ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายาก

น.ส.กมลลักษณ์ สุขเสน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2564

QUANTUM COMPACT GENETIC ALGORITHM FOR HARD PROBLEMS

Miss Kamonluk Suksen

A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy in Computer Engineering
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2021

หัวข้อวิทยานิพนธ์	ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายาก
โดย	น.ส.กมลลักษณ์ สุขเสน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

..... กรรมการ
(รองศาสตราจารย์ ดร.ธนารัตน์ ชลิดาพงศ์)

..... กรรมการ
(รองศาสตราจารย์ ดร.ดวงดาว วิชาดากุล)

..... กรรมการภายนอกมหาวิทยาลัย
(ศาสตราจารย์ ดร.บุญเจริญ ศิริเนาวกุล)

กมลลักษณ์ สุขเสน : ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายาก. (QUANTUM COMPACT GENETIC ALGORITHM FOR HARD PROBLEMS) อ. ที่ปรึกษาหลัก : ศ. ดร.ประภาส จงสถิตย์วัฒนา

งานวิจัยทางด้านคอมพิวเตอร์เชิงควอนตัม (Quantum computer) ยังคงเป็นความท้าทายสำหรับนักวิจัยในการพัฒนาเทคนิคต่างๆเพื่อให้การประมวลผลข้อมูลควอนตัมขนาดใหญ่สามารถทำได้จริง มีงานวิจัยหลากหลายสาขาเกี่ยวกับการจำลองระบบควอนตัม โดยเฉพาะด้านอัลกอริทึมควอนตัมสำหรับแก้ปัญหา NP-hard ซึ่งใช้เวลาแก้ปัญหานานเกินกว่าจะเป็นไปได้จริงในเครื่องคอมพิวเตอร์ดั้งเดิม

งานวิจัยนี้นำเสนอขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม ซึ่งเป็นการนำข้อได้เปรียบจากการประมวลผลเชิงควอนตัม ได้แก่ สภาวะซ้อนทับของสถานะควอนตัม และการประมวลผลควอนตัมแบบขนานในอัลกอริทึมการค้นหาของโกรเวอร์ (Grover's search algorithm) มาประยุกต์ใช้ในกระบวนการคัดเลือกโครโมโซมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกโครโมโซมที่ดี เพื่อให้ได้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่มีประสิทธิภาพดีขึ้นในแง่ของความถูกต้องของคำตอบ ขั้นตอนวิธีที่นำเสนอสามารถแก้ปัญหา traveling salesman ขนาดเล็กได้บนเครื่องจำลองคอมพิวเตอร์ควอนตัม เนื่องจากจำนวนคิวบิตที่มีอย่างจำกัดจึงไม่สามารถทำการทดลองกับปัญหา traveling salesman ขนาดใหญ่ได้ แม้ว่าจำนวนฟังก์ชันที่ใช้ในการประเมินค่าความเหมาะสมของขั้นตอนวิธีที่นำเสนอมีแนวโน้มเพิ่มขึ้นเป็นเอ็กโปเน็นเชียลเมื่อจำนวนเมืองเพิ่มขึ้น แต่ยังสามารถหาคำตอบที่ถูกต้องได้ดีกว่าขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม นอกจากนี้ผลการทดลองแสดงให้เห็นว่าจำนวนรอบของโกรเวอร์ที่เหมาะสมช่วยเพิ่มประสิทธิภาพในการหาคำตอบที่เหมาะสมที่สุดในขณะที่จำนวนช็อตหรือจำนวนรอบที่รันอัลกอริทึมช่วยเพิ่มความน่าเชื่อถือให้กับคำตอบที่ได้จากการวัดค่าสถานะคิวบิต

สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

ปีการศึกษา 2564 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6071401321 : MAJOR COMPUTER ENGINEERING

KEYWORD: Quantum computing, Grover's search algorithm, Compact genetic algorithm (cGA), Traveling salesman problem

Kamonluk Suksen : QUANTUM COMPACT GENETIC ALGORITHM FOR HARD PROBLEMS. Advisor: Prof. PRABHAS CHONGSTITVATANA, Ph.D.

Research in the field of quantum technology remains a great challenge to researchers in the future to develop techniques for making large-scale quantum information processing a reality. The simulation of quantum systems is an important problem in many fields. A quantum algorithm is one of the topics being studied for solving NP-hard problems that take too long to solve on classical computers.

This paper aims to propose the Quantum compact genetic algorithm, that exploits the advantages of quantum computing. There is quantum superposition and quantum parallelism in Grover's search algorithm. It was combined with a compact genetic algorithm with an elite (cGA with an elite) in the selection process to get a higher performance in term of the solution quality. The proposed algorithm can be run on an IBM QASM simulator to solve the small-sized traveling salesman problem (TSP). Although the number of function evaluations of the proposed algorithm grows exponentially as the number of cities grows, it still outperforms the cGA with an elite in finding the optimal solution. The results also showed that utilizing the right number of Grover iterations increases the efficiency in finding the optimal solution, whereas the number of shots increases the reliability of the answers.

Field of Study: Computer Engineering Student's Signature .....

Academic Year: 2021 Advisor's Signature .....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ผู้เขียนได้รับเงินทุนสนับสนุนในการทำวิจัยจากหน่วยงาน NSRF ผ่านหน่วยบริหารโครงการทรัพยากรบุคคลและการพัฒนาสถาบัน การวิจัยและนวัตกรรม ทุนหมายเลข B05F640051

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี ด้วยความช่วยเหลือของศาสตราจารย์ ดร. ประภาส จงสถิตย์วัฒนา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งท่านได้ให้คำแนะนำและข้อคิดเห็นต่างๆ อันเป็นประโยชน์อย่างยิ่งในการทำวิจัย อีกทั้งยังช่วยแก้ปัญหาต่างๆ ที่เกิดขึ้นระหว่างการดำเนินงานอีกด้วย ขอกราบขอบพระคุณเป็นอย่างสูง

ขอกราบขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ทุกท่านเป็นอย่างสูง ได้แก่ ศาสตราจารย์ ดร. บุญเจริญ ศิริเนาวกุล ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ รศ.ดร.ธนารัตน์ ชลิดาพงศ์ และ รศ. ดร.ดวงดาว วิชาดากุล ที่สละเวลาอันมีค่ามาชี้ให้เห็นถึงข้อบกพร่อง พร้อมทั้งให้ข้อคิดและคำแนะนำอันเป็นประโยชน์อย่างยิ่งต่องานวิจัย

ขอขอบคุณพี่ๆและน้องๆทุกคนในห้องปฏิบัติการ ISL (Intelligent System Laboratory) ที่ให้คำแนะนำช่วยแก้ปัญหาต่างๆที่เกี่ยวข้องกับงานวิจัยนี้และปัญหาอื่นๆ รวมถึงให้กำลังใจตลอดมา

ขอขอบคุณเจ้าหน้าที่ประจำภาควิชาวิศวกรรมคอมพิวเตอร์ทุกท่านที่มีส่วนช่วยเหลือทำให้วิทยานิพนธ์ฉบับนี้สำเร็จเรียบร้อยลงด้วยดีทุกประการ

สุดท้ายนี้ ผู้วิจัยขอขอบพระคุณบิดามารดา และครอบครัว ซึ่งเปิดโอกาสให้ได้รับการศึกษาเล่าเรียน ตลอดจนคอยช่วยเหลือและให้กำลังใจผู้วิจัยเสมอมาจนสำเร็จการศึกษา

กมลลักษณ์ สุขเสน

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ปัจจุบันนี้คอมพิวเตอร์กลายเป็นเครื่องมือหลักทั้งด้านการทำงานและความบันเทิงของมนุษย์ คอมพิวเตอร์ได้เข้ามามีบทบาทในการทำงานแทนมนุษย์หลายลักษณะ เช่น การทำงานที่เสี่ยงอันตราย การคำนวณทางวิศวกรรม การคำนวณทางสถิติ การทำธุรกิจทั้งภายในและภายนอกองค์กร การสร้างงานศิลปะ การออกแบบสื่อสิ่งพิมพ์ การติดต่อสื่อสารผ่านเครือข่ายอินเทอร์เน็ต การจัดการงานเอกสาร เป็นต้น รวมถึงการใช้งานคอมพิวเตอร์ในด้านความบันเทิง เช่น ดูหนัง ฟังเพลง ชมรายการต่างๆ ของสถานีโทรทัศน์ และเล่นเกมผ่านคอมพิวเตอร์ ด้วยเหตุนี้คอมพิวเตอร์จึงถูกพัฒนาให้มีความสามารถในการทำงานที่รวดเร็ว ถูกต้อง แม่นยำ เก็บข้อมูลได้มากมายมหาศาล และทำงานได้โดยอัตโนมัติ อย่างไรก็ตามความเร็วของคอมพิวเตอร์ในอดีตถึงปัจจุบันเกิดจากการพัฒนาให้ระบบประมวลผลมีขนาดเล็กลงและเร็วขึ้นเรื่อยๆ เช่น หน่วยประมวลผลของสมาร์ทโฟนที่เร็วขึ้นทุกๆปี เป็นต้น ปัจจุบันทรานซิสเตอร์หรือหน่วยประมวลผลมีขนาดเหลือประมาณ 70 อะตอมของซิลิคอนเท่านั้น[1] ปัญหาคือเมื่อพัฒนาไปถึงจุดหนึ่งจนหน่วยประมวลผลเล็กลงไปเหลือเพียง 1 อะตอมของซิลิคอน จะเริ่มส่งผลต่อความเสถียรของการคำนวณบิต 0 กับ 1 และฟิสิกส์ในชีวิตประจำวันอาจไม่สามารถอธิบายปรากฏการณ์ต่างๆที่เกิดขึ้นภายในวงจรได้อีกต่อไป จากข้อจำกัดดังกล่าว นักวิทยาศาสตร์และวิศวกรจึงมีการทำวิจัยร่วมกันเพื่อพัฒนาคอมพิวเตอร์แบบใหม่ที่สามารถใช้ทรานซิสเตอร์ขนาดเทียบเท่าอะตอม และยังสามารถทำงานอย่างเสถียรได้ ซึ่งจำเป็นต้องใช้ความรู้พื้นฐานจากกลศาสตร์ควอนตัม (Quantum mechanics) เพื่ออธิบายการทำงานของทรานซิสเตอร์ดังกล่าว เราเรียกคอมพิวเตอร์ที่มีหน่วยประมวลผลประกอบด้วยทรานซิสเตอร์ขนาดเล็กระดับอะตอมนี้ว่า “คอมพิวเตอร์เชิงควอนตัม” (Quantum computer)

“Quantum computing” คือระบบคอมพิวเตอร์ที่เปลี่ยนจากการทำงานบนแผงวงจรมาใช้คุณสมบัติพิเศษของอะตอมที่เรียกว่า “Quantum superposition” โดยจากเดิมที่คอมพิวเตอร์ปัจจุบันจะแทนค่าข้อมูลด้วยบิต อันประกอบด้วยตัวเลข 0 กับ 1 ทีละตัวแล้วนำไปประกอบกัน แต่ระบบ Quantum computing จะใช้อะตอมที่มีคุณสมบัติการซ้อนทับของควอนตัม (Quantum superposition) เรียกว่าควอนตัมบิต หรือ คิวบิต ซึ่งเป็นหน่วยประมวลผลที่เล็กที่สุดของ

คอมพิวเตอร์เชิงควอนตัม สามารถประมวลผลเป็นตัวเลข 0 หรือ 1 พร้อมกันได้[2] คุณสมบัติดังกล่าวทำให้แต่ละคิวบิตทำงานได้เร็วกว่าบิตอย่างมหาศาล ยกตัวอย่างเช่น Quantum device ที่ประกอบด้วย 32 คิวบิต ทำให้ได้ระบบที่อยู่ในสภาวะซ้อนทับของ 4,294,967,296 สถานะย่อยในเวลาเดียวกัน และจะปรากฏรูปแบบที่ชัดเจนก็ต่อเมื่อถูกสังเกต (Observe) แล้วเท่านั้น เมื่อสถานะซ้อนทับถูกสังเกต หรือถูกรบกวนจากสภาวะภายนอกก็จะเกิดการยุบตัวของสถานะ (Collapses) เหลือเพียงค่า 1 หรือ 0 อย่างใดอย่างหนึ่ง นอกจากนี้คิวบิตยังมีคุณสมบัติการพัวพันในเชิงควอนตัม (Quantum entanglement) ทำให้สามารถสื่อสารกับอะตอมที่เป็นคิวบิตด้วยกันได้โดยไม่ต้องผ่านสื่อกลาง ทำให้คิวบิตสามารถประมวลผลร่วมกันได้ราบรื่นและรวดเร็ว รวมถึงรองรับการทำงานแบบ Multitasking ได้ง่ายกว่า เมื่อปี 2015 Google ประกาศว่าการคำนวณแบบ Quantum annealing ของ D-Wave 2X ทำงานได้เร็วกว่า Simulated annealing ของคอมพิวเตอร์ทั่วไปถึง 100,000,000 เท่า[3] แต่อย่างไรก็ตามระบบ Quantum computing ก็มีข้อจำกัดอยู่ เช่น ตัวคิวบิตที่มีขนาดเล็กกว่าอะตอมและเปราะบาง หากมีสิ่งรบกวนเพียงเล็กน้อย คิวบิต ดังกล่าวก็จะหายไปพร้อมข้อมูลภายใน อีกทั้งยังไม่พบวิธีการคัดลอกคิวบิตเพื่อสำรองข้อมูลโดยสมบูรณ์ นอกจากนี้การเก็บรักษาคิวบิตให้พร้อมใช้งาน จำเป็นต้องเก็บอยู่ในอุณหภูมิศูนย์องศาสัมบูรณ์หรือ -273.15 องศาเซลเซียส

งานวิจัยเกี่ยวกับการนำ Quantum มาใช้กับคอมพิวเตอร์ เริ่มตั้งแต่ปี 1980 แต่เนื่องจากสมัยนั้นมีความซับซ้อนทางฟิสิกส์ค่อนข้างสูงมาก รวมถึงต้องทำงานวิจัยในสภาพแวดล้อมที่เหมาะสม การวิจัยจึงยังอยู่ในวงจำกัด ต่อมาเมื่อเทคโนโลยีพัฒนาขึ้นงานวิจัยเกี่ยวกับระบบ Quantum computing จึงได้รับการพัฒนาต่อโดยบริษัทไอทียักษ์ใหญ่ และประเทศเศรษฐกิจชั้นนำ ตัวอย่างการนำระบบ Quantum computing ไปใช้ในด้านต่างๆ เช่น ระบบ Online security ที่ใช้ Quantum computing ในการถอดรหัสภายในเวลาอันรวดเร็ว การเร่งกระบวนการเรียนรู้ของ AI หรือปัญญาประดิษฐ์ให้เร็วกว่าที่เป็นอยู่เพื่อแก้ปัญหาเฉพาะหน้าได้ดียิ่งขึ้น การทดลองทางเคมีเพื่อพัฒนายารักษาโรคซึ่งต้องอาศัยการคำนวณอันละเอียดและแม่นยำ Quantum Computing ไม่เพียงแต่ทำได้รวดเร็ว แต่ยังสามารถคำนวณค่าต่างๆ พร้อมกัน อีกทั้งในอนาคตการออกแบบยารักษาโรคจะลงลึกไปถึงในระดับวิเคราะห์ DNA เพื่อผลิตยาที่เหมาะกับแต่ละคน และการนำไปใช้พัฒนาการพยากรณ์อากาศให้แม่นยำยิ่งขึ้น โดยปัจจุบัน หน่วยงานพยากรณ์อากาศแห่งชาติของสหราชอาณาจักรได้นำเทคโนโลยี Quantum computing มาใช้เพื่อจำลองแนวโน้มสภาพอากาศในปัจจุบัน

ทำให้มีข้อมูลมากพอจะคาดเดาอากาศได้แม่นยำขึ้น นอกจากนี้ระบบ Quantum computing สามารถนำมาใช้ประเมินเส้นทางให้เราเดินทางได้ประสิทธิภาพสูงสุด ช่วยประหยัดเวลา ลดค่าใช้จ่าย เพิ่มความปลอดภัยให้ผู้ใช้ทางบนวิถีการจราจรอันซับซ้อนขึ้นทุกวัน[4]

การคำนวณทางควอนตัม (Quantum computing) มีการใช้งานอัลกอริทึมควอนตัมเพื่อการประมวลผลที่รวดเร็วขึ้น เช่น อัลกอริทึมการค้นหาของ Grover (Grover's Search Algorithm) และอัลกอริทึมการแยกตัวประกอบที่รวดเร็วของ Shor (Shor's Fast Factoring Algorithm) นอกจากนี้ Genetic Algorithm (GA) ซึ่งเป็นอัลกอริทึมการค้นหาพื้นฐานที่ใช้หลักการวิวัฒนาการทางชีววิทยา เช่น การคัดเลือกโดยธรรมชาติ (Natural selection) การถ่ายทอดทางพันธุกรรม (Genetic inheritance) และการกลายพันธุ์ (Mutation) ก็ได้ถูกพยายามนำมาใช้กับงานวิจัยทางด้านควอนตัม เช่น Quantum Genetic Optimization Algorithm[5] ที่มีการเพิ่มประสิทธิภาพของขั้นตอนวิธีทางพันธุกรรมคลาสสิกโดยใช้ประโยชน์จากอัลกอริทึมการค้นหาของ Grover ในการจำกัดฟังก์ชันฟิตเนสที่มีค่าความเหมาะสมสูงเพื่อเร่งความเร็วในการหาคำตอบที่เหมาะสมที่สุดด้วยขั้นตอนทางพันธุกรรม

งานวิจัยนี้ต้องการนำเสนอขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายาก ได้แก่ปัญหาการเดินทางของพนักงานขาย (Travelling salesman problem) โดยเป็นการนำขั้นตอนวิธีทางพันธุกรรมแบบกระชับ (Compact genetic algorithm) มาประยุกต์ใช้กับอัลกอริทึมควอนตัมคืออัลกอริทึมการค้นหาของ Grover เพื่อช่วยในการหาคำตอบที่ดีที่สุดของปัญหาการเดินทางของพนักงานขายภายในเวลาอันรวดเร็ว และใช้จำนวนครั้งที่ประเมินค่าความเหมาะสม (function evaluation) น้อยกว่าขั้นตอนวิธีทางพันธุกรรมแบบกระชับที่ประมวลผลอยู่บนคอมพิวเตอร์ดั้งเดิม (Classical computer) โดยสามารถโปรแกรมขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัมและทำงานจริงบนเครื่อง IBMQ QASM simulator ของไอบีเอ็มได้ สำหรับคอมพิวเตอร์ดั้งเดิม Compact Genetic Algorithm (cGA) เป็นอัลกอริทึมที่แทนโครงสร้างของประชากรด้วยเวกเตอร์ความน่าจะเป็น ทำให้ cGA สามารถทำงานได้ในพื้นที่หน่วยความจำที่มีขนาดเล็กกว่า Genetic Algorithm (GA) แต่สำหรับคอมพิวเตอร์เชิงควอนตัม cGA จะแทนโครงสร้างของประชากรด้วยมุมเวกเตอร์สถานะของคิวบิตใน Quantum register ซึ่งเป็นวิธีเดียวกันกับที่ใช้ในงานวิจัย[6] งานวิจัยนี้ถูกทดสอบบนเครื่อง IBMQ QASM simulator เนื่องจากปัญหาการเดินทางของพนักงานขาย ซึ่งเป็นปัญหาต้นแบบที่ผู้วิจัยใช้สำหรับงานวิจัยนี้ จำเป็นต้องใช้คิวบิตจำนวนมากใน

