที่น้อยที่สุดเป็นของ MATS#2 ได้เพียง 17.60% กรณีของ ShF เป็นฟังก์ชันเดียวที่มีคำตอบวงกว้างมากกว่าหนึ่งคำตอบ มี MATS#32 ได้ค่าการลดลงของเวลาค้นหาสูงสุด 33.37% และน้อยที่สุดเป็นของ MATS#4 ด้วยค่าการลดลงของเวลาค้นหาสูงสุด 7.68% กรณีของ ScF1 ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาสูงสุดเป็นของ MATS#8 โดยทำได้ 26.00% และ MATS#32 ได้น้อยที่สุดเป็นคือ 8.56% กรณีสุดท้ายเป็นของ ScF2 คู่เหมือนในความต่างของ ScF1 เนื่องจากต้องใช้ฟังก์ชันทางคณิตศาสตร์ร่วมกันแต่ปริภูมิการค้นหาต่างกัน กรณีนี้เป็นกรณีพิเศษ เนื่องจากมีทั้งค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาแบบบวกและค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาแบบลบซึ่งหมายถึงช้ากว่า ATS แบบเดี่ยว รวมอยู่ด้วย โดย MATS#2 เป็นเพียงชนิดที่ให้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาเป็นบวกคือ 4.08% ส่วนที่เหลือได้แก่ MATS#4 MATS#8 MATS#16 และ MATS#32 ให้ค่าอัตราส่วนเปอร์เซ็นต์การลดลงของเวลาค้นหาเป็นลบคือ -3.45% -19.31% -27.40% และ -72.48% ตามลำดับ โดยเฉพาะ MATS#64 ให้ค่าเปอร์เซ็นต์การลดลงของเวลาค้นหาติดลบมากที่สุด ซึ่งหมายถึงทำการค้นหาคำตอบวงกว้างได้ช้าที่สุดในของทั้งกลุ่ม ATS แบบเดี่ยว และ MATS ด้วยค่าเปอร์เซ็นต์การลดลงของเวลาค้นหา -501.86%

“ค่าเปอร์เซ็นต์การลดลงของรอบค้นหา” หรือ “% search round reduction” ดังแสดงในตารางที่ 5.8 ได้จากการแปลงข้อมูลจากตารางที่ 5.7 ด้วยสมการที่ (5.11)

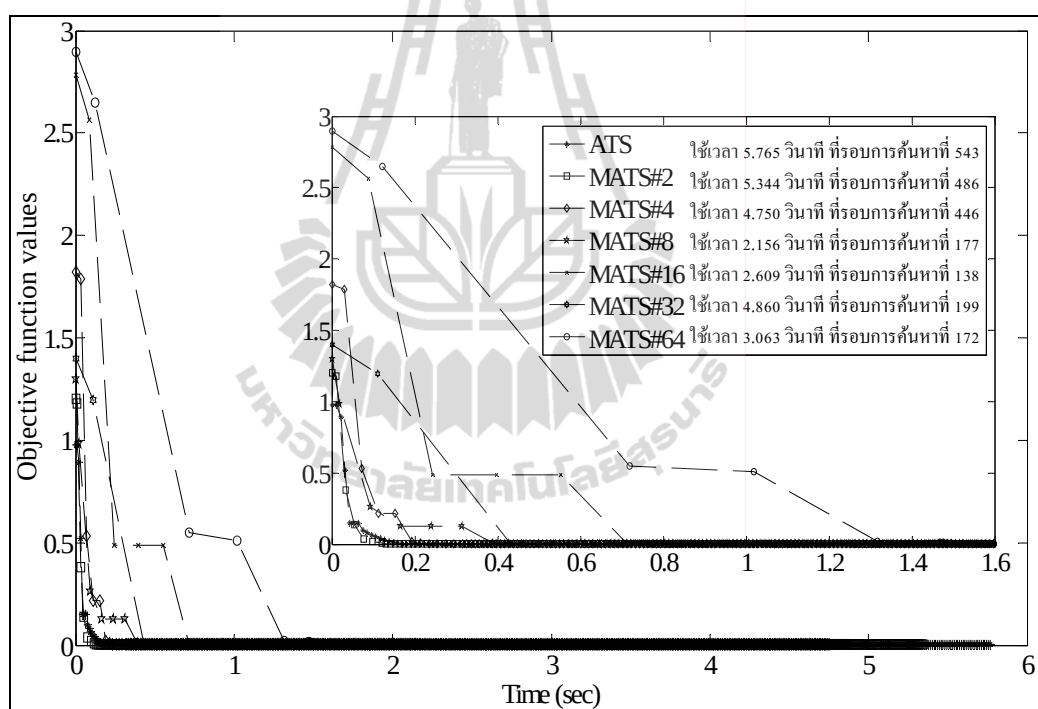
$$\begin{aligned} & \% \text{ search round reduction} \\ &= \frac{(\text{average search round of ATS} - \text{MATS search rounds equivalent to ATS})}{\text{average search round of ATS}} \times 100 \end{aligned} \quad (5.11)$$

เมื่อ average search round of ATS คือจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว และ MATS search rounds equivalent to ATS คือจำนวนรอบการค้นหาเฉลี่ยของ MATS ที่สมมูลกับของ ATS แบบเดี่ยว จากตารางที่ 5.8 กรณีของฟังก์ชัน BF พบว่า MATS#8 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 56.51% ส่วนที่น้อยที่สุดคือ 3.75% เป็นของ MATS#2 สำหรับกรณีของฟังก์ชัน RF พบว่า MATS#64 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 65.34% ส่วน MATS#4 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงน้อยที่สุดคือ 0.79% กรณีของฟังก์ชัน SF พบว่าให้ผลตรงกันข้ามกับกรณีของ RF คือ MATS#4 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 25.19% ส่วน MATS#64 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงน้อยที่สุดคือ 8.37% กรณีของฟังก์ชัน ShF พบว่า MATS#32 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 19.80% ส่วน MATS#64 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงน้อยที่สุดคือ 0.6% กรณีฟังก์ชัน ScF1 พบว่า MATS#64 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 42.27% ส่วน MATS#2 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงน้อยที่สุดคือ 7.84% และกรณีสุดท้ายฟังก์ชัน ScF2 พบว่า MATS#2 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงมากที่สุดคือ 0.70% ส่วน MATS#64 ให้เปอร์เซ็นต์การลดจำนวนรอบการค้นหาสมมูลลงน้อยที่สุดคือ -165.56% ที่เป็นลบเนื่องจาก MATS มีจำนวนรอบการค้นหาเฉลี่ยสมมูลมากกว่าจำนวนรอบการค้นหาเฉลี่ยของ ATS แบบเดี่ยว นั่นเอง และอาจสังเกตได้ว่าค่าเปอร์เซ็นต์การลดลงของจำนวนรอบการค้นหา มีแนวโน้มของตัวเลขไปในลักษณะคล้ายกันกับค่าเปอร์เซ็นต์การลดลงของเวลาการค้นหา

5.4 วิเคราะห์ผลการทดสอบ

จากผลการทดสอบที่อยู่ในรูปเปอร์เซ็นต์การลดลงของเวลาการค้นหา ดังแสดงในตารางที่ 5.6 ช่วยคัดแยกคุณภาพของ MATS ที่ดีขึ้นกว่า ATS แบบเดี่ยวได้ง่าย ๆ ด้วยการสังเกตที่เครื่องหมาย หากมีค่าเป็นบวกแสดงว่า MATS มีสมรรถนะที่ดีกว่า ATS แบบเดี่ยว แต่หากมี

เครื่องหมายติดลบแสดงว่า MATS มีสมรรถนะต่ำกว่า ซึ่งมีเพียงกรณีฟังก์ชัน ScF2 สำหรับ MATS เกินกว่า 2 เส้นทางการค้นหาเท่านั้น ส่วนอีกห้าฟังก์ชันที่เหลือ MATS ให้ผลการทดสอบส่วนดีกว่า ATS แบบเดี่ยวทั้งสิ้น เมื่อพิจารณาจากลักษณะพื้นผิวของทั้งหกฟังก์ชันประกอบกับผลการทดสอบที่ได้ สามารถสรุปได้ว่า MATS ให้ผลการค้นหาที่ดีกว่า ATS แบบเดี่ยวได้เมื่อปัญหามีความซับซ้อนและต้องการการสำรวจที่หลากหลายและกว้างขวางเพียงพอ อย่างกรณีของฟังก์ชัน BF RF SF ShF และ ScF1 เป็นต้น ถ้ากรณีของฟังก์ชันที่ไม่มีความซับซ้อนอย่าง ScF2 การนำ MATS ที่มีความซับซ้อนของการคำนวณมากกว่ามาเปรียบเทียบกับ ATS แบบเดี่ยว ที่ใช้การคำนวณไม่มากก็สามารถให้พบคำตอบวงกว้างได้ในเวลาอันรวดเร็ว



รูปที่ 5.3 กราฟผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา BF

กราฟการผู้เข้าหาคำตอบวงกว้าง ดังรูปที่ 5.3 ต้องการแสดงให้เห็นถึงคุณสมบัติการผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน BF นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9740 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 543 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ 3.3177×10^{-10} ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 5.7650 วินาที แต่ละคำตอบห่างกันประมาณ $t = 5.7650$ วินาที/543 รอบการค้นหา ≈ 0.0106 วินาที

MATS แบบ 2 เส้นทางค้นหา มีหน่วยค้นหาที่ 1 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 1.2018 จนกระทั่งรอบการค้นหาที่ 486 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.3224×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 5.3440 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 3 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 1.8234 จนกระทั่งรอบการค้นหาที่ 446 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.6243×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 4.8600 วินาที

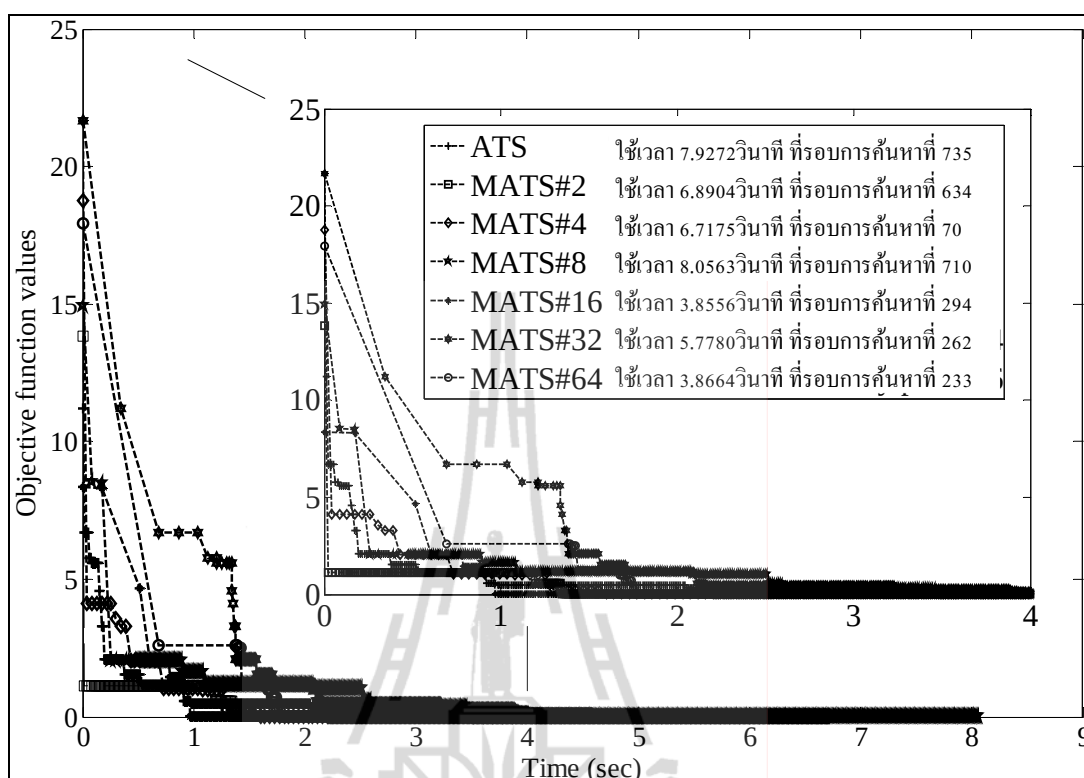
MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 2 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 1.2972 จนกระทั่งรอบการค้นหาที่ 177 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 1.8157×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 2.1500 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 9 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 2.7822 จนกระทั่งรอบการค้นหาที่ 138 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 1.7153×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 2.6090 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 11 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 1.3982 จนกระทั่งรอบการค้นหาที่ 199 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.6220×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 2.1500 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 13 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 2.8961 จนกระทั่งรอบการค้นหาที่ 172 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 2.9307×10^{-10} โดยใช้เวลาค้นหาไปทั้งสิ้น 3.0630 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.3 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากวงรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา $13t$ เมื่อแทนด้วย $t \approx 0.0106$ วินาที คิดเป็นเวลา $13 \times 0.0106 = 0.1378$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ $64t$ จะเป็นเวลาประมาณ $64 \times 0.0106 = 0.6784$ วินาที



รูปที่ 5.4 กราฟการลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา RF

กราฟการลู่เข้าหาคำตอบวงกว้าง ดังรูปที่ 5.4 ต้องการแสดงให้เห็นถึงคุณสมบัติการลู่เข้าหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน RF นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการอธิบายสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาคำด้วยค่าฟังก์ชันวัตถุประสงค์ 21.6416 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 735 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ 1.4393×10^{-9} ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 7.9272 วินาที แต่ละคำตอบห่างกันประมาณ $t = 7.9272 \text{ วินาที} / 735 \text{ รอบการค้นหา} \approx 0.0108 \text{ วินาที}$

MATS แบบ 2 เส้นทางค้นหา มีหน่วยค้นหาที่ 2 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาคำด้วยค่าฟังก์ชันวัตถุประสงค์ 13.8441 จนกระทั่งรอบการค้นหาที่ 634 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 5.9878×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 6.8904 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 1 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 18.7604 จนกระทั่งรอบการค้นหาที่ 470 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 1.8926×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 6.7175 วินาที

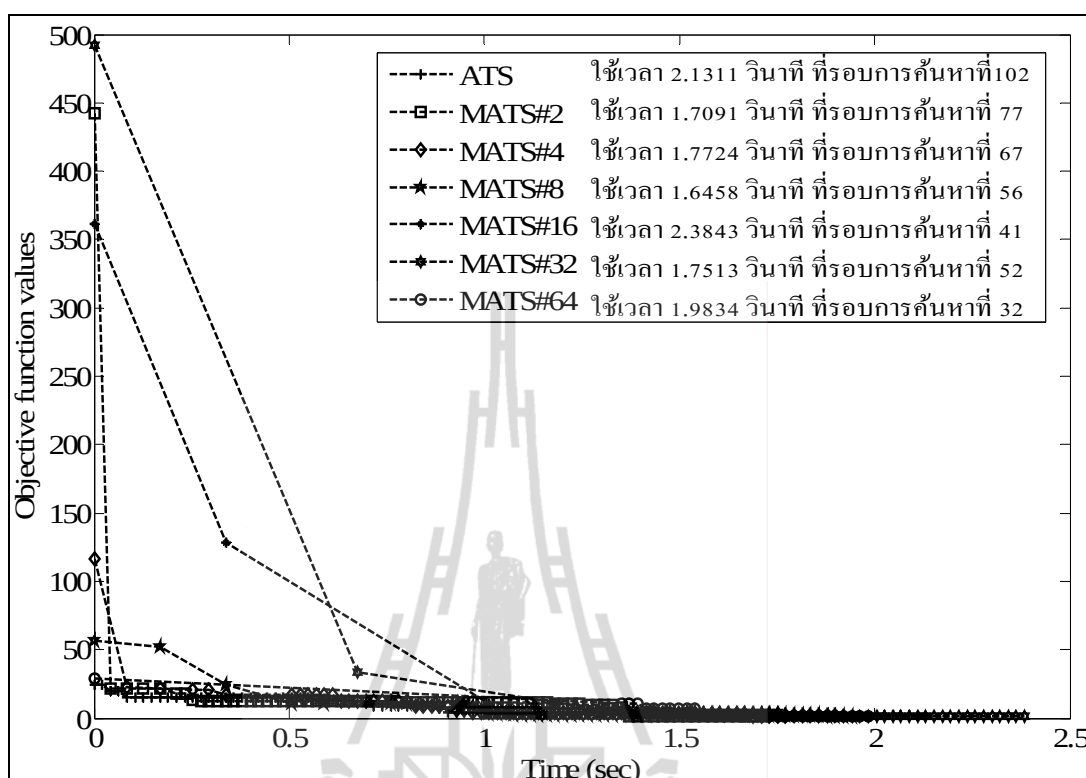
MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 2 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 14.9392 จนกระทั่งรอบการค้นหาที่ 710 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 2.9660×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 8.0563 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 5 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 8.3607 จนกระทั่งรอบการค้นหาที่ 294 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 2.141×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 3.8556 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 24 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 8.8237 จนกระทั่งรอบการค้นหาที่ 262 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 9.3481×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 5.7780 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 36 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 17.9332 จนกระทั่งรอบการค้นหาที่ 233 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 4.8136×10^{-9} โดยใช้เวลาค้นหาไปทั้งสิ้น 3.8664 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.4 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากวงรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา 13t เมื่อแทนด้วย $t \approx 0.0108$ วินาที คิดเป็นเวลา $36 \times 0.0108 = 0.3888$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ 64t จะเป็นเวลาประมาณ $64 \times 0.0108 = 0.6912$ วินาที



รูปที่ 5.5 กราฟการหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา SF

กราฟการหาคำตอบวงกว้าง ดังรูปที่ 5.5 ต้องการแสดงให้เห็นถึงคุณสมบัติการหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน SF นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการอธิบายสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 24.6516 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 102 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9980 ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 2.1311 วินาที แต่ละคำตอบห่างกันประมาณ $t = 2.2837$ วินาที/108.22 รอบการค้นหา ≈ 0.0211 วินาที

MATS แบบ 2 เส้นทางค้นหา มีหน่วยค้นหาที่ 2 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มดำเนินการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 442.5451 จนกระทั่งรอบการค้นหาที่ 77 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9984 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.7091 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 3 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 116.1422 จนกระทั่งรอบการค้นหาที่ 67 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9985 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.7724 วินาที

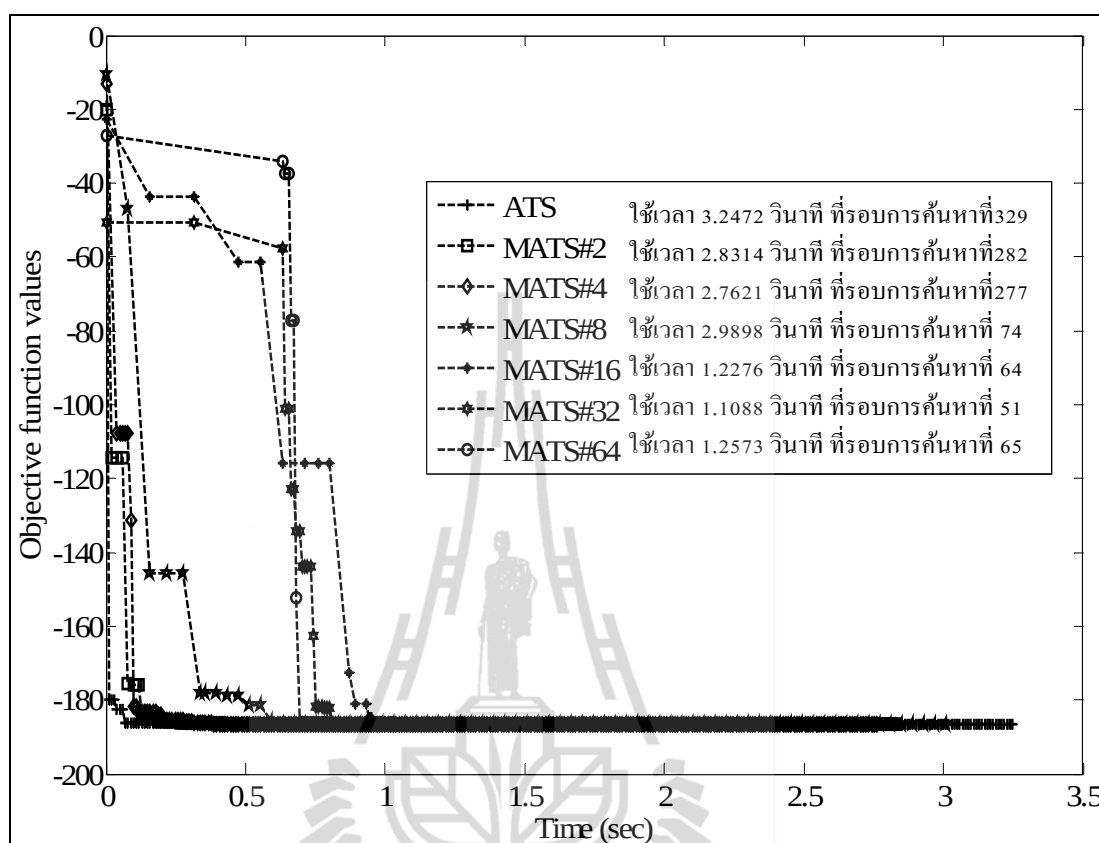
MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 6 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 56.1972 จนกระทั่งรอบการค้นหาที่ 56 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9980 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.6458 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 9 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 361.0743 จนกระทั่งรอบการค้นหาที่ 41 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9981 โดยใช้เวลาค้นหาไปทั้งสิ้น 2.3843 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 23 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 492.4344 จนกระทั่งรอบการค้นหาที่ 52 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9982 โดยใช้เวลาค้นหาไปทั้งสิ้น 2.3843 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 54 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 29.2753 จนกระทั่งรอบการค้นหาที่ 32 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 0.9980 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.9834 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.5 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากวงรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา $54t$ เมื่อแทนด้วย $t \approx 0.0211$ วินาที คิดเป็นเวลา $54 \times 0.0211 = 1.1394$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ $64t$ จะเป็นเวลาประมาณ $64 \times 0.0211 = 1.3504$ วินาที



รูปที่ 5.6 กราฟการหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ShF

กราฟการหาคำตอบวงกว้าง ดังรูปที่ 5.6 ต้องการแสดงให้เห็นถึงคุณสมบัติการหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน ShF นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการอธิบายสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาคำฟังก์ชันวัตถุประสงค์ -18.9087 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 329 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7053 ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 3.2472 วินาที แต่ละคำตอบห่างกันประมาณ $t = 3.2472 \text{ วินาที} / 329 \text{ รอบการค้นหา} \approx 0.0099 \text{ วินาที}$

MATS แบบ 2 เส้นทางค้นหา มีหน่วยค้นหาที่ 1 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มดำเนินการค้นหาคำฟังก์ชันวัตถุประสงค์ -20.1502 จนกระทั่งรอบการค้นหาที่ 282 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7088 โดยใช้เวลาค้นหาไปทั้งสิ้น 2.8314 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 4 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ -13.1016 จนกระทั่งรอบการค้นหาที่ 277 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7259 โดยใช้เวลาค้นหาไปทั้งสิ้น 2.7621 วินาที

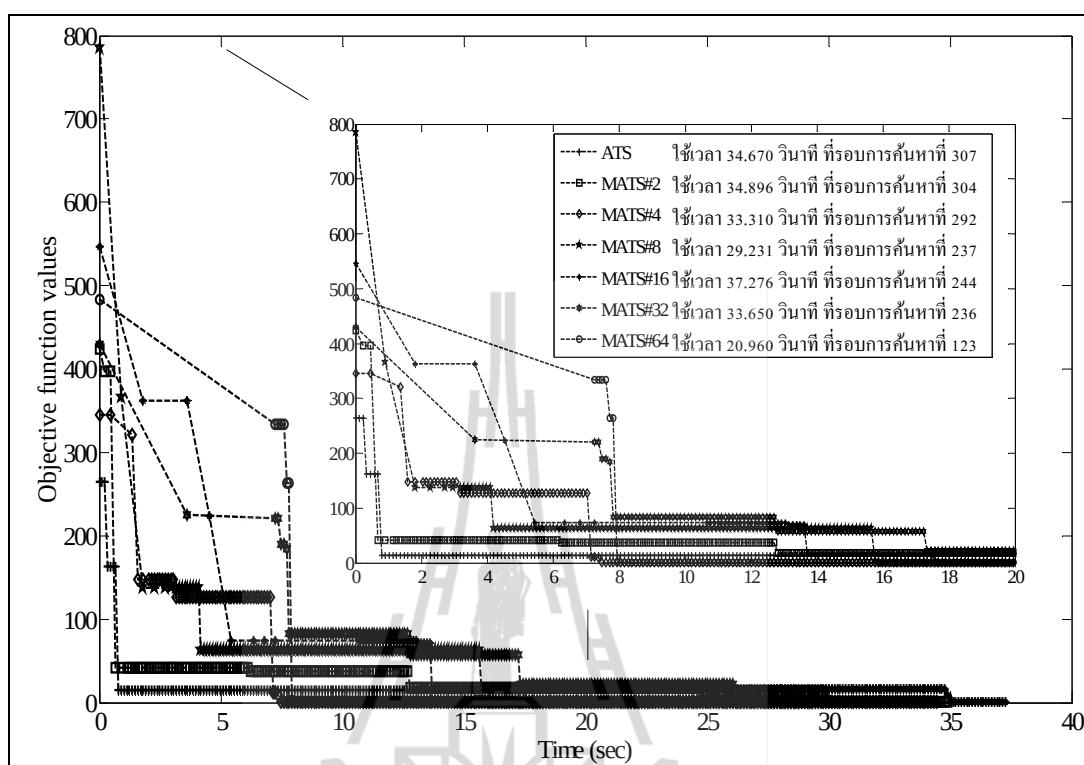
MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 5 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ -10.4622 จนกระทั่งรอบการค้นหาที่ 74 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7214 โดยใช้เวลาค้นหาไปทั้งสิ้น 2.9898 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 15 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ -22.8046 จนกระทั่งรอบการค้นหาที่ 64 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7286 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.2276 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 15 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ -50.7880 จนกระทั่งรอบการค้นหาที่ 51 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7270 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.1088 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 29 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ -26.9757 จนกระทั่งรอบการค้นหาที่ 65 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ -186.7307 โดยใช้เวลาค้นหาไปทั้งสิ้น 1.2573 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.6 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากวงรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา $29t$ เมื่อแทนด้วย $t \approx 0.0099$ วินาที คิดเป็นเวลา $29 \times 0.0099 = 0.2871$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ $64t$ จะเป็นเวลาประมาณ $64 \times 0.0099 = 0.6336$ วินาที



รูปที่ 5.7 กราฟผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ScF1

กราฟการผู้เข้าหาคำตอบวงกว้าง ดังรูปที่ 5.7 ต้องการแสดงให้เห็นถึงคุณสมบัติการผู้เข้าหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน ScF1 นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการอธิบายสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาคำด้วยค่าฟังก์ชันวัตถุประสงค์ 264.4531 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 307 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ 7.5484×10^{-5} ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 34.6698 วินาที แต่ละคำตอบห่างกันประมาณ $t = 34.6698$ วินาที/307 รอบการค้นหา ≈ 0.1129 วินาที

MATS แบบ 2 เส้นทางการค้นหา มีหน่วยค้นหาที่ 2 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาคำด้วยค่าฟังก์ชันวัตถุประสงค์ 424.5487 จนกระทั่งรอบการค้นหาที่ 304 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 2.8246×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 34.8964 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 4 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 345.5825 จนกระทั่งรอบการค้นหาที่ 292 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 2.6840×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 33.3102 วินาที

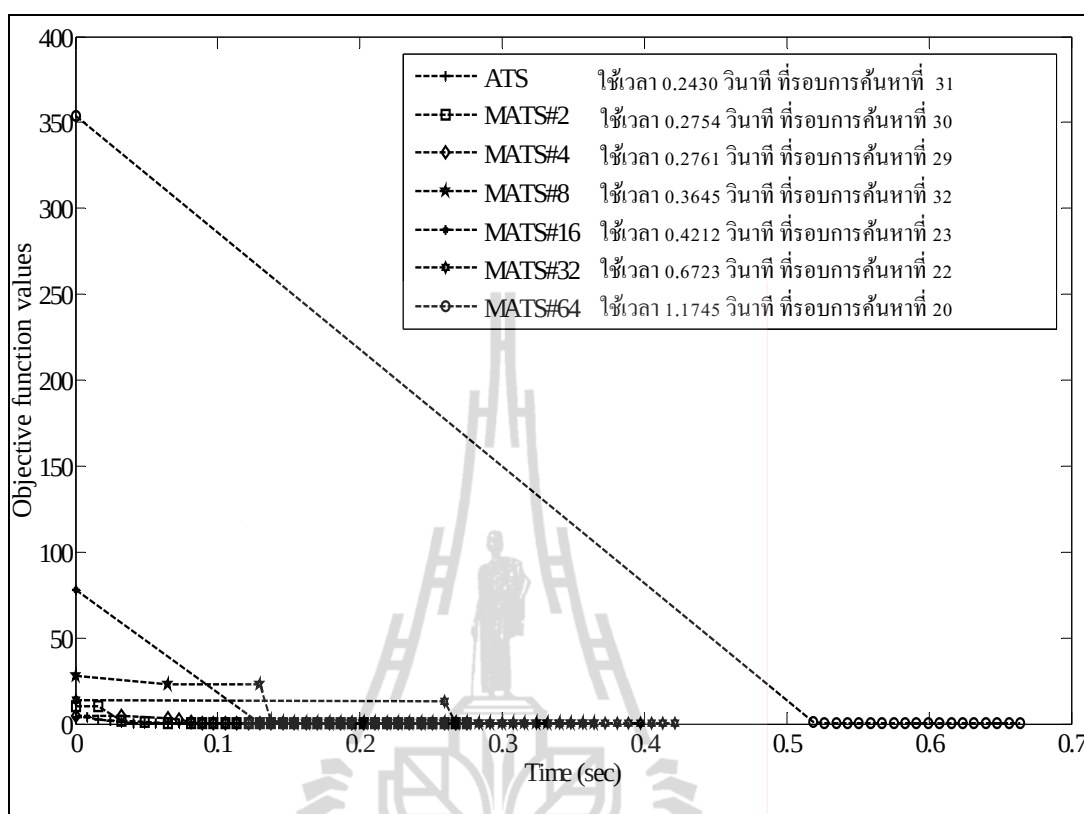
MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 7 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 785.3802 จนกระทั่งรอบการค้นหาที่ 237 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.4241×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 29.2314 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 12 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 546.5490 จนกระทั่งรอบการค้นหาที่ 244 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 4.5600×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 37.2757 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 31 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 429.7522 จนกระทั่งรอบการค้นหาที่ 236 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 5.3005×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 33.6501 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 62 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 483.3091 จนกระทั่งรอบการค้นหาที่ 123 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 9.6385×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 20.9605 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.7 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา $62t$ เมื่อแทนด้วย $t \approx 0.1129$ วินาที คิดเป็นเวลา $62 \times 0.1129 = 6.9998$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ $64t$ จะเป็นเวลาประมาณ $64 \times 0.1129 = 7.2256$ วินาที



รูปที่ 5.8 กราฟการหาคำตอบวงกว้างของ ATS และ MATS ทั้ง 6 ชนิด กับปัญหา ScF2

กราฟการหาคำตอบวงกว้าง ดังรูปที่ 5.8 ต้องการแสดงให้เห็นถึงคุณสมบัติการหาคำตอบวงกว้างของ ATS และ MATS ทุกชนิด โดยยกตัวอย่างฟังก์ชัน ScF2 นอกจากนี้ภาพขยายภายในภาพใหญ่ยังต้องการอธิบายสื่อถึงผู้อ่านให้เข้าใจผลของ SM ที่ทำให้เกิดเวลาคอยซึ่งทำให้ระยะห่างของคำตอบในช่วงแรกยาวนานกว่าช่วงท้าย แต่พอมีการเรียกใช้ DM เป็นระยะ ๆ ช่วงห่างของคำตอบในลำดับคำตอบของ MATS ก็ค่อย ๆ ลดลง จนเหลือเพียง t เดียว โดยที่ t คือเวลาเฉลี่ยที่ ATS ย่อย หรือ ATS แบบเดี่ยว ใช้ในการค้นหาคำตอบแต่ละรอบ

ATS เริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 3.8933 เมื่อดำเนินการค้นหาต่อไปจนถึงรอบการค้นหาที่ 31 จึงพบกับคำตอบด้วยค่าฟังก์ชันวัตถุประสงค์ 9.2335×10^{-5} ซึ่งสอดคล้องกับเงื่อนไขยุติการค้นหา โดยใช้เวลาค้นหาไปทั้งสิ้น 0.2430 วินาที แต่ละคำตอบห่างกันประมาณ $t = 0.2430$ วินาที/31 รอบการค้นหา ≈ 0.0078 วินาที

MATS แบบ 2 เส้นทางค้นหา มีหน่วยค้นหาที่ 1 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 10.2621 จนกระทั่งรอบการค้นหาที่ 30 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.5441×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 0.2754 วินาที

MATS แบบ 4 เส้นทางค้นหา มีหน่วยค้นหาที่ 3 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 4.8644 จนกระทั่งรอบการค้นหาที่ 29 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 6.0615×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 0.2761 วินาที

MATS แบบ 8 เส้นทางค้นหา มีหน่วยค้นหาที่ 6 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 27.7010 จนกระทั่งรอบการค้นหาที่ 32 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 3.1892×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 0.3645 วินาที

MATS แบบ 16 เส้นทางค้นหา มีหน่วยค้นหาที่ 16 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 77.9601 จนกระทั่งรอบการค้นหาที่ 23 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 7.1796×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 0.4212 วินาที

MATS แบบ 32 เส้นทางค้นหา มีหน่วยค้นหาที่ 28 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 13.7801 จนกระทั่งรอบการค้นหาที่ 22 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 7.5479×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 0.6723 วินาที

MATS แบบ 64 เส้นทางค้นหา มีหน่วยค้นหาที่ 59 เป็นผู้พบคำตอบวงกว้าง โดยเริ่มต้นการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์ 353.8478 จนกระทั่งรอบการค้นหาที่ 20 จึงพบกับคำตอบวงกว้างด้วยค่าฟังก์ชันวัตถุประสงค์ 3.6446×10^{-5} โดยใช้เวลาค้นหาไปทั้งสิ้น 1.1745 วินาที

ผลของ DM ต่อเวลาการค้นหาสามารถสังเกตได้จากระยะห่างของคำตอบที่อยู่ติดกันบนเส้นกราฟการลู่เข้าหาคำตอบวงกว้าง ดังแสดงในรูปที่ 5.8 เส้นบนสุดเป็นของ MATS แบบ 64 เส้นทางค้นหา คำตอบจากรอบที่ 1 ห่างจากคำตอบค่าเริ่มต้น เป็นเวลา $20t$ เมื่อแทนด้วย $t \approx 0.0078$ วินาที คิดเป็นเวลา $20 \times 0.0078 = 0.1560$ วินาที และห่างจากคำตอบวงรอบที่ 2 เป็นเวลาประมาณ $64t$ จะเป็นเวลาประมาณ $64 \times 0.0078 = 0.4992$ วินาที

5.5 สรุป

การประเมินสมรรถนะของ MATS ได้ดำเนินงานกับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวน 6 รูปแบบ ได้แก่ (1) ฟังก์ชันโบฮาเชฟสกี (BF) (2) ฟังก์ชันราสทริกิน (RF) (3) ฟังก์ชันชีเคิลฟอกซ์โฮล (SF) (4) ฟังก์ชันชูเบิร์ต (ShF) (5) ฟังก์ชันซวีเฟลแบบที่หนึ่ง (ScF1) และ (6) ฟังก์ชันซวีเฟลแบบที่สอง (ScF2) โดยเปรียบเทียบกับ ATS แบบเดียว ข้อเสนอแนะสำหรับผู้ใช้งานเพื่อเป็นแนวทางการตั้งค่าพารามิเตอร์ PM กรณีปัญหาขนาด 2 มิติ ผู้ใช้สามารถแบ่งปฏิภูมิการค้นหาออกเป็นรูปทรงเรขาคณิต เช่น สี่เหลี่ยมจัตุรัส หรือ สี่เหลี่ยมผืนผ้า ทั้งนี้เพื่อความสะดวก ส่วนพารามิเตอร์ SM กรณีนี้กำหนดคงที่ไว้ 1 รอบการค้นหา เท่ากันทุกหน่วยค้นหา ผู้ใช้อาจปรับเปลี่ยนพารามิเตอร์จำนวนรอบการค้นหาของ SM ได้แตกต่างกันในแต่ละหน่วยค้นหา และพารามิเตอร์สุดท้าย DM การตั้งค่ารอบการเรียกใช้ DM รวมถึงรูปแบบการลดจำนวนหน่วยค้นหา ก็สามารถตั้ง

ได้โดยผู้ใช้ตามความเหมาะสมของปัญหา การเก็บผลการทดสอบทุกครั้ง ทั้งของ MATS และ ATS แบบเดี่ยวจะเริ่มต้นด้วยการสุ่มคำตอบเริ่มต้นภายใต้ปริภูมิการค้นหาแล้ว บันทึกเวลาที่อัลกอริทึมแต่ละตัวใช้เพื่อหาคำตอบวงกว้าง ทำซ้ำจำนวน 50 ครั้ง แล้วหาค่าเฉลี่ยเวลาการค้นหา หลังจากนั้นจึงนำค่าเวลาการค้นหาเฉลี่ยทั้งสองอัลกอริทึมมาหาเปอร์เซ็นต์เวลาการค้นหาที่ลดลง ผลการทดสอบแสดงให้เห็นว่ากรณีฟังก์ชัน BF เวลาการค้นหาของ MATS สามารถลดลงได้ระหว่าง 6% - 131% เมื่อเทียบกับของ ATS แบบเดี่ยว กรณีฟังก์ชัน RF ลดลงได้ระหว่าง 12% - 81% กรณีฟังก์ชัน SF ลดลงได้ระหว่าง 21% - 64% และ กรณีฟังก์ชัน ScF1 MATS สามารถลดลงได้ระหว่าง 9% - 35% เมื่อเทียบกับของ ATS แบบเดี่ยว แต่กรณีของ ScF2 มีเพียง MATS ชนิด 2 หน่วยค้นหาเท่านั้น ที่สามารถลดเวลาการค้นหาเฉลี่ยลงได้น้อยกว่าของ ATS แบบเดี่ยว ส่วน MATS ชนิดอื่น ๆ ไม่สามารถลดเวลาการค้นหาเฉลี่ยให้น้อยลงกว่าของ ATS แบบเดี่ยวได้ ทั้งนี้เนื่องจากฟังก์ชัน ScF2 มีลักษณะพื้นผิวที่ง่ายต่อการค้นหาคำตอบวงกว้าง จึงเป็นโอกาสให้อัลกอริทึมที่มีโครงสร้างอัลกอริทึมไม่ซับซ้อนอย่าง ATS แบบเดี่ยว ทำความเร็วการค้นหาได้ดีกว่าของ MATS นั่นเอง นอกจากนี้ผู้อ่านสามารถศึกษาเพิ่มเติมเกี่ยวกับผลของความสมมาตรต่อสมรรถนะการค้นหาได้จากรายละเอียดเพิ่มเติมของทั้งสามกลไกข้างต้น สามารถศึกษาเพิ่มเติมได้จาก (Kluabwang et al, 2009) หัวข้อถัดไปเป็นตัวอย่างการประยุกต์ใช้ MATS เพื่อแก้ปัญหาทางด้านวิศวกรรมเรื่องการออกแบบตัวควบคุมสำหรับระบบจำลองยานยนต์ยี่ห้อส่วนแห่งอิลลินอยส์ (Illinois roadway simulator : IRS)

บทที่ 6

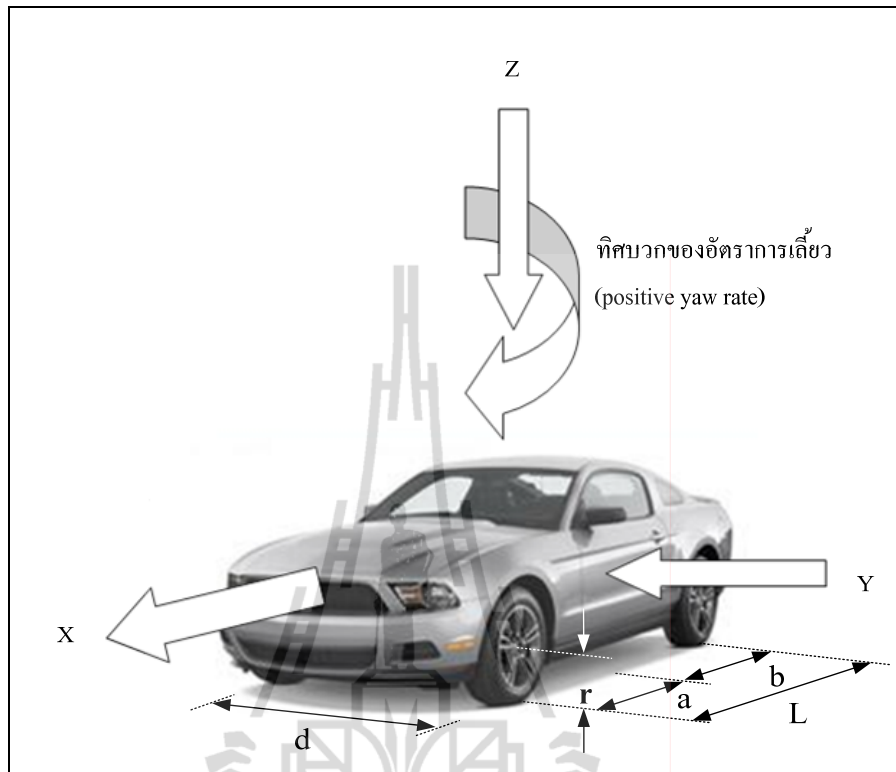
การประยุกต์ใช้งาน MATS กับการออกแบบตัวควบคุม

6.1 กล่าวนำ

การประยุกต์ใช้งาน MATS กับปัญหาจริงในทางวิศวกรรม สำหรับวิทยานิพนธ์ฉบับนี้มีอยู่ด้วยไว้สองปัญหาที่แตกต่างกันได้แก่ ปัญหาการระบุเอกลักษณ์ระบบพลวัตซึ่งจะกล่าวรายละเอียดในบทถัดไป และ ปัญหาการออกแบบตัวควบคุม โดยบทนี้จะกล่าวถึงการประยุกต์ใช้งาน MATS กับปัญหาการออกแบบตัวควบคุมสำหรับระบบจำลองยานยนต์ยี่ห้ออิลลินอยส์ (Illinois roadway simulator : IRS) บทนี้ได้เสนอแนวทางการปรับปรุงสมรรถนะผลตอบสนองของระบบ IRS ไว้สามแนวทาง ได้แก่ แนวทางที่หนึ่งการคาดเดาตัวควบคุมพีดีเข้ากับตัวควบคุมดั้งเดิม แนวทางที่สองการปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิม และแนวทางที่สามการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุมพีดี การดำเนินงานทั้งหมดอาศัยโปรแกรม MATLAB ผลการค้นหาค้นหาด้วย MATS สามารถปรับปรุงผลตอบสนองของระบบ IRS ให้ดีขึ้นได้อย่างน่าพอใจ ซึ่งมีรายละเอียดดังต่อไปนี้

6.2 ระบบจำลองยานยนต์ยี่ห้ออิลลินอยส์

ระบบทดสอบหรือระบบที่ใช้เพื่อศึกษากระบวนการประยุกต์ใช้อัลกอริธึมการค้นหาลักษณะเชิงปรับตัวหลายวิธี เพื่อความสะดวกในการศึกษาจึงควรมีข้อมูลของระบบอย่างเพียงพอ ซึ่งผู้เขียนได้เลือกใช้ระบบจำลองยานยนต์ยี่ห้ออิลลินอยส์ (Illinois roadway simulators : IRS) ดังปรากฏใน Brennan and Alleyne (2000); Brennan and Alleyne (2001) ข้อตกลงเรื่องทิศทางเพื่อการศึกษาในระบบ ปรากฏอยู่ในรูปที่ 6.1



รูปที่ 6.1 ภาพแสดงทิศทางเพื่อการวิเคราะห์ของแบบจำลองย่อยส่วนพาหนะ IRS

การย่อยส่วนขนาดจริงให้เป็นขนาดระดับห้องทดลองสำหรับระบบ IRS ของคณะวิจัยนี้ อ้างอิงทฤษฎีบท Buckingham-Pi ที่ได้รับการยอมรับว่าน่าเชื่อถือและมีประสิทธิภาพใกล้เคียงกับการใช้ของจริงแบบจำลองย่อยส่วนรถยนต์ IRS ที่ได้จึงมีประโยชน์ต่อวงการศึกษาศึกษาและภาคอุตสาหกรรมยานยนต์เป็นอย่างยิ่ง สำหรับวิธีการทำแบบจำลองย่อยส่วนรถยนต์ IRS นี้ เพื่อความสะดวกจึงดำเนินการวิเคราะห์อย่างง่าย ๆ ในแบบสองระดับอิสระ (two degrees of freedom) โดยพิจารณาที่ค่าตำแหน่งแนวระดับ (lateral position) และ ค่ามุมเลี้ยว (yaw angle) นอกจากนี้ยังได้ประมาณให้ค่ามุมเลี้ยวของล้อคู่หน้าเท่ากันและแรงขับของทั้งสี่ล้อก็เท่ากันอีกด้วย ในขั้นต้นเราจะได้แบบจำลองทางคณิตศาสตร์ของแบบจำลองย่อยส่วนรถยนต์ IRS ที่อยู่ในรูปสมการสถานะตามสมการที่ (6.1) ข้างล่างนี้

$$\dot{x} = \begin{bmatrix} -\frac{C_{\alpha f} + C_{\alpha r}}{mU} & -U - \frac{aC_{\alpha f} - bC_{\alpha r}}{mU} \\ -\frac{aC_{\alpha f} - bC_{\alpha r}}{I_z U} & -\frac{a^2 C_{\alpha f} + b^2 C_{\alpha r}}{I_z U} \end{bmatrix} x + \begin{bmatrix} \frac{C_{\alpha f}}{m} & 0 \\ \frac{aC_{\alpha f}}{I_z} & \frac{d}{rI_z} \end{bmatrix} u \quad (6.1)$$

โดยที่ $x = [V \ \dot{\psi}]^T$ $u = [\delta_f \ \Delta T]^T$ m คือมวลของรถยนต์ I_z คือโมเมนต์แรงเฉื่อยตามแนวแกน z U คือความเร็วของรถยนต์ตามแนวยาว ซึ่งประมาณให้มีค่าคงที่ ส่วน V คือความเร็วของรถยนต์ตามแนวขวาง $\dot{\psi}$ คือมุมเลี้ยวของรถยนต์วัดเทียบกับพื้นดิน $C_{\alpha f}$ และ $C_{\alpha r}$ คือแรงเสียดทานขณะเลี้ยวโค้งบริเวณล้อหน้าและล้อหลังตามลำดับ δ_f คือมุมขับเคลื่อนของล้อหน้า ΔT คืออินพุตส่วนต่างของแรงบิดขับเคลื่อน (differential steering torque input) พารามิเตอร์ต่าง ๆ ดังกล่าวข้างต้นล้วนได้จากการทดสอบจริง อย่างเช่น m I_z และ V แต่ก็มีพารามิเตอร์บางค่าที่ต้องทำการวัดด้วยความยากลำบาก อย่างเช่น $C_{\alpha f}$ และ $C_{\alpha r}$ เนื่องจากต้องสื่อถึงความสัมพันธ์ระหว่างแรงขับเคลื่อนล้อกับมุมขับเคลื่อนล้อ ณ มุมขับเคลื่อนล้อเล็ก ๆ ให้นั้นเอง ตัวอย่างค่าพารามิเตอร์ของรถจักรยานและรถยนต์ขับเคลื่อน 4 ล้อ แสดงอยู่ในตารางที่ 6.1

ตารางที่ 6.1 พารามิเตอร์ระบบย่อยส่วนพาหนะบนผิวถนน IRS (ของรถจักรยาน เพื่อเป็นตัวอย่าง)

ประเภทพาหนะ	m (kg)	I_z (kg-m ²)	a (m)	b (m)	$C_{\alpha f}$ (N/rad)	$C_{\alpha r}$ (N/rad)	U_{scale} (m/s)
รถจักรยาน 2WS2WD1997	1.47	0.0236	0.13	0.15	46	46	1.2
รถยนต์ 4WS4WD1998	2.61	0.0656	0.15	0.185	117	97	1.2

แบบจำลองพลวัตของระบบ IRS ดังสมการที่ (6.1) บอกให้ทราบว่าตำแหน่งโพลของระบบอยู่หนึ่ง เมื่อเทียบกับอินพุตขับเคลื่อน จะได้สมการลักษณะเฉพาะ (characteristic equation) ซึ่งแทนด้วย $\Delta(s)$ ดังสมการที่ (6.2)

$$\Delta(s) = s^2 + \left(\frac{(C_{\alpha f} + C_{\alpha r})}{mU} + \frac{(a^2 C_{\alpha f} + b^2 C_{\alpha r})}{I_z U} \right) s + \frac{C_{\alpha f} C_{\alpha r} L^2}{m I_z U^2} - \frac{(a C_{\alpha f} - b C_{\alpha r})}{I_z} = 0 \quad (6.2)$$

ฟังก์ชันถ่ายโอนระหว่างอินพุตการขับเคลื่อนของล้อหน้ากับเอาต์พุตอัตราเลี้ยวแสดงได้ด้วยสมการที่ (6.3) และฟังก์ชันถ่ายโอนของอินพุตส่วนต่างของแรงบิดขับเคลื่อนไปยังอัตราเลี้ยวเอาต์พุตอัตราเลี้ยวแสดงได้ด้วยสมการที่ (6.4) ตามลำดับ

$$\frac{\dot{\psi}(s)}{\delta_f(s)} = \frac{\frac{aC_{af}}{I_z}s + \frac{C_{af}C_{ar}L}{mUI_z}}{\Delta(s)} \quad (6.3)$$

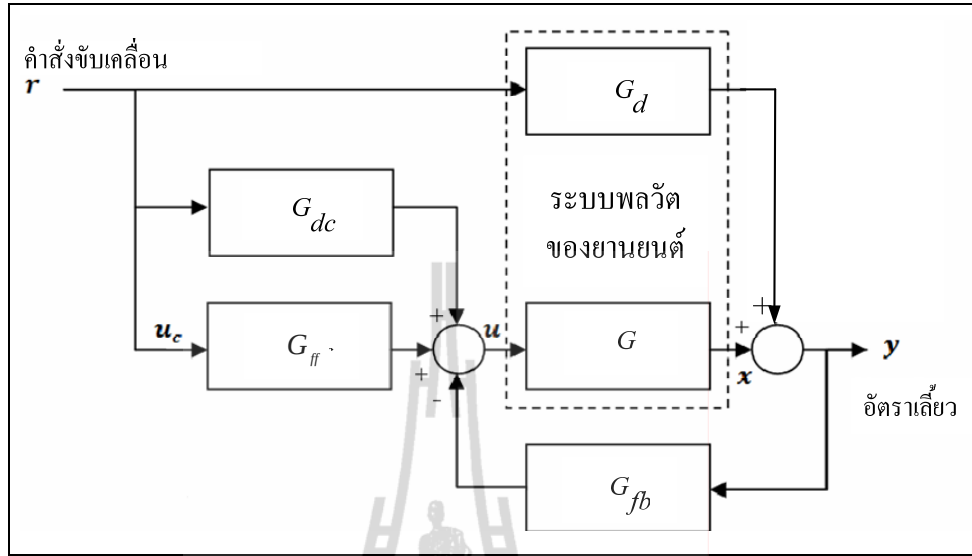
$$\frac{\dot{\psi}(s)}{\Delta T(s)} = \frac{\frac{d}{rI_z}s + \frac{d(C_{af} + C_{ar})}{rmUI_z}}{\Delta(s)} \quad (6.4)$$

สมการที่ (6.3) และ (6.4) เมื่อแทนค่าพารามิเตอร์จากตารางที่ (6.1) ของรถจักรยานร่วมกับ การประมาณค่าจากการทดสอบจริง (Brennan and Alleyne, 2001) จะได้แบบจำลองระบบพลวัต ของรถจักรยานย่อยส่วน เป็นแบบฟังก์ชันเศษส่วนพหุนามอันดับสอง ดังสมการที่ (6.5) และ (6.6) ตามลำดับ

$$G_f = \frac{\dot{\psi}(s)}{\delta_f(s)} = \frac{81.3s + 679.9}{s^2 + 18.97s + 90.54} \quad (6.5)$$

$$G_T = \frac{\dot{\psi}(s)}{\Delta T(s)} = \frac{7.097s + 58.41}{s^2 + 18.97s + 90.54} \quad (6.6)$$

ในทางปฏิบัติ Brennan and Alleyne ได้สร้างชุดศึกษา สำหรับกรณีของแบบจำลองส่วน รถจักรยาน แสดงด้วยแผนภาพ ดังรูปที่ 6.2 ที่มีอินพุตเป็นคำสั่งขับเคลื่อน (steering command) และ เอาต์พุตเป็นอัตราเลี้ยว (yaw rate)



รูปที่ 6.2 แผนภาพบล็อกของแบบจำลองย่อยส่วนรถจักรยาน IRS พร้อมระบบควบคุม

แผนภาพของระบบ IRS สำหรับรถจักรยาน แสดงดังรูปที่ 6.2 ประกอบด้วยสองส่วนหลัก ได้แก่ ส่วนของพลาเน็ตซึ่งมีอยู่ด้วยกันสองบล็อกคือบล็อก G_d ที่เป็นการสัมพันธ์จากอินพุตขับเคลื่อนหน้าไปเอาต์พุต กับบล็อก G ที่เป็นการสัมพันธ์จากอินพุตแรงบิดขับเคลื่อนไปเอาต์พุต ส่วนที่สองได้แก่ระบบควบคุมมีอยู่ด้วยกันสามบล็อก ได้แก่บล็อก G_{dc} ทำหน้าที่ชดเชยการรบกวน บล็อก G_{ff} ในส่วนป้อนไปหน้า และบล็อก G_{fb} ในส่วนป้อนกลับ สมการที่ (6.7) แสดงที่มาของ บล็อก G และ สมการที่ (6.8) แสดงที่มาของบล็อก G_d ตามลำดับ

$$G = \frac{B(s)}{A(s)} = \begin{bmatrix} \text{อัตราทดรอบ} \\ \text{เฟืองขับ} \end{bmatrix} \cdot \begin{bmatrix} \text{พลวัตของ} \\ \text{แรงบิดอินพุต} \end{bmatrix} \cdot \begin{bmatrix} \text{พลวัตของจักรยาน} \\ \text{กับอินพุตแรงบิดขับ} \end{bmatrix}$$

$$= \left[\frac{1}{1.5} \right] \cdot \left[\frac{(10(2\pi))^2}{s^2 + 10(2\pi)(0.7)s + (10(2\pi))^2} \right] \cdot \left[\frac{7.097s + 58.41}{s^2 + 18.97s + 90.54} \right] \quad (6.7)$$

$$G_d = \frac{B_d(s)}{A_d(s)} = \begin{bmatrix} \text{อัตราขยาย} \\ \text{การเชื่อมต่อขับ} \end{bmatrix} \cdot \begin{bmatrix} \text{พลวัตของ} \\ \text{เซอร์โวขับ} \end{bmatrix} \cdot \begin{bmatrix} \text{พลวัตของจักรยาน} \\ \text{กับอินพุตขับล้อหน้า} \end{bmatrix}$$

$$= \left[\frac{1}{1.3} \right] \cdot \left[\frac{(5(2\pi))^2}{s^2 + 5(2\pi)(0.7)s + (5(2\pi))^2} \right] \cdot \left[\frac{81.3s + 679.9}{s^2 + 18.97s + 90.54} \right] \quad (6.8)$$

ระบบควบคุมดั้งเดิมออกแบบตามหลักการของตัวควบคุมอิงแบบจำลอง (model reference controller) ทำให้ได้ค่าของบล็อก G_{dc} G_{ff} และ G_{fb} ดังสมการที่ (6.9) (6.10) และ (6.11)

$$G_{dc} = -\frac{AB_d}{BA_d} = -\frac{G_d(s)}{G(s)} \quad (6.9)$$

$$G_{ff} = \frac{T(s)}{R(s)} \quad (6.10)$$

$$G_{fb} = \frac{S(s)}{R(s)} \quad (6.11)$$

เมื่อฟังก์ชันถ่ายโอน $R(s)$ $S(s)$ และ $T(s)$ แสดงดังสมการที่ (6.12) - (6.14) ตามลำดับ

$$R(s) = s^3 + 155.3s^2 + 3727s + 20714 \quad (6.12)$$

$$S(s) = -5.181s^3 + 49.98s^2 + 5294s + 48142 \quad (6.13)$$

$$T(s) = 103s^2 + 18542s + 834385 \quad (6.14)$$

พลาตต์ที่แสดงในสมการที่ (6.7) และ (6.8) สามารถเขียนได้ใหม่ในรูปแบบฟังก์ชันถ่ายโอนดังสมการที่ (6.15) และ (6.16) ตามลำดับ

$$G = \frac{2.802 \times 10^4 s + 2.306 \times 10^5}{1.5s^4 + 94.43s^3 + 7.308 \times 10^4 s^2 + 1.183 \times 10^5 s + 5.361 \times 10^5} \quad (6.15)$$

$$G_d = \frac{8.024 \times 10^4 s + 6.710 \times 10^5}{1.3s^4 + 53.250s^3 + 1.943 \times 10^3 s^2 + 2.693 \times 10^4 s + 1.162 \times 10^5} \quad (6.16)$$

ตัวควบคุม G_{ff} สามารถคำนวณได้ตามสมการที่ (6.9) โดยอาศัยค่าจากสมการที่ (6.7) และ (6.8) ได้ผลลัพธ์แสดงดังสมการที่ (6.17) ส่วนของตัวควบคุม G_{fb} และ G_{dc} สามารถคำนวณได้ตามสมการที่ (6.10) และ (6.11) โดยอาศัยค่าจากสมการที่ (6.12) - (6.14) ได้ผลลัพธ์แสดงดังสมการที่ (6.18) และ (6.19) ตามลำดับ

$$G_{ff} = \frac{103s^2 + 1.8542 \times 10^4 s + 8.34385 \times 10^5}{s^3 + 155.3s^2 + 3.727 \times 10^3 s + 2.0714 \times 10^4} \quad (6.17)$$

$$G_{fb} = \frac{-5.181s^3 + 49.98s^2 + 5.294 \times 10^3 s + 4.8142 \times 10^4}{s^3 + 155.3s^2 + 3.727 \times 10^3 s + 2.0714 \times 10^4} \quad (6.18)$$

$$G_{dc} = \frac{-1.2036 \times 10^5 s^5 - 8.584 \times 10^6 s^4 - 6.497 \times 10^8 s^3}{3.6426 \times 10^4 s^5 + 1.792 \times 10^6 s^4 + 6.671 \times 10^7 s^3} \\ = \frac{-1.44 \times 10^{10} s^2 - 1.224 \times 10^{11} s - 3.598 \times 10^{11}}{+1.202 \times 10^9 s^2 + 9.464 \times 10^9 s + 2.679 \times 10^{10}} \quad (6.19)$$

พิจารณาระบบดังแผนภาพรูปที่ 6.2 สามารถเขียนแสดงฟังก์ชันถ่ายโอนของระบบ $G_{irs}(s)$ ได้ดังต่อไปนี้

$$G_{irs}(s) = \frac{Y(s)}{R(s)} = \frac{G_d + G \cdot G_{dc} + G \cdot G_{ff}}{1 + G \cdot G_{fb}} \quad (6.20)$$

เมื่อนำข้อมูลฟังก์ชันถ่ายโอนดังสมการที่ (6.15) - (6.19) แทนลงในสมการที่ (6.20) จะได้ฟังก์ชันถ่ายโอน $G_{irs}(s)$ มีรายละเอียดดังสมการที่ (6.21) ซึ่งมีอันดับ 27 และพจน์ตัวตั้งมีอันดับ 23 (มีซีโร 23 ตัวและโพล 27 ตัว) และมีโพลซีโรแสดงในตารางที่ 6.2

$$\begin{aligned}
G_{irs} = & \frac{3.075 \times 10^{11} s^{23} + 1.721 \times 10^{14} s^{22} + 4.525 \times 10^{16} s^{21}}{1.598 \times 10^5 s^{27} + 9.423 \times 10^7 s^{26} + 2.671 \times 10^{10} s^{25}} \\
& \frac{+7.624 \times 10^{18} s^{20} + 9.327 \times 10^{20} s^{19} + 8.798 \times 10^{22} s^{18}}{+4.957 \times 10^{12} s^{24} + 6.805 \times 10^{14} s^{23} + 7.331 \times 10^{16} s^{22}} \\
& \frac{+6.618 \times 10^{24} s^{17} + 4.055 \times 10^{26} s^{16} + 2.048 \times 10^{28} s^{15}}{+6.423 \times 10^{18} s^{21} + 4.678 \times 10^{20} s^{20} + 2.873 \times 10^{22} s^{19}} \\
& \frac{+8.590 \times 10^{29} s^{14} + 3.006 \times 10^{31} s^{13} + 8.779 \times 10^{32} s^{12}}{+1.503 \times 10^{24} s^{18} + 6.729 \times 10^{25} s^{17} + 2.590 \times 10^{27} s^{16}} \\
& \frac{+2.136 \times 10^{34} s^{11} + 4.307 \times 10^{35} s^{10} + 7.143 \times 10^{36} s^9}{+8.580 \times 10^{28} s^{15} + 2.445 \times 10^{30} s^{14} + 5.979 \times 10^{31} s^{13}} \\
& \frac{+9.645 \times 10^{37} s^8 + 1.048 \times 10^{39} s^7 + 9.037 \times 10^{39} s^6}{+1.251 \times 10^{33} s^{12} + 2.225 \times 10^{34} s^{11} + 3.346 \times 10^{35} s^{10}} \\
& \frac{+6.075 \times 10^{40} s^5 + 3.107 \times 10^{41} s^4 + 1.165 \times 10^{42} s^3}{+4.219 \times 10^{36} s^9 + 4.417 \times 10^{37} s^8 + 3.797 \times 10^{38} s^7} \\
& \frac{+3.016 \times 10^{42} s^2 + 4.813 \times 10^{42} s}{+2.643 \times 10^{39} s^6 + 1.462 \times 10^{40} s^5 + 6.265 \times 10^{40} s^4} \\
& \frac{+3.565 \times 10^{42}}{+2.001 \times 10^{41} s^3 + 4.480 \times 10^{41} s^2 + 6.265 \times 10^{41} s + 4.115 \times 10^{41}}
\end{aligned} \tag{6.21}$$

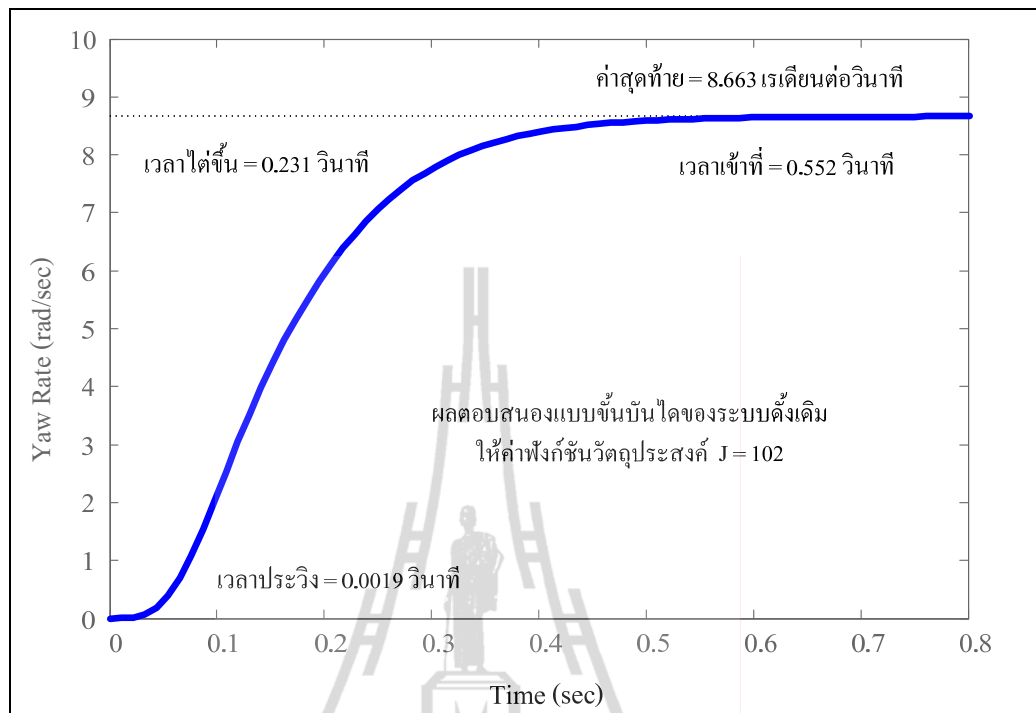
ตารางที่ 6.2 โพลและซีโรของระบบ $G_{irs}(s)$

โพล	ซีโร
-127.3012	-127.3012
-115.8802	-90.926
-21.9887 + 58.8502i	-89.0961
-21.9887 - 58.8502i	-21.9887 + 58.8501i
-21.9888 + 58.8500i	-21.9887 - 58.8501i
-21.9888 - 58.8500i	-21.9887 + 58.8501i
-18.5383 + 47.4510i	-21.9887 - 58.8501i
-18.5383 - 47.4510i	-11.0577 + 29.5344i

ตารางที่ 6.2 โพลและซีโรของระบบ $G_{irs}(s)$ (ต่อ)

โพล	ซีโร
$-10.9913 + 29.4219i$	$-11.0577 - 29.5344i$
$-10.9913 - 29.4219i$	$-10.9234 + 29.3097i$
$-10.9906 + 29.4219i$	$-10.9234 - 29.3097i$
$-10.9906 - 29.4219i$	-19.7672
-28.1278	$-9.7597 + 0.7484i$
-19.7672	$-9.7597 - 0.7484i$
$-14.4695 + 1.8194i$	$-9.5470 + 0.9747i$
$-14.4695 - 1.8194i$	$-9.5470 - 0.9747i$
$-10.2948 + 0.5874i$	$-9.2581 + 0.8693i$
$-10.2948 - 0.5874i$	$-9.2581 - 0.8693i$
$-9.8044 + 1.2121i$	$-9.3953 + 0.2974i$
$-9.8044 - 1.2121i$	$-9.3953 - 0.2974i$
$-9.2252 + 1.2026i$	-8.3718
$-9.2252 - 1.2026i$	$-8.1530 + 0.0785i$
$-8.7626 + 0.7548i$	$-8.1530 - 0.0785i$
$-8.7626 - 0.7548i$	
$-8.1946 + 0.2609i$	
$-8.1946 - 0.2609i$	
-8.0341	

เมื่อจำลองระบบ $G_{irs}(s)$ ตามสมการ (6.21) เพื่อตรวจสอบผลตอบสนองต่อสัญญาณขั้นบันไดหนึ่งหน่วย ได้ผลดังแสดงในรูปที่ 6.3 ซึ่งมีเวลาไต่ระดับ (rise time) 0.231 วินาที เวลาเข้าที่ (settling time) 0.552 วินาที เวลาประวิง (delay time) 0.0019 วินาที และ ระดับผลตอบสนองในสภาวะอยู่ตัว 8.663 เรเดียนต่อวินาที