การประมวลผลข้อมูล ดังนั้นเครื่อง IBMQ QASM simulator จึงมีความเหมาะสมที่สุดเพราะสามารถรองรับจำนวนคิวบิตได้มากถึง 32 คิวบิต และผู้วิจัยทั่วไปสามารถใช้งานได้โดยไม่มีค่าใช้จ่าย ผู้วิจัยได้ทำการเปรียบเทียบประสิทธิภาพการประมวลผลระหว่างการใช้ขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) และขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) ใน 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ (accuracy) และจำนวนครั้งที่ประเมินค่าความเหมาะสม (function evaluation) จนได้คำตอบที่เหมาะสมที่สุด

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 นำเสนอขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายากที่สามารถประมวลผลได้จริงบนเครื่องคอมพิวเตอร์เชิงควอนตัมของไอบีเอ็มและหรือเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองของไอบีเอ็ม

1.2.2 วิเคราะห์ประสิทธิภาพขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายากเปรียบเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมใน 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ (accuracy) และจำนวนครั้งที่ประเมินค่าความเหมาะสม (function evaluation) จนได้คำตอบที่เหมาะสม

1.3 ขอบเขตของงานวิจัย

1.3.1 นำเสนอขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม

1.3.2 ศึกษาการนำขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมมาหาคำตอบของปัญหาง่ายและยาก ได้แก่ One-max และ Travelling salesman problem (TSP)

1.3.3 แสดงผลการหาคำตอบของปัญหาโดยขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม พร้อมวิเคราะห์ประสิทธิภาพของขั้นตอนวิธีที่นำเสนอ

1.4 ขั้นตอนและวิธีการดำเนินงานวิจัย

1.4.1 ศึกษาข้อมูลเกี่ยวกับการประมวลผลเชิงควอนตัม (Quantum Computing) ข้อมูลควอนตัม (Quantum Information) วงจรควอนตัม (Quantum Circuit) และควอนตัมเรจิ-สเตอร์ (Quantum register)

1.4.2 ศึกษางานวิจัยที่เกี่ยวข้อง

1.4.3 ศึกษาเครื่องมือและโปรแกรมที่ใช้สำหรับงานวิจัย

1.4.4 ศึกษาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (cGA) สำหรับปัญหาง่ายและยาก

1.4.5 วิเคราะห์และออกแบบขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเพื่อหาคำตอบของปัญหาง่าย

1.4.6 พัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเพื่อหาคำตอบของปัญหาง่าย

1.4.7 ทดสอบและประเมินผลขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเพื่อหาคำตอบของปัญหาง่าย

1.4.8 พัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเพื่อหาคำตอบของปัญหายาก

1.4.9 ทดสอบและประเมินผลขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเพื่อหาคำตอบของปัญหายาก

1.4.10 ปรับปรุงประสิทธิภาพของขั้นตอนวิธีที่นำเสนอ

1.4.11 สรุปผลและจัดทำวิทยานิพนธ์

1.4.12 เผยแพร่ผลงานตีพิมพ์

1.5 ประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย

1.5.1 ได้ศึกษาการนำขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับมาประยุกต์ใช้กับอัลกอริทึมการค้นหาของ Grover เพื่อให้ได้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่สามารถประมวลผลได้จริงบนเครื่องคอมพิวเตอร์เชิงควอนตัมของไอบีเอ็มและหรือเครื่องคอมพิวเตอร์เชิงควอนตัมแบบจำลองของไอบีเอ็ม

1.5.2 ได้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่สามารถหาคำตอบของปัญหายากได้อย่างถูกต้องและมีประสิทธิภาพดีกว่าขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับดั้งเดิม (cGA) ในแง่ของความถูกต้องของคำตอบ

1.5.3 วิธีการที่นำเสนอสามารถนำไปประยุกต์ใช้เพื่อหาคำตอบของปัญหาการหาค่าเหมาะสม (optimization problem) อื่นๆได้

1.6 รายชื่อผลงานที่ได้เผยแพร่ขณะที่กำลังดำเนินงานวิจัย

ส่วนหนึ่งของงานวิจัยนี้ได้รับการตอบรับให้ตีพิมพ์เป็นบทความทางวิชาการในหัวข้อเรื่อง “Exploiting Building Blocks in Hard Problems with Modified Compact Genetic Algorithm” โดย กมลลักษณ์ สุขเสน และ ประภาส จงสถิตย์วัฒนา ในงานประชุมวิชาการ “The 15th International Joint Conference on Computer Science and Software Engineering (JCSSE2018)” ณ คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล วิทยาเขตศาลายา จังหวัดนครปฐม ในระหว่างวันที่ 11-13 กรกฎาคม 2561

1.7 ตารางระยะเวลาดำเนินงานวิจัย

ขั้นตอนการทำงาน/ ภาคการศึกษา	ภาคการศึกษา 1/2561						ภาคการศึกษา 2/2561 – 1/2564						ภาค การศึกษา 2/2564		
1. จัดทำโครงร่างวิทยานิพนธ์	■	■	■	■											
2. สอบโครงร่างและปรับแก้วิทยานิพนธ์					■	■									
3. ดำเนินตามแผนวิจัย															
3.1 ทำการปรับปรุงขั้นตอนวิธีที่นำเสนอสำหรับปัญหายากให้ครบตามที่เสนอ							■	■	■	■	■	■			
3.2 วิเคราะห์ผล และสรุปผล							■	■	■	■	■	■			
4. เตรียมบทความเพื่อตีพิมพ์วารสาร													■	■	■
5. เตรียมทำเล่มวิทยานิพนธ์ฉบับสมบูรณ์													■	■	■

ตารางที่ 1 ตารางระยะเวลาดำเนินงานวิจัย

1.8 รายละเอียดเนื้อหาในวิทยานิพนธ์

สาระสำคัญของวิทยานิพนธ์ฉบับนี้มีรายละเอียดดังนี้

1.8.1 ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted compact genetic algorithm) สำหรับแก้ปัญหายาก โดยนำการประมวลผลเชิงควอนตัมมาใช้สำหรับสร้างโครโมโซม เนื่องจากผู้วิจัยเลือกใช้ selection rate เท่ากับ 2 ดังนั้นจึงต้องมีการสร้างโครโมโซมมา 2 ตัว เพื่อทำการเปรียบเทียบกัน สำหรับโครโมโซมแรก (first individual) เป็นการนำค่าความน่าจะเป็นในเวกเตอร์ความน่าจะเป็น มาแปลงเป็นค่ามุมสำหรับเป็นอินพุตของเกตควอนตัมเพื่อหมุนคิวบิตให้ได้สถานะคิวบิตที่ต้องการ สำหรับโครโมโซมที่สอง (second individual) เป็นการนำอัลกอริทึมควอนตัม ได้แก่ อัลกอริทึมการค้นหาของโกรเวอร์มาเป็นส่วนหนึ่งของการสร้าง second individual ในขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม โดยอาศัยประโยชน์จากอัลกอริทึมการค้นหาของโกรเวอร์ในการขยายสัญญาณแอมพลิจูดของสถานะควอนตัมที่ทำให้ได้คำตอบที่เป็น feasible path สำหรับปัญหา TSP จากทั้งหมด 2^n รูปแบบ เมื่อ n คือจำนวนคิวบิตที่ใช้ในการแทนรูปแบบคำตอบ โดยผลจากการขยายสัญญาณแอมพลิจูดทำให้ค่าความน่าจะเป็นของการวัดค่าสถานะควอนตัมแล้วได้รูปแบบคำตอบที่เป็น feasible path สูงมากขึ้น โดยส่วนของการประมวลผลเชิงควอนตัมถูกประมวลผลอยู่บนเครื่อง IBM QASM simulator จากนั้นจึงทำการเปรียบเทียบค่าความเหมาะสมระหว่างโครโมโซมทั้งสองตัว และทำการอัพเดตเวกเตอร์ความน่าจะเป็นตามโครโมโซมที่ดีกว่า ซึ่งส่วนนี้จะทำงานบนคอมพิวเตอร์ดั้งเดิม เวกเตอร์ความน่าจะเป็นที่ถูกอัพเดตจะถูกนำมาใช้ในขั้นตอนกำหนดสถานะเริ่มต้นของคิวบิตในรอบถัดไป จากนั้นจึงดำเนินการสร้างโครโมโซมด้วยขั้นตอนดังที่กล่าวไปแล้วในข้างต้น กระบวนการดังกล่าวจะถูกวนซ้ำจนกว่าเวกเตอร์ความน่าจะเป็นลู่เข้าค่า 1 หรือ 0

1.8.2 ผลการเปรียบเทียบประสิทธิภาพของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกโครโมโซมที่ดี (compact genetic algorithm with an elite) และ Grover-assisted compact genetic algorithm ในแง่ของความถูกต้องของคำตอบ (accuracy) และจำนวนครั้งที่ประเมินค่าความเหมาะสม (function evaluation) จนได้

คำตอบที่เหมาะสม พร้อมวิเคราะห์ผลการทดลอง และความซับซ้อนในเชิงควอนตัม (quantum complexity)

1.8.3 การปรับปรุงฟังก์ชันโอราเคิลของ Grover-assisted compact genetic algorithm ให้สามารถใช้ประโยชน์จากการประมวลผลเชิงควอนตัมแบบขนาน (quantum parallelism) จากสถานะซ้อนทับเชิงตำแหน่งของคิวบิต (quantum superposition) ได้เต็มประสิทธิภาพมากขึ้น โดยทำให้ฟังก์ชันโอราเคิลสามารถคำนวณระยะทางของทุกๆ สถานะควอนตัมที่เป็นไปได้ในเวลาเดียวกัน โดยใช้หลักการของควอนตัมฟูริเยร์ทรานสฟอร์ม และสามารถเปรียบเทียบสถานะควอนตัมดังกล่าวกับรูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน โดยใช้ quantum comparator เพื่อช่วยให้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับสามารถสร้าง second individual ที่ไม่แย่กว่ารูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน

1.8.4 ผลการเปรียบเทียบประสิทธิภาพของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกโครโมโซมที่ดี (compact genetic algorithm with an elite) และ new Grover-assisted compact genetic algorithm ในแง่ของความถูกต้องของคำตอบ (accuracy) และจำนวนครั้งที่ประเมินค่าความเหมาะสม (function evaluation) จนได้คำตอบที่เหมาะสม พร้อมวิเคราะห์ผลการทดลอง และความซับซ้อนในเชิงควอนตัม (quantum complexity)

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 คุณสมบัติเชิงคณิตศาสตร์ของคิวบิต (Qubit)

คอมพิวเตอร์ที่เราใช้ในปัจจุบัน (classical computer) มีหน่วยเก็บข้อมูลที่เล็กที่สุดเรียกว่า “บิต” แต่สำหรับคอมพิวเตอร์เชิงควอนตัม หน่วยเก็บข้อมูลที่เล็กที่สุดเรียกว่า “คิวบิต” คิวบิตคือสถานะของอนุภาค (particle) เช่น ทิศทางที่อนุภาคเคลื่อนที่ไป หรือทิศทางการหมุนของอนุภาค แต่สำหรับอนุภาคในระดับที่เล็กมากๆ เช่น อิเล็กตรอนนั้น อนุภาคอาจจะมีการเคลื่อนที่หรือหมุนไปในสองทิศทางได้พร้อมๆ กัน ซึ่งมีคุณสมบัติคล้ายกับคลื่น (Wave) เพราะคลื่นสามารถเคลื่อนที่ไปในทุกทิศทางพร้อมๆกันได้ เช่น คลื่นเสียง เป็นต้น ดังนั้นจึงอาจระบุได้ว่าอนุภาคก็สามารถประพฤติตัวเป็นคลื่นได้เช่นกัน ดังจะเห็นได้จากการทดลองทางวิทยาศาสตร์หลายๆ ชิ้นบ่งชี้ว่า อิเล็กตรอนเป็นทั้งคลื่นและอนุภาคในเวลาเดียวกัน ถ้าเราไม่ไปรบกวนอิเล็กตรอนโดยการพยายามวัดตำแหน่งหรือทิศทางการเคลื่อนที่ของมัน อิเล็กตรอนจะประพฤติตัวเป็นคลื่น คือเคลื่อนที่ไปในหลายทิศทาง และมีการแทรกสอด (interference) ซึ่งเป็นพฤติกรรมปกติของคลื่น แต่ถ้าเราไปรบกวนมัน อิเล็กตรอนจะประพฤติตัวเป็นอนุภาค คือมีตำแหน่งและทิศทางการเคลื่อนที่ที่แน่นอน อิเล็กตรอนจึงมีคุณสมบัติเป็นได้ทั้งอนุภาคและคลื่นในเวลาเดียวกัน ซึ่งเป็นความจริงที่สามารถพิสูจน์ได้ด้วยการทดลองทางวิทยาศาสตร์ง่ายๆ ตราบเท่าที่ยังไม่มีการทดลองหรือทฤษฎีใหม่ๆ มาหักล้าง

ปรากฏการณ์ที่นักฟิสิกส์ค้นพบว่าอิเล็กตรอนเป็นทั้งคลื่นและอนุภาคในเวลาเดียวกัน จึงนำมาสู่ปัญหาเรื่องการหาตัวแบบทางคณิตศาสตร์เพื่อมาอธิบายปรากฏการณ์ที่ค้นพบ ดังนั้นนักฟิสิกส์จึงได้ประดิษฐ์ตัวแบบทางคณิตศาสตร์ขึ้นมาใหม่ เรียกว่า “state vector notation” ซึ่งเป็นที่มาของ “คิวบิต” หรือสถานะของอนุภาค สามารถเขียนอธิบายได้ดังนี้

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad (1)$$

โดยที่ $|\ \rangle$ คือสัญลักษณ์เรียกว่า ket ค่าที่อยู่ภายใน ket คือสถานะที่เป็นไปได้ เช่น สถานะที่เป็นไปได้ของคิวบิตคือ 0 หรือ 1 โดยที่

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix},\ |1\rangle = \begin{bmatrix}0\\1\end{bmatrix} \quad (2)$$

α, β คือแอมพลิจูด (amplitude) เป็นจำนวนเชิงซ้อน (complex number) เพื่อแทนทั้งแอมพลิจูด (amplitude) และเฟส (phase) ของคลื่น โดยมีเงื่อนไขว่า

$$|\alpha|^2 + |\beta|^2 = 1 \quad (3)$$

โดยที่ $|\alpha|^2$ คือความน่าจะเป็นที่ถ้าเราวัดจะสังเกตเห็นสถานะ $|0\rangle$ และ $|\beta|^2$ คือความน่าจะเป็นที่ถ้าเราวัดจะสังเกตเห็นสถานะ $|1\rangle$ แต่ถ้าเราไม่ไปรบกวนหรือไม่ไปวัดมัน คิวบิตจะอยู่ในสถานะที่เรียกว่า Superposition คือเป็นทั้ง $|0\rangle$ และ $|1\rangle$ ในเวลาเดียวกัน โดยมีแอมพลิจูดเท่ากับ α และ β ตามลำดับ การวัดจะเป็นการรบกวนคิวบิตทำให้สถานะ Superposition ยุบ (collapse) แล้วสถานะของคิวบิตจะกลายเป็น $|0\rangle$ หรือ $|1\rangle$ ตามความน่าจะเป็น

นอกจากการอธิบายสถานะของคิวบิตโดยใช้ state vector notation แล้ว เรายังสามารถเขียนอธิบายสถานะของคิวบิตโดยใช้ density matrices ได้ ดังนี้

$$|\varphi_{AB}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\1\end{bmatrix}$$

$$\rho_{AB} = |\varphi_{AB}\rangle\langle\varphi_{AB}|$$

$$\rho_{AB} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\1\end{bmatrix}\left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix}\right) \quad (4)$$

$$\rho_{AB} = \frac{1}{2}\begin{bmatrix}1 & 0 & 0 & 1\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\\1 & 0 & 0 & 1\end{bmatrix}$$

2.2 ควอนตัมเรจิสเตอร์ (Quantum register)

ควอนตัมเรจิสเตอร์คือการประกอบคิวบิตหลายๆตัวเข้าด้วยกัน เช่น ถ้าเอาคิวบิต 2 ตัวมารวมกัน สามารถเขียนอธิบายได้ดังนี้

$$\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle \quad (5)$$

สังเกตว่าผลรวมของแอมพลิจูดยกกำลังสอง $\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 = 1$ เสมอเพราะเป็นความน่าจะเป็น และบางสถานะ เช่นสถานะข้างต้นอาจจะแยกตัวประกอบได้ (Decomposable) ได้ดังนี้

$$\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) \quad (6)$$

วงเล็บทางซ้ายคือคิวบิตตัวแรก วงเล็บทางขวาคือคิวบิตตัวที่สอง แต่บางสถานะก็แยกตัวประกอบไม่ได้ เช่น EPR pair (Einstein, Podolsky, Rosen) ดังนี้

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle \quad (7)$$

จาก (7) มีข้อสังเกตว่าถ้าวัดบิตแรกของ ERP pair ได้ 0 ทั้งสองคิวบิตจะยุบ (Collapse) เป็น $|00\rangle$ ทำให้บิตที่สองเป็น 0 แน่ๆ โดยที่ไม่จำเป็นต้องไปวัดอีก ทำนองเดียวกันกับกรณีที่วัดบิตแรกของ ERP pair ได้ 1 บิตที่สองก็จะเป็น 1 เช่นเดียวกัน คำถามที่นักวิทยาศาตร์ยังคงสงสัยคือถ้าจับสองคิวบิต (สองอนุภาค เช่น อิเล็กตรอนสองตัว) ที่เป็น EPR pair แยกกันไว้คนละสุดขอบจักรวาล จะยังมีพฤติกรรมเช่นนี้อีกหรือไม่

2.3 วงจรควอนตัม (Quantum circuit)

วงจรควอนตัมมีลักษณะคล้ายๆ กับวงจรเชิงผสม (Combinatorial circuit) กล่าวคือวงจรทำหน้าที่รับอินพุต และให้เอาต์พุตออกไป ต่างกันตรงที่วงจรควอนตัมรับอินพุตและเอาต์พุตเป็นคิวบิต

ถ้าเขียนอธิบายด้วยสมการทางคณิตศาสตร์ วงจรควอนตัมแทนด้วยเมทริกซ์ M อินพุตที่เป็นคิวบิตแทนด้วยคอลัมน์เมทริกซ์ A และเอาต์พุตแทนด้วยคอลัมน์เมทริกซ์ B การทำงานของวงจรจะเป็นตามสมการข้างล่าง

$$MA = B \tag{8}$$

ค่าที่อยู่ในเมทริกซ์ A และ B คือ แอมพลิจูด (amplitude) เช่น ถ้าอินพุตคือ $\alpha|0\rangle + \beta|1\rangle$ เมทริกซ์ A คือ $[\alpha, \beta]^T$ หรือถ้าอินพุตคือ $a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ เมทริกซ์ A คือ $[a, b, c, d]^T$ ถ้าอินพุตมี n คิวบิต เมทริกซ์ A และ B จะมีขนาด 2^n x 1 สังเกตว่าถ้าเราจำลองการทำงานของคอมพิวเตอร์เชิงควอนตัมด้วยคอมพิวเตอร์แบบคลาสสิกอย่างที่ใช้ในปัจจุบันจะต้องเสียเวลาเพิ่มขึ้นเป็นเอ็กโปเน็นเชียล (Exponential) ตามจำนวนคิวบิตที่เพิ่มขึ้น

ยกตัวอย่างวงจรควอนตัมที่มีอินพุตแค่ 1 หรือ 2 คิวบิต หรือเรียกว่าเกตควอนตัม (Quantum gate) เราสามารถนำเกตควอนตัมหลายๆ ตัวมาประกอบกันให้เป็นวงจรที่ใหญ่ขึ้นได้

Quantum not gate

$$M_N = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{9}$$

เรียกว่า Quantum Not Gate เพราะทำหน้าที่กลับสถานะของคิวบิตดังนี้ $M_N[1,0]^T = [0,1]^T$ และ $M_N [0,1]^T = [1,0]^T$ เมื่อ $[1,0]^T$ คือสถานะ $|0\rangle$ และ $[0,1]^T$ คือสถานะ $|1\rangle$

Square root of the not gate

$$M_S = \begin{pmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{pmatrix} \tag{10}$$

เรียกว่า square root of the not gate เพราะว่าทำแค่ครึ่งเดียวของ Quantum not gate $M_S M_S = M_N$

Hadamard gate

$$M_H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \quad (11)$$

Hadamard gate ทำหน้าที่แปลงสถานะของคิวบิตให้อยู่ในสถานะซ้อนทับเชิงตำแหน่งของคิวบิต (Superposition) จะทำหน้าที่เหมือนเมทริกซ์เอกลักษณ์ (Identity matrix, I) แต่ทำเพียงครึ่งเดียว (IA = A) จะได้ $M_H M_H = I$ ทำให้ $M_H M_H [1,0]^T = [1,0]^T$ และ $M_H M_H [0,1]^T = [0,1]^T$

Phase flip

$$MP = \begin{pmatrix} 1 & 0 \\ 0 & e^{-i\theta} \end{pmatrix} \quad (12)$$

Phase flip ทำหน้าที่เปลี่ยนเฟส (Phase) ของส่วนที่มีคุณสมบัติเป็นคลื่น โดยจะเปลี่ยนเฉพาะเฟส ไม่เปลี่ยนขนาดแอมพลิจูด

วงจรควอนตัมในมุมมองทางคณิตศาสตร์ก็คือเมทริกซ์ที่เปลี่ยนสถานะควอนตัมเรจิสเตอร์จากสถานะหนึ่งไปยังอีกสถานะหนึ่ง และเมทริกซ์นี้เป็น unitary matrix ด้วย ทำให้ผลลัพธ์ที่ได้จะยังรักษากฎของความน่าจะเป็นอยู่ คือผลรวมของแอมพลิจูดยกกำลังสองต้องได้ 1 เสมอ

2.4 ผลคูณเท็นเซอร์ (Tensor product)

ผลคูณเท็นเซอร์ (Tensor product) เป็นตัวดำเนินการที่สำคัญสาหรับการคูณเมทริกซ์ เนื่องจากเกตควอนตัมพื้นฐานก็คือเมทริกซ์ ดังนั้นเมื่อเราเอาเกตหรือวงจรเล็กๆ มาต่อเข้าด้วยกัน ก็เหมือนประกอบเมทริกซ์เล็กๆ ให้เป็นเมทริกซ์ที่ใหญ่ขึ้น และจำเป็นต้องใช้ตัวดำเนินการที่เรียกว่าผล

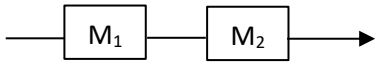
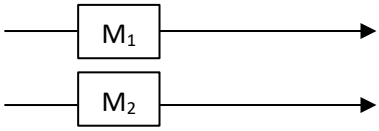
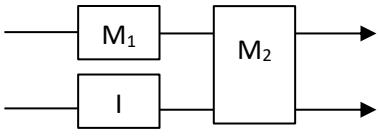
คูณเท็นเซอร์ (tensor product) เพื่อรวมเมทริกซ์หลายๆ อันเข้าด้วยกัน ผลคูณเท็นเซอร์ระหว่าง A และ B เขียนแทนด้วย A $\otimes$ B ดังตัวอย่างต่อไปนี้

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1s} \\ a_{21} & a_{22} & \cdots & a_{2s} \\ \cdots & \cdots & \cdots & \cdots \\ a_{r1} & a_{r2} & \cdots & a_{rs} \end{pmatrix} \quad B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1u} \\ b_{21} & b_{22} & \cdots & b_{2u} \\ \cdots & \cdots & \cdots & \cdots \\ b_{t1} & b_{t2} & \cdots & b_{tu} \end{pmatrix}$$

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1s}B \\ a_{21}B & a_{22}B & \cdots & a_{2s}B \\ \cdots & \cdots & \cdots & \cdots \\ a_{r1}B & a_{r2}B & \cdots & a_{rs}B \end{pmatrix} \quad (13)$$

เมทริกซ์ผลลัพธ์หรือ A $\otimes$ B จะมีขนาด rt × su

การประกอบวงจรเล็กๆ เข้าด้วยกันประกอบด้วยรูปแบบดังนี้ (กรณีที่เป็นสายเปล่า เทียบเท่ากับมีเมทริกซ์เอกลักษณ์อยู่)

วงจรควอนตัม	เมทริกซ์ที่แทนวงจรทั้งหมด
	$M = M_2M_1$
	$M = M_1 \otimes M_2$
	$M = M_2(M_1 \otimes I)$

ตารางที่ 2 ตารางแสดงการดำเนินการทางเมทริกซ์สำหรับการประกอบวงจรควอนตัมแบบต่างๆ

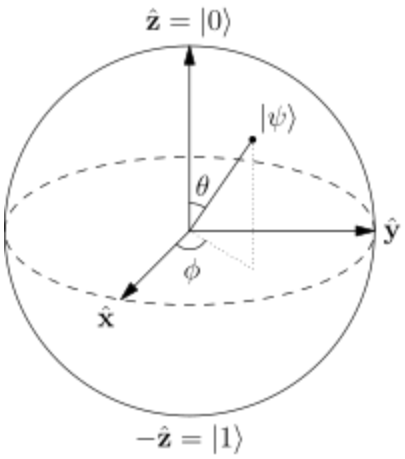
2.5 การหมุนคิวบิต (Qubit rotation)

สถานะของคิวบิตจาก (1) สามารถเขียนในรูปผลรวมเชิงเส้นของสถานะของคิวบิตโดยใช้ตรีโกณมิติได้ เนื่องจาก $\sqrt{\alpha^2 + \beta^2} = 1$ เพราะเป็นค่าความน่าจะเป็นของสถานะ $|0\rangle$ และ $|1\rangle$

และจากฟังก์ชันตรีโกณมิติ $\sqrt{\sin^2 x + \cos^2 x} = 1$ เราจึงสามารถเขียนสถานะของคิวบิตโดยใช้ฟังก์ชันตรีโกณมิติได้ดังนี้

$$|\varphi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\emptyset}\sin\frac{\theta}{2}|1\rangle \tag{14}$$

เมื่อ θ และ $\emptyset$ คือจำนวนจริง โดยที่ $0 \leq \theta \leq \pi$ และ $0 \leq \emptyset \leq 2\pi$ เราจึงสามารถใช้ทรงกลมโบลช (Bloach sphere) เพื่อแสดงสถานะของคิวบิต ดังภาพที่ 1

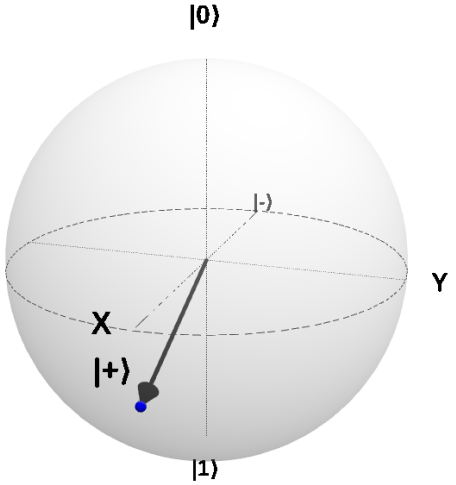


ภาพที่ 1 ทรงกลมโบลชแสดงสถานะของคิวบิต $|\varphi\rangle$

การหมุนคิวบิตรอบแกน Y ด้วยมุม θ และ $\emptyset$ จะทำให้สถานะของคิวบิต $|\varphi\rangle$ เปลี่ยนแปลงไป โดยการหมุนคิวบิตรอบแกน Y สามารถอธิบายได้โดยใช้การแปลงเชิงเมทริกซ์ (Matrix transformation) ดังสมการ (15)

$$RY(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{15}$$

โดยค่าพารามิเตอร์ θ ขึ้นอยู่กับความน่าจะเป็นที่สถานะของคิวบิตนั้นเป็น $|0\rangle$ หรือ $|1\rangle$ ยกตัวอย่างเช่นการหมุนคิวบิตรอบแกน Y จากสถานะซ้อนทับของคิวบิต โดยกำหนดให้ $\theta = \frac{\pi}{8}$ และ $\emptyset = 0$ จะทำให้ความน่าจะเป็นที่เมื่อวัดสถานะของคิวบิตและได้ผลลัพธ์เป็นสถานะ $|1\rangle$ มากกว่าสถานะ $|0\rangle$ ดังภาพที่ 2



ภาพที่ 2 ทรงกลมโบลชแสดงสถานะของคิวบิตที่ใกล้เคียงกับสถานะ $|1\rangle$ มากกว่าสถานะ $|0\rangle$

2.6 อัลกอริทึมการค้นหาของ Grover (Grover's search algorithm)

อัลกอริทึมการค้นหาของ Grover (Grover's search Algorithm) เป็นอัลกอริทึมควอนตัมที่แสดงให้เห็นว่าสามารถนำระบบควอนตัมมาใช้เพื่อปรับปรุงขอบเขตรันไทม์ที่ต่ำกว่าของอัลกอริทึมแบบดั้งเดิมบนคอมพิวเตอร์คลาสสิกสำหรับค้นหาข้อมูลที่ไม่มีโครงสร้าง (unstructured data) ได้ เช่น ถ้าข้อมูลที่ต้องการค้นหามีจำนวน 1,000,000 ชุด อัลกอริทึมแบบดั้งเดิมอาจจะใช้เวลาในการค้นหานานที่สุดคือ 1,000,000 รอบ time complexity เท่ากับ $O(N)$ แต่อัลกอริทึมการค้นหาของ Grover สามารถค้นหาข้อมูลที่ไม่มีโครงสร้าง ได้โดยใช้เวลาเพียง 1,000 รอบ time complexity เท่ากับ $O(\sqrt{N})$ เมื่อ N คือจำนวนข้อมูลทั้งหมด การเพิ่ม speed up ในการค้นหาข้อมูลของอัลกอริทึมการค้นหาของ Grover ทำได้โดยใช้เคล็ดลับการขยายแอมพลิจูดของ Grover[7]

อัลกอริทึมการค้นหาของ Grover มีกระบวนการ ดังนี้

2.6.1 เตรียม Quantum register สำหรับเก็บข้อมูลจำนวน n คิวบิต เมื่อ n คือจำนวนคิวบิตที่จำเป็นในการแทนที่ search space ขนาด 2^n โดยกำหนดให้ $2^n = N$ และทุกคิวบิตมีค่าเริ่มต้นเป็น $|0\rangle$

$$|0\rangle^{\otimes n} = |0\rangle \tag{16}$$

2.6.2 เตรียมสถานะ Superposition ของคิวบิต ให้ความน่าจะเป็นของการวัดคิวบิตแล้วได้สถานะต่างๆ นั้นมีค่าเท่ากัน โดยผ่านการทำ Hadamard transform $(H^{\otimes n})$ ใช้เวลา *O*(*N*)

$$|\varphi\rangle = H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^{n-1}}|x\rangle \quad (17)$$

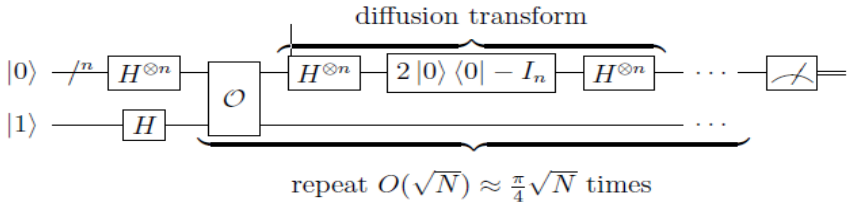
2.6.3 ทำการแปลงค่าแอมพลิจูดโดยใช้ Grover iteration ซึ่งประกอบด้วยขั้นตอนดังนี้

2.6.3.1 กำหนดฟังก์ชัน quantum oracle เพื่อทำการระบุ quantum state ที่ต้องการ quantum oracle ใช้สัญลักษณ์ O เป็นฟังก์ชัน black-box ที่สามารถปรับเปลี่ยนระบบเพื่อให้ได้ค่าสถานะของคิวบิตที่ต้องการ นั่นหมายความว่า quantum black-box สามารถสังเกตและปรับเปลี่ยนค่าแอมพลิจูดของระบบได้โดยไม่ยุบสถานะของคิวบิตให้กลายเป็นสถานะดั้งเดิม (classical state) และยังสามารถระบุได้ว่าระบบอยู่ในสถานะที่ถูกต้องแล้วหรือยัง ถ้าระบบอยู่ในสถานะที่ถูกต้อง oracle จะหมุนเฟสไป π เรเดียน แต่ถ้าไม่ถูกต้องก็จะไม่ทำอะไร การหมุนเฟสทำได้โดยการคูณ amplitude ของสถานะนั้นของคิวบิตด้วย -1 แอมพลิจูดสำหรับสถานะนั้นเปลี่ยนแปลง แต่ความน่าจะเป็นที่จะอยู่ในสถานะนั้นยังเท่าเดิม สามารถเขียนการทำงานของ quantum oracle ได้ดังนี้

$$|x\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle \quad (18)$$

โดยที่ $f(x) = 1$ ถ้า x เป็นสถานะที่ถูกต้อง และ $f(x) = 0$ ถ้า x เป็นสถานะที่ไม่ถูกต้อง อย่างไรก็ตามการกำหนดค่า output ของ $f(x)$ ขึ้นอยู่กับปัญหาที่ต้องการค้นหาคำตอบ

2.6.3.2 ทำ diffusion transform โดยหาค่าผกผันของค่าเฉลี่ยสำหรับค่าแอมพลิจูดของคิวบิตสถานะต่างๆทุกสถานะ ซึ่งค่าเฉลี่ยของแอมพลิจูดใหม่นี้จะมีค่าลดลงจากค่าเฉลี่ยของแอมพลิจูดก่อนหน้าที่ยังไม่ผ่านการแปลง diffusion transform มีวงจรการทำงานดังภาพที่ 3



ภาพที่ 3 วงจรไดอะแกรมสำหรับ Grover's algorithm[8]

2.6.3.3 คำนวณความน่าจะเป็นของสถานะข้อมูลทุกๆ สถานะใหม่ ซึ่งจะทำให้สถานะของคิวบิตที่สนใจมีความน่าจะเป็นเพิ่มขึ้น ในขณะที่สถานะของคิวบิตอื่นๆมีความน่าจะเป็นลดลงตามสูตรดังนี้ โดยผ่าน Grover iteration จำนวน R ครั้ง

$$[(2|\varphi\rangle\langle\varphi| - I)O]^R|\varphi\rangle \approx |x_0\rangle \quad R \approx \frac{\pi}{4}\sqrt{\frac{N}{T}} \text{ ครั้ง} \tag{19}$$

Grover iteration จะถูกทำซ้ำเป็นจำนวน $\frac{\pi}{4}\sqrt{\frac{N}{T}}$ ครั้ง [7] เมื่อ T คือ จำนวนของคำตอบทั้งหมด เพื่อที่จะให้ได้ค่าความน่าจะเป็นสูงสุดของสถานะของคิวบิตที่สนใจ

2.6.4 เมื่อทำ Grover iteration จนครบจำนวนครั้งที่เพียงพอแล้ว ต้องทำการ reset ancilla qubit ทั้งหมดก่อน แล้วจึงทำการวัดเพื่อหาผลลัพธ์ ซึ่งจะถูกต้องตามความน่าจะเป็น $O(1)$ และถือว่าสิ้นสุดกระบวนการทำงานอัลกอริทึมการค้นหาของ Grover

2.7 วงจรบวกโดยใช้ควอนตัมฟูริเยร์ทรานสฟอร์ม (Quantum Fourier transform)

ควอนตัมฟูริเยร์ทรานสฟอร์ม (Quantum Fourier transform) คือการแปลงฟูริเยร์แบบไม่ต่อเนื่อง (Discreate Fourier transform) สำหรับการคำนวณด้วยเครื่องคอมพิวเตอร์เชิงควอนตัม ควอนตัมฟูริเยร์ทรานสฟอร์ม หรือ (QFT) ประกอบด้วยเวกเตอร์ของชุดข้อมูล โดยที่แต่ละค่าในเวกเตอร์แทนความน่าจะเป็นที่สถานะของคิวบิตจะถูกวัดออกมาเป็นค่า 0 หรือ 1 QFT จะทำการแปลงพื้นฐานการคำนวณทั่วไป (Computational basis) ให้กลายเป็นการพื้นฐานการคำนวณฟูริเยร์ (Fourier basis) เราสามารถแปลงจากพื้นฐานการคำนวณทั่วไปไปเป็นพื้นฐานการคำนวณฟูริเยร์โดย