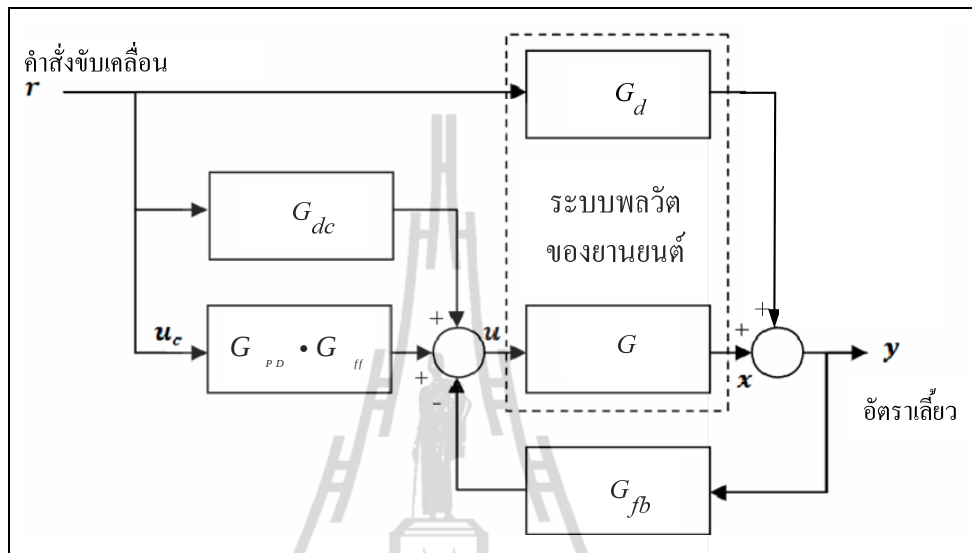
รูปที่ 6.3 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs}(s)$

นับถึงตอนนี้เราได้พลาเน็ตเพื่อใช้ศึกษาพร้อมค่าพารามิเตอร์ครบทุกตัว ดังปรากฏในสมการ (6.15) และ (6.16) นอกจากนี้ยังมี ตัวควบคุมดั้งเดิม ทราบค่าพารามิเตอร์ครบทุกตัวเช่นกัน ดังปรากฏในสมการ (6.17) (6.19) ซึ่งจะใช้อ้างอิงเพื่อการออกแบบตัวควบคุมใหม่ด้วยอัลกอริทึม การค้นหาแบบตามูเชิงปรับตัวหลายวิธีให้ระบบ $G_{irs}(s)$ มีสมรรถนะด้านผลตอบสนองแบบขั้นบันไดที่ดีขึ้นกว่าเดิม โดยหัวข้อถัดไปจะกล่าวถึงแนวทางการออกแบบตัวควบคุมใหม่ด้วยการแทรกตัวควบคุม PD เข้ากับบล็อก G_{ff} หรือการต่อคาสเคด (cascade) นั้นเอง

6.3 การคาสเคดตัวควบคุม PD เข้ากับบล็อก G_{ff}

เป้าหมายในการออกแบบตัวควบคุมใหม่นี้ เพื่อลดเวลาประวิงลง และเพิ่มความเร็วในการตอบสนองให้ไวขึ้น ซึ่งก็เป็นที่ยอมรับกันว่าตัวควบคุม PD สามารถช่วยแก้ปัญหาเวลาประวิงด้วยพจน์ D และลดเวลาไต่ระดับหรือเพิ่มความเร็วตอบสนองด้วยพจน์ P รายละเอียดเพิ่มเติมปรากฏอยู่ใน Sujitjorn et al. (2010) รูปแบบฟังก์ชันถ่ายโอนของตัวควบคุม PD หรือ G_{PD} ที่จะใช้ในการออกแบบครั้งนี้ แสดงอยู่ในสมการที่ (6.22) และตำแหน่งของ G_{PD} คาสเคดอยู่กับบล็อกของ G_{ff} แสดงดังรูปที่ 6.4

$$G_{PD} = K_D s + K_P \quad (6.22)$$



รูปที่ 6.4 ระบบ IRS ที่มีการควบคุมด้วยตัวควบคุม PD หรือ G_{PD} กับบล็อก G_{ff}

MATS ถูกพัฒนาขึ้นด้วยภาษา MATLABTM เพื่อความสะดวกของกระบวนการค้นหา จึงสร้างฟังก์ชันวัตถุประสงค์ด้วยภาษา MATLABTM เช่นกัน และฟังก์ชันวัตถุประสงค์หมายถึง ฟังก์ชันที่ใช้สำหรับการประเมินสิ่งที่สนใจจากอินพุตที่รับเข้ามาแล้วให้อาต์พุตออกไปเป็นตัวชี้วัด ในกระบวนการค้นหาของเทคนิควิธีการปัญหาการหาค่าเหมาะที่สุดเชิงศึกษาคำนึง สิ่งที่น่าสนใจ สำหรับปัญหานี้คือผลตอบสนองแบบขั้นบันไดของระบบวงรอบปิด G_{irs} ได้อาศัยคำสั่ง step ของ MATLABTM Control System Toolbox เพื่อจำลองผลตอบสนองแบบขั้นบันไดจากพารามิเตอร์ของ ระบบที่ป้อนให้ เมื่อได้ผลตอบสนองออกมาจาก Toolbox ก็นำมาผลตอบสนองที่ได้มาวิเคราะห์ เวลาไต่ระดับ (t_r) เวลาเข้าที่ (t_s) เวลาประวิง (t_d) และ เปอร์เซนต์การพุ่งเกิน (P.O.) ผลตอบสนอง แบบขั้นบันไดที่ฟังก์ชันภายหลังการเพิ่มตัวควบคุม PD ได้แก่ การลดลงของเวลาไต่ระดับ เวลา เข้าที่ เวลาประวิง และ ปราศจากการพุ่งเกิน แต่การสร้างฟังก์ชันวัตถุประสงค์โดยทำการรวมทั้งสี่ พารามิเตอร์ไว้ในสมการเดียวกัน จำเป็นต้องทำให้ทั้งสี่พารามิเตอร์มีหน่วยเดียวกันหรือทำให้ ไร้นหน่วย (dimensionless) ก็ได้ ในที่นี้เลือกใช้วิธีการทำให้ไร้นหน่วย ให้อยู่ในรูปของเปอร์เซนต์ และ ได้สร้างฟังก์ชันวัตถุประสงค์จากข้อกำหนดสมรรถนะสามประการ คือ t_r , t_s และ P.O. โดยมีความคาดหวังว่า t_d จะลดลงด้วยเป็นผลพลอยได้

กรณีของเวลาไต่ระดับจะหาเปอร์เซ็นต์จากสัดส่วนของค่าเวลาไต่ระดับใหม่ต่อค่าเวลาไต่ระดับดั้งเดิม แสดงดังสมการที่ (6.23)

$$t_{rj} = \frac{t_{r_new}}{t_{r_old}} \times 100 \quad (6.23)$$

เมื่อ t_{rj} คือ เปอร์เซนต์ของเวลาไต่ระดับใหม่ต่อเวลาไต่ระดับดั้งเดิม
 t_{r_new} คือ เวลาไต่ระดับใหม่ที่ได้จากการจำลอง
 t_{r_old} คือ เวลาไต่ระดับดั้งเดิมของระบบก่อนการปรับปรุง กำหนดไว้ที่ 0.231 วินาที

จากรูปที่ (6.3) หากระบบใหม่ที่ได้มีผลตอบสนองที่รวดเร็วขึ้นยอมทำให้ t_r สั้นกว่าเดิม ทำให้ได้ค่า t_{rj} น้อยกว่าหนึ่งร้อยเปอร์เซ็นต์ ตัวอย่างเช่น $t_{r_new} = 0.161$ วินาที เมื่อแทนลงในสมการที่ (6.23) จะได้ดังสมการที่ (6.24)

$$t_{rj} = \frac{0.161}{0.231} \times 100 = 69.70\% \quad (6.24)$$

แต่ถ้าหากระบบใหม่ที่ได้มีผลตอบสนองที่ช้าลงยอมทำให้ t_r ยาวกว่าเดิม ส่งผลทำให้ได้ค่า t_{rj} มีค่าเกินหนึ่งร้อยเปอร์เซ็นต์ ซึ่งถือว่าเป็นผลตอบสนองที่ไม่ต้องการในที่นี้ ตัวอย่างเช่น $t_{r_new} = 0.242$ วินาที เมื่อแทนลงในสมการที่ (6.23) จะได้ดังสมการที่ (6.25)

$$t_{rj} = \frac{0.242}{0.231} \times 100 = 104.76\% \quad (6.25)$$

เงื่อนไขเพื่อคัดสรรคุณสมบัติ t_r ที่ต้องการถูกกำหนดด้วยตรรกะข้างล่างนี้ “ถ้า $t_{rj} < 100$ ให้ยอมรับค่า t_{rj} นั่นคือ $t_{rj} = t_{rj}$ มิเช่นนั้นให้ปฏิเสธด้วยการปรับโทษเป็น $t_{rj} = 100$ ”

กรณีของเวลาเข้าที่ หาเปอร์เซ็นต์จากสัดส่วนของค่าเวลาเข้าที่ใหม่ต่อค่าเวลาไต่ระดับดั้งเดิม แสดงดังสมการที่ (6.26)

$$t_{sj} = \frac{t_{s_new}}{t_{r_old}} \times 100 \quad (6.26)$$

เมื่อ t_{sj} คือ เปอร์เซนต์ของเวลาเข้าที่ใหม่ต่อเวลาไต่ระดับดั้งเดิม
 t_{s_new} คือ เวลาเข้าที่ใหม่ที่ได้จากการจำลอง
 t_{r_old} คือ เวลาไต่ระดับดั้งเดิมของระบบก่อนการปรับปรุง กำหนดไว้ที่ 0.231 วินาที

ปกติแล้ว t_s จะมีค่าไม่เกินห้าเท่าของ t_r จึงกำหนดเป็นเงื่อนไขเพื่อคัดสรรคุณสมบัติ t_{sj} ที่ได้จากสมการที่ (6.26) ด้วยตรรกะข้างล่างนี้ “ถ้า $t_{sj} < 5 * t_{rj}$ ให้ยอมรับค่า t_{sj} นั่นคือ $t_{sj} = 1$ มิเช่นนั้นให้ปฏิเสธด้วยการปรับโทษ t_{sj} เป็น $t_{sj} = 100$ ”

กรณีของเปอร์เซนต์การพุ่งเกินเป็นค่าเปอร์เซนต์อยู่แล้วจึงไม่ต้องเปลี่ยนแปลง แต่เพิ่มเงื่อนไขการคัดสรรค่า $P.O.$ ด้วยตรรกะข้างล่างนี้ “ถ้า $P.O. \leq 0$ ให้ยอมรับค่า $P.O.$ นั่นคือ $P.O. = 1$ มิเช่นนั้นให้ปฏิเสธด้วยการปรับโทษ $P.O.$ เป็น $P.O. = 100$ ”

ในที่นี้ละเลยผลของเวลาประวิงไปก่อน ดังนั้นขั้นตอนต่อไปจึงดำเนินการสร้างฟังก์ชันวัตถุประสงค์ด้วยผลรวมทางพีชคณิตเชิงเส้นของทั้งสามพารามิเตอร์ คือ t_{rj} t_{sj} และ $P.O.$ สำหรับ MATS ดำเนินการค้นหาคำตอบที่เหมาะสมที่สุดสามารถเขียนได้ดังสมการที่ (6.27)

$$J = \alpha_1 t_{sj} + \alpha_2 t_{rj} + \alpha_3 P.O. \quad (6.27)$$

เมื่อ J คือ ค่าฟังก์ชันวัตถุประสงค์
 t_{rj} คือ เปอร์เซนต์ของเวลาไต่ระดับ
 t_{sj} คือ เปอร์เซนต์ของเวลาเข้าที่
 $P.O.$ คือ เปอร์เซนต์การพุ่งเกิน
 α_1 α_2 และ α_3 คือ สัมประสิทธิ์ถ่วงน้ำหนักของ t_{rj} t_{sj} และ $P.O.$ ตามลำดับ

สำหรับการออกแบบด้วยวิธีนี้ให้ความสำคัญกับทั้งสามพารามิเตอร์เท่า ๆ กันจึงกำหนดให้ขนาดของสัมประสิทธิ์ทั้งสามตัวเท่ากัน จึงกำหนดให้เป็น $\alpha_1 = \alpha_2 = \alpha_3 = 1$ จะได้ฟังก์ชันวัตถุประสงค์ที่จะใช้ในการออกแบบตัวควบคุม PD นี้ แสดงดังสมการที่ (6.28)

$$J = t_{sj} + t_{rj} + P.O. \quad (6.28)$$

ทั้งนี้หากผู้อ่านต้องการเพิ่มความสำคัญของพารามิเตอร์ตัวใดก็เพียงปรับขนาดสัมประสิทธิ์ α_i ให้สูงกว่าตัวอื่น ในทางตรงกันข้ามหากต้องการลดความสำคัญของพารามิเตอร์ ตัวใดก็เพียงปรับขนาดสัมประสิทธิ์นั้นให้ต่ำกว่าตัวอื่นนั่นเอง ขั้นตอนการประเมินค่าของฟังก์ชันวัตถุประสงค์สามารถสรุปได้ดังรูปที่ 6.5

ขั้นตอนการคำนวณของฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 1 (การเริ่มต้น)

รับค่า K_p และ K_D

ขั้นตอนที่ 2 (จำลองการทำงานระบบ IRS ในโปรแกรม MATLAB)

จำลองการทำงานระบบด้วยพารามิเตอร์เดิมร่วมกับตัวควบคุม PD โดยทดสอบกับสัญญาณขั้นบันได

ขั้นตอนที่ 3 (สังเคราะห์พารามิเตอร์ที่สนใจ)

นำผลการจำลองที่เป็นอัตราเลี้ยวมาหาค่าเวลาไต่ขึ้น (t_u) เวลาเข้าที่ (t_{sj}) และเปอร์เซ็นต์ฟุ้งเกิน (P.O.)

ขั้นตอนที่ 4 (คำนวณค่าฟังก์ชันวัตถุประสงค์)

คำนวณค่าฟังก์ชันวัตถุประสงค์ $J = t_u + t_{sj} + P.O.$ แล้วรายงานค่า J

รูปที่ 6.5 ขั้นตอนการประเมินค่าของฟังก์ชันวัตถุประสงค์ในการค้นหาตัวควบคุม PD

ภายหลังที่ได้ J ตามสมการที่ (6.28) ก็นำค่า J ที่ได้มาผ่านเงื่อนไขคัดสรร J อีกครั้ง ซึ่งโดยปกติแล้วค่า J ที่ดีควรมีขนาดไม่เกิน 102 ตรรกะที่ใช้คัดสรรแสดงดังต่อไปนี้ “ถ้า $J < 102$ ให้ยอมรับค่า J นั่นคือ $J = J$ มิเช่นนั้นให้ปฏิเสธด้วยการปรับโทษ J โดยให้ $J = 1000$ ”

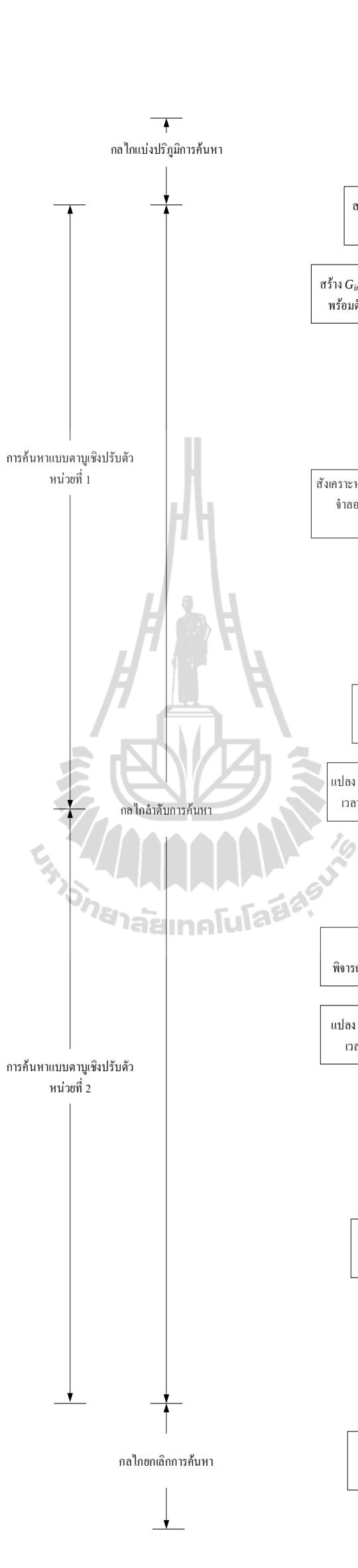
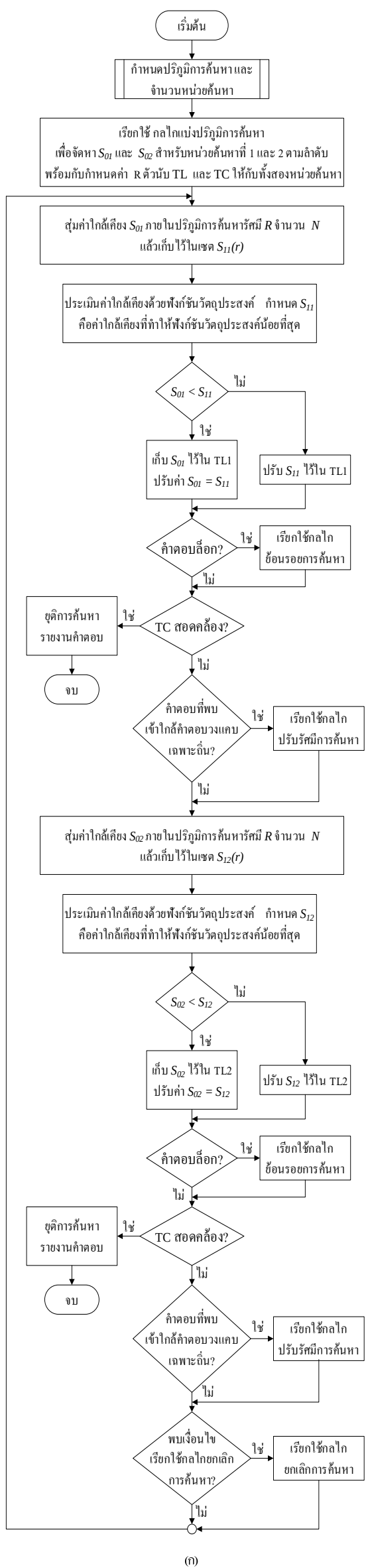
จากรูปที่ 6.5 เราจะได้ J ของปัญหาการหาค่าเหมาะที่สุด ซึ่งสามารถจัดให้อยู่ในรูปทั่วไปได้ดังความสัมพันธ์ (6.29)

$$\begin{aligned}
 &\min J \\
 &\text{s.t. } P.O. \leq 0\% \\
 &\quad t_r < 0.231\text{sec} \\
 &\quad t_s < 5 \times 0.231\text{sec}
 \end{aligned} \tag{6.29}$$

ปัญหาตามสมการที่ (6.29) เป็นปัญหาแบบสองมิติ คือมีอินพุตสองค่าคือ K_p และ K_D จึงเลือกที่จะใช้ MATS แบบ 2 เส้นทางค้นหา ดำเนินการหาคำตอบ อย่างไรก็ตาม MATS แบบที่มีจำนวนเส้นทางค้นหามากกว่านี้ก็สามารถดำเนินการได้ เช่นเดียวกับปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่ที่ได้นำเสนอไปแล้วในบทที่ 4 การกำหนดปริภูมิการค้นหาคำเนินการด้วยการศึกษาและทดสอบเบื้องต้นทำให้ได้ค่าปริภูมิการค้นหาคือ $[K_p \ K_D] = [2 \ 2; 0 \ 0]$ หมายความว่า K_p มีขอบเขตบนเท่ากับ 2 และขอบเขตล่างเท่ากับ 0 ส่วนของ K_D ก็เช่นเดียวกัน

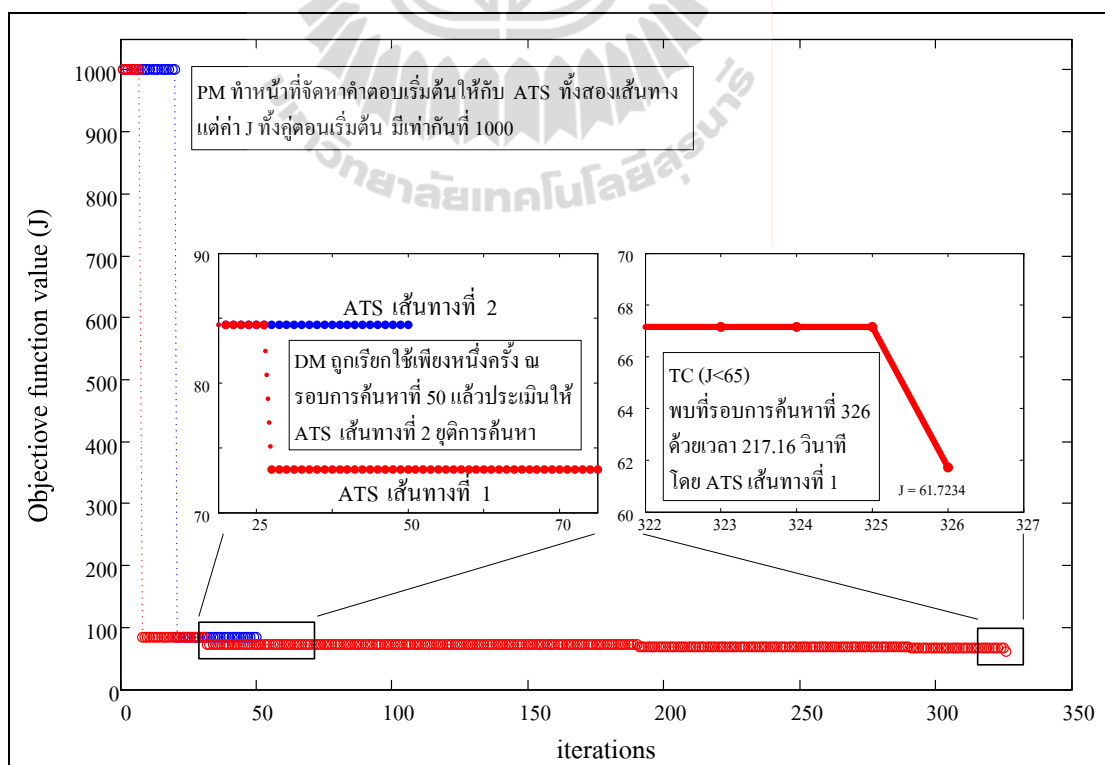
ค่าพารามิเตอร์ของ ATS ย่อย ทั้งสองเส้นทางค้นหาจะตั้งไว้เหมือนกันทุกประการ โดยกำหนดตามคำแนะนำการใช้งาน ATS (Sujijorn et al., 2006) ดังนี้ รัศมีการค้นหาเริ่มต้น $R = 0.2$ คิดเป็น 10% ของปริภูมิการค้นหา ค่าตอบข้างเคียงจำนวน 5 ค่า การหาคำตอบสูงสุดเท่ากับจำนวน 5 ครั้ง คำตอบย้อนหลังเลือกที่ลำดับ 5 การปรับรัศมีการใช้สองช่วง ได้แก่ ช่วงที่หนึ่งเมื่อค่า J น้อยกว่า 90 ให้ใช้รัศมีการค้นหา $R = 0.1$ และ ช่วงที่สองเมื่อค่า J น้อยกว่า 70 ให้ใช้รัศมีการค้นหา $R = 0.05$ เหนี่ยวนำการค้นหาด้วยจำนวนรอบสูงสุด 10,000 รอบ และ ค่า J น้อยกว่า 65

สำหรับค่าพารามิเตอร์ของ MATS แบบ 2 เส้นทางค้นหา PM แบ่งปริภูมิการค้นหาคือเป็นปริภูมิการค้นหาย่อยที่ 1 คือ $[2 \ 2; 1 \ 0]$ และปริภูมิการค้นหาย่อยที่ 2 คือ $[1 \ 2; 0 \ 0]$ ส่วน DM ตั้งค่าการเรียกใช้ไว้หนึ่งครั้งในรอบการค้นหาที่ 50 แล้วกำจัดออกหนึ่งหน่วยค้นหาเพื่อให้เหลือเพียงหน่วยค้นหาเดียว รายละเอียดขั้นตอนการทำงานของ MATS และฟังก์ชันวัตถุประสงค์แสดงด้วยแผนภูมิการทำงานดังรูปที่ 6.6

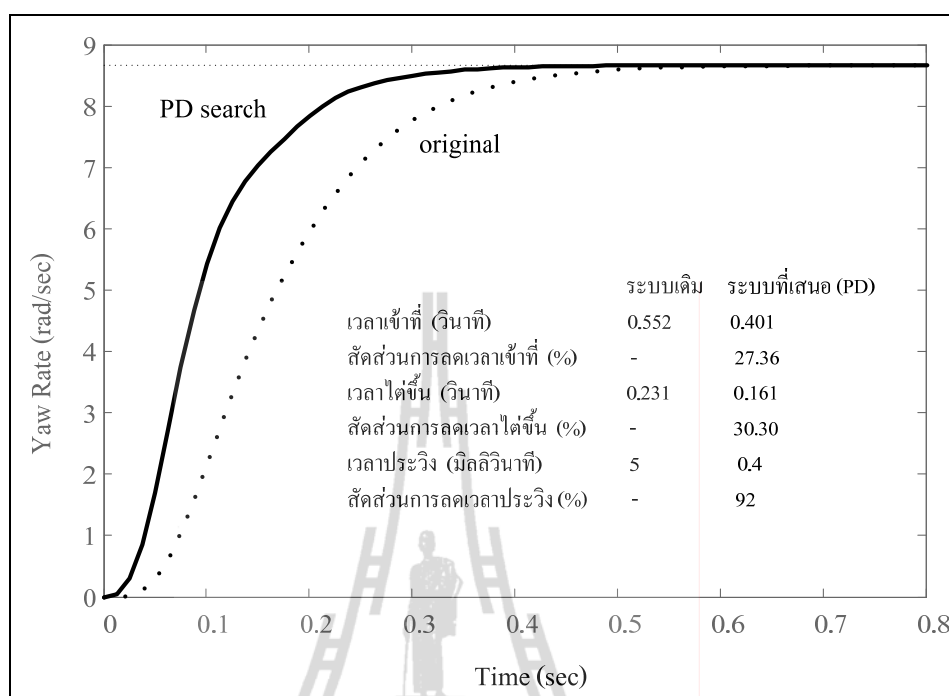


รูปที่ 6.6 (ก) แผนภูมิการทำงานของการค้นหาแบบตามเชิงปรับตัวหลายวิธีเพื่อการออกแบบตัวควบคุมพีดีสำหรับระบบจำลองรถยนต์แห่งอิลลินอยส์
(ข) แผนภูมิการทำงานของหาค่าฟังก์ชันวัตถุประสงค์

การค้นหาค่าเงินการบนคอมพิวเตอร์ Pentium 4 ความเร็ว 2.4 จิกกะเฮิร์ตซ์ ซึ่งมีหน่วยความจำ RAM ขนาด 640 เมกกะไบต์ หน่วยความจำ HDD ขนาด 40 จิกกะไบต์ ได้ผลการค้นหานี้ ค่าตอบเริ่มต้นของหน่วยการค้นหาที่ 1 จากการสุ่มในปริภูมิการค้นหาย่อยที่ 1 ได้ค่า $K_p = 1.824$ และ $K_D = 0.153$ ซึ่งให้ค่า $J=1000$ ส่วนค่าตอบเริ่มต้นของหน่วยการค้นหาที่ 2 จากการสุ่มในปริภูมิการค้นหาย่อยที่ 2 ได้ค่า $K_p = 0.329$ และ $K_D = 0.776$ ด้วยค่า $J=1000$ เช่นกัน เมื่อดำเนินการค้นหาต่อไปเรื่อย ๆ จนถึงรอบการค้นหาที่ 50 ก็มีการเรียกใช้ DM ผลการประเมินแนวโน้มการพบคำตอบด้วยค่า J ที่น้อยกว่าทำให้ต้องยกเลิกหน่วยค้นหาที่สอง เหลือแต่หน่วยค้นหาที่หนึ่งทำการค้นหาต่อไปจนพบคำตอบที่เหมาะสมภายใต้เกณฑ์ยุติการค้นหาที่ว่า J น้อยกว่า 65 ณ รอบการค้นหาที่ 326 ด้วยเวลาการค้นหา 217.1648 วินาที ค่าตอบที่ได้นั้นให้ค่า $J=61.7234$ ด้วยค่า $K_p = 1.0$ และ $K_D = 0.07$ รายงานการค้นหาตัวควบคุม PD ด้วย MATS แบบ 2 เส้นทางค้นหาสามารถนำเสนอด้วยกราฟการลู่เข้าหาคำตอบ แสดงดังรูปที่ 6.7 และเมื่อนำคำตอบ K_p และ K_D ที่ได้นั้นไปจำลองการทำงานจะได้ผลตอบแทนแบบขั้นบันไดแสดงดังรูปที่ 6.8



รูปที่ 6.7 กราฟการลู่เข้าหาผลเฉลยของ MATS ในการค้นหาตัวควบคุม PD



รูปที่ 6.8 ผลตอบสนองแบบขึ้นบันไดของระบบ $G_{irs,PD}(s)$

ผลตอบสนองแบบขึ้นบันไดของระบบใหม่ที่มีตัวควบคุม PD สามารถให้ผลตอบสนองที่ดีขึ้น พิจารณาได้จากการลดลงของเวลาเข้าที่ (t_s) จากเดิม ใช้ 0.552 วินาที แต่ระบบใหม่ ใช้เพียง 0.401 วินาที ใช้เวลาสั้นกว่าเดิม 27.36% หรือพิจารณาจากเวลาไต่ระดับ (t_r) ให้ผลที่สอดคล้องกับ t_s นั่นคือเดิมใช้เวลาถึง 0.231 วินาที แต่ภายหลังเพิ่มตัวควบคุม PD เวลาไต่ระดับใหม่ใช้เพียง 0.161 วินาที ใช้เวลาสั้นกว่าเดิม 30.30% และพารามิเตอร์ตัวสุดท้ายของผลตอบสนองแบบขึ้นบันไดจากรูปที่ 6.8 ที่พิจารณาได้แก่ เวลาประวิง (t_d) จากเดิม มีประมาณ 5 มิลลิวินาที เมื่อปรับปรุงระบบควบคุมแล้วเสร็จ เวลาประวิงลดลงเหลือ เพียง 0.4 มิลลิวินาที คิดเป็น 92% ของเวลาประวิงที่ลดลงไป จากผลตอบสนองข้างต้นนี้สามารถยืนยันได้เป็นอย่างดีว่า ตัวควบคุม PD ที่เพิ่มเข้ามาร่วมกับการค้นหาอย่างมีประสิทธิภาพของ MATS สามารถกำจัดปัญหาเวลาประวิงของระบบได้เกือบหมด และยังทำให้ผลตอบสนองทางเวลาของระบบเข้าสู่สภาวะอยู่ตัวที่ต้องการได้ในเวลาที่รวดเร็วขึ้นอีกด้วย

6.4 การปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิมด้วย MATS

เพื่อศึกษาแนวทางการปรับปรุงสมรรถนะของระบบ IRS โดยการคัดสรรค่าพารามิเตอร์ของตัวควบคุมใหม่ทั้งหมด 23 ค่า ด้วย MATS (Puangdownreong et al., 2010) พิจารณารูปที่ 6.2 พบว่าระบบ IRS ประกอบด้วยตัวควบคุมสามชุด คือ G_{dc} G_{ff} และ G_{fb} แสดงรายละเอียดดังสมการที่ (6.17) - (6.19) ตามลำดับ ซึ่งในหัวข้อนี้ต้องการศึกษาแนวทางการปรับแต่งค่าพารามิเตอร์ทั้ง 23 ตัวนี้ใหม่ จึงทำการกำหนดชื่อพารามิเตอร์เหล่านี้เป็น a_{1-6} b_{1-6} p_{1-3} q_{1-4} และ r_{1-4} แสดงดังสมการที่ (6.30) - (6.32)

$$G_{dc} = \frac{b_1 s^5 + b_2 s^4 + b_3 s^3 + b_4 s^2 + b_5 s + b_6}{a_1 s^5 + a_2 s^4 + a_3 s^3 + a_4 s^2 + a_5 s + a_6} \quad (6.30)$$

$$G_{ff} = \frac{p_1 s^2 + p_2 s + p_3}{r_1 s^3 + r_2 s^2 + r_3 s + r_4} \quad (6.31)$$

$$G_{fb} = \frac{q_1 s^3 + q_2 s^2 + q_3 s + q_4}{r_1 s^3 + r_2 s^2 + r_3 s + r_4} \quad (6.32)$$

ปัญหานี้จึงมีพารามิเตอร์ที่ต้องการทราบค่าถึง 23 ตัว แตกต่างจากปัญหาก่อนหน้าที่มีเพียง 2 ตัว แต่ข้อมูลและวิธีการออกแบบส่วนใหญ่ยังสามารถดำเนินการตามรูปแบบที่บรรยายไว้ในหัวข้อ 6.3 ได้ เพียงแต่ตัดส่วนของตัวควบคุม PD ออก และ ทำการปรับปรุงฟังก์ชันวัตถุประสงค์ตามขั้นตอน ดังรูปที่ 6.9

ขั้นตอนการคำนวณของฟังก์ชันวัตถุประสงค์

ขั้นตอนที่ 1 (การเริ่มต้น)

รับค่าพารามิเตอร์ของตัวควบคุมทั้ง 23 ตัว

ขั้นตอนที่ 2 (จำลองการทำงานระบบ IRS ในโปรแกรม MATLAB)

จำลองการทำงานระบบด้วยพลาเน็ตเดิมแต่พารามิเตอร์ตัวควบคุมใหม่โดยทดสอบกับสัญญาณขั้นบันได

ขั้นตอนที่ 3 (ตั้งค่าพารามิเตอร์ที่สนใจ)

นำผลการจำลองที่เป็นอัตราเลี้ยวมาหาค่าเวลาไต่ขึ้น ($t_{\uparrow}$) เวลาเข้าที่ ($t_{\downarrow}$) และเปอร์เซ็นต์ฟุ้งเกิน (P.O.)

ขั้นตอนที่ 4 (คำนวณค่าฟังก์ชันวัตถุประสงค์)

คำนวณค่าฟังก์ชันวัตถุประสงค์ $J = t_{\uparrow} + t_{\downarrow} + \text{P.O.}$ แล้วรายงานค่า J

รูปที่ 6.9 ขั้นตอนการประเมินค่าของฟังก์ชันวัตถุประสงค์ในการค้นหาตัวควบคุมใหม่

นอกจากนี้ยังเปลี่ยนแปลงค่าการปรับโทษ J จาก 1000 เป็น 102 ตามความสัมพันธ์ที่ (6.29) ด้วยเหตุผลที่ว่าหัวข้อนี้เป็นการปรับแต่งพารามิเตอร์ใหม่ทั้ง 23 ตัว เพื่อให้ได้ผลตอบสนองที่ดีขึ้นกว่าเดิม ดังนั้นคำตอบที่ได้ระหว่างการค้นหาด้วย MATS หากให้คุณภาพเท่าเดิมหรือด้อยกว่าเดิมให้สะท้อนออกมาที่ตัวชี้วัด J ว่าเท่าเดิมนั้นคือ $J = 100 + 1 + 1 = 102$ นั่นเอง แสดงดังตรรกะข้างล่างนี้ “ถ้า $J < 102$ ให้ยอมรับค่า J นั่นคือ $J = J$ มิเช่นนั้นให้ปฏิเสธด้วยการปรับโทษเป็น $J = 102$ ”

อาศัยความสัมพันธ์ที่ (6.29) แก้ปัญหาการหาเหมาะที่สุดของค่า J จากรูปที่ 6.9 ซึ่งเป็นปัญหาแบบ 23 มิติ คือมีอินพุต 23 ค่า ประกอบด้วยพารามิเตอร์ a จำนวน 6 ค่า พารามิเตอร์ b จำนวน 6 ค่า พารามิเตอร์ p จำนวน 3 ค่า พารามิเตอร์ q จำนวน 4 ค่า และ พารามิเตอร์ r จำนวน 4 ค่า จากความสัมพันธ์ที่กำหนดดังสมการที่ (6.30) - (6.32) จึงเลือกที่จะใช้ MATS แบบ 23 เส้นทางค้นหา ดำเนินการหาคำตอบที่เหมาะสม การกำหนดกฎการค้นหาคำเนินการด้วยการศึกษาและทดสอบเบื้องต้นจากค่าดั้งเดิมทำหน้าที่เป็นแกนของการแบ่งปริภูมิการค้นหาย่อยต่าง ๆ จึงกำหนดให้ค่าดั้งเดิมเป็นคำตอบแกน (core solution : S_c) ทำให้ได้ค่าปริภูมิการค้นหาแสดงดังตารางที่ 6.3

ตารางที่ 6.3 ค่าตอบแทนที่ได้จากพารามิเตอร์ดั้งเดิมและปฏิภูมิการค้นหาทั้ง 23 มิติ

ค่าตอบแทน				
p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
p1:[103] p2:[1.85x10 ⁴] p3:[8.34x10 ⁵]	q1:[-5.18] q2:[49.98] q3:[5.29x10 ³] q4:[4.81x10 ⁴]	r1:[1.0] r2:[155.3] r3:[3.73x10 ³] r4:[2.07x10 ⁴]	a1:[3.64x10 ⁴] a2:[1.79x10 ⁶] a3:[6.67x10 ⁷] a4:[1.20x10 ⁹] a5:[9.46x10 ⁹] a6:[2.67x10 ¹⁰]	b1:[-1.20x10 ⁵] b2:[-8.58x10 ⁶] b3:[-6.50x10 ⁸] b4:[-1.44x10 ⁰] b5:[-1.22x10 ⁻¹] b6:[-3.60x10 ⁻¹]
ปฏิภูมิการค้นหา				
p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
p1:[203 -103] p2:[2.85x10 ⁴ -1.85x10 ⁴] p3:[8.54x10 ⁵ 8.34x10 ⁵]	q1:[-5.18 -6.18] q2:[49.98 40.98] q3:[5.30x10 ³ -5.30x10 ³] q4:[4.82x10 ⁴ 4.81x10 ⁴]	r1:[3.0 0.1] r2:[155.3 -150.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.06x10 ³]	a1:[3.64x10 ⁴ -1.64x10 ⁴] a2:[1.79x10 ⁶ 1.78x10 ⁶] a3:[6.67x10 ⁷ 6.66x10 ⁷] a4:[1.30x10 ⁹ 1.00x10 ⁹] a5:[9.56x10 ⁹ 9.36x10 ⁹] a6:[2.68x10 ¹⁰ 2.67x10 ¹⁰]	b1:[-1.10x10 ⁵ -1.40x10 ⁵] b2:[-6.58x10 ⁶ -16.58x10 ⁶] b3:[-6.50x10 ⁸ -6.70x10 ⁸] b4:[-1.44x10 ⁰ -1.54x10 ⁰] b5:[-1.22x10 ⁻¹ -1.42x10 ⁻¹] b6:[-3.60x10 ⁻¹ -3.70x10 ⁻¹]

ค่าพารามิเตอร์ของ ATS ย่อย ทั้ง 23 เส้นทางค้นหาจะตั้งไว้เหมือนกันทุกประการ โดยกำหนดตามคำแนะนำการใช้งาน ATS (Sujitjorn et al., 2006) ดังนี้ รัศมีการค้นหาเริ่มต้น $R = 0.2$ คิดเป็น 10% ของปฏิภูมิการค้นหา ค่าตอบข้างเคียงจำนวน 5 ค่า การเข้าคำตอบสูงสุดเท่ากับจำนวน 5 ครั้ง ค่าตอบย้อนหลังเลือกที่ลำดับ 5 การปรับรัศมีการใช้สองช่วง ได้แก่ ช่วงที่หนึ่งเมื่อค่า J น้อยกว่า 90 ให้ใช้รัศมีการค้นหา $R = 0.1$ และ ช่วงที่สองเมื่อค่า J น้อยกว่า 70 ให้ใช้รัศมีการค้นหา $R = 0.05$ เกณฑ์ยุติการค้นหาด้วยจำนวนรอบสูงสุด 10,000 รอบ และ ค่า J น้อยกว่า 65

การดำเนินงานค้นหาอาศัย MATS แบบ 23 เส้นทางค้นหา ซึ่งแบ่งตามจำนวนมิติของปัญหา โดยแผนภูมิการทำงานของ MATS แบบ 23 เส้นทางค้นหาและฟังก์ชันวัตถุประสงค์ แสดงดังรูปที่ 6.10 ค่าตอบแทนหรือแนวการแบ่งปฏิภูมิการค้นหาย่อยพร้อมปฏิภูมิการค้นหาแสดงดังตารางที่ 6.3 ส่วนตัวอย่างปฏิภูมิการค้นหาย่อยที่แบ่งได้ แสดงดังตารางที่ 6.4

ตารางที่ 6.4 ปริภูมิการค้นหาย่อย

P_{1-3}	Q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
ATS#1 p1:[203 -103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#1 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#1 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#1 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#1 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#2 p1:[103 103] p2:[2.85x10 ⁴ -1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#2 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#2 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#2 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#2 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#3 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.54x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#3 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#3 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#3 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#3 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#4 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#4 q1:[-5.18 -6.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#4 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#4 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#4 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#5 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#5 q1:[-5.18 -5.18] q2:[49.98 40.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#5 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#5 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#5 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#6 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#6 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.30x10 ³ -5.30x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#6 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#6 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#6 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#7 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#7 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.82x10 ⁴ 4.81x10 ⁴]	ATS#7 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#7 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#7 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#8 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁻⁵ 8.34x10 ⁻⁵]	ATS#8 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#8 r1:[3.0 0.1] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#8 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#8 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]

ตารางที่ 6.4 ปริภูมิการค้นหาย่อย (ต่อ)

P_{1-3}	Q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
ATS#9 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#9 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#9 r1:[1.0 1.0] r2:[155.3 -150.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#9 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#9 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#10 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#10 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#10 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10³ 3.73x10²] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#10 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#10 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#11 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#11 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#11 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10⁴ 2.06x10³]	ATS#11 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#11 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#12 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#12 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#12 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#12 a1:[3.64x10⁴ -1.64x10⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#12 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#13 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#13 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#13 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#13 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10⁶ 1.78x10⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#13 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#14 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#14 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#14 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#14 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10⁷ 6.66x10⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#14 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#15 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#15 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#15 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#15 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.30x10⁹ 1.00x10⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#15 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#16 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#16 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#16 r1:[3.0 0.1] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#16 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.56x10⁹ 9.36x10⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#16 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]

ตารางที่ 6.4 ปริภูมิการค้นหาย่อย (ต่อ)

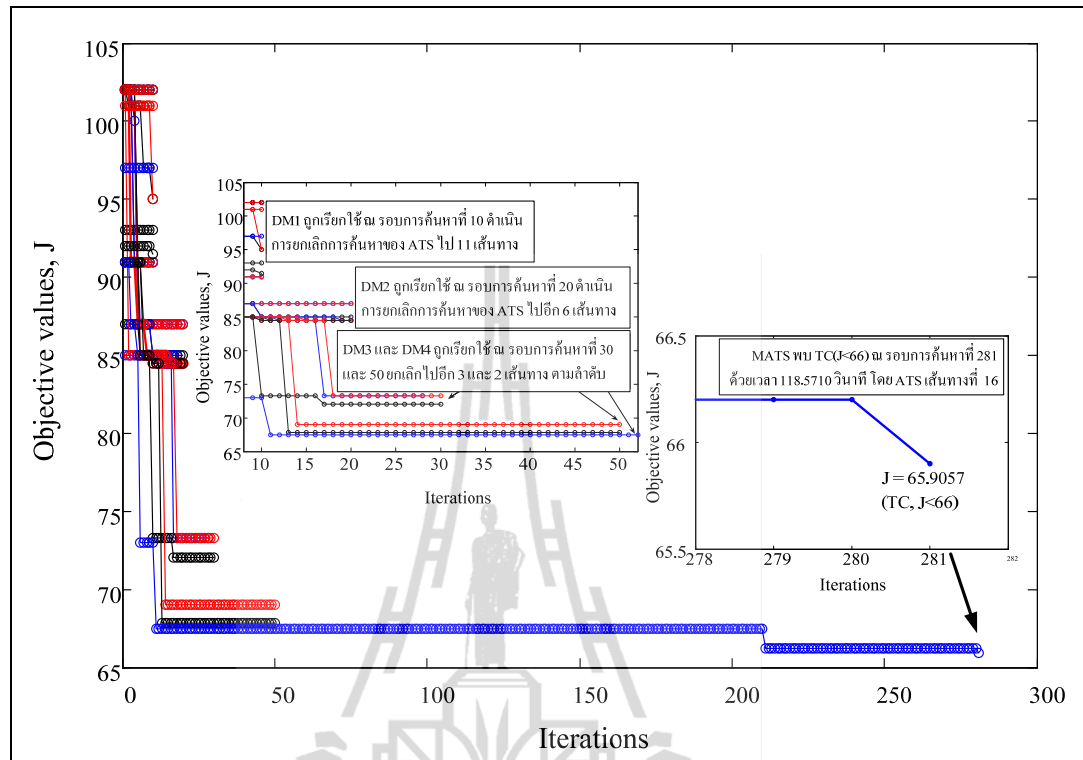
p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
ATS#17 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#17 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#17 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#17 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.68x10¹⁰ 2.67x10¹⁰]	ATS#17 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#18 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#18 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#18 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#18 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#18 b1:[-1.10x10⁵ -1.40x10⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#19 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#19 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#19 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#19 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10¹⁰ 2.67x10¹⁰]	ATS#19 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-6.58x10⁶ -16.58x10⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#20 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#20 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#20 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#20 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#20 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10⁸ -6.70x10⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#21 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#21 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#21 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#21 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10¹⁰ 2.67x10¹⁰]	ATS#21 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10¹⁰ -1.54x10¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#22 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#22 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#22 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#22 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#22 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10¹¹ -1.42x10¹¹] b6:[-3.60x10 ¹¹ -3.60x10 ¹¹]
ATS#23 p1:[103 103] p2:[1.85x10 ⁴ 1.85x10 ⁴] p3:[8.34x10 ⁵ 8.34x10 ⁵]	ATS#23 q1:[-5.18 -5.18] q2:[49.98 49.98] q3:[5.29x10 ³ 5.29x10 ³] q4:[4.81x10 ⁴ 4.81x10 ⁴]	ATS#23 r1:[1.0 1.0] r2:[155.3 155.3] r3:[3.73x10 ³ 3.73x10 ³] r4:[2.07x10 ⁴ 2.07x10 ⁴]	ATS#23 a1:[3.64x10 ⁴ 3.64x10 ⁴] a2:[1.79x10 ⁶ 1.79x10 ⁶] a3:[6.67x10 ⁷ 6.67x10 ⁷] a4:[1.20x10 ⁹ 1.20x10 ⁹] a5:[9.46x10 ⁹ 9.46x10 ⁹] a6:[2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#23 b1:[-1.20x10 ⁵ -1.20x10 ⁵] b2:[-8.58x10 ⁶ -8.58x10 ⁶] b3:[-6.50x10 ⁸ -6.50x10 ⁸] b4:[-1.44x10 ¹⁰ -1.44x10 ¹⁰] b5:[-1.22x10 ¹¹ -1.22x10 ¹¹] b6:[-3.60x10¹¹ -3.70x10¹¹]

ส่วน DM ตั้งค่าการเรียกใช้ไว้สี่ครั้ง ในรอบการค้นหาที่ 10 20 30 และ 50 โดยทำการลดจำนวนหน่วยค้นหาจาก 23 เหลือ 12 6 3 และ 1 หน่วยค้นหาตามลำดับ

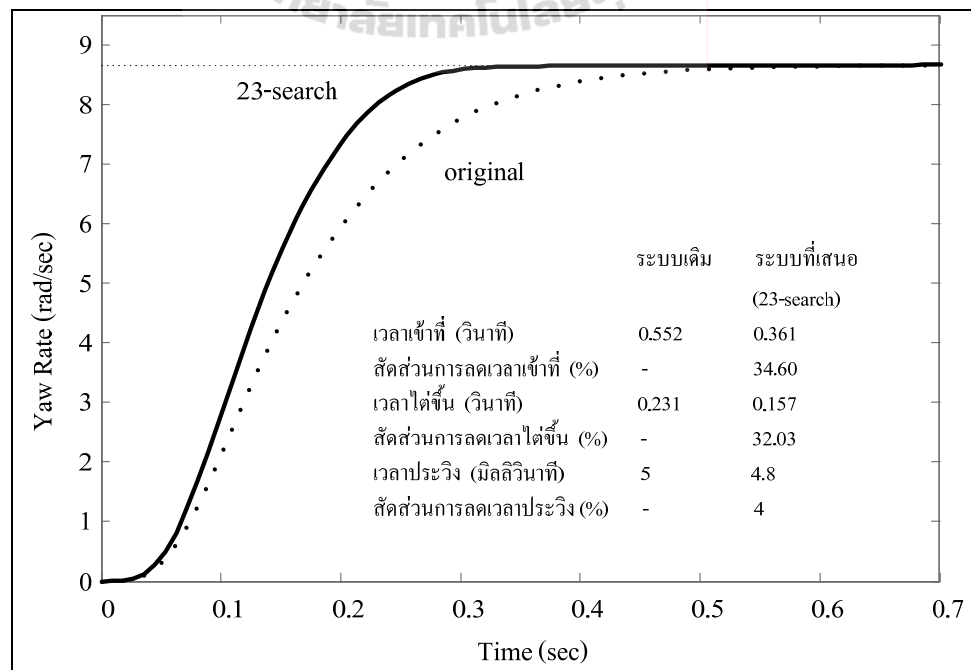
การค้นหาดำเนินการบนคอมพิวเตอร์ Pentium 4 ความเร็ว 2.4 จิกกะเฮิรตซ์ ซึ่งมีหน่วยความจำ RAM ขนาด 640 เมกกะไบต์ หน่วยความจำ HDD ขนาด 40 จิกกะไบต์ MATS เริ่มต้นการค้นหาด้วย PM โดยทำการสุ่มคำตอบที่เหมาะสมจากแต่ละปฏิภูมิการค้นหาย่อย แล้วกำหนดให้เป็นคำตอบเริ่มต้นสำหรับแต่ละ ATS ย่อย เมื่อทุกหน่วยค้นหาได้รับคำตอบเริ่มต้นจาก PM เรียบร้อยก็พร้อมจะเริ่มต้นหาการค้นหาคำตอบวงกว้างอย่างอิสระในปฏิภูมิการค้นหาเดียวกัน ภายหลัง MATS เริ่มดำเนินการค้นหาจนถึงรอบการค้นหาที่ 10 มีการเรียกใช้ DM ผลการประเมินแนวโน้มการพบคำตอบด้วยค่า J ที่น้อยกว่าจำนวน 11 หน่วยค้นหาจะต้องถูกยกเลิกการค้นหา เหลือเพียง 12 หน่วยค้นหาให้ทำการค้นหาต่อไป เมื่อดำเนินการค้นหาจนถึงรอบการค้นหาที่ 20 มีการเรียกใช้ DM ครั้งที่สอง DM ลดจำนวนหน่วยค้นหาลงอีก 6 หน่วยค้นหา เหลือเพียง 6 หน่วยค้นหาในวงรอบการค้นหาของ MATS ส่วนอีกสอง DM ที่เหลือได้แก่ DM3 และ DM4 ถูกเรียกภายหลังในรอบการค้นหาที่ 30 และ 40 ตามลำดับ ลดจำนวนหน่วยค้นหาลง 3 และ 2 หน่วยค้นหา จนเหลือเพียงหนึ่งหน่วยค้นหาซึ่งในที่นี้ได้แก่ ATS#16 ทำการค้นหาต่อไปจนพบคำตอบที่เหมาะสมภายใต้เกณฑ์ยุติการค้นหาที่ว่า J น้อยกว่า 66 ณ รอบการค้นหาที่ 281 ด้วยเวลาการค้นหา 118.5710 วินาที คำตอบที่ได้นั้นให้ค่า $J = 65.9057$ ด้วยค่าใหม่ 23 พารามิเตอร์ แสดงดังตารางที่ 6.4 กราฟการลู่เข้าหาคำตอบของการค้นหาพารามิเตอร์ใหม่ทั้ง 23 ตัวสำหรับตัวควบคุม MATS แบบ 23 เส้นทางค้นหา สามารถนำเสนอได้ดังรูปที่ 6.11 และเมื่อนำคำตอบที่ได้ทั้ง 23 ค่าไปจำลองการทำงานจะได้ผลตอบสนองแบบขั้นบันได แสดงดังรูปที่ 6.12

ตารางที่ 6.5 คำตอบที่ได้หลังการค้นหาด้วย MATS #23

P_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
p1:[102.194] p2:[2.7113x10 ⁴] p3:[8.3450x10 ⁵]	q1:[-5.325] q2:[48.768] q3:[4.7555x10 ³] q4:[4.8142x10 ⁴]	r1:[1.15] r2:[155.129] r3:[2.8627x10 ³] r4:[2.0696x10 ⁴]	a1:[3.6382x10 ⁴] a2:[1.7917x10 ⁶] a3:[6.6708x10 ⁷] a4:[1.2196x10 ⁹] a5:[9.4599x10 ⁹] a6:[2.6788x10 ¹⁰]	b1:[-1.1972x10 ⁻⁵] b2:[-8.5835x10 ⁶] b3:[-6.4974x10 ⁸] b4:[-1.4397x10 ¹⁰] b5:[-1.2241x10 ¹¹] b6:[-3.5978x10 ¹¹]



รูปที่ 6.11 กราฟการลู่เข้าหาผลเฉลยของ MATS ในการค้นหาพารามิเตอร์ 23 ตัว



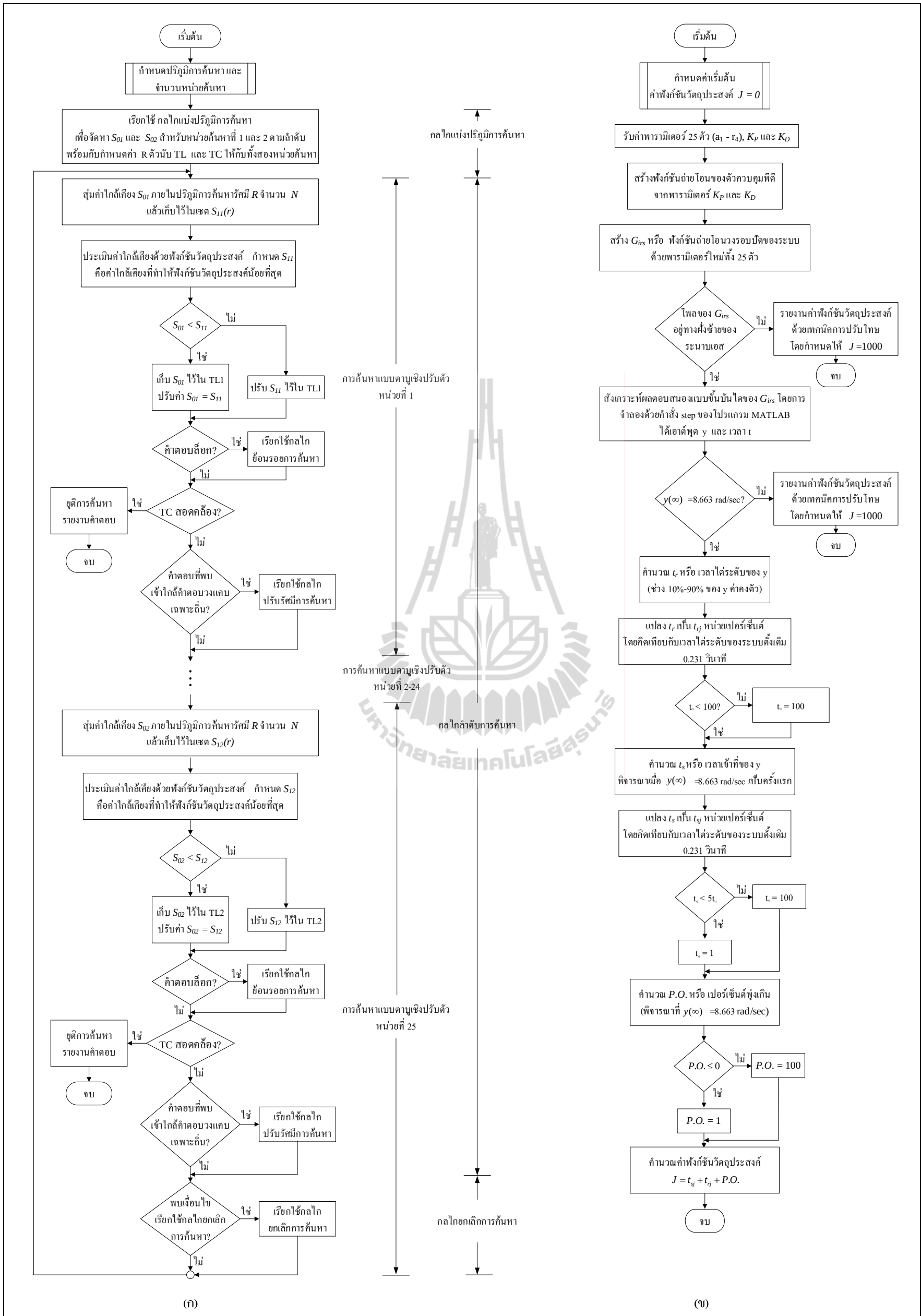
รูปที่ 6.12 ผลตอบสนองแบบขึ้นบันไดของระบบ $G_{irs,23-s}(s)$

ผลตอบสนองแบบขั้นบันไดของระบบกับพารามิเตอร์ตัวควบคุมใหม่ ดังรูปที่ 6.11 สามารถให้ผลตอบสนองที่ดีขึ้น พิจารณาได้จากการลดลงของเวลาเข้าที่ (t_s) จากเดิม ใช้ 0.552 วินาที แต่ระบบใหม่ ใช้เพียง 0.361 วินาที คิดเป็นปริมาณที่ลดลงถึง 34.60% หรือพิจารณาจากเวลาไถ่ระดับ (t_r) ก็ให้ผลที่สอดคล้องกับ t_s นั่นคือเดิมใช้เวลาถึง 0.231 วินาที แต่ภายหลังเปลี่ยนพารามิเตอร์ของตัวควบคุมใหม่ ทำให้ได้ t_s ใหม่เป็น 0.157 วินาที ซึ่งลดลง 32.03% ส่วนเวลาประวิง (t_d) ลดลงเพียงเล็กน้อยคือ 4.8 มิลลิวินาทีจากเดิม ประมาณ 5 มิลลิวินาที คิดเป็น 4% ของ t_d ที่ลดลงไป จากผลตอบสนองข้างต้นนี้ สามารถยืนยันได้เป็นอย่างดีว่าพารามิเตอร์ใหม่ทั้ง 23 ตัวของตัวควบคุมที่ผ่านการคัดสรรอย่างดีจาก MATS สามารถลดเวลาเข้าที่และเวลาไถ่ระดับทำให้ระบบเข้าสู่สภาวะอยู่ตัวตามที่ต้องการได้อย่างน่าพอใจแม้ว่าจะลดปัญหาเวลาประวิงของระบบได้เล็กน้อยก็ตาม

มีข้อสังเกตว่า ผลตอบสนองแบบขั้นบันไดที่ได้จากแนวทางการเพิ่มตัวควบคุม PD แสดงดังรูปที่ 6.8 ให้ผลดีอย่างโดดเด่นสำหรับการแก้ปัญหาเวลาประวิง ส่วนแนวทางการคัดสรรพารามิเตอร์ตัวควบคุมใหม่ 23 ตัว แสดงดังรูปที่ 6.12 ให้ผลดีในเรื่องการลดเวลาเข้าที่และเวลาไถ่ระดับอย่างชัดเจน จึงผสานส่วนดีของทั้งสองแนวทางออกมาเป็นแนวทางที่สาม นั่นคือแนวทางการปรับปรุงสมรรถนะระบบ IRS ด้วยการแทรกตัวควบคุม PD พร้อมกับการคัดสรรพารามิเตอร์ใหม่ทั้ง 23 ค่า ดังรายละเอียดที่จะกล่าวถึงในหัวข้อต่อไป

6.5 การปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD

เพื่อศึกษาแนวทางการปรับปรุงสมรรถนะของระบบ IRS ด้วยองค์ความรู้ที่ได้จากสองวิธีการที่ได้นำเสนอไปให้ในหัวข้อก่อนหน้านี้ เกิดเป็นวิธีการที่สาม นั่นคือการแทรกตัวควบคุม PD จากของวิธีการที่หนึ่ง (หัวข้อ 6.3) ร่วมกับการคัดสรรค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิมทั้ง 23 ตัว ของวิธีการที่สอง (หัวข้อ 6.4) ฟังก์ชันวัตถุประสงค์สำหรับวิธีการนี้เลือกแบบเดียวกันกับของวิธีการที่หนึ่ง ตามสมการที่ (6.28) และ (6.29) แล้วดำเนินการค้นหาพารามิเตอร์ทั้ง 25 ตัวของระบบควบคุมด้วย MATS ซึ่งในที่นี้เลือกใช้จำนวนหน่วยค้นหาเท่ากับจำนวนมิติของปัญหา นั่นคือ MATS แบบ 25 เส้นทาง หรือ MATS #25 ซึ่งแผนภูมิการทำงานของ MATS #25 และฟังก์ชันวัตถุประสงค์สำหรับวิธีการปรับแต่งพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุมที่คาสเคดด้วย แสดงดังในรูปที่ 6.13 การแบ่งปริภูมิย่อยของ PM ได้อาศัยแนวคิดคำตอบที่ได้จากวิธีการที่หนึ่งทำหน้าที่เป็นคำตอบแกนหรือ S_c เพื่อช่วยกำหนดปริภูมิการค้นหาย่อยสำหรับวิธีการนี้อีกด้วย รายละเอียดของ S_c และปริภูมิการค้นหาสำหรับ ปรากฏอยู่ในตารางที่ 6.6



รูปที่ 6.12 (ก) แผนภูมิการทำงานของ MATS แบบ 25 หน่วยค้นหาเพื่อการปรับแต่งค่าพารามิเตอร์ใหม่ให้ทั้งตัวควบคุมดั้งเดิมและตัวควบคุมฟีดกลับสำหรับระบบ IRS (ข) แผนภูมิการทำงานของหาค่าฟังก์ชันวัตถุประสงค์

ตารางที่ 6.6 ปริภูมิการค้นหาทั้ง 25 มิติ ของปัญหาการคัดสรรพารามิเตอร์ตัวควบคุมระบบ IRS

คำตอบแทน				
K_p, K_D และ p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
K_p : [1.00] K_D : [0.07] $p1$: [103] $p2$: [1.8542x10 ⁴] $p3$: [8.3438x10 ⁵]	$q1$: [-5.18] $q2$: [49.98] $q3$: [5.294x10 ³] $q4$: [4.8142x10 ⁴]	$r1$: [1.0] $r2$: [155.30] $r3$: [3.727x10 ³] $r4$: [2.0714x10 ⁴]	$a1$: [3.6426x10 ⁴] $a2$: [1.7918x10 ⁶] $a3$: [6.6708x10 ⁷] $a4$: [1.2024x10 ⁹] $a5$: [9.4644x10 ⁹] $a6$: [2.6788x10 ¹⁰]	$b1$: [-1.2036x10 ⁵] $b2$: [-8.5835x10 ⁶] $b3$: [-6.4974x10 ⁸] $b4$: [-1.4397x10 ¹⁰] $b5$: [-1.2241x10 ¹¹] $b6$: [-3.5978x10 ¹¹]
ปริภูมิการค้นหา				
K_p, K_D และ p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
K_p : [2 0] K_D : [2 0] $p1$: [503 -103] $p2$: [2.85x10 ⁴ -1.85x10 ⁴] $p3$: [8.54x10 ⁵ 8.34x10 ⁵]	$q1$: [5.18 -5.18] $q2$: [49.98 -49.98] $q3$: [5.30x10 ³ -5.30x10 ³] $q4$: [4.82x10 ⁴ 4.81x10 ⁴]	$r1$: [3.0 0.1] $r2$: [155.3 -150.3] $r3$: [3.73x10 ³ 3.73x10 ³] $r4$: [2.08x10 ⁴ 2.06x10 ⁴]	$a1$: [3.84x10 ⁴ -1.64x10 ⁴] $a2$: [1.80x10 ⁶ 1.70x10 ⁶] $a3$: [6.69x10 ⁷ 6.66x10 ⁷] $a4$: [1.30x10 ⁹ 1.00x10 ⁹] $a5$: [9.56x10 ⁹ 9.36x10 ⁹] $a6$: [2.68x10 ¹⁰ 2.66x10 ¹⁰]	$b1$: [-1.10x10 ⁵ -1.40x10 ⁵] $b2$: [-6.58x10 ⁶ -16.58x10 ⁶] $b3$: [-6.20x10 ⁸ -6.80x10 ⁸] $b4$: [-1.20x10 ¹⁰ -1.60x10 ¹⁰] $b5$: [-1.20x10 ¹¹ -1.60x10 ¹¹] $b6$: [-3.20x10 ¹¹ -3.80x10 ¹¹]

ฟังก์ชันวัตถุประสงค์สำหรับวิธีการนี้เลือกแบบเดียวกันกับของวิธีการที่หนึ่ง ตามสมการที่ (6.28) และ (6.29) โดยจำนวนอินพุตจากเดิม K_p และ K_D จำนวนสองค่า เพิ่มพารามิเตอร์ของตัวควบคุมเดิมอีกจำนวน 23 ค่า รวมเป็นพารามิเตอร์ที่ MATS ต้องค้นหาจำนวนทั้งสิ้น 25 ค่า

ตารางที่ 6.7 ปริภูมิการค้นหาย่อยทั้ง 25 พารามิเตอร์ (ต่อ)

ตารางที่ 6.7 ปริภูมิการค้นหาย่อยทั้ง 25 พารามิเตอร์ (ต่อ)