ใช้ direct Fourier transformation ในทางกลับกัน เราสามารถแปลงจากพื้นฐานการคำนวณฟูริเยร์ไปเป็นพื้นฐานการคำนวณทั่วไปโดยใช้ inverse Fourier transformation

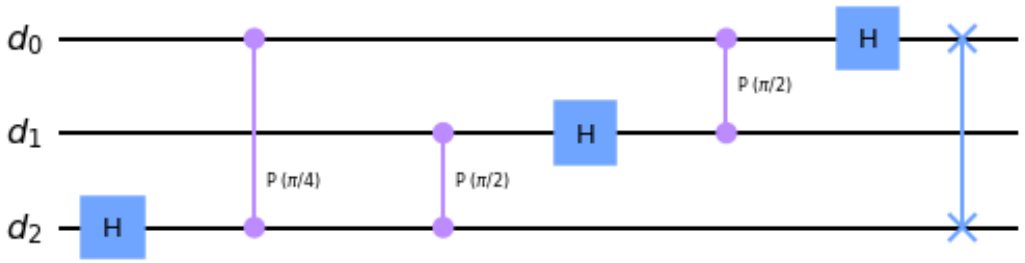
การแปลงฟูริเยร์เชิงควอนตัมของสถานะ $|x\rangle$ จากพื้นฐานการคำนวณทั่วไปได้แก่ $|0\rangle$, $|1\rangle$, ..., $|n\text{-}1\rangle$, โดยใช้ direct Fourier transformation สามารถเขียนได้ดังนี้

$$QFT|x\rangle = \frac{1}{\sqrt{n}}\sum_{y=0}^{n-1} e^{i\frac{2xy\pi}{n}} |y\rangle \quad (20)$$

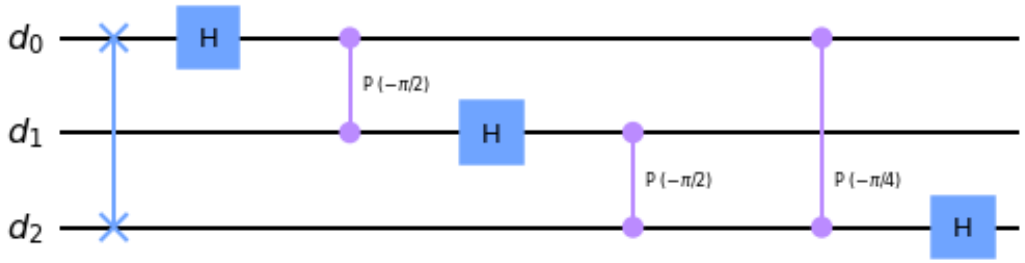
QFT จะทำการเข้ารหัสสถานะ $|x\rangle$ ไปยังเฟส $e^{i\frac{2xy\pi}{n}}$ ของทุกสถานะ $|y\rangle$ โดยมีแอมพลิจูดของทุกสถานะเท่ากันคือ $\frac{1}{\sqrt{n}}$ ซึ่งเป็นสถานะซ้อนทับของคิวบิต ในทางกลับกัน inverse Fourier transformation สำหรับแปลงจากพื้นฐานการคำนวณฟูริเยร์กลับไปเป็นพื้นฐานการคำนวณทั่วไปสามารถเขียนได้ดังนี้

$$QFT^{+}|y\rangle = \frac{1}{\sqrt{n}}\sum_{y=0}^{n-1} e^{-i\frac{2xy\pi}{n}} |x\rangle \quad (21)$$

วงจร QFT และ QFT^{+} แสดงดังภาพที่ 4 และ 5 ตามลำดับ



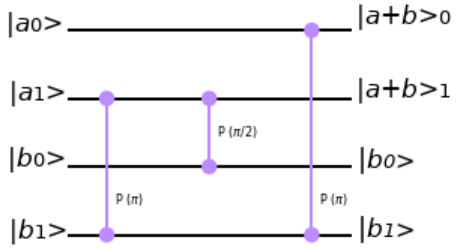
ภาพที่ 4 วงจร QFT สำหรับแปลงจากพื้นฐานการคำนวณทั่วไปไปเป็นพื้นฐานการคำนวณฟูริเยร์สำหรับ 3 คิวบิต



ภาพที่ 5 วงจร QFT^+ สำหรับแปลงจากพื้นฐานการคำนวณฟูริเยร์ไปเป็นพื้นฐานการคำนวณทั่วไป สำหรับ 3 คิวบิต

การเข้ารหัสจากพื้นฐานการคำนวณทั่วไป เช่น สถานะ $|6\rangle$ ไปเป็นพื้นฐานการคำนวณฟูริเยร์ สถานะ $|\tilde{6}\rangle$ สำหรับการประมวลผล 4 คิวบิตที่มีการเรียงคิวบิตแบบ little-endian คิวบิตซ้ายสุดจะถูกหมุนไปด้วยมุม $\frac{6}{2^4} \times 2\pi$ และค่ามุมนี้จะเพิ่มขึ้นเป็น 2 เท่า สำหรับคิวบิตถัดไป และเป็นเช่นนี้เรื่อยไปจนครบทุกคิวบิต

คุณสมบัติอย่างหนึ่งที่สำคัญของพื้นฐานการคำนวณฟูริเยร์คือผลรวมของการแปลงฟูริเยร์ของสองสัญญาณเท่ากับการแปลงฟูริเยร์ของผลรวมของสองสัญญาณนั้น ด้วยคุณสมบัติดังกล่าวจึงสามารถนำการแปลงฟูริเยร์มาใช้สำหรับสร้างวงจรบวกได้ ปี 1998 Thomas G. Dapper[9] ได้เสนอวงจรบวกที่ทำงานบนเครื่องคอมคอมพิวเตอร์เชิงควอนตัมโดยใช้หลักการการแปลงฟูริเยร์ ลำดับของเงื่อนไขการสับเปลี่ยนระหว่างค่ามุมสำหรับการหมุนคิวบิตถูกนำมาใช้เพื่อทำให้การบวกด้วยวงจรควอนตัมประสบความสำเร็จ โครงสร้างวงจรบวกเชิงควอนตัมค่อนข้างใกล้เคียงกับการแปลงฟูริเยร์เชิงควอนตัม (Quantum Fourier transform) แตกต่างที่เงื่อนไขการหมุนคิวบิต ซึ่งจะหมุนเฉพาะ *n* คิวบิตภายนอกที่ควบคุมโดยคิวบิตภายใน ตัวอย่างวงจรบวกเชิงควอนตัมที่ใช้หลักการแปลงฟูริเยร์แสดงดังภาพที่ 6



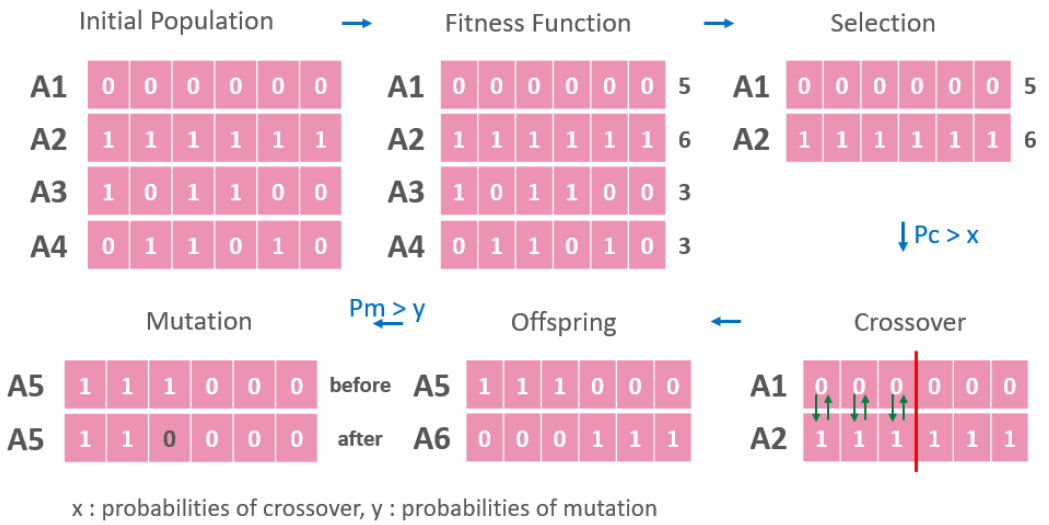
ภาพที่ 6 วงจรบวกเชิงควอนตัมสำหรับ 2 คิวบิต

ตัวเลขฐานสอง a จะถูกเข้ารหัสเพื่อแปลงเป็นพื้นฐานการคำนวณฟูริเยร์โดยใช้ QFT ผลบวกของเลขฐานสอง a และเลขฐานสอง b คำนวณจากการแปลง a ที่เป็นพื้นฐานการคำนวณทั่วไปให้เป็นพื้นฐานการคำนวณฟูริเยร์ $f(a)$ และหลังจากนั้นจึงเพิ่ม b เข้ามา และแปลง a+b เป็นพื้นฐานการคำนวณฟูริเยร์ $f(a+b)$ สุดท้ายพื้นฐานการคำนวณฟูริเยร์จะถูกแปลงกลับไปเป็นพื้นฐานการคำนวณทั่วไปโดยทำ inverse QFT เพื่อให้สามารถทำการวัดค่าสถานะคิวบิตได้

2.8 ขั้นตอนวิธีเชิงพันธุกรรม (Genetic algorithm)

ขั้นตอนวิธีเชิงพันธุกรรมเป็นหนึ่งในอัลกอริทึมสำหรับค้นหาคำตอบที่เหมาะสมที่มีประสิทธิภาพสูง โดยใช้หลักการคัดเลือกแบบธรรมชาติจากการจำลองแนวคิดวิวัฒนาการของสิ่งมีชีวิต[10] Genetic Algorithm (GA) เริ่มต้นโดยการสุ่มค่าคำตอบตามจำนวนประชากรที่กำหนดไว้แล้วทำการเข้ารหัส (encoding) ค่าคำตอบให้เป็นโครโมโซม (Chromosome) ซึ่งโครโมโซมแต่ละตัวจะประกอบไปด้วยยีนเรียงต่อกัน โดยรูปแบบการเข้ารหัสของโครโมโซมเป็นเลขฐานสอง (Binary bit string) จากนั้นจึงเริ่มขั้นตอนกระบวนการทางพันธุกรรม ประกอบด้วย 2 กระบวนการ คือ การสลับสายพันธุ์ (Crossover) และการกลายพันธุ์ (mutation) โดยการสลับสายพันธุ์นั้น จะทำการสุ่มโครโมโซม 2 โครโมโซมที่มีค่าความเหมาะสมมากที่สุด และทำการแลกเปลี่ยนยีนระหว่างโครโมโซม สำหรับวิธีการแลกเปลี่ยนยีนที่ง่ายที่สุด คือ การสลับสายพันธุ์แบบจุดเดียว (One point crossover) ซึ่งจะใช้วิธีการสุ่มจุดที่จะทำการแลกเปลี่ยนยีน ส่วนกระบวนการกลายพันธุ์นั้นจะสุ่มโครโมโซมมาหนึ่งค่า แล้วทำการสุ่มเปลี่ยนค่าของยีน โดยปกติจะใช้การกลายพันธุ์แบบจุดเดียว (One point mutation) ซึ่งจะทำให้เกิดโครโมโซมใหม่เรียกว่า “โครโมโซมลูก (Offspring)” ในแต่ละรอบของกระบวนการ GA นั้น มีตัวแปร 2 ตัวที่ใช้กำหนดว่าแต่ละรอบจะเกิดการสลับสายพันธุ์ หรือการกลาย

พันธุ์ หรือไม่ ได้แก่ ความน่าจะเป็นในการสลับสายพันธุ์ (Probabilities of crossover: P_c) และความน่าจะเป็นในการกลายพันธุ์ (Probabilities of mutation: P_m) ขั้นตอนถัดมาคือการคำนวณค่าความเหมาะสมของโครโมโซม โครโมโซมทั้งหมดจะถูกประเมินค่าความเหมาะสม (Fitness value) โดยสมการค่าความเหมาะสม (Fitness function) ซึ่งอาจแตกต่างกันไป ขึ้นอยู่กับปัญหาที่ต้องการหาคำตอบ ขั้นตอนที่ 4 คือการคัดเลือก (Selection) วิธีการในการคัดเลือกโครโมโซมเพื่อเป็นประชากรในรุ่นถัดไป หรือเพื่อเป็นพ่อแม่ (Parent) สำหรับผลิตประชากรรุ่นถัดไป (offspring) มีอยู่หลายวิธี เช่น การคัดเลือกแบบแข่งขัน (Tournament selection) การคัดเลือกด้วยการสุ่มอย่างทั่วถึง (Stochastic universal sampling) และการคัดเลือกด้วยวงล้อเสี่ยงทาย (Roulette wheel selection) เป็นต้น ขั้นตอนสุดท้ายคือตรวจสอบเงื่อนไขการสิ้นสุดกระบวนการ GA โดย GA จะสิ้นสุดการทำงานก็ต่อเมื่อดำเนินการจนครบตามจำนวนรุ่นของประชากรตามที่ผู้ใช้กำหนด หรือเมื่อค่าคำตอบในหลายๆรอบที่ผ่านมาไม่มีการเปลี่ยนแปลงอย่างมีนัยสำคัญ ภาพที่ 7 แสดงกระบวนการทำงานของ GA โดยใช้วิธีการคัดเลือกแบบ tournament selection



ภาพที่ 7 โครงสร้างทั่วไปของขั้นตอนวิธีเชิงพันธุกรรม (Genetic algorithm) ที่ใช้การคัดเลือกแบบ tournament selection

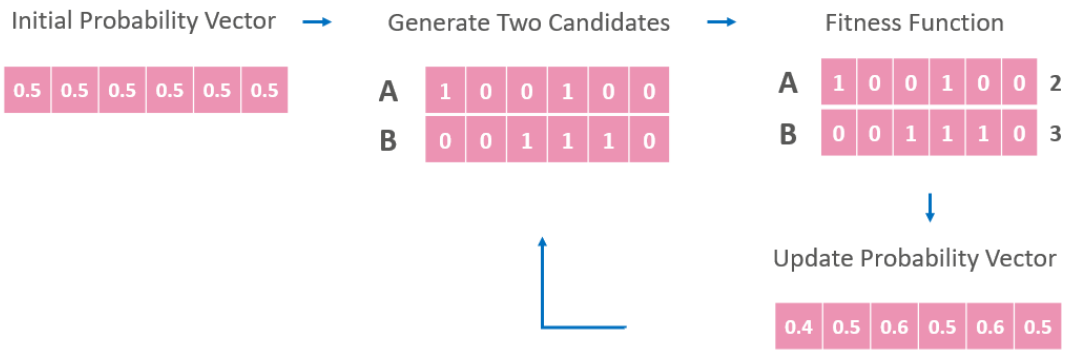
2.9 ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (Compact genetic algorithm)

ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ เป็นหนึ่งในขั้นตอนวิธีเชิงวิวัฒนาการแบบใหม่ ที่มีแนวความคิดในการใช้ตัวแบบความน่าจะเป็น (Probabilistic model) แทนการใช้กลุ่มประชากรในการค้นหาคำตอบ ทำให้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับไม่จำเป็นต้องใช้หน่วยความจำในการเก็บประชากรอีกต่อไป อีกทั้งยังไม่ต้องอาศัยการดำเนินการเชิงพันธุกรรม เช่น การสลับสายพันธุ์ และการกลายพันธุ์ ทำให้การประมวลผลทำได้รวดเร็วยิ่งขึ้น โดยที่ประสิทธิภาพของอัลกอริทึมยังคงเทียบเท่ากับขั้นตอนวิธีเชิงพันธุกรรมแบบเดิม ดังปรากฏในงานวิจัย[11]

การแทนคำตอบของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับจะอยู่ในรูปแบบของเวกเตอร์ความน่าจะเป็น (Probability vector) ซึ่งจะใช้เป็นตัวแบบในการหาการกระจายตัวของคำตอบ โดยแต่ละมิติ (Dimension) ของเวกเตอร์เป็นค่าความน่าจะเป็นที่บิตนั้นๆจะเป็น 1 ประชากรของขั้นตอนวิธีเชิงพันธุกรรมที่มีโครโมโซมยาว 6 บิต เวกเตอร์ความน่าจะเป็นของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับอาจเป็นดังภาพที่ 8 ซึ่งจะมีจำนวนมิติคือ 6 ตามความยาวของโครโมโซม และค่าที่จัดเก็บในเวกเตอร์ความน่าจะเป็นคือค่าความน่าจะเป็นที่แต่ละบิตจะมีค่าเป็น 1 เช่น บิตซ้ายมือสุดมีค่าความน่าจะเป็นที่บิตนี้จะเป็น 1 เท่ากับ 0.4 เป็นต้น ดังนั้นการแทนค่าคำตอบของปัญหาที่สนใจโดยใช้เวกเตอร์ความน่าจะเป็น สามารถลดพื้นที่หน่วยความจำสำหรับจัดเก็บประชากรลงไปได้อย่างมาก

การปรับปรุงคำตอบของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ สามารถทำได้โดยการปรับค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นตามคำตอบที่ดีกว่า โดยการสุ่มค่าคำตอบจากเวกเตอร์ความน่าจะเป็นมา 2 ค่า แล้วทำการพิจารณาว่าคำตอบตัวใดมีค่าความเหมาะสม (Fitness value) มากกว่ากัน จากนั้นจึงทำการปรับปรุงค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นตามคำตอบที่มีค่าความเหมะสมมากกว่า เช่น ถ้าคำตอบที่มีค่าความเหมาะสมดีกว่า บิตที่ 1 คือ 0 ในขณะที่คำตอบที่มีค่าความเหมาะสมน้อยกว่า บิตที่ 1 คือ 1 ดังนั้นในรอบถัดไปเวกเตอร์ความน่าจะเป็นจะถูกปรับค่าความน่าจะเป็นลดลงเพื่อให้เข้าใกล้ 0 มากขึ้น โอกาสที่บิตที่ 1 จะถูกสร้างมาเป็น 0 ในรอบถัดไปก็จะเพิ่มขึ้นด้วย กระบวนการทั้งหมดของ cGA จะถูกวนซ้ำจนกว่าจะตรวจสอบเจอเงื่อนไขการสิ้นสุดกระบวนการ ได้แก่ ค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์เท่ากับ 0 หรือ 1 หรือ เมื่อค่าคำตอบในหลายๆรอบที่ผ่านมาไม่มีการเปลี่ยนแปลงอย่างมีนัยสำคัญ สุดท้ายเวกเตอร์ความน่าจะเป็นจะสะท้อนรูปแบบการกระจายตัวของคำตอบที่ดี