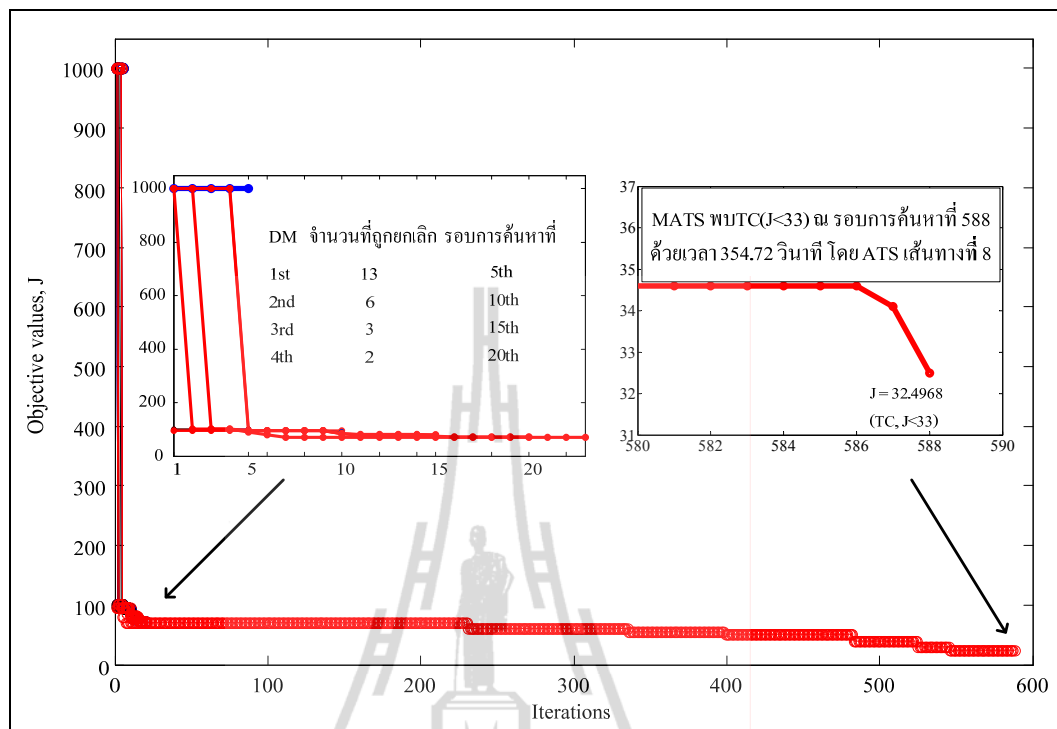
P_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
ATS#25 K_p : [1 1] K_D : [0.07 0.07] p_1 : [103 103] p_2 : [1.85x10 ⁴ 1.85x10 ⁴] p_3 : [8.34x10 ⁵ 8.34x10 ⁵]	ATS#25 q_1 : [-5.18 -5.18] q_2 : [49.98 49.98] q_3 : [5.29x10 ³ 5.29x10 ³] q_4 : [4.81x10 ⁴ 4.81x10 ⁴]	ATS#25 r_1 : [1.0 1.0] r_2 : [155.3 155.3] r_3 : [3.73x10 ³ 3.73x10 ³] r_4 : [2.07x10 ⁴ 2.07x10 ⁴]	ATS#25 a_1 : [3.64x10 ⁴ 3.64x10 ⁴] a_2 : [1.79x10 ⁶ 1.79x10 ⁶] a_3 : [6.67x10 ⁷ 6.67x10 ⁷] a_4 : [1.20x10 ⁹ 1.20x10 ⁹] a_5 : [9.46x10 ⁹ 9.46x10 ⁹] a_6 : [2.67x10 ¹⁰ 2.67x10 ¹⁰]	ATS#25 b_1 : [-1.20x10 ⁵ -1.20x10 ⁵] b_2 : [-8.58x10 ⁶ -8.58x10 ⁶] b_3 : [-6.50x10 ⁸ -6.50x10 ⁸] b_4 : [-1.44x10 ¹⁰ -1.44x10 ¹⁰] b_5 : [-1.22x10 ¹¹ -1.22x10 ¹¹] b_6: [-3.20x10¹¹ -3.80x10¹¹]

ส่วน DM ตั้งค่าการเรียกใช้ไว้สี่ครั้ง ในรอบการค้นหาที่ 5 10 15 และ 20 โดยลดจำนวนหน่วยค้นหาจาก 25 เหลือ 13 6 3 และ 1 หน่วยค้นหาตามลำดับ

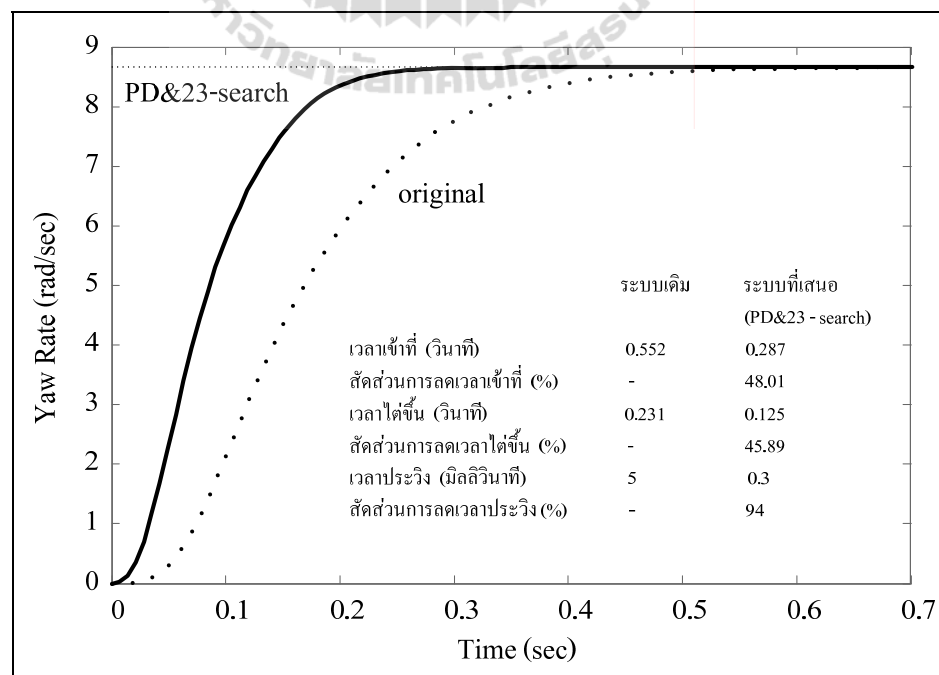
การค้นหาดำเนินการบนคอมพิวเตอร์ Pentium 4 ความเร็ว 2.4 จิกกะเฮิรตซ์ ซึ่งมีหน่วยความจำ RAM ขนาด 640 เมกกะไบต์ หน่วยความจำ HDD ขนาด 40 จิกกะไบต์ MATS เริ่มต้นการค้นหาด้วย PM ทำการสุ่มคำตอบที่เหมาะสมจากแต่ละปริภูมิการค้นหาย่อย แล้วกำหนดให้เป็นคำตอบเริ่มต้นสำหรับแต่ละ ATS ย่อย เมื่อทุกหน่วยค้นหาได้รับคำตอบเริ่มต้นจาก PM เรียบร้อยก็พร้อมจะเริ่มต้นหาการค้นหาคำตอบวงกว้างอย่างอิสระในปริภูมิการค้นหาเดียวกัน ภายหลัง MATS เริ่มดำเนินการค้นหาจนถึงรอบการค้นหาที่ 5 มีการเรียกใช้ DM เมื่อประเมินค่า J แล้วจึงยกเลิก 13 หน่วยค้นหา เหลือเพียง 12 หน่วยค้นหาให้ทำการค้นหาต่อไป เมื่อดำเนินการค้นหาจนถึงรอบการค้นหาที่ 10 DM ทำงานเป็นครั้งที่สอง ลดจำนวนหน่วยค้นหาอีก 6 หน่วยค้นหา เหลือเพียง 6 หน่วยค้นหาการทำงานของ DM อีกสองครั้ง ได้แก่ DM3 และ DM4 ถูกเรียกในรอบการค้นหาที่ 15 และ 20 ตามลำดับ ลดจำนวนหน่วยค้นหา 3 และ 2 หน่วยค้นหา จนเหลือเพียงหนึ่งหน่วยค้นหาซึ่งในที่นี้คือ ATS#8 ที่อยู่ดำเนินการค้นหาต่อไปจนพบคำตอบที่เหมาะสมภายใต้เกณฑ์ยุติการค้นหาที่ว่า J น้อยกว่า 33 ณ รอบการค้นหาที่ 588 ด้วยเวลาการค้นหา 354.72 วินาที คำตอบที่ได้นั้นแสดงอยู่ในตารางที่ 6.8 มีค่า $J=32.4968$ กราฟการถ่วงน้ำหนักคำตอบวงกว้างของ MATS แบบ 25 หน่วยค้นหาแสดงดังรูปที่ 6.14 และเมื่อนำคำตอบที่ได้ไปจำลองการทำงานของระบบควบคุมได้ผลตอบแทนแบบขั้นบันไดแสดงดังรูปที่ 6.15

ตารางที่ 6.8 คำตอบที่ได้หลังการค้นหาด้วย MATS#25

K_p, K_D and p_{1-3}	q_{1-4}	r_{1-4}	a_{1-6}	b_{1-6}
K_p : [1.00] K_D : [0.07] p_1 : [318.7260] p_2 : [1.8542x10 ⁴] p_3 : [8.3438x10 ⁵]	q_1 : [-0.6192] q_2 : [25.2519] q_3 : [4.3801x10 ³] q_4 : [4.8153x10 ⁴]	r_1 : [1.1441] r_2 : [122.0677] r_3 : [3.5002x10 ³] r_4 : [2.0704x10 ⁴]	a_1 : [3.6426x10 ⁴] a_2 : [1.7918x10 ⁶] a_3 : [6.6708x10 ⁷] a_4 : [1.2024x10 ⁹] a_5 : [9.4644x10 ⁹] a_6 : [2.6788x10 ¹⁰]	b_1 : [-1.2036x10 ⁵] b_2 : [-8.5835x10 ⁶] b_3 : [-6.4974x10 ⁸] b_4 : [-1.4397x10 ¹⁰] b_5 : [-1.2241x10 ¹¹] b_6 : [-3.5978x10 ¹¹]



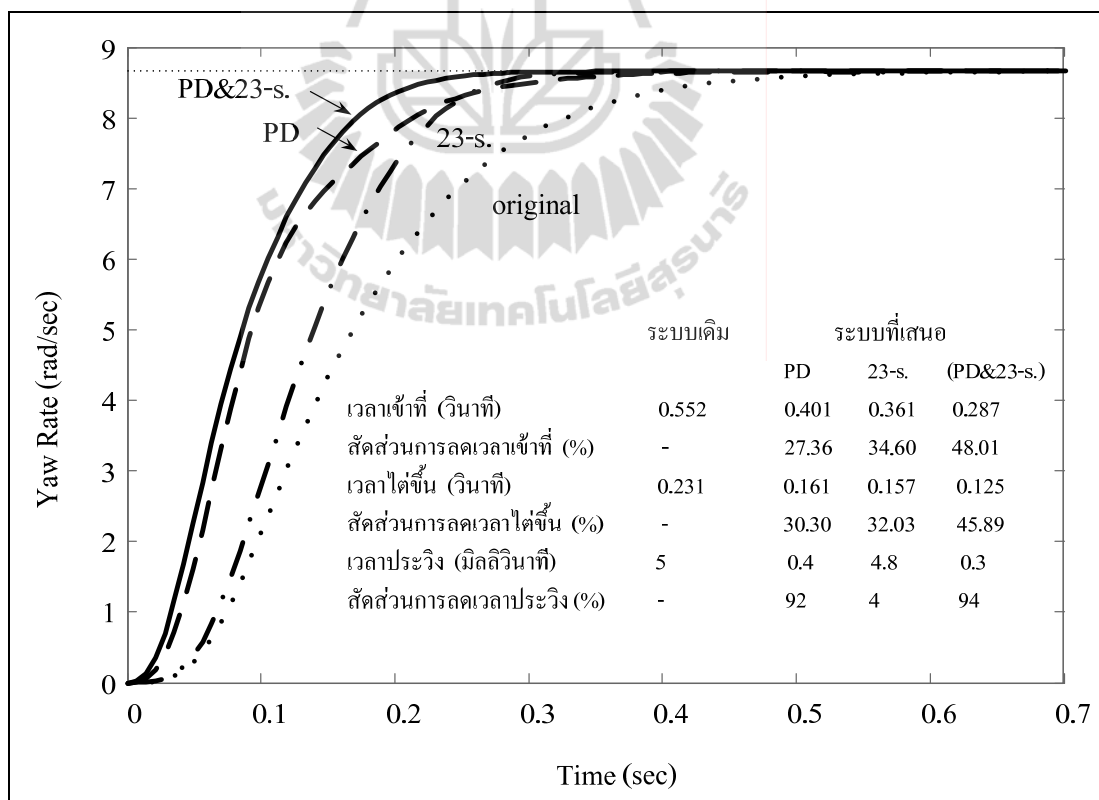
รูปที่ 6.14 กราฟการลู่เข้าหาผลเฉลยของ MATS กับการค้นหาพารามิเตอร์ 25 ตัวของตัวควบคุม



รูปที่ 6.15 ผลตอบสนองแบบขั้นบันไดของระบบ $G_{irs, PD\&23-s}$ (s)

ผลตอบสนองแบบขั้นบันไดของระบบที่มีการแทรกตัวควบคุม PD ร่วมกับการคัดสรรพารามิเตอร์ใหม่ทั้ง 23 ค่า แสดงดังรูปที่ 6.15 สามารถให้ผลตอบสนองที่ดีขึ้นอย่างน่าพอใจ ทั้ง t_s , t_r และ t_d โดย t_s ใช้เวลาเพียง 0.287 วินาที สั้นกว่าเวลาเดิมถึง 48.01% ส่วน t_r ก็ให้ผลที่สอดคล้องกับ t_s นั้น คือเดิมใช้เวลาเพียง 0.125 วินาที สั้นกว่าเวลาเดิม 45.89% และโดยเฉพาะ t_d สามารถลดลงได้เหลือเพียง 0.3 มิลลิวินาทีจากเดิม ประมาณ 5 มิลลิวินาที คิดเป็นเวลาประวิงที่ลดลงถึง 94% ซึ่งถือได้ว่าเกือบไม่มีเวลาประวิงเหลืออยู่เลย ดังนั้นจึงสามารถสรุปได้ว่าการแทรกตัว PD ช่วยลดผลของเวลาประวิง ร่วมกับการคัดสรรพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิมไปพร้อม ๆ กัน ช่วยทำให้เราได้สมรรถนะของระบบรวมดีขึ้นตามที่ต้องการได้

เพื่อให้เห็นภาพรวมพัฒนาการของศึกษาแนวทางปรับปรุงสมรรถนะระบบ IRS ทั้งสามแนวทางดังที่กล่าวไปแล้วข้างต้น จึงได้นำผลตอบสนองระบบดั้งเดิมกับผลตอบสนองของระบบที่เสนอใหม่ทั้งหมดรวมกันไว้ในรูปเดียว แสดงดังรูปที่ 6.16



รูปที่ 6.16 ภาพรวมผลตอบสนองแบบขั้นบันไดที่ได้ศึกษาทั้งหมด

เปรียบเทียบผลสนองแบบขั้นบันไดของทั้งสามวิธี ดังแสดงในรูปที่ 6.16 เมื่อพิจารณาแง่ของเวลาเข้าที่ t_s อ้างอิงกับ t_s เดิม 0.552 วินาที พบว่าวิธีที่สาม (PD + 23-s.) ให้ผลดีที่สุด โดยใช้เวลาเพียง 0.287 วินาที คิดเป็นเวลาสั้นกว่าเดิม 48.01% รองลงไปเป็นวิธีที่สอง (23-s.) ใช้เวลาเพียง 0.361 วินาที ประหยัดเวลาได้ 34.60% และสุดท้ายเป็นของวิธีที่หนึ่ง (PD) ใช้เวลาเพียง 0.401 วินาที ลดเวลาลงได้ 27.36%

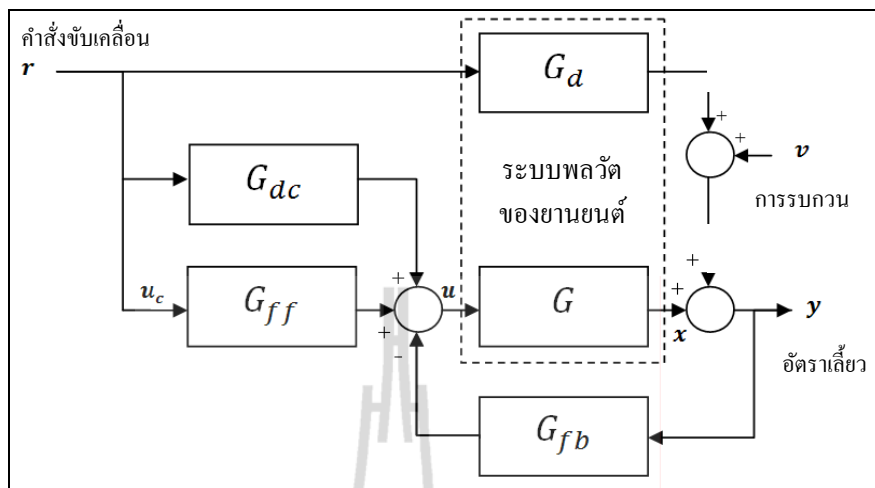
พิจารณาแง่ของเวลาไต่ขึ้น t_r อ้างอิงกับ t_r เดิม 0.231 วินาที พบว่าได้ผลทำนองเดียวกันกับของ t_s นั้นเรียงลำดับจากวิธีที่ดีที่สุดไปจนถึงวิธีที่ให้ผลด้อยสุด ได้แก่ วิธีการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD วิธีการปรับแต่งค่าพารามิเตอร์ให้กับตัวควบคุมดั้งเดิม และวิธีการคาสเคดตัวควบคุม PD เข้ากับบล็อก G_{ff} ตามลำดับ โดยใช้เวลาไป 0.125 0.157 และ 0.161 วินาที ตามลำดับ แต่เมื่อคิดเป็นเปอร์เซ็นต์ของเวลาที่ลดลงจะได้ 45.89% 32.03% และ 30.30% ตามลำดับ

พิจารณาแง่ของเวลาประวิง t_d อ้างอิงกับ t_d เดิมประมาณ 5 มิลลิวินาที พบว่าวิธีการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD ให้ผลดีที่สุด เหลือเพียง 0.3 มิลลิวินาที ลดเวลาประวิงลงได้ 94% รองลงไปเป็นวิธีการคาสเคดตัวควบคุม PD เข้ากับบล็อก G_{ff} เหลือ t_d เพียง 0.4 มิลลิวินาที เวลาประวิงลดลง 92% และสุดท้ายเป็นของวิธีการปรับแต่งค่าพารามิเตอร์ให้กับตัวควบคุมดั้งเดิม สามารถลด t_d ไปให้เหลือเพียง 0.2 มิลลิวินาที คิดเป็น 4% ของเวลาที่ลดลงไปเท่านั้น

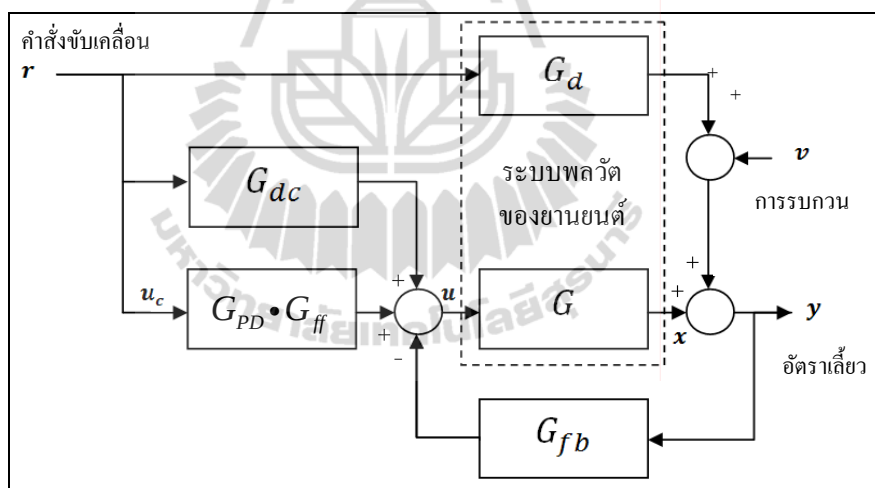
พิจารณาในแง่ของฟังก์ชันวัตถุประสงค์ J อ้างอิงกับ J เดิมเท่ากับ 102 พบว่าวิธีการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุม PD ให้ผลดีที่สุด สามารถให้ลดค่า J ลงได้เหลือ 32.4968 รองลงไปเป็นวิธีการคาสเคดตัวควบคุม PD เข้ากับบล็อก G_{ff} ให้ค่า $J = 61.7134$ และสุดท้ายเป็นของวิธีการปรับแต่งค่าพารามิเตอร์ให้กับตัวควบคุมดั้งเดิม ให้ค่า $J = 65.9057$

6.6 การวิเคราะห์ผลของการรบกวน

หัวข้อนี้เป็นการศึกษาผลกระทบของการรบกวน (disturbance : v) ที่ตำแหน่งสัญญาณเอาต์พุต สำหรับระบบ IRS ดั้งเดิม และที่มีการปรับแต่งค่าพารามิเตอร์ใหม่ แสดงดังรูปที่ 6.17 และ 6.18 ตามลำดับ



รูปที่ 6.17 แผนภาพระบบ IRS ดั้งเดิมที่มีการรบกวน v

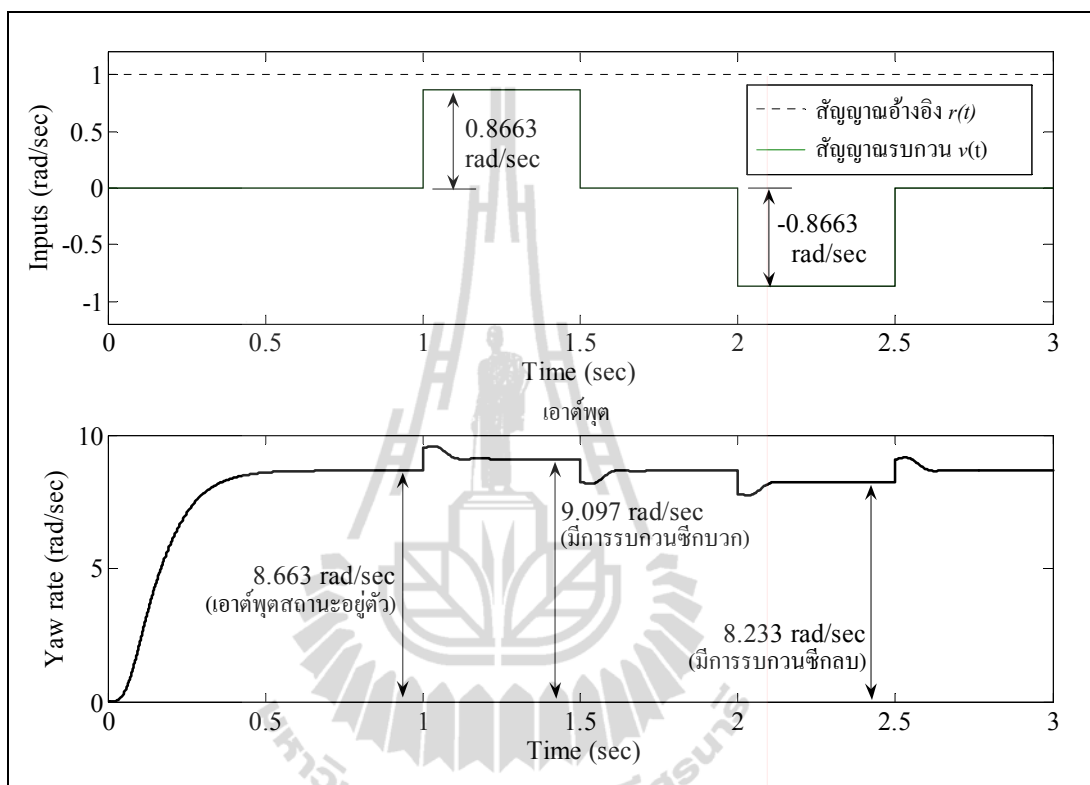


รูปที่ 6.18 แผนภาพระบบ IRS ที่มีการปรับแต่งพารามิเตอร์พร้อมตัวควบคุม PD และปรากฏการณ์รบกวน v

การรบกวนอาศัยสัญญาณพัลส์สี่เหลี่ยมขนาด $\pm 10\%$ ของขนาดเอาต์พุตในสถานะคงตัว (8.663 เรเดียนต่อวินาที) มีค่าเท่ากับ ± 0.8663 เรเดียนต่อวินาที แสดงสัญญาณการรบกวนและสัญญาณอ้างอิงดังรูปบนของ รูปที่ 6.19

หากกำหนดขนาดเปอร์เซ็นต์ความผิดพลาดคงตัว ($\%e_{ss}$) ที่ยอมรับได้ไว้ที่ 5% จากการรบกวนขนาด 10% ของเอาต์พุต แล้วทำการจำลองการทำงานของระบบ G_{irs} ได้ผลตอบแทนต่อ

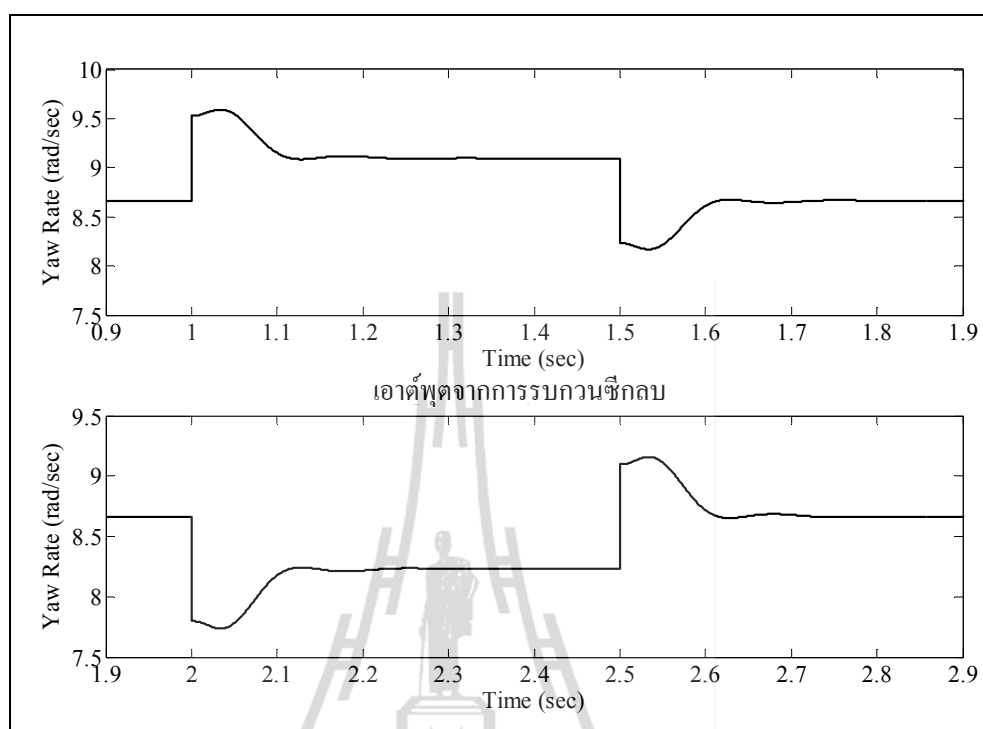
สัญญาณอ้างอิงและการรบกวนของระบบแบบจำลองรถยนต์แห่งอิสลินอยส์ (IRS) ดังเดิม ตามโครงสร้างดังรูปที่ 6.17 แสดงดังรูปล่างของรูปที่ 6.19



รูปที่ 6.19 สัญญาณอ้างอิง r สัญญาณการรบกวน v และผลตอบสนองของระบบ IRS ดังเดิม

จากรูปที่ 6.19 พบว่าการรบกวนมีผลทำให้ค่าคงตัวของเอาต์พุตเปลี่ยนไปจากเดิมเอาต์พุตเดิมมีค่า 8.663 เรเดียนต่อวินาที เมื่อพิจารณาการรบกวนซิกบวกรูปแบบสี่เหลี่ยมขนาด 0.8663 เรเดียนต่อวินาที ในช่วงวินาทีที่ 1 ถึงวินาทีที่ 1.5 หลังจากทีระบบเข้าสู่สภาวะคงตัวใหม่ประมาณวินาทีที่ 1.23 เอาต์พุตใหม่มีค่าเป็น 9.097 เรเดียนต่อวินาที ซึ่งสูงขึ้น 0.434 เรเดียนต่อวินาที คิดเป็น $\%e_{ss} = 5$ ซึ่งอยู่ในเกณฑ์ที่ยอมรับได้พอดี และเมื่อพิจารณาการรบกวนซิกลบด้วยแบบพัลส์สี่เหลี่ยม -0.8663 เรเดียนต่อวินาที ในช่วงวินาทีที่ 2 ถึงวินาทีที่ 2.5 รอยจนกระทั่งระบบเข้าสู่สภาวะคงตัวใหม่ประมาณวินาทีที่ 2.49 เอาต์พุตใหม่มีค่าเป็น 8.233 เรเดียนต่อวินาที ซึ่งลดลง 0.430 เรเดียนต่อวินาที เมื่อคิดเป็น $\%e_{ss} = 4.96$ ซึ่งอยู่ในเกณฑ์ที่ยอมรับได้เช่นเดียวกัน

ภาพขยายผลตอบสนองของเอาต์พุตต่อการรบกวนซิกบวกร แสดงอยู่ในรูปบนของรูปที่ 6.19 และการรบกวนซิกลบแสดงอยู่ในรูปล่าง รูปที่ 6.19 ตามลำดับ



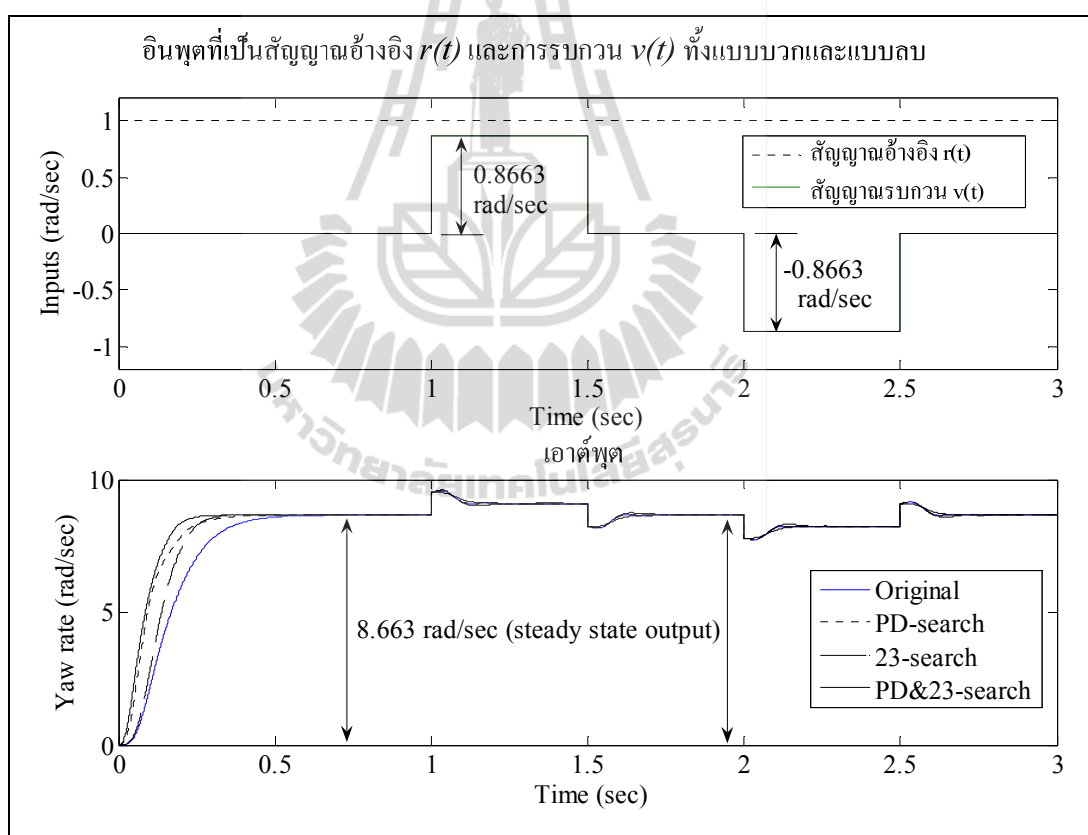
รูปที่ 6.20 ภาพขยายเอาต์พุตจากการรบกวนเชิงบวก (ภาพบน) และการรบกวนเชิงลบ (ภาพล่าง)

จากรูปที่ 6.20 หากประเมินสมรรถนะระบบด้วยการพิจารณาค่าสูงสุดของผลตอบสนอง เอาต์พุตต่อกรณีการรบกวนสำหรับการรบกวนเชิงบวก แสดงดังรูปบนของรูปที่ 6.20 อ่านค่าได้ 9.5916 เรเดียนต่อวินาที ณ วินาทีที่ 1.035 และระบบใช้เวลาเพื่อเข้าสู่สภาวะคงตัวใหม่ 0.354 วินาที ณ วินาทีที่ 1.354 นับจากที่ปรากฏพบการรบกวน ส่วนท้ายการรบกวนเชิงบวกจะพิจารณาจากค่าต่ำสุดของผลตอบสนองเอาต์พุตต่อกรณีการรบกวนซึ่งอ่านค่าได้ 8.1691 เรเดียนต่อวินาที ณ วินาทีที่ 1.534 และระบบใช้เวลาเพื่อเข้าสู่สภาวะคงตัวใหม่ 0.311 วินาที ณ วินาทีที่ 1.811

กรณีของการรบกวนเชิงลบ แสดงดังรูปล่างของรูปที่ 6.20 ค่าต่ำสุดของผลตอบสนอง เอาต์พุตต่อกรณีการรบกวนสำหรับการรบกวนเชิงลบ อ่านค่าได้ 7.7362 เรเดียนต่อวินาที ณ วินาทีที่ 2.035 และระบบใช้เวลาเพื่อเข้าสู่สภาวะคงตัวใหม่ 0.281 วินาที ณ วินาทีที่ 2.281 นับจากที่ปรากฏพบการรบกวน ส่วนท้ายการรบกวนเชิงลบจะพิจารณาจากค่าต่ำสุดของผลตอบสนอง เอาต์พุตต่อกรณีการรบกวนซึ่งอ่านค่าได้ 9.1586 เรเดียนต่อวินาที ณ วินาทีที่ 2.534 และระบบใช้เวลาเพื่อเข้าสู่สภาวะคงตัวใหม่ 0.297 วินาที ณ วินาทีที่ 2.797

เพื่อเปรียบเทียบผลของการรบกวนต่อระบบทั้งสี่ระบบได้แก่ ระบบที่หนึ่งคือระบบจำลองรถยนต์แห่งอิลลินอยส์แบบดั้งเดิม ซึ่งได้กล่าวถึงแล้วตอนต้น ระบบที่สองคือระบบจำลองรถยนต์แห่งอิลลินอยส์แบบดั้งเดิมที่มีการคาดเดาคงตัวควบคุมพิดีเข้ากับบล็อก G_{ff} จากหัวข้อ 6.3 ซึ่งต่อไปนี

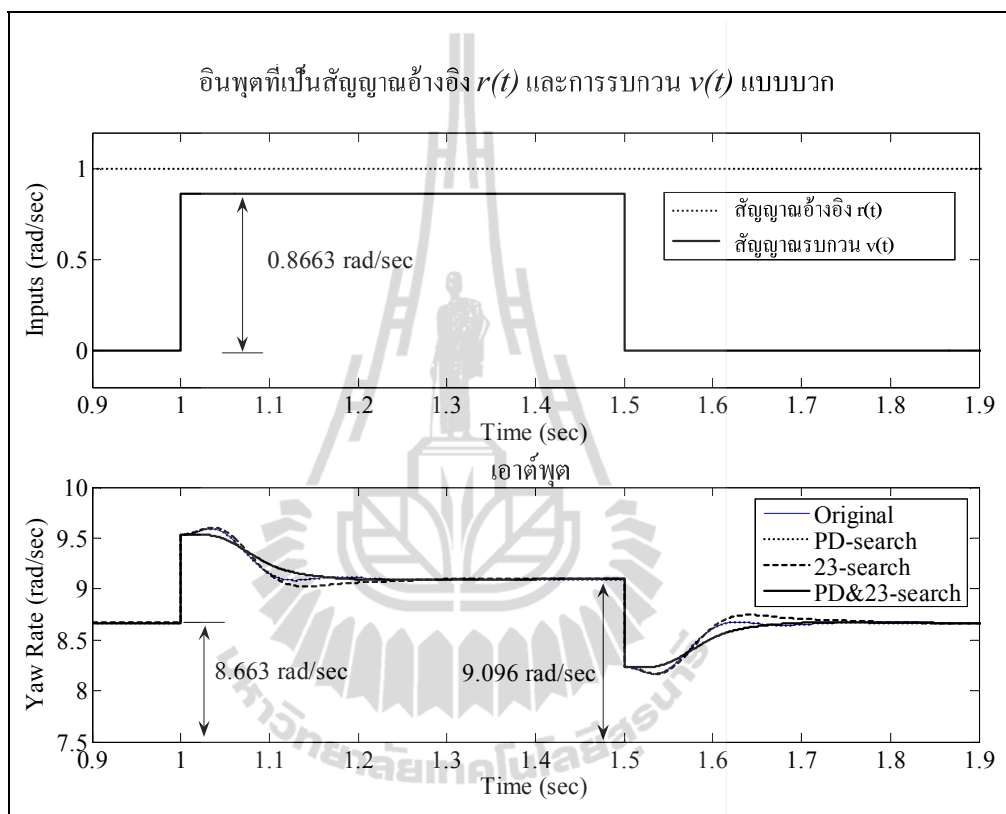
จะเรียกสั้น ๆ ว่าระบบ PD-search เนื่องจากมีการค้นหาพารามิเตอร์ของตัวควบคุมพีดี 2 ตัว ด้วย MATS ระบบที่สามคือระบบจำลองรถยนต์แห่งอิลลินอยส์ที่มีการปรับแต่งค่าพารามิเตอร์ใหม่ด้วย MATS จากหัวข้อ 6.4 ซึ่งต่อไปนี้จะเรียกสั้น ๆ ว่าระบบ 23-search เนื่องจากมีการค้นหาพารามิเตอร์ 23 ตัว และระบบที่สี่คือระบบจำลองรถยนต์แห่งอิลลินอยส์พร้อมตัวควบคุมพีดีที่มีการปรับแต่งค่าพารามิเตอร์ใหม่ด้วย MATS จากหัวข้อ 6.5 ซึ่งต่อไปนี้จะเรียกสั้น ๆ ว่าระบบ PD&23-search เนื่องจากมีการค้นหาพารามิเตอร์ของตัวควบคุมพีดีจำนวน 2 ตัวร่วมกับพารามิเตอร์อื่นอีก 23 ตัว ผลการจำลองสถานการณ์เพื่อศึกษาผลกระทบของการรบกวนทั้งเชิงบวกและเชิงลบ ดังแสดงในรูปที่ 6.21



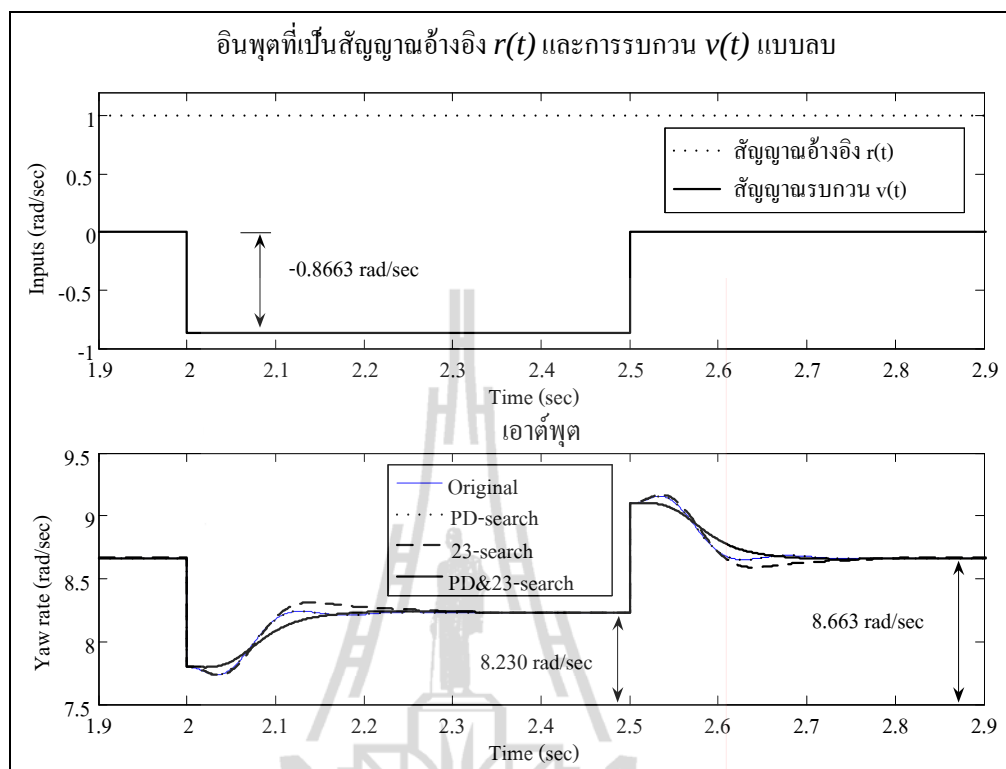
รูปที่ 6.21 เอาต์พุตของทั้งสี่ระบบต่ออินพุตที่เป็นสัญญาณอ้างอิงแบบขั้นบันไดและการรบกวน

เมื่อพิจารณาภาพขยายของการรบกวนเชิงบวก แสดงดังรูปที่ 6.22 พบว่าระบบ PD&23-search กลับคืนสู่สภาวะคงตัวได้รวดเร็วที่สุดด้วยเวลาเพียง 0.2 วินาที โดยไม่มีการพุ่งเกิน ทั้งช่วงเวลาขาเข้าและช่วงเวลาขาออกของการรบกวน รองลงมาเป็นของระบบดั้งเดิมกับระบบ PD-search ซึ่งทั้งสองระบบประกอบด้วยตัวควบคุมป้อนกลับที่กำหนดที่กำจัดการรบกวนเหมือนกัน

จึงทำให้ได้ผลตอบสนองต่อการรบกวนที่เหมือนกัน โดยใช้เวลาเข้าสู่สภาวะคงตัวประมาณ 0.23 วินาที และการพุ่งเกิน 1.15 เปอร์เซ็นต์ ส่วนระบบที่ให้ผลตอบสนองต่อการรบกวนช้าที่สุดเป็นระบบ 23-search ใช้เวลาเข้าสู่สภาวะคงตัวประมาณ 0.27 วินาที และการพุ่งเกิน 1.5 เปอร์เซ็นต์



รูปที่ 6.22 ภาพขยายเอาต์พุตของทั้งสี่ระบบต่อการรบกวนเชิงบวก



รูปที่ 6.23 ภาพขยายเอาต์พุตของทั้งสี่ระบบต่อการรบกวนชிகลบ

ภาพขยายของการรบกวนชิกลบ แสดงดังรูปที่ 6.23 เมื่อปรากฏการรบกวนชิกลบขึ้นส่งผลให้เอาต์พุตมีขนาดลดลงซึ่งตรงกันข้ามกับการรบกวนชิกบวก แต่ขนาดและเวลาของผลสนองต่อการรบกวนของทั้งสี่ระบบยังคงให้ผลทำนองเดียวกันกับการรบกวนชิกบวก ซึ่งรายละเอียดได้กล่าวถึงแล้วตอนต้นแล้ว

% e_{ss} ทั้งของการรบกวนชิกบวกจากรูปที่ 6.22 และของชิกลบจากรูปที่ 6.23 คิดเป็น 4.99% เท่ากันซึ่งอยู่ในเกณฑ์ 5% ตามที่กำหนดไว้

6.7 สรุป

บทที่ 6 นี้ได้กล่าวถึงแนวทางการปรับปรุงสมรรถนะผลตอบสนองของระบบจำลองยานพาหนะบนผิวถนนแห้งอลลิโนยส์หรือ ระบบ IRS แบ่งไว้สามแนวทาง ได้แก่ แนวทางที่หนึ่ง การคาดเดาตัวควบคุมฟีดี้เข้ากับตัวควบคุมดั้งเดิม (2 พารามิเตอร์) แนวทางที่สองการปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิม (23 พารามิเตอร์) และแนวทางที่สามการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุมฟีดี้ (25 พารามิเตอร์) ใช้ MATS ดำเนินการค้นหาค่าพารามิเตอร์และพัฒนาด้วย MATLAB ผลการค้นหาค่าพารามิเตอร์ด้วย MATS สามารถปรับปรุง

ผลตอบสนองของระบบ IRS ให้ดีขึ้นได้ดังนี้ แนวทางที่หนึ่งการทดสอบตัวควบคุมพีดีเข้าตัวควบคุมดั้งเดิมใช้เวลาเข้าสู่สภาวะคงตัวดีขึ้นโดยลดลงจากเดิม 27.36% แนวทางที่สองการปรับแต่งค่าพารามิเตอร์ใหม่ให้กับตัวควบคุมดั้งเดิมสามารถลดเวลาเข้าสู่สภาวะคงตัวได้ 34.60% และแนวทางที่สามการปรับแต่งค่าพารามิเตอร์ของทั้งตัวควบคุมดั้งเดิมและตัวควบคุมพีดีให้ผลตอบสนองดีที่สุดที่สามารถลดเวลาเข้าสู่สภาวะคงตัวได้ถึง 48.01% และลดเวลาประวิงได้ถึง 94% เมื่อพิจารณาผลของการรบกวนแนวทางที่สามหรือระบบ PD&23-search ยังคงให้ผลตอบสนองที่ดีกว่าแนวทางอื่นในแง่เวลาที่ใช้เข้าสู่สภาวะคงตัวรวดเร็วที่สุดคือ 0.20 วินาที เทียบกับระบบดั้งเดิมกับระบบ PD-search ใช้เวลา 0.23 วินาที และช้าที่สุดเป็นของระบบ 23-search ใช้เวลา 0.27 วินาที นอกจากนี้ยังได้ศึกษาผลของการรบกวนด้วยรูปคลื่นสี่เหลี่ยมขนาด 10% ได้ผลตอบสนองมีค่าเปอร์เซ็นต์ความผิดพลาดสภาวะคงตัวไม่เกิน 5% ตามที่กำหนดไว้ ดังนั้นจึงสามารถกล่าวได้ว่า MATS สามารถประยุกต์ใช้งานเพื่อการออกแบบตัวควบคุมสำหรับระบบจำลองยานพาหนะบนฟิวดนนแห่งอิลลินอยส์ได้ผลเป็นที่น่าพอใจ

หัวข้อถัดไปเป็นการประยุกต์ใช้งานจริงของ MATS สำหรับการระบุเอกลักษณ์ระบบพลวัตของระบบลูกค้อนนำพิกานวกรถ หรือ CPP



บทที่ 7

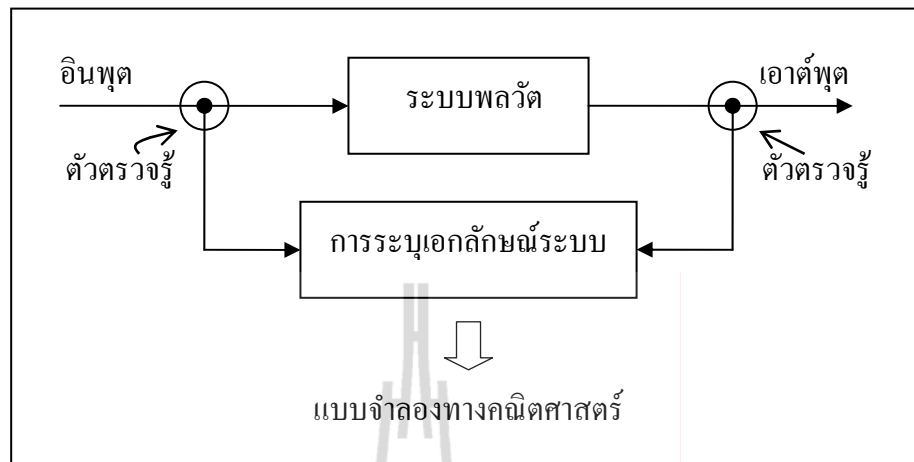
การประยุกต์ใช้งาน MATS กับการระบุเอกลักษณ์ระบบพลวัต

7.1 กล่าวนำ

การประยุกต์ใช้งาน MATS บทที่ผ่านมามีวัตถุประสงค์เพื่อการออกแบบตัวควบคุม แต่บทนี้เป็นอีกรูปแบบหนึ่ง เพื่อระบุเอกลักษณ์ระบบลูกตุ้มนาฬิกาพวงกรร (cart plus pendulum) ปัญหาของการระบุเอกลักษณ์สำหรับระบบนี้ คือแบบจำลองยังขาดความแม่นยำในย่านการทำงานแบบไม่เชิงเส้น ผู้วิจัยจึงเสนอแนวทางพัฒนาแบบจำลอง ด้วยการเสนอฟังก์ชันแรงกระทำต่อรถ (f) ซึ่งประกอบด้วยพารามิเตอร์จำนวน 15 ตัว ที่ต้องการคัดสรรอย่างเหมาะสมจึงอาศัย MATS ชนิด 15 หน่วยค้นหา ทำหน้าที่ค้นหาชุดพารามิเตอร์ดังกล่าว โดยประเมินความเหมาะสมของแบบจำลองระบบที่ได้ ด้วยค่าผลรวมกำลังสองค่าคลาดเคลื่อน (sum of square error : SSE) น้อยที่สุด จากการค้นหาชุดพารามิเตอร์ของฟังก์ชัน f ด้วย MATS ทำให้ได้แบบจำลองที่มีความแม่นยำสูงอย่างน่าพอใจ

7.2 ความรู้พื้นฐานการระบุเอกลักษณ์ระบบ

การระบุเอกลักษณ์ระบบ (system identification) เป็นกระบวนการที่สำคัญกระบวนการหนึ่งในบริบทระบบควบคุม กระบวนการดังกล่าวเป็นการคำนวณหาแบบจำลองทางคณิตศาสตร์ที่สมบูรณ์ของระบบ โดยอาศัยข้อมูลอินพุตและเอาต์พุตของระบบที่ได้จากการตรวจวัดโดยตรงจากตัวตรวจรู้ โดยอาจแทนด้วยแผนภาพได้ ดังรูปที่ 7.1



รูปที่ 7.1 การระบุเอกลักษณ์ระบบ

การระบุเอกลักษณ์ระบบพลวัต ดังรูปที่ 7.1 อาศัยข้อมูลความรู้เกี่ยวกับอินพุตและเอาต์พุตของระบบ ผ่านกระบวนการทางคณิตศาสตร์เพื่อพัฒนาแบบจำลองที่เหมาะสม ตามปกติแบบจำลองกล่องดำ (black-box model) จะแสดงในรูปของเวลาเต็มหน่วย ซึ่งข้อมูลอินพุตและเอาต์พุตของระบบได้รับการบันทึกผ่านการชักตัวอย่าง โดยใช้ตัวปฏิบัติการเลื่อน q (shift operator) กระทำกับสมการผลต่างเชิงเส้น (linear difference equation) โดยที่ $q^{nk} y(t) = y(t + nk)$ และ $q^{nk} y(t) = y(t + nk)$ สัญลักษณ์ t ที่ปรากฏ มีความหมายเป็นเวลาเต็มหน่วย (discrete time) แบบจำลองกล่องดำเชิงเส้น ชนิดเวลาเต็มหน่วยแสดงดังสมการที่ (7.1)

$$y(t) = \eta(t) + \omega(t) \quad (7.1)$$

เมื่อพจน์ $\eta(t)$ คือแบบจำลองเอาต์พุตที่อิสระต่อสัญญาณรบกวน (noise-free output model) และพจน์ $\omega(t)$ คือแบบจำลองการรบกวน (disturbance model) ซึ่งแสดงในสมการที่ (7.2) และ (7.3) ตามลำดับ

$$\eta(t) = G(q, \theta)u(t) \quad (7.2)$$

$$\omega(t) = H(q, \theta)e(t) \quad (7.3)$$

เมื่อพจน์ $u(t)$ คือสัญญาณอินพุต $e(t)$ คือสัญญาณรบกวน $y(t)$ คือสัญญาณเอาต์พุต และ θ คือเวกเตอร์ของพารามิเตอร์ในแบบจำลอง

กำหนดให้ $G(q, \theta)$ และ $H(q, \theta)$ เป็นฟังก์ชันถ่ายโอนในโดเมน q ดังสมการที่ (7.4) และ (7.5) ตามลำดับ

$$G(q, \theta) = \frac{B(q)}{F(q)} = \frac{b_1 q^{-nk} + b_2 q^{-nk-1} + \dots + b_{nb} q^{-nk-nb+1}}{1 + f_1 q^{-1} + \dots + f_{nf} q^{-nf}} \quad (7.4)$$

$$H(q, \theta) = \frac{C(q)}{D(q)} = \frac{1 + c_1 q^{-1} + \dots + c_{nc} q^{-nc}}{1 + d_1 q^{-1} + \dots + d_{nd} q^{-nd}} \quad (7.5)$$

ดังนั้นสมการที่ (7.2) อาจเขียนในอีกรูปแบบหนึ่งได้ดังสมการที่ (7.6)

$$\begin{aligned} & \eta(t) + f_1 \eta(t-1) + \dots + f_{nf} \eta(t-nf) \\ &= b_1 u(t-nk) + \dots + b_{nb} u[t-(nb+nk-1)] \end{aligned} \quad (7.6)$$

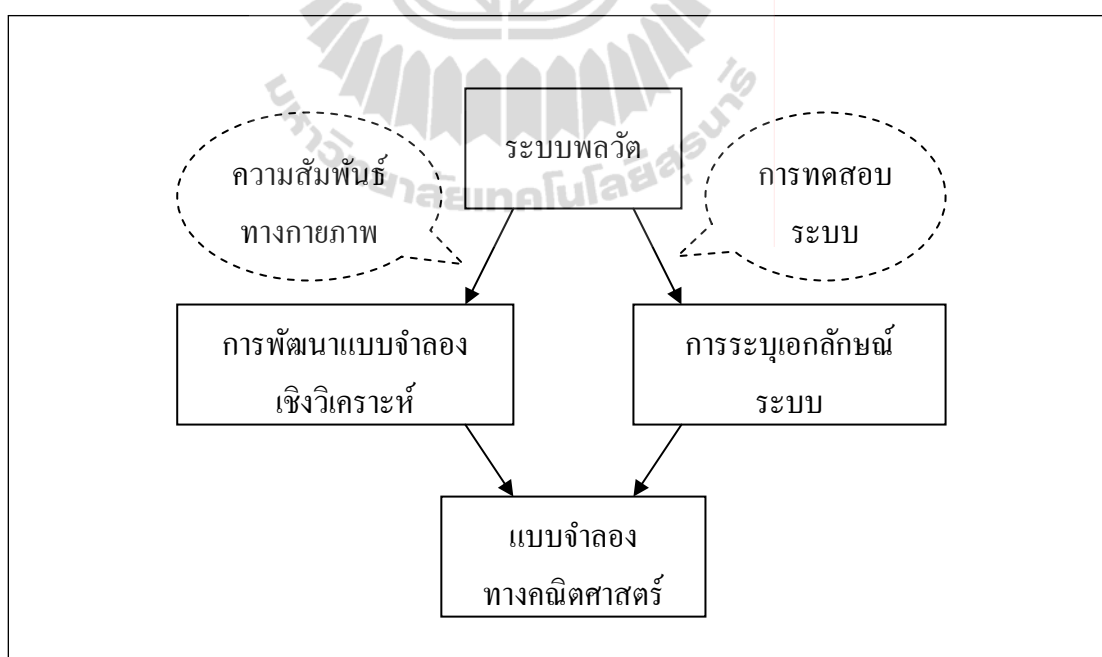
เมื่อ nk คือช่วงเวลาระวัง (delay time) ของการชักตัวอย่าง กำหนดให้คาบเวลาของการชักตัวอย่างเป็น T ซึ่งมีค่าคงที่ และเมื่อ $e(t)$ เป็นสัญญาณรบกวนขาว (white noise) สมการที่ (7.1) สามารถเขียนใหม่ได้ดังสมการที่ (7.7)

$$y(t) = G(q, \theta)u(t) + H(q, \theta)e(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \quad (7.7)$$

เมื่อ θ คือเวกเตอร์ของพารามิเตอร์ b_i c_i d_i และ f_i ของฟังก์ชันถ่ายโอน จะเห็นว่าแบบจำลองกล่องดำ ถูกกำกับไว้ด้วยอันดับของพหุนามจำนวน 5 ค่า คือ nb nc nd nf และ nk แบบจำลองในสมการ (7.7) เป็นที่รู้จักในนามแบบจำลอง BJ (Box-Jenkins model) นอกจากแบบจำลอง BJ ยังมีแบบจำลอง OE (output error model) แบบจำลอง ARX (autoregression extra input model) และ แบบจำลอง ARMAX (autoregression moving average of white noise extra (exogenous) input model) อาศัยความสัมพันธ์ของอินพุตและสัญญาณรบกวนเพื่อสร้างเอาต์พุต

ซึ่งรายละเอียดผู้สนใจสามารถอ่านเพิ่มเติมได้ในเอกสารประกอบคำสอนรายวิชาแบบจำลองระบบ (สราวุฒิ สุจิตจร, 2547)

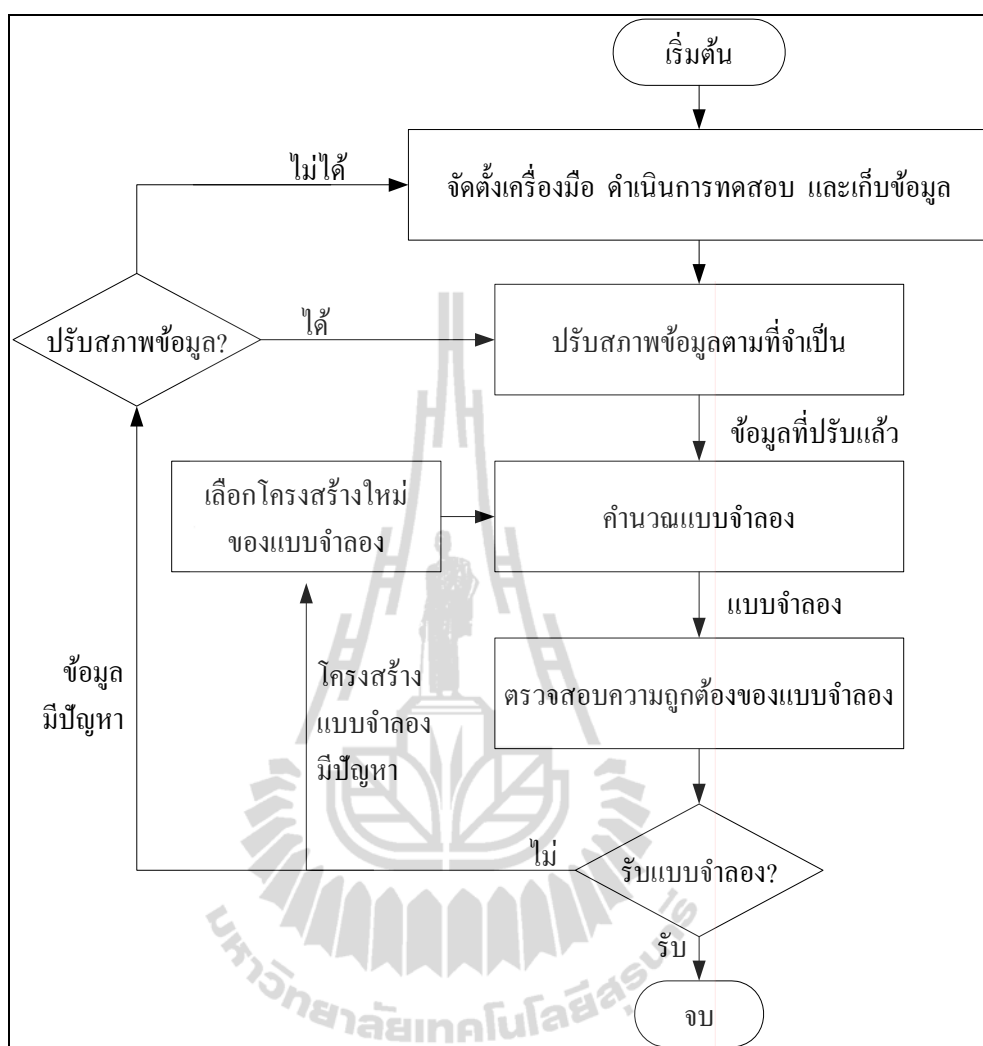
โดยทั่วไป การพัฒนาแบบจำลองทางคณิตศาสตร์ของระบบพลวัต อาศัยความสัมพันธ์ทางกายภาพของระบบ ซึ่งเป็นกฎทางธรรมชาติ ที่สามารถอธิบายพฤติกรรมทางพลวัตด้วยสมการคณิตศาสตร์ ดังเช่นกฎการเคลื่อนที่ของนิวตัน เป็นต้น วิธีการดังกล่าวเป็นพื้นฐานของการพัฒนาแบบจำลองทางคณิตศาสตร์ของระบบพลวัตต่าง ๆ ซึ่งอาจเรียกแบบจำลองที่ได้จากวิธีการดังกล่าวว่าแบบจำลองเชิงทฤษฎี (theoretical model) หรือแบบจำลองเชิงวิเคราะห์ (analytical model) ในระบบที่มีความซับซ้อนสูง อาจประสบปัญหาในเรื่องการหาสมการทางคณิตศาสตร์มาใช้อธิบายความสัมพันธ์ทางกายภาพของระบบ ส่งผลให้ไม่สามารถพัฒนาแบบจำลองทางคณิตศาสตร์ของระบบอย่างถูกต้อง วิธีการหนึ่งที่ถูกนำมาใช้แก้ปัญหาดังกล่าวคือ การทดสอบระบบด้วยสัญญาณอินพุตที่เหมาะสม สัญญาณอินพุตและเอาต์พุตของระบบได้รับการบันทึกและคำนวณด้วยกระบวนการทางคณิตศาสตร์ดังกล่าวเรียกว่า การระบุเอกลักษณ์ระบบ (system identification) รูปที่ 7.2 อธิบายกระบวนการพัฒนาแบบจำลองทางคณิตศาสตร์ทั้งสองแนวทางดังกล่าวมาข้างต้น



รูปที่ 7.2 การพัฒนาแบบจำลองทางคณิตศาสตร์

ขั้นตอนการระบุเอกลักษณ์ระบบสามารถอธิบายได้ด้วยแผนผังลำดับงาน ดังแสดงในรูปที่ 7.3 โดยเริ่มต้นด้วยการจัดตั้งเครื่องมือทดสอบ แล้วก็ดำเนินการทดสอบเพื่อเก็บข้อมูล ข้อมูลที่ได้นี้จะถูกปรับสภาพตามที่จำเป็น ต่อจากนั้นจึงแบ่งข้อมูลออกเป็นสองส่วนเพื่อการระบุเอกลักษณ์ระบบและเพื่อการตรวจสอบความถูกต้องของแบบจำลอง หากผลการตรวจสอบความถูกต้องของแบบจำลองเป็นที่ยอมรับได้ กระบวนการระบุเอกลักษณ์ก็จะสิ้นสุด หากผลการตรวจสอบความถูกต้องของแบบจำลองไม่เป็นที่ยอมรับ และพบว่าปัญหาเกิดจากโครงสร้างของแบบจำลอง ก็จะมีกระบวนการคัดเลือกโครงสร้างแบบจำลองใหม่ทดแทนของเดิม แล้ววนกลับเข้าสู่กระบวนการคำนวณแบบจำลองอีกครั้งหนึ่ง แต่หากพบว่าปัญหาเกิดจากข้อมูลที่ใช้ระบุเอกลักษณ์มีสภาพไม่สมบูรณ์ ก็จำเป็นต้องส่งกลับขั้นตอนการปรับสภาพข้อมูลใหม่ แล้วกลับเข้าสู่กระบวนการระบุเอกลักษณ์อีกครั้งหนึ่ง หรือหากสภาพข้อมูลชำรุดเกินจะแก้ไข ก็จำเป็นต้องกลับไปพิจารณาที่กระบวนการจัดตั้งเครื่องมือทดสอบและเก็บข้อมูลใหม่ แล้วดำเนินการซ้ำทั้งหมด จนกระทั่งแบบจำลองที่ได้เป็นที่ยอมรับตามเกณฑ์ที่กำหนดไว้ เกณฑ์การยอมรับแบบจำลองโดยทั่วไปใช้วิธีกำลังสองน้อยที่สุด (least square method) ระหว่างเอาต์พุตจากการวัด (y_{meas}) และเอาต์พุตจากการคำนวณแบบจำลองที่ได้ (y_{comp}) แสดงในรูปความแปรปรวน (variance) ดังสมการที่ (7.8)

$$V_N = \frac{1}{N} \sum_{t=1}^N (y_{meas,t} - y_{comp,t})^2 \quad (7.8)$$



รูปที่ 7.3 แผนผังลำดับงานระบุเอกลักษณ์ระบบ

เพื่อให้ได้ข้อมูลพลวัตของระบบที่มีความครบถ้วนสำหรับการระบุเอกลักษณ์ระบบ การวางแผนทดสอบระบบจึงมีความสำคัญ และจำเป็นต้องคำนึงถึงปัจจัยต่าง ๆ ได้แก่ การเลือกสัญญาณอินพุตในการทดสอบ การเลือกช่วงเวลาชักตัวอย่างสัญญาณ การปรับแต่งข้อมูล การเลือกโครงสร้างแบบจำลอง และการตรวจสอบความถูกต้องของแบบจำลอง ดังรายละเอียดต่อไปนี้

การเลือกข้อมูลสัญญาณอินพุต สัญญาณอินพุตที่เหมาะสมควรเป็นสัญญาณที่กระตุ้นระบบอย่างสม่ำเสมอ เช่น สัญญาณรายคาบต่าง ๆ เป็นต้น สำหรับการทดสอบระบบในโดเมนความถี่ ควรเลือกสัญญาณอินพุตที่มีการเลื่อนขนาดสองระดับอย่างสุ่ม (Eykhoff, 1974) สัญญาณดังกล่าวจะครอบคลุมทุกความถี่ ในกรณีการทดสอบระบบไม่เชิงเส้น ระดับของสัญญาณควรมากกว่าสองระดับ สำหรับการทดสอบระบบในโดเมนเวลา ควรเริ่มต้นทดสอบด้วยอินพุตแบบ

ขึ้นบันได เพื่อพิจารณาค่าคงที่เวลา (time constant) ของระบบ จากนั้นอาจประยุกต์สัญญาณขบวนพัลส์ (pulse train) ที่มีช่วงความกว้างแตกต่างกัน และในขบวนพัลส์ดังกล่าวควรมีอย่างน้อยหนึ่งช่วงความกว้างที่สามารถใช้แทนสัญญาณแบบขึ้นบันไดได้

ในทางปฏิบัติ สัญญาณที่มักถูกใช้เป็นอินพุตสำหรับทดสอบเพื่อการระบุเอกลักษณ์ระบบคือสัญญาณทวิภาคสุ่มเทียม หรือสัญญาณ PRBS (pseudo random binary signal) ซึ่งมีลักษณะเป็นลำดับพัลส์สองระดับที่ผ่านการมอดูเลตเชิงความกว้างอย่างสุ่ม และมีความเป็นรายคาบที่ยาวนานมาก สัญญาณทวิภาคสุ่มเทียมสามารถผลิตได้จากรีจิสเตอร์เลื่อนที่มีการป้อนกลับ ควบคู่ไปกับพีชคณิตมอดูโล (Modulo algebra) (Landau, 1990; Eykhoff, 1974; Johansson, 1993; Davies, 1970) ดังนั้นการเลือกสัญญาณอินพุตเพื่อทดสอบระบบอาจกล่าวได้ว่า สัญญาณทวิภาค หรือสัญญาณสองระดับ เหมาะสมสำหรับการทดสอบเชิงเส้น การเลือกย่านความถี่ของสัญญาณอินพุต ควรเลือกให้ครอบคลุมย่านความถี่ที่ระบบสามารถปฏิบัติงานได้ โดยอาจพิจารณาจากแบนด์วิดท์ (bandwidth) ของผลตอบสนองทางความถี่ของระบบ สำหรับการเลือกอินพุตเป็นสัญญาณขบวนพัลส์ จะต้องมีความถี่ของพัลส์ที่แตกต่างกัน และมีอย่างน้อยหนึ่งช่วงความกว้างพัลส์ที่สามารถใช้แทนสัญญาณแบบขึ้นบันได เพื่อตรวจสอบหาค่าคงที่เวลาของระบบ

การเลือกช่วงเวลาชักตัวอย่างสัญญาณ ในทฤษฎีบทการชักตัวอย่างสัญญาณ (sampling theorem) ความถี่ในการชักตัวอย่างสัญญาณต้องมีค่ามากกว่าความถี่ในควิส (Nyquist frequency) ไม่เช่นนั้นแล้ว ข้อมูลที่ได้จะปรากฏการเคลือบแฝง (alias) ซึ่งไม่สามารถใช้แทนพลวัตของระบบที่แท้จริงได้ ในทางปฏิบัติ ช่วงเวลาชักตัวอย่างสัญญาณ (sampling interval) สามารถเลือกให้มีการชักตัวอย่างสัญญาณอย่างน้อย 5 ถึง 8 ครั้ง ภายในช่วงเวลาไต่ขึ้น (rise time) ของผลตอบสนองในโดเมนของระบบ ซึ่งจะกำหนดให้มีการชักตัวอย่างสัญญาณที่ละเอียดกว่านี้ก็ได้ ความถี่ของการชักตัวอย่างสัญญาณควรมีค่าประมาณ 10 เท่าของแบนด์วิดท์ของระบบ (Ljung and Glad, 1994)

การปรับแต่งข้อมูลในการทดสอบระบบและเก็บข้อมูลนั้น ข้อมูลบางส่วนอาจมีความไม่เหมาะสม เช่น ระดับของข้อมูลเกิดการขยับเลื่อนไปจากกลุ่มข้อมูลหลัก หรือข้อมูลได้รับการรบกวนด้วยสัญญาณรบกวนความถี่สูง เป็นต้น ดังนั้นข้อมูลจะต้องได้รับการปรับแต่งให้เหมาะสมก่อนที่จะนำไปทำการระบุเอกลักษณ์ระบบ การปรุงแต่งข้อมูลอาจใช้ตัวกรองแบบต่าง ๆ เช่น ใช้ตัวกรองผ่านต่ำ (low-pass filter) เพื่อกรองสัญญาณรบกวนความถี่สูงออกไป ในบางกรณีอาจต้องพึ่งพาเทคนิคเชิงตัวเลข ตลอดจนเทคนิคการทำนายข้อมูล เป็นต้น

การตรวจสอบความถูกต้องของแบบจำลอง เป็นการทดสอบความถูกต้องของแบบจำลอง เพื่อดูความสมเหตุสมผลในการใช้แบบจำลองอธิบายพลวัตของระบบ การตรวจสอบจะสอดคล้องกับการพิจารณาคุณภาพของแบบจำลอง โดยให้การพิจารณาถึงสองประเด็นต่อไปนี้ (1) แบบจำลองสามารถพัฒนาขึ้นจากชุดข้อมูลที่แตกต่างกันหรือคนละชุดกัน (จากระบบพลวัตระบบเดียวกัน)

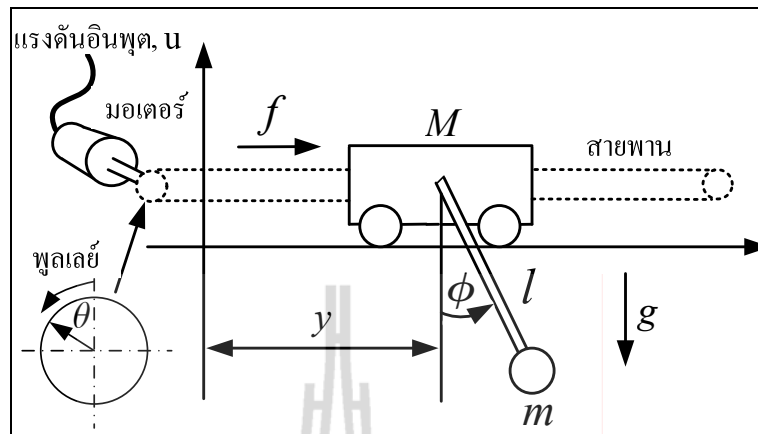
โดยที่แบบจำลองที่พัฒนาขึ้นจะต้องมีคุณสมบัติที่เหมือนกันหรือคล้ายคลึงกัน กล่าวอีกนัยหนึ่งก็คือไม่ว่าโครงสร้างของแบบจำลองจะมีลักษณะแตกต่างกันอย่างไร จะต้องสามารถสื่อความหมายถึงพลวัตของระบบได้อย่างครอบคลุม แม้ว่าแบบจำลองนั้น ๆ จะพัฒนาขึ้นมาจากข้อมูลต่างชุดกันก็ตาม (2) แบบจำลองจะต้องแสดงถึงพฤติกรรมทางพลวัตของระบบได้อย่างถูกต้อง ซึ่งหมายความว่าเอาต์พุตของแบบจำลองจะต้องสอดคล้องกับเอาต์พุตของระบบที่ได้จากการทดสอบ แม้ว่าข้อมูลอินพุตที่ใช้ทดสอบความถูกต้องของแบบจำลองเพื่อดูพฤติกรรมทางพลวัตนั้นจะเป็นข้อมูลต่างชุดกับข้อมูลอินพุตที่ใช้ในการระบุเอกลักษณ์ระบบก็ตาม

แนวทางปฏิบัติทั่วไปนั้น ข้อมูลที่ได้จากการทดสอบจะถูกแบ่งออกเป็นสองส่วน ข้อมูลส่วนแรก ใช้เพื่อการระบุเอกลักษณ์แบบจำลองของระบบ ข้อมูลส่วนหลังใช้เพื่อตรวจสอบความถูกต้องของแบบจำลอง หากผลการตรวจสอบความถูกต้องของแบบจำลองด้วยข้อมูลส่วนหลังนี้มีความสมเหตุสมผล จะถือว่าแบบจำลองดังกล่าวสามารถใช้เพื่ออธิบายพฤติกรรมทางพลวัตของระบบและสามารถใช้แทนระบบพลวัตตามวัตถุประสงค์ในบริบทระบบควบคุมต่อไปได้

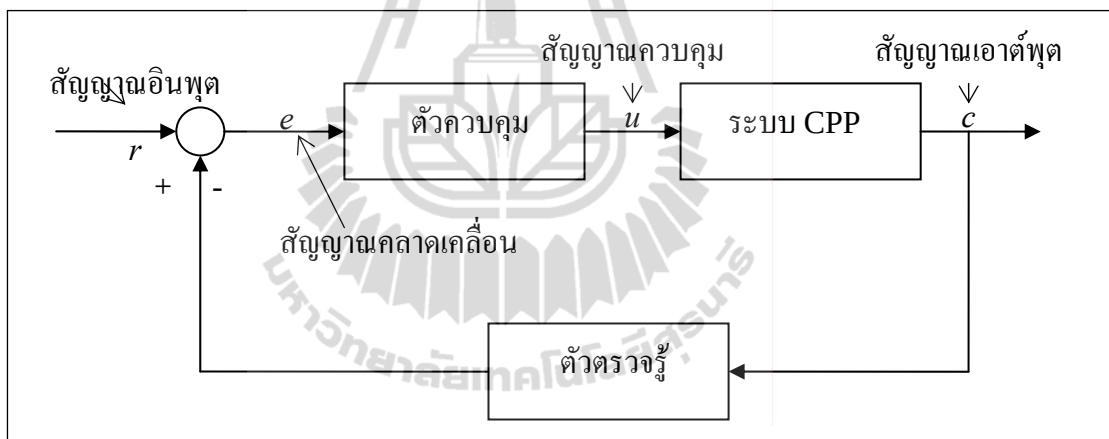
แบบจำลองทางคณิตศาสตร์ของระบบพลวัต อาจมีรูปแบบที่แตกต่างกันออกไป เช่นอยู่ในรูปของฟังก์ชันถ่ายโอน สมการอนุพันธ์สามัญ (ODE) หรือสมการอนุพันธ์ย่อย (PDE) หรือสมการอนุพันธ์ไม่เชิงเส้น ทั้งนี้รูปแบบของแบบจำลองทางคณิตศาสตร์ของระบบ จะขึ้นอยู่กับธรรมชาติของระบบเป็นสำคัญ ดังที่ได้กล่าวมาข้างต้นแล้วว่า แบบจำลองทางคณิตศาสตร์ที่ให้ความสนใจในขณะนี้ เป็นแบบจำลองกลองเตา ที่มีรูปแบบเป็นสมการอนุพันธ์ไม่เชิงเส้น ซึ่งจะกล่าวถึงรายละเอียดในหัวข้อถัดไป

7.3 ระบบลูกตุ้มนาฬิกาพวงนกรถ

ระบบลูกตุ้มนาฬิกาพวงนกรถ (cart plus pendulum : CPP) สามารถแสดงส่วนประกอบที่สำคัญได้ดังรูปที่ 7.4 เป็นระบบที่ประกอบด้วยลูกตุ้ม (pendulum) ที่ต่อเข้ากับรถ (cart) โดยที่ลูกตุ้มจะแกว่งไกวอยู่ในระนาบ x และ y อย่างอิสระตามการเคลื่อนที่ของรถที่ถูกขับด้วยแรง f (นิวตัน) แรงดังกล่าวกำเนิดขึ้นจากการหมุนของมอเตอร์และส่งผ่านแรงบิดมาตามสายพาน สายพานจึงเป็นอุปกรณ์แปลงแนวแรงจากแรงบิดไปเป็นแรงเชิงเส้น โดยที่มุม ϕ คือมุมการแกว่งไกวของลูกตุ้ม (เรเดียน) u คือสัญญาณอินพุตของมอเตอร์ (โวลต์) y คือตำแหน่งของรถ (เมตร) M คือมวลของรถ (กิโลกรัม) m คือมวลของลูกตุ้ม (กิโลกรัม) l คือความยาวของก้านลูกตุ้ม (เมตร) g คือความเร่งเนื่องจากแรงโน้มถ่วง (เมตร/วินาที²)



รูปที่ 7.4 ระบบ CPP



รูปที่ 7.5 โครงสร้างระบบ CPP ของบริษัท Feedback

ระบบ CPP ที่ใช้ในงานวิจัยนี้เป็นของบริษัท Feedback (Feedback Instruments Ltd.) มีโครงสร้างเป็นระบบควบคุมวงรอบปิด ดังรูปที่ 7.5 มีตัวควบคุมแบบฐานกฎ ที่แบ่งโหมดการควบคุมออกเป็นสองโหมดคือ โหมดเครน (crane mode) และโหมดแกว่งขึ้น (swing-up mode) การควบคุมระบบดำเนินการผ่านโปรแกรมที่บริษัทได้พัฒนาขึ้นจาก MATLAB™ และ Simulink โครงสร้างของระบบควบคุมรวมทั้งโปรแกรมการควบคุมหลักไม่ได้รับการอนุญาตให้มีการเปลี่ยนแปลง ดังนั้นแนวทางการทดสอบระบบจะใช้การทดสอบแบบวงรอบปิด จากรูปที่ 7.3 เมื่อต้องการระบุเอกลักษณ์เฉพาะระบบ CPP ข้อมูลความรู้ที่ใช้คือข้อมูลของสัญญาณควบคุม u ซึ่งถือเป็นสัญญาณอินพุตของระบบ CPP และข้อมูลของสัญญาณเอาต์พุต c (ตำแหน่งเชิงมุมของ

ลูกตุ้ม) ซึ่งถือเป็นสัญญาณเอาต์พุตของระบบ CPP การระบุเอกลักษณ์ระบบในลักษณะดังกล่าวถือเป็นการประมาณแบบจำลองระบบย่อย (subsystem) (Johansson, 1993)

แบบจำลองทางคณิตศาสตร์ของระบบ CPP ในงานวิจัยนี้เป็นแบบจำลองไม่เชิงเส้นพัฒนาขึ้นมาจากสมการการเคลื่อนที่ของลากรานจ์ (Lagrange's equation of motion) ซึ่ง ลากรานเจียนถูกกำหนดให้เท่ากับผลต่างระหว่างพลังงานจลน์ (kinetic energy) และพลังงานศักย์ (potential energy) ของระบบ (Cannon, 1967; Bloch, Leonard, and Marden, 2000) ดังแสดงในสมการที่ (7.9)

$$L(\phi, y, \dot{\phi}, \dot{y}) \triangleq K(\phi, y, \dot{\phi}, \dot{y}) - V(\phi) \quad (7.9)$$

เมื่อ $L(\cdot)$ คือ ฟังก์ชันลากรานจ์ $K(\cdot)$ คือ พลังงานจลน์ $V(\cdot)$ คือ พลังงานศักย์ พลังงานจลน์ $K(\cdot)$ เป็นผลรวมของพลังงานจลน์ของรถ (K_{cart}) และพลังงานจลน์ของลูกตุ้ม (K_{pend}) ดังแสดงในสมการที่ (7.10) และ (7.11) ตามลำดับ

$$K(\phi, y, \dot{\phi}, \dot{y}) = K_{cart} + K_{pend} \quad (7.10)$$

$$K(\phi, y, \dot{\phi}, \dot{y}) = \frac{1}{2} M (\dot{v}_{cart})^2 + \frac{1}{2} m (\dot{v}_{pend})^2 \quad (7.11)$$

เมื่อ $v_{cart} = \dot{y}$ ขณะที่ $v_{pend} = [v_{pend_horizontal}, v_{pend_vertical}]$ ซึ่งจะต้องพิจารณาดังนี้คือเมื่อกำหนดให้ลูกตุ้มแกว่งไกวอยู่ด้านล่างของรถ $v_{pend} = [\dot{y} + l \cos(\phi) \dot{\phi}, l \sin(\phi) \dot{\phi}]$ และเมื่อกำหนดให้ลูกตุ้มแกว่งไกวอยู่ด้านบนของรถ $v_{pend} = [\dot{y} + l \cos(\phi) \dot{\phi}, -l \sin(\phi) \dot{\phi}]$ ดังนั้นสมการที่ (7.10) สามารถเขียนได้ใหม่เป็นดังสมการที่ (7.12) และ (7.13) สมการดังกล่าวเป็นสมการพลังงานจลน์ของระบบที่สามารถใช้แทนได้ทั้งในกรณีที่ลูกตุ้มแกว่งไกวอยู่ด้านล่างและด้านบนของรถ

$$K(\phi, y, \dot{\phi}, \dot{y}) = \frac{1}{2} M (\dot{y})^2 + \frac{1}{2} m [(\dot{y})^2 + 2l \dot{\phi} \dot{y} \cos(\phi) + l^2 (\dot{\phi})^2] \quad (7.12)$$

$$K(\phi, y, \dot{\phi}, \dot{y}) = \frac{1}{2} \begin{bmatrix} \dot{\phi} & \dot{y} \end{bmatrix} \begin{bmatrix} ml^2 & ml \cos(\phi) \\ ml \cos(\phi) & M + m \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{y} \end{bmatrix} \quad (7.13)$$

สำหรับพลังงานศักย์ $V(\cdot)$ จะเป็นผลรวมของพลังงานศักย์ของรถ และพลังงานศักย์ของลูกตุ้ม แต่เนื่องจากพลังงานศักย์ของรถมีค่าเท่ากับศูนย์ (รถอยู่ในระดับอ้างอิง) ดังนั้นพลังงานศักย์ $V(\cdot)$ จะพิจารณาจากพลังงานศักย์ของลูกตุ้มเพียงอย่างเดียว ในกรณีที่ลูกตุ้มแกว่งไกวอยู่ด้านบนและด้านล่างของรถ พลังงานศักย์จะแสดงดังสมการที่ (7.14) และ (7.15) ตามลำดับ

$$V(\phi) = -mgl \cos(\phi) \quad (7.14)$$

$$V(\phi) = mgl \cos(\phi) \quad (7.15)$$

กำหนดให้ $\alpha = ml^2$ $\beta = ml$ $\gamma = (M+m)$ และ $D = mgl$ เมื่อลูกตุ้มแกว่งไกวอยู่ด้านล่างของรถและ $D = -mgl$ เมื่อลูกตุ้มแกว่งไกวอยู่ด้านบนของรถ ฟังก์ชันลากรางจ์ $L(\cdot)$ จะเขียนได้ดังสมการที่ (7.16)

$$L(\phi, y, \dot{\phi}, \dot{y}) = \frac{1}{2} \left[\alpha (\dot{\phi})^2 + 2\beta \cos(\phi) \dot{\phi} \dot{y} + \gamma (\dot{y})^2 \right] + D \cos(\phi) \quad (7.16)$$

กำหนดให้ p_ϕ คือโมเมนตัมเชิงมุม และ p_y คือโมเมนตัมเชิงเส้น ซึ่งแสดงในสมการที่ (7.17) และ (7.18) ตามลำดับ

$$p_\phi = \frac{\partial L}{\partial \dot{\phi}} = \alpha \dot{\phi} + \beta \cos(\phi) \dot{y} \quad (7.17)$$

$$p_y = \frac{\partial L}{\partial \dot{y}} = \gamma \dot{y} + \beta \cos(\phi) \dot{\phi} \quad (7.18)$$

สมการการเคลื่อนที่ของลากรางจ์สำหรับระบบ CPP แสดงดังสมการที่ (7.19) และ (7.20)

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\phi}} - \frac{\partial L}{\partial \phi} = 0 \quad (7.19)$$

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{y}} = f \quad (7.20)$$

จากสมการที่ (7.19) และ (7.20) สามารถเขียนได้ใหม่เป็นดังสมการที่ (7.21) และ (7.22) ตามลำดับ

$$\alpha \ddot{\phi} + \beta \cos(\phi) \ddot{y} + D \sin(\phi) = 0 \quad (7.21)$$

$$\gamma \ddot{y} + \beta \cos(\phi) \ddot{\phi} - \beta (\dot{\phi})^2 \sin(\phi) = f \quad (7.22)$$

เมื่อแทนค่า α β γ และ D ลงในสมการที่ (7.21) และ (7.22) จะได้แบบจำลองทางคณิตศาสตร์ของระบบ CPP ในรูปของสมการอนุพันธ์ไม่เชิงเส้น เมื่อพิจารณาให้ลูกตุ้มแกว่งไกวอยู่ด้านล่างของรถ แบบจำลองทางคณิตศาสตร์ของระบบ CPP ไม่เชิงเส้น แสดงดังสมการที่ (7.23) และ (7.24) ตามลำดับ ซึ่งเป็นกรณีศึกษาสำหรับงานวิจัยนี้

$$\ddot{\phi} = \frac{f \cos(\phi) + 0.5ml \sin(2\phi)(\dot{\phi})^2 + (M + m)g \sin(\phi)}{l[m \cos^2(\phi) - (M + m)]} \quad (7.23)$$

$$\ddot{y} = \frac{f + 0.5mg \sin(2\phi) + ml \sin(\phi)(\dot{\phi})^2}{[(M + m) - m \cos^2(\phi)]} \quad (7.24)$$

ทั้งนี้สมการ (7.2) ถึง (7.17) มีรายละเอียดปรากฏในวิทยานิพนธ์ดุษฎีบัณฑิต (เดชาวงดาวเรือง, 2547) รวมถึงพารามิเตอร์ที่ได้จากการวัดของระบบ CPP ได้แก่ $m = 0.251$ กิโลกรัม $M = 0.9549$ กิโลกรัม $l = 0.4$ เมตร และ $g = 9.81$ เมตร/วินาที² เป็นต้น

7.4 การระบุเอกลักษณ์ระบบ CPP

การระบุเอกลักษณ์ระบบ CPP สำหรับงานวิจัยนี้ อาศัยแบบจำลองกล่องเทา โดยพึ่งพาข้อมูลจากสองส่วน คือ จากการทดลองระบบจริงและจากการคำนวณเชิงตัวเลข ซึ่งความเหมาะสมของแบบจำลองกล่องเทาที่ได้ จะพิจารณาจากผลรวมกำลังสองของค่าคลาดเคลื่อน (sum of square error : SSE) ทำหน้าที่เป็นตัวชี้วัด ยิ่งมีค่า SSE ต่ำ แบบจำลองยิ่งเหมาะสมมาก

สมการ (7.23) และ (7.24) เป็นแบบจำลองกล่องขาว (white-box model) เพราะมีที่มาจากทฤษฎีและกฎทางกลศาสตร์ที่ได้รับการยอมรับอย่างกว้างขวาง แต่ในสมการยังมีตัวแปรบางค่าที่ยังไม่ทราบค่าแน่นอนได้แก่แรง f ที่กระทำกับตัวรถ ประกอบกับการยากที่จะทำการติดตั้งตัวตรวจวัดค่าแรง f ดังนั้นงานวิจัยนี้จึงเสนอแบบจำลองกล่องดำ (black-box model) เพื่อประมาณค่าแรง f ด้วยค่าที่วัดได้ของ θ y ϕ และ u ก่อรูปดังสมการที่ (7.25)

$$f(\theta, y, \phi, u) = A\ddot{\theta} + B\dot{\theta} + C\theta + D\ddot{y} + E\dot{y} + Fy + G\ddot{\phi} + H\dot{\phi} + I\phi + \dots \\ \dots Je^{Ku} + Lu^2 + Mu + Nu^{-1} + O \quad (7.25)$$

A B และ C สะท้อนความสัมพันธ์ของพิกัด ความเร็วและความเร่งเชิงมุมของโรเตอร์ ต่อแรงกระทำต่อรถ f

D E และ F สะท้อนความสัมพันธ์ของพิกัด ความเร็วและความเร่งเชิงเส้นของรถต่อแรงกระทำต่อรถ f

G H และ I สะท้อนความสัมพันธ์ของพิกัด ความเร็วและความเร่งเชิงมุมของเพนดูลัม ต่อแรงกระทำต่อรถ f

J K L M และ N สะท้อนความสัมพันธ์ของแรงดันอินพุตของมอเตอร์ไฟฟ้ากระแสตรง ต่อแรงกระทำต่อรถ f

O แทนค่าคงค้างใด ๆ (arbitrary residues) ต่อแรงกระทำต่อรถ f

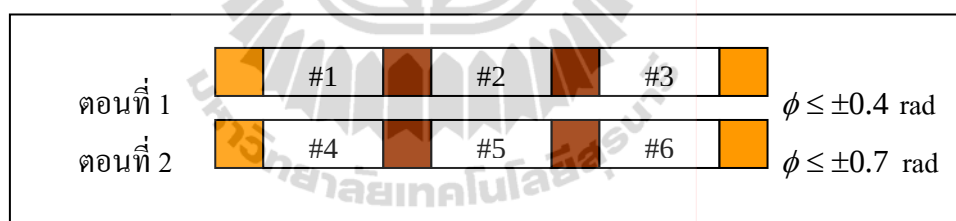
แรง f ที่ถูกประมาณด้วยแบบจำลองกล่องดำตามสมการที่ (7.25) เมื่อแทนลงไป ในสมการพลวัตของระบบ CPP ตามแบบจำลองกล่องขาว ดังสมการที่ (7.23) และ (7.24) จึงเข้าใจได้ โดยง่ายว่าการระบุเอกลักษณ์ระบบ CPP กระบวนการที่เสนอข้างต้นเป็นการหนึ่งในวิธีการระบุเอกลักษณ์กล่องเทาหรือแบบจำลองกล่องเทา ซึ่งจะได้กล่าวถึงรายละเอียดของวิธีดำเนินงานดังต่อไปนี้

7.5 การเตรียมข้อมูลจากการทดสอบ

ตามหลักการระบุเอกลักษณ์ (identification) ก่อนที่นำข้อมูลที่ได้จากการทดลองเข้ากระบวนการระบุเอกลักษณ์ควรตัดข้อมูลส่วนต้นและส่วนปลายออก ให้เหลือเฉพาะส่วนกลางที่มากพอสำหรับดำเนินการระบุเอกลักษณ์และการตรวจสอบความถูกต้องของแบบจำลอง (validation) ซึ่งโดยทั่วไปข้อมูลที่จะใช้ระบุเอกลักษณ์จะใช้ประมาณ 60% - 80% ที่เหลือจะใช้ประเมินความถูกต้องของแบบจำลอง

ข้อมูลสำหรับการระบุเอกลักษณ์ครั้งนี้ ได้จากการทดลองสองชุด ตามขนาดของมุม ϕ ที่กว้าง โดยชุดที่หนึ่งมีค่า ϕ อยู่ในย่าน ± 0.4 เรเดียน ส่วนชุดที่สองมีค่า ϕ อยู่ในย่าน ± 0.7 เรเดียน ข้อมูลดิบที่เก็บได้จะถูกตัดส่วนหัวและท้ายออกตามหลักการเตรียมข้อมูลทางสถิติ แล้วนำข้อมูลดิบที่ได้แต่ละชุดมาแบ่งออกเป็นสามช่วง ช่วงละ 250 ค่า แต่ละช่วงมีระยะห่างของข้อมูล 50 ค่า ด้วยช่วงเวลาการชักตัวอย่าง 0.15 วินาที

เมื่อเสร็จสิ้นกระบวนการเตรียมข้อมูลจะได้ข้อมูลทั้งหมด 6 ช่วง ๆ ละ 250 ค่า จากข้อมูลดิบ 2 ตอน ดังรูปที่ 7.6



รูป 7.6 การเตรียมข้อมูลดิบสำหรับการระบุเอกลักษณ์

ข้อมูลช่วงที่ 2 - 5 จำนวน 4 ช่วง จากทั้งหมด 6 ช่วง ซึ่งคิดเป็นร้อยละ 66.67 ของข้อมูลทั้งหมด ได้รับการนำไปใช้เพื่อระบุเอกลักษณ์ ส่วนข้อมูลที่เหลือได้แก่ช่วงที่ 1 และ 6 จำนวน 2 ช่วง ซึ่งคิดเป็นร้อยละ 33.33 ของข้อมูลทั้งหมด ใช้เพื่อการประเมินความถูกต้องของแบบจำลอง สอดคล้องกับหลักการแบ่งข้อมูลเพื่อการระบุเอกลักษณ์

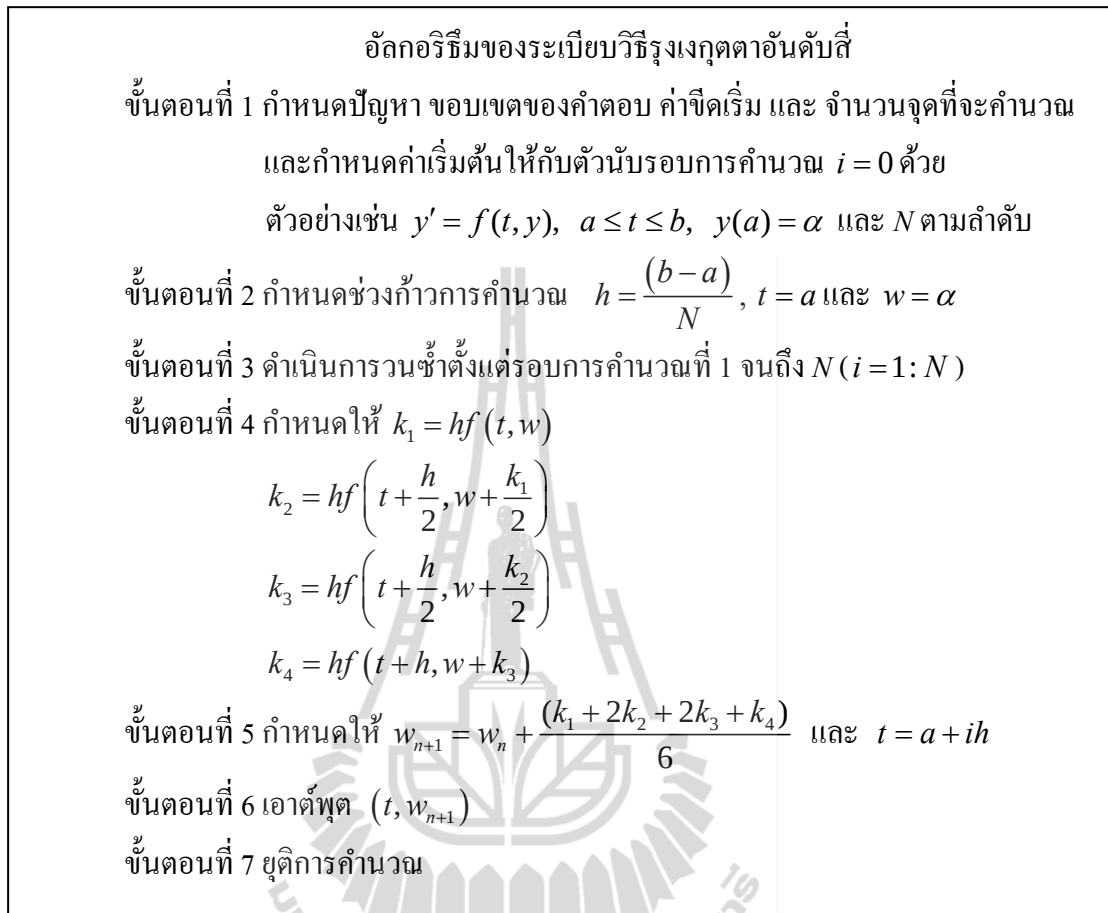
ข้อมูลที่ได้จากการเตรียมด้วยวิธีการดังกล่าวข้างต้น จะถูกนำไปใช้ร่วมกับการแก้สมการที่ (7.16) และ (7.17) โดยเลือกแก้ปัญหาแบบการคำนวณเชิงตัวเลขด้วยระเบียบวิธีรุงเงตตามันดับ 4 ร่วมกับข้อมูลที่ได้จากการทดลองที่ผ่านการเตรียมดังรูปที่ 7.6 เพื่อหาค่าของ θ $\dot{\theta}$ $\ddot{\theta}$ ϕ $\dot{\phi}$ $\ddot{\phi}$ y $\dot{y}$ $\ddot{y}$ u โดยการประมวลผลบนเครื่องคอมพิวเตอร์ตั้งโต๊ะ Pentium 4 ความเร็ว 2.4 จิกะเฮิร์ตซ์ หน่วยความจำ RAM ขนาด 640 เมกกะไบต์ ด้วยโปรแกรม MATLAB™

7.6 การระบุเอกลักษณ์ระบบพลวัตด้วย MATS

การระบุเอกลักษณ์ระบบดังกล่าวดำเนินการด้วย MATS เพื่อหาค่าพารามิเตอร์ทั้ง 15 ตัว คือ A B C D E F G H I J K L M และ O (รายละเอียดปรากฏในหัวข้อ 7.4) ที่เหมาะสม พารามิเตอร์เหล่านี้จะถูกใช้คำนวณหาค่าฟังก์ชันวัตถุประสงค์ ซึ่งถูกเขียนด้วย MATLABTM โดยพิจารณาที่ผลรวมค่าความคลาดเคลื่อนกำลังสอง (sum square of error : SSE) ซึ่งได้จากการเปรียบเทียบกันระหว่าง มุมเบี่ยงเบนของลูกตุ้มนาฬิกาที่ได้จากการคำนวณ (ϕ_{comp}) กับมุมเบี่ยงเบนของลูกตุ้มนาฬิกาที่ได้จากการวัด (ϕ_{meas}) แสดงดังสมการ (7.26)

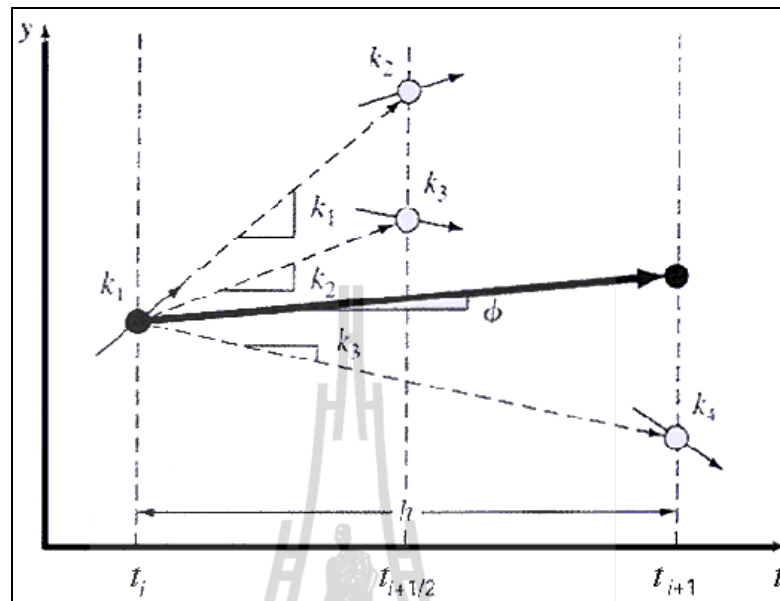
$$SSE = \sum_{i=1}^k (\phi_{meas,i} - \phi_{comp,i})^2 \quad (7.26)$$

การคำนวณค่า ϕ_{comp} เริ่มต้นด้วยด้วยการกำหนดค่า f ตามสมการที่ (7.25) แล้วแทนลงในสมการที่ (7.23) และ (7.24) หลังจากจึงดำเนินการแก้สมการด้วยระเบียบวิธีรุงเงกุตตา (Runge Kutta method) เพื่อหาคำตอบของสมการที่ (7.23) และ (7.24) นั่นคือค่า ϕ_{comp} นั้นเอง ดังมีการดำเนินงานต่อไปนี้ ระเบียบวิธีรุงเงกุตตาอันดับสี่ ดังปรากฏใน Richard and Fairs (2001) ประกอบด้วย 7 ขั้นตอนหลัก แสดงดังรูปที่ 7.7 ส่วนรายละเอียดโปรแกรมแสดงอยู่ในภาคผนวก ค.



รูปที่ 7.7 ขั้นตอนของระเบียบวิธีรุงเงดาคอนดั้ม

y คือฟังก์ชันเอาต์พุต t คือเวลา $y' = dy/dt$ คืออนุพันธ์ของ y เทียบกับเวลา $f(t, y)$ คือฟังก์ชันที่เกิดจาก t และ y a และ b คือขอบเขตล่างและขอบเขตบนของช่วงเวลา t ตามลำดับ α คือค่าของฟังก์ชัน y ณ เวลา $t = a$ N คือจำนวนจุดที่ต้องการคำนวณ h คือช่วงก้าวการคำนวณ w คือค่าฟังก์ชันเอาต์พุตในวงรอบการคำนวณ k_1 คือค่าผลคูณของช่วงก้าวการคำนวณ h กับค่าฟังก์ชันเอาต์พุตที่แทนมีอินพุตเป็น t และ w k_2 คือค่าผลคูณ h กับฟังก์ชันเอาต์พุตที่แทนมีอินพุตเป็น $(t + h/2)$ และ $(w + k_1/2)$ k_3 คือค่าผลคูณ h กับฟังก์ชันเอาต์พุตที่แทนมีอินพุตเป็น $(t + h/2)$ และ $(w + k_2/2)$ และ k_4 คือค่าผลคูณ h กับฟังก์ชันเอาต์พุตที่แทนมีอินพุตเป็น $(t + h)$ และ $(w + k_3)$ สามารถแสดงความสัมพันธ์ของปริมาณเหล่านี้ได้ดังรูปที่ 7.8



รูปที่ 7.8 ลักษณะทั่วไปของ k_1 k_2 k_3 k_4 และ h ในระเบียบวิธีรุงเงกุตตาอันดับสี่

สำหรับปัญหานี้ $y' = f(t, y)$ ก็คือสมการที่ (7.23) และ (7.24) ที่แทนค่าแรงด้วยฟังก์ชันจากสมการที่ (7.25) นั่นเอง การคำนวณเลือกตั้งค่า h จากคาบเวลาการชักตัวอย่างจริง $h = 0.15$ วินาที และ $N = 250$ ส่วนค่าขีดเริ่มต่าง ๆ อาศัยค่าผลการทดสอบ เอาต์พุตที่สนใจในที่นี้คือ ϕ หรือตำแหน่งเชิงมุมของลูกตุ้ม เมื่อคำนวณเสร็จสิ้นจะได้ผลเฉลยของชุดสมการ (7.23) (7.24) และ (7.25) เป็นค่า ϕ_{comp} จำนวน 250 ค่า ซึ่งรายละเอียดโปรแกรมการแก้ชุดสมการดังกล่าวด้วยระเบียบวิธีรุงเงกุตตา ปรากฏอยู่ในภาคผนวก ค โปรแกรมการระบุเอกลักษณ์ระบบลูกตุ้มผกผัน

ส่วนค่า ϕ_{meas} ได้จากการผลการทดสอบจริง โดยแบ่งข้อมูลออกเป็นสองส่วน ส่วนที่หนึ่งเป็นส่วนใหญ่สำหรับการระบุเอกลักษณ์ ได้แก่ข้อมูลช่วงที่ 2.5 จำนวน 4 ช่วง ๆ ละ 250 ค่า ($k = 250$) และส่วนที่สองเป็นส่วนใหญ่ใช้ตรวจสอบความถูกต้องแบบจำลอง ได้แก่ข้อมูลช่วงที่ 1 และ ช่วงที่ 6 จำนวน 2 ช่วง ๆ ละ 250 ค่า เช่นกัน ขั้นตอนการคำนวณค่า SSE สามารถสรุปได้ดังรูปที่ 7.9

ขั้นตอนการคำนวณค่า SSE สามารถสรุปได้ดังนี้

ขั้นตอนที่ 1 (การเริ่มต้น)

รับค่าพารามิเตอร์ทั้ง 15 ตัว

ขั้นตอนที่ 2 (คำนวณหาค่ามุมแกว่งลูกตุ้มนาฬิกา)

แทนค่าพารามิเตอร์ทั้ง 15 ตัว ลงในสมการ (7.23) - (7.25) แล้วใช้ RK4 คำนวณหาค่ามุมแกว่งของลูกตุ้มนาฬิกา ϕ_{comp}

ขั้นตอนที่ 3 (คำนวณหาค่า SSE)

รับค่ามุมแกว่งลูกตุ้มนาฬิกาจากการวัด ϕ_{meas} และจากการคำนวณ ϕ_{comp} แทนลงในสมการ (7.26) คำนวณหาค่า SSE

รูปที่ 7.9 ขั้นตอนการคำนวณค่า SSE

การระบุเอกลักษณ์ตามขั้นตอนดังกล่าวข้างต้น เริ่มต้นด้วย $\phi_{meas,2i}$ (ข้อมูลช่วงที่ 2 เพื่อการระบุเอกลักษณ์) จะได้ $\phi_{comp,2i}$ และ SSE_{2i} แล้วดำเนินการทำนองเดียวกันกับข้อมูลช่วงที่ 3 4 และ 5 จะได้ค่า SSE_{3i} SSE_{4i} และ SSE_{5i} ตามลำดับ ส่วนของการตรวจสอบความถูกต้องของแบบจำลอง อาศัยวิธีการเดียวกันกับการระบุเอกลักษณ์ แต่แตกต่างกันที่ช่วงข้อมูล โดยจะใช้ข้อมูลช่วงที่ 1 และ ช่วงที่ 6 สิ่งที่ได้ก็คือ SSE_{1v} และ SSE_{6v} ตามลำดับ กระบวนการดังรูปที่ 7.8 จะถูกเรียกใช้ทั้งหมด 6 ครั้ง จึงจะได้ SSE ครบทั้งหกค่า

เป้าหมายของการแก้ปัญหาการระบุเอกลักษณ์ครั้งนี้คือการหาคำตอบที่เป็นชุดพารามิเตอร์ของแรง f ที่เหมาะสมกับแบบจำลองมากที่สุด แต่ตัววัด (measures) ได้เลือกใช้ค่า SSE ซึ่งมีด้วยกันหกค่า จึงจัดเป็นปัญหาการหาค่าเหมาะที่สุดแบบหลายวัตถุประสงค์ และทุกวัตถุประสงค์ก็ต้องการค่า SSE ต่ำที่สุด จึงดำเนินการสร้างฟังก์ชันวัตถุประสงค์เพื่อให้ MATS ดำเนินการค้นหาด้วยการผนวกค่า SSE ทั้งหกค่าด้วยสมการผลรวมพีชคณิตเชิงเส้นแล้วถ่วงน้ำหนักด้วยค่า α_i , $i = 1, \dots, 6$ ดังสมการที่ (7.27)

$$J = \{\alpha_2 SSE_{2i} + \alpha_3 SSE_{3i} + \alpha_4 SSE_{4i} + \alpha_5 SSE_{5i}\}_{ident} + \{\alpha_1 SSE_{1v} + \alpha_6 SSE_{6v}\}_{valid} \quad (7.27)$$

กำหนดให้ค่าถ่วงน้ำหนักเท่ากันคือ $\alpha_i = 1/6$ $i = 1, \dots, 6$ มีความหมายเดียวกันกับว่าค่า J คือค่าเฉลี่ยค่าของ SSE ทั้งหมดนั่นเอง การได้ค่า J ยิ่งน้อยย่อมแสดงว่าโดยเฉลี่ยแล้ว

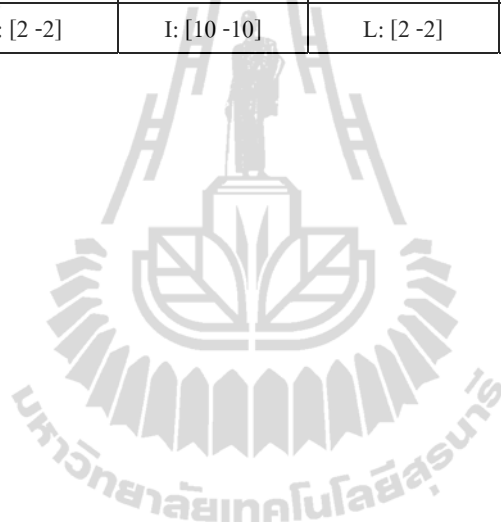
ค่า SSE ทั้งหมดมีค่าน้อย ๆ ด้วย ดังนั้นการมุ่งค้นหาค่า J น้อย ๆ ย่อมทำให้กระบวนการระบุเอกลักษณ์ได้ผลที่ดีตามไปด้วยนั่นเอง เพราะ J เป็นตัวแทนของทั้งส่วนของการระบุเอกลักษณ์แบบจำลองที่มีตัวห้อย “ident” และส่วนของการตรวจสอบความถูกต้องแบบจำลองที่มีตัวห้อย “valid” ด้วยในเวลาเดียวกัน เมื่อได้ฟังก์ชันวัตถุประสงค์ ดังสมการที่ (7.27) แล้ว จึงเป็นการง่ายที่จะให้อยู่ในรูปทั่วไปของปัญหาการหาค่าเหมาะที่สุด แบบปัญหาค่าต่ำสุด หรือ $\min J$

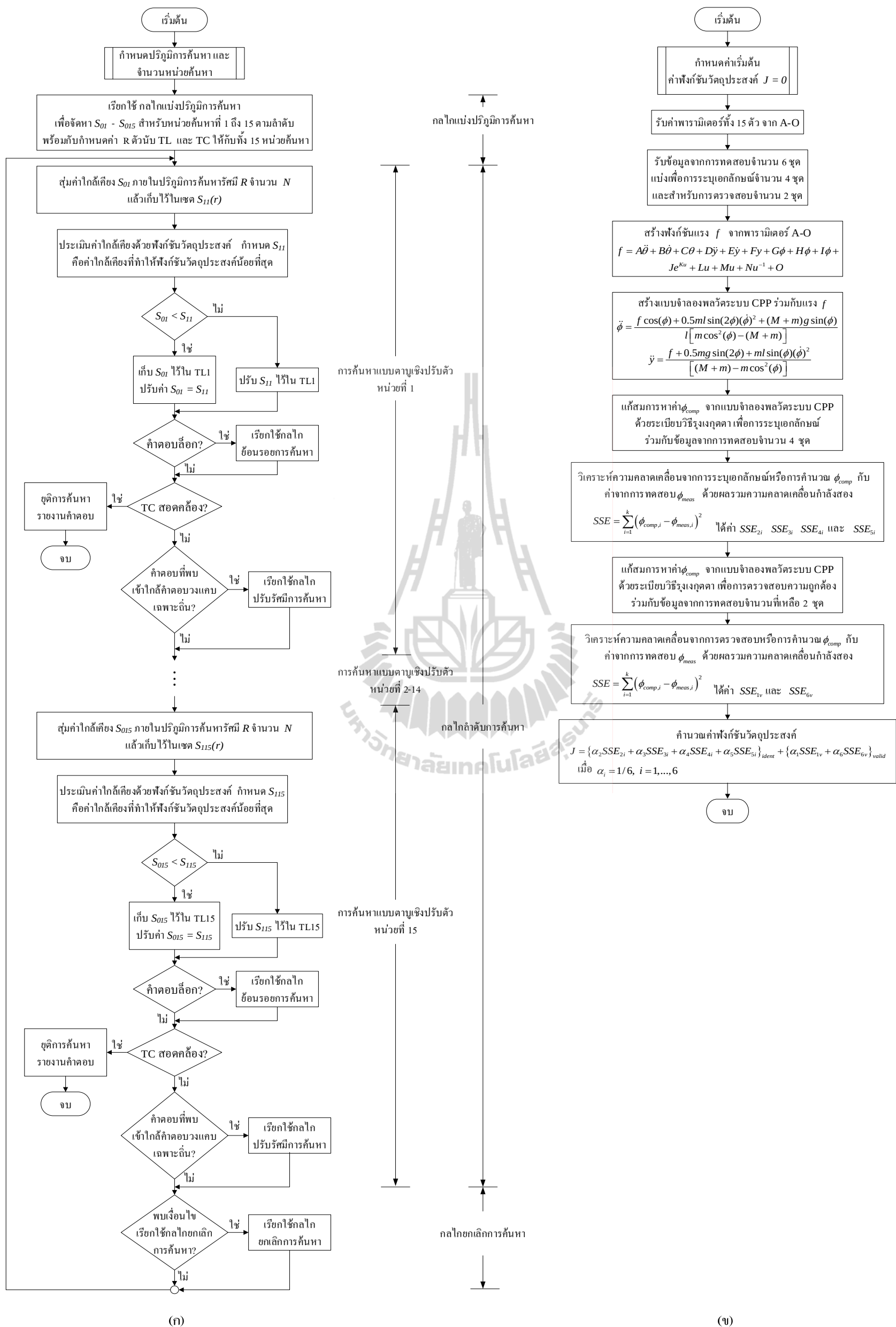
รายละเอียดขั้นตอนการค้นหามิเตอร์ทั้ง 15 ตัวด้วย MATS สามารถสรุปด้วยแผนภูมิการดำเนินงานของผังรูปที่ 7.10 (ก) ส่วนรูปที่ 7.10 (ข) แสดงแผนแผนภูมิการดำเนินงานเพื่อคำนวณของฟังก์ชันวัตถุประสงค์ที่ใช้ประเมินคุณภาพคำตอบของการค้นหา จากการศึกษาเบื้องต้นเพื่อแก้ปัญหาการหาค่าเหมาะที่สุดชนิดนี้ดังรูปสมการที่ (7.27) ด้วย ATS พบว่าข้อมูลที่เหมาะสมสำหรับการตั้งค่าพารามิเตอร์ของ ATS ย่อย เป็นดังนี้ $\text{search_radius} = 0.1$ $\text{Number_neighb} = 5$ กลไกย้อนรอยการค้นหาค่าพารามิเตอร์ดังนี้ $k^{\text{th}}_{\text{backward}} = 5$ $N_{\text{re_max}} = 5$ กลไกปรับรัศมีการค้นหาตั้งพารามิเตอร์ไว้สามช่วงดังต่อไปนี้ AR_1 ถ้า $J < 20$ $\text{search_radius} = 0.01$ AR_2 ถ้า $J < 8$ $\text{search_radius} = 0.001$ และ AR_3 ถ้า $J < 3.5$ $\text{radius} = 0.0001$ เงื่อนไขยุติการค้นหากำหนดไว้ที่ $\text{max_count} = 10,000$ และ $J < 3.1895$

การกำหนดจำนวนหน่วยค้นหาให้กับ MATS สำหรับกรณีนี้ พิจารณาจากขนาดของปัญหาซึ่งพบว่าเป็นปัญหาแบบ 15 มิติ จึงเลือกจำนวนหน่วยค้นหาของ MATS เท่ากับจำนวนมิติของปัญหาหรือ MATS#15 ทั้งนี้เพื่อความสะดวกและให้ MATS มีโอกาสสำรวจครอบคลุมทุกมิติของปัญหา แต่อย่างไรก็ตามผู้อ่านมีอิสระในการเลือกจำนวนหน่วยค้นหาให้มากหรือน้อยกว่านี้ก็ได้ เพียงแต่ต้องคำนึงถึงผลที่ได้ทั้งแง่ของเวลาที่ใช้ในการค้นหาและคุณภาพของคำตอบที่ได้ด้วย เมื่อตัดสินใจเลือกใช้ MATS#15 เพื่อแก้ปัญหานี้แล้วจึงวางแผนการทำงานให้ PM ดังนี้ เนื่องจากมีอินพุต 15 ค่า คือพารามิเตอร์ A ถึง O จึงกำหนดให้ ATS ย่อย แต่ละเส้นทางเริ่มต้นการค้นหาด้วยคำตอบที่สุ่มและคัดเลือกมาจากแต่ละปริภูมีย่อยที่เกิดจากการแบ่งปริภูมิการค้นหาออกเป็น ส่วน ๆ สำหรับปัญหานี้มีจำนวนมิติของปัญหามากกว่าสองจึงอาศัยแนวคิดคำตอบแกน (core solution : S_c) ทำหน้าที่เป็นแกนการแบ่งปริภูมิเพื่อให้ปริภูมีย่อยกระจายทั่วถึงทุกมิติของปริภูมิการค้นหา ซึ่งจากการทดสอบ (pre-search) ได้เลือก S_c และกำหนดปริภูมิการค้นหาทั้งหมด ดังรายละเอียดในตารางที่ 7.1

ตารางที่ 7.1 ค่าตอบแทนและปริมาณการค้นหาของ MATS#15

พารามิเตอร์ 15 ตัว (A-O) ที่จะทำการค้นหา					ค่า Cost SSE (J)
A - C	D - F	G - I	J - L	M - O	
ค่าตอบแทน					-
A: 0.1	D: 0.1	G: 0.1	J: 0.1	M: 0.1	41.4846
B: 0.1	E: 0.1	H: 0.1	K: 0.1	N: 0.1	-
C: 0.1	F: 0.1	I: 0.1	L: 0.1	O: 0.1	-
ปริมาณการค้นหา					-
A: [5 -5]	D: [2 -2]	G: [6 -6]	J: [1 -1]	M: [1 -1]	-
B: [5 -5]	E: [2 -2]	H: [5 -5]	K: [1 -1]	N: [1 -1]	-
C: [5 -1]	F: [2 -2]	I: [10 -10]	L: [2 -2]	O: [5 -5]	-





รูปที่ 7.10 (ก) แผนภูมิการทำงานของการค้นหาแบบตามเชิงปรับตัวหลายวิธีเพื่อการระบุเอกลักษณ์ระบบพลวัตสำหรับระบบรบกวนกเพนดูลัม

(ข) แผนภูมิการทำงานของการหาค่าฟังก์ชันวัตถุประสงค์

เมื่อทราบจำนวนหน่วยค้นหา ปฏิบัติการค้นหารวม และ ค่าตอบแทนแล้ว PM ก็สามารถกำหนดปฏิบัติการค้นหาย่อยเพื่อจัดหาคำตอบเริ่มต้นให้กับแต่ละหน่วยค้นหาย่อยได้แล้ว ตัวอย่างการแบ่งปฏิบัติการค้นหาย่อยสำหรับ MATS#15 พร้อมคำตอบเริ่มต้นที่สุ่มคัดเลือกขึ้นได้จากแต่ละปฏิบัตีย่อย แสดงอยู่ในตารางที่ 7.2

ตารางที่ 7.2 คำตอบเริ่มต้นและค่า SSE ของ MATS ทั้ง 15 หน่วยค้นหาย่อย

หน่วยค้นหา	คำตอบเริ่มต้นที่ได้จาก PM	ค่า SSE (<i>J</i>)
ATS#1	A = -0.3268 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	28.9233
ATS#2	B = 0.3011 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	77.5293
ATS#3	C = 0.4259 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	65.5967
ATS#4	D = -1.2395 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	39.1685
ATS#5	E = -1.8882 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	36.8527
ATS#6	F = 1.9328 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	42.5111
ATS#7	G = 1.0027 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	57.1245
ATS#8	H = -1.7569 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	29.4764
ATS#9	I = -0.8157 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	42.3232
ATS#10	J = -0.1025 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	43.6102
ATS#11	K = 0.5116 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	42.9631
ATS#12	L = -0.0928 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	39.3411
ATS#13	M = -0.3918 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	34.9892
ATS#14	N = 0.0152 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	19.8972
ATS#15	O = -0.0121 ส่วนค่าอื่นอีก 14 ค่า กำหนดไว้เท่ากับ 0.1	42.7764

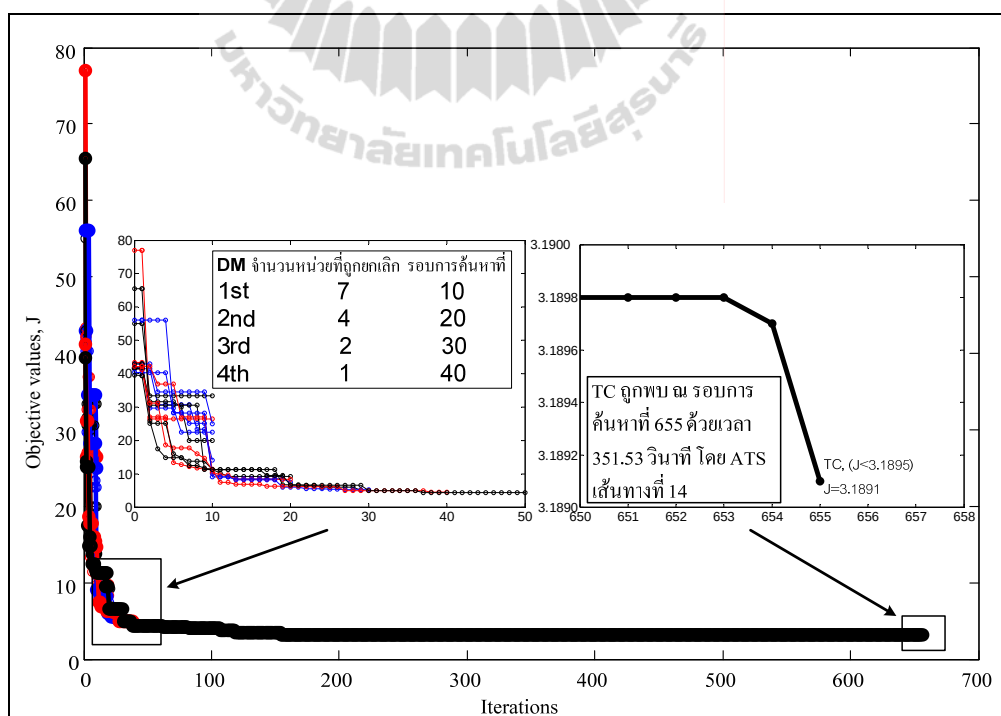
DM ตั้งค่าให้มีการเรียกใช้สี่ครั้งเพื่อลดจำนวนหน่วยค้นหาจาก 15 หน่วยค้นหา จนเหลือหนึ่งหน่วยค้นหา ดังรายละเอียดอยู่ในตารางที่ 7.3 ข้อมูลแถวที่ 1 ในตารางแสดงรอบการค้นหาที่มีการเรียกใช้ DM ต่าง ๆ แถวที่สองแสดงจำนวนหน่วยค้นหาที่ถูกยกเลิกการค้นหาแต่ละครั้งของ DM และแถวที่สามสุดท้ายแสดงจำนวนหน่วยค้นหาที่เหลือภายหลังการเรียกใช้ DM นั้น ๆ

ตารางที่ 7.3 การตั้งค่าพารามิเตอร์ DM

พารามิเตอร์	DM th			
	1 st	2 nd	3 rd	4 th
รอบการค้นหาที่มีการเรียกใช้ DM	10	20	30	40
จำนวนหน่วยค้นหาที่ถูกยกเลิกการค้นหา	7	4	2	1
จำนวนหน่วยค้นหาที่ยังคงดำเนินการค้นหา	8	4	2	1

7.7 ผลการค้นหาด้วย MATS แบบ 15 เส้นทางค้นหา

เมื่อกำหนดค่าพารามิเตอร์ของทั้ง ATS และ MATS จนครบ จึงดำเนินการค้นหาคำตอบวงกว้างตามที่ต้องการ ได้ผลการค้นหาของ MATS#15 แสดงอยู่ในรูปกราฟการลู่เข้าหาคำตอบวงกว้างแสดงดังรูปที่ 7.11 โดยมี ATS ย่อยที่ 14 พบคำตอบภายใต้เงื่อนไขชุดการค้นหาที่มีค่า $J < 3.1895$ ในรอบการค้นหาที่ 655 ด้วยเวลาการค้นหา 351.53 วินาที คำตอบวงกว้างที่ได้นั้น แสดงรายละเอียดอยู่ในตารางที่ 7.4 แล้ว



รูป 7.11 กราฟการลู่เข้าหาผลเฉลยของ MATS กับปัญหาการระบุเอกลักษณ์ระบบพลวัต CPP

ตารางที่ 7.4 ค่าตอบที่ได้จากการค้นหาด้วย MATS#15

พารามิเตอร์ทั้ง 15 ตัว (A - O) ที่ได้จากการค้นหา					ค่า Cost (SSE)		
A - C	D - F	G - I	J - L	M - O	<i>J</i>	<i>indent</i>	<i>valid</i>
A: -0.442	D: -1.322	G: -3.793	J: 0.368	M: -0.099	3.1981	1.9564 ⁽²⁾	1.3529 ⁽¹⁾
B: 0.540	E: 0.638	H: -0.630	K: -0.277	N: -0.007	-	2.2980 ⁽³⁾	4.3520 ⁽⁶⁾
C: 0.046	F: 0.214	I: -7.663	L: 0.144	O: -0.391	-	3.1300 ⁽⁴⁾	-
-	-	-	-	-	-	6.0454 ⁽⁵⁾	-

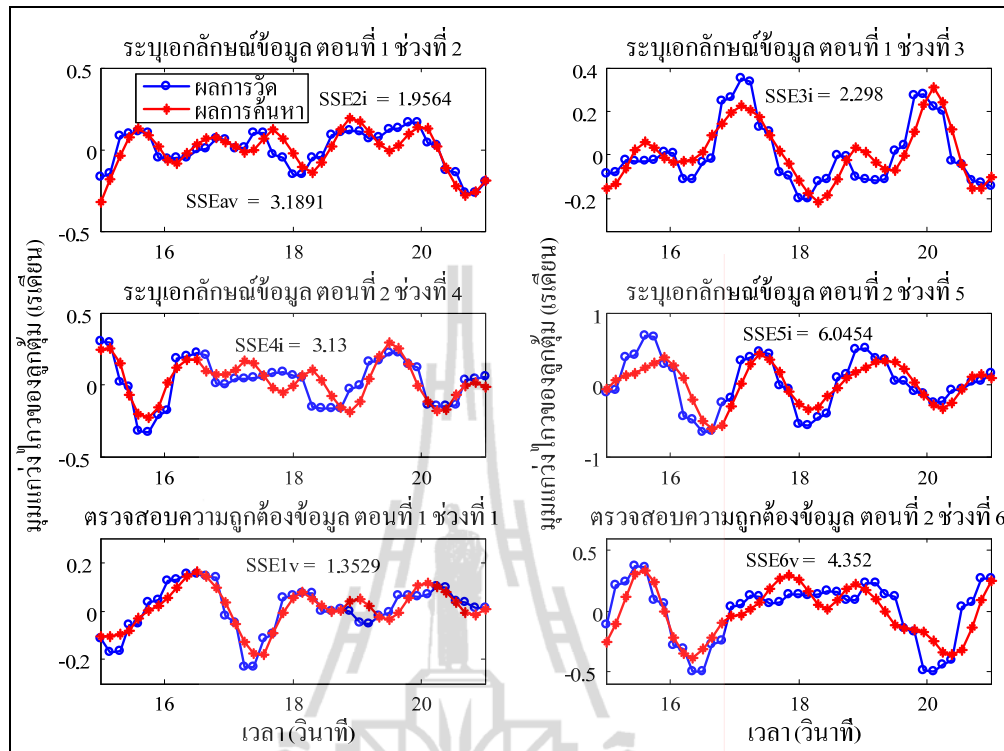
หมายเหตุ : (1) ถึง (6) หมายถึงค่าจากการวิเคราะห์ข้อมูลทดสอบชุด 1 ถึง 6 ตามลำดับ

ข้อมูลจากตารางที่ 7.4 ตั้งแต่หลักที่ 1-5 แสดงค่าพารามิเตอร์ทั้ง 15 ตัว ของค่าตอบที่ได้จากการค้นหา ส่วนหลักที่ 6 - 8 เป็นค่า SSE ของฟังก์ชันวัตถุประสงค์ *J* การระบุเอกลักษณ์ (*indent* ในตาราง) และ การตรวจสอบความถูกต้อง (*valid* ในตาราง) ตามลำดับ เมื่อพารามิเตอร์ทั้ง 15 ตัว ถูกแทนลงในสมการที่ (7.25) จะได้สมการที่ (7.28)

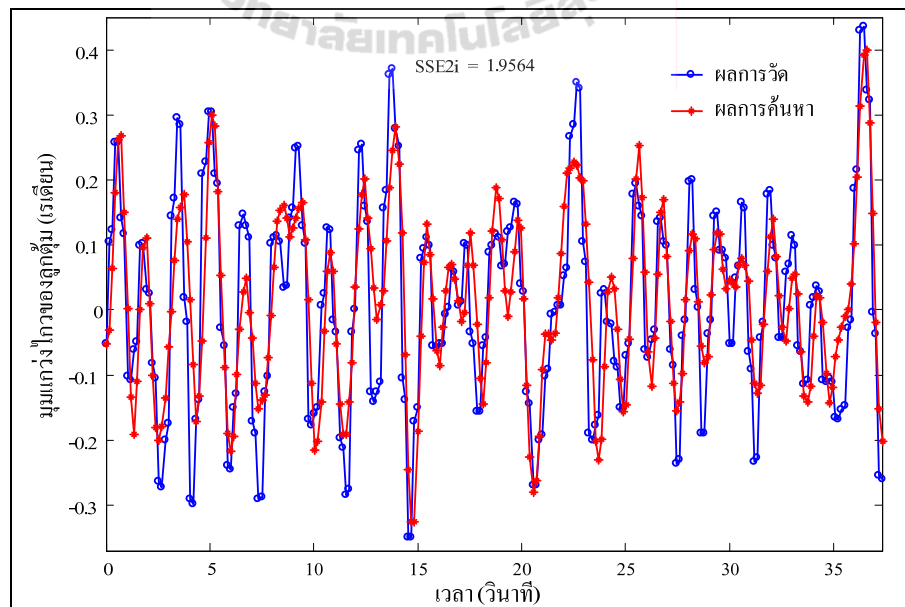
$$\begin{aligned}
 f(\theta, y, \phi, u) = & -0.4425\ddot{\theta} + 0.5408\dot{\theta} + 0.0466\theta \\
 & -1.3226\ddot{y} + 0.6382\dot{y} + 0.2141y \\
 & -3.7936\ddot{\phi} - 0.6304\dot{\phi} - 7.6637\phi \\
 & + 0.3684e^{-0.2778u} + 0.1446u^2 \\
 & -0.0996u - 0.0069u^{-1} - 0.3913
 \end{aligned} \tag{7.28}$$

เมื่อนำสมการที่ (7.28) แทนลงในสมการที่ (7.23) และ (7.24) แล้วคำนวณหามุมแกว่งของเพนดูลัม เมื่อนำผลที่ได้จากการคำนวณนี้แสดงเป็นกราฟเปรียบเทียบกับผลที่ได้จากการทดสอบปรากฏ ดังรูปที่ 7.12 โดยเส้น -o- ได้จากการวัดและเส้น -*-* ได้จากการคำนวณหรือการค้นหาตัวเอง

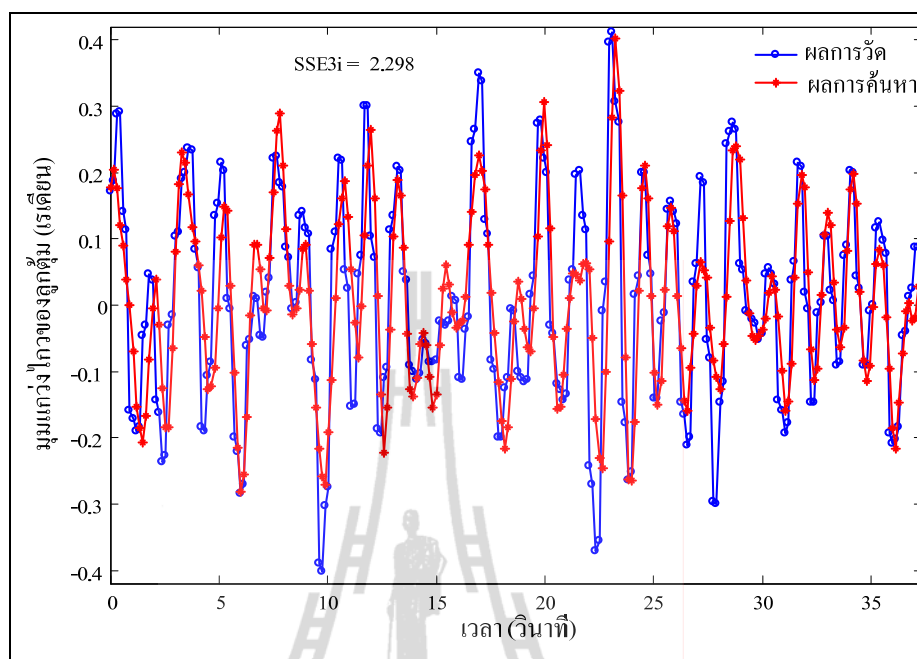
ภาพสัญญาณเอาต์พุตที่แทนตำแหน่งเชิงมุมของลูกตุ้ม ซึ่งได้จากการระบุเอกลักษณ์ แสดงไว้ในรูปที่ 7.13 ถึง รูปที่ 7.16 ตามลำดับ อาจสังเกตเห็นได้ว่าผลการคำนวณสอดคล้องกับผลการวัดทั้งทางขนาดและเฟสเป็นอย่างมาก



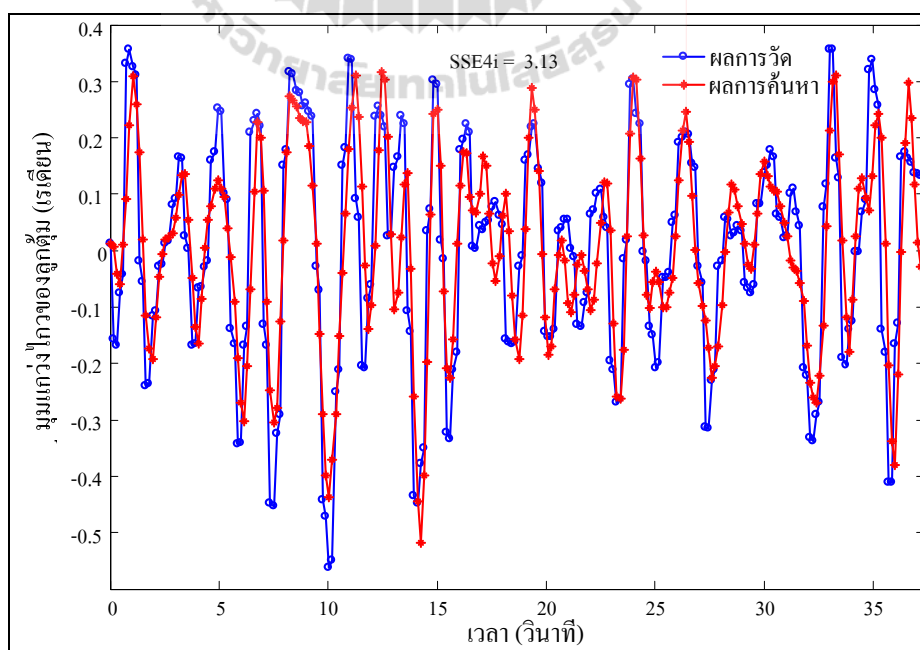
รูป 7.12 ผลการระบุเอกลักษณ์ ที่ค่า $SSE = 3.1891$