ภาพที่ 8 การแทนค่าคำตอบด้วยเวกเตอร์ความน่าจะเป็นตามความยาวของโครโมโซม



ภาพที่ 9 โครงสร้างทั่วไปของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (Compact genetic algorithm)

นอกจากนี้มีการปรับปรุงกระบวนการคัดเลือกโครโมโซมหรือคำตอบในขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ โดยการเลือกคำตอบจากกลุ่มคำตอบที่ดีที่สุดที่เคยบันทึกไว้ มาใช้ในการเปรียบเทียบกัน ก่อนจะหาคำตอบที่ดีกว่าเพื่อใช้เป็นต้นแบบในการปรับปรุงค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็น วิธีการดังกล่าวเรียกว่า “Compact genetic algorithm with an elite”[12]

บทที่ 3
งานวิจัยที่เกี่ยวข้อง

งานวิจัยทางควอนตัมที่กล่าวถึงความเป็นไปได้ในการนำกฎวิวัฒนาการของดาร์วิน (Darwinian evolution) มาจำลองการทำงานบนเครื่องคอมพิวเตอร์เชิงควอนตัม โดยนักวิจัยฟิสิกส์ที่ชื่อ Wojciech Zurek เขาได้เสนอแนวคิดเรื่อง Quantum Darwinism[13] โดยอธิบายว่าโลกคลาสสิคเกิดขึ้นจากโลกควอนตัม เนื่องจากมีเพียงสถานะควอนตัมเท่านั้นที่สามารถส่งผ่านข้อมูลทางสภาพแวดล้อมในทิศทางที่ถูกต้อง และมีหลายๆสำเนาได้ ซึ่งสถานะควอนตัมสามารถสังเกตได้ในระดับมหภาค และการส่งผ่านข้อมูลสถานะควอนตัมถูกจำกัดจากสภาพแวดล้อมจนเหลือเพียงตัวชี้สถานะ (Pointer states) ซึ่งเป็นสิ่งที่เราสังเกตได้จากมุมมองแบบดั้งเดิม ดังนั้นมุมมองแบบดั้งเดิมของจักรวาลจึงถูกกำหนดโดยสถานะควอนตัมที่รอดชีวิตจากการส่งผ่านข้อมูลทางสิ่งแวดล้อม ซึ่งเป็นไปตามหลักการของการคัดเลือกโดยธรรมชาติของดาร์วิน นอกจากนี้งานวิจัยนี้ยังเสนอวิธีการวัดสถานะควอนตัมซึ่งเป็นความท้าทายอย่างมากสำหรับทฤษฎีกลศาสตร์ควอนตัม เนื่องจากเวกเตอร์สถานะควอนตัมที่อธิบายโดยสมการชโรดิงเจอร์ (Schrödinger equation) เป็นสมการเชิงเส้นที่อธิบายสถานะซ้อนทับของสถานะควอนตัมที่แตกต่างกัน และนำมาใช้ทำนายสถานการณ์ขัดแย้ง เช่น แมวของชโรดิงเจอร์ (Schrödinger's cat) ซึ่งเป็นสถานการณ์ที่ไม่เคยเกิดขึ้นในโลกความเป็นจริงดั้งเดิม โดยใช้วิธีการ non-unitary transformation ของสถานะเวกเตอร์ ณ เวลาที่ทำการวัด เพื่อทำนายค่าที่แน่นอนของสถานะควอนตัมในรูปแบบของความน่าจะเป็นสำหรับแต่ละค่าการวัดที่เป็นไปได้ ดังนั้นแนวคิดของดาร์วินเรื่องการคัดเลือกโดยธรรมชาติเพื่อให้เหลือผู้รอดชีวิตที่แข็งแกร่งที่สุดสามารถถูกจำลองกระบวนการทำงานในอุปกรณ์ใดๆก็ได้ที่สามารถจัดการข้อมูลได้บนพื้นฐานของปรากฏการณ์ทางกลศาตร์ควอนตัม[14]

ปี 1996 Narayanan และ Moore[15] ได้นำเสนออัลกอริทึมที่ได้รับแรงบันดาลใจจากควอนตัม (Quantum-inspired genetic algorithm) และสามารถนำไปแก้ปัญหา TSP (Travelling salesman problem) ได้สำเร็จโดยใช้วิธี Interference crossover คือ กำหนดจำนวน Universe โดยแต่ละ Universe จะมีประชากรที่มีโครโมโซมของตนเอง และสามารถ Crossover กับประชากรของ Universe อื่นๆในแต่ละรุ่นได้ การเชื่อมโยงกันแบบโต้ตอบระหว่าง universe ทำให้การ

คำนวณหาคำตอบของแต่ละ universe สามารถหยุดหาคำตอบด้วยตัวมันเองได้ เมื่อมีหนึ่ง universe หาคำตอบที่เหมาะสมเจอแล้ว

ปี 2000 Bart Terry James และ Jim[16] ได้เสนอ Quantum genetic algorithm (QGA) เป็นอัลกอริทึมที่ใช้ประโยชน์จากผลกระทบเชิงควอนตัม (Quantum effects) ได้แก่ Superposition และ entanglement มาใช้ในการประมวลผลควอนตัมแบบคู่ขนาน (Quantum parallelism) โดยเริ่มจากกำหนดจำนวน Quantum register ตามขนาดของประชากรเพื่อใช้เก็บข้อมูล แต่ละ Quantum register จะถูกทำให้อยู่ในสถานะ Superposition คือแสดงทุกๆสถานะที่เป็นไปได้ของ individual นั่นคือแต่ละ register จะเก็บค่าที่เป็นไปได้ทั้งหมดของ individual จากนั้นประยุกต์ Fitness function เข้าไปในแต่ละ Quantum register จนครบทุกตัว และเก็บผลลัพธ์ของค่า Fitness ที่เป็นไปได้ทั้งหมดไว้ใน Register ชุดที่ 2 การประยุกต์ Fitness function ถือเป็นการสร้าง entanglement ระหว่าง Quantum register ชุดที่ 1 และชุดที่ 2 โดยชุดแรกจะเก็บทุกค่าที่เป็นไปได้ของ Individual ส่วนชุดที่ 2 จะเก็บทุกค่า Fitness ที่เป็นไปได้จากแต่ละค่า Individual การวัดค่าเพื่อหาค่า Fitness จะทำให้ Quantum register ชุดที่ 2 ถูกยุบสถานะ Superposition จนเหลือค่า Fitness เพียงค่าเดียว ในขณะที่ Quantum register ชุดแรก ที่มี entanglement กับชุดที่ 2 จะถูกยุบให้มีสถานะ Superposition ที่เล็กลง เนื่องจากสามารถมีได้หลายค่า เช่น ค่า Fitness ที่วัดได้คือ 7 จะสามารถหาค่า individual ที่มีค่า Fitness เท่ากับ 7 ได้หลายค่า เป็นต้น โดยใช้ตัวดำเนินการทางคณิตศาสตร์ช่วยในการวัดค่า Fitness เพื่อไม่ให้ทำลายสถานะ Superposition ของ individual และการคัดลอกสถานะ Superposition โดยใช้ Linked list แทนการใช้อาร์เรย์ เพื่อให้การทำ crossover สามารถทำได้โดยการย้าย pointer ระหว่างสองรายการแทนที่การคัดลอกทั้งอาร์เรย์ อย่างไรก็ตามยังไม่มีโมเดลทางฟิสิกส์ที่สามารถยืนยันได้ว่าการใช้ linked list จะสามารถรักษาสถานะ Quantum superposition ได้ ถึงแม้ว่าการคัดลอกสถานะ Superposition เป็นเรื่องที่ยากและดูเหมือนจะเป็นไปไม่ได้ที่จะทำการคัดลอกสำเนาที่แน่นอนของสถานะ Superposition แต่ก็เป็นไปได้ที่จะสามารถทำการคัดลอกสำเนาที่ไม่แน่นอน หากข้อผิดพลาดจากการคัดลอกมีขนาดเล็กพอ ก็จะถือว่าเป็นรูปแบบของการกลายพันธุ์โดยธรรมชาติ งานวิจัยนี้สรุปว่า QGA มีข้อดีกว่า GA ทั่วไปเนื่องจากความสามารถของ Individual ในการเป็น Superposition ทำให้มีโอกาสน้อยที่ Individual ที่ดีจะหายไป และที่สำคัญกว่านั้นขนาดของประชากรที่ดูเหมือนจะเพิ่มขึ้นทำให้ได้ประโยชน์จากการขยาย

Building block ที่ดีด้วย เพราะเพิ่มโอกาสที่จะ Crossover แล้วได้ Offspring ที่ดีขึ้นโดย Building block ไม่หายไป ดังนั้นจึงสามารถช่วยปรับปรุงประสิทธิภาพการค้นหาได้อย่างมากมาย อย่างไรก็ตามข้อได้เปรียบเหล่านี้ยังไม่สามารถพิสูจน์ได้ว่าเป็นจริงในปัจจุบัน

ปี 2002 Han[17] แนะนำอัลกอริทึมวิวัฒนาการใหม่ (Novel evolutionary algorithm) ที่ได้รับแรงบันดาลใจจากการคำนวณควอนตัม เขาเรียกมันว่า Quantum-inspired evolutionary algorithm หรือ QEA ซึ่งมีการกำหนดตัวแทนของแต่ละ individual ฟังก์ชันการประเมินผล (Evaluation function) และการเปลี่ยนแปลงของประชากร โดยกำหนดคิวบิตเป็นหน่วยข้อมูลที่เล็กที่สุดสำหรับแสดงความน่าจะเป็น และแทน Individual ด้วยสายอักขระของคิวบิต และ Q-gate ถูกนำมาใช้เป็นตัวดำเนินการเพื่อขับเคลื่อน Individual ไปสู่ผลลัพธ์ที่ดีขึ้น หลังจากงานวิจัยนี้ถูกตีพิมพ์ก็มีงานวิจัยที่เกี่ยวข้องกับอัลกอริทึมทางพันธุกรรมที่ได้รับแรงบันดาลใจจากควอนตัมเพิ่มขึ้นอีกมากมาย

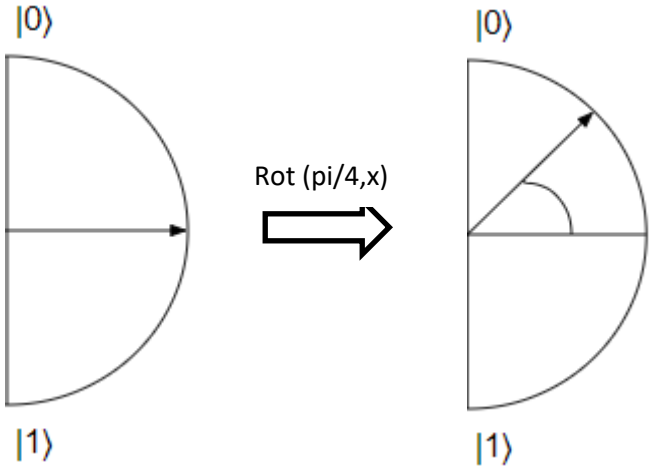
งานวิจัยเกี่ยวกับ QGA[18, 19] ที่ใช้ Quantum state vector เพื่อแสดงรหัสพันธุกรรม (Genetic code) และใช้ Quantum gate เพื่อให้เกิดการวิวัฒนาการของโครโมโซม โดยพบว่าประสิทธิภาพของ QGA ดีกว่าอัลกอริทึมทางพันธุกรรมแบบดั้งเดิม (Conventional genetic algorithm) อย่างมีนัยสำคัญ QGA ใช้ขนาดประชากรน้อย ลู่เข้าผลลัพธ์อย่างรวดเร็ว มีความสามารถในการเพิ่มประสิทธิภาพสำหรับการหาค่าที่เหมาะสมที่สุด และมีความคงทนของวิธีการ อย่างไรก็ตาม QGA ก็ยังมีปัญหาบางอย่าง เช่น การกำหนดทิศทางของมุมในการหมุนโดยอาศัยตารางการค้นหา อันนำไปสู่การเขียนโปรแกรมที่มีเงื่อนไขการตัดสินใจจำนวนมาก นอกจากนี้การกำหนดค่าคงที่ขององศาในการหมุนของมุม ทำให้ค้นหาคำตอบได้ช้าลง[20] และเพื่อแก้ไขปัญหาข้างต้น จึงมีงานวิจัยที่พยายามปรับปรุงกระบวนการวิวัฒนาการของ QGA ให้ดีขึ้นโดยใช้ Quantum mutation operation และ Quantum disaster operation[21] quantum mutation ทำให้บาง Individual มีวิวัฒนาการที่เบี่ยงเบนไปเล็กน้อยจากทิศทางของวิวัฒนาการในปัจจุบัน และป้องกันไม่ให้ individual มีวิวัฒนาการไปทาง Local optimal solution นอกจากนี้ Quantum mutation ยังสามารถทำให้เกิดการย้อนกลับของวิวัฒนาการของแต่ละ individual โดยการสลับค่าความน่าจะเป็นหรือค่าแอมพลิจูดของคิวบิต นอกจากนี้ Quantum NOT gate ถูกนำมาใช้เพื่อให้เกิดความแปรปรวนของโครโมโซม และมาประยุกต์ใช้ในการแปลงจำนวนคิวบิตที่เลือกตามความน่าจะเป็นของ

การกลายพันธุ์แบบสุ่มเพื่อแปลงคิวบิตให้สอดคล้องกับความน่าจะเป็นที่เป็นไปได้ของคิวบิต Quantum mutation จึงช่วยเพิ่มความหลากหลายของประชากร และลดความน่าจะเป็นของการลู่เข้าหาคำตอบของ individual ก่อนถึงเวลาที่เหมาะสม ส่วน Quantum disaster operation ช่วยให้ individual ที่มีวิวัฒนาการไปทาง Local optimal solution สามารถออกมาจาก Local optimal solution ได้ โดยใช้การรบกวน Individual จำนวนมากในกลุ่มประชากร และสุ่มสร้าง individual ใหม่

ปี 2010 Ying[22] ได้เสนอว่าวิธีการนำการคำนวณเชิงควอนตัมมาใช้เพื่อให้บรรลุเป้าหมายบางอย่างในปัญญาประดิษฐ์ (AI) สามารถจำแนกได้ทั้งหมด 3 คลาส ได้แก่ การออกแบบอัลกอริทึมควอนตัมเพื่อแก้ปัญหาใน AI ให้มีประสิทธิภาพมากยิ่งขึ้น การพัฒนาวิธีการสำหรับแก้ปัญหา AI ที่มีประสิทธิภาพยิ่งขึ้นโดยการใช้ไอเดียจากทฤษฎีควอนตัม และการพัฒนาเทคนิค AI ใหม่ๆเพื่อจัดการกับปัญหาในโลกควอนตัม สำหรับคลาสแรกนั้นยังอยู่ในช่วงเริ่มต้นของการพัฒนา และยังไม่มีความคืบหน้าทางด้านงานวิจัยมากนัก ส่วนคลาสที่ 2 งานวิจัยยังคงค่อนข้างกระจัดกระจายและขาดความเชื่อมโยงระหว่างงานวิจัย อย่างไรก็ตาม งานวิจัยสำหรับการพัฒนาวิธีการแก้ปัญหา AI ที่มีประสิทธิภาพยิ่งขึ้นโดยการใช้ไอเดียจากทฤษฎีควอนตัมนั้นได้รับความนิยมเป็นอย่างมากเมื่อเทียบกับงานวิจัยคลาสอื่นๆ แต่บางส่วนของงานวิจัยเหล่านี้ยังจำเป็นต้องมีการวิเคราะห์เชิงทฤษฎีอย่างลึกซึ้งถึงวิธีการที่เป็นทางการสำหรับการพัฒนางานวิจัยเหล่านี้ โดยเฉพาะอย่างยิ่ง จำเป็นต้องมีการวิจัยเชิงทดลองเพิ่มเติมเพื่อทดสอบประสิทธิภาพของอัลกอริทึม และคลาสที่ 3 เป็นงานวิจัยที่ต้องอาศัยการทำงานร่วมกันระหว่างนักวิจัยที่พัฒนาเทคโนโลยี AI กับนักฟิสิกส์ จึงมีความเป็นไปได้ที่เทคโนโลยี AI ที่พัฒนาขึ้นใหม่อาจไม่มีประโยชน์และไม่ได้รับการยอมรับหรือชื่นชมจากนักฟิสิกส์เท่าไหร่นัก ถึงกระนั้นการทำงานร่วมกันอาจนำไปสู่งานวิจัยใหม่ๆที่มีผลกระทบต่อชีวิตประจำวันของคนทั่วไปค่อนข้างสูง

ปี 2012 มีงานวิจัยที่พัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (Compact genetic algorithm) บนเครื่องคอมพิวเตอร์เชิงควอนตัมแบบจำลอง[6] โดยใช้ QCL (Quantum computing language) ซึ่งเป็นภาษาที่ใช้ในการเขียนโปรแกรมรูปแบบหนึ่งสำหรับเครื่องคอมพิวเตอร์เชิงควอนตัม มีการกำหนด Quantum register เพื่อใช้เก็บข้อมูลประชากร และใช้ Hadamard gate เพื่อแปลงสถานะของ Quantum register ให้อยู่ในสถานะ Superposition จากนั้นจึงนำ Qubit rotation มา

ใช้ในการปรับมุมการหมุนของคิวบิตเพื่อให้สถานะของคิวบิตเข้าใกล้สถานะที่ต้องการ เมื่อจำลองภาพสถานะ Superposition ของคิวบิตและการหมุนคิวบิตด้วย Bloch's sphere ได้ดังภาพที่ 2 ซึ่งทำให้ภายหลังการปรับสถานะเวกเตอร์ของคิวบิต ทำให้คิวบิตมีสถานะใกล้ $|0\rangle$ มากขึ้น



ภาพที่ 10 การนำ qubit rotation มาใช้ปรับสถานะเวกเตอร์ของคิวบิตโดยปรับ pi/4

งานวิจัยนี้ยังได้ทำการเปรียบเทียบผลการรันโปรแกรมควอนตัมเพื่อหาคำตอบของปัญหา onemax เทียบกับอัลกอริทึมขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับแบบดั้งเดิม (cGA) ซึ่งพบว่าความถูกต้องและความรวดเร็วของโปรแกรมไม่แตกต่างกัน จึงได้มีการนำเสนอ Quantum cGA แบบใหม่ที่ใช้หลักการการประมวลผลเชิงควอนตัมแบบขนาน (Quantum parallelism) คือการพยายามสร้าง Second candidate ที่มีค่า Fitness ดีกว่าหรือเท่ากับ First candidate โดยใช้ฟังก์ชันควอนตัมที่มีความสามารถในการแสดงสถานะเวกเตอร์ของทุกๆคิวบิตได้ในเวลาเดียวกัน และเมื่อทำการเปรียบเทียบกับอัลกอริทึมดั้งเดิมกับปัญหา Trap ก็พบว่า Quantum cGA ที่ใช้หลักการการประมวลผลเชิงควอนตัมแบบขนานสามารถหาคำตอบที่ถูกต้องได้มากกว่าและประมวลผลได้รวดเร็วกว่า