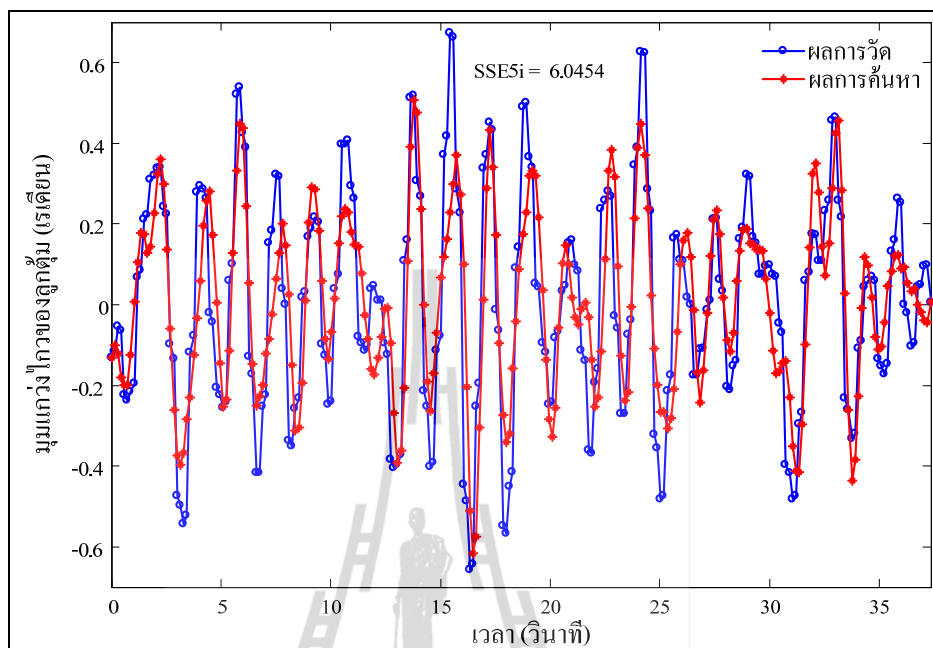
รูป 7.13 ตำแหน่งเชิงมุมของลูกตุ้มนำพิกะจากการวัดและการระบุเอกลักษณ์ด้วยข้อมูลชุดที่ 2 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที



รูป 7.14 ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและการระบุเอกลักษณ์ด้วยข้อมูลชุดที่ 3
จำนวน 250 ชุด เป็นเวลา 37.5 วินาที



รูป 7.15 ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาจากการวัดและการระบุเอกลักษณ์ด้วยข้อมูลชุดที่ 4
จำนวน 250 ชุด เป็นเวลา 37.5 วินาที



รูป 7.16 ตำแหน่งเชิงมุมของลูกตุ้มนำพิกจากการวัดและการระบุเอกลักษณ์ด้วยข้อมูลชุดที่ 5 จำนวน 250 ชุด เป็นเวลา 37.5 วินาที

การตรวจสอบความถูกต้องของแบบจำลอง ดำเนินการกับข้อมูลทดสอบช่วงข้อมูลที่ 1 กับ ช่วงข้อมูลที่ 6 โดยข้อมูลทดสอบแต่ละช่วงประกอบด้วยข้อมูลจำนวน 250 ชุด แต่ละชุดห่างกัน 0.15 วินาที ความคลาดเคลื่อน (errors) ในที่นี้กำหนดให้เป็นความต่างระหว่างเอาต์พุตที่ได้จากการวัดกับเอาต์พุตที่ได้จากการคำนวณโดยอาศัยแบบจำลอง ดังสมการที่ (7.23) ส่วนการวิเคราะห์ความคลาดเคลื่อนจะพิจารณาด้วย ค่าความคลาดเคลื่อนสัมบูรณ์ (absolute errors) ผลรวมของค่าความคลาดเคลื่อนสัมบูรณ์ (sum of absolute errors : SAE) ค่าเฉลี่ยของค่าความคลาดเคลื่อนสัมบูรณ์ (means of absolute errors : MAE) และ ค่าเบี่ยงเบนมาตรฐานของค่าความคลาดเคลื่อนสัมบูรณ์ (standard deviations of absolute errors : STD) แทนด้วยความสัมพันธ์ ดังสมการที่ (7.29) ถึง (7.33) ตามลำดับ

$$errors = \phi_{meas} - \phi_{comp} \quad (7.29)$$

$$absolute \ errors = e = \|\phi_{meas} - \phi_{comp}\| \quad (7.30)$$

$$SAE = \sum_{i=1}^{N=250} \|\phi_{meas,i} - \phi_{comp,i}\| \quad (7.31)$$

$$MAE = \bar{e} = \frac{1}{N} \sum_{i=1}^{N=250} \|\phi_{meas,i} - \phi_{comp,i}\| \quad (7.32)$$

$$STD = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N=250} (e_i - \bar{e})^2} \quad (7.33)$$

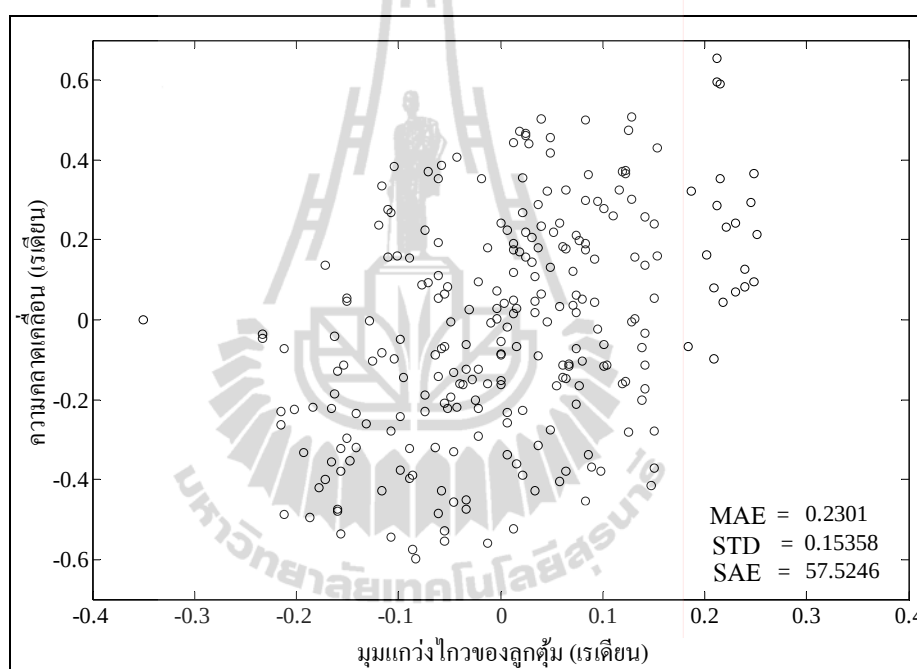
งานวิจัยก่อนหน้านี้ (เดชา พวงดาวเรือง, 2547) ได้แทนแรง f ด้วยฟังก์ชันพหุนาม ดังสมการที่ (7.34)

$$f(u) = a_7 u^7 + a_6 u^6 + a_5 u^5 + a_4 u^4 + a_3 u^3 + a_2 u^2 + a_1 u + a_0 \quad (7.34)$$

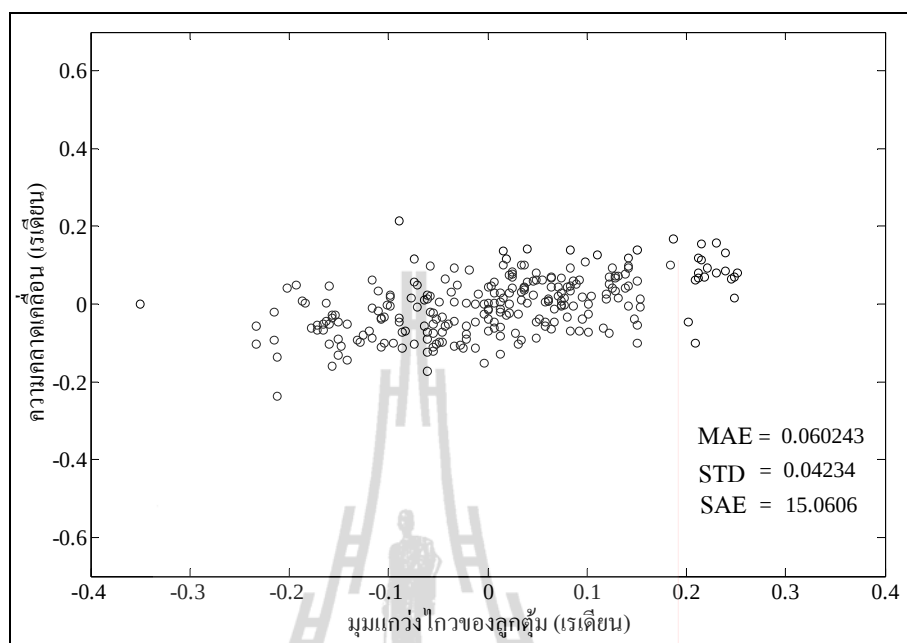
เมื่อ u คืออินพุตระบบ CPP และ สัมประสิทธิ์ที่ได้จากการค้นหาด้วย ATS แบบเดียว คือ $a_0 = -0.1482$ $a_1 = 2.4923$ $a_2 = -2.3259$ $a_3 = -4.9867$ $a_4 = -5.6048$ $a_5 = -1.8275$ $a_6 = 6.9857$ และ $a_7 = 4.0937$ เมื่อนำสมการที่ (7.34) แทนลงใน (7.23) และ (7.24) แล้วคำนวณหาค่า ตำแหน่งเชิงมุมของลูกตุ้มด้วยระเบียบวิธีรุงเงกุตตาซึ่งได้กล่าวไว้แล้วตอนต้น เมื่อนำมาประเมิน ความถูกต้องของแบบจำลอง โดยพิจารณาค่าความคลาดเคลื่อนของช่วงข้อมูลที่ 1 แสดงดังรูปที่ 7.17 ด้วย ค่า MAE เท่ากับ 0.2301 ค่า STD เท่ากับ 0.15358 และ ค่า SAE เท่ากับ 57.5246 เมื่อเทียบกับ ค่าความคลาดเคลื่อนของช่วงข้อมูลที่ 1 ของแนวทางที่นำเสนอซึ่งแทนแรง f ด้วยฟังก์ชันดัง สมการที่ (7.25) ให้ค่า MAE เท่ากับ 0.06024 ค่า STD เท่ากับ 0.04234 และค่า SAE เท่ากับ 15.0606 แสดงดังรูปที่ 7.18 พบว่าฟังก์ชันแรง f ที่เสนอใหม่นี้ช่วยให้แบบจำลองระบบ CPP ในย่านการ แกว่งของลูกตุ้ม ± 0.4 เรเดียน มีค่าความคลาดเคลื่อนน้อยลงหรือมีความแม่นยำมากขึ้นนั่นเอง

เมื่อเปลี่ยนเป็นช่วงข้อมูลที่ 6 ซึ่งมีย่านการแกว่งของลูกตุ้มมากขึ้นเป็น ± 0.6 เรเดียน แบบจำลองที่ใช้ฟังก์ชันแรง f เดิม แสดงการกระจายค่าความคลาดเคลื่อนดังรูปที่ 7.19 ด้วย ค่า MAE เท่ากับ 0.65441 ค่า STD เท่ากับ 0.40365 และค่า SAE เท่ากับ 160.6033 ส่วนแบบจำลองที่ เสนอใช้ฟังก์ชันแรง f ใหม่ แสดงค่าความคลาดเคลื่อนดังรูปที่ 7.20 ให้ค่า MAE เท่ากับ 0.10459 ค่า STD เท่ากับ 0.0807 และค่า SAE เท่ากับ 26.1487 พบว่าผลที่ได้คล้ายกับของช่วงข้อมูลที่ 6 นั่นคือแบบจำลองใหม่ให้ค่าความคลาดเคลื่อนลดลงจากของแบบจำลองเดิมอย่างมาก

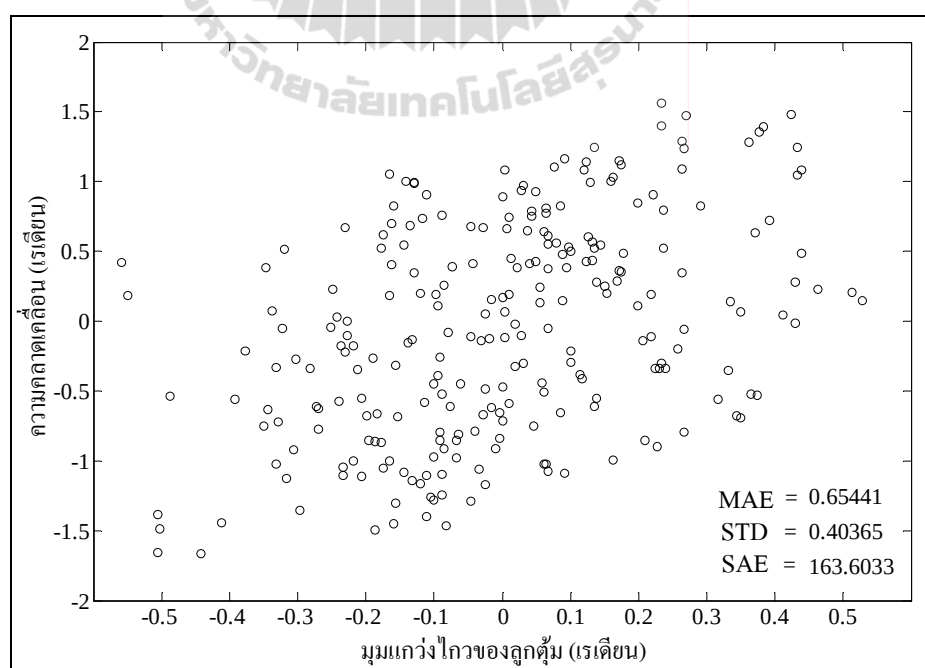
จากผลการตรวจสอบความถูกต้องของแบบจำลอง CPP ใหม่ด้วยการวิเคราะห์ค่าความคลาดเคลื่อนทั้งย่านการแกว่งของลูกตุ้มแคบ ๆ (ช่วงข้อมูลที่ 1) และย่านการแกว่งของลูกตุ้มกว้าง (ช่วงข้อมูลที่ 6) โดยเปรียบเทียบกับแบบจำลองของงานวิจัยก่อนหน้านี้หรือแบบจำลองเดิม พบว่าค่าความคลาดเคลื่อนของแบบจำลองใหม่น้อยกว่าของแบบจำลองเดิม ดังนั้นสามารถกล่าวได้ว่าแบบจำลองระบบ CPP ใหม่สำหรับย่านไม่เชิงเส้นให้ความแม่นยำที่สูงขึ้นกว่าแบบจำลองเดิมอย่างน่าพอใจ



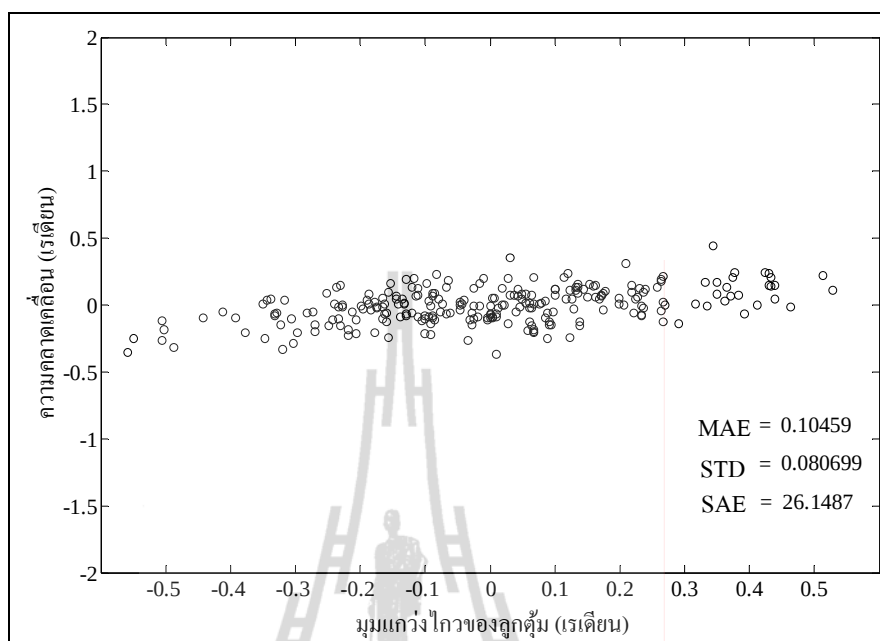
รูป 7.17 ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากข้อมูลช่วงที่ 1 ของฟังก์ชันแรงเดิม (การแกว่งของลูกตุ้มในย่าน ± 0.4 เรเดียน)



รูป 7.18 ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากข้อมูลช่วงที่ 1 ของฟังก์ชันแรงใหม่
(การแกว่งของลูกตุ้มในย่าน ± 0.4 เรเดียน)

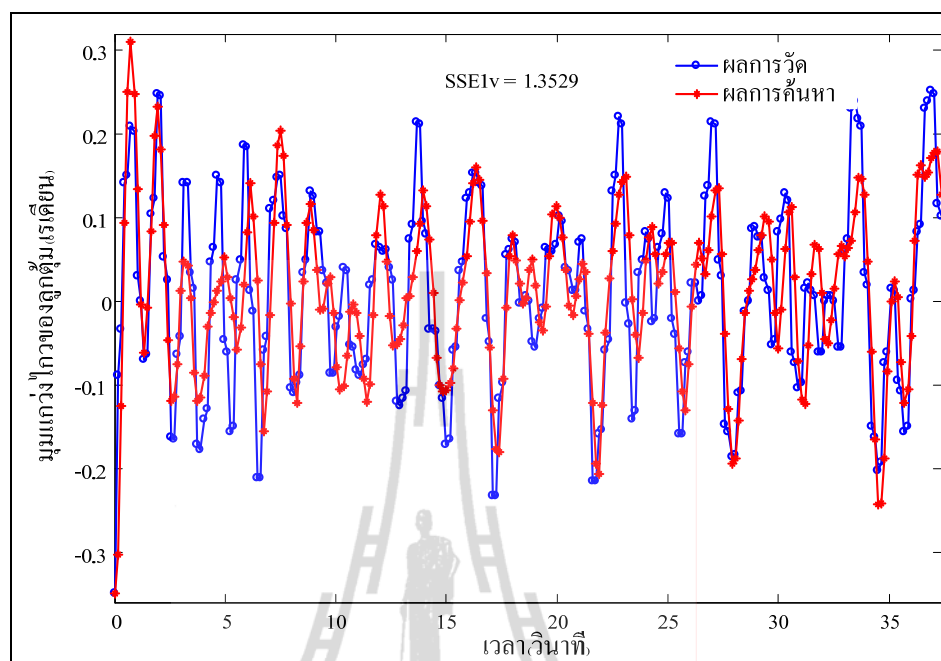


รูป 7.19 ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากข้อมูลช่วงที่ 6 ของฟังก์ชันแรงเดิม
(การแกว่งของลูกตุ้มในย่าน ± 0.6 เรเดียน)

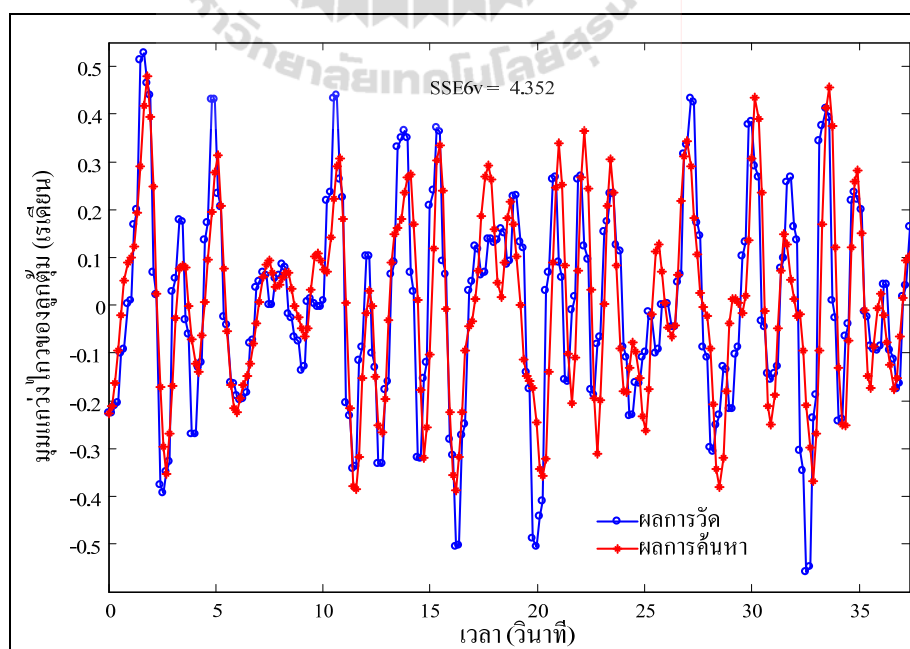


รูป 7.20 ความคลาดเคลื่อนของตำแหน่งเชิงมุมของลูกตุ้มจากข้อมูลช่วงที่ 1 ของฟังก์ชันแรงใหม่
(การแกว่งของลูกตุ้มในย่าน ± 0.6 เรเดียน)

ภาพสัญญาณเอาต์พุตที่แทนตำแหน่งเชิงมุมของลูกตุ้ม ที่ได้จากการตรวจสอบความถูกต้องของแบบจำลอง ถูกแสดงไว้ในรูปที่ 7.21 และ 7.22 ตามลำดับ



รูป 7.21 ตำแหน่งเชิงมุมของลูกตุ้มนำพิกะจากการวัดและจากการตรวจสอบความถูกต้องของแบบจำลองด้วยข้อมูลตอนที่ 1 ช่วงที่ 1 จำนวน 250 จุด คิดเป็นเวลา 37.5 วินาที



รูป 7.22 ตำแหน่งเชิงมุมของลูกตุ้มนำพิกะจากการวัดและจากการตรวจสอบความถูกต้องของแบบจำลองด้วยข้อมูลตอนที่ 2 ช่วงที่ 6 จำนวน 250 จุด คิดเป็นเวลา 37.5 วินาที

7.8 สรุป

บทที่ 7 นี้ได้กล่าวถึงการระบุเอกลักษณ์ระบบพลวัตของระบบลูกตุ้มนาฬิกาพวงกรงให้มีความแม่นยำมากยิ่งขึ้น ด้วยการประมาณแรง f ด้วยฟังก์ชันไม่เป็นเชิงเส้น ซึ่งประกอบขึ้นจากพจน์ตำแหน่งเชิงมุมของลูกตุ้มนาฬิกา พจน์ตำแหน่งเชิงมุมของพูลเลย์ พจน์ตำแหน่งของรถ พจน์สัญญาณด้านเข้า และพจน์อนุพันธ์ ต่าง ๆ แล้วถ่วงน้ำหนักพจน์ต่าง ๆ เหล่านั้นด้วยพารามิเตอร์จำนวน 15 ตัว การตรวจสอบความถูกต้องของแบบจำลองพิจารณาเอาต์พุตที่เป็นค่าตำแหน่งเชิงมุมของลูกตุ้มนาฬิกาโดยเปรียบเทียบจากค่าการวัดจริงและค่าจากการคำนวณแบบจำลอง แล้วสังเคราะห์ค่าผลรวมกำลังสองความคลาดเคลื่อน หรือ SSE ทำหน้าที่เป็นตัวชี้วัดความเหมาะสมของแบบจำลองที่มีอินพุตเป็นพารามิเตอร์ทั้ง 15 ตัวของฟังก์ชัน f นั่นเอง การคัดสรรค่าพารามิเตอร์ทั้ง 15 ตัวให้ได้ค่า SSE น้อยที่สุด ดำเนินการค้นหาด้วย MATS ชนิด 15 หน่วยค้นหาภายใต้เงื่อนไขยุติการค้นหาเมื่อ $J < 3.1895$ จนกระทั่งรอบการค้นหาที่ 655 หน่วยค้นหาย่อยที่ 14 หรือ ATS#14 ได้พบคำตอบ ด้วยค่า $J = 3.1891$ (SSE) โดยใช้เวลาการค้นหาทั้งสิ้น 351.53 วินาที เมื่อนำคำตอบที่ได้มาตรวจสอบความถูกต้องของแบบจำลองด้วยการวิเคราะห์ค่าความคลาดเคลื่อนด้วยค่าผลรวมของค่าความคลาดเคลื่อน (SAE) สำหรับช่วงข้อมูลที่หนึ่งแบบจำลองใหม่ให้ค่า $SAE = 15.0606$ ซึ่งน้อยกว่าของแบบจำลองเดิมที่มี $SAE = 57.5246$ ส่วนช่วงข้อมูลที่หกแบบจำลองใหม่ให้ค่า $SAE = 26.1487$ ซึ่งน้อยกว่าของแบบจำลองเดิมที่มี $SAE = 163.6033$ อีกทั้งข้อมูลค่าคลาดเคลื่อนจากการใช้แบบจำลองแรง f ที่นำเสนอ มีการกระจายตัวต่ำหมายถึงข้อมูลดังกล่าวบีบตัวแน่นโดยรอบค่าศูนย์ และมีแนวโน้มของข้อมูลต่ำใกล้เคียงศูนย์ ซึ่งแสดงให้เห็นว่าคุณภาพของแบบจำลองระบบพลวัตของระบบลูกตุ้มนาฬิกาพวงกรงสำหรับย่านไม่เชิงเส้นมีความแม่นยำสูงขึ้นอย่างน่าพอใจจากการใช้แรงไม่เป็นเชิงเส้น f ตามที่นำเสนอ

บทที่ 8

บทสรุปและข้อเสนอแนะ

8.1 ข้อสรุป

งานวิจัยวิทยานิพนธ์นี้เกี่ยวข้องกับการพัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัว เพื่อให้มีประสิทธิภาพการค้นหาสูงขึ้น โดยอาศัยเทคนิคการกระจายหน่วยค้นหาพร้อมกับกระบวนการคัดสรรหน่วยค้นหาที่มีโอกาสพบคำตอบกว้างสูงกว่าเท่านั้นที่จะดำเนินการค้นหาต่อไปได้ เกิดเป็นอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี การดำเนินงานวิจัยวิทยานิพนธ์นี้สำเร็จ ลุล่วงตามวัตถุประสงค์โดยสามารถสรุปผลการศึกษาวิจัยและพัฒนาทางวิศวกรรมเป็นข้อสรุปได้ดังต่อไปนี้

1. การปรัศน์วรรณกรรมได้รับการรายงานไว้ในบทที่ 2 ซึ่งได้พบว่าแนวทางการพัฒนาอัลกอริธึมการค้นหาแบบตามู อาจแบ่งได้เป็นหกแนวทางได้แก่ การค้นหาแบบตามูเชิงปรับตัว การค้นหาแบบตามูเชิงปฏิกริยา การค้นหาแบบตามูเชิงขนาน การค้นหาแบบตามูเชิงดัดแปลง การค้นหาแบบตามูเชิงความน่าจะเป็นและการค้นหาแบบตามูเชิงผสมผสาน และการค้นหาแบบตามูเชิงปรับตัวฉบับมหาวิทยาลัยเทคโนโลยีสุรนารีเป็นแบบปรับตัวเพื่อหนีการติดลือกคำตอบวงแคบเฉพาะถิ่นและปรับตัวเพื่อเร่งความเร็วการค้นหาเข้าสู่คำตอบกว้าง ส่วนงานวิจัยวิทยานิพนธ์ฉบับนี้เป็นการต่อยอดพัฒนาอัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวฉบับดังกล่าว โดยเสนอโครงสร้างการจัดการหน่วยค้นหา เพื่อให้มีประสิทธิภาพการค้นหารวมที่ดีขึ้น

2. อัลกอริธึมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี หรือ multi-path adaptive tabu search algorithm (MATS) ที่พัฒนาขึ้นนี้อาศัยสองกลยุทธ์สำคัญของอัลกอริธึมการค้นหาแบบตามู ได้แก่ กลยุทธ์ความหลากหลาย (diversification strategy) และกลยุทธ์ความเข้มข้น (intensification strategy) โดยประกอบด้วยสามกลไกหลัก ได้แก่ กลไกที่หนึ่ง คือกลไกแบ่งปริภูมิการค้นหา (partitioning mechanism : PM) ทำหน้าที่แบ่งปริภูมิการค้นหาออกเป็นปริภูมิการค้นหาย่อยเพื่อจัดสรรคำตอบเริ่มต้นให้แก่หน่วยค้นหาของ MATS กลไกที่สอง คือกลไกลำดับการค้นหา (sequencing mechanism : SM) ทำหน้าที่บังคับให้แต่ละหน่วยค้นหาทำงานเชิงลำดับในแต่ละวงรอบการค้นหาของ MATS และกลไกที่สาม คือกลไกยกเลิกการค้นหา (discarding mechanism : DM) ทำหน้าที่ยกเลิกการค้นหาสำหรับหน่วยค้นหาที่ถูกการประเมินว่ามีโอกาสพบคำตอบกว้างได้น้อยกว่าหน่วยค้นหาอื่น ๆ กลไกนี้มีส่วนอย่างมากที่ทำให้เวลาแต่ละรอบการค้นหาของ MATS สั้นลงเนื่องจากทำให้จำนวนหน่วยค้นหาลดลงตามลำดับจากการถูกยกเลิกการค้นหานั่นเอง

3. คุณสมบัติการรู้เข้าผลเฉลยของ MATS ได้รับการพิสูจน์ว่ายังคงมีเช่นเดียวกับของ ATS แบบเดี่ยว เนื่องจากทั้งสามกลไกหลักคือ PM SM และ DM ภายใน MATS มิได้ทำให้ ATS ย่อยในกำกับของ MATS สูญเสียคุณสมบัติการรู้เข้าหาผลเฉลยนี้แต่อย่างใดนั้นหมายความว่า หาก ATS ย่อย ทุกตัวในกำกับของ MATS มีคุณสมบัติการรู้เข้าหาผลเฉลยแล้ว หนึ่งใน ATS ย่อย เหล่านั้นจะต้องเป็นค้นพบคำตอบวงกว้าง เท่ากับว่า MATS จึงมีคุณสมบัติการรู้เข้าผลเฉลยนี้ด้วยเช่นกัน รายละเอียดบทพิสูจน์การรู้เข้าหาผลเฉลยของ MATS ได้รับการกล่าวถึงแล้วในบทที่ 4

4. MATS ถูกพัฒนาขึ้นเป็นโปรแกรม MATLABTM และทดสอบการทำงานบนเครื่องคอมพิวเตอร์ Intel Celeron(R) ความเร็ว 2.6 จิกกะเฮิร์ตซ์ ด้วยหน่วยความจำ RAM ขนาด 238 เมกะไบต์ และหน่วยความจำ HDD ขนาด 30 จิกกะไบต์ เพื่อประเมินสมรรถนะการค้นหาคำตอบ ปัญหาการหาค่าเหมาะที่สุดเชิงพื้นที่จำนวน 6 ฟังก์ชัน ได้แก่ (1) ฟังก์ชันโบฮาเซฟสกี (BF) (2) ฟังก์ชันราสทริกิน (RF) (3) ฟังก์ชันชีเคิลฟอกซ์โฮล (SF) (4) ฟังก์ชันชูเบิร์ต (ShF) (5) ฟังก์ชันชีเฟลแบบที่หนึ่ง (ScF#I) และ (6) ฟังก์ชันชีเฟลแบบที่สอง (ScF#II) ผลการทดสอบ พบว่า MATS สามารถใช้เวลาการค้นหาคำตอบวงกว้างได้น้อยกว่า ATS แบบเดี่ยว ได้ถึง 5 ฟังก์ชัน ได้แก่ฟังก์ชัน BF RF SFSHF และ ScF#I ส่วน ScF#II ซึ่งมีลักษณะพื้นผิวเรียบที่ง่ายต่อการค้นหาคำตอบวงกว้าง จึงทำให้ ATS แบบเดี่ยว ใช้เวลาการค้นหาน้อยกว่า MATS จึงกล่าวได้ว่าการแก้ปัญหาการหาค่าเหมาะที่สุดบางชนิดที่มีความซับซ้อนสูงเพียงพอ MATS สามารถดำเนินงานค้นหาคำตอบวงกว้างได้รวดเร็วกว่า ATS แบบเดี่ยว และเป็นทางเลือกใหม่ให้กับการแก้ปัญหาดังกล่าวได้อย่างน่าพอใจ ทั้งนี้มีรายละเอียดปรากฏในบทที่ 5

5. ระบบจำลองรถยนต์แห่งอิลลินอยส์ หรือระบบ IRS เป็นระบบพลวัตที่ได้รับการประยุกต์เพื่อการออกแบบตัวควบคุมด้วย MATS ในงานวิจัยวิทยานิพนธ์นี้ ระบบดังกล่าวได้รับการทบทวนในเรื่องแบบจำลองระบบและฟังก์ชันถ่ายโอนวงรอบปิดของระบบพร้อมตัวควบคุมดั้งเดิม ตลอดจนเอาต์พุตอ้างอิงจากการกระตุ้นด้วยสัญญาณขั้นบันไดหนึ่งหน่วย ดังรายละเอียดที่ปรากฏในบทที่ 6 การออกแบบตัวควบคุมแบ่งเป็น 3 แนวทาง ได้แก่ แนวทางที่หนึ่ง การคาดเดาตัวควบคุม PD กับบล็อกตัวควบคุมป้อนไปหน้า แนวทางที่สอง การคัดสรรพารามิเตอร์ใหม่สำหรับตัวควบคุมดั้งเดิม และแนวทางที่สาม การคัดสรรพารามิเตอร์ใหม่สำหรับตัวควบคุมดั้งเดิมกับตัวควบคุม PD ผลตอบสนองของระบบตามแนวทางที่หนึ่งสามารถลดเวลาประวิงได้เป็นอย่างดีน่าพอใจ ผลตอบสนองของระบบตามแนวทางที่สองสามารถลดเวลาเข้าที่ไต่ระดับหนึ่งแต่ไม่สามารถลดเวลาประวิงได้ ส่วนแนวทางที่สามทั้งสามสามารถลดเวลาเข้าที่และกำจัดเวลาประวิงได้ด้วย จึงให้ผลตอบสนองของระบบได้ดีที่สุดเมื่อเปรียบเทียบทั้งสามแนวทาง

6. ระบบลูกตุ้มนาฬิกาพวงกรร หรือระบบ CPP เป็นระบบพลวัตที่ใช้เพื่อการระบุเอกลักษณ์ด้วย MATS ในงานวิจัยวิทยานิพนธ์นี้ บทที่ 7 ได้ทบทวนวิธีการหาแบบจำลองระบบ

หลักการทั่วไปในการระบุเอกลักษณ์ระบบตลอดจนงานวิจัยก่อนหน้านี้ที่ได้ดำเนินการระบุเอกลักษณ์ระบบ CPP แบบไม่เชิงเส้น ด้วยการแทนแรง f ที่กระทำต่อรถด้วยฟังก์ชันพหุนามอันดับเจ็ดที่เป็นฟังก์ชันของสัญญาณอินพุต u แต่ยังไม่สามารถทำให้แบบจำลองมีความแม่นยำตรงตามที่ต้องการได้งานวิจัยวิทยานิพนธ์ฉบับนี้จึงได้เสนอแนวทางอธิบายพลวัตของแรงที่ส่งผ่านสายพานไปยังตัวรถด้วยฟังก์ชัน f เพื่อให้แบบจำลองมีความเที่ยงตรงมากขึ้น โดยฟังก์ชัน f นี้สร้างด้วยเทคนิคผลรวมเชิงเส้นของพจน์ ν θ $\dot{\theta}$ $\ddot{\theta}$ ϕ $\dot{\phi}$ $\ddot{\phi}$ ψ $\dot{\psi}$ และ $\ddot{y}$ จำนวน 15 พจน์ เมื่อนำไปแทนในแบบจำลองระบบ CPP แบบไม่เชิงเส้น แล้วระบุเอกลักษณ์ พบว่าแบบจำลองมีความคลาดเคลื่อนน้อยลงเป็นที่น่าพอใจ เมื่อเปรียบเทียบกับวิธีการดั้งเดิม

8.2 ข้อเสนอแนะ

จากการดำเนินงานวิจัยวิทยานิพนธ์ที่ผ่านมาและผลที่ได้ ทำให้เกิดแนวคิดและข้อเสนอแนะในการดำเนินงานวิจัยต่อไปในอนาคต ดังต่อไปนี้

1. แนวคิดส่วนจัดการหน่วยค้นหาแทน (management agent) สามารถนำไปประยุกต์ใช้ได้กับหน่วยค้นหาแทนประเภทอื่นเช่น จีเนติกอัลกอริทึม (genetic algorithm : GA) เทคนิคการหาค่าเหมาะที่สุดแบบฝูงอนุภาค (particle swarm optimization : PSO) และ เทคนิคการหาค่าเหมาะที่สุดแบบอาณานิคมมด (ant colony optimization : ACO) เป็นต้น เพื่อเพิ่มสมรรถนะการค้นหาให้สูงขึ้น

2. การปรับปรุงพัฒนาอัลกอริทึม MATS มีแนวทางต่าง ๆ ดังต่อไปนี้

- กลไกแบ่งปริภูมิการค้นหา หรือ PM ปัจจุบันใช้การแบ่งด้วยประสิทธิภาพและความชำนาญของผู้ใช้ ซึ่งอาจมีขีดจำกัดทำให้การเข้าถึงพื้นที่เป้าหมายทำได้ไม่ดีเท่าที่ควร ดังนั้น หากเทคนิคการแบ่งปริภูมิการค้นหาของ PM ได้รับการพัฒนาให้ดีขึ้นและปรับตัวเข้ากับธรรมชาติของปัญหาได้หลากหลายขึ้น จะเป็นการเพิ่มสมรรถนะของ MATS ได้

- การเรียกใช้ PM ใหม่ re-PM หรืออาจเป็นอีกหนึ่งเทคนิคสำหรับการแก้ปัญหาการติดล๊อคคำตอบแบบเฉพาะถิ่นของหน่วยค้นหาสุดท้ายของ MATS นอกเหนือจาก BT ของ ATS ย่อยเอง เพราะ MATS เมื่อดำเนินการถึงระยะเวลาหนึ่งจำนวนหน่วยค้นหาจะลดลงเหลือเพียงหนึ่งหน่วยค้นหาแต่ปัญหาการติดล๊อคคำตอบแบบเฉพาะถิ่นก็อาจเกิดขึ้นกับหน่วยค้นหาดังกล่าวได้ ดังนั้น re-PM จะเป็นตัวช่วยจัดการหน่วยหา ซึ่งหากไม่มี re-PM หน่วยค้นหานั้นก็จะดำเนินการไปจนกระทั่งครบจำนวนรอบการค้นหาสูงสุด และอาจรายงานคำตอบที่ยังมีคุณภาพไม่ดีก็ได้ ดังนั้น แนวทางที่จะพัฒนา MATS ให้มีสมรรถนะการค้นหาที่สูงขึ้นก็คือการออกแบบ re-PM ให้เหมาะสม

- กลไกลำดับการค้นหาหรือ SM สำหรับงานวิจัยวิทยานิพนธ์ปัจจุบัน อนุญาตให้แต่ละหน่วยค้นหาย่อยดำเนินการค้นหาได้เพียงหนึ่งรอบค้นหาในแต่ละรอบการค้นหาของ MATS หากมีการทดสอบเพื่อศึกษาผลของจำนวนรอบภายใน SM ต่อสมรรถนะการค้นหารวมของ

MATS เช่นการเปลี่ยนจำนวนรอบภายใน SM เป็น 2 3 4 และอื่น ๆ แล้วเปรียบเทียบผลการค้นหาหลาย ๆ ปัญหาการหาเหมาะที่สุดเชิงพื้นผิว หรือปัญหารูปแบบอื่นอาจได้ข้อมูลที่เป็นประโยชน์เพื่อเป็นแนวทางการพัฒนา MATS ได้ในบริบทของ SM ต่อไปนี้ แนวทางนี้ถือว่าเป็นการศึกษาผลของ SM เชิงสถิติ

- การสร้าง SM เชิงปรับตัว มีแนวคิดคล้ายคลึงกับกลไกปรับรัศมีหรือ AR ของATS ซึ่งทำหน้าที่เร่งการค้นหาของ ATS ให้เข้าสู่คำตอบได้เร็วขึ้น ส่วน SM เชิงปรับตัวนี้ จะให้สิทธิ์การค้นหาสำหรับหน่วยค้นหาที่ผ่านเกณฑ์ประเมินว่าเข้าใกล้คำตอบ สิทธิดังกล่าวคือการได้สิทธิ์ดำเนินการค้นหาได้มากกว่าหน่วยค้นหาอื่น ๆ เช่น มากกว่าหนึ่งรอบหรือสองรอบตามความเหมาะสม ซึ่งผลการเพิ่มเพียงหนึ่งหรือสองรอบการค้นหาของหน่วยค้นหาที่มีคุณภาพอาจทำให้ MATS พบคำตอบได้ไวขึ้น จึงเป็นการพัฒนา MATS ได้อีกรูปแบบหนึ่งในบริบทของ SM

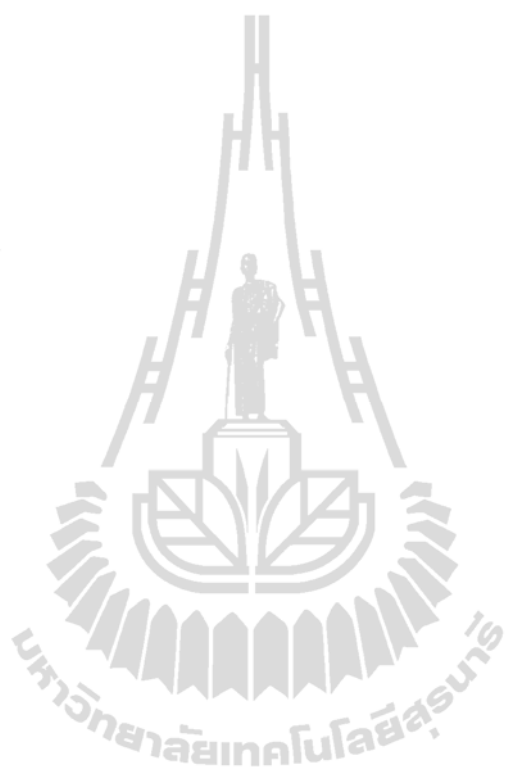
- การเรียกใช้กลไกยกเลิกการค้นหาหรือ DM สำหรับงานวิจัยวิทยานิพนธ์ปัจจุบัน เรียกใช้ตามจำนวนรอบที่ผู้ใช้กำหนด ซึ่งได้จากประสบการณ์ที่ดำเนินงานกับปัญหานั้นมาแล้วระดับหนึ่ง แนวทางการปรับปรุงเทคนิคการเรียกใช้ DM โดยพิจารณาจากคุณภาพของคำตอบปัจจุบันที่หาได้ เช่น หากคุณภาพของคำตอบหนึ่งของทุกหน่วยค้นหาถึงเกณฑ์ที่กำหนดไว้ก็เรียกใช้ DM อาจช่วยให้ MATS ได้สมรรถนะการค้นหาที่ดีขึ้นได้

นอกจากนั้น รูปแบบของ DM สำหรับงานวิจัยวิทยานิพนธ์ปัจจุบัน ส่วนใหญ่ใช้การลดลงแบบทีละกึ่งหนึ่งจนเหลือเพียงหน่วยค้นหาเดียว กับอีกรูปแบบหนึ่งคือการลดลงแบบทันทีทันใด เช่น จาก 31 หน่วยค้นหาเหลือเพียง 1 หน่วยค้นหา หรือจาก 64 หน่วยค้นหาเหลือเพียง 1 หน่วยค้นหา เป็นต้น รูปแบบเหล่านี้ได้จากการทดสอบของผู้ใช้ หากตั้งให้รูปแบบของ DM เป็นแบบปรับตัวได้ ตามลักษณะของปัญหา ซึ่งหมายถึง DM รูปแบบใหม่จะต้องรวมเอารูปแบบการเรียกใช้ DM และรูปแบบการลดจำนวนหน่วยค้นหา เพื่อให้บรรลุเป้าหมายการเพิ่มสมรรถนะของ MATS นั่นเอง

3. ในด้านการประยุกต์อัลกอริธึม MATS สำหรับระบบ IRS ดังเดิม ยังปรากฏผลของการรบกวนภายนอกอยู่ในรูปอัตราขยายคงที่ (dc gain) หากมีการออกแบบตัวควบคุมใหม่ให้สามารถกำจัดผลของการรบกวนได้จะทำให้ระบบมีสมรรถนะที่ดีขึ้น ทั้งนี้ต้องอยู่ภายใต้การรักษาอัตราขยายวงรอบปิดรวมไว้

การพัฒนาการระบุเอกลักษณ์แบบจำลองระบบ CPP แบบไม่เชิงเส้น อาจดำเนินการด้วยความเชื่อที่ว่าหากสามารถอธิบายพลวัตของแรงที่ส่งผ่านสายพานไปยังตัวรถด้วยฟังก์ชัน f ที่มีความเที่ยงตรงมากขึ้น ผ่านกระบวนการระบุเอกลักษณ์ที่เหมาะสม จะทำให้เราได้แบบจำลองระบบ CPP แบบไม่เชิงเส้น ที่แม่นยำมากขึ้นนั่นเอง

นอกจากนั้น มีความเป็นไปได้ที่จะประยุกต์ MATS กับการแก้ปัญหาที่ซับซ้อนอื่น ๆ ไม่ว่าจะเป็นทางวิศวกรรมศาสตร์ เศรษฐศาสตร์ เป็นต้น ปัญหาที่น่าสนใจอาจเป็นของการจัดการพลังงานไฟฟ้ารูปแบบต่าง ๆ การบริหารจัดการน้ำ การลงทุนในตลาดหลักทรัพย์ ตลอดจนการควบคุมอุปกรณ์สวิตช์ในโครงข่ายโทรคมนาคมขนาดใหญ่ เหล่านี้เป็นต้น



รายการอ้างอิง

- เดชา พวงดาวเรือง. (2547). การระบุเอกลักษณ์ของระบบพลวัตด้วยการประมวลผลภาพ. วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต, มหาวิทยาลัยเทคโนโลยีสุรนารี, นครราชสีมา.
- สราวุฒิ สุจิตจร. (2547). เอกสารประกอบการสอนระดับบัณฑิตศึกษารายวิชา 403604 แบบจำลองระบบ. มหาวิทยาลัยเทคโนโลยีสุรนารี, นครราชสีมา.
- Battiti, R., and Tecchiolli, G. (1994). The reactive tabu search. **ORSA Journal on Computing**. 6(2): 126-140.
- Bloch, A.M., Leonard, N.E., and Marden, J.E. (2000). Controlled lagrangians and the stabilization on mechanical system I: the first matching theorem. **IEEE Transaction on Automatic Control**. 45(12): 2253-2270.
- Brennan, S., and Alleyne, A. (2000). The Illinois roadway simulator: a mechanic testbed for vehicle dynamics. **IEEE/ASME Transaction on Mechanics**. 5(4): 349-359.
- Brennan, S., and Alleyne, A. (2001). Using a scaled tested: controller design and evaluation. **IEEE Control Systems Magazine**. 21(3): 15-26.
- Cannon, R.H. (1967). **Dynamic of physical systems**. McGraw-Hill.
- Chen, J-S., Pan, J. C-H., and Wu, C-K. (2006). Minimizing makespan in reentrant flow-shops using hybrid tabu search. **Int. J Adv Manuf Technol**. Springer Verlag London. (online).
- Crainic, T.G., Toulouse, M., and Gendreau, M. (1996). Parallel asynchronous tabu search for multicommodity location-allocation with balancing requirements. **Annals of Operations Research**. 63: 277-299.
- Davies, W.D.T. (1970). **System identification for self-adaptive control**. John Wiley & Sons.
- Eykhoff, P. (1974). **System identification: Parameter and state estimation**. John Wiley & Sons. Feedback Instruments Ltd. Digital pendulum control system.
- Franca, P. M., Sosa, N. M., and Pureza, V. (1999). An adaptive tabu search algorithm for the capacitated clustering problem. **International Transaction In Operation Research**. 6: 665-678.

- Ghosh, D. (2003). Solving medium to large sized Euclidean generalized minimum spanning tree problem. Technical report NEP-CMP-2003-09-28. **Indian Institute of Management, Research and Publication Department**, India.
- Glover, F. (1978). Parametric branch and bound. **The International Journal of Management Science**. 6: 145-152.
- Glover, F. (1986). Future paths for integer programming and link to artificial intelligence. **Computers and Operation Research**. 13: 533-549.
- Glover, F. (1989). Tabu search – Part I. **ORSA Journal on Computing**. 1: 190-206.
- Glover, F. (1990). Tabu search – Part II. **ORSA Journal on Computing**. 2: 4-32.
- Glover, F., and Laguna, M. (1997). **Tabu Search**. Kluwer Academic Publishers. Norwell.
- Glover, F. (2006). Parametric tabu-search for mixed integer programs. **Journal of Computers and Operations Research**. 33: 2449-2494.
- Gong, D., Zhou, Y., Guo, X., Ma, X., and Li, M. (2002). Study on an adaptive tabu search genetic algorithm. In **Proceedings of the 4th World Congress on Intelligent Control and Automation**. (pp.3063-3065).
- Johansson, R. (1993). **System Modeling and Identification**. Prentice-Hall.
- Kalinli, A. (2004). Component value selection for active filters using parallel tabu search algorithm. **Int. J. Electron. Commun. (AEU)**. 60: 85-92.
- Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2008). Management agent for search algorithms with surface optimization applications. **WSEAS Transactions on Computers**. 7(6): 791-803.
- Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2009). Performance assessment of search management agent under asymmetrical problems and control design applications. **WSEAS Transactions on Computers**. 8(4): 691-704.
- Kluabwang, J., and Sujitjorn, S. (2009). Tabu search: Its modified forms and management agent. **Recent Patents on Electrical Engineering**. 2(2): 135-145.
- Kochetov, Y.A., and Goncharov, E.N. (2000). Probabilistic tabu search algorithm for multi-stage uncapacitated facility location problem. In. **Operations Research Proceeding. Springer**. (pp.65-70).

- Kolahan, F., and Liang, M. (1998). An adaptive tabu search approach to JIT sequencing with variable processing times and sequence-dependent setups. **European Journal of Operational Research**. 109: 142-159.
- Landau, I.D. (1990). **System identification and control design using P.I.M. and software**. Prentice-Hall.
- Ljung, L., and Glad, T. (1994). **Modeling of dynamic systems**. PRT Prentice-Hall.
- Liu, Yo., Liu, Ya., Wang, L., and Chen, K. (2005). A hybrid tabu search based clustering algorithm. **Lecture Notes in Artificial Intelligence**. Springer-Verlag Berlin Heidelberg. 3682: 66-77.
- Mori, H., and Tani, H. (2003). Fast hybrid method of parallel tabu search and ordinal optimization for on-line capacitor control in distribution systems. **IEEE General Meeting of Power Engineering Society**. 3: 1553-1558.
- Nowicki, E., and Smutnicki, C. (2005). An advanced tabu search algorithm for the job shop problem. **Journal of Scheduling**. 8: 145-159.
- Ongsakul, W., and Bhasaputra, P. (2002). Optimal power flow with FACTS devices by hybrids TS/SA approach. **Journal of Electrical & Energy Systems**. 24: 851-857.
- Pedroso, J. P., and Kubo, M. (2005). Hybrid tabu search for lot sizing problem. **Lecture Notes in Computer Science**. Springer-Verlag Berlin Heidelberg. 3636: 66-77.
- Puangdownreong, D., Kluabwang, J., and Sujitjorn, S. (2010). Application of adaptive tabu search under management agent to control synthesis of scaled vehicle. **IASTED International Conference on Modelling Identification and Control (MIC 2010)**. (pp.1-6).
- Richard, L. B., and Fairs, J. D. (2001), **Numerical Analysis**. BROOKS/COLE.
- Sait, S. M., Youssef, H., Barada, H. R., and Ahmad A.-Y. (2000). Parallel tabu search algorithm for VLSI standard-cell placement. **IEEE International Symposium on Circuits and Systems ISCAS 2000**. (pp.581-584).
- Sujitjorn, S., Kulworawanichpong, T., Puangdownreong, D., and Areerak, K-N. (2006). Adaptive tabu search and applications in engineering design. **Frontiers in Artificial Intelligent and Applications**. IOS Press, Netherland. (pp.233-257)

- Sujitjorn, S., Kluabwang, J., Puangdownreong, D., and Sarasiri, N. (2010). Adaptive tabu search and management agent. **ECTI Transactions on Electrical Engineering, Electronics and Communications**. 8(1): 1-10.
- Swarnkar, R., and Tiwari, M.K. (2004). Modeling machine loading problem of FMSs and its solution methodology using a hybrid tabu search and simulated annealing-based heuristic approach. **Journal of Robotics and Computer-Integrated Manufacturing**. 20: 199-209.
- Tahir, M. A., Bouridane A., and Kurugollu, F. (2007). Simultaneous feature selection and feature weighting using hybrid tabu search/K-nearest neighbor classifier. **Journal of Pattern Recognition Letters**. 28: 438-446.
- Talbi, E.G., Hafidi, Z., and Geib, J-M. (1998). A parallel adaptive tabu search approach. **Parallel Computing**. 24: 2003-2019.
- Victoire, T. A. A., and Jeyakumar, A. E. (2006). A tabu search based hybrid optimization approach for a fuzzy modeled unit commitment problem. **Journal of Electric Power Systems Research**. 76: 413-425.
- Wassan, N. (2006). A reactive tabu search for the vehicle routing problem. **Journal of the ORSA Journal on Society**. 57: 111-116.
- Xu, J., Chiu, S. Y., and Glover, F. (1997). Probabilistic tabu search for telecommunications network design. **Journal of Combinatorial Optimization: Theory and Practice**. 1(1): 69-94.
- Zhang, D., Fu, Z., and Zhang, L. (2007). An improved TS algorithm for loss-minimum reconfiguration in large-scale distribution systems. **Journal of Electric Power Systems Research**. 77: 685-694.



ภาคผนวก ก

โปรแกรมการค้นหาแบบตาบู่เชิงปรับตัวหลายวิธี

ก.1 โปรแกรมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี

โปรแกรมการค้นหาแบบตามูเชิงปรับตัวหลายวิธี หรือ โปรแกรม MATS ในงานวิจัยวิทยานิพนธ์ฉบับนี้ ได้พัฒนาขึ้นโดยอาศัยโปรแกรม MATLABTM ดังรายละเอียดดังต่อไปนี้ (แบบสองมิติทำการค้นหากับฟังก์ชันราสตริกิน เพื่อเป็นตัวอย่าง)

ก.1.1 โปรแกรม MATS ที่พัฒนาด้วย MATLABTM

```
% โปรแกรม ATS ที่พัฒนาด้วย MATLABTM
% เริ่มต้นประดิษฐ์ โดยกองพัน อารีรักษ์ (2002)
% พัฒนาและปรับปรุง โดยเดชา พวงดาวเรือง (2003)
% พัฒนาเป็นโปรแกรม MATS โดยจักรกฤษณ์ เคลือบวัง (2008)
% สาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารี

โปรแกรม MATS ชนิด 2 หน่วยค้นหา หรือ DATS.m
function [overall_best_neighbor,overall_best_error, time, radius,count,local1]=DATS(N)
% โปรแกรมหลักมี N เป็นอินพุตของโปรแกรม
% N คือจำนวนพารามิเตอร์ที่ต้องการค้นหา (N = 2 เพื่อเป็นตัวอย่าง)
% เอาต์พุตของโปรแกรมคือ
% ค่าพารามิเตอร์ที่ดีที่สุดหรือค่าค่าตอบวงกว้าง (overall_best_neighbor)
% ค่าฟังก์ชันวัตถุประสงค์ที่ดีที่สุด (overall_best_error)
% ค่าเวลาที่ใช้ในการค้นหา (time)
% ค่าจำนวนรอบการค้นหาทั้งหมด (count)
% ค่ารัศมีการค้นหาของ ATS1 และ ATS2 (radius1 และ radius2)
% ค่าค่าตอบวงแคบท้องถิ่นของ ATS1 และ ATS2 (local1 และ local2)

xlimit=[2 2;-2 -2];
% กำหนดปริภูมิการค้นหาของพารามิเตอร์
% แถวที่ 1 ของ xlimit คือขอบเขตบนของพารามิเตอร์
% แถวที่ 2 ของ xlimit คือขอบเขตล่างของพารามิเตอร์

% Start Partitioning Mechanism
% PM แบ่งปริภูมิการค้นหาของพารามิเตอร์ออกเป็นสองส่วนตามแนวนอน
xlimit1=[ xlimit(1) xlimit(3);mean([xlimit(1) xlimit(2)]) xlimit(4)];
xlimit2=[ mean([xlimit(1) xlimit(2)]) xlimit(3); xlimit(2) xlimit(4)];

for r=1:5
S(r,:)=(xlimit1(1,:)-xlimit1(2,:)).*rand(1,N))+xlimit1(2,:);
end

% PM สุ่มคำตอบเริ่มต้นภายในปริภูมิการค้นหาทั้งสองแต่ละตัวจำนวน 5 ชุด
% โดยอาศัยความสัมพันธ์  $(x_{h,i} - x_{l,i})\text{rand} + x_{l,i}$ , for  $i=1, \dots, 5$  ของหน่วยค้นหาที่หนึ่ง
```

```

for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2)); % ประเมินค่าคำตอบด้วยโปรแกรม obj.m แล้วเก็บไว้ในตัวแปร ysim
    costvalue(k,1)=ysim; % เก็บค่าฟังก์ชันวัตถุประสงค์ทั้ง 5 ค่าไว้ใน costvalue
end
[best_error,index]=min(costvalue); % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดใน costvalue แล้วเก็บไว้ใน best_error
S01=S(index,:); % และค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดจะถูกกำหนดให้เป็นคำตอบเริ่มต้นของหน่วย
% ค้นหาที่สอง โดยเก็บไว้ใน S01

% PM สุ่มคำตอบเริ่มต้นภายในปริภูมิการค้นหาย่อยทั้งสองแต่ละตัวจำนวน 5 ชุด
% โดยอาศัยความสัมพันธ์  $(x_{h,i} - x_{i,i})\text{rand} + x_{i,i}$ , for  $i=1,\dots,5$ 
% สำหรับหน่วยค้นหาที่สอง

for r=1:5
    S(r,:)=((xlimit2(1,:)-xlimit2(2,:)).*rand(1,N))+xlimit2(2,:);
end

for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2)); % ประเมินค่าคำตอบได้แล้วเก็บไว้ในตัวแปร ysim
    costvalue(k,1)=ysim; % เก็บค่าฟังก์ชันวัตถุประสงค์ทั้ง 5 ค่าไว้ใน costvalue
end
[best_error,index]=min(costvalue); % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดใน costvalue แล้วเก็บไว้ใน best_error
S02=S(index,:); % และค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดจะถูกกำหนดให้เป็นคำตอบเริ่มต้นของหน่วย
% ค้นหาที่สอง โดยเก็บไว้ใน S02

S00=[S01;S02]; % เก็บพารามิเตอร์ทั้งของ S01 และ S02 ไว้ใน S00

% Initial solutions evaluated
best_error01=obj(S01(1),S01(2)); % ประเมินค่าฟังก์ชันวัตถุประสงค์ของคำตอบ S01
best_error02=obj(S02(1),S02(2)); % ประเมินค่าฟังก์ชันวัตถุประสงค์ของคำตอบ S02

xlimit1=xlimit; % ค้นหาปริภูมิการค้นหาของพารามิเตอร์ให้กับทั้ง ATS1
xlimit2=xlimit; % ค้นหาปริภูมิการค้นหาของพารามิเตอร์ให้กับทั้ง ATS2

% Shared parameters
max_count=10000; % จำนวนรอบการค้นหาสูงสุด
radius=0.2; % รัศมีการค้นหาที่กำหนดไว้ 0.2 คิดเป็น 5% ของปริภูมิการค้นหา
Number_neighb=30; % จำนวนคำตอบข้างเคียงที่กำหนดไว้ 30 ค่า
count=0; % ตัวนับจำนวนรอบการค้นหาเริ่มต้นกำหนดให้เป็นศูนย์

% Initialize all individual parameters
% sub search space # 1
radius1=0.2; % รัศมีการค้นหาเริ่มต้นของหน่วยค้นหาที่หนึ่งกำหนดไว้ 0.2
best_neighbor01=S01; % เก็บคำตอบเริ่มต้น ATS1 ไว้ที่ best_neighbor01
overall_best_error1=best_error01; % กำหนด overall_best_error1 เท่ากับ best_neighbor01
overall_best_neighbor1=best_neighbor01; % กำหนด overall_best_neighbor1 เท่ากับ best_neighbor01

n1=0; % กำหนดค่าเริ่มต้นของจำนวนการซ้ำของคำตอบ ของ ATS1
t1=0; % กำหนดค่าตัวเลื่อนสำหรับ local ของ ATS1

```

```

tt1=0; % กำหนดค่าตัวเลื่อนสำหรับ tabu_list ของ ATS1
ttt1=0; % กำหนดค่าตัวเลื่อนสำหรับ best_error_list ของ ATS1
n_back_tracking1=0; % กำหนดค่าเริ่มต้นของจำนวนการเรียกใช้กลไกย้อนรอยการค้นหาของ ATS1

% sub search space # 2
radius2=0.2; % รัศมีการค้นหาเริ่มต้นของหน่วยค้นหาที่สองกำหนดไว้ 0.2
best_neighbor02=S02; % เก็บค่าตอบเริ่มต้น ATS2 ไว้ที่ best_neighbor02
overall_best_error2=best_error02; % กำหนด overall_best_error2 เท่ากับ best_neighbor02
overall_best_neighbor2=best_neighbor02; % กำหนด overall_best_neighbor2 เท่ากับ best_neighbor02

n2=0; % กำหนดค่าเริ่มต้นของจำนวนการซ้ำของค่าคำตอบ ของ ATS2
t2=0; % กำหนดค่าตัวเลื่อนสำหรับ local ของ ATS2
tt2=0; % กำหนดค่าตัวเลื่อนสำหรับ tabu_list ของ ATS2
ttt2=0; % กำหนดค่าตัวเลื่อนสำหรับ best_error_list ของ ATS2
n_back_tracking2=0; % กำหนดค่าเริ่มต้นของจำนวนการเรียกใช้กลไกย้อนรอยการค้นหาของ ATS2
tic; % เริ่มต้นคำนวณเวลาที่ใช้ในการค้นหา

% Start ATS

t1=t1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local1 ของ ATS1
tt1=tt1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1 ของ ATS1
ttt1=ttt1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list1 ของ ATS1

t2=t2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local2 ของ ATS2
tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2 ของ ATS2
ttt2=ttt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list2 ของ ATS2

disp([count overall_best_error1 overall_best_error2]);
% แสดงค่า count, overall_best_neighbor1 และ overall_best_neighbor/

% Sub search space # 1
local1(t1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local1
local1(t1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local1
local1(t1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ local1
tabu_list1(tt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list1
tabu_list1(tt1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list 1
tabu_list1(tt1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ tabu_list 1
best_error_list1(ttt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list1
best_error_list1(ttt1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list1
best_error_list1(ttt1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ best_error_list1

% Sub search space # 2
local2(t2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local2
local2(t2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local2
local2(t2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ local2

```

```

tabu_list2(tt2,1)=count;          % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list2
tabu_list2(tt2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list 2
tabu_list2(tt2,4)=best_error02;   % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ tabu_list 2
best_error_list2(ttt2,1)=count;   % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list2
best_error_list2(ttt2,2:3)=best_neighbor02;
                                % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list2
best_error_list2(ttt2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ best_error_list2

% Path Enabling
flag1=1;                        % กำหนดค่าเริ่มต้น flag1 ของ ATS1 เท่ากับหนึ่งเพื่อให้ทำงานได้
flag2=1;                        % กำหนดค่าเริ่มต้น flag2 ของ ATS/2 เท่ากับหนึ่งเพื่อให้ทำงานได้

% Start ATS
while (count<max_count)        % วงรอบการค้นหา
count=count+1;                 % ปรับเพิ่ม count ขึ้นจากเดิมหนึ่งค่า

% Path 1
if (flag1==1)                  % การคงอยู่ในวงรอบการค้นหาของ ATS1 เกิดขึ้นภายใต้เงื่อนไข flag1==1 เท่านั้น
S101=random_neigh(Number_neighb,radius1,xlimit1,S01);
                                % เรียกใช้โปรแกรม random_neigh.m เพื่อสุ่มค่าใกล้เคียงรอบคำตอบ S01
                                % อินพุตของฟังก์ชันนี้คือ Number_neighb, radius1, xlimit1 และ S01
                                % เอาท์พุตคือค่าใกล้เคียง S101 จำนวนเท่ากับ Number_neighb (30 ตัว)
[best_error101,best_neighbor101,best_error01,best_neighbor01]=objective(S101,best_error01,S01);
                                % เรียกใช้โปรแกรม objective.m เพื่อประเมินค่าใกล้เคียงที่สุ่มได้
                                % อินพุตของฟังก์ชันนี้คือ S101, best_error01, และ S01
                                % เอาท์พุตคือ best_error101, best_neighbor101, best_error01 และ
                                % best_neighbor01

if (best_error101-best_error01)>1e-18
n1=n1+1;
else
n1=0;
end                             % ตรวจสอบการซ้ำของคำตอบโดยเปรียบเทียบระหว่าง best_error101 และ
                                % best_error01 ถ้ามีการซ้ำให้ปรับเพิ่มค่า n1

tt1=tt1+1;                      % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1 ของ ATS1
tabu_list1(tt1,1)=count;        % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list1

tabu_list1(tt1,2:3)=best_neighbor101;
                                % เก็บค่า best_neighbor101 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list1
tabu_list1(tt1,4)=best_error101; % เก็บค่า best_error101 ไว้ในคอลัมน์ที่ 4 ของ tabu_list1
ttt1= ttt1+1;                  % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list1 ของ ATS1
best_error_list1(ttt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list1
best_error_list1(ttt1,2:3)=best_neighbor01;
                                % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list1
best_error_list1(ttt1,4)=best_error01;
                                % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ best_error_list1