ในช่วงหลายปีที่ผ่านมามีงานวิจัยที่พยายามนำการคำนวณเชิงควอนตัมมารวมกันกับขั้นตอนวิธีเชิงพันธุกรรม (Genetic algorithm) โดยหวังว่าขั้นตอนวิธีเชิงพันธุกรรมที่ใช้การคำนวณเชิงควอนตัมจะให้ผลลัพธ์ที่รวดเร็วกว่า หรือให้แนวความคิดใหม่ๆในการออกแบบและพัฒนาอัลกอริทึม เช่น quantum-assisted genetic algorithm[5, 21, 23, 24] ที่มีการมอบหมายงานบางส่วนของ

อัลกอริทึม เช่น Mutation operation และ Probabilistic elements ให้คอมพิวเตอร์เชิงควอนตัมประมวลผลให้ ในขณะที่ส่วนการทำ Crossover และการปรับปรุงประชากร (Population update) ยังคงประมวลผลอยู่บนเครื่องคอมพิวเตอร์ดั้งเดิม นอกจากนี้ยังมีงานวิจัยที่พยายามกำหนดขั้นตอนวิธีเชิงพันธุกรรมแบบใหม่ในบริบทของการคำนวณเชิงควอนตัม[25, 26] โดยการสร้างประชากรด้วยสถานะซ้อนทับของคิวบิต ประเมินค่าความเหมาะสมของแต่ละประชากรเพื่อลดมิติของพื้นที่ และใช้การครอสโอเวอร์ระหว่างประชากร โดยยังมีประชากรบางส่วนที่มีสถานะพัวพันกันอยู่ จากนั้นจึงขยายพื้นที่กลับไปเป็นสถานะซ้อนทับใหม่เพื่อขยายมิติพื้นที่กลับไปเหมือนเดิม กระบวนการดังกล่าวจะถูกทำซ้ำหลายๆครั้งจนกระทั่งได้คำตอบที่เหมาะสมที่สุด หรือพบเงื่อนไขในการหยุดกระบวนการ

นอกเหนือจากการนำการคำนวณเชิงควอนตัมมาร่วมกับขั้นตอนวิธีเชิงพันธุกรรมที่มีการเข้ารหัสข้อมูลเชิงควอนตัมในรูปแบบเลขฐานสอง ซึ่งประกอบด้วย 0 กับ 1 แล้ว ปี 2018 มีงานวิจัยที่นำเสนอ Quantum genetic algorithms ที่ปรับวิธีเข้ารหัสข้อมูลเชิงควอนตัมในมิติที่สูงขึ้น[27] นั่นคือใช้การเข้ารหัสข้อมูลในรูปแบบเลขฐานสาม เรียกว่า ternary numeral system หรือ Trit ซึ่งมีหน่วยข้อมูลที่เล็กที่สุดสามสถานะ คือ 0 1 และ 2 ดังนั้นสถานะคิวบิตสามารถแสดงได้สูงสุดสามสถานะในเวลาเดียวกัน เรียกว่า Qutrit เมื่อเปรียบเทียบประสิทธิภาพกับ Quantum genetic algorithms ที่ใช้การเข้ารหัสข้อมูลเชิงควอนตัมในรูปแบบเลขฐานสอง พบว่า Quantum genetic algorithm ที่ใช้การเข้ารหัสข้อมูลเชิงควอนตัมแบบ Qutrit มีประสิทธิภาพที่ดีกว่า และใช้เวลาในการประมวลผลน้อยกว่า นอกจากนี้งานวิจัยนี้ได้นำหลักการของ Quantum disaster operation มาช่วยในการปรับค่าสถานะคิวบิตให้เป็นค่าสถานะเริ่มต้นที่แอมพลิจูดของทุกสถานะย่อยมีค่าเท่ากัน เมื่อการวิวัฒนาการของรูปแบบคำตอบลู่เข้าค่า local optimal solution เพื่อขยายขนาดของประชากรและขยายพื้นที่ในการค้นหาคำตอบที่เป็น global optimal solution

ตั้งแต่เริ่มมีงานวิจัยเกี่ยวกับ Quantum genetic algorithms (QGAs) จนถึงปัจจุบันนี้ QGAs ถูกนำเสนอในวารสารทางวิทยาศาตร์จำนวนมาก[28-32] และประสบความสำเร็จในการนำอัลกอริทึมวิวัฒนาการเชิงควอนตัมหลากหลายรูปแบบไปใช้กับปัญหาการหาค่าเหมาะสม เช่น ปัญหาการกำหนดเวลาบุคลากร (Personnel scheduling problem)[33] ปัญหาการขนส่งอย่างประหยัด (Dynamic economic dispatch problem) เช่น การจ่ายโหลดอย่างประหยัดในระบบโรงไฟฟ้า[34]

การลงทะเบียนภาพหลายเซ็นเซอร์ (Multi-sensor image registration)[35] และปัญหาการถอดรหัส (Cryptanalysis)[36]

แม้ว่างานวิจัยเกี่ยวกับเทคโนโลยีควอนตัมจะเริ่มต้นมาตั้งแต่ปี 1982 โดยนักฟิสิกส์ Richard Feynman ผู้ซึ่งได้รับรางวัลโนเบลสาขาฟิสิกส์ในปี 1957 ได้มีการพูดถึงเครื่องจักรที่สามารถทำงานบนหลักการทางกลศาสตร์ควอนตัม เพื่อจำลองพฤติกรรมของระบบควอนตัมโดยใช้ระบบควอนตัมอีกระบบหนึ่ง[37] ถัดมาในปี 1985 David Deutsch ได้นำเสนอเครื่องควอนตัมทัวริง (Quantum Turing machine) โดยอ้างอิงจากงานวิจัยบุกเบิกของ Alan Turing และได้ระบุอัลกอริทึมที่ออกแบบสำหรับทำงานบนเครื่องคอมพิวเตอร์เชิงควอนตัม[38] นอกเหนือจากงานวิจัยของนักวิจัยด้านฟิสิกส์ควอนตัมและวิทยาการคอมพิวเตอร์เชิงทฤษฎีแล้ว สาขาวิจัยนี้ได้รับความสนใจเป็นอย่างมากในช่วงกลางทศวรรษ 90 เนื่องด้วยในปี 1994 นักคณิตศาสตร์ Peter Shor ได้เสนออัลกอริทึมควอนตัมสำหรับแยกตัวประกอบที่เร็วกว่าอัลกอริทึมที่รู้จักกันดีที่สุดในปัจจุบันซึ่งทำงานบนคอมพิวเตอร์แบบดั้งเดิม เรียกว่า “อัลกอริทึมของชอร์ (Shor’s algorithm)”[39] อย่างที่ทราบกันดีว่าปัจจุบันการเข้ารหัส (Encryption) ส่วนใหญ่มีพื้นฐานมาจากแนวคิดการแยกตัวประกอบจำนวนเต็มขนาดใหญ่ที่เป็นผลคูณของจำนวนเฉพาะขนาดใหญ่สองตัว เช่น RSA encryption เป็นต้น ซึ่งเป็นเรื่องที่ยากมากที่คอมพิวเตอร์แบบดั้งเดิมจะสามารถค้นหาจำนวนเฉพาะขนาดใหญ่สองตัวนั้นเพื่อทำการถอดรหัสคอมพิวเตอร์แบบดั้งเดิมอาจใช้เวลาทั้งชีวิตหรือนานเกินกว่าที่จะสามารถคำนวณได้ อย่างไรก็ตามหากเราสามารถดำเนินการกับคิวบิตจำนวนมากบนเครื่องคอมพิวเตอร์เชิงควอนตัมได้ การค้นหาจำนวนเฉพาะก็สามารถทำได้อย่างมีประสิทธิภาพในเวลา Polynomial time ทำให้พื้นฐานของการเข้ารหัสณ ปัจจุบันส่วนใหญ่ที่ใช้บนอินเทอร์เน็ต มีความเสี่ยงที่จะสามารถถูกถอดรหัสได้โดยง่าย รวมถึงความเสี่ยงในการสร้างความเสียหายให้กับบิตคอยน์ (Bitcoins) และสกุลเงินดิจิทัล (Cryptocurrencies) อื่นๆ แม้ว่างานวิจัยเทคโนโลยีเชิงควอนตัมจะมีปริมาณเพิ่มมากขึ้นอย่างต่อเนื่อง และมีความก้าวหน้าทางเทคโนโลยีเชิงควอนตัมเพิ่มมากขึ้น นักวิจัยยังคงต้องเผชิญกับอุปสรรคสำคัญนั่นคือการควบคุมการคำนวณบนเครื่องคอมพิวเตอร์เชิงควอนตัมให้สามารถทำการคำนวณได้โดยปราศจากสิ่งรบกวนจากภายนอก (Outside noise) เพื่อลดข้อผิดพลาดในการคำนวณเชิงควอนตัม ซึ่งนำมาสู่การหาวิธีการตรวจจับและแก้ไขข้อผิดพลาด (Error correction) ของคิวบิต นอกจากนี้ยังมีปัญหาการสูญเสียอาพันธ์ (Decoherence) คือคิวบิตคงสถานะซ้อนทับได้ในช่วงระยะเวลาหนึ่งเท่านั้น

[40]รวมถึงปัญหา no-cloning ทำให้ไม่สามารถสำเนาค่าคิวบิตได้ เมื่อระบบมีความซับซ้อนมากขึ้น จำนวนคิวบิตที่จำเป็นต้องใช้จะเพิ่มมากขึ้นตามไปด้วย ซึ่งยิ่งจำนวนคิวบิตมาก ก็ยิ่งเพิ่มความไวต่อสัญญาณรบกวนมากขึ้น นี่จึงเป็นเหตุผลสำคัญที่การสร้างเครื่องคอมพิวเตอร์เชิงควอนตัมให้สามารถใช้งานได้จริงเป็นเรื่องที่ท้าทายมาก และยังคงมีงานวิจัยเทคโนโลยีเชิงควอนตัมที่ถูกเผยแพร่ออกมาเพื่อพยายามหาคำตอบของประเด็นปัญหาเหล่านี้อย่างต่อเนื่อง

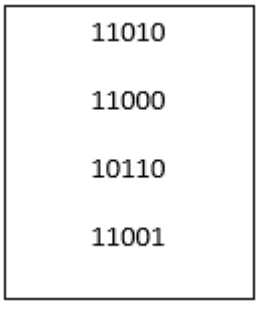
บทที่ 4

แนวคิดและวิธีการดำเนินงานวิจัย

งานวิจัยนี้นำเสนอขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum compact genetic algorithm) สำหรับปัญหายาก โดยได้แบ่งขั้นตอนการดำเนินงานออกเป็น 6 ส่วนหลักได้แก่

4.1 การศึกษาอัลกอริทึมขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (cGA)

ผู้วิจัยได้ทำการศึกษาอัลกอริทึมขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ ซึ่งเป็นหนึ่งในขั้นตอนวิธีเชิงวิวัฒนาการแบบใหม่ ที่มีแนวความคิดในการใช้ตัวแบบความน่าจะเป็น (Probabilistic model) แทนการใช้กลุ่มประชากรแบบเดิมในการค้นหาคำตอบ แนวความคิดนี้ทำให้ขั้นตอนวิธีเชิงพันธุกรรมใช้หน่วยความจำในการเก็บประชากรน้อยลง เนื่องจากไม่มีการใช้ประชากรอีกต่อไป อีกทั้งยังไม่ต้องอาศัยการดำเนินการเชิงพันธุกรรม เช่น การไขว้เปลี่ยน หรือ การกลายพันธุ์ ทำให้การประมวลผลทำได้รวดเร็วยิ่งขึ้น โดยที่ยังคงความสามารถเทียบเท่ากับขั้นตอนวิธีเชิงพันธุกรรมอย่างง่ายที่ใช้อยู่เดิม [11] การแทนคำตอบของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับจะอยู่ในรูปแบบของเวคเตอร์ความน่าจะเป็น (probability vector) ซึ่งจะใช้เป็นตัวแบบในการหาการกระจายตัวของคำตอบ โดยแต่ละมิติ (dimension) ของเวคเตอร์เป็นค่าความน่าจะเป็นที่แต่ละบิตจะเป็น 1 ตัวอย่างเช่น สมมุติว่าประชากรของขั้นตอนวิธีเชิงพันธุกรรมแบบเดิมมีโครโมโซมยาว 5 บิตดังภาพที่ 11 ตัวอย่างของเวคเตอร์ความน่าจะเป็นของขั้นตอนวิธีเชิงพันธุกรรมอาจเป็นดังภาพที่ 12 ซึ่งจะมีจำนวนมิติคือ 5 ตามความยาวของโครโมโซม โดยในเวคเตอร์ความน่าจะเป็นนี้ แต่ละมิติจะเป็นค่าความน่าจะเป็นของการเกิดบิตที่เป็น 1 เช่น ค่า 1.0 แทนความน่าจะเป็นที่โครโมโซมบิตแรกจะเป็น 1 จากประชากรทั้งหมด เป็นต้น



ภาพที่ 11 การแทนประชากรของขั้นตอนวิธีเชิงพันธุกรรมแบบเดิมที่มีโครโมโซมยาว 5 บิต

1.0	0.75	0.25	0.5	0.25

ภาพที่ 12 เวคเตอร์ความน่าจะเป็นของขั้นตอนวิธีเชิงพันธุกรรมแบบเดิมที่มีโครโมโซมยาว 5 บิต

การปรับปรุงคำตอบของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ สามารถทำได้โดยการปรับค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นตามคำตอบที่ดีกว่า โดยในการทำงานของอัลกอริทึมนี้จะมีการสุ่มสร้างตัวอย่าง 2 ตัว ขึ้นมาจากเวกเตอร์ความน่าจะเป็น ซึ่งโครโมโซมของตัวอย่างที่สุ่มขึ้นมานี้ในแต่ละบิตจะเป็น 0 หรือ 1 ขึ้นกับค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็น เช่น ถ้าเวกเตอร์ความน่าจะเป็นมีค่า 0.5 ในทุกมิติ ตัวอย่างประชากรที่สุ่มสร้างขึ้นมาจะมีโอกาสเป็นบิต 1 หรือ 0 เท่าๆ กัน แต่ถ้าเวกเตอร์ความน่าจะเป็นมีค่า 1.0 ทุกมิติ ตัวอย่างที่สุ่มออกมาได้จะเป็นโครโมโซมที่เป็นบิต 1 ทั้งหมด เมื่อสุ่มสร้างตัวอย่างขึ้นมาแล้วก็จะพิจารณาว่าคำตอบตัวใดมีค่าความเหมาะสมมากกว่ากัน เวกเตอร์ความน่าจะเป็นในแต่ละมิติจะถูกปรับตามบิตของคำตอบตัวที่ดีกว่า ถ้าบิตนั้นมีค่าเป็น 1 ความน่าจะเป็นในมิตินั้นก็จะปรับเข้าใกล้ 1 แต่ถ้าเป็น 0 ค่าความน่าจะเป็นในมิตินั้นก็จะลดค่าลง โอกาสที่บิตนั้นจะถูกสร้างมาเป็น 0 ในรอบถัดไปก็จะเพิ่มตามด้วย ในทางกลับกันถ้าบิตของโครโมโซมทั้ง 2 ตัวเหมือนกัน จะไม่มีการปรับค่าเวกเตอร์ความน่าจะเป็นของบิตนั้น ดังนั้นเมื่อผ่านกระบวนการวิวัฒนาการไประยะหนึ่ง เวกเตอร์ความน่าจะเป็นก็จะเป็นรูปแบบการกระจายตัวของคำตอบที่ดี ซึ่งมีขั้นตอนโดยสรุปดังนี้

4.1.1 กำหนดค่าเริ่มต้นของตัวแปรในเวกเตอร์ความน่าจะเป็นในทุกๆ มิติให้มีค่าเป็น 0.5

4.1.2 สุ่มสร้างตัวอย่างคำตอบจากเวกเตอร์ความน่าจะเป็นขึ้นมา 2 ตัว

4.1.3 คำนวณค่าความเหมาะสมของตัวอย่างที่สุ่มมาได้ แล้วพิจารณาว่าตัวใดเป็นผู้ชนะและผู้แพ้

4.1.4 ปรับค่าเวกเตอร์ความน่าจะเป็นตามผู้ชนะ โดยพิจารณาเฉพาะบิตที่ผู้ชนะและผู้แพ้ไม่เหมือนกัน โดยปรับตามเงื่อนไขดังนี้

4.1.4.1 ถ้าบิตตำแหน่งที่ i ของผู้ชนะ เป็น 1 จะปรับค่าเวกเตอร์ความน่าจะเป็นที่ตำแหน่ง i โดยนำค่าความน่าจะเป็นเดิมบวกกับ 1/n เมื่อ n คือจำนวนประชากรที่ถูกกำหนดใช้ในขั้นตอนวิธีเชิงพันธุกรรมอย่างง่าย

4.1.4.2 ถ้าบิตตำแหน่งที่ i ของผู้ชนะ เป็น 0 จะปรับค่าเวกเตอร์ความน่าจะเป็นที่ตำแหน่ง i โดยนำค่าความน่าจะเป็นเดิมลบด้วย 1/n

4.1.5 ตรวจสอบว่าเวกเตอร์ความน่าจะเป็นลู่เข้าสู่คำตอบแล้วหรือไม่ โดยพิจารณาจากค่าความน่าจะเป็นของแต่ละมิติว่าเป็น 0.0 หรือ 1.0 ครบแล้วหรือไม่ ถ้าค่าความน่าจะเป็นลู่เข้าสู่ 0.0 หรือ 1.0 หมดแล้ว ให้จบการทำงาน แต่ถ้ายังลู่เข้าไม่ครบ ให้สุ่มตัวอย่างจากเวกเตอร์ความน่าจะเป็นขึ้นมาใหม่แล้วทำซ้ำขั้นตอนที่ 2 ถึง 5 ต่อไป

4.2 การศึกษาอัลกอริทึมการค้นหาของ Grover โดยใช้ Qiskit ซึ่งเป็นชุดพัฒนาซอฟต์แวร์แบบ open-source สำหรับทำงานกับ OpenQASM และ quantum processor ของ IBM