```

```

% Start AR mechanism                                (กำหนดให้มี 2 เงื่อนไขเพื่อเป็นตัวอย่าง)

if best_error01<=1e-3                                % เงื่อนไขที่ 1) ถ้า best_error01 ≤ 1 x 10-1 แล้ว radius1=2 x 10-3
radius1=2e-3;
end
if best_error01<=1e-3                                % เงื่อนไขที่ 2) ถ้า best_error01 ≤ 1 x 10-3 แล้ว radius1=2 x 10-4
radius1=2e-4;
end
if (overall_best_error1<1e-9)                        % ตรวจสอบเกณฑ์ยุติการค้นหา เมื่อ best_error01 < 1 x 10-9 แล้ว
t1=t1+1;                                              % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local1
tt1=tt1+1;                                            % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1
local1(t1,1)=count;                                  % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local1
local1(t1,2:3)=best_neighbor01;                     % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local1
local1(t1,4)=best_error01;                           % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ local1
disp('Path 1 Strikes');                              % แสดงผลว่า 'Path 1 Strikes'
break;                                                % ยุติการค้นหา
end

% Start BT mechanism                                % เมื่อจำนวนการซ้ำของค่าคำตอบเท่ากับ 5 (เพื่อเป็นตัวอย่าง)

% sub search space # 1
if n1>=5
n_back_tracking1=n_back_tracking1+1;
% เพิ่มค่า n_back_tracking1
TEMP1=tabu_list1(count-3:count+1,:);
% เก็บค่าพารามิเตอร์จำนวน 5 ชุดสุดท้ายก่อนทำการจัดลำดับไว้ใน TEMP1
[ MAX1,INDEX1 ] = max(TEMP1(:,4));
% หาค่าฟังก์ชันวัตถุประสงค์สูงสุดและลำดับที่ใน TEMP1
RANK1(5,:) = TEMP1(INDEX1,:);
% เก็บค่าพารามิเตอร์ที่มีฟังก์ชันวัตถุประสงค์สูงสุดไว้ใน RANK1
neighbor1=RANK1(5,2:3);
% เก็บค่า RANK1 ไว้ใน neighbor1
S01=neighbor1;
% แทนค่า S01 ด้วย neighbor1
best_error01=RANK1(5,4);
% แทนค่า best_error01 ด้วย RANK1
n1=0;
% ปรับค่า n1=0
else
% ไม่เช่นนั้นแล้ว
S01=best_neighbor01;
% แทนค่า S01 ด้วย best_neighbor01
best_error01=best_error01;
% แทนค่า best_error01 ด้วย best_error01
end

% Path # 1
if overall_best_error1>best_error01
overall_best_error1=best_error01;
overall_best_neighbor1=best_neighbor01;
end
% ถ้า overall_best_error1 > best_error01 แล้ว
% overall_best_error1=best_error01
% overall_best_neighbor1=best_neighbor01

count1=count;
% แทนค่า count1 ด้วย count1
end
% สิ้นสุดวนรอบการค้นหาของ ATS1

```

```

% Path 2
if (flag2==1) % การคงอยู่ในวงรอบการค้นหาของ ATS2 เกิดขึ้นภายใต้เงื่อนไข flag2==1 เท่านั้น

% sub search space # 2
S102=random_neigh(Number_neighb,radius2,xlimit2,S02);
% เรียกใช้ฟังก์ชัน random_neigh เพื่อสุ่มค่าใกล้เคียงรอบคำตอบ S02
% อินพุตของฟังก์ชันนี้คือ Number_neighb, radius2, xlimit2 และ S02
% เอาท์พุตคือค่าใกล้เคียง S102 จำนวนเท่ากับ Number_neighb (30 ตัว)
[best_error102,best_neighbor102,best_error02,best_neighbor02]=objective(S102,best_error02,S02);
% เรียกใช้ฟังก์ชัน objective เพื่อประเมินค่าใกล้เคียงที่สุ่มได้
% อินพุตของฟังก์ชันนี้คือ S102, best_error02, และ S02
% เอาท์พุตคือ best_error102, best_neighbor102, best_error02 และ
% best_neighbor02

% Path # 2
if (best_error102-best_error02)>1e-18
n2=n2+1;
else
n2=0;
end % ตรวจสอบการซ้ำของค่าคำตอบโดยเปรียบเทียบระหว่าง best_error102 และ
% best_error02 ถ้ามีการซ้ำให้ปรับเพิ่มค่า n2

tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2 ของ ATS2
tabu_list2(tt2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list2
tabu_list2(tt2,2:3)=best_neighbor102; % เก็บค่า best_neighbor102 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list2
tabu_list2(tt2,4)=best_error102; % เก็บค่า best_error102 ไว้ในคอลัมน์ที่ 4 ของ tabu_list2
ttt2= ttt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list2 ของ ATS2
best_error_list2(ttt2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list2
best_error_list2(ttt2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list2
best_error_list2(ttt2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ best_error_list2

% Start AR (กำหนดให้มี 2 เงื่อนไขเพื่อเป็นตัวอย่าง)
if best_error02<=1e-1 % เงื่อนไขที่ 1) ถ้า best_error02 ≤ 1 x 10-1 แล้ว radius1=2 x 10-3
radius2=2e-3;
end
if best_error02<=1e-3 % เงื่อนไขที่ 2) ถ้า best_error02 ≤ 1 x 10-3 แล้ว radius1=2 x 10-4
radius2=2e-4;
end
if (overall_best_error2<1e-9) % ตรวจสอบเกณฑ์ยุติการค้นหา เมื่อ best_error02 < 1 x 10-9 แล้ว
t2=t2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local2
tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2
local2(t2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local2
local2(t2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local2
local2(t2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ local2

```

```

disp('Path 2 Strikes');          % แสดงผลว่า 'Path 2 Strikes'
break;                          % ยุติการค้นหา
end

% Start BT                      % ตรวจสอบเกณฑ์ยุติการค้นหา เมื่อ best_error01 < 1 x 10-9 แล้ว
if n2 >= 5
    n_back_tracking2 = n_back_tracking2 + 1;
    % เพิ่มค่า n_back_tracking2
    TEMP2 = tabu_list2(count-3:count+1,:);
    % เก็บค่าพารามิเตอร์จำนวน 5 ชุดสุดท้ายก่อนทำการจัดลำดับไว้ใน TEMP2
    [MAX2, INDEX2] = max(TEMP2(:,4));
    % หาค่าฟังก์ชันวัตถุประสงค์สูงสุดและลำดับที่ใน TEMP2
    RANK2(5,:) = TEMP2(INDEX2,:); % เก็บค่า RANK2 ไว้ใน neighbor2
    neighbor2 = RANK2(5,2:3);     % เก็บค่า RANK2 ไว้ใน neighbor2
    S02 = neighbor2;             % แทนค่า S02 ด้วย neighbor2
    best_error02 = RANK2(5,4);    % แทนค่า best_error02 ด้วย RANK2
    n2 = 0;                     % ปรับค่า n2 = 0
else
    % ไม่เช่นนั้นแล้ว
    S02 = best_neighbor02;        % แทนค่า S02 ด้วย best_neighbor02
    best_error02 = best_error02; % แทนค่า best_error02 ด้วย best_error02
end

% Updating Path # 2
if overall_best_error2 > best_error02
    overall_best_error2 = best_error02;
    overall_best_neighbor2 = best_neighbor02;
end
% ถ้า overall_best_error2 > best_error02 แล้ว
% overall_best_error2 = best_error02
% overall_best_neighbor2 = best_neighbor02

count2 = count;                % แทนค่า count1 ด้วย count2
end
% สิ้นสุดวนรอบการค้นหาของ ATS2

% Start Discarding Mechanism
if count == 250
    % เนื่องจากการเรียกใช้ DM กำหนดตามรอบการค้นหา
    % (เลือกรอบการค้นหาที่ 250 เพื่อเป็นตัวอย่าง)
    if (overall_best_error2 < overall_best_error1)
        % เปรียบเทียบค่าฟังก์ชันวัตถุประสงค์ปัจจุบันของ ATS1 และ ATS2
        % เลือกการจัดหน่วยค้นหาที่มีค่าฟังก์ชันวัตถุประสงค์สูงกว่าออกจากวงรอบการค้นหา
        flag1 = 0;
        % ด้วยการกำหนดค่า flag = 0
    else
        flag2 = 0;
    end
end

disp([count overall_best_error1 overall_best_error2]);
% แสดงค่า count, overall_best_error1, overall_best_error2

end
time = toc
% ยุติการคำนวณการค้นหาของ DATS
return

```

โปรแกรม obj.m

```
function y=obj(x,y) % โปรแกรม obj คือโปรแกรมคำนวณค่าฟังก์ชันวัตถุประสงค์
% กำหนดให้เป็นฟังก์ชันราศตรีโกณ (เพื่อเป็นตัวอย่าง)
% อินพุตของโปรแกรมคือพิกัดคำตอบ (x,y)
% เอาต์พุตคือค่าฟังก์ชันวัตถุประสงค์ของพิกัดคำตอบ (x,y) ดังสมการต่อไปนี้
y=(x.^2)+(y.^2)-(10*cos(2*pi*x))- (10*cos(2*pi*y))+20;
return
```

โปรแกรม objective.m

```
function [best_error1,best_neighbor1,best_error,best_neighbor]=objective(S1,best_error,S0)
% โปรแกรม objective เป็นโปรแกรมประเมินค่าใกล้เคียงที่สุ่มได้รอบ S0
% อินพุตของโปรแกรมนี้คือ S1, best_error, และ S0
% เอาต์พุตคือ best_error1, best_neighbor1,
% best_error, และ best_neighbor
error=[]; % ตัวแปร error สำหรับรองรับค่าฟังก์ชันวัตถุประสงค์
% ของค่าใกล้เคียงที่ได้จากการสุ่ม
for k=1:size(S1,1)
ysim=obj(S1(k,1),S1(k,2)); % ประเมินค่าใกล้เคียงใน S1 ด้วยฟังก์ชันวัตถุประสงค์ในโปรแกรม obj
error(k,1)=ysim; % เก็บค่าฟังก์ชันวัตถุประสงค์ของค่าใกล้เคียงทั้งหมด
end
[best_error1,index]=min(error); % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดในบรรดาค่าใกล้เคียง
% และเก็บไว้ใน best_error1
best_neighbor1=S1(index,:); % เก็บค่าพารามิเตอร์ของค่าใกล้เคียงที่ทำให้
% ค่าฟังก์ชันวัตถุประสงค์มีค่าน้อยที่สุดไว้ใน best_neighbor1
if best_error1<best_error % ปรับค่า best_error
best_error=best_error1; % ถ้า best_error1 < best_error
% ให้แทนค่า best_error ด้วย best_error1
best_neighbor=S1(index,:); % และแทนค่าพารามิเตอร์ best_neighbor ด้วย S1(index,:)
else % ไม่เช่นนั้นแล้ว
best_neighbor=S0; % แทนค่า best_neighbor ด้วย S0
end
return
```

โปรแกรม rand1.m

```
function x=rand1(a,b)           % โปรแกรม rand1 เป็นโปรแกรมสุ่มค่าพารามิเตอร์ในช่วงที่กำหนด
                                % อินพุตของโปรแกรมคือขอบเขตบนของพารามิเตอร์ (b) และ
                                % ขอบเขตล่างของพารามิเตอร์ (a)
                                % โดยอาศัยความสัมพันธ์  $(b-a)\text{rand} + a$ 
                                % เอาต์พุตของโปรแกรมคือผลของการสุ่มค่าพารามิเตอร์ในช่วง a ถึง b

    x=a+rand*(b-a);
return
```

โปรแกรม random_neigh.m

```
function S1=random_neigh(Number_neighb,radius,xlimit,S0)
                                % โปรแกรม random_neigh เป็นโปรแกรมสุ่มค่าใกล้เคียงรอบคำตอบ S0
                                % อินพุตของโปรแกรมนี้คือ Number_neighb, radius, xlimit, และ S0
                                % เอาต์พุตคือค่าใกล้เคียง (S1) จำนวนเท่ากับ Number_neighb

    for u=1:Number_neighb
    for k=1:size(xlimit,2)
        S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));
                                % ทำการสุ่มค่าใกล้เคียงโดยอาศัยความสัมพันธ์
                                %  $S_{1,i} = S_0 + (\text{radius} \cdot \text{rand1}_i \cdot (-1,1) \cdot (x_{h,i} - x_{l,i}))$ ,
                                % for  $i = 1, \dots, \text{Number\_neighb}$ 
                                % เรียกใช้โปรแกรม rand1 โดยให้ทำการสุ่มค่าในช่วง -1 ถึง 1
                                % นั่นหมายความว่าค่าสุ่มมีพื้นที่อยู่ในช่วง  $S_0 + ((\text{radius}) \cdot (x_{h,i} - x_{l,i}))$ 

        while ( S1(u,k)>xlimit(1,k) | S1(u,k)<xlimit(2,k) )
                                % ค่าใกล้เคียงที่สุ่มได้จะต้องอยู่ภายในขอบเขตบนและขอบเขตล่าง
                                % ของค่าพารามิเตอร์ ถ้าค่าใกล้เคียงที่สุ่มได้
                                % มีค่าเกินขอบเขตบนและขอบเขตล่างที่กำหนด
                                % ให้ทำการสุ่มค่าใกล้เคียงใหม่

        S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));
    end
    end
    end
return
```



ภาคผนวก ข

โปรแกรมออกแบบตัวควบคุมระบบ IRS

ข.1 โปรแกรมออกแบบตัวควบคุมระบบ IRS ด้วย MATS

โปรแกรมการค้นหาแบบตามูเชิงปรับตัวหลายวิธีสำหรับการออกแบบตัวควบคุมสำหรับระบบ IRS มีรายละเอียดดังต่อไปนี้

ข.1.1 โปรแกรม MATS ที่พัฒนาด้วย MATLAB™

```
% โปรแกรมนี้เป็นโปรแกรมออกแบบตัวควบคุมสำหรับระบบ IRS
% เป็นการออกแบบด้วยการค้นหาแบบตามูเชิงปรับตัวหลายวิธี
% โปรแกรมได้รับการพัฒนาด้วย MATLAB™
% พัฒนา โดยจักรกฤษณ์ เคลือบวัง (2008)
% สาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารี

โปรแกรม MATS ชนิด 2 หน่วยค้นหา หรือ DATS_PD.m
function [overall_best_neighbor,overall_best_error, time, radius,count,local1]=DATS_PD(N)
    % โปรแกรมหลักมี N เป็นอินพุตของโปรแกรม
    % N คือจำนวนพารามิเตอร์ที่ต้องการค้นหา (N = 2)
    % เอาต์พุตของโปรแกรมคือ
    % ค่าพารามิเตอร์ที่ดีที่สุดหรือค่าคำตอบวงกว้าง (overall_best_neighbor)
    % ค่าฟังก์ชันวัตถุประสงค์ที่ดีที่สุด (overall_best_error)
    % ค่าเวลาที่ใช้ในการค้นหา (time)
    % ค่าจำนวนรอบการค้นหาทั้งหมด (count)
    % ค่ารัศมีการค้นหาของ ATS1 และ ATS2 (radius1 และ radius2)
    % ค่าคำตอบวงแคบท้องถิ่นของ ATS1 และ ATS2 (local1 และ local2)
    % กำหนดปริภูมิการค้นหาของพารามิเตอร์
    % แถวที่ 1 ของ xlimit คือขอบเขตบนของพารามิเตอร์
    % แถวที่ 2 ของ xlimit คือขอบเขตล่างของพารามิเตอร์

    xlimit=[2 2;0 0];

    % Start Partitioning Mechanism
    % PM แบ่งปริภูมิการค้นหาของพารามิเตอร์ออกเป็นสองส่วนตามแนวนอน
    xlimit1=[ xlimit(1) mean([xlimit(3) xlimit(4)]); xlimit(2) xlimit(4)];
    xlimit2=[ mean([xlimit(1) xlimit(2)]) xlimit(3); xlimit(2) xlimit(4)];

    % PM สุ่มคำตอบเริ่มต้นภายในปริภูมิการค้นหาทั้งสองแต่ละตัวจำนวน 5 ชุด
    % โดยอาศัยความสัมพันธ์  $(x_{h,i} - x_{l,i})\text{rand} + x_{l,i}$ , for  $i=1,\dots,5$ 
    % สำหรับหน่วยค้นหาที่หนึ่ง
    for r=1:5
        S(r,:)=((xlimit1(1,:)-xlimit1(2,:)).*rand(1,N))+xlimit1(2,:);
    end
```

```

for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2));           % ประเมินค่าคำตอบด้วยโปรแกรม obj.m แล้วเก็บไว้ในตัวแปร ysim
    costvalue(k,1)=ysim;               % เก็บค่าฟังก์ชันวัตถุประสงค์ทั้ง 5 ค่าไว้ใน costvalue
end
[best_error,index]=min(costvalue);    % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดใน costvalue แล้วเก็บไว้ใน best_error
S01=S(index,:);                      % และค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดจะถูกกำหนดให้เป็นคำตอบเริ่มต้นของหน่วย
                                     % ค้นหาที่สอง โดยเก็บไว้ใน S01

                                     % PM คู่คำตอบเริ่มต้นภายในปริภูมิการค้นหาทั้งสองแต่ละตัวจำนวน 5 ชุด
                                     % โดยอาศัยความสัมพันธ์  $(x_{h,i} - x_{l,i})\text{rand} + x_{l,i}$ , for  $i=1,\dots,5$ 
                                     % สำหรับหน่วยค้นหาที่สอง

for r=1:5
    S(r,:)=((xlimit2(1,:)-xlimit2(2,:)).*rand(1,N))+xlimit2(2,:);
end

for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2));           % ประเมินค่าคำตอบได้แล้วเก็บไว้ในตัวแปร ysim
    costvalue(k,1)=ysim;               % เก็บค่าฟังก์ชันวัตถุประสงค์ทั้ง 5 ค่าไว้ใน costvalue
end
[best_error,index]=min(costvalue);    % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดใน costvalue แล้วเก็บไว้ใน best_error
S02=S(index,:);                      % และค่าฟังก์ชันวัตถุประสงค์น้อยที่สุดจะถูกกำหนดให้เป็นคำตอบเริ่มต้นของหน่วย
                                     % ค้นหาที่สอง โดยเก็บไว้ใน S02

S00=[S01;S02];                       % เก็บพารามิเตอร์ทั้งของ S01 และ S02ไว้ใน S00

% Initial solutions evaluated
best_error01=obj(S01(1),S01(2));      % ประเมินค่าฟังก์ชันวัตถุประสงค์ของคำตอบ S01
best_error02=obj(S02(1),S02(2));      % ประเมินค่าฟังก์ชันวัตถุประสงค์ของคำตอบ S02

xlimit1=xlimit;                       % คำนวณค่าปริภูมิการค้นหาของพารามิเตอร์ให้กับทั้ง ATS1
xlimit2=xlimit;                       % คำนวณค่าปริภูมิการค้นหาของพารามิเตอร์ให้กับทั้ง ATS2

% Shared parameters
max_count=10000;                      % จำนวนรอบการค้นหาสูงสุด
Number_neighb1=5;
Number_neighb2=5;

                                     % จำนวนคำตอบข้างเคียงกำหนดไว้ 30 ค่า

count=0;                              % ตัวนับจำนวนรอบการค้นหาเริ่มต้นกำหนดให้เป็นศูนย์

% Initialize all individual parameters
% sub search space # 1
radius1=0.2;                          % รัศมีการค้นหาเริ่มต้นของหน่วยค้นหาที่หนึ่งกำหนดไว้ 0.2
best_neighbor01=S01;                  % เก็บคำตอบเริ่มต้น ATS1 ไว้ที่ best_neighbor01
overall_best_error1=best_error01;     % กำหนด overall_best_error1 เท่ากับ best_neighbor01
overall_best_neighbor1=best_neighbor01;
                                     % กำหนด overall_best_neighbor1 เท่ากับ best_neighbor01

n1=0;                                % กำหนดค่าเริ่มต้นของจำนวนการซ้ำของคำตอบ ของ ATS1
t1=0;                                % กำหนดค่าตัวเลื่อนสำหรับ local ของ ATS1

```

```

tt1=0; % กำหนดค่าตัวเลื่อนสำหรับ tabu_list ของ ATS1
ttt1=0; % กำหนดค่าตัวเลื่อนสำหรับ best_error_list ของ ATS1
n_back_tracking1=0; % กำหนดค่าเริ่มต้นของจำนวนการเรียกใช้กลไกย้อนรอยการค้นหาของ ATS1

% sub search space # 2
radius2=0.2; % รัศมีการค้นหาเริ่มต้นของหน่วยค้นหาที่สองกำหนดไว้ 0.2
best_neighbor02=S02; % เก็บค่าตอบเริ่มต้น ATS2 ไว้ที่ best_neighbor02
overall_best_error2=best_error02; % กำหนด overall_best_error2 เท่ากับ best_neighbor02
overall_best_neighbor2=best_neighbor02; % กำหนด overall_best_neighbor2 เท่ากับ best_neighbor02

n2=0; % กำหนดค่าเริ่มต้นของจำนวนการซ้ำของค่าคำตอบ ของ ATS2
t2=0; % กำหนดค่าตัวเลื่อนสำหรับ local ของ ATS2
tt2=0; % กำหนดค่าตัวเลื่อนสำหรับ tabu_list ของ ATS2
ttt2=0; % กำหนดค่าตัวเลื่อนสำหรับ best_error_list ของ ATS2
n_back_tracking2=0; % กำหนดค่าเริ่มต้นของจำนวนการเรียกใช้กลไกย้อนรอยการค้นหาของ ATS2
tic; % เริ่มต้นคำนวณเวลาที่ใช้ในการค้นหา

% Start ATS

t1=t1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local1 ของ ATS1
tt1=tt1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1 ของ ATS1
ttt1=ttt1+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list1 ของ ATS1

t2=t2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local2 ของ ATS2
tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2 ของ ATS2
ttt2=ttt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list2 ของ ATS2

disp([count overall_best_error1 overall_best_error2]);
% แสดงค่า count, overall_best_neighbor1 และ overall_best_neighbor/

% Sub search space # 1
local1(t1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local1
local1(t1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local1
local1(t1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ local1
tabu_list1(tt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list1
tabu_list1(tt1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list 1
tabu_list1(tt1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ tabu_list 1
best_error_list1(ttt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list1
best_error_list1(ttt1,2:3)=best_neighbor01; % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list1
best_error_list1(ttt1,4)=best_error01; % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ best_error_list1

% Sub search space # 2
local2(t2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local2
local2(t2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local2
local2(t2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ local2

```

```

tabu_list2(tt2,1)=count;          % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list2
tabu_list2(tt2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list 2
tabu_list2(tt2,4)=best_error02;   % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ tabu_list 2
best_error_list2(ttt2,1)=count;   % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list2
best_error_list2(ttt2,2:3)=best_neighbor02;
                                % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list2
best_error_list2(ttt2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ best_error_list2

% Path Enabling
flag1=1;                        % กำหนดค่าเริ่มต้น flag1 ของ ATS1 เท่ากับหนึ่งเพื่อให้ทำงานได้
flag2=1;                        % กำหนดค่าเริ่มต้น flag2 ของ ATS/2 เท่ากับหนึ่งเพื่อให้ทำงานได้

% Start ATS
while (count<max_count)         % วนรอบการค้นหา
count=count+1;                  % ปรับเพิ่ม count ขึ้นจากเดิมหนึ่งค่า

% Path 1
if (flag1==1)                   % การคงอยู่ในวงรอบการค้นหาของ ATS1 เกิดขึ้นภายใต้เงื่อนไข flag1==1 เท่านั้น
S101=random_neigh(Number_neighb,radius1,xlimit1,S01);
                                % เรียกใช้โปรแกรม random_neigh.m เพื่อสุ่มค่าใกล้เคียงรอบคำตอบ S01
                                % อินพุตของฟังก์ชันนี้คือ Number_neighb, radius1, xlimit1 และ S01
                                % เอาท์พุตคือค่าใกล้เคียง S101 จำนวนเท่ากับ Number_neighb (30 ตัว)
[best_error101,best_neighbor101,best_error01,best_neighbor01]=objective(S101,best_error01,S01);
                                % เรียกใช้โปรแกรม objective.m เพื่อประเมินค่าใกล้เคียงที่สุ่มได้
                                % อินพุตของฟังก์ชันนี้คือ S101, best_error01, และ S01
                                % เอาท์พุตคือ best_error101, best_neighbor101, best_error01 และ
                                % best_neighbor01

if (best_error101-best_error01)>1e-18
n1=n1+1;
else
n1=0;
end
                                % ตรวจสอบการซ้ำของคำตอบโดยเปรียบเทียบระหว่าง best_error101 และ
                                % best_error01 ถ้ามีการซ้ำให้ปรับเพิ่มค่า n1

tt1=tt1+1;                      % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1 ของ ATS1
tabu_list1(tt1,1)=count;        % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list1

tabu_list1(tt1,2:3)=best_neighbor101;
                                % เก็บค่า best_neighbor101 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list1
tabu_list1(tt1,4)=best_error101; % เก็บค่า best_error101 ไว้ในคอลัมน์ที่ 4 ของ tabu_list1
ttt1= ttt1+1;                   % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list1 ของ ATS1
best_error_list1(ttt1,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list1
best_error_list1(ttt1,2:3)=best_neighbor01;
                                % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list1
best_error_list1(ttt1,4)=best_error01;
                                % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ best_error_list1

```

```

% Start AR mechanism                                (กำหนดให้ 2 เงื่อนไขเพื่อเป็นตัวอย่าง)

if best_error01<90                                % เงื่อนไขที่ 1) ถ้า best_error01< 90 แล้ว   radius1=0.1
radius1=0.1;
end
if best_error01<70                                % เงื่อนไขที่ 2) ถ้า best_error01< 70 แล้ว   radius1=0.05
radius1=0.05;
end
if (overall_best_error1<65)                        % ตรวจสอบเกณฑ์ยุติการค้นหา เมื่อ best_error01< 65 แล้ว
t1=t1+1;                                           % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local1
tt1=tt1+1;                                         % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list1
local1(t1,1)=count;                               % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local1
local1(t1,2:3)=best_neighbor01;                   % เก็บค่า best_neighbor01 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local1
local1(t1,4)=best_error01;                       % เก็บค่า best_error01 ไว้ในคอลัมน์ที่ 4 ของ local1
disp('Path 1 Strikes');                           % แสดงผลว่า 'Path 1 Strikes'
break;                                             % ยุติการค้นหา
end

% Start BT mechanism                                % เมื่อจำนวนการซ้ำของค่าคำตอบเท่ากับ 5 (เพื่อเป็นตัวอย่าง)

% sub search space # 1
if n1>=5
n_back_tracking1=n_back_tracking1+1;
% เพิ่มค่า n_back_tracking1
TEMP1=tabu_list1(count-3:count+1,:);
% เก็บค่าพารามิเตอร์จำนวน 5 ชุดสุดท้ายก่อนทำการจัดลำดับ ไว้ใน TEMP1
[ MAX1,INDEX1 ] = max(TEMP1(:,4));
% หาค่าฟังก์ชันวัตถุประสงค์สูงสุดและลำดับที่ใน TEMP1
RANK1(5,:) = TEMP1(INDEX1,:);
% เก็บค่าพารามิเตอร์ที่มีฟังก์ชันวัตถุประสงค์สูงสุดไว้ใน RANK1
neighbor1=RANK1(5,2:3);
% เก็บค่า RANK1 ไว้ใน neighbor1
S01=neighbor1;
% แทนค่า S01 ด้วย neighbor1
best_error01=RANK1(5,4);
% แทนค่า best_error01 ด้วย RANK1
n1=0;
% ปรับค่า n1=0
else
% ไม่เช่นนั้นแล้ว
S01=best_neighbor01;
% แทนค่า S01 ด้วย best_neighbor01
best_error01=best_error01;
% แทนค่า best_error01 ด้วย best_error01
end

% Path # 1
if overall_best_error1>best_error01
overall_best_error1=best_error01;
overall_best_neighbor1=best_neighbor01;
end
% ถ้า overall_best_error1 > best_error01 แล้ว
% overall_best_error1=best_error01
% overall_best_neighbor1=best_neighbor01

count1=count;
% แทนค่า count1 ด้วย count1
end
% สิ้นสุดวนรอบการค้นหาของ ATS1

```

```

% Path 2
if (flag2==1) % การคงอยู่ในวงรอบการค้นหาของ ATS2 เกิดขึ้นภายใต้เงื่อนไข flag2==1 เท่านั้น

% sub search space # 2
S102=random_neigh(Number_neighb,radius2,xlimit2,S02);
% เรียกใช้ฟังก์ชัน random_neigh เพื่อสุ่มค่าใกล้เคียงรอบคำตอบ S02
% อินพุตของฟังก์ชันนี้คือ Number_neighb, radius2, xlimit2 และ S02
% เอาท์พุตคือค่าใกล้เคียง S102 จำนวนเท่ากับ Number_neighb (30 ตัว)
[best_error102,best_neighbor102,best_error02,best_neighbor02]=objective(S102,best_error02,S02);
% เรียกใช้ฟังก์ชัน objective เพื่อประเมินค่าใกล้เคียงที่สุ่มได้
% อินพุตของฟังก์ชันนี้คือ S102, best_error02, และ S02
% เอาท์พุตคือ best_error102, best_neighbor102, best_error02 และ
% best_neighbor02

% Path # 2
if (best_error102-best_error02)>1e-18
n2=n2+1;
else
n2=0;
end % ตรวจสอบการซ้ำของค่าคำตอบโดยเปรียบเทียบระหว่าง best_error102 และ
% best_error02 ถ้ามีการซ้ำให้ปรับเพิ่มค่า n2

tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2 ของ ATS2
tabu_list2(tt2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ tabu_list2
tabu_list2(tt2,2:3)=best_neighbor102; % เก็บค่า best_neighbor102 ไว้ในคอลัมน์ที่ 2 และ 3 ของ tabu_list2
tabu_list2(tt2,4)=best_error102; % เก็บค่า best_error102 ไว้ในคอลัมน์ที่ 4 ของ tabu_list2
ttt2= ttt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ best_error_list2 ของ ATS2
best_error_list2(ttt2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ best_error_list2

best_error_list2(ttt2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ best_error_list2
best_error_list2(ttt2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ best_error_list2

% Start AR (กำหนดให้มี 2 เงื่อนไขเพื่อเป็นตัวอย่าง)
if best_error02<90 % เงื่อนไขที่ 1) ถ้า best_error02< 90 แล้ว radius2=0.1
radius2=0.1;
end
if best_error02<70 % เงื่อนไขที่ 2) ถ้า best_error02< 70 แล้ว radius2=0.05
radius2=0.05;
end
if (overall_best_error2<65) % ตรวจสอบเกณฑ์ยุติการค้นหา เมื่อ best_error02< 65 แล้ว
t2=t2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ local2
tt2=tt2+1; % ปรับเพิ่มค่าตัวเลื่อนสำหรับ tabu_list2
local2(t2,1)=count; % เก็บค่า count ไว้ในคอลัมน์ที่ 1 ของ local2
local2(t2,2:3)=best_neighbor02; % เก็บค่า best_neighbor02 ไว้ในคอลัมน์ที่ 2 และ 3 ของ local2
local2(t2,4)=best_error02; % เก็บค่า best_error02 ไว้ในคอลัมน์ที่ 4 ของ local2

```

```

disp('Path 2 Strikes');          % แสดงผลว่า 'Path 2 Strikes'
break;                          % ยุติการค้นหา
end

% Start BT                      % เรียกใช้ BT
if n2>=5
n_back_tracking2=n_back_tracking2+1;
                                % เพิ่มค่า n_back_tracking2
TEMP2=tabu_list2(count-3:count+1,:);
                                % เก็บค่าพารามิเตอร์จำนวน 5 ชุดสุดท้ายก่อนทำการจัดลำดับ ไว้ใน TEMP2
[ MAX2,INDEX2] = max(TEMP2(:,4));
                                % หาค่าฟังก์ชันวัตถุประสงค์สูงสุดและลำดับที่ใน TEMP2
RANK2(5,:) = TEMP2(INDEX2,:); % เก็บค่า RANK2 ไว้ใน neighbor2
neighbor2=RANK2(5,2:3);        % เก็บค่า RANK2 ไว้ใน neighbor2
S02=neighbor2;                  % แทนค่า S02 ด้วย neighbor2
best_error02=RANK2(5,4);        % แทนค่า best_error02 ด้วย RANK2
n2=0;                          % ปรับค่า n2=0
else                            % ไม่เช่นนั้นแล้ว
S02=best_neighbor02;            % แทนค่า S02 ด้วย best_neighbor02
best_error02=best_error02; % แทนค่า best_error02 ด้วย best_error02
end

% Updating Path # 2
if overall_best_error2>best_error02
overall_best_error2=best_error02;
overall_best_neighbor2=best_neighbor02;
end
                                % ถ้า overall_best_error2 > best_error02 แล้ว
                                % overall_best_error2=best_error02
                                % overall_best_neighbor2=best_neighbor02

count2=count;                  % แทนค่า count1 ด้วย count2
end                             % สิ้นสุดวนรอบการค้นหาของ ATS2

% Start Discarding Mechanism
if count==50                   % เนื่องจากการเรียกใช้ DM กำหนดตามรอบการค้นหา
                                % (เลือกรอบการค้นหาที่ 50 เพื่อเป็นตัวอย่าง)
    if (overall_best_error2<overall_best_error1)
                                % เปรียบเทียบค่าฟังก์ชันวัตถุประสงค์ปัจจุบันของ ATS1 และ ATS2
                                % เลือกกำหนดหน่วยค้นหาที่มีค่าฟังก์ชันวัตถุประสงค์สูงกว่าออกจากวงรอบการค้นหา
        flag1=0;                % ด้วยการกำหนดค่า flag = 0
    else
        flag2=0;
    end
end
end

disp([count overall_best_error1 overall_best_error2]);
                                % แสดงค่า count, overall_best_error1, overall_best_error2

end
time=toc                       % ยุติการคำนวณการค้นหาของ DATS
return

```

โปรแกรม obj.m

```
function J=obj(Kd,Kp) % โปรแกรม obj คือโปรแกรมคำนวณค่าฟังก์ชันวัตถุประสงค์
% อินพุตของโปรแกรมคือพารามิเตอร์ Kd และ Kp
% เอาต์พุตคือค่าฟังก์ชันวัตถุประสงค์ที่ได้จากคำตอบ (Kd, Kp)

% # 1 G : Main Plant
G=tf([0.2802e5 2.3057e5],[1.5 94.4284 7.3078e3 1.1831e5 5.3616e5]);
% สร้างฟังก์ชันถ่ายโอนของพลานต์หลัก

% # 2 Gd : Additional Plant
Gd=tf([0.8024e5 6.7103e5],[1.3 53.2495 1.9425e3 2.6928e4 1.1617e5]);
% สร้างฟังก์ชันถ่ายโอนของพลานต์รอง

% # 3 Gdc = Bc/Ac : Compensator:
b1=-1.2036e5;
b2=-8.5835e6;
b3=-6.4974e8;
b4=-1.4397e10;
b5=-1.2241e11;
b6=-3.5978e11;
a1=3.6426e4;
a2=1.7918e6;
a3=6.6708e7;
a4=1.2024e9;
a5=9.4644e9;
a6=2.6788e10;
Gdc=tf([b1 b2 b3 b4 b5 b6],[a1 a2 a3 a4 a5 a6]);
% สร้างฟังก์ชันถ่ายโอนของตัวควบคุม Gdc

% # 4 Gff = T/R : Feedforward module
r1=1;
r2=155.3;
r3=3.727e3;
r4=2.0714e4;
t1=103;
t2=1.8542e4;
t3=8.34385e5;
Gff=tf([t1 t2 t3],[r1 r2 r3 r4]);
% สร้างฟังก์ชันถ่ายโอนของตัวควบคุม Gff

% # 5 Gfb = S/R : Feedback module
s1=-5.181;
s2=49.98;
s3=5.294e3;
s4=4.8142e4;
Gfb=tf([s1 s2 s3 s4],[r1 r2 r3 r4]);
% สร้างฟังก์ชันถ่ายโอนของตัวควบคุม Gfb

% # 6 Gdl = S/R : PD controller
Gdl=tf([Kd Kp],[1 0]);
% สร้างฟังก์ชันถ่ายโอนของตัวควบคุม Gdl

% The original IRS system
Girs0=(Gd+G*Gdc+G*Gff)/(1+G*Gfb);
% สร้างฟังก์ชันถ่ายโอนวงรอบปิดของระบบดั้งเดิม Girs0

% IRS system
Girs=(Gd+G*Gdc+G*Gdl*Gff)/(1+G*Gfb);
% สร้างฟังก์ชันถ่ายโอนวงรอบปิดของระบบใหม่ Girs (พร้อม Gdl)

a=real(pole(Girs)); % หาค่ารากของ Girs
```

```

if a<0                                % เงื่อนไขสำคัญ Girs ต้องมีเสถียรภาพโดยพิจารณาจากโพลทุกตัวต้องเป็นบวก
[y,t]=step(Girs);                    % หาผลตอบสนองแบบขั้นบันไดของ Girs ด้วยคำสั่ง step
                                    % เอาที่พหุคูณ y กับ t

if y(length(y))>8.66 & y(length(y))<8.67
                                    % เงื่อนไขของ y ต้องมีค่าคงตัวอยู่ระหว่าง 8.66 กับ 8.67

% Find rise time
i=find(y>0.1*(y(length(y))-y(1))&y<0.9*(y(length(y))-y(1)));
tr=t(i(length(i)))-t(i(1));          % หาค่าเวลาไต่ขึ้นจาก y ช่วง 10-90%
trj=(tr*100)/0.23096393759434;       % แปลงค่าเวลาไต่ขึ้นที่ได้เป็นค่าหนึ่งหน่วย (normalization)
                                    % ด้วยค่าเวลาไต่ขึ้นดั้งเดิม tr =0.23096393759434
                                    % แล้วทำให้เป็นเปอร์เซ็นต์

% Find overshoot
Mp=100*(max(y)-y(length(y)))/y(length(y));
Mpj=Mp;                             % หาค่าการพุ่งเกิน

% Find settling time
for i=1:length(y)
    y(length(y)+1)=y(length(y));
    dy(i)=y(i+1)-y(i);
end
i=find(~(dy<1e-3));                  % พิจารณา ณ เวลาที่ y อยู่นิ่งหรือเปลี่ยนแปลงน้อยกว่า  $1 \times 10^{-3}$ 
ts=t(i(length(i)))-t(i(1));
tsj=(ts*100)/(0.23096393759434*5); % แปลงค่าเวลาดังตัวที่ได้เป็นค่าหนึ่งหน่วย (normalization)
                                    % ด้วยค่า 5 เท่าของเวลาไต่ขึ้นดั้งเดิม แล้วทำให้เป็นเปอร์เซ็นต์

% Set constraints
% For rise time
if trj<100                           % กำหนดเงื่อนไข tr ใหม่ต้องลดลงกว่าเดิม นั่นคือ trj ต้องน้อยกว่า 100%
    trj=trj;
else
    trj=1000;                        % ไม่เช่นนั้นจะปรับโทษให้ trj=1000
end

% For percent of overshoot
if Mpj<0.001 % Need no Overshoot    % กำหนดเงื่อนไขต้องไม่มีการพุ่งเกินนั่นคือ Mpj<0.001
    Mpj=1;
else
    Mpj=1000;                        % ไม่เช่นนั้นจะปรับโทษให้ Mpj=1000
end

% For settling time
if tsj<5*trj                         % กำหนดเงื่อนไข ts ใหม่ต้องลดลงกว่าเดิม นั่นคือ tsj 5*trj
    tsj=1;
else
    tsj=1000;                        % ไม่เช่นนั้นจะปรับโทษให้ tsj=1000
end

% Set objective function
J=trj+Mpj+tsj;                      % สร้างฟังก์ชันวัตถุประสงค์ที่รวมเอาคุณสมบัติที่ต้องการคือ tr ts และ Mp
                                    % ไว้ในค่า J

```

```

else                                     % ไม่เช่นนั้นปรับโทษให้ J = 1000
    J=1000;
end
else                                     % ไม่เช่นนั้นปรับโทษให้ J = 1000
    J=1000;
end
return

```

โปรแกรม objective.m

```

function [best_error1,best_neighbor1,best_error,best_neighbor]=objective(S1,best_error,S0)

% โปรแกรม objective เป็น โปรแกรมประเมินค่าใกล้เคียงที่สุ่มได้รอบ S0
% อินพุตของ โปรแกรมนี้คือ S1, best_error, และ S0
% เอาต์พุตคือ best_error1, best_neighbor1,
% best_error, และ best_neighbor

error=[];
% ตัวแปร error สำหรับรองรับค่าฟังก์ชันวัตถุประสงค์
% ของค่าใกล้เคียงที่ได้จากการสุ่ม

for k=1:size(S1,1)
    ysim=obj(S1(k,1),S1(k,2)); % ประเมินค่าใกล้เคียงใน S1 ด้วยฟังก์ชันวัตถุประสงค์ในโปรแกรม obj
    error(k,1)=ysim;           % เก็บค่าฟังก์ชันวัตถุประสงค์ของค่าใกล้เคียงทั้งหมด
end
[best_error1,index]=min(error); % หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดในบรรดาค่าใกล้เคียง
                                % และเก็บไว้ใน best_error1
best_neighbor1=S1(index,:);     % เก็บค่าพารามิเตอร์ของค่าใกล้เคียงที่ทำให้
                                % ค่าฟังก์ชันวัตถุประสงค์มีค่าน้อยที่สุดไว้ใน best_neighbor1

if best_error1<best_error
    % ปรับค่า best_error
    best_error=best_error1;     % ถ้า best_error1 < best_error
                                % ให้แทนค่า best_error ด้วย best_error1

    best_neighbor=S1(index,:);  % และแทนค่าพารามิเตอร์ best_neighbor ด้วย S1(index,:)
else
    % ไม่เช่นนั้นแล้ว
    best_neighbor=S0;           % แทนค่า best_neighbor ด้วย S0
end
return

```

โปรแกรม rand1.m

```

function x=rand1(a,b)

% โปรแกรม rand1 เป็น โปรแกรมสุ่มค่าพารามิเตอร์ในช่วงที่กำหนด
% อินพุตของ โปรแกรมคือขอบเขตบนของพารามิเตอร์ (b) และ
% ขอบเขตล่างของพารามิเตอร์ (a)
% โดยอาศัยความสัมพันธ์ (b-a)rand + a
% เอาต์พุตของ โปรแกรมคือผลของการสุ่มค่าพารามิเตอร์ในช่วง a ถึง b

x=a+rand*(b-a);
return

```

```

โปรแกรม random_
neigh.m
function S1=random_neigh(Number_neighb,radius,xlimit,S0)

% โปรแกรม random_neigh เป็นโปรแกรมสุ่มค่าใกล้เคียงรอบค่าตอบ S0
% อินพุตของโปรแกรมนี้อือ Number_neighb, radius, xlimit, และ S0
% เอาต์พุตคือค่าใกล้เคียง (S1) จำนวนเท่ากับ Number_neighb

for u=1:Number_neighb
for k=1:size(xlimit,2)
S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));

% ทำการสุ่มค่าใกล้เคียงโดยอาศัยความสัมพันธ
%  $S_{1,i} = S_0 + (\text{radius} \cdot \text{rand1}_i(-1,1) \cdot (x_{h,i} - x_{l,i}))$ ,
% for  $i = 1, \dots, \text{Number\_neighb}$ 
% เรียกใช้โปรแกรม rand1 โดยให้ทำการสุ่มค่าในช่วง -1 ถึง 1
% นั่นหมายความว่าค่าสุ่มมีพื้นที่อยู่ในช่วง  $S_0 + ((\text{radius}) \cdot (x_{h,i} - x_{l,i}))$ 

while ( S1(u,k)>xlimit(1,k) | S1(u,k)<xlimit(2,k) )

% ค่าใกล้เคียงที่สุ่มได้จะต้องอยู่ภายในขอบเขตบนและขอบเขตล่าง
% ของค่าพารามิเตอร์ ถ้าค่าใกล้เคียงที่สุ่มได้
% มีค่าเกินขอบเขตบนและขอบเขตล่างที่กำหนด
% ให้ทำการสุ่มค่าใกล้เคียงใหม่

S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));
end
end
end
return

```



ค.1 โปรแกรมระบุเอกลักษณ์ระบบ CPP ด้วย MATS

โปรแกรมการค้นหาลักษณะแบบตามเชิงปรับตัวหลายวิธีสำหรับการระบุเอกลักษณ์ระบบ CPP ในงานวิจัยวิทยานิพนธ์ฉบับนี้ มีรายละเอียดดังต่อไปนี้

ค.1.1 โปรแกรม MATS ที่พัฒนาด้วย MATLAB™

```
% โปรแกรมนี้เป็นโปรแกรมระบุเอกลักษณ์สำหรับระบบ CPP
% เป็นการออกแบบด้วยการค้นหาแบบตามเชิงปรับตัวหลายวิธี
% โปรแกรมได้รับการพัฒนาด้วย MATLAB™
% พัฒนา โดยจักรกฤษณ์ เคลือบวัง (2009)
% สาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีสุรนารี

โปรแกรม MATS ชนิด 15 หน่วยค้นหา หรือ MATS15.m
function [time,overall_best_error, overall_best_neighbor,count]=MATS15(N)
    % โปรแกรมหลักมี N เป็นอินพุตของโปรแกรม
    % N คือจำนวนพารามิเตอร์ที่ต้องการค้นหา (N = 15)
    % เอาต์พุตของโปรแกรมคือ
    % ค่าพารามิเตอร์ที่ดีที่สุดหรือค่าคำตอบวงกว้าง (overall_best_neighbor)
    % ค่าฟังก์ชันวัตถุประสงค์ที่ดีที่สุด (overall_best_error)
    % ค่าจำนวนรอบการค้นหาทั้งหมด (count)

    TC=3.1895; % เกณฑ์ยุติการค้นหาด้วยค่าฟังก์ชันวัตถุประสงค์

    xlimit=[ 5 5 5 2 2 2 6 5 10 1 1 2 1 1 5;...
            -5 -5 -1 -2 -2 -2 -6 -5 -10 -1 -1 -2 -1 -1 -5];
    % กำหนดปริภูมิการค้นหาของพารามิเตอร์ทั้ง 15 ตัว
    % แถวที่ 1 ของ xlimit คือขอบเขตบนของพารามิเตอร์
    % แถวที่ 2 ของ xlimit คือขอบเขตล่างของพารามิเตอร์

    xlimit1=[5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            -5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit2=[0.1 5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 -5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit3=[0.1 0.1 5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 -1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit4=[0.1 0.1 0.1 2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 -2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit5=[0.1 0.1 0.1 0.1 2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 -2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit6=[0.1 0.1 0.1 0.1 0.1 2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 -2 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit7=[0.1 0.1 0.1 0.1 0.1 0.1 6 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 0.1 -6 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit8=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 5 0.1 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 0.1 0.1 -5 0.1 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit9=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 10 0.1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -10 0.1 0.1 0.1 0.1 0.1 0.1];
    xlimit10=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -1 0.1 0.1 0.1 0.1 0.1];
    xlimit11=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1 0.1 0.1 0.1 0.1 0.1;...
            0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -1 0.1 0.1 0.1 0.1 0.1];
```

```

xlimit12=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 2 0.1 0.1 0.1;...
          0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -2 0.1 0.1 0.1];
xlimit13=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1 0.1 0.1;...
          0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -1 0.1 0.1];
xlimit14=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1 0.1;...
          0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -1 0.1];
xlimit15=[0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 5;...
          0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 -5];
          % กำหนดปริภูมิการค้นหาลดเหลือ 15 ปริภูมิ

for r=1:5
    S(r,:)=((xlimit1(1,:)-xlimit1(2,:)).*rand(1,N))+xlimit1(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
            S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S01=S(index,:);
          % สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 1 ในปริภูมิการค้นหาลดเหลือที่ 1

for r=1:5
    S(r,:)=((xlimit2(1,:)-xlimit2(2,:)).*rand(1,N))+xlimit2(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
            S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S02=S(index,:);
          % สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 2 ในปริภูมิการค้นหาลดเหลือที่ 2

for r=1:5
    S(r,:)=((xlimit3(1,:)-xlimit3(2,:)).*rand(1,N))+xlimit3(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
            S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S03=S(index,:);
          % สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 3 ในปริภูมิการค้นหาลดเหลือที่ 3

for r=1:5
    S(r,:)=((xlimit4(1,:)-xlimit4(2,:)).*rand(1,N))+xlimit4(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
            S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S04=S(index,:);
          % สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 4 ในปริภูมิการค้นหาลดเหลือที่ 4

```

```

for r=1:5
    S(r,:)=((xlimit5(1,:)-xlimit5(2,:)).*rand(1,N))+xlimit5(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S05=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 5 ในปริภูมิการค้นหาย่อยที่ 5

for r=1:5
    S(r,:)=((xlimit6(1,:)-xlimit6(2,:)).*rand(1,N))+xlimit6(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S06=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 6 ในปริภูมิการค้นหาย่อยที่ 6

for r=1:5
    S(r,:)=((xlimit7(1,:)-xlimit7(2,:)).*rand(1,N))+xlimit7(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S07=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 7 ในปริภูมิการค้นหาย่อยที่ 7

for r=1:5
    S(r,:)=((xlimit8(1,:)-xlimit8(2,:)).*rand(1,N))+xlimit8(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S08=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 8 ในปริภูมิการค้นหาย่อยที่ 8

for r=1:5
    S(r,:)=((xlimit9(1,:)-xlimit9(2,:)).*rand(1,N))+xlimit9(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S09=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 9 ในปริภูมิการค้นหาย่อยที่ 9

```

```

for r=1:5
    S(r,:)=((xlimit10(1,:)-xlimit10(2,:)).*rand(1,N))+xlimit10(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S10=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 10 ในปริภูมิการค้นหาย่อยที่ 10

for r=1:5
    S(r,:)=((xlimit11(1,:)-xlimit11(2,:)).*rand(1,N))+xlimit11(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S11=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 11 ในปริภูมิการค้นหาย่อยที่ 11

for r=1:5
    S(r,:)=((xlimit12(1,:)-xlimit12(2,:)).*rand(1,N))+xlimit12(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S12=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 12 ในปริภูมิการค้นหาย่อยที่ 12

for r=1:5
    S(r,:)=((xlimit13(1,:)-xlimit13(2,:)).*rand(1,N))+xlimit13(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S13=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 13 ในปริภูมิการค้นหาย่อยที่ 13

for r=1:5
    S(r,:)=((xlimit14(1,:)-xlimit14(2,:)).*rand(1,N))+xlimit14(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S14=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 14 ในปริภูมิการค้นหาย่อยที่ 14

```

```

for r=1:5
    S(r,:)=((xlimit15(1,:)-xlimit15(2,:)).*rand(1,N))+xlimit15(2,:);
end
for k=1:size(S,1)
    ysim=obj(S(k,1),S(k,2),S(k,3),S(k,4),S(k,5),S(k,6),S(k,7),S(k,8),S(k,9),S(k,10),S(k,11),S(k,12),S(k,13),S(k,14),
        S(k,15));
    costvalue(k,1)=ysim;
end
[best_error,index]=min(costvalue);
S15=S(index,:);

% สุ่มเลือกคำตอบเริ่มต้นสำหรับหน่วยค้นหาที่ 15 ในปฏิบัติการค้นหาที่ย่อยที่ 15

xlimit1=xlimit;
xlimit2=xlimit;
xlimit3=xlimit;
xlimit4=xlimit;
xlimit5=xlimit;
xlimit6=xlimit;
xlimit7=xlimit;
xlimit8=xlimit;
xlimit9=xlimit;
xlimit10=xlimit;
xlimit11=xlimit;
xlimit12=xlimit;
xlimit13=xlimit;
xlimit14=xlimit;
xlimit15=xlimit;

% กำหนดปฏิบัติการค้นหาที่ย่อยทั้งหมดให้เท่ากันกับปฏิบัติการค้นหาหรรวม
ภายหลัง % เสร็จสิ้น PM

S00=[S01;S02;S03;S04;S05;S06;S07;S08;S09;S10;S11;S12;S13;S14;S15];
% เก็บคำตอบเริ่มต้นทั้ง 15 ตัวไว้ใน S00

best_neighbor01=S01;
best_neighbor02=S02;
best_neighbor03=S03;
best_neighbor04=S04;
best_neighbor05=S05;
best_neighbor06=S06;
best_neighbor07=S07;
best_neighbor08=S08;
best_neighbor09=S09;
best_neighbor10=S10;
best_neighbor11=S11;
best_neighbor12=S12;
best_neighbor13=S13;
best_neighbor14=S14;
best_neighbor15=S15;

% เก็บคำตอบเริ่มต้นแต่ละค่าไว้ในชื่อ best_neighbor

best_error01=obj(S01(1),S01(2),S01(3),S01(4),S01(5),S01(6),S01(7),S01(8),S01(9),S01(10),S01(11),S01(12),
    S01(13),S01(14),S01(15));
best_error02=obj(S02(1),S02(2),S02(3),S02(4),S02(5),S02(6),S02(7),S02(8),S02(9),S02(10),S02(11),S02(12),
    S02(13),S02(14),S02(15));
best_error03=obj(S03(1),S03(2),S03(3),S03(4),S03(5),S03(6),S03(7),S03(8),S03(9),S03(10),S03(11),S03(12),
    S03(13),S03(14),S03(15));
best_error04=obj(S04(1),S04(2),S04(3),S04(4),S04(5),S04(6),S04(7),S04(8),S04(9),S04(10),S04(11),S04(12),
    S04(13),S04(14),S04(15));
best_error05=obj(S05(1),S05(2),S05(3),S05(4),S05(5),S05(6),S05(7),S05(8),S05(9),S05(10),S05(11),S05(12),
    S05(13),S05(14),S05(15));

```

```

best_error06=obj(S06(1),S06(2),S06(3),S06(4),S06(5),S06(6),S06(7),S06(8),S06(9),S06(10),S06(11),S06(12),
S06(13),S06(14),S06(15));
best_error07=obj(S07(1),S07(2),S07(3),S07(4),S07(5),S07(6),S07(7),S07(8),S07(9),S07(10),S07(11),S07(12),
S07(13),S07(14),S07(15));
best_error08=obj(S08(1),S08(2),S08(3),S08(4),S08(5),S08(6),S08(7),S08(8),S08(9),S08(10),S08(11),S08(12),
S08(13),S08(14),S08(15));
best_error09=obj(S09(1),S09(2),S09(3),S09(4),S09(5),S09(6),S09(7),S09(8),S09(9),S09(10),S09(11),S09(12),
S09(13),S09(14),S09(15));
best_error10=obj(S10(1),S10(2),S10(3),S10(4),S10(5),S10(6),S10(7),S10(8),S10(9),S10(10),S10(11),S10(12),
S10(13),S10(14),S10(15));
best_error11=obj(S11(1),S11(2),S11(3),S11(4),S11(5),S11(6),S11(7),S11(8),S11(9),S11(10),S11(11),S11(12),
S11(13),S11(14),S11(15));
best_error12=obj(S12(1),S12(2),S12(3),S12(4),S12(5),S12(6),S12(7),S12(8),S12(9),S12(10),S12(11),S12(12),
S12(13),S12(14),S12(15));
best_error13=obj(S13(1),S13(2),S13(3),S13(4),S13(5),S13(6),S13(7),S13(8),S13(9),S13(10),S13(11),S13(12),
S13(13),S13(14),S13(15));
best_error14=obj(S14(1),S14(2),S14(3),S14(4),S14(5),S14(6),S14(7),S14(8),S14(9),S14(10),S14(11),S14(12),
S14(13),S14(14),S14(15));
best_error15=obj(S15(1),S15(2),S15(3),S15(4),S15(5),S15(6),S15(7),S15(8),S15(9),S15(10),S15(11),S15(12),
S15(13),S15(14),S15(15));

% หากค่าฟังก์ชันวัตถุประสงค์ของคำตอบเริ่มต้นแต่ละตัวแล้วเก็บแยกไว้ในตัวแปร
% ชื่อ best_error แต่ละตัว

[Y,I]=min([best_error01 best_error02 best_error03 best_error04 best_error05 best_error06 best_error07
best_error08 best_error09 best_error10 best_error11 best_error12 best_error13 best_error14 best_error15]);

% หากค่าฟังก์ชันวัตถุประสงค์ของคำตอบเริ่มต้นที่น้อยที่สุด

overall_best_error=Y; % กำหนดค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดที่ได้ให้เป็น overall_best_error

if I==1
    overall_best_neighbor=S01;
elseif I==2
    overall_best_neighbor=S02;
elseif I==3
    overall_best_neighbor=S03;
elseif I==4
    overall_best_neighbor=S04;
elseif I==5
    overall_best_neighbor=S05;
elseif I==6
    overall_best_neighbor=S06;
elseif I==7
    overall_best_neighbor=S07;
elseif I==8
    overall_best_neighbor=S08;
elseif I==9
    overall_best_neighbor=S09;
elseif I==10
    overall_best_neighbor=S10;
elseif I==11
    overall_best_neighbor=S11;
elseif I==12
    overall_best_neighbor=S12;
elseif I==13
    overall_best_neighbor=S13;
elseif I==14
    overall_best_neighbor=S14;
elseif I==15
    overall_best_neighbor=S15;
end

% พิกัดตำแหน่ง overall_best_neighbor ของค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุด

```

```

% Shared parameters
max_count=10000;          % กำหนดการค้นหาค้นหาด้วยจำนวนรอบการค้นหาสูงสุด
count=0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Initialize all individual parameters %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% sub search space # 1
overall_best_error1=best_error01; % กำหนดให้ overall_best_error1 มีค่าเท่ากับ best_error01
overall_best_neighbor1=best_neighbor01;
                                     % กำหนดให้ overall_best_neighbor1 มีค่าเท่ากับ best_neighbor01

radius1=radius;                % รัศมีการค้นหา radius1 เท่ากับ radius
n1=0;
t1=0;
tt1=0;
ttt1=0;
n_back_tracking1=0;
count1=count;

% sub search space # 2
overall_best_error2=best_error02;
overall_best_neighbor2=best_neighbor02;
radius2=radius;
n2=0;
t2=0;
tt2=0;
ttt2=0;
n_back_tracking2=0;
count2=count;
% sub search space # 3
overall_best_error3=best_error03;
overall_best_neighbor3=best_neighbor03;
radius3=radius;
n3=0;
t3=0;
tt3=0;
ttt3=0;
n_back_tracking3=0;
count3=count;
% sub search space # 4
overall_best_error4=best_error04;
overall_best_neighbor4=best_neighbor04;
radius4=radius;
n4=0;
t4=0;
tt4=0;
ttt4=0;
n_back_tracking4=0;
count4=count;

% sub search space # 5
overall_best_error5=best_error05;
overall_best_neighbor5=best_neighbor05;
radius5=radius;
n5=0;
t5=0;
tt5=0;
ttt5=0;
n_back_tracking5=0;

```

```

count5=count;

% sub search space # 6
overall_best_error6=best_error06;
overall_best_neighbor6=best_neighbor06;
radius6=radius;
n6=0;
t6=0;
tt6=0;
ttt6=0;
n_back_tracking6=0;
count6=count;

% sub search space # 7
overall_best_error7=best_error07;
overall_best_neighbor7=best_neighbor07;
radius7=radius;
n7=0;
t7=0;
tt7=0;
ttt7=0;
n_back_tracking7=0;
count7=count;

% sub search space # 8
overall_best_error8=best_error08;
overall_best_neighbor8=best_neighbor08;
radius8=radius;
n8=0;
t8=0;
tt8=0;
ttt8=0;
n_back_tracking8=0;
count8=count;

% sub search space # 9
overall_best_error9=best_error09;
overall_best_neighbor9=best_neighbor09;
radius9=radius;
n9=0;
t9=0;
tt9=0;
ttt9=0;
n_back_tracking9=0;
count9=count;

% sub search space # 10
overall_best_error10=best_error10;
overall_best_neighbor10=best_neighbor10;
radius10=radius;
n10=0;
t10=0;
tt10=0;
ttt10=0;
n_back_tracking10=0;
count10=count;
% sub search space # 11
overall_best_error11=best_error11;
overall_best_neighbor11=best_neighbor11;
radius11=radius;
n11=0;
t11=0;
tt11=0;

```

```

ttt11=0;
n_back_tracking11=0;
count11=count;

% sub search space # 12
overall_best_error12=best_error12;
overall_best_neighbor12=best_neighbor12;
radius12=radius;
n12=0;
t12=0;
tt12=0;
ttt12=0;
n_back_tracking12=0;
count12=count;

% sub search space # 13
overall_best_error13=best_error13;
overall_best_neighbor13=best_neighbor13;
radius13=radius;
n13=0;
t13=0;
tt13=0;
ttt13=0;
n_back_tracking13=0;
count13=count;

% sub search space # 14
overall_best_error14=best_error14;
overall_best_neighbor14=best_neighbor14;
radius14=radius;
n14=0;
t14=0;
tt14=0;
ttt14=0;
n_back_tracking14=0;
count14=count;

% sub search space # 15
overall_best_error15=best_error15;
overall_best_neighbor15=best_neighbor15;
radius15=radius;
n15=0;
t15=0;
tt15=0;
ttt15=0;
n_back_tracking15=0;
count15=count;
%%%%%%%%%%%%%%
tic;
%%%%%%%%%%%%%%
% Path # 1
t1=t1+1;
tt1=tt1+1;
ttt1=ttt1+1;

% Path # 2
t2=t2+1;
tt2=tt2+1;
ttt2=ttt2+1;

% Path # 3
t3=t3+1;
tt3=tt3+1;

```

```
ttt3=ttt3+1;
```

```
% Path # 4  
t4=t4+1;  
tt4=tt4+1;  
ttt4=ttt4+1;
```

```
% Path # 5  
t5=t5+1;  
tt5=tt5+1;  
ttt5=ttt5+1;
```

```
% Path # 6  
t6=t6+1;  
tt6=tt6+1;  
ttt6=ttt6+1;
```

```
% Path # 7  
t7=t7+1;  
tt7=tt7+1;  
ttt7=ttt7+1;
```

```
% Path # 8  
t8=t8+1;  
tt8=tt8+1;  
ttt8=ttt8+1;
```

```
% Path # 9  
t9=t9+1;  
tt9=tt9+1;  
ttt9=ttt9+1;
```

```
% Path # 10  
t10=t10+1;  
tt10=tt10+1;  
ttt10=ttt10+1;
```

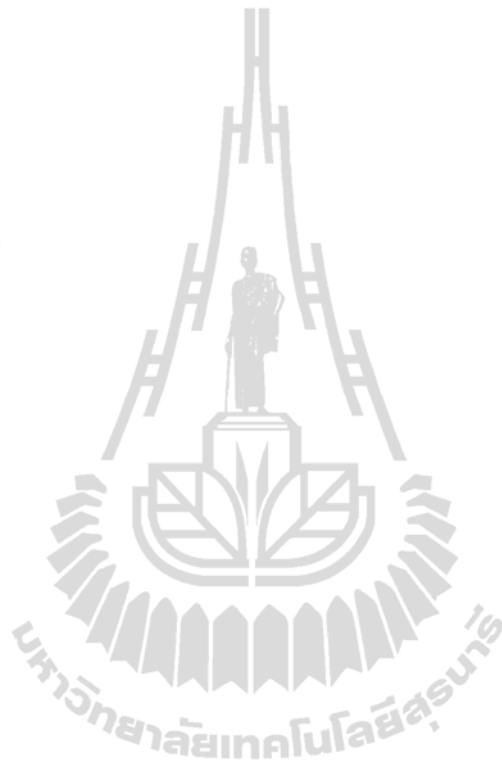
```
% Path # 11  
t11=t11+1;  
tt11=tt11+1;  
ttt11=ttt11+1;
```

```
% Path # 12  
t12=t12+1;  
tt12=tt12+1;  
ttt12=ttt12+1;
```

```
% Path # 13  
t13=t13+1;  
tt13=tt13+1;  
ttt13=ttt13+1;
```

```
% Path # 14  
t14=t14+1;  
tt14=tt14+1;  
ttt14=ttt14+1;
```

```
% Path # 15  
t15=t15+1;  
tt15=tt15+1;  
ttt15=ttt15+1;
```



```

disp([count overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4
      overall_best_error5 ...
      overall_best_error6 overall_best_error7 overall_best_error8]);
disp([count overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12 ...
      overall_best_error13 overall_best_error14 overall_best_error15]);
% Sub search space # 1
local1(t1,1)=count1;
local1(t1,2:16)=best_neighbor01;
local1(t1,17)=best_error01;
tabu_list1(tt1,1)=count1;
tabu_list1(tt1,2:16)=best_neighbor01;
tabu_list1(tt1,17)=best_error01;
best_error_list1(ttt1,1)=count1;
best_error_list1(ttt1,2:16)=best_neighbor01;
best_error_list1(ttt1,17)=best_error01;

% Sub search space # 2
local2(t2,1)=count2;
local2(t2,2:16)=best_neighbor02;
local2(t2,17)=best_error02;
tabu_list2(tt2,1)=count2;
tabu_list2(tt2,2:16)=best_neighbor02;
tabu_list2(tt2,17)=best_error02;
best_error_list2(ttt2,1)=count2;
best_error_list2(ttt2,2:16)=best_neighbor02;
best_error_list2(ttt2,17)=best_error02;

% Sub search space # 3
local3(t3,1)=count3;
local3(t3,2:16)=best_neighbor03;
local3(t3,17)=best_error03;
tabu_list3(tt3,1)=count3;
tabu_list3(tt3,2:16)=best_neighbor03;
tabu_list3(tt3,17)=best_error03;
best_error_list3(ttt3,1)=count3;
best_error_list3(ttt3,2:16)=best_neighbor03;
best_error_list3(ttt3,17)=best_error03;

% Sub search space # 4
local4(t4,1)=count4;
local4(t4,2:16)=best_neighbor04;
local4(t4,17)=best_error04;
tabu_list4(tt4,1)=count4;
tabu_list4(tt4,2:16)=best_neighbor04;
tabu_list4(tt4,17)=best_error04;
best_error_list4(ttt4,1)=count4;
best_error_list4(ttt4,2:16)=best_neighbor04;
best_error_list4(ttt4,17)=best_error04;

% Sub search space # 5
local5(t5,1)=count5;
local5(t5,2:16)=best_neighbor05;
local5(t5,17)=best_error05;
tabu_list5(tt5,1)=count5;
tabu_list5(tt5,2:16)=best_neighbor05;
tabu_list5(tt5,17)=best_error05;
best_error_list5(ttt5,1)=count5;
best_error_list5(ttt5,2:16)=best_neighbor05;
best_error_list5(ttt5,17)=best_error05;

% Sub search space # 6
local6(t6,1)=count6;
local6(t6,2:16)=best_neighbor06;