อัลกอริทึมการค้นหาของ Grover[7] เป็นอัลกอริทึมควอนตัมที่สามารถนำมาใช้ในการเร่งเวลาหาคำตอบของปัญหาการค้นหาข้อมูลแบบไม่มีโครงสร้าง (unstructured search) ที่คอมพิวเตอร์ทั่วไปใช้เวลาในการค้นหาคำตอบนานมาก โดยใช้กลวิธีในการขยายขนาดของแอมพลิจูดสำหรับปัญหาการค้นหาข้อมูลแบบไม่มีโครงสร้างสามารถจำลองได้ดังภาพที่ 13

1	2	3	...	...	W	...	$N = 2^n$

ภาพที่ 13 แสดงรายการที่ต้องการค้นหา(w) โดยกำหนดเป็นสีชมพู ขณะที่รายการอื่นเป็นสีเทา[41]

สมมติมีรายการขนาดใหญ่ *N* รายการ ซึ่งในรายการเหล่านี้มี 1 รายการที่มีคุณสมบัติที่เป็นเอกลักษณ์ที่เราต้องการค้นหา เราจะเรียกรายการนี้ว่าผู้ชนะ และกำหนดให้ผู้ชนะมีสีชมพู ขณะที่รายการอื่นๆเป็นสีเทา ในการหากล่องสีชมพูโดยใช้การคำนวณแบบคลาสสิกอาจจะต้องตรวจสอบเป็นจำนวนครั้งโดยเฉลี่ยทั้งหมด *N*/2 ครั้ง และกรณีที่แย่สุดคือต้องตรวจสอบเป็นจำนวน *N* ครั้ง อย่างไรก็ตามในคอมพิวเตอร์เชิงควอนตัม เราสามารถค้นหารายการที่ต้องการค้นหาด้วยจำนวนครั้งโดยประมาณ $\sqrt{N}$ ด้วยเทคนิคการขยายแอมพลิจูดของ Grover ซึ่งทำให้ประหยัดเวลาเป็นอย่างมากสำหรับการค้นหารายการที่ต้องการจากจำนวนรายการทั้งหมดที่มีขนาดใหญ่มาก นอกจากนี้อัลกอริทึมไม่ได้ใช้โครงสร้างภายในของรายการ ซึ่งทำให้มันใช้ได้กับทุกปัญหาทั่วไป โดยรายการต่างๆจะถูกส่งไปยังเครื่องคอมพิวเตอร์เชิงควอนตัมผ่านการเข้ารหัสรายการให้อยู่ในรูปของฟังก์ชัน ซึ่งจะ

คืนค่าเป็น 0 เมื่อรายการนั้นไม่ใช่รายการที่ต้องการค้นหา และจะคืนค่าเป็น 1 เมื่อรายการนั้นคือรายการที่ต้องการ (ผู้ชนะ) เพื่อให้ใช้เครื่องคอมพิวเตอร์เชิงควอนตัมสำหรับปัญหานี้ได้ จำเป็นจะต้องทำให้ทุกรายการอยู่ในสถานะซ้อนทับ (Superposition) สำหรับฟังก์ชัน และจึงเข้ารหัสฟังก์ชันให้อยู่ในรูปของ unitary matrix เราเรียกฟังก์ชันนี้ว่า Oracle โดยลำดับการทำงานของ Oracle เป็นดังนี้

4.2.1 ทำการเข้ารหัสรายการทั้งหมดในรูปแบบเลขฐานสอง

$$x, w \in (0, 1)^n \ , N = 2^n \tag{22}$$

x คือรายการที่ไม่ต้องการ

w คือรายการที่ต้องการ

N คือขนาดทั้งหมดของรายการ

ดังนั้นเราจะสามารถแทนรายการในรูปของคิวบิตบนเครื่องคอมพิวเตอร์เชิงควอนตัมได้

4.2.2 กำหนด oracle matrix (U_f) เพื่อใช้เป็นตัวดำเนินการกับสถานะพื้นฐานของรายการต่างๆ

4.2.2.1 ถ้าเป็นรายการที่ไม่ต้องการ $|x\rangle$ oracle จะดำเนินการดังนี้

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle = |x\rangle \quad \text{เนื่องจาก } f(x) = 0 \tag{23}$$

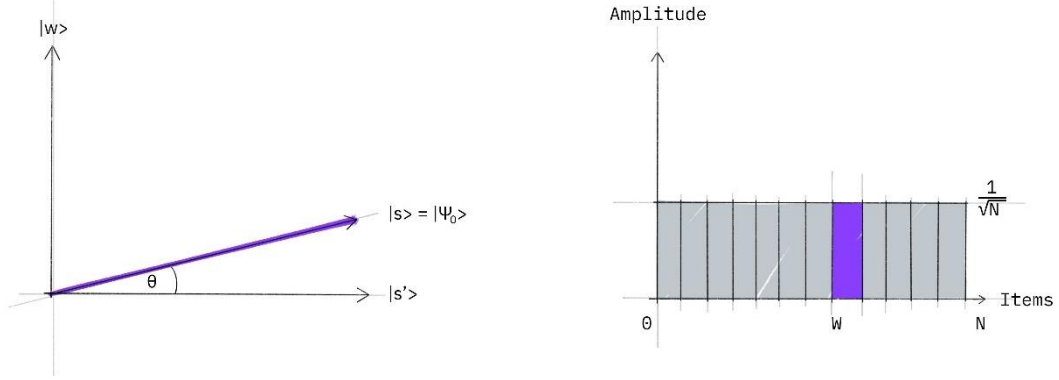
4.2.2.2 ถ้าเป็นรายการที่ต้องการ $|w\rangle$ oracle จะดำเนินการดังนี้

$$U_f |w\rangle = (-1)^{f(w)} |w\rangle = (-1) |w\rangle \quad \text{เนื่องจาก } f(w) = 1 \tag{24}$$

เนื่องจากเราไม่สามารถรู้ตำแหน่งของรายการที่ต้องการค้นหา ดังนั้นเราจึงจำเป็นต้องเดาตำแหน่งของรายการนั้นสามารถทำได้ในรูปของสถานะควอนตัมของคิวบิต เรียกว่า การซ้อนทับของสถานะอย่างสม่ำเสมอ (unitary Superposition) เมื่อเรามีการวัดสถานะเพื่อดูว่าเป็นรายการที่ต้องการค้นหาหรือไม่ จะทำให้สถานะซ้อนทับของคิวบิตถูกยุบจนเหลือเพียงสถานะเดียว ซึ่งความน่าจะเป็นที่จะวัดสถานะของคิวบิตและเจอรายการที่ต้องการค้นหาจากจำนวนรายการทั้งหมด 2^n รายการคือ $1/2^n$ เท่านั้น นั่นหมายความว่าเราอาจจะต้องเดาถึง 2^n ครั้งจึงจะเจอรายการที่ต้องการ การขยายแอมพลิจูดจึงเป็นวิธีการที่คอมพิวเตอร์เชิงควอนตัมช่วยเพิ่มความน่าจะเป็นโดยการขยายแอมพลิจูดของรายการที่ต้องการค้นหา ในขณะเดียวกันก็ลดแอมพลิจูดของรายการที่ไม่ต้องการ ดังนั้นการวัดสถานะสุดท้ายจะคืนรายการที่ต้องการในที่สุด โดยใช้หลักการทางเรขาคณิตเรื่องการ

สะท้อน 2 ครั้งสำหรับการหมุนในระนาบสองมิติ ซึ่งมีการพิจารณาเพียง 2 สถานะ คือสถานะของรายการที่ต้องการ $|w\rangle$ และ สถานะซ้อนทับอย่างสม่ำเสมอ $|s\rangle$ สถานะเวกเตอร์ 2 ตัวนี้มีระนาบ 2 มิติในปริภูมิเวกเตอร์แต่ไม่สามารถตั้งฉากกันได้เนื่องจาก $|w\rangle$ อยู่ในสถานะซ้อนทับซึ่งมีค่าแอมพลิจูดอยู่ที่ $1/\sqrt{N}$ ดังนั้นจึงเพิ่มเวกเตอร์สถานะใหม่คือ $|s'\rangle$ ที่อยู่ระหว่างเวกเตอร์ 2 ตัวนี้และตั้งฉากกับ $|w\rangle$ โดยแปลงมาจาก $|s\rangle$ ที่ไม่รวม $|w\rangle$ และปรับขนาดใหม่ ขั้นตอนการขยายแอมพลิจูดสรุปได้ดังนี้

4.2.2.2.1 ที่เวลา t = 0 กำหนดค่าแอมพลิจูดเริ่มต้นของทุกเวกเตอร์สถานะคือ $1/\sqrt{N}$ และกำหนดให้ $|\psi_0\rangle = |s\rangle$ ดังภาพที่ 14

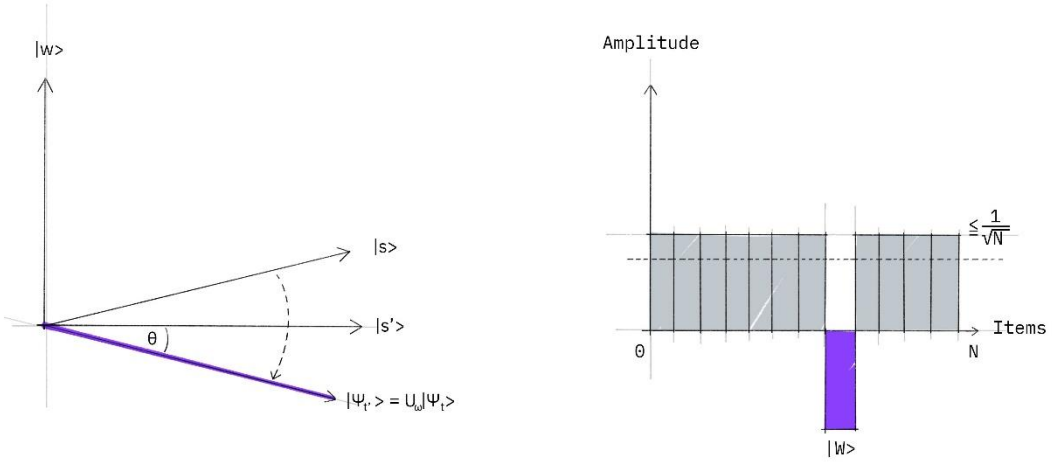


ภาพที่ 14 แสดงค่าแอมพลิจูดเริ่มต้นของทุกสถานะสำหรับ 2 คิวบิต และปริภูมิเวกเตอร์ในระนาบ 2 มิติของ $|w\rangle$ และ $|\psi_0\rangle$[41]

4.2.2.2.2 ประยุกต์การสะท้อนของ oracle (oracle reflection) U_f ไปที่เวกเตอร์สถานะ $|\psi_0\rangle$ ดังสมการ

$$U_f|\psi_t\rangle = |\psi_{t'}\rangle \tag{25}$$

ค่าแอมพลิจูดของสถานะที่สนใจจะถูกกลับค่า ทำให้ค่าเฉลี่ยของแอมพลิจูดของทุกสถานะลดลงดังภาพที่ 15



ภาพที่ 15 แสดงการเปลี่ยนค่าแอมพลิจูดเมื่อประยุกต์ oracle reflection ครั้งที่ 1[41]

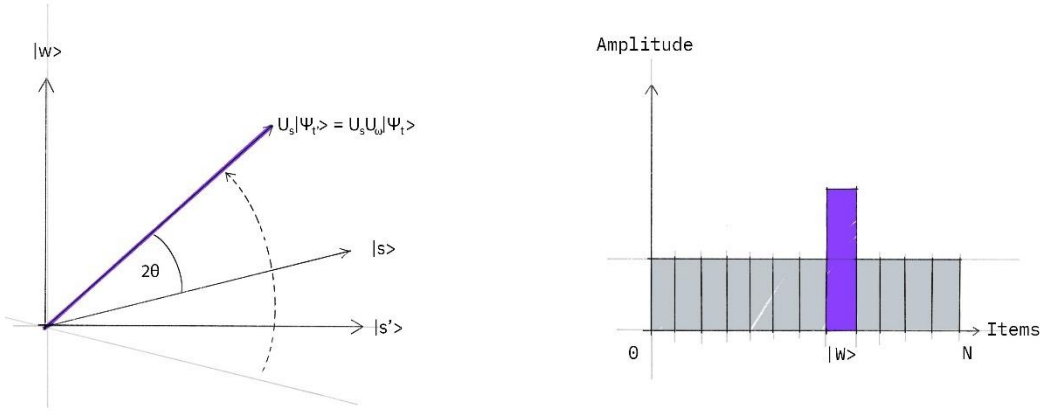
4.2.2.2.3 ประยุกต์การสะท้อนของ oracle อีกครั้งหนึ่งไปที่ $|s\rangle$ สามารถเขียนสัญลักษณ์ได้ดังนี้

$$U_s = 2|s\rangle\langle s| - 1 \tag{26}$$

การแปลงนี้จะทำให้กลายเป็นสถานะ $U_s|\psi_{t'}\rangle$ และเสร็จสิ้นการแปลงจะได้

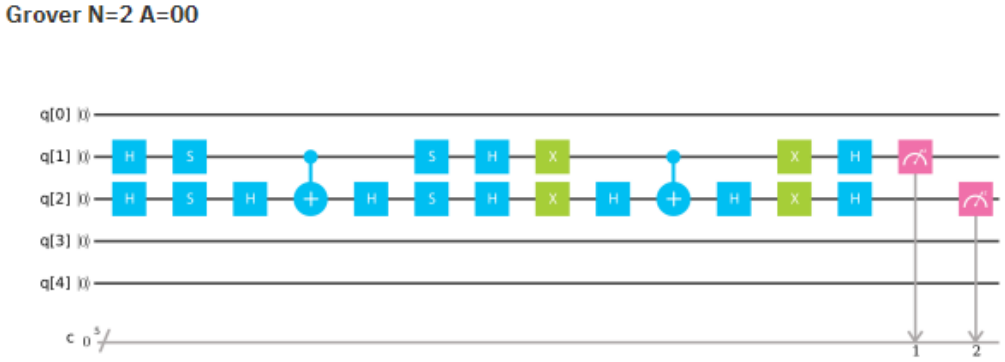
$$|\psi_{t+1}\rangle = U_s U_f |\psi_t\rangle \tag{27}$$

การแปลงโดยประยุกต์ oracle reflection $U_s U_f$ ทำให้สถานะเริ่มต้นของ $|s\rangle$ เปลี่ยนไปมีค่าแอมพลิจูดของสถานะใกล้กับ $|w\rangle$ มากขึ้นดังภาพที่ 16



ภาพที่ 16 แสดงการเปลี่ยนค่าแอมพลิจูดเมื่อประยุกต์ oracle reflection ครั้งที่ 2[41]

ตัวอย่างวงจร Grover ที่รันบน IBM quantum processor สำหรับ 2 คิวบิตที่มีสถานะที่ต้องการคือ 00 ดังภาพที่ 17



ภาพที่ 17 แสดงตัวอย่างวงจร Grover ของ IBM สำหรับ 2 คิวบิตที่มีสถานะที่ต้องการคือ 00[41]

4.3 การประยุกต์ใช้อัลกอริทึมการค้นหาของ Grover เข้ากับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ (cGA) โดยทดสอบกับปัญหา One-max บนเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองของ IBM และ/หรือ เครื่องคอมพิวเตอร์เชิงควอนตัมจริงของ IBM

เนื่องจากอัลกอริทึมการค้นหาของ Grover เป็นอัลกอริทึมที่ใช้ประโยชน์จากการขยายค่าแอมพลิจูดเพื่อเพิ่มความน่าจะเป็นของเวกเตอร์สถานะของคิวบิตที่สนใจมาใช้ในการค้นหาข้อมูลที่ไม่เป็นโครงสร้างได้เร็วกว่าคอมพิวเตอร์ทั่วไปที่ใช้การคำนวณแบบดั้งเดิม และขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับก็ใช้หลักการหาคำตอบที่เหมาะสมจากเวกเตอร์ความน่าจะเป็น ผู้วิจัยจึงได้พัฒนาอัลกอริทึมการค้นหาของ Grover โดยใช้ Qiskit library ของ IBM สำหรับจำลองวงจรควอนตัม และประยุกต์เข้ากับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับเพื่อหาคำตอบของปัญหา โดยเริ่มต้นจากปัญหาพื้นฐานอย่างง่าย ได้แก่ ปัญหาจำนวนบิตหนึ่งมากที่สุด (One-max) ปัญหานี้เป็นปัญหาสมมุติ (Toy problem) และเป็นปัญหาที่แก้ง่ายเมื่อใช้ขั้นตอนวิธีเชิงพันธุกรรม ซึ่งมักจะถูกใช้เป็นปัญหาทดสอบพฤติกรรมของอัลกอริทึมเพื่อเปรียบเทียบความสามารถในการแก้ปัญหาแบบง่าย โดยค่าความเหมาะสมของคำตอบจะเท่ากับจำนวนบิตที่เป็น 1 ถ้ามีบิตที่เป็น 1 ยิ่งมากค่าความเหมาะสมจะมากขึ้น และค่าสูงสุดที่เป็นไปได้จะเกิดในกรณีที่ทุกบิตเป็น 1 ทั้งหมด ในกรณีนี้ค่าความเหมาะสมจะเท่ากับความยาวของโครโมโซม

สำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม เริ่มต้นจากกำหนดวงจรควอนตัม (Quantum circuit) ประกอบด้วย Quantum register และ Classical register ซึ่ง Quantum register จะประกอบด้วยคิวบิตหลายๆคิวบิต และ Classical register ไว้สำหรับเก็บค่าของคิวบิตที่ทำการวัดแล้ว จากนั้นกำหนดเวกเตอร์สถานะเริ่มต้นของ Quantum register ใน Quantum Circuit โดยกำหนดค่าแอมพลิจูดหรือมุมเริ่มต้นของเวกเตอร์สถานะของแต่ละคิวบิตคือ $\frac{\pi}{2}$ โครโมโซมชุดแรกจะถูกสร้างจากเวกเตอร์สถานะของแต่ละคิวบิตใน Quantum register และทำการสังเกตโครโมโซมที่ได้โดยใช้การวัด (Measurement) ซึ่งจะทำให้เวกเตอร์สถานะที่เป็น Superposition ของคิวบิตถูกยุบเหลือเพียงสถานะเดียวและถูกเก็บค่าไว้ที่ Classical register โครโมโซมชุดที่ 2 จะถูกสร้างจากเวกเตอร์สถานะของแต่ละคิวบิตใน Quantum register เช่นเดียวกับโครโมโซมชุดแรก แต่จะมีการประยุกต์อัลกอริทึม Grover 1 รอบ เพื่อให้โครโมโซมชุดที่สองมีค่าแอมพลิจูดหรือมุมเข้าใกล้ค่าความเหมาะสมมากขึ้น ก่อนจะนำไปเปรียบเทียบกับโครโมโซมชุดแรกเพื่อหาโครโมโซมที่มีค่า