```

```

local6(t6,17)=best_error06;
tabu_list6(tt6,1)=count6;
tabu_list6(tt6,2:16)=best_neighbor06;
tabu_list6(tt6,17)=best_error06;
best_error_list6(ttt6,1)=count6;
best_error_list6(tt6,2:16)=best_neighbor06;
best_error_list6(ttt6,17)=best_error06;

% Sub search space # 7
local7(t7,1)=count7;
local7(t7,2:16)=best_neighbor07;
local7(t7,17)=best_error07;
tabu_list7(tt7,1)=count7;
tabu_list7(tt7,2:16)=best_neighbor07;
tabu_list7(tt7,17)=best_error07;
best_error_list7(ttt7,1)=count7;
best_error_list7(tt7,2:16)=best_neighbor07;
best_error_list7(ttt7,17)=best_error07;

% Sub search space # 8
local8(t8,1)=count8;
local8(t8,2:16)=best_neighbor08;
local8(t8,17)=best_error08;
tabu_list8(tt8,1)=count8;
tabu_list8(tt8,2:16)=best_neighbor08;
tabu_list8(tt8,17)=best_error08;
best_error_list8(ttt8,1)=count2;
best_error_list8(tt8,2:16)=best_neighbor08;
best_error_list8(ttt8,17)=best_error08;

% Sub search space # 9
local9(t9,1)=count9;
local9(t9,2:16)=best_neighbor09;
local9(t9,17)=best_error09;
tabu_list9(tt9,1)=count9;
tabu_list9(tt9,2:16)=best_neighbor09;
tabu_list9(tt9,17)=best_error09;
best_error_list9(ttt9,1)=count9;
best_error_list9(tt9,2:16)=best_neighbor09;
best_error_list9(ttt9,17)=best_error09;

% Sub search space # 10
local10(t10,1)=count10;
local10(t10,2:16)=best_neighbor10;
local10(t10,17)=best_error10;
tabu_list10(tt10,1)=count10;
tabu_list10(tt10,2:16)=best_neighbor10;
tabu_list10(tt10,17)=best_error10;
best_error_list10(ttt10,1)=count10;
best_error_list10(tt10,2:16)=best_neighbor10;
best_error_list10(ttt10,17)=best_error10;

% Sub search space # 11
local11(t11,1)=count11;
local11(t11,2:16)=best_neighbor11;
local11(t11,17)=best_error11;
tabu_list11(tt11,1)=count11;
tabu_list11(tt11,2:16)=best_neighbor11;
tabu_list11(tt11,17)=best_error11;
best_error_list11(ttt11,1)=count11;
best_error_list11(tt11,2:16)=best_neighbor11;
best_error_list11(ttt11,17)=best_error11;

```

```

% Sub search space # 12
local12(t12,1)=count12;
local12(t12,2:16)=best_neighbor12;
local12(t12,17)=best_error12;
tabu_list12(tt12,1)=count12;
tabu_list12(tt12,2:16)=best_neighbor12;
tabu_list12(tt12,17)=best_error12;
best_error_list12(ttt12,1)=count12;
best_error_list12(ttt12,2:16)=best_neighbor12;
best_error_list12(ttt12,17)=best_error12;

% Sub search space # 13
local13(t13,1)=count13;
local13(t13,2:16)=best_neighbor13;
local13(t13,17)=best_error13;
tabu_list13(tt13,1)=count13;
tabu_list13(tt13,2:16)=best_neighbor13;
tabu_list13(tt13,17)=best_error13;
best_error_list13(ttt13,1)=count13;
best_error_list13(ttt13,2:16)=best_neighbor13;
best_error_list13(ttt13,17)=best_error13;

% Sub search space # 14
local14(t14,1)=count14;
local14(t14,2:16)=best_neighbor14;
local14(t14,17)=best_error14;
tabu_list14(tt14,1)=count14;
tabu_list14(tt14,2:16)=best_neighbor14;
tabu_list14(tt14,17)=best_error14;
best_error_list14(ttt14,1)=count14;
best_error_list14(ttt14,2:16)=best_neighbor14;
best_error_list14(ttt14,17)=best_error14;

% Sub search space # 15
local15(t15,1)=count15;
local15(t15,2:16)=best_neighbor15;
local15(t15,17)=best_error15;
tabu_list15(tt15,1)=count15;
tabu_list15(tt15,2:16)=best_neighbor15;
tabu_list15(tt15,17)=best_error15;
best_error_list15(ttt15,1)=count15;
best_error_list15(ttt15,2:16)=best_neighbor15;
best_error_list15(ttt15,17)=best_error15;

% Define flag
flag1=1;
flag2=1;
flag3=1;
flag4=1;
flag5=1;
flag6=1;
flag6=1;
flag7=1;
flag8=1;
flag9=1;
flag10=1;
flag11=1;
flag12=1;
flag13=1;
flag14=1;
flag15=1;

```

```

% Start ATS
while (count<max_count)
    count=count+1;

%#####
%                Path 1
%#####

if (flag1==1)

% sub search space # 1
S101=random_neigh(Number_neighb,radius1,xlimit1,S01);
[best_error101,best_neighbor101,best_error01,best_neighbor01]=objective(S101,best_error01,S01);

% Path # 1
if (best_error101-best_error01)>1e-18
    n1=n1+1;
else
    n1=0;
end
tt1=tt1+1;
tabu_list1(tt1,1)=count;
tabu_list1(tt1,2:16)=best_neighbor101;
tabu_list1(tt1,17)=best_error101;
ttt1= ttt1+1;
best_error_list1(ttt1,1)=count;
best_error_list1(ttt1,2:16)=best_neighbor01;
best_error_list1(ttt1,17)=best_error01;

% Start AR mechanism

% Path # 1
if best_error01<20
    radius1=1e-2;
end
if best_error01<8
    radius1=1e-3;
end
if best_error01<3.5
    radius1=1e-4;
end
if (overall_best_error1<=TC)
    t1=t1+1;
    tt1=tt1+1;
    local1(t1,1)=count;
    local1(t1,2:16)=best_neighbor01;
    local1(t1,17)=best_error01;
    pp='Path 1 Strikes';
    overall_best_error=best_error01;
    overall_best_neighbor=best_neighbor01;
    break;
end

% Start BT mechanism

% sub search space # 1
if n1>=5
    n_back_tracking1=n_back_tracking1+1;
    TEMP1=tabu_list1(count-3:count+1,:);
    [MAX1,INDEX1] = max(TEMP1(:,17));
    RANK1(5,:) = TEMP1(INDEX1,:);
    neighbor1=RANK1(5,2:16);

```

```

S01=neighbor1;
best_error01=RANK1(5,17);
n1=0;
else
S01=best_neighbor01;
best_error01=best_error01;
end

% Updating Path 1
% Path # 1
if overall_best_error1>best_error01
    overall_best_error1=best_error01;
    overall_best_neighbor1=best_neighbor01;
end
count1=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if (flag2==1)

% sub search space # 2
S102=random_neigh(Number_neighb,radius2,xlimit2,S02);
[best_error102,best_neighbor102,best_error02,best_neighbor02]=objective(S102,best_error02,S02);

% Path # 2
if (best_error102-best_error02)>1e-18
    n2=n2+1;
else
    n2=0;
end
tt2=tt2+1;
tabu_list2(tt2,1)=count;
tabu_list2(tt2,2:16)=best_neighbor102;
tabu_list2(tt2,17)=best_error102;
t2=t2+1;
best_error_list2(t2,1)=count;
best_error_list2(t2,2:16)=best_neighbor02;
best_error_list2(t2,17)=best_error02;

% Start AR

% Path # 2
if best_error02<20
    radius2=1e-2;
end
if best_error02<8
    radius2=1e-3;
end
if best_error02<3.5
    radius2=1e-4;
end
if (overall_best_error2<TC)
    t2=t2+1;
    tt2=tt2+1;
    local2(t2,1)=count;
    local2(t2,2:16)=best_neighbor02;
    local2(t2,17)=best_error02;
    pp='Path 2 Strikes';
    overall_best_error=best_error02;

```

```

overall_best_neighbor=best_neighbor02;
break;
end

% Start BT
% sub search space # 2
if n2>=5
n_back_tracking2=n_back_tracking2+1;
TEMP2=tabu_list2(count-3:count+1,:);
[ MAX2,INDEX2] = max(TEMP2(:,17));
RANK2(5,:) = TEMP2(INDEX2,:);
neighbor2=RANK2(5,2:16);
S02=neighbor2;
best_error02=RANK2(5,17);
n2=0;
else
S02=best_neighbor02;
best_error02=best_error02;
end

% Updating Path 2
if overall_best_error2>best_error02
overall_best_error2=best_error02;
overall_best_neighbor2=best_neighbor02;
end
count2=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if (flag3==1)

% sub search space # 3
S103=random_neigh(Number_neigh,radius3,xlimit3,S03);
[best_error103,best_neighbor103,best_error03,best_neighbor03]=objective(S103,best_error03,S03);

% Path # 3
if (best_error103-best_error03)>1e-18
n3=n3+1;
else
n3=0;
end
tt3=tt3+1;
tabu_list3(tt3,1)=count;
tabu_list3(tt3,2:16)=best_neighbor103;
tabu_list3(tt3,17)=best_error103;
t3= t3+1;
best_error_list3(t3,1)=count;
best_error_list3(t3,2:16)=best_neighbor03;
best_error_list3(t3,17)=best_error03;

% Start AR mechanism

% Path # 3
if best_error03<20
radius3=1e-2;
end
if best_error03<8
radius3=1e-3;

```

```

end
if best_error03<3.5
radius3=1e-4;
end
if (overall_best_error3<TC)
t3=t3+1;
tt3=tt3+1;
local3(t3,1)=count;
local3(t3,2:16)=best_neighbor03;
local3(t3,17)=best_error03;
pp='Path 3 Strikes';
overall_best_error=best_error03;
overall_best_neighbor=best_neighbor03;
break;
end

% Start BT mechanism

% sub search space # 3
if n3>=5
n_back_tracking3=n_back_tracking3+1;
TEMP3=tabu_list3(count-3:count+1,:);
[ MAX3,INDEX3] = max(TEMP3(:,17));
RANK3(5,:)=TEMP3(INDEX3,:);
neighbor3=RANK3(5,2:16);
S03=neighbor3;
best_error03=RANK3(5,17);
n3=0;
else
S03=best_neighbor03;
best_error03=best_error03;
end

% Updating Path 3
% Path # 3
if overall_best_error3>best_error03
overall_best_error3=best_error03;
overall_best_neighbor3=best_neighbor03;
end
count3=count;
end

%#####
% Path 4
%#####
if (flag4==1)
% sub search space # 4
S104=random_neigh(Number_neighb,radius4,xlimit4,S04);
[best_error104,best_neighbor104,best_error04,best_neighbor04]=objective(S104,best_error04,S04);

% Path # 4
if (best_error104-best_error04)>1e-18
n4=n4+1;
else
n4=0;
end
tt4=tt4+1;
tabu_list4(tt4,1)=count;
tabu_list4(tt4,2:16)=best_neighbor104;
tabu_list4(tt4,17)=best_error104;
ttt4= ttt4+1;
best_error_list4(ttt4,1)=count;

```

```

best_error_list4(ttt4,2:16)=best_neighbor04;
best_error_list4(ttt4,17)=best_error04;

#####
%           Start AR
#####
% Path # 4
if best_error04<20
radius4=1e-2;
end
if best_error04<8
radius4=1e-3;
end
if best_error04<3.5
radius4=1e-4;
end
if (overall_best_error4<=TC)
t4=t4+1;
tt4=tt4+1;
local4(t4,1)=count;
local4(t4,2:16)=best_neighbor04;
local4(t4,17)=best_error04;
pp='Path 4 Strikes';
overall_best_error=best_error04;
overall_best_neighbor=best_neighbor04;
break;
end

#####
%           Start BT
#####

% sub search space # 4
if n4>=5
n_back_tracking4=n_back_tracking4+1;
TEMP4=tabu_list4(count-3:count+1,:);
[MAX4,INDEX4] = max(TEMP4(:,17));
RANK4(5,:)=TEMP4(INDEX4,:);
neighbor4=RANK4(5,2:16);
S04=neighbor4;
best_error04=RANK4(5,17);
n4=0;
else
S04=best_neighbor04;
best_error04=best_error04;
end

% Updating Path 4
% Path # 4
if overall_best_error4>best_error04
overall_best_error4=best_error04;
overall_best_neighbor4=best_neighbor04;
end
count4=count;
end

#####
%           Path 5
#####
if (flag5==1)
% sub search space # 5
S105=random_neigh(Number_neighb,radius5,xlimit5,S05);
[best_error105,best_neighbor105,best_error05,best_neighbor05]=objective(S105,best_error05,S05);

```

```

% Path # 4
if (best_error105-best_error05)>1e-18
    n5=n5+1;
else
    n5=0;
end
tt5=tt5+1;
tabu_list5(tt5,1)=count;
tabu_list5(tt5,2:16)=best_neighbor105;
tabu_list5(tt5,17)=best_error105;
ttt5= ttt5+1;
best_error_list5(ttt5,1)=count;
best_error_list5(ttt5,2:16)=best_neighbor05;
best_error_list5(ttt5,17)=best_error05;

#####
%                               Start AR
#####
% Path # 5
if best_error05<20
    radius5=1e-2;
end
if best_error05<8
    radius5=1e-3;
end
if best_error05<3.5
    radius5=1e-4;
end
if (overall_best_error5<=TC)
    t5=t5+1;
    tt5=tt5+1;
    local5(t5,1)=count;
    local5(t5,2:16)=best_neighbor05;
    local5(t5,17)=best_error05;
    pp='Path 5 Strikes';
    overall_best_error=best_error05;
    overall_best_neighbor=best_neighbor05;
    break;
end
#####
%                               Start BT
#####

% sub search space # 5
if n5>=5
    n_back_tracking5=n_back_tracking5+1;
    TEMP5=tabu_list5(count-3:count+1,:);
    [MAX5,INDEX5] = max(TEMP5(:,17));
    RANK5(5,:)=TEMP5(INDEX5,:);
    neighbor5=RANK5(5,2:16);
    S05=neighbor5;
    best_error05=RANK5(5,17);
    n5=0;
else
    S05=best_neighbor05;
    best_error05=best_error05;
end

% Updating Path 5
% Path # 5
if overall_best_error5>best_error05
    overall_best_error5=best_error05;
    overall_best_neighbor5=best_neighbor05;

```

```

end
count5=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 6
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (flag6==1)
% sub search space # 6
S106=random_neigh(Number_neighb,radius6,xlimit6,S06);
[best_error106,best_neighbor106,best_error06,best_neighbor06]=objective(S106,best_error06,S06);

% Path # 6
if (best_error106-best_error06)>1e-18
    n6=n6+1;
else
    n6=0;
end
tt6=tt6+1;
tabu_list6(tt6,1)=count;
tabu_list6(tt6,2:16)=best_neighbor106;
tabu_list6(tt6,17)=best_error106;
ttt6= ttt6+1;
best_error_list6(ttt6,1)=count;
best_error_list6(ttt6,2:16)=best_neighbor06;
best_error_list6(ttt6,17)=best_error06;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start AR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Path # 6
if best_error06<20
radius6=1e-2;
end
if best_error06<8
radius6=1e-3;
end
if best_error06<3.5
radius6=1e-4;
end
if (overall_best_error6<=TC)
t6=t6+1;
tt6=tt6+1;
local6(t6,1)=count;
local6(t6,2:16)=best_neighbor06;
local6(t6,17)=best_error06;
pp='Path 6 Strikes';
overall_best_error=best_error06;
overall_best_neighbor=best_neighbor06;
break;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start BT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% sub search space # 6
if n6>=5
n_back_tracking6=n_back_tracking6+1;
TEMP6=tabu_list6(count-3:count+1,:);
[MAX6,INDEX6] = max(TEMP6(:,17));
RANK6(5,:)= TEMP6(INDEX6,:);

```

```

neighbor6=RANK6(5,2:16);
S06=neighbor6;
best_error06=RANK6(5,17);
n6=0;
else
S06=best_neighbor06;
best_error06=best_error06;
end

% Updating Path 6
% Path # 6
if overall_best_error6>best_error06
    overall_best_error6=best_error06;
    overall_best_neighbor6=best_neighbor06;
end
count6=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 7
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (flag7==1)
% sub search space # 7
S107=random_neigh(Number_neighb,radius7,xlimit7,S07);
[best_error107,best_neighbor107,best_error07,best_neighbor07]=objective(S107,best_error07,S07);

% Path # 7
if (best_error107-best_error07)>1e-18
    n7=n7+1;
else
    n7=0;
end
tt7=tt7+1;
tabu_list7(tt7,1)=count;
tabu_list7(tt7,2:16)=best_neighbor107;
tabu_list7(tt7,17)=best_error107;
t7= t7+1;
best_error_list7(t7,1)=count;
best_error_list7(t7,2:16)=best_neighbor07;
best_error_list7(t7,17)=best_error07;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start AR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Path # 7
if best_error07<20
    radius7=1e-2;
end
if best_error07<8
    radius7=1e-3;
end
if best_error07<3.5
    radius7=1e-4;
end
if (overall_best_error7<=TC)
    t7=t7+1;
    tt7=tt7+1;
    local7(t7,1)=count;
    local7(t7,2:16)=best_neighbor07;
    local7(t7,17)=best_error07;
    pp='Path 7 Strikes';
    overall_best_error=best_error07;
    overall_best_neighbor=best_neighbor07;

```

```

break;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start BT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% sub search space # 7
if n7>=5
    n_back_tracking7=n_back_tracking7+1;
    TEMP7=tabu_list7(count-3:count+1,:);
    [MAX7,INDEX7] = max(TEMP7(:,17));
    RANK7(5,:) = TEMP7(INDEX7,:);
    neighbor7=RANK7(5,2:16);
    S07=neighbor7;
    best_error07=RANK7(5,17);
    n7=0;
else
    S07=best_neighbor07;
    best_error07=best_error07;
end

% Updating Path 7
% Path # 7
if overall_best_error7>best_error07
    overall_best_error7=best_error07;
    overall_best_neighbor7=best_neighbor07;
end
count7=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 8
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (flag8==1)
% sub search space # 8
S108=random_neigh(Number_neighb,radius8,xlimit8,S08);
[best_error108,best_neighbor108,best_error08,best_neighbor08]=objective(S108,best_error08,S08);

% Path # 8
if (best_error108-best_error08)>1e-18
    n8=n8+1;
else
    n8=0;
end
tt8=tt8+1;
tabu_list8(tt8,1)=count;
tabu_list8(tt8,2:16)=best_neighbor108;
tabu_list8(tt8,17)=best_error108;
ttt8= ttt8+1;
best_error_list8(ttt8,1)=count;
best_error_list8(ttt8,2:16)=best_neighbor08;
best_error_list8(ttt8,17)=best_error08;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start AR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Path # 8
if best_error08<20
    radius8=1e-2;
end
if best_error08<8

```

```

radius8=1e-3;
end
if best_error08<3.5
radius8=1e-4;
end
if (overall_best_error8<=TC)
t8=t8+1;
tt8=tt8+1;
local8(t8,1)=count;
local8(t8,2:16)=best_neighbor08;
local8(t8,17)=best_error08;
pp='Path 8 Strikes';
overall_best_error=best_error08;
overall_best_neighbor=best_neighbor08;
break;
end

#####
%          Start BT
#####

% sub search space # 8
if n8>=5
n_back_tracking8=n_back_tracking8+1;
TEMP8=tabu_list8(count-3:count+1,:);
[MAX8,INDEX8] = max(TEMP8(:,17));
RANK8(5,:) = TEMP8(INDEX8,:);
neighbor8=RANK8(5,2:16);
S08=neighbor8;
best_error08=RANK8(5,17);
n8=0;
else
S08=best_neighbor08;
best_error08=best_error08;
end

% Updating Path 8
% Path # 8
if overall_best_error8>best_error08
overall_best_error8=best_error08;
overall_best_neighbor8=best_neighbor08;
end
count8=count;
end

#####
%          Path 9
#####
if (flag9==1)
% sub search space # 9
S109=random_neigh(Number_neighb,radius9,xlimit9,S09);
[best_error109,best_neighbor109,best_error09,best_neighbor09]=objective(S109,best_error09,S09);

% Path # 9
if (best_error109-best_error09)>1e-18
n9=n9+1;
else
n9=0;
end
tt9=tt9+1;
tabu_list9(tt9,1)=count;
tabu_list9(tt9,2:16)=best_neighbor109;

```

```

tabu_list9(tt9,17)=best_error109;
ttt9= ttt9+1;
best_error_list9(ttt9,1)=count;
best_error_list9(ttt9,2:16)=best_neighbor09;
best_error_list9(ttt9,17)=best_error09;

#####
%           Start AR
#####
% Path # 9
if best_error09<20
radius9=1e-2;
end
if best_error09<8
radius9=1e-3;
end
if best_error09<3.5
radius9=1e-4;
end
if (overall_best_error9<=TC)
t9=t9+1;
tt9=tt9+1;
local9(t9,1)=count;
local9(t9,2:16)=best_neighbor09;
local9(t9,17)=best_error09;
pp='Path 9 Strikes';
overall_best_error=best_error09;
overall_best_neighbor=best_neighbor09;
break;
end
#####
%           Start BT
#####

% sub search space # 9
if n9>=5
n_back_tracking9=n_back_tracking9+1;
TEMP9=tabu_list9(count-3:count+1,:);
[MAX9,INDEX9] = max(TEMP9(:,17));
RANK9(5,:) = TEMP9(INDEX9,:);
neighbor9=RANK9(5,2:16);
S09=neighbor9;
best_error09=RANK9(5,17);
n9=0;
else
S09=best_neighbor09;
best_error09=best_error09;
end

% Updating Path 9
% Path # 9
if overall_best_error9>best_error09
overall_best_error9=best_error09;
overall_best_neighbor9=best_neighbor09;
end
count9=count;
end

#####
%           Path 10
#####
if (flag10==1)
% sub search space # 10

```

```

S110=random_neigh(Number_neighb,radius10,xlimit10,S10);
[best_error110,best_neighbor110,best_error10,best_neighbor10]=objective(S110,best_error10,S10);

% Path # 10
if (best_error110-best_error10)>1e-18
    n10=n10+1;
else
    n10=0;
end
tt10=tt10+1;
tabu_list10(tt10,1)=count;
tabu_list10(tt10,2:16)=best_neighbor110;
tabu_list10(tt10,17)=best_error110;
ttt10= ttt10+1;
best_error_list10(ttt10,1)=count;
best_error_list10(ttt10,2:16)=best_neighbor110;
best_error_list10(ttt10,17)=best_error110;

#####
%                Start AR
#####
% Path # 10
if best_error10<20
    radius10=1e-2;
end
if best_error10<8
    radius10=1e-3;
end
if best_error10<3.5
    radius10=1e-4;
end
if (overall_best_error10<=TC)
    t10=t10+1;
    tt10=tt10+1;
    local10(t10,1)=count;
    local10(t10,2:16)=best_neighbor10;
    local10(t10,17)=best_error10;
    pp='Path 10 Strikes';
    overall_best_error=best_error10;
    overall_best_neighbor=best_neighbor10;
    break;
end

#####
%                Start BT
#####

% sub search space # 10
if n10>=5
    n_back_tracking10=n_back_tracking10+1;
    TEMP10=tabu_list10(count-3:count+1,:);
    [MAX10,INDEX10] = max(TEMP10(:,17));
    RANK10(5,:) = TEMP10(INDEX10,:);
    neighbor10=RANK10(5,2:16);
    S10=neighbor10;
    best_error10=RANK10(5,17);
    n10=0;
else
    S10=best_neighbor10;
    best_error10=best_error10;
end

% Updating Path 10

```

```

% Path # 10
if overall_best_error10>best_error10
    overall_best_error10=best_error10;
    overall_best_neighbor10=best_neighbor10;
end
count10=count;
end

#####
%
% Path 11
#####
if (flag11==1)
% sub search space # 11
S111=random_neigh(Number_neighb,radius11,xlimit11,S11);
[best_error111,best_neighbor111,best_error11,best_neighbor11]=objective(S111,best_error11,S11);

% Path # 11
if (best_error111-best_error11)>1e-18
    n11=n11+1;
else
    n11=0;
end
tt11=tt11+1;
tabu_list11(tt11,1)=count;
tabu_list11(tt11,2:16)=best_neighbor111;
tabu_list11(tt11,17)=best_error111;
ttt11= ttt11+1;
best_error_list11(ttt11,1)=count;
best_error_list11(ttt11,2:16)=best_neighbor11;
best_error_list11(ttt11,17)=best_error11;

#####
%
% Start AR
#####
% Path # 11
if best_error11<20
radius11=1e-2;
end
if best_error11<8
radius11=1e-3;
end
if best_error11<3.5
radius11=1e-4;
end
if (overall_best_error11<=TC)
t11=t11+1;
tt11=tt11+1;
local11(t11,1)=count;
local11(t11,2:16)=best_neighbor11;
local11(t11,17)=best_error11;
pp='Path 11 Strikes';
overall_best_error=best_error11;
overall_best_neighbor=best_neighbor11;
break;
end

#####
%
% Start BT
#####

% sub search space # 11
if n11>=5
    n_back_tracking11=n_back_tracking11+1;

```

```

TEMP11=tabu_list11(count-3:count+1,:);
[MAX11,INDEX11] = max(TEMP11(:,17));
RANK11(5,:) = TEMP11(INDEX11,:);
neighbor11=RANK11(5,2:16);
S11=neighbor11;
best_error11=RANK11(5,17);
n11=0;
else
S11=best_neighbor11;
best_error11=best_error11;
end

% Updating Path 11
% Path # 11
if overall_best_error11>best_error11
    overall_best_error11=best_error11;
    overall_best_neighbor11=best_neighbor11;
end
count11=count;
end

#####
%
% Path 12
#####
if (flag12==1)
% sub search space # 12
S112=random_neigh(Number_neighb,radius12,xlimit12,S12);
[best_error112,best_neighbor112,best_error12,best_neighbor12]=objective(S112,best_error12,S12);

% Path # 12
if (best_error112-best_error12)>1e-18
    n12=n12+1;
else
    n12=0;
end
tt12=tt12+1;
tabu_list12(tt12,1)=count;
tabu_list12(tt12,2:16)=best_neighbor112;
tabu_list12(tt12,17)=best_error112;
t12= t12+1;
best_error_list12(t12,1)=count;
best_error_list12(t12,2:16)=best_neighbor12;
best_error_list12(t12,17)=best_error12;

#####
%
% Start AR
#####
% Path # 12
if best_error12<20
    radius12=1e-2;
end
if best_error12<8
    radius12=1e-3;
end
if best_error12<3.5
    radius12=1e-4;
end
if (overall_best_error12<=TC)
    t12=t12+1;
    tt12=tt12+1;
    local12(t12,1)=count;
    local12(t12,2:16)=best_neighbor12;
    local12(t12,17)=best_error12;

```

```

pp='Path 12 Strikes';
overall_best_error=best_error12;
overall_best_neighbor=best_neighbor12;
break;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Start BT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% sub search space # 12
if n12>=5
    n_back_tracking12=n_back_tracking12+1;
    TEMP12=tabu_list12(count-3:count+1,:);
    [MAX12,INDEX12] = max(TEMP12(:,17));
    RANK12(5,:) = TEMP12(INDEX12,:);
    neighbor12=RANK12(5,2:16);
    S12=neighbor12;
    best_error12=RANK12(5,17);
    n12=0;
else
    S12=best_neighbor12;
    best_error12=best_error12;
end

% Updating Path 12
% Path # 12
if overall_best_error12>best_error12
    overall_best_error12=best_error12;
    overall_best_neighbor12=best_neighbor12;
end
count12=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Path 13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (flag13==1)
% sub search space # 13
S113=random_neigh(Number_neighb,radius13,xlimit13,S13);
[best_error113,best_neighbor113,best_error13,best_neighbor13]=objective(S113,best_error13,S13);

% Path # 13
if (best_error113-best_error13)>1e-18
    n13=n13+1;
else
    n13=0;
end
tt13=tt13+1;
tabu_list13(tt13,1)=count;
tabu_list13(tt13,2:16)=best_neighbor113;
tabu_list13(tt13,17)=best_error113;
ttt13= ttt13+1;
best_error_list13(ttt13,1)=count;
best_error_list13(ttt13,2:16)=best_neighbor13;
best_error_list13(ttt13,17)=best_error13;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Start AR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Path # 13
if best_error13<20
    radius13=1e-2;

```

```

end
if best_error13<8
radius13=1e-3;
end
if best_error13<3.5
radius13=1e-5;
end
if (overall_best_error13<=TC)
t13=t13+1;
tt13=tt13+1;
local13(t13,1)=count;
local13(t13,2:16)=best_neighbor13;
local13(t13,17)=best_error13;
pp='Path 13 Strikes';
overall_best_error=best_error13;
overall_best_neighbor=best_neighbor13;
break;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Start BT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% sub search space # 13
if n13>=5
n_back_tracking13=n_back_tracking13+1;
TEMP13=tabu_list13(count-3:count+1,:);
[MAX13,INDEX13] = max(TEMP13(:,17));
RANK13(5,:) = TEMP13(INDEX13,:);
neighbor13=RANK13(5,2:16);
S13=neighbor13;
best_error13=RANK13(5,17);
n13=0;
else
S13=best_neighbor13;
best_error13=best_error13;
end

% Updating Path 13
% Path # 13
if overall_best_error13>best_error13
overall_best_error13=best_error13;
overall_best_neighbor13=best_neighbor13;
end
count13=count;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Path 14
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (flag14==1)
% sub search space # 14
S114=random_neigh(Number_neighb,radius14,xlimit14,S14);
[best_error114,best_neighbor114,best_error14,best_neighbor14]=objective(S114,best_error14,S14);

% Path # 14
if (best_error114-best_error14)>1e-18
n14=n14+1;
else
n14=0;
end
tt14=tt14+1;
tabu_list14(tt14,1)=count;

```

```

tabu_list14(tt14,2:16)=best_neighbor14;
tabu_list14(tt14,17)=best_error14;
ttt14= ttt14+1;
best_error_list14(ttt14,1)=count;
best_error_list14(ttt14,2:16)=best_neighbor14;
best_error_list14(ttt14,17)=best_error14;

#####
%                               Start AR
#####
% Path # 14
if best_error14<20
radius14=1e-2;
end
if best_error14<8
radius14=1e-3;
end
if best_error14<3.5
radius14=1e-4;
end
if (overall_best_error14<=TC)
t14=t14+1;
tt14=tt14+1;
local14(t14,1)=count;
local14(t14,2:16)=best_neighbor14;
local14(t14,17)=best_error14;
pp='Path 14 Strikes';
overall_best_error=best_error14;
overall_best_neighbor=best_neighbor14;
break;
end

#####
%                               Start BT
#####

% sub search space # 14
if n14>=5
n_back_tracking14=n_back_tracking14+1;
TEMP14=tabu_list14(count-3:count+1,:);
[MAX14,INDEX14] = max(TEMP14(:,17));
RANK14(5,:) = TEMP14(INDEX14,:);
neighbor14=RANK14(5,2:16);
S14=neighbor14;
best_error14=RANK14(5,17);
n14=0;
else
S14=best_neighbor14;
best_error14=best_error14;
end

% Updating Path 14
% Path # 14
if overall_best_error14>best_error14
overall_best_error14=best_error14;
overall_best_neighbor14=best_neighbor14;
end
count14=count;
end

#####
%                               Path 15
#####

```

```

if (flag15==1)
% sub search space # 15
S115=random_neigh(Number_neighb,radius15,xlimit15,S15);
[best_error115,best_neighbor115,best_error15,best_neighbor15]=objective(S115,best_error15,S15);

% Path # 15
if (best_error115-best_error15)>1e-18
    n15=n15+1;
else
    n15=0;
end
tt15=tt15+1;
tabu_list15(tt15,1)=count;
tabu_list15(tt15,2:16)=best_neighbor115;
tabu_list15(tt15,17)=best_error115;
t15= t15+1;
best_error_list15(t15,1)=count;
best_error_list15(t15,2:16)=best_neighbor15;
best_error_list15(t15,17)=best_error15;

#####
%          Start AR
#####
% Path # 15
if best_error15<20
    radius15=1e-2;
end
if best_error15<8
    radius15=1e-3;
end
if best_error15<3.5
    radius15=1e-5;
end
if (overall_best_error15<=TC)
    t15=t15+1;
    tt15=tt15+1;
    local15(t15,1)=count;
    local15(t15,2:16)=best_neighbor15;
    local15(t15,17)=best_error15;
    pp='Path 15 Strikes';
    overall_best_error=best_error15;
    overall_best_neighbor=best_neighbor15;
    break;
end

#####
%          Start BT
#####

% sub search space # 15
if n15>=5
    n_back_tracking15=n_back_tracking15+1;
    TEMP15=tabu_list15(count-3:count+1,:);
    [MAX15,INDEX15] = max(TEMP15(:,17));
    RANK15(5,:) = TEMP15(INDEX15,:);
    neighbor15=RANK15(5,2:16);
    S15=neighbor15;
    best_error15=RANK15(5,17);
    n15=0;
else
    S15=best_neighbor15;
    best_error15=best_error15;
end

```

```

% Updating Path 15
% Path # 15
if overall_best_error15>best_error15
    overall_best_error15=best_error15;
    overall_best_neighbor15=best_neighbor15;
end
count15=count;
end

disp([count overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4 overall_best_error5 .
overall_best_error6 overall_best_error7 overall_best_error8]);
disp([count overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12 ...
overall_best_error13 overall_best_error14 overall_best_error15]);

% DM Stage I reduce no. of paths from 15 to 8
if (count==10)
[Y,I]=sort([overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4...
overall_best_error5 overall_best_error6 overall_best_error7 overall_best_error8...
overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12...
overall_best_error13 overall_best_error14 overall_best_error15]);
for i=9:15
    if I(i)==1
        flag1=0;
    elseif I(i)==2
        flag2=0;
    elseif I(i)==3
        flag3=0;
    elseif I(i)==4
        flag4=0;
    elseif I(i)==5
        flag5=0;
    elseif I(i)==6
        flag6=0;
    elseif I(i)==7
        flag7=0;
    elseif I(i)==8
        flag8=0;
    elseif I(i)==9
        flag9=0;
    elseif I(i)==10
        flag10=0;
    elseif I(i)==11
        flag11=0;
    elseif I(i)==12
        flag12=0;
    elseif I(i)==13
        flag13=0;
    elseif I(i)==14
        flag14=0;
    elseif I(i)==15
        flag15=0;
    end
end
end

% DM Stage II reduce no. of paths from 8 to 4
if (count==20)
[Y,I]=sort([overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4...
overall_best_error5 overall_best_error6 overall_best_error7 overall_best_error8...
overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12...
overall_best_error13 overall_best_error14 overall_best_error15]);
for i=5:8

```

```

if I(i)==1
    flag1=0;
elseif I(i)==2
    flag2=0;
elseif I(i)==3
    flag3=0;
elseif I(i)==4
    flag4=0;
elseif I(i)==5
    flag5=0;
elseif I(i)==6
    flag6=0;
elseif I(i)==7
    flag7=0;
elseif I(i)==8
    flag8=0;
elseif I(i)==9
    flag9=0;
elseif I(i)==10
    flag10=0;
elseif I(i)==11
    flag11=0;
elseif I(i)==12
    flag12=0;
elseif I(i)==13
    flag13=0;
elseif I(i)==14
    flag14=0;
elseif I(i)==15
    flag15=0;

end
end
end

% DM Stage III reduce no. of paths from 4 to 2
if (count==30)
[Y,I]=sort([overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4...
    overall_best_error5 overall_best_error6 overall_best_error7 overall_best_error8...
    overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12...
    overall_best_error13 overall_best_error14 overall_best_error15]);
for i=3:4
    if I(i)==1
        flag1=0;
    elseif I(i)==2
        flag2=0;
    elseif I(i)==3
        flag3=0;
    elseif I(i)==4
        flag4=0;
    elseif I(i)==5
        flag5=0;
    elseif I(i)==6
        flag6=0;
    elseif I(i)==7
        flag7=0;
    elseif I(i)==8
        flag8=0;
    elseif I(i)==9
        flag9=0;
    elseif I(i)==10
        flag10=0;
    elseif I(i)==11

```

```

        flag11=0;
    elseif I(i)==12
        flag12=0;
    elseif I(i)==13
        flag13=0;
    elseif I(i)==14
        flag14=0;
    elseif I(i)==15
        flag15=0;

    end
end
end

% DM Stage IV reduce no. of paths from 2 to 1
if (count==40)
[Y,I]=sort([overall_best_error1 overall_best_error2 overall_best_error3 overall_best_error4...
    overall_best_error5 overall_best_error6 overall_best_error7 overall_best_error8...
    overall_best_error9 overall_best_error10 overall_best_error11 overall_best_error12...
    overall_best_error13 overall_best_error14 overall_best_error15]);
for i=2:2
    if I(i)==1
        flag1=0;
    elseif I(i)==2
        flag2=0;
    elseif I(i)==3
        flag3=0;
    elseif I(i)==4
        flag4=0;
    elseif I(i)==5
        flag5=0;
    elseif I(i)==6
        flag6=0;
    elseif I(i)==7
        flag7=0;
    elseif I(i)==8
        flag8=0;
    elseif I(i)==9
        flag9=0;
    elseif I(i)==10
        flag10=0;
    elseif I(i)==11
        flag11=0;
    elseif I(i)==12
        flag12=0;
    elseif I(i)==13
        flag13=0;
    elseif I(i)==14
        flag14=0;
    elseif I(i)==15
        flag15=0;
    end
end
end
end
time=toc
% pp
overall_best_error
% overall_best_neighbor
% obj_plot(overall_best_neighbor(1),overall_best_neighbor(2),overall_best_neighbor(3),...
% overall_best_neighbor(4),overall_best_neighbor(5),overall_best_neighbor(6),...
% overall_best_neighbor(7),overall_best_neighbor(8),overall_best_neighbor(9),overall_best_neighbor(10)...
```

```
%
,overall_best_neighbor(11),overall_best_neighbor(12),overall_best_neighbor(13),overall_best_neighb
or(14),overall_best_neighbor(15))
return
```

โปรแกรม obj.m

```
function [SSE]=obj(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O)
% global SSE SSE1v SSE2i SSE3i SSE4i SSE5i SSE6v

% Runge-Kutta method to solve nonlinear CPP model
% by Jukkrit Kluabwang
% Tue 23nd June 2009
% Notation list
% x = iteration time
% h = sampling time
% x1 = angular angle in radian
% x2 = derivative of angular angle in radian/sec
% x3 = horizontal displacement in meter
% x4 = derivative of horizontal displacement in meter/sec
% m = mass of pendulum in kilogram
% l = length of pendulum rod in meter
% g = gravitational acceleration in square meter/sec
% M = mass of cart in kilogram
% A = coefficient of the 7th input variable
% B = coefficient of the 6th input variable
% C = coefficient of the 5th input variable
% D = coefficient of the 4th input variable
% E = coefficient of the 3rd input variable
% F = coefficient of the 2nd input variable
% G = coefficient of the 1st input variable
% H = the last coefficient of the variable

% clear all,close all,clc

% Initialization; values of M-H came from Prof.Deacha's thesis
x=0 ;
h=0.15;
m=0.251;
l=0.40;
g=9.81;
T=0.15;
R=0.025;

% Loading input u
load DATA_PENDULUM_AFTER_CORRECTION_TAKE_2.mat

u1=U_Sensor(31:280);
a1=AP_Sensor(30:280);
y1=CP_Sensor(30:280);

u2=U_Sensor(331:580);
a2=AP_Sensor(330:580);
y2=CP_Sensor(330:580);

u3=U_Sensor(631:880);
a3=AP_Sensor(630:880);
y3=CP_Sensor(630:880);

load DATA_PENDULUM_AFTER_CORRECTION_TAKE_3.mat

u4=U_Sensor(31:280);
a4=AP_Sensor(30:280);
```

```

y4=CP_Sensor(30:280);

u5=U_Sensor(331:580);
a5=AP_Sensor(330:580);
y5=CP_Sensor(330:580);

u6=U_Sensor(631:880);
a6=AP_Sensor(630:880);
y6=CP_Sensor(630:880);

%##### Lot#1 #####
x1=-0.3497;
x2=0.0153;
x3=0.0154;
x4=0.002;
u=u1;
aa=a1;
yy=y1;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
for i=2:250
    a2dot(1)=adot(1);
    a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);
    ydot(i)=y(i)-y(i-1);
end
for i=2:250
    y2dot(1)=ydot(1);
    y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250
    % Phase I
    k1=h*obj1(x2);

    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

```

```

p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
,J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
,I,J,K,L,M,N,O,m,l,g);
    n3=h*obj3(x4+p2/2);

p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
,J,K,L,M,N,O,m,l,g);
    % Phase IV
    k4=h*obj1(x2+m3);

m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,
K,L,M,N,O,m,l,g);
    n4=h*obj3(x4+p3);

p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K
,L,M,N,O,m,l,g);
    % Derivative stage
    del_x1=(k1+2*k2+2*k3+k4)/6;
    del_x2=(m1+2*m2+2*m3+m4)/6;
    del_x3=(n1+2*n2+2*n3+n4)/6;
    del_x4=(p1+2*p2+2*p3+p4)/6;
    % Saving solution
    solu(i)=x1;
    % Updating variables
    x1=x1+del_x1;
    x2=x2+del_x2;
    x3=x3+del_x3;
    x4=x4+del_x4;
    % Updating iteration time
    x=x+h;
end
% Sum Square Error (SSE)
SSE1v=sum((a-solu').^2);

%##### Lot#2 #####
x1=-0.0522;
x2=0.0092;
x3=-0.176;
x4=-4e-4;
u=u2;
aa=a2;
yy=y2;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
end
for i=2:250

```

```

a2dot(1)=adot(1);
a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);
    ydot(i)=y(i)-y(i-1);
end
for i=2:250
    y2dot(1)=ydot(1);
    y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250
    % Phase I
    k1=h*obj1(x2);

    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

    p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

    m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n3=h*obj3(x4+p2/2);

    p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase IV
    k4=h*obj1(x2+m3);

    m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,
    K,L,M,N,O,m,l,g);
    n4=h*obj3(x4+p3);

    p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K
    ,L,M,N,O,m,l,g);
    % Derivative stage
    del_x1=(k1+2*k2+2*k3+k4)/6;
    del_x2=(m1+2*m2+2*m3+m4)/6;
    del_x3=(n1+2*n2+2*n3+n4)/6;
    del_x4=(p1+2*p2+2*p3+p4)/6;
    % Saving solution
    solu(i)=x1;
    % Updating variables
    x1=x1+del_x1;
    x2=x2+del_x2;
    x3=x3+del_x3;
    x4=x4+del_x4;
    % Updating iteration time
    x=x+h;

```

```

end
% Sum Square Error (SSE)
SSE2i=sum((a-solu').^2);

%##### Lot#3 #####
x1=0.178;
x2=0.1657;
x3=-0.1212;
x4=0.0317;
u=u3;
aa=a3;
yy=y3;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
for i=2:250
    a2dot(1)=adot(1);
    a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);
    ydot(i)=y(i)-y(i-1);
end
for i=2:250
    y2dot(1)=ydot(1);
    y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250
    % Phase I
    k1=h*obj1(x2);

    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

    p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,
    J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

```

```

m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
n3=h*obj3(x4+p2/2);

p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
% Phase IV
k4=h*obj1(x2+m3);

m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
n4=h*obj3(x4+p3);

p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
% Derivative stage
del_x1=(k1+2*k2+2*k3+k4)/6;
del_x2=(m1+2*m2+2*m3+m4)/6;
del_x3=(n1+2*n2+2*n3+n4)/6;
del_x4=(p1+2*p2+2*p3+p4)/6;
% Saving solution
solu(i)=x1;
% Updating variables
x1=x1+del_x1;
x2=x2+del_x2;
x3=x3+del_x3;
x4=x4+del_x4;
% Updating iteration time
x=x+h;
end
% Sum Square Error (SSE)
SSE3i=sum((a-solu').^2);

%##### Lot#4 #####
x1=0.0123;
x2=-0.0123;
x3=-0.0092;
x4=-0.007;
u=u4;
aa=a4;
yy=y4;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
for i=2:250
    a2dot(1)=adot(1);
    a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);

```

```

    ydot(i)=y(i)-y(i-1);
end
for i=2:250
    y2dot(1)=ydot(1);
    y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250
    % Phase I
    k1=h*obj1(x2);

    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

    p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

    m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n3=h*obj3(x4+p2/2);

    p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase IV
    k4=h*obj1(x2+m3);

    m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,
    K,L,M,N,O,m,l,g);
    n4=h*obj3(x4+p3);

    p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K
    ,L,M,N,O,m,l,g);
    % Derivative stage
    del_x1=(k1+2*k2+2*k3+k4)/6;
    del_x2=(m1+2*m2+2*m3+m4)/6;
    del_x3=(n1+2*n2+2*n3+n4)/6;
    del_x4=(p1+2*p2+2*p3+p4)/6;
    % Saving solution
    solu(i)=x1;
    % Updating variables
    x1=x1+del_x1;
    x2=x2+del_x2;
    x3=x3+del_x3;
    x4=x4+del_x4;
    % Updating iteration time
    x=x+h;
end
% Sum Square Error (SSE)
SSE4i=sum((a-solu').^2);

```

```

##### Lot#5 #####
x1=-0.1319;
x2=0.1350;
x3=0.1757;
x4=0.0735;
u=u5;
aa=a5;
yy=y5;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
for i=2:250
    a2dot(1)=adot(1);
    a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);
    ydot(i)=y(i)-y(i-1);
end
for i=2:250
    y2dot(1)=ydot(1);
    y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250
    % Phase I
    k1=h*obj1(x2);
    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

    p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

    m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n3=h*obj3(x4+p2/2);

```

```

p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
% Phase IV

k4=h*obj1(x2+m3);

m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
n4=h*obj3(x4+p3);

p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g);
% Derivative stage
del_x1=(k1+2*k2+2*k3+k4)/6;
del_x2=(m1+2*m2+2*m3+m4)/6;
del_x3=(n1+2*n2+2*n3+n4)/6;
del_x4=(p1+2*p2+2*p3+p4)/6;
% Saving solution
solu(i)=x1;
% Updating variables
x1=x1+del_x1;
x2=x2+del_x2;
x3=x3+del_x3;
x4=x4+del_x4;
% Updating iteration time
x=x+h;
end
% Sum Square Error (SSE)
SSE5i=sum((a-solu').^2);

%##### Lot#6 #####
x1=-0.227;
x2=-0.0153;
x3=0.081;
x4=-0.0077;
u=u6;
aa=a6;
yy=y6;
a=aa(2:251);
y=yy(2:251);
rr=yy/R;
r=rr(2:251);
for i=1:250
    rdot(i)=rr(i+1)-rr(i);
end
for i=2:250
    r2dot(1)=rdot(1);
    r2dot(i)=rdot(i)-rdot(i-1);
end
for i=2:250
    adot(1)=aa(2)-aa(1);
    adot(i)=a(i)-a(i-1);
end
for i=2:250
    a2dot(1)=adot(1);
    a2dot(i)=adot(i)-adot(i-1);
end
for i=2:250
    ydot(1)=yy(2)-yy(1);
    ydot(i)=y(i)-y(i-1);
end
for i=2:250

```

```

y2dot(1)=ydot(1);
y2dot(i)=ydot(i)-ydot(i-1);
end
% Numerical process according to Runge-Kutta approach
for i=1:250

    % Phase I
    k1=h*obj1(x2);

    m1=h*obj2(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    n1=h*obj3(x4);

    p1=h*obj4(x1,x2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,m,l,g);
    % Phase II
    k2=h*obj1(x2+m1/2);

    m2=h*obj2(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n2=h*obj3(x4+p1/2);

    p2=h*obj4(x1+k1/2,x2+m1/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase III
    k3=h*obj1(x2+m2/2);

    m3=h*obj2(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,
    I,J,K,L,M,N,O,m,l,g);
    n3=h*obj3(x4+p2/2);

    p3=h*obj4(x1+k2/2,x2+m2/2,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I
    ,J,K,L,M,N,O,m,l,g);
    % Phase IV
    k4=h*obj1(x2+m3);

    m4=h*obj2(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,
    K,L,M,N,O,m,l,g);
    n4=h*obj3(x4+p3);

    p4=h*obj4(x1+k3,x2+m3,u(i),y(i),ydot(i),y2dot(i),r(i),rdot(i),r2dot(i),a(i),adot(i),a2dot(i),A,B,C,D,E,F,G,H,I,J,
    ,L,M,N,O,m,l,g);
    % Derivative stage
    del_x1=(k1+2*k2+2*k3+k4)/6;
    del_x2=(m1+2*m2+2*m3+m4)/6;
    del_x3=(n1+2*n2+2*n3+n4)/6;
    del_x4=(p1+2*p2+2*p3+p4)/6;
    % Saving solution
    solu(i)=x1;
    % Updating variables
    x1=x1+del_x1;
    x2=x2+del_x2;
    x3=x3+del_x3;
    x4=x4+del_x4;
    % Updating iteration time
    x=x+h;
end
% Sum Square Error (SSE)
SSE6v=sum((a-solu').^2);
%#####

```

```

SSEsi=SSE2i+SSE3i+SSE4i+SSE5i;
SSEsv=SSE1v+SSE6v;
SSEs=SSEsi+SSEsv;
SSE=mean([SSE1v SSE2i SSE3i SSE4i SSE5i SSE6v]);
Return

```

โปรแกรม obj1.m

```

function y=obj1(x2)
    y=x2;
return

```

โปรแกรม obj2.m

```

function y=obj2(x1,x2,u,y,ydot,y2dot,r,rdot,r2dot,a,adot,a2dot,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g)

y=((A*r2dot+B*rdot+C*r+D*y2dot+E*ydot+F*y+G*a2dot+H*adot+I*a+(J*exp(K*u)+L*u^2+M*u+N/u+O))*
cos(x1)+...
    0.5*m*l*x2^2*sin(2*x1)+(0.9549+m)*g*sin(x1))/(l*(m*cos(x1)^2-(0.9549+m)));
Return

```

โปรแกรม obj3.m

```

function y=obj3(x4)
    y=x4;
return

```

โปรแกรม obj4.m

```

function y=obj4(x1,x2,u,y,ydot,y2dot,r,rdot,r2dot,a,adot,a2dot,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,m,l,g)

y=((A*r2dot+B*rdot+C*r+D*y2dot+E*ydot+F*y+G*a2dot+H*adot+I*a+(J*exp(K*u)+L*u^2+M*u+N/u+O))+
(l*m*x2^2*sin(x1)+0.5*g*m*sin(2*x1)))/...
    ((0.9549+m)-m*cos(x1)^2);
return

```

โปรแกรม objective.m

```

function [best_error1,best_neighbor1,best_error,best_neighbor]=objective(S1,best_error,S0)

% โปรแกรม objective เป็นโปรแกรมประเมินค่าใกล้เคียงที่หาค่าใกล้เคียง S0
% อินพุตของโปรแกรมนี้คือ S1, best_error, และ S0
% เอาต์พุตคือ best_error1, best_neighbor1,
% best_error, และ best_neighbor

error=[];
% ตัวแปร error สำหรับรองรับค่าฟังก์ชันวัตถุประสงค์
% ของค่าใกล้เคียงที่ได้จากการหาค่าใกล้เคียง

for k=1:size(S1,1)
    ysim=obj(S1(k,1),S1(k,2),S1(k,3),S1(k,4),S1(k,5),S1(k,6),S1(k,7),S1(k,8),S1(k,9),S1(k,10),S1(k,11),S1(k,12),
        S1(k,13),S1(k,14),S1(k,15));
    % ประเมินค่าใกล้เคียงใน S1 ด้วยฟังก์ชันวัตถุประสงค์ในโปรแกรม obj

    error(k,1)=ysim;
    % เก็บค่าฟังก์ชันวัตถุประสงค์ของค่าใกล้เคียงทั้งหมด

end

[best_error1,index]=min(error);
% หาค่าฟังก์ชันวัตถุประสงค์ที่น้อยที่สุดในบรรดาค่าใกล้เคียง
% และเก็บไว้ใน best_error1

best_neighbor1=S1(index,:);
% เก็บค่าพารามิเตอร์ของค่าใกล้เคียงที่ทำให้
% ค่าฟังก์ชันวัตถุประสงค์มีค่าน้อยที่สุดใน best_neighbor1

```

```

if best_error1<best_error      % ปรับค่า best_error
best_error=best_error1;      % ถ้า best_error1 < best_error
                                % ให้แทนค่า best_error ด้วย best_error1

best_neighbor=S1(index,:);    % และแทนค่าพารามิเตอร์ best_neighbor ด้วย S1(index,:)

else                            % ไม่เช่นนั้นแล้ว
best_neighbor=S0;              % แทนค่า best_neighbor ด้วย S0
end
return

```

โปรแกรม rand1.m

```

function x=rand1(a,b)          % โปรแกรม rand1 เป็นโปรแกรมสุ่มค่าพารามิเตอร์ในช่วงที่กำหนด
% อินพุตของโปรแกรมคือขอบเขตบนของพารามิเตอร์ (b) และ
% ขอบเขตล่างของพารามิเตอร์ (a)
% โดยอาศัยความสัมพันธ์ (b- a)rand + a
% เอาต์พุตของโปรแกรมคือผลของการสุ่มค่าพารามิเตอร์ในช่วง a ถึง b
    x=a+rand*(b-a);
return

```

โปรแกรม random_neigh.m

```

function S1=random_neigh(Number_neighb,radius,xlimit,S0)
% โปรแกรม random_neigh เป็นโปรแกรมสุ่มค่าใกล้เคียงรอบค่าตอบ S0
% อินพุตของโปรแกรมนี้คือ Number_neighb, radius, xlimit, และ S0
% เอาต์พุตคือค่าใกล้เคียง (S1) จำนวนเท่ากับ Number_neighb

    for u=1:Number_neighb
        for k=1:size(xlimit,2)
            S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));

% ทำการสุ่มค่าใกล้เคียงโดยอาศัยความสัมพันธ์
%  $S_{1,i} = S_0 + (\text{radius} \cdot \text{rand1}_i \cdot (-1,1) \cdot (x_{h,i} - x_{l,i}))$ ,
% for i = 1, ... ,Number _ neighb
% เรียกใช้โปรแกรม rand1 โดยให้ทำการสุ่มค่าในช่วง -1 ถึง 1
% นั่นหมายความว่าค่าสุ่มมีพื้นที่อยู่ในช่วง  $S_0 + ((\text{radius}) \cdot (x_{h,i} - x_{l,i}))$ 
            while ( S1(u,k)>xlimit(1,k) | S1(u,k)<xlimit(2,k) )
% ค่าใกล้เคียงที่สุ่มได้จะต้องอยู่ภายในขอบเขตบนและขอบเขตล่าง
% ของค่าพารามิเตอร์ ถ้าค่าใกล้เคียงที่สุ่มได้
% มีค่าเกินขอบเขตบนและขอบเขตล่างที่กำหนดให้ทำการสุ่มค่าใกล้เคียงใหม่
            S1(u,k)=S0(1,k)+(radius*(xlimit(1,k)-xlimit(2,k))*rand1(-1,1));
        end
    end
end
end
return

```



ภาคผนวก ง

รายชื่อบทความวิจัยที่ได้รับการตีพิมพ์

รายชื่อบทความวิจัยที่ได้รับการตีพิมพ์

รายชื่อบทความวิจัยที่ได้รับการตีพิมพ์ในวารสารวิชาการนานาชาติ

Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2008). Management agent for search algorithms with surface optimization applications. **WSEAS Transactions on Computers**. 7(6): 791-803.

Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2009). Performance assessment of search management agent under asymmetrical problems and control design applications. **WSEAS Transactions on Computers**. 8(4): 691-704.

Kluabwang, J., and Sujitjorn, S. (2009). Tabu search: Its modified forms and management agent. **Recent Patents on Electrical Engineering**. 2(2): 135-145.

Sujitjorn, S., Kluabwang, J., Puangdownreong, D., and Sarasiri, N. (2010). Adaptive tabu search and management agent. **ECTI Transactions on Electrical Engineering, Electronics and Communications**. 8(1): 1-10.

รายชื่อบทความวิจัยที่ได้รับการตีพิมพ์ในการประชุมวิชาการนานาชาติ

Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2008). Management agent for search algorithms. In **Proc. WSEAS Int. Conf. on Computers (CSCC'08)**. (pp.375-382).

Kluabwang, J., Puangdownreong, D., and Sujitjorn, S. (2009). Investigation of the effects of asymmetrical problems on the performance of search management agent. In **Proc. WSEAS Int. Conf. on Evolutionary Computing (EC'09)**. (pp.40-45).

Puangdownreong, D., Kluabwang, J., and Sujitjorn, S. (2010). Application of adaptive tabu search under management agent to control synthesis of scaled vehicle. In **Proc. IASTED Int. Conf. on Modelling, Identification and Control (MIC'2010)**. (pp.1-6).

Performance Assessment of Search Management Agent under Asymmetrical Problems and Control Design Applications

Jukkrit Kluabwang¹ Deacha Puangdownreong² and Sarawut Sujitjorn^{1*}

¹School of Electrical Engineering, Institute of Engineering
Suranaree University of Technology
Nakhon Ratchasima, Thailand

²Department of Electrical Engineering, Faculty of Engineering
South-East Asia University
Bangkok, Thailand

* corresponding author: sarawut@sut.ac.th
<http://www.sut.ac.th/engineering/electrical/carg/>

Abstract: - The article presents the performance evaluation of the management agent (MA) containing the adaptive tabu search (ATS) as its search core. In particular, asymmetrical surface optimization problems have been considered. It has been found that symmetrical property of the problems has a significant effect on search performance of the ATS, but MA(ATS). As an average, the MA(ATS) is about 2 times faster than the ATS under both symmetrical and asymmetrical problems. The article also gives reviews on the ATS and the MA(ATS) algorithms. An application on controller design for a coupled system with three-degree-of-freedom is also elaborated.

Key-Words: - asymmetrical surface optimization problems, adaptive tabu search, management agent

1 Introduction

In recent years, intelligent search techniques have become major approaches for solving various optimization problems. Among those, tabu search (TS) has been widely applied for combinatorial optimization problems [1,2]. The TS consists of two main strategies: intensification and diversification [3,4]. Since the TS is very flexible to use for various problems, Sujitjorn et. al. [5,6] have launched a modified version of the TS namely adaptive tabu search (ATS). The ATS possesses the backtracking (BT) and adaptive radius (AR) mechanisms. The former is regarded as a diversification strategy, while the latter as an intensification one. The convergence proof and performance assessment of the ATS can be found in [5,6]. Readers can find ATS's applications, such as electrical system protection in [7], system identification in [8,9], control synthesis in [10,11], and acoustic signal processing in [12]. In 2008, the authors proposed a management agent for search algorithms called shortly as MA($\cdot$). ($\cdot$) can be replaced by any search algorithms such as ATS, simulated annealing, genetic algorithms, etc. The MA contains three major mechanisms: partitioning mechanism (PM),

discarding mechanism (DM), and sequencing mechanism (SM).

This article reports our computing results in order to assess the effects of symmetry of the problems on search performances of the MA(ATS), i.e. the MA having the ATS as its search core. Section 2 of the paper reviews the ATS and the MA. Sections 3 and 4 present our problem formulation, results and discussions, respectively. An application on control system design appears in section 5. Conclusion is in section 6.

2 Reviews of Algorithms

2.1 Adaptive Tabu Search (ATS)

The ATS [5,6] preserves the main search mechanisms of the generic TS [3,4]. Regarding this, the tabu list (TL), the aspiration criteria (AC), and the termination criteria (TC) are effective. The additional mechanisms are the backtracking (BT), and the adaptive search radius (AR), respectively. The BT is for escaping the lock by local solutions via looking backward the TL for some previous solutions and adopting them for generating new search directions. The AR subsequently reduces the search radii in search of a high fidelity solution. The proofs elaborated in [5,6] confirm the ATS's rapid

search performance, and convergence.

The following description summarizes the algorithms.

STEP 0 Initialization: search radius R , TL , $count$, $count_{max}$, TC , BT , and AR .

STEP 1 Random an initial solution S_0 . Set S_0 as a local solution. Generate the neighbourhood of radius R around S_0 . Keep N solutions of the neighbourhood in the set X .

STEP 2 Evaluate the cost, C , of each solution belonging to X . The solution having the minimum cost is designated as S_1 . If $C(S_1) < C(S_0)$, move S_0 into TL and assign $S_0 = S_1$, otherwise move S_1 into TL .

STEP 3 If the solution cycling occurs, invoke the BT mechanism.

STEP 4 Exit with S_0 as the best solution if the TC are met.

STEP 5 If the current solution S_0 in the vicinity of the local or the global minimum, invoke the AR mechanism. Update $count$, and goto STEP 1.

Interested readers can find the useful recommendations with examples for the benefit of employing the algorithms from [5-12].

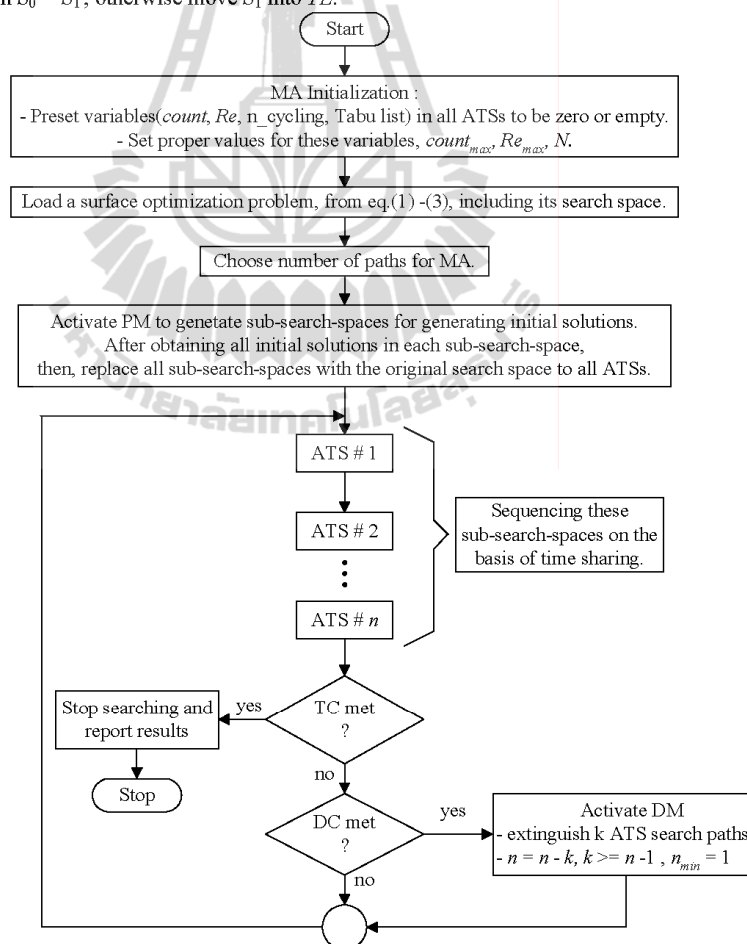


Fig. 1 Flow diagram of the MA. [13,14]

2.2 Management Agent for Search

Recently, the authors have launched the search management agent (MA) flexible enough to be used with any search algorithms, particularly the meta-heuristic ones [13,14]. The MA does not intrude the main algorithms used as the search core unit. So, the convergence property of the search core is always preserved.

The MA consisted of 3 main strategies namely the partitioning, the sequencing, and the discarding mechanisms (PM, SM, and DM), respectively. The PM serves to split an entire search space into many sub-search-spaces. It also initiates searches correspondingly to those sub-search-spaces. After the search initiations, the search boundaries, once arisen from the PM, are immediately removed. The search space is then the original one containing several to many search paths. The PM concept is not new, and has been applied to data fusion [15] and genetic algorithms [16], for instance. The SM on a single CPU platform is crucial because it organizes the search units to perform searches in a sequential manner without a conflict. The SM can be regarded as a time-sharing strategy. As soon as all search paths finish their searches at the k^{th} round, the DM interrupts all search paths, and interrogates them for the current solutions with associated cost values. The DM, apparently behaves similarly to the location management algorithms [17,18], is an important tactic to accelerate the search. It determines the chance of success of each search path, and eliminates those considered to have less chances. For instance, some search paths that are locked by local minima are rapidly deleted from the search plateau. The DM is activated upon the discarding criteria (DC). The DC is simply the measurement of errors between the cost of the global solution and that of the current one. The concept of MA becomes clearer with the flow diagram reproduced from [13,14] and illustrated in Fig. 1. Referring to the flow diagram, the search core implemented is the ATS. In addition to this, the algorithms MA(ATS) can be summarized as follows:

STEP 0 Define search spaces and search paths.

Generate initial solutions (random or well-educated guess). Activate PM.

STEP 1 Invoke SM and ATS of multiple paths (or other search algorithms).

STEP 2 Exit with the best solution if the TC are met, otherwise invoke DM.

STEP 3 Update counter and goto STEP 1.

3 Performance Evaluations

Our evaluation was conducted against the Shekel's Foxholes function (see Figs. 2 and 3) expressed by equation (1). The global solution sits on (-32,-32) coordinates. Its cost value is 1. Fig. 4 shows the top view locations of the global solutions for the symmetrical and asymmetrical problems. Table 1 summarizes the corresponding search spaces. Our MATLABTM codes were run on a Pentium IV 2.4 GHz 640 Mbytes of SD-RAM.

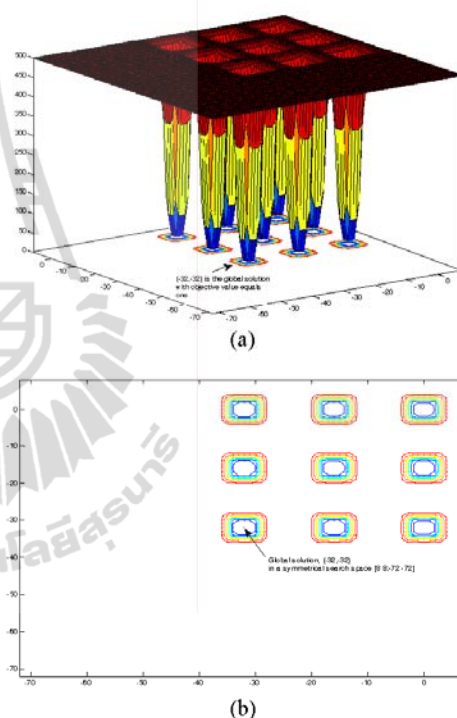


Fig. 2 Shekel's Foxholes surface under symmetrical problem (a) 3-D and (b) 2-D.

$$f(x_1, x_2) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1} \quad (1),$$

in which

$$a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}.$$

Our tests were also conducted against the Bohachevsky's, and the Rastrigin's functions. They

are represented by the expressions (2) and (3), respectively, with their corresponding surface plots shown in Figs. 5 and 6. For asymmetrical problems, the global solutions are eccentric in the similar manner to what shown in Fig. 4(b), otherwise symmetrical.

$$f(x, y) = x^2 + 2y^2 - 0.3 \cos(3\pi x) - 0.4 \cos(4\pi y) + 0.7 \quad (2),$$

$$f(x, y) = x^2 + y^2 - 10 \cos(2\pi x) - 10 \cos(2\pi y) + 20 \quad (3).$$

Table 1 Corresponding search spaces.

Types	search spaces	
	BF & RF	SF
Sym.	[2 2;-2 -2]	[8 8;-72 -72]
Asym.	[3 3;-1 -1]	[40 40;-40 -40]

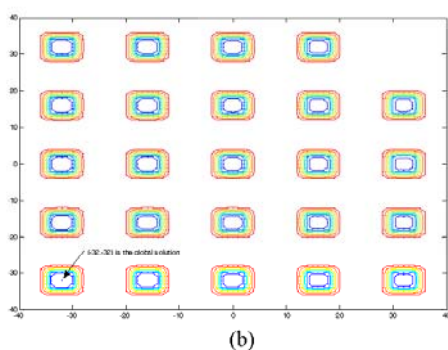
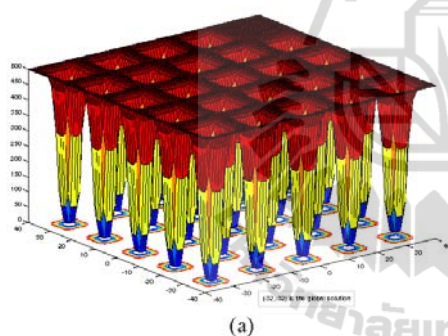


Fig. 3 Shekel's Foxholes surface under asymmetrical problem (a) 3-D and (b) 2-D.

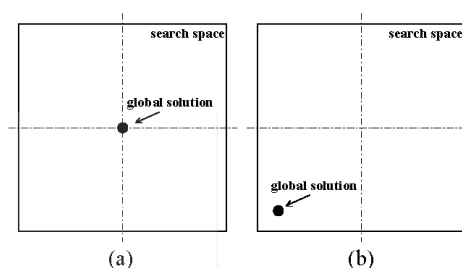


Fig. 4 Top view locations of the global solutions (a) symmetrical and (b) asymmetrical problems.

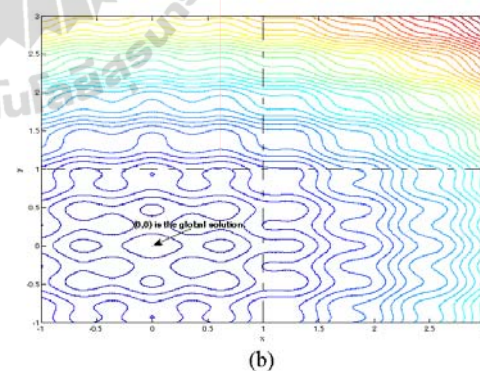
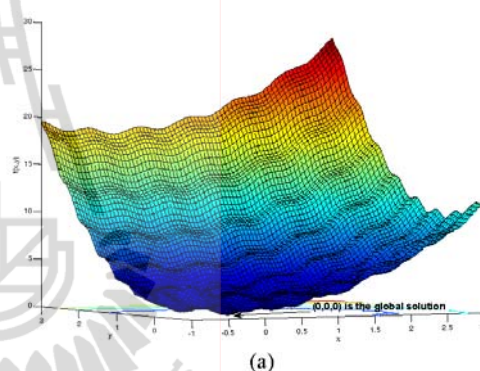


Fig. 5 Bohachevsky's surface under asymmetrical problem (a) 3-D and (b) 2-D.

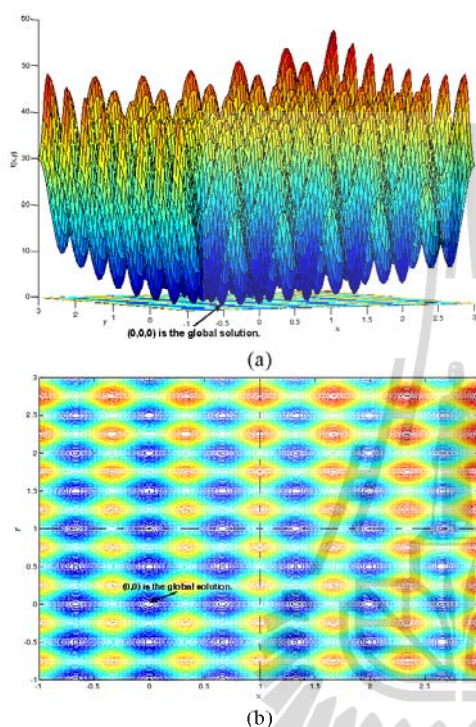


Fig. 6 Rastrigin's surface under asymmetrical problem (a) 3-D and (b) 2-D.

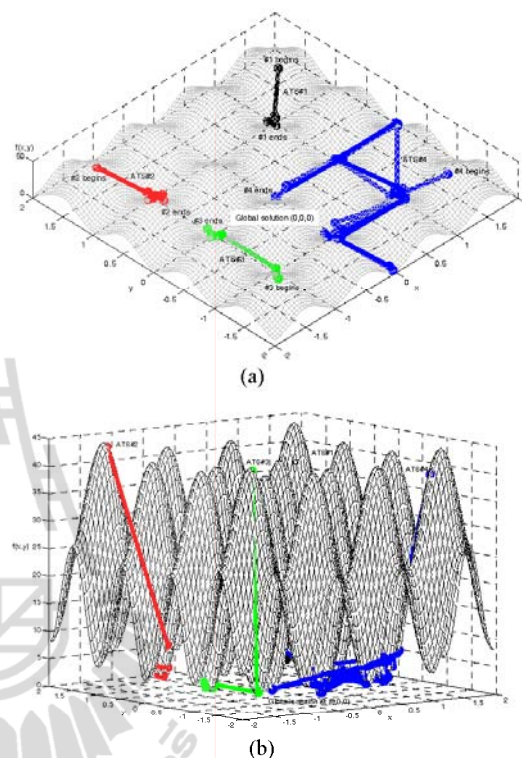


Fig. 7 Search trajectories (4 paths) on Rastrigin's surface (a) bird's eye view and (b) front view.

4 Results and Discussions

Fig. 7 demonstrates the case of 4 search trajectories on the Rastrigin's surface. The ATS#1 and #2 are discarded on the 15th iteration, subsequently ATS#3 on the 25th iteration. The ATS#4 spent 14,184 seconds to hit the global optimum by the 372nd iteration and at $(4.08 \times 10^{-6}, -3.71 \times 10^{-6})$, with the cost value of 0.578×10^{-8} . Moreover, Figs.8-10 illustrate the traces of search trajectories (64 paths) on the three contours of both symmetrical and asymmetrical problems.

Table 2 Average search time.

Test functions	ATS	average search time (seconds)					
		MA(ATS)					
		#2	#4	#8	#16	#32	#64
BF	Sym.	12.8996	11.2039	12.5100	4.0728	4.6429	6.1919
	Asym.	13.8378	12.5574	12.2330	3.9040	4.9901	6.3966
RF	Sym.	14.7950	14.2214	14.5549	12.4603	5.8682	7.5916
	Asym.	19.1713	17.6766	8.5792	15.2682	8.1967	9.4051
SF	Sym.	3.9596	3.9266	3.7271	3.6978	2.3271	2.1720
	Asym.	4.3564	3.0986	3.2957	2.9820	2.4380	3.1135

Referring to the average search time in seconds summarized by table 2, the search time consumed by the ATS is longer than that used by the MA(ATS) for each case. Under asymmetrical problems, the ATS consumes longer search time than that used under symmetrical problems. For the MA(ATS), the following details are clarified. With the BF, 66.67% of MA(ATS), i.e., 2, 16, 32 and 64 paths, reach the global solution faster under symmetrical problems than under asymmetrical ones. Similar situations occur with the RF and the SF. 83.33% of MA(ATS), i.e., 2, 16, 32 and 64 paths, perform faster searches with the symmetrical RF. 50% of MA(ATS), i.e., 16, 32 and 64 paths, perform faster searches with the symmetrical SF.

$$\text{speedup ratios} = \frac{\text{average search time by ATS}}{\text{average search time by MA(ATS)}} \quad (4)$$

The speed up ratios according to (4) are calculated and summarized by the bar graphs shown in Fig. 11.

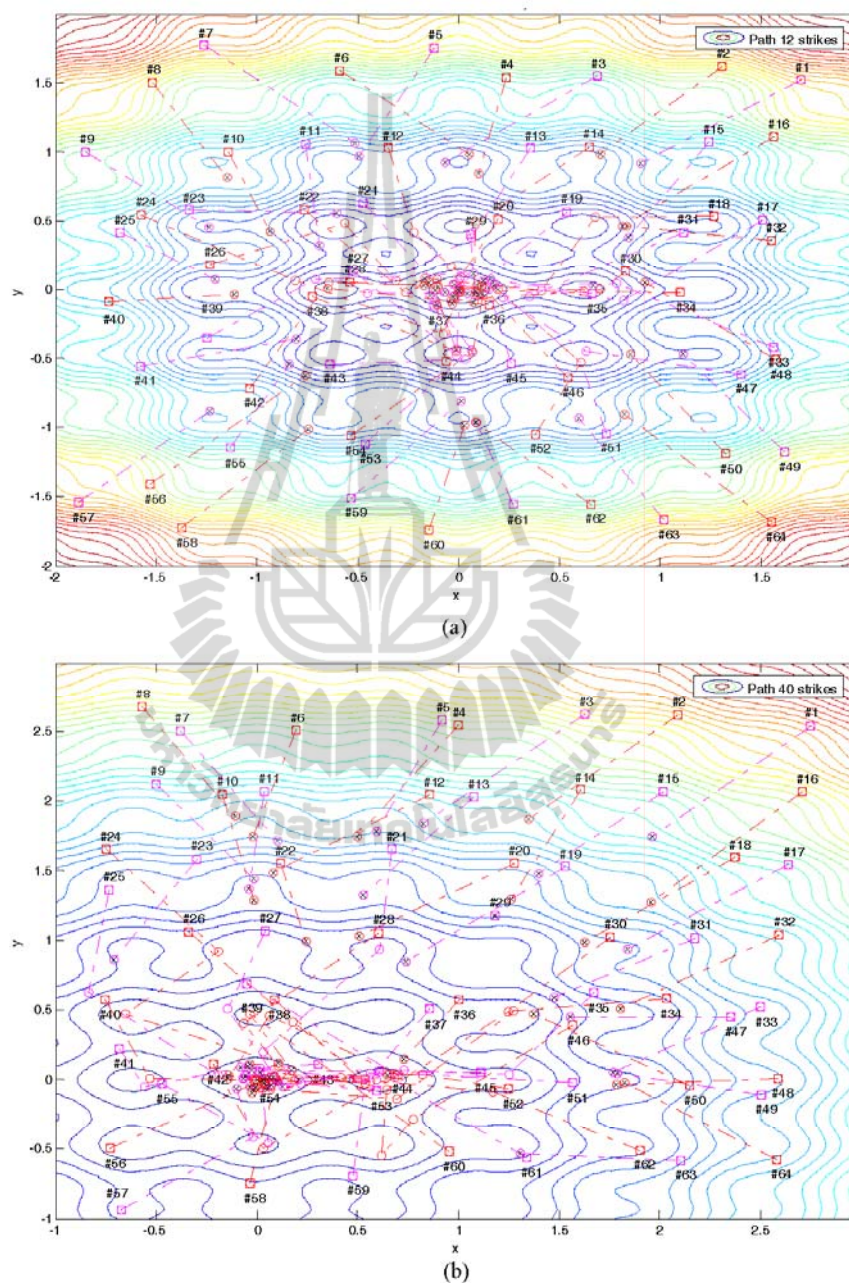


Fig. 8 Traces of search paths (64 paths) on Bohachevsky's contour (a) symmetrical case and (b) asymmetrical case.

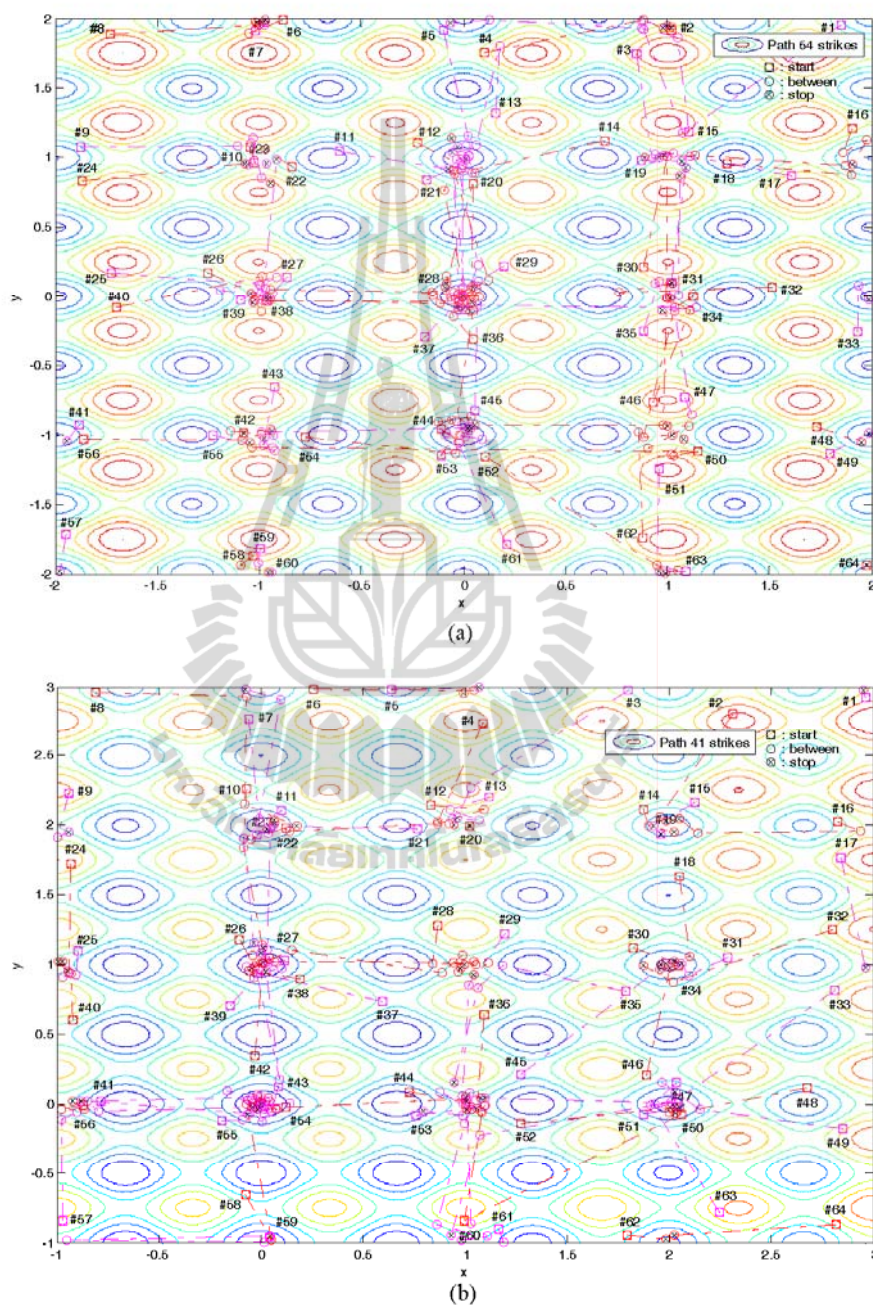


Fig. 9 Traces of search paths (64 paths) on Rastrigin's contour (a) symmetrical case and (b) asymmetrical case.

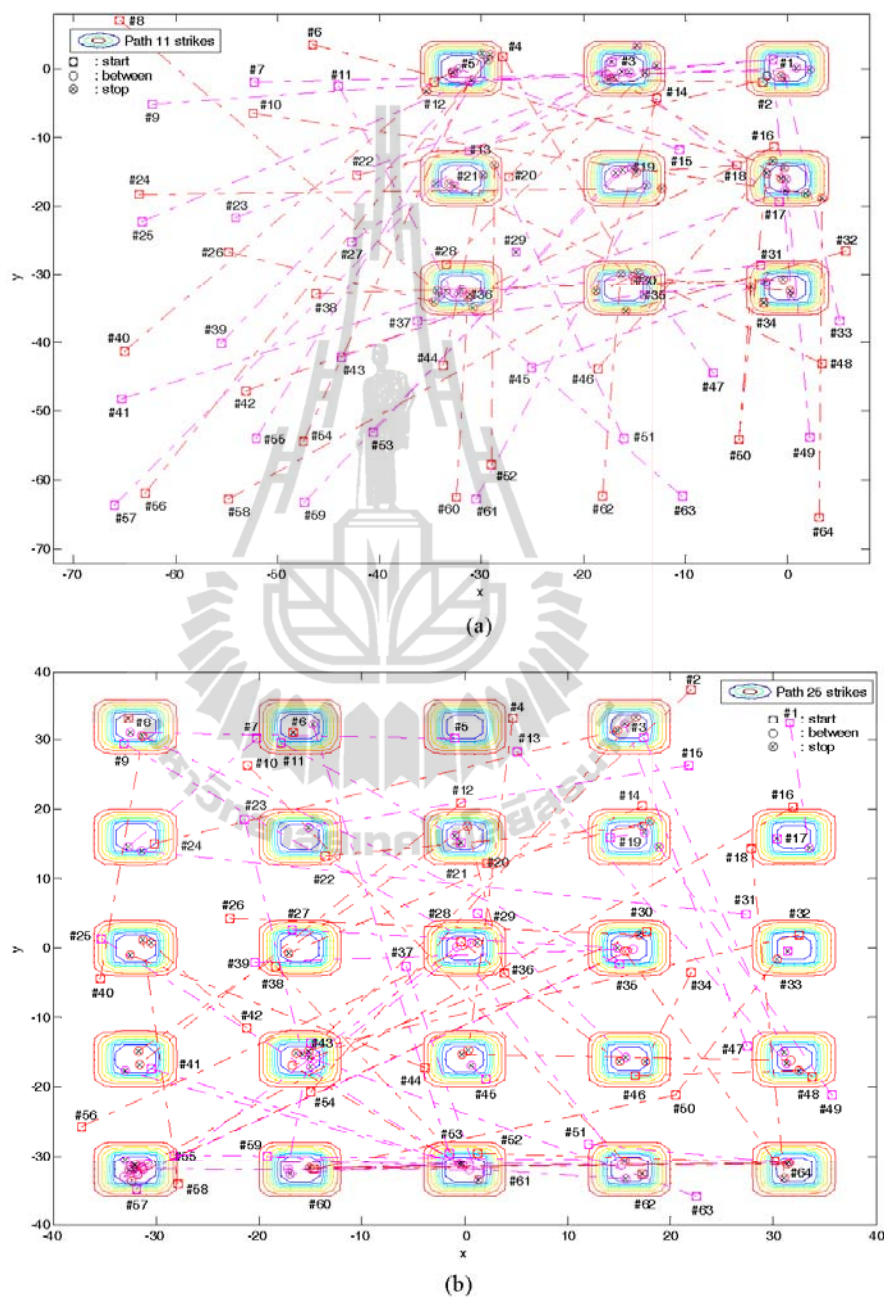


Fig. 10 Traces of search paths (64 paths) on Shekel's foxhole contour (a) symmetrical case and (b) asymmetrical case.

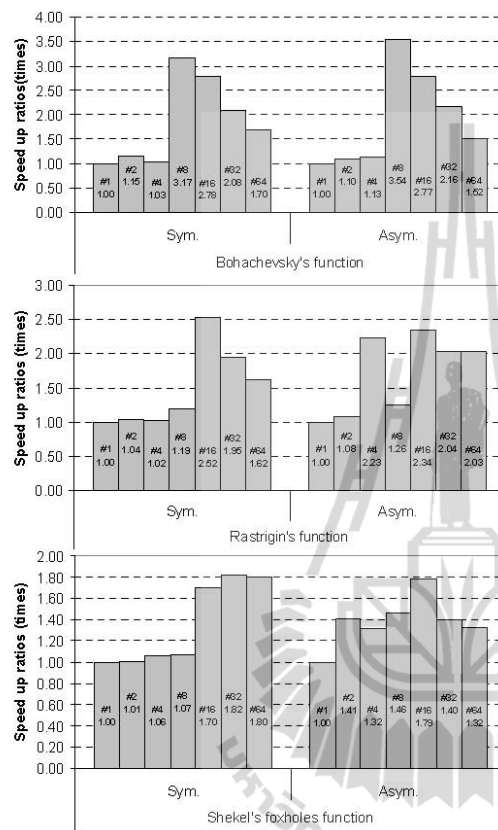


Fig. 11 Speed up ratios of the MA(ATS).

The top window of Fig. 11 shows the results of the MA(ATS) searching through the Bohachevsky's surface. The numeric figures embedded in each bar, for instance, (#1, 1.00) means that it is of the case single path ATS, thus having the speed up ratio equal to 1. As another example, consider the asymmetrical case appearing in the same window, the bar with (#8, 3.54) means the case of 8-path ATS having the speed up ratios of 3.54 times that of the single path ATS. The similar explanation is applied to the rest of the results. Noticeably, the speed up ratios of the symmetrical and the asymmetrical problems bear very similar figures. This means that the symmetrical property of the problems does not greatly affect the search performance of the MA(ATS). To give an average figure, the MA(ATS) performs search in sequential manner about 1.713 times faster than the ATS does. Average speed up ratios of 2,4,...,64 paths

are summarized by table 3. The figures indicate that the MA(ATS) of 16 paths provides the fastest search.

Table 3 Average speed up ratios.

no. of paths	2	4	8	16	32	64
average speed up ratios (times)	1.13	1.30	1.95	2.32	1.91	1.67

5 Applications

Control engineering applications are discussed herein. The problem is to fine tune an existing control system such that optimally seeked controllers could be obtained. The control system of the Illinois Roadway Simulator (IRS) system [19,20] in which the yaw rate control (see Fig. 12) is of prime objective.

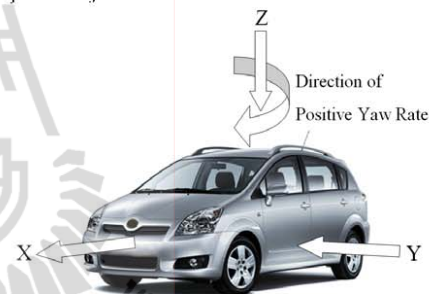


Fig. 12 Car reference frame.

The system can be represented by the block diagram shown in Fig. 13. Referring to Fig.13, two s-polynomial ratios namely G and G_d represent the vehicle dynamics, and expressed by

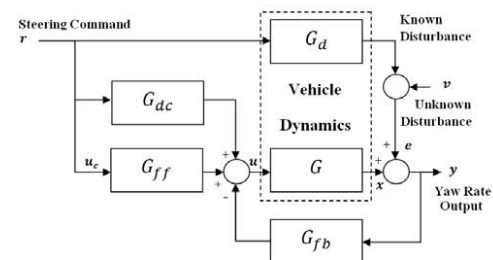


Fig. 13 Block diagram representing the yaw rate control.

$$G = \frac{2.802 \times 10^4 s + 2.306 \times 10^5}{1.5s^4 + 94.43s^3 + 7.308 \times 10^4 s^2 + 1.183 \times 10^5 s + 5.361 \times 10^5} \quad (5)$$

$$G_d = \frac{8.024 \times 10^4 s + 6.710 \times 10^5}{1.3s^4 + 53.250s^3 + 1.943 \times 10^5 s^2 + 2.693 \times 10^4 s + 1.162 \times 10^5} \quad (6).$$

The control system has three degrees-of-freedom, and possesses the following controllers:

$$G_{dc} = \frac{-1.2036 \times 10^5 s^5 - 8.584 \times 10^6 s^4 - 6.497 \times 10^8 s^3 - \dots}{3.6426 \times 10^4 s^5 + 1.792 \times 10^6 s^4 + 6.671 \times 10^7 s^3 + \dots} \quad (7)$$

$$G_g = \frac{103s^2 + 1.8542 \times 10^4 s + 8.34385 \times 10^5}{s^3 + 155.3s^2 + 3.727 \times 10^3 s + 2.0714 \times 10^4} \quad (8)$$

$$G_p = \frac{-5.181s^3 + 49.98s^2 + 5.294 \times 10^3 s + 4.8142 \times 10^4}{s^3 + 155.3s^2 + 3.727 \times 10^3 s + 2.0714 \times 10^4} \quad (9)$$

The transfer function $G_{irs} = \frac{Y(s)}{R(s)} \Big|_{V(z)=0}$ can be written as

$$G_{irs} = \frac{3.075 \times 10^{11} s^{23} + 1.721 \times 10^{14} s^{22} + 4.525 \times 10^{16} s^{21} + \dots}{1.598 \times 10^5 s^{27} + 9.423 \times 10^7 s^{26} + 2.671 \times 10^{10} s^{25} + \dots} \quad (10)$$

Fig. 14 shows the simulation results of the original control system. The yaw rate response due to unit-step input is quite smooth and contains a delay time of 0.005 sec. It is noticed that the system has the DC-gain of 8.6638 rad/sec.

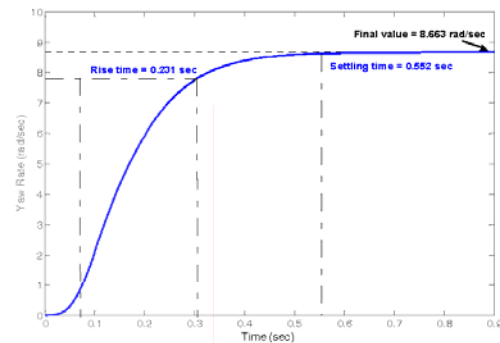


Fig. 14 Yaw rate response of the original system [19,20].

```
function J=obj(r1,r2,r3,r4,s1,s2,s3,s4,t1,t2,t3,a1,a2,a3,a4,a5,a6, b1, b2,...
b3,b4,b5,b6)
% Declarations
G=tf([0.2802e5 2.3057e5],[1.5 94.4284 7.3078e3 1.1831e5 5.3616e5]);
Gd=tf([0.8024e5 6.7103e5],[1.3 53.2495 1.9425e3 2.6928e4 1.1617e5]);
Gdc=tf([b1 b2 b3 b4 b5 b6],[a1 a2 a3 a4 a5 a6]);
Gff=tf([t1 t2 t3],[r1 r2 r3 r4]);
Gfb=tf([s1 s2 s3 s4],[r1 r2 r3 r4]);
Girs=(Gd+G*Gdc+G*Gff)/(1+G*Gfb);
a=real(pole(Girs)); % Poles of overall Closedloop System.
if a<0 % Only Stable System allowed come in.
[y,t]=step(Girs); % Step response data recorded.
if y(length(y))>8.66 & y(length(y))<8.67 % Only same DC gain.
i=find(y>0.1*(y(length(y))-y(1))&y<0.9*(y(length(y))-y(1)));
tr=t(i(length(i)))-t(i(1)); % Find trj.
Mpj=100*(max(y)-y(length(y)))/y(length(y)); % Find Mpj.
for i=1:length(y) % Find tsj.
y(length(y)+1)-y(length(y));
dy(i)=y(i+1)-y(i);
end
i=find(dy<1e-3);
ts=t(i(length(i)))-t(i(1));
tsj=(ts-100)/(0.23096393759434*5); % Constraint 1, trj must be less than 100.
if trj<100 % Constraint 1, otherwise trj = 1,000.
trj=trj;
else
trj=1000;
end
if Mpj<5 % Constraint 2, Mpj must be less than 5.
Mpj=1; % Constraint 2, then Mpj = 1.
else % Constraint 2, otherwise Mpj = 1,000.
Mpj=1000;
end
if tsj<5*trj % Constraint 3, tsj must be < 5*trj.
tsj=1; % Constraint 3, then tsj = 1.
else % Constraint 3, otherwise tsj = 1,000.
tsj=1000;
end
J=trj+Mpj+tsj; % Objective function, J.
else
J=1000;
end
else
J=1000;
end
return % Note that the original system has the objective function J = 102.
```

Fig. 15 Objective function coding.

To obtain 3 controllers that render an optimum response is not a trivial task. In conventional practice, the controllers have to be designed one-by-one with an expectation of having a satisfactory response of the complete system. Synthesis all the controllers simultaneously is possible via an intelligent search method. This control synthesis can be formulated as a combinatorial optimization problem to minimize J , where J is a combined objective function namely

$$J = t_{rj} + M_{pj} + t_{sj} \quad (11).$$

t_{rj} , M_{pj} and t_{sj} in the equation (11) are normalized, hence unitless, and stand for rise time, overshoot and settling time, respectively. In our software implementation, the penalty concept of using 1,000 as a magnification factor is applied to t_r , M_p and t_s in order to obtain a fine quality solution. The step performance specifications are $t_r < 0.231$ sec, $M_p < 5\%$, and t_s defined as the first time the response stops oscillating completely. Moreover, it is mandated that the system maintains the DC-gain. Defining such performance specifications, the MA(ATS) is expected to produce three polynomial controllers optimally rendering a better response than that of the original. Fig. 15 provides the code list of the objective function, J .

Our control problem assumes the nominal plant models whose parameters are fixed. The three controllers namely G_{dc} , G_{ff} and G_{fb} are rewritten as follows:

$$G_{dc} = \frac{b_1 s^5 + b_2 s^4 + b_3 s^3 + b_4 s^2 + b_5 s + b_6}{a_1 s^5 + a_2 s^4 + a_3 s^3 + a_4 s^2 + a_5 s + a_6} \quad (12)$$

$$G_{ff} = \frac{p_1 s^2 + p_2 s + p_3}{r_1 s^3 + r_2 s^2 + r_3 s + r_4} \quad (13)$$

$$G_{fb} = \frac{q_1 s^2 + q_2 s + q_3}{r_1 s^3 + r_2 s^2 + r_3 s + r_4} \quad (14)$$

whose parameter r 's, q 's, p 's, a 's and b 's are to be searched. Regarding to the original design, these 23 parameters are summarized in table 4. Some initial trials referred to as pre-search process were carried out in order to establish search ranges of all influential parameters. The TC for the pre-search

process was either $J < 50$ or $\text{count}_{\max} = 20$. The parameters were searched one-by-one. During the pre-search, one parameter was searched while the others were kept constant as their original values in order to investigate its influence on the system time-domain performance, stability as well as max-min range for search. Referring to the table 4, the parameters shown in bold characters are those influencing the performance while being stability insensitive. Some shown in ordinary characters do not have much influence on the performance, but they are stability sensitive. Table 4 also shows the max-min ranges of the parameters obtained from the pre-search. These ranges are further used as search spaces for the corresponding parameters. Fig. 16 depicts the 13 influential parameters affecting the system performance in terms of the percent reduction in rise time. These parameters printed in bold in the table 4 are to be tuned via our MA(ATS).

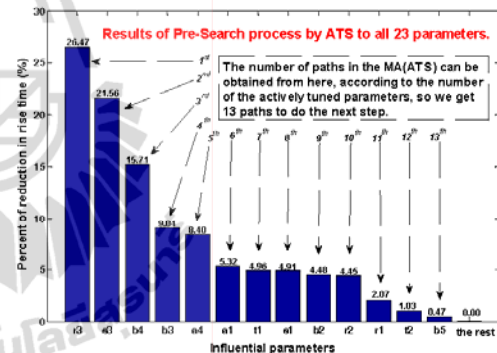


Fig. 16 Results of the pre-search process.

The search for these 13 parameters are distributed among 13 ATS paths with the corresponding search spaces shown by bold characters in the table 4. At the beginning, all ATS paths employ different initial solutions and the following heuristically set search parameters: $N_{\text{neighbour}} = 5$, $R = 0.1$, AR_1 -if $J < 90$ then $R = 0.5R$, AR_2 -if $J < 80$ then $R = 0.1R$, k^{th} backward = 5, $N_{\text{re_max}} = 5$, $\text{max_count} = 100$, TC if $J < 70$ or max_count hit, DM_1 -at 5th iteration reduces 13 to 7 paths, DM_2 -at 10th iteration reduces 7 to 3 paths and DM_3 -at 15th iteration reduces 3 to 1 paths.

Referring to the table 4, 13 ATS paths perform a mixed type of multiple points single strategy (MPSS) and single point single strategy (SPSS) searches. The PM of the MA(ATS) designates the search in the following manner:

- within the 1st neighbourhood, ATS#1 to #13 perform independently search for the individual parameter r_1 , r_2 , r_3 , q_1 , q_3 , p_1 , p_2 , a_1 , a_4 , b_2 , b_3 , b_4 , and

Table 4 Summary of controllers' parameters and corresponding search spaces.

no.		parameters of original controllers	13 parameters obtained from MA(ATS)	max-min ranges of parameters	ATS (at PM)
1	r1	1.0000	0.9548	[1 0.1]	ATS#1
2	r2	155.3000	153.1941	[155.3 0.1]	ATS#2
3	r3	3.727×10^3	2.541×10^3	$[3.727 \times 10^3 - 3.727 \times 10^3]$	ATS#3
4	r4	2.0714×10^4	2.0714×10^4	$[2.0714 \times 10^4 - 2.0714 \times 10^4]$	
5	q1	-5.1810	-5.3916	[-5.1810 -10.1810]	ATS#4
6	q2	49.98	49.98	[49.98 49.98]	
7	q3	5.294×10^3	4.1739×10^3	$[5.294 \times 10^3 - 5.294 \times 10^3]$	ATS#5
8	q4	4.8142×10^4	4.8142×10^4	$[4.8142 \times 10^4 - 4.8142 \times 10^4]$	
9	p1	103	87.9221	[103 -103]	ATS#6
10	p2	1.8542×10^4	1.5448×10^4	$[1.8542 \times 10^4 - 1.8542 \times 10^4]$	ATS#7
11	p3	8.34385×10^5	8.34385×10^5	$[8.34385 \times 10^5 - 8.34385 \times 10^5]$	
12	a1	3.6426×10^4	3.3336×10^4	$[3.6426 \times 10^4 - 0]$	ATS#8
13	a2	1.7918×10^6	1.7918×10^6	$[1.7918 \times 10^6 - 1.7918 \times 10^6]$	
14	a3	6.6708×10^7	6.6708×10^7	$[6.6708 \times 10^7 - 6.6708 \times 10^7]$	
15	a4	1.2024×10^9	1.0774×10^9	$[1.2024 \times 10^9 - 1.2024 \times 10^9]$	ATS#9
16	a5	9.4644×10^9	9.4644×10^9	$[9.4644 \times 10^9 - 9.4644 \times 10^9]$	
17	a6	2.6788×10^{10}	2.6788×10^{10}	$[2.6788 \times 10^{10} - 2.6788 \times 10^{10}]$	
18	b1	-1.2036×10^5	-1.2036×10^5	$[-1.2036 \times 10^5 - 1.2036 \times 10^5]$	
19	b2	-8.5835×10^6	-8.8750×10^6	$[-8.5835 \times 10^6 - 16.5835 \times 10^6]$	ATS#10
20	b3	-6.4974×10^8	-6.5163×10^8	$[-6.4974 \times 10^8 - 7.4974 \times 10^8]$	ATS#11
21	b4	-1.4397×10^{10}	-1.4516×10^{10}	$[-1.4397 \times 10^{10} - 2.4397 \times 10^{10}]$	ATS#12
22	b5	-1.2241×10^{11}	-1.3084×10^{11}	$[-1.2241 \times 10^{11} - 2.2241 \times 10^{11}]$	ATS#13
23	b6	-3.5978×10^{11}	-3.5978×10^{11}	$[-3.5978 \times 10^{11} - 3.5978 \times 10^{11}]$	

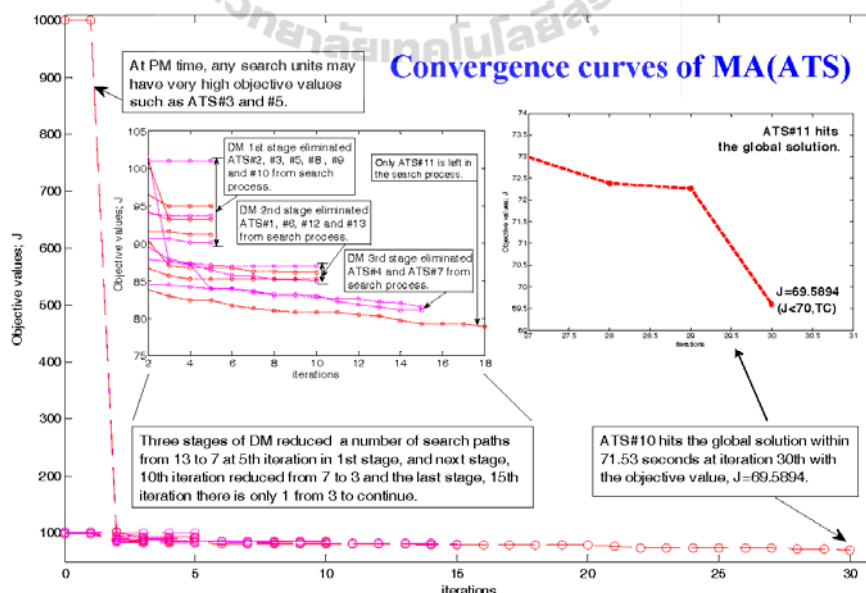


Fig. 17 Convergence curves of the control design problem.

b_5 , respectively; during this search stage, the 22 parameters not being subjected to the search are kept constant as the original values; each ATS path would obtain its best solution to be used as an initial solution for a new search generation;

- after finishing the 1st neighbourhood search, all ATS paths with their corresponding initial solutions for the parameters are released to search freely for all 13 parameters.

The cost values monitored are presented as the convergence curves in Fig. 17. Among the 13 ATS paths, the 11th path succeeded in reaching the solutions with $J=69.5894$, and within 71.53 sec. The parameters obtained from search are shown in bold characters in the table 4. They result in the following controllers

$$G_{dc}^* = \frac{-1.203 \times 10^5 s^5 - 8.875 \times 10^6 s^4 - 6.516 \times 10^8 s^3 - \dots}{3.333 \times 10^4 s^5 + 1.792 \times 10^6 s^4 + 6.671 \times 10^7 s^3 + \dots} \quad (15)$$

$$G_{ff}^* = \frac{87.922s^2 + 1.854 \times 10^4 s + 8.344 \times 10^5}{0.955s^3 + 153.194s^2 + 2.541 \times 10^3 s + 2.071 \times 10^4} \quad (16)$$

$$G_{fb}^* = \frac{-5.391s^3 + 49.98s^2 + 4.174 \times 10^3 s + 4.814 \times 10^4}{0.955s^3 + 153.194s^2 + 2.541 \times 10^3 s + 2.071 \times 10^4} \quad (17)$$

The overall transfer function of the system having the controllers in (15)-(17) is expressed by

$$G_m^* = \frac{-7.628 \times 10^8 s^{24} - 1.877 \times 10^{11} s^{23} + 2.525 \times 10^{13} s^{22} + \dots}{1.333 \times 10^5 s^{27} + 8.060 \times 10^7 s^{26} + 2.295 \times 10^{10} s^{25} + \dots}$$

$$\dots \frac{1.784 \times 10^{16} s^{21} + 3.947 \times 10^{18} s^{20} + 5.535 \times 10^{20} s^{19} + \dots}{4.240 \times 10^{12} s^{24} + 5.772 \times 10^{14} s^{23} + 6.144 \times 10^{16} s^{22} + \dots}$$

$$\dots \frac{5.672 \times 10^{22} s^{18} + 4.504 \times 10^{24} s^{17} + 2.861 \times 10^{26} s^{16} + \dots}{5.307 \times 10^{18} s^{21} + 3.801 \times 10^{20} s^{20} + 2.291 \times 10^{22} s^{19} + \dots}$$

$$\dots \frac{1.477 \times 10^{28} s^{15} + 6.267 \times 10^{29} s^{14} + 2.197 \times 10^{31} s^{13} + \dots}{1.173 \times 10^{24} s^{18} + 5.137 \times 10^{25} s^{17} + 1.930 \times 10^{27} s^{16} + \dots}$$

$$\dots \frac{6.391 \times 10^{32} s^{12} + 1.544 \times 10^{34} s^{11} + 3.094 \times 10^{35} s^{10} + \dots}{6.238 \times 10^{28} s^{15} + 1.734 \times 10^{30} s^{14} + 4.144 \times 10^{31} s^{13} + \dots}$$

$$\dots \frac{5.120 \times 10^{36} s^9 + 6.947 \times 10^{37} s^8 + 7.654 \times 10^{38} s^7 + \dots}{8.493 \times 10^{32} s^{12} + 1.488 \times 10^{34} s^{11} + 2.217 \times 10^{35} s^{10} + \dots}$$

$$\dots \frac{6.762 \times 10^{39} s^6 + 4.705 \times 10^{40} s^5 + 2.513 \times 10^{41} s^4 + \dots}{2.794 \times 10^{36} s^9 + 2.953 \times 10^{37} s^8 + 2.592 \times 10^{38} s^7 + \dots}$$

$$\dots \frac{9.908 \times 10^{41} s^3 + 2.709 \times 10^{42} s^2 + 4.568 \times 10^{42} s + \dots}{1.862 \times 10^{39} s^6 + 1.074 \times 10^{40} s^5 + 4.847 \times 10^{40} s^4 + \dots}$$

$$\dots \frac{3.565 \times 10^{42}}{1.643 \times 10^{41} s^3 + 3.922 \times 10^{41} s^2 + 5.863 \times 10^{41} s + 4.115 \times 10^{41}} \quad (18).$$

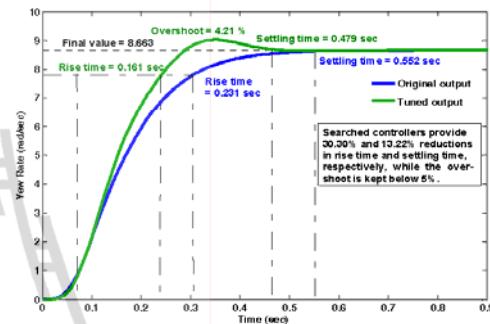


Fig. 18 Comparison of time-domain performances.

Fig. 18 shows the time-domain performances of the original and the proposed systems for comparison. It can be noticed that the new controllers provide much better performance in terms of 30.30% and 13.22% reductions in rise time and settling time, respectively, while the overshoot of 4.21% is acceptable.

6 Conclusions

The performance assessment of the management agent for search, in particular MA(ATS), has been presented. Computing experiments were conducted against symmetrical and asymmetrical problems. In this context, a symmetrical problem has its global solution situated at the center of the search space, while that of an asymmetrical problem is eccentric. Surface optimization problems have been utilized based on the Bohachevsky's, Rastrigin's, and Shekel's Foxholes functions, respectively. The derived speedup ratios indicated that (i) the MA(ATS) under symmetrical problems is 1.01-3.16 times faster than the ATS, and (ii) under asymmetrical problems the MA(ATS) is 1.08-3.54 times faster than the ATS. As an average, the MA(ATS) is about 1.713 times faster than the ATS. The MA(ATS) of 16 paths has the most rapid performance, i.e. 2.32 times the ATS, on surface optimization problems. From the speed up ratios of both cases, we can conclude that the MA(ATS) performs extremely well under symmetrical and asymmetrical problems, it is superior to the ATS, and the symmetrical property of the problems does not have a significant effect on search performance

of the MA(ATS). The method has been successfully applied for a control design problem that requires three controllers. The MA(ATS) with 13 ATS paths spent 71.53 sec. on a Pentium IV platform to finish the search. The control system having the new controllers obtained from search has better step response than the original as elaborated in section 5 of this paper. Opportunities open for further investigations of the search performance of having different search algorithms as the search core of the MA(·).

Acknowledgements

The authors express their thanks to Asst. Prof. Dr. Arthit Sriekaw, School of Electrical Engineering, Suranaree University of Technology for his valuable suggestion on the symmetrical property of search problems. Financial supports from Rajamangala University of Technology Lanna, Tak Campus, and Suranaree University of Technology are very much appreciated.

References:

- [1] F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.
- [2] F. Glover, Parametric tabu search for mixed integer programs, *Computers and Operations Research*, Vol.33, No.9, 2006, pp. 2449-2494.
- [3] F. Glover, Tabu search-Part I, *ORSA Journal on Computing*, Vol.1, No.3, 1989, pp. 190-206.
- [4] F. Glover, Tabu search-Part II, *ORSA Journal on Computing*, Vol.2, No.1, 1990, pp. 4-32.
- [5] S. Sujitjorn, T. Kulworawanichpong and D. Puangdownreong and K-N Areerak, Adaptive tabu search and applications in engineering design, *Book Chapter in Integrated Intelligent Systems for Engineering Design* (ed. X. F. Zha and R.J.Howlett), IOS Press, The Netherlands, 2006, pp. 233-257.
- [6] D. Puangdownreong, T. Kulworawanichpong and S. Sujitjorn, Finite convergence and performance evaluation of adaptive tabu search, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [7] K-N. Areerak, T. Kulworawanichpong and S. Sujitjorn, Moving towards a new era of intelligent protection through digital relaying in systems, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [8] T. Kulworawanichpong, K-L. Areerak, K-N. Areerak and S. Sujitjorn, Harmonic identification for active power filters via adaptive tabu search, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [9] D. Puangdownreong and S. Sujitjorn, Image approach to system identification, *WSEAS Transactions on Systems*, Vol.5, No.9, 2006, pp. 930-938.
- [10] D. Puangdownreong and S. Sujitjorn, Obtaining an optimum PID controller using adaptive tabu search, *Lecture Note in Computer Science*, Springer-Verlag Heidelberg, 2007. (online)
- [11] D. Puangdownreong, T. Kulworawanichpong and S. Sujitjorn, Input weighting optimization for PID controllers based on the adaptive tabu search, *Proc. IEEE TENCON 2004*, Vol. 4, 2004, pp. 451-454.
- [12] N. Sriyingyong, and K. Attakitmongcol, Wavelet-based audio watermarking using adaptive tabu search, *Proc. Wireless Pervasive Computing, 2006 1st International Symposium on*, pp. 1-5.
- [13] J. Kluabwang, D. Puangdownreong and S. Sujitjorn, Management agent for search algorithms with surface optimization applications, *WSEAS Transactions on Computers*, Vol.7, No.6, 2008, pp. 791-803.
- [14] J. Kluabwang, D. Puangdownreong and S. Sujitjorn, Management agent for search algorithms, *In Proc. 12th WSEAS Int. Conf. on Computers*, Vol.2, No.2, 2008, pp.375-382.
- [15] X. Haidong and L. Jianhua, Partition of computing granularity of security data fusion under high performance computing environment *WSEAS Transactions on Computers*, Vol.5, No.1, 2006, pp. 61-68.
- [16] L. Phillip and R. Bart, Problem partitioning in hybrid genetic algorithms, *WSEAS Transactions on Systems*, Vol.6, No.2, 2007, pp. 395-399.
- [17] R-B. Patel, M. Nikos and K. Garq, Mobile agent location management in global network, *WSEAS Transactions on Computers*, Vol.4, No.7, 2005, pp. 697-710.
- [18] R.B. Patel, and M. Nikos, Locating mobile agents in a wide distributed system: a dynamic approach, *WSEAS Transactions on Mathematics*, Vol.4, No.3, 2005, pp. 246-259.
- [19] S. Brennan, and A. Alleyne, The Illinois roadway simulator: a mechatronic testbed for vehicle dynamics and control, *IEEE/ASME Transaction on Mechatronics*, Vol.5, No.4, 2000, pp.349-359.
- [20] S. Brennan, and A. Alleyne, Using a scaled testbed: controller design and evaluation, *IEEE Control Systems Magazine*, Vol.21, No.3, 2001, pp.15-26.

Investigation of the Effects of Asymmetrical Problems on the Performance of Search Management Agent

Jukkrit Kluabwang¹ Deacha Puangdownreong² and Sarawut Sujitjorn^{1*}

¹School of Electrical Engineering, Institute of Engineering
Suranaree University of Technology
Nakhon Ratchasima, Thailand

²Department of Electrical Engineering, Faculty of Engineering
South-East Asia University
Bangkok, Thailand

* corresponding author: sarawut@sut.ac.th
<http://www.sut.ac.th/engineering/electrical/carg/>

Abstract: - The article presents the performance evaluation of the management agent (MA) containing the adaptive tabu search (ATS) as its search core. In particular, asymmetrical surface optimization problems have been considered. It has been found that symmetrical property of the problems has a significant effect on search performance of the ATS, but MA(ATS). On an average, the MA(ATS) is about 2 times faster than the ATS under both symmetrical and asymmetrical problems. The article also gives reviews on the ATS and the MA(ATS) algorithms.

Key-Words: - asymmetrical surface optimization problems, adaptive tabu search, management agent

1 Introduction

In recent years, intelligent search techniques have become major approaches for solving various optimization problems. Among those, tabu search (TS) has been widely applied for combinatorial optimization problems [1,2]. The TS consists of two main strategies: intensification and diversification [3,4]. Since the TS is very flexible to use for various problems, Sujitjorn et. al. [5,6] have launched a modified version of the TS namely adaptive tabu search (ATS). The ATS possesses the back-tracking (BT) and adaptive radius (AR) mechanisms. The former is regarded as a diversification strategy, while the later as an intensification one. The convergence proof and performance assessment of the ATS can be found in [5,6]. Readers can find ATS's applications, such as electrical system protection in [7], system identification in [8,9], control synthesis in [10,11], and acoustic signal processing in [12]. In 2008, the authors proposed a management agent for search algorithms called shortly as MA($\cdot$). ($\cdot$) can be replaced by any search algorithms such as ATS, simulated annealing, genetic algorithms, etc. The MA contains three major mechanisms: partitioning mechanism (PM), discarding mechanism (DM), and sequencing mechanism (SM).

anism (SM).

This article reports our computing results in order to assess the effects of symmetry of the problems on search performances of the MA(ATS), i.e. the MA having the ATS as its search one. Section II of the paper reviews the ATS and the MA. Sections III and IV present our results and discussions, respectively. Conclusion is in Section V.

2 Algorithms Reviews

2.1 Adaptive Tabu Search (ATS)

The ATS [5,6] preserves the main search mechanisms of the generic TS [3,4]. Regarding this, the tabu list (TL), the aspiration criteria (AC), and the termination criteria (TC) are effective. The additional mechanisms are the back-tracking (BT), and the adaptive radius search (AR), respectively. The BT is for escaping the lock by local solutions via looking backward the TL for some previous solutions and adopting them for new search directions. The AR subsequently reduces the search radii in search of a high fidelity solution. The proofs elaborated in [5,6] confirm the ATS's rapid search performance, and convergence.

The following description summarizes the algorithms.

STEP 0 Initialization: search radius R , TL , $count$, and $count_{max}$.

STEP 1 Random an initial solution S_0 . Set S_0 as a local solution. Generate the neighbourhood of radius R around S_0 . Keep N solutions of the neighbourhood in the set X .

STEP 2 Evaluate the cost, C , of each solution belonging to X . The solution having the minimum cost is designated as S_1 . If $C(S_1) < C(S_0)$, move S_0 into TL and assign $S_0 = S_1$, otherwise move S_1 into TL .

STEP 3 If the solution cycling occurs, invoke the BT mechanism.

STEP 4 Exit with S_0 as the best solution if the TC are met.

STEP 5 If the current solution S_0 in the vicinity of the local or the global minimum, invoke the AR mechanism. Update count, and goto STEP 1.

Interested readers can find the useful recommendations with examples for the benefit of employing the algorithms from [5-12].

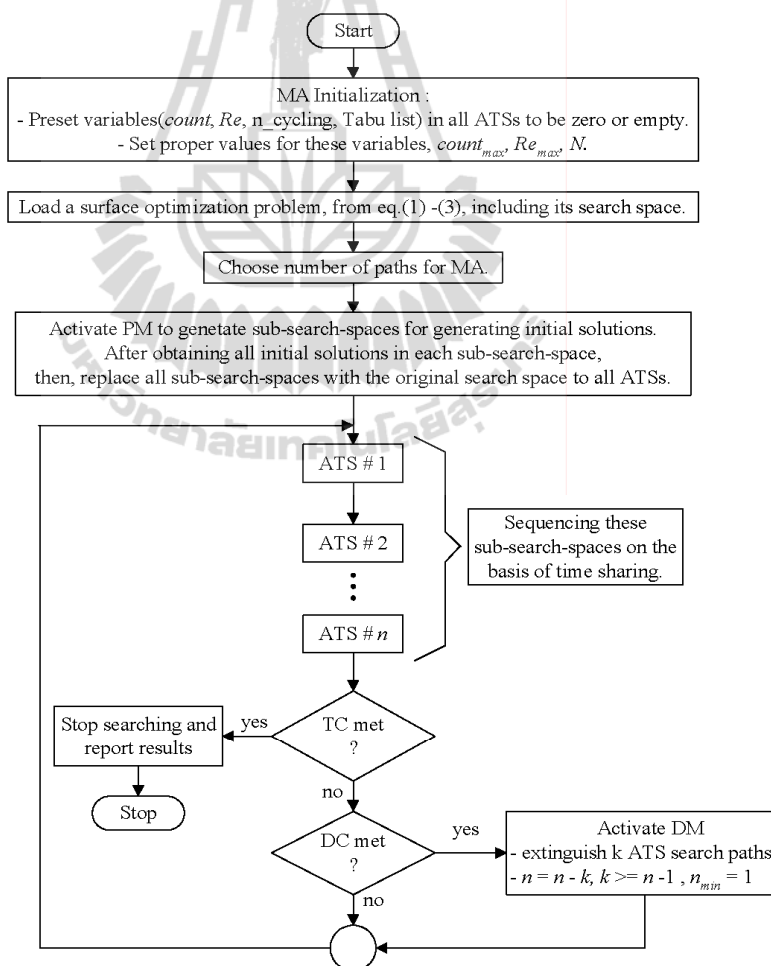


Fig. 1 Flow diagram of the MA. [13]

2.2 Management Agent for Search

Recently, the authors have launched the search management agent (MA) flexible enough to be used with any search algorithms, particularly the meta-heuristic ones [13,14]. The MA does not disturb the main algorithms used as the search core unit. So, the convergence property of the search core is always preserved.

The MA consisted of 3 main strategies namely the partitioning, the sequencing, and the discarding mechanisms (PM, SM, and DM), respectively. The PM serves to split an entire search space into many sub-search-spaces. It also initiates searches correspondingly to those sub-search-spaces. After the search initiations, the search boundaries, once arisen from the PM, are immediately removed. The search space is then the original one containing several to many search paths. The PM concept is not new, and has been applied to data fusion [15] and genetic algorithms [16], for instance. The SM on a single CPU platform is crucial because it organizes the search units to perform searches in a sequential manner without a conflict. The SM can be regarded as a time-sharing strategy. As soon as all search paths finish their searches at the k^{th} round, the SM exports the search results to the DM. The DM, apparently behaves similarly to the location management algorithms [17,18], is an important tactic to accelerate the search. It determines the chance of success of each search path, and eliminates those considered to have less chances. For instance, some search paths that are locked by local minima are rapidly deleted from the search plateau. The DM is activated upon the discarding criteria (DC). The DC is simply the measurement of errors between the cost of the global solution and that of the current one. The concept of MA becomes clearer with the flow diagram reproduced from [13,14] and illustrated in Figure 1. Referring to the flow diagram, the search core implemented is the ATS. In addition to this, the algorithms MA(ATS) can be summarized as follows:

- STEP 0** Define search spaces and search paths. Generate initial solutions (random or well-educated guess). Activate PM.
- STEP 1** Invoke SM and ATS of multiple paths (or other search algorithms).
- STEP 2** Exit with the best solution if the TC are met, otherwise invoke DM.
- STEP 3** Update counter and goto STEP 1.

3 Performance Evaluations

Our evaluation was conducted against the Shekel's Foxholes function (see Figs. 2 and 3) expressed by equation (1). The global solution sits on (-32,-32) coordinates. Its cost value is 1. Figure 4 shows the top view locations of the global solutions for the symmetrical and asymmetrical problems. Table 1 summarizes the corresponding search spaces. Our MATLABTM codes were run on a Pentium IV 2.4 GHz 640 Mbytes of SD-RAM.

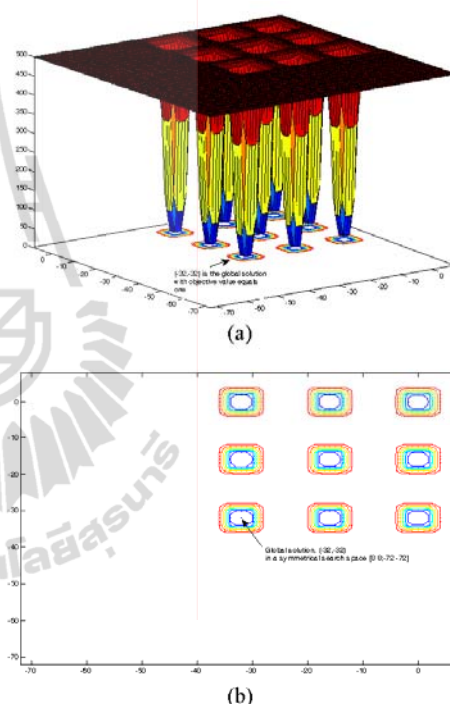


Fig. 2 Shekel's Foxholes surface under symmetrical problem (a) 3-D and (b) 2-D.

$$f(x_1, x_2) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})^6} \right]^{-1} \quad (1),$$

in which

$$a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}.$$

Our tests were also conducted against the Bohachevsky's, and the Rastrigin's functions,

respectively. The surface plots are not shown here. However, they are represented by the expressions (2) and (3), respectively.

$$f(x, y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7 \quad (2),$$

$$f(x, y) = x^2 + y^2 - 10\cos(2\pi x) - 10\cos(2\pi y) + 20 \quad (3).$$

Table 1 Corresponding search spaces.

Types	search spaces	
	BF & RF	SF
Sym.	[2 2; -2 -2]	[8 8; -72 -72]
Asym.	[3 3; -1 -1]	[40 40; -40 -40]

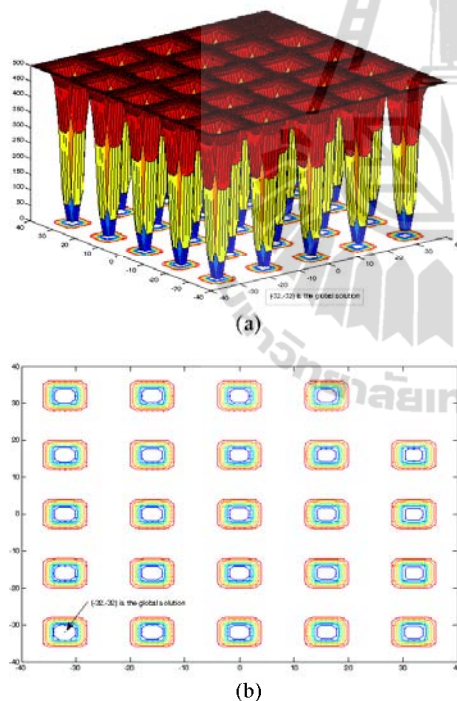


Fig. 3 Shekel's Foxholes surface under asymmetrical problem (a) 3-D and (b) 2-D.

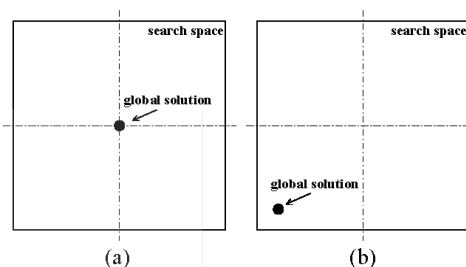


Fig. 4 Top view locations of the global solutions (a) symmetrical and (b) asymmetrical problems.

4 Results and Discussions

Referring to the average search time in seconds summarized by Tables 2, the search time consumed by the ATS is longer than that used by the MA(ATS) for each case. Under asymmetrical problems, the ATS consumes longer search time than that used under symmetrical problems. For the MA(ATS), the following details are clarified. With the BF, 66.67% of MA(ATS), i.e., 2, 16, 32 and 64 paths, reach the global solution faster under symmetrical problems than under asymmetrical ones. Similar situations occur with the RF and the SF. 83.33% of MA(ATS), i.e., 2, 8, 16, 32 and 64 paths, perform faster searches with the symmetrical RF. 50% of MA(ATS), i.e., 16, 32 and 64 paths, perform faster searches with the symmetrical SF. The speed-up ratios according to (4) are calculated and summarized by the bar graph shown in figure 5.

Table 2 Average search time.

Test functions		average search time (seconds)						
		ATS	MA(ATS)					
			#2	#4	#8	#16	#32	#64
BF	Sym.	12.8996	11.2039	12.5100	4.0728	4.6429	6.1919	7.6006
	Asym.	13.8378	12.5574	12.2330	3.9040	4.9901	6.3966	9.1036
RF	Sym.	14.7950	14.2214	14.5549	12.4603	5.8682	7.5916	9.1136
	Asym.	19.1713	17.6766	8.5792	15.2682	8.1967	9.4051	9.4274
SF	Sym.	3.9596	3.9266	3.7271	3.6978	2.3271	2.1720	2.1988
	Asym.	4.3564	3.0986	3.2957	2.9820	2.4380	3.1135	3.2915

$$\text{speed up ratios} = \frac{\text{average searchtime by ATS}}{\text{average searchtime by MA(ATS)}} \quad (4)$$

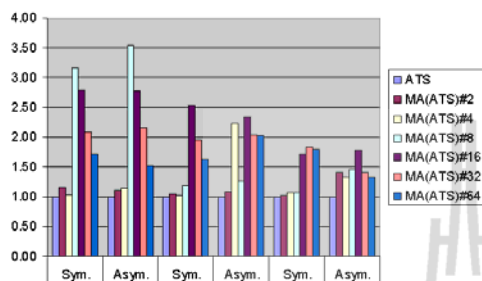


Fig. 5 Comparison of speed up ratios.

5 Conclusions

The performance assessment of the management agent for search, in particular MA(ATS), has been presented. Computing experiments were conducted against symmetrical and asymmetrical problems. In this context, a symmetrical problem has its global solution situated at the center of the search space, while that of an asymmetrical problem is eccentric. Surface optimization problem has been utilized based on the Bohachevsky's, Rastrigin's, and Shekel's Foxholes functions, respectively. The derived speed-up ratios indicated that (i) the MA(ATS) under symmetrical problems is 1.01-3.16 times faster than the ATS, and (ii) under asymmetrical problems the MA(ATS) is 1.08-3.54 times faster than the ATS. From the speed-up ratios of both cases, we can conclude that the MA(ATS) performs extremely well under symmetrical and asymmetrical problems, its is superior to the ATS, and the symmetrical property of the problems does not have a significant effect on search performance of the MA(ATS).

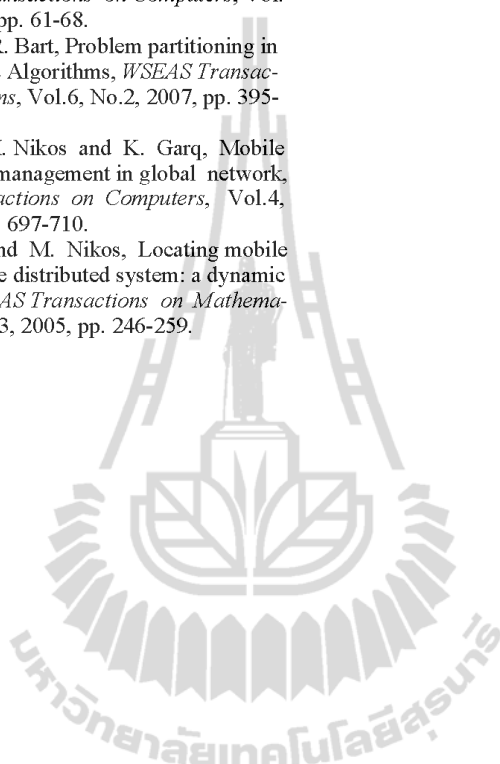
Acknowledgements

The authors express their thanks to Asst. Prof. Dr. Arthit Srikeaw, School of Electrical Engineering, Suranaree University of Technology for his valuable suggestion on the symmetrical property of search problems. Financial supports from Rajamangala University of Technology Lanna, Tak Campus, and Suranaree University of Technology are very much appreciated.

References:

- [1] F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.
- [2] F. Glover, Parametric Tabu Search for Mixed Integer Programs, *Computers and Operations Research*, Vol.33, No.9, 2006, pp. 2449-2494.
- [3] F. Glover, Tabu search-Part I, *ORSA Journal on Computing*, Vol.1, No.3, 1989, pp. 190-206.
- [4] F. Glover, Tabu search-Part II, *ORSA Journal on Computing*, Vol.2, No.1, 1990, pp. 4-32.
- [5] S. Sujitjorn, T. Kulworawanichpong and D. Puangdownreong and K-N Areerak, Adaptive tabu search and applications in engineering design, *Book Chapters in Integrated Intelligent Systems for Engineering Design* (ed. X. F. Zha and R.J.Howlett), IOS Press, The Netherlands, 2006, pp. 233-257.
- [6] D. Puangdownreong, T. Kulworawanichpong and S. Sujitjorn, Finite convergence and performance evaluation of adaptive tabu search, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [7] K-N. Areerak, T. Kulworawanichpong and S. Sujitjorn, Moving towards a new era of intelligent protection through digital relaying in systems, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [8] T. Kulworawanichpong, K-L. Areerak, K-N. Areerak and S. Sujitjorn, Harmonic identification for active power filters via adaptive tabu search, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, 2004. (online)
- [9] D. Puangdownreong and S. Sujitjorn, Image approach to system identification, *WSEAS Transactions on Systems*, Vol.5, No.9, 2006, pp. 930-938.
- [10] D. Puangdownreong and S. Sujitjorn, Optimizing an optimum PID controller using adaptive tabu search, *Lecture Note in Computer Science*, Springer-Verlag Heidelberg, 2007. (online)
- [11] D. Puangdownreong, T. Kulworawanichpong and S. Sujitjorn, Input weighting optimization for PID controllers based on the adaptive tabu search, *Proc. IEEE TENCON 2004*, Vol. 4, 2004, pp. 451-454.
- [12] N. Sriyingyong, and K. Attakitmongcol, Wavelet-based audio watermarking using adaptive tabu search, *Proc. Wireless Pervasive Computing, 2006 1st International Symposium on*, pp. 1-5.
- [13] J. Kluabwang, D. Puangdownreong and S. Sujitjorn, Management agent for search algorithms with surface optimization applications, *WSEAS Transactions on Computers*, Vol.7, No.6, 2008, pp. 791-803.
- [14] J. Kluabwang, D. Puangdownreong and S. Sujitjorn, Management agent for search algorithms, *In Proc. 12th WSEAS Int. Conf. on Computers*, Vol.2, No.2, 2008, pp.375-382.

- [15] X. Haidong and L. Jianhua, Partition of computing granularity of security data fusion under high performance computing environment, *WSEAS Transactions on Computers*, Vol. 5, No.1, 2006, pp. 61-68.
- [16] L. Phillip and R. Bart, Problem partitioning in Hybrid Genetic Algorithms, *WSEAS Transactions on Systems*, Vol.6, No.2, 2007, pp. 395-399.
- [17] R-B. Patel, M. Nikos and K. Garq, Mobile agent location management in global network, *WSEAS Transactions on Computers*, Vol.4, No.7, 2005, pp. 697-710.
- [18] R.B. Patel, and M. Nikos, Locating mobile agents in a wide distributed system: a dynamic approach, *WSEAS Transactions on Mathematics*, Vol.4, No.3, 2005, pp. 246-259.



ประวัติผู้เขียน

นายจักรกฤษณ์ เคลือบวัง เกิดเมื่อวันที่ 12 มกราคม พ.ศ. 2516 ที่โรงพยาบาลจุฬาลงกรณ์ เขตปทุมวัน กรุงเทพมหานคร ประสพการณ์การศึกษา สำเร็จการศึกษาระดับปริญญาตรี สาขาวิศวกรรมไฟฟ้า จากมหาวิทยาลัยเชียงใหม่ เมื่อเดือนตุลาคม พ.ศ. 2538 และระดับปริญญาโท สาขาวิศวกรรมไฟฟ้า จากจุฬาลงกรณ์มหาวิทยาลัย เดือนตุลาคม พ.ศ. 2544 ปัจจุบันกำลังศึกษาในระดับปริญญาเอก สาขาวิชาวิศวกรรมไฟฟ้า ณ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา ประสพการณ์ทำงาน ปี พ.ศ. 2538 ทำงานตำแหน่งวิศวกรสนาม บริษัทวีเคเคจำกัด จังหวัดเชียงใหม่ ปี พ.ศ. 2539 ทำงานตำแหน่งวิศวกรคุณภาพ บริษัทอัลฟาซอร์สจำกัด (มหาชน) จังหวัดปทุมธานี และ ปี พ.ศ. 2540 เริ่มเข้ารับราชการตำแหน่งอาจารย์ 1 ระดับ 3 ของสถาบันเทคโนโลยีราชมงคล วิทยาเขตตาก จังหวัดตาก ปัจจุบันรับราชการตำแหน่งอาจารย์ ระดับ 5 สังกัดภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา เขตพื้นที่ตาก ด้านงานวิจัย มีความสนใจในการประยุกต์ใช้เทคนิคทางปัญญาประดิษฐ์เพื่อแก้ปัญหาทางวิศวกรรม โดยเฉพาะงานด้านอิเล็กทรอนิกส์กำลัง