ความเหมาะสมที่สุด การเปรียบเทียบค่าความเหมาะสมของโครโมโซมใช้วิธีทางกลศาสตร์ควอนตัมในการเปรียบเทียบ เรียกว่า Quantum comparator เพื่อให้สามารถเปรียบเทียบค่าแต่ละคิวบิตของโครโมโซมได้ จากนั้นจึงทำการปรับเวกเตอร์สถานะของแต่ละคิวบิตตามโครโมโซมที่มีค่าความเหมาะสมที่สุดโดยการปรับค่าแอมพลิจูดหรือมุมของเวกเตอร์สถานะทีละ 0.05 ขั้นตอนดังกล่าวจะถูกวนซ้ำเรื่อยๆจนกว่าค่าแอมพลิจูดหรือมุมเวกเตอร์สถานะจะลู่เข้าสู่รูปแบบการกระจายตัวของคำตอบให้ไปในทิศทางที่ผลเฉลยมีค่าความเหมาะสมสูงขึ้น

สามารถเขียนสรุปขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมได้ ดังนี้

4.3.1 สร้างเวกเตอร์สถานะของคิวบิตด้วยการแจกแจงความน่าจะเป็นอย่างสม่ำเสมอ โดยกำหนดค่าแอมพลิจูดหรือค่ามุมเริ่มต้นคือ $\frac{\pi}{2}$

4.3.2 สังเกตโครโมโซมชุดที่หนึ่งโดยใช้การวัด

4.3.3 สร้างเวกเตอร์สถานะใหม่และประยุกต์ Grover 1 รอบ โดยใช้ค่าความเหมาะสม

4.3.4 สังเกตโครโมโซมชุดที่สองโดยใช้การวัด

4.3.5 เปรียบเทียบค่าความเหมาะสมระหว่างโครโมโซมชุดที่หนึ่งและชุดที่สอง เพื่อหาโครโมโซมที่มีค่าความเหมาะสมที่สุด

4.3.6 ปรับปรุงเวกเตอร์สถานะตามค่าแอมพลิจูดหรือมุมของโครโมโซมที่มีค่าความเหมาะสมที่สุด

4.3.7 ทำซ้ำขั้นตอนที่ 2 – 6 จนกระทั่งเวกเตอร์สถานะของคิวบิตลู่เข้าค่าเหมาะสมที่สุดที่ต้องการ

4.4 เปรียบเทียบประสิทธิภาพในการประมวลผลระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับปัญหา One-max

สำหรับการทดลองเพื่อเปรียบเทียบประสิทธิภาพในการประมวลผลของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) กับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) เป็นการทดลองเพื่อหาคำตอบของปัญหา One-max Quantum cGA จะถูกประมวลผลบนเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองที่รันอยู่บนคลาวด์ของ IBM และ/หรือ เครื่องคอมพิวเตอร์เชิงควอนตัมจริงของ IBM ส่วน Classical cGA จะถูกประมวลผลบนเครื่องคอมพิวเตอร์

ของผู้วิจัยคือคอมพิวเตอร์ยี่ห้อ ASUS CPU Intel Core i5-3317U (1.70 GHz, 3 MB L3 Cache, up to 2.60 GHz) เพื่อวิเคราะห์ความแตกต่างของประสิทธิภาพในการประมวลผลใน 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ และจำนวนครั้งในการประเมินค่าความเหมาะสม (Function evaluation)

ปัจจุบันเครื่องคอมพิวเตอร์เชิงควอนตัมจริงของ IBM มีอัตราความผิดพลาดของเครื่องคอมพิวเตอร์เชิงควอนตัมอันเนื่องมาจากอัตราความผิดพลาดของควอนตัมเกต การวัด การสื่อสารข้ามอุปกรณ์ และประสิทธิภาพคอมไพเลอร์ของวงจรควอนตัม[42] ดังแสดงในตารางที่ 3 ซึ่งเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองยังไม่ได้มีการเปิดเผยจาก IBM ถึงอัตราความผิดพลาดในการทำงาน นอกจากนี้เทคโนโลยีของเครื่องคอมพิวเตอร์เชิงควอนตัมจริงของ IBM ในปัจจุบันยังมีข้อจำกัดคือรองรับการทำงานที่จำนวนคิวบิตไม่มาก ทำให้การทดลองรันขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) บนเครื่องคอมพิวเตอร์เชิงควอนตัมจริงของ IBM สำหรับปัญหาการหาค่าเหมาะสมใดๆ สามารถทำได้เฉพาะกับปัญหาการหาค่าเหมาะสมที่สามารถเข้ารหัสด้วยจำนวนคิวบิตน้อยๆ เช่น One-max เป็นต้น ในขณะที่เครื่องคอมพิวเตอร์เชิงควอนตัมจำลองที่รันอยู่บนคลาวด์ของ IBM สามารถรองรับการทำงานของจำนวนคิวบิตที่มากกว่า ทำให้ผู้วิจัยสามารถทดลองรันขั้นตอนวิธีที่นำเสนอเพื่อหาคำตอบของปัญหา One-max ที่เข้ารหัสด้วยจำนวนคิวบิตที่มากขึ้น รวมทั้งยังสามารถนำไปหาคำตอบของปัญหาการหาค่าเหมาะสมที่มีความซับซ้อนมากขึ้นได้แก่ TSP

	Tenerife (IBM Q Experience)	Tokyo (IBM Q Network)	Melbourne (IBM Q Experience)	IBM Q System One (IBM Q Network)
Two-qubit (CNOT) error rates x10-2				
mean	4.02	2.84	N/A	1.69
best	2.24	1.47	N/A	0.97
worst	5.76	7.12	N/A	2.85
Single-qubit error rates x10-3				
mean	1.65	1.99	N/A	0.41
best	0.69	0.64	2.54	0.19
worst	3.44	6.09	N/A	0.82

ตารางที่ 3 ตารางแสดงอัตราความผิดพลาดของเครื่องคอมพิวเตอร์เชิงควอนตัมจำนวน 4 เครื่องของ IBMQ

4.4.1. <u>รายละเอียดการกำหนดค่าต่างๆ ข้อจำกัด และสภาพแวดล้อมในการทดลอง</u>

ในเบื้องต้นผู้วิจัยได้นำปัญหา One-max มาใช้ทดสอบประสิทธิภาพในการประมวลผลของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) และขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับดั้งเดิม (Classical cGA) โดยclassical cGA ถูกรันบนแมชชีน CPU Intel Core i5-3317U (1.70 GHz, 3 MB L3 Cache, up to 2.60 GHz) และ Chipset คือ Mobile Intel HM76 Express Chipset ส่วน Quantum cGA ถูกรันบนเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองบนคลาวด์ของ IBM ที่รองรับการประมวลผลวงจรควอนตัม โดยกำหนดขนาดของประชากรสำหรับ Classical cGA คือ 20 กำหนดจำนวน shot ของ Quantum cGA คือ 1024 shot และกำหนด step size ในการปรับค่ามุมหรือแอมพลิจูดของ Quantum cGA คือ 0.05 เนื่องจากขนาดของประชากรสำหรับ Classical cGA คือตัวกำหนดขนาดของการอัพเดตเวกเตอร์ความน่าจะเป็น (step size) ซึ่งคำนวณจากอัตราส่วนระหว่าง 1 และขนาดประชากร ดังนั้น step size ของ cGA จึงเท่ากับ 0.05 ซึ่งเทียบเท่ากับขนาดของการอัพเดตมุมหรือแอมพลิจูดของ Quantum cGA โดยผู้วิจัยเลือก

จำนวนบิตที่ใช้ในการทดลองเริ่มต้นจากจำนวนคิวบิตน้อยๆ คือ 4 คิวบิต ถึง 6 คิวบิต เพราะข้อจำกัดของ quantum processor simulator ของ IBM ที่ผู้วิจัยใช้ในการทดลองยังไม่สามารถรองรับการประมวลผลเชิงควอนตัมของจำนวนคิวบิตตั้งแต่ 8 คิวบิตเป็นต้นไป ผู้วิจัยทดลองรันทั้งหมด 50 ครั้งเพื่อหาค่าเฉลี่ยของประสิทธิภาพที่ต้องการเปรียบเทียบ โดยจำแนกการเปรียบเทียบประสิทธิภาพในการประมวลผลออกเป็น 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ และจำนวนครั้งในการประเมินค่าความเหมาะสม (function evaluation)

4.4.2. <u>การเปรียบเทียบประสิทธิภาพเรื่องความถูกต้องของคำตอบ</u>

การเปรียบเทียบประสิทธิภาพในด้านความถูกต้องของคำตอบ เนื่องจากปัญหาที่นำมาทดสอบประสิทธิภาพเปรียบเทียบระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) และขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) คือปัญหา One-max ซึ่งผลเฉลยที่ดีที่สุดของปัญหา One-max คือทุกบิตของโครโมโซมมีค่าเป็น 1 ซึ่งพบว่าขั้นตอนวิธีทั้งสองสามารถหาผลเฉลยที่ดีที่สุดของปัญหา One-max ได้ จึงกล่าวได้ว่าประสิทธิภาพในการสามารถหาผลเฉลยที่ดีที่สุดของปัญหา One-max ของทั้งสองขั้นตอนวิธีเท่ากัน

4.4.3. <u>การเปรียบเทียบประสิทธิภาพเรื่องจำนวนครั้งในการประเมินค่าความเหมาะสม</u>

การเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสม (function evaluation) คือการเปรียบเทียบจำนวนครั้งที่ใช้ประเมินค่าความเหมาะสมระหว่าง 2 อัลกอริทึม จากตารางที่ 4 แสดงให้เห็นว่า ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) ให้ผลลัพธ์ที่มีประสิทธิภาพดีกว่าขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) โดย Quantum cGA ใช้จำนวนครั้งโดยเฉลี่ยในการประเมินค่าความเหมาะสมของ 4 คิวบิตอยู่ที่ 4.3 ครั้ง ในขณะที่ Classical cGA ใช้จำนวนครั้งโดยเฉลี่ยในการประเมินค่าความเหมาะสมของ 4 บิตเท่ากับ 5.6 ครั้ง ซึ่งมากกว่า Quantum cGA ประมาณ 1.3 เท่า เมื่อพิจารณาที่ 5 คิวบิต Quantum cGA ใช้จำนวนครั้งโดยเฉลี่ยในการประเมินค่าความเหมาะสมเท่ากับ 6.3 ครั้ง ขณะที่ Classical cGA ใช้จำนวนครั้งโดยเฉลี่ยในการประเมินค่าความเหมาะสมสำหรับ 5 บิต เท่ากับ 8.2 ครั้ง ซึ่งมากกว่าประมาณ 1.3 เท่า และเมื่อพิจารณาจำนวนครั้งโดยเฉลี่ยในการประเมินค่าความเหมาะสมของ Quantum cGA สำหรับ 6 บิตเท่ากับ 13.5 ครั้ง ส่วน Classical cGA ใช้จำนวนครั้งโดยเฉลี่ยในการประเมิน

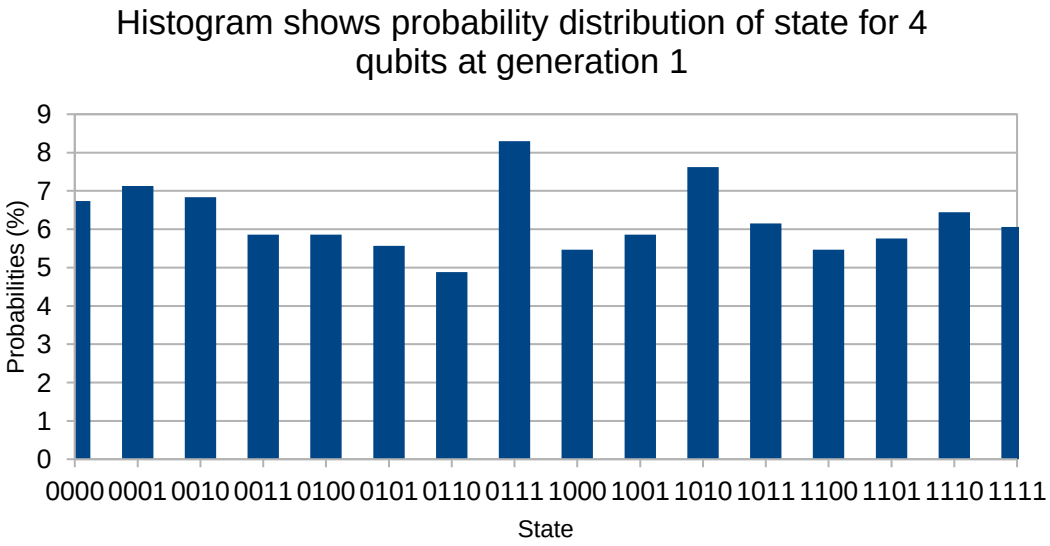
ค่าความเหมาะสมสำหรับ 6 คิวบิตเท่ากับ 14 ครั้ง ซึ่งมากกว่า Quantum cGA ประมาณ 1 เท่า สาเหตุที่ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) ให้ผลลัพธ์ในเชิงของจำนวน function evaluation ที่ดีกว่าขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) เป็นเพราะวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่นำเสนอมีการประยุกต์อัลกอริทึมการค้นหาของ Grover ซึ่งเป็นอัลกอริทึมสำหรับการค้นหาฐานข้อมูลที่ไม่เรียงลำดับเพื่อเพิ่มค่าความน่าจะเป็นของสถานะคำตอบที่สนใจ จึงทำให้มีโอกาสที่จะได้คำตอบที่ต้องการสูงขึ้น และเป็นการปรับปรุงกระบวนการวิวัฒนาการของ Classical cGA ให้ลู่เข้าหาคำตอบที่เหมาะสมได้เร็วขึ้น โดยใช้ประโยชน์จากอัลกอริทึมที่มีคุณสมบัติการคำนวณเชิงควอนตัม

บิต/คิวบิต	จำนวน function evaluation ของ Classical cGA (ครั้ง)			จำนวน function evaluation ของ Quantum cGA (ครั้ง)		
	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3
4	6.2	5.4	5.2	4.4	3.6	4.8
5	8.5	8.7	7.4	6.2	6.4	6.2
6	14.9	13.1	14.1	14.8	13.2	12.6

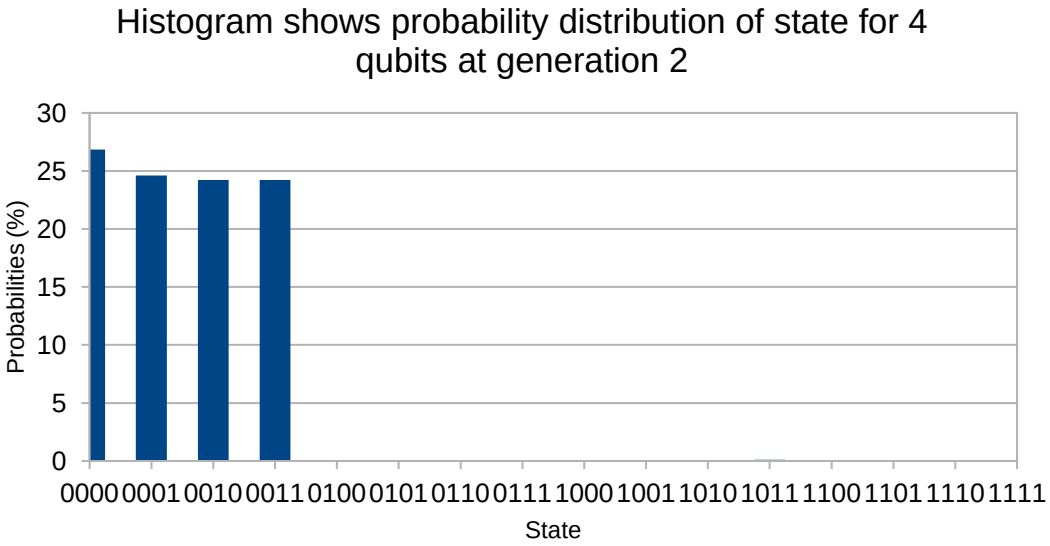
ตารางที่ 4 ตารางแสดงการเปรียบเทียบจำนวน function evaluation (ครั้ง) ระหว่าง Classical cGA และ Quantum cGA สำหรับปัญหา One-max

4.4.4. Histrogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ

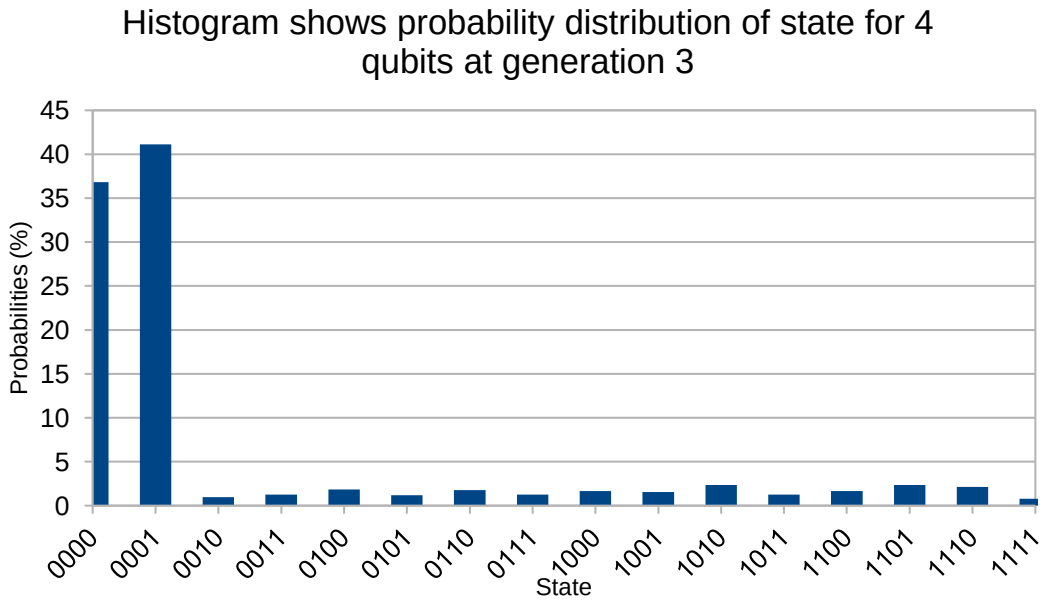
4.4.4.1 ตัวอย่างภาพ Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 4 คิวบิต สำหรับ generation ที่ 1 - 4 แสดงดังภาพที่ 18 – 21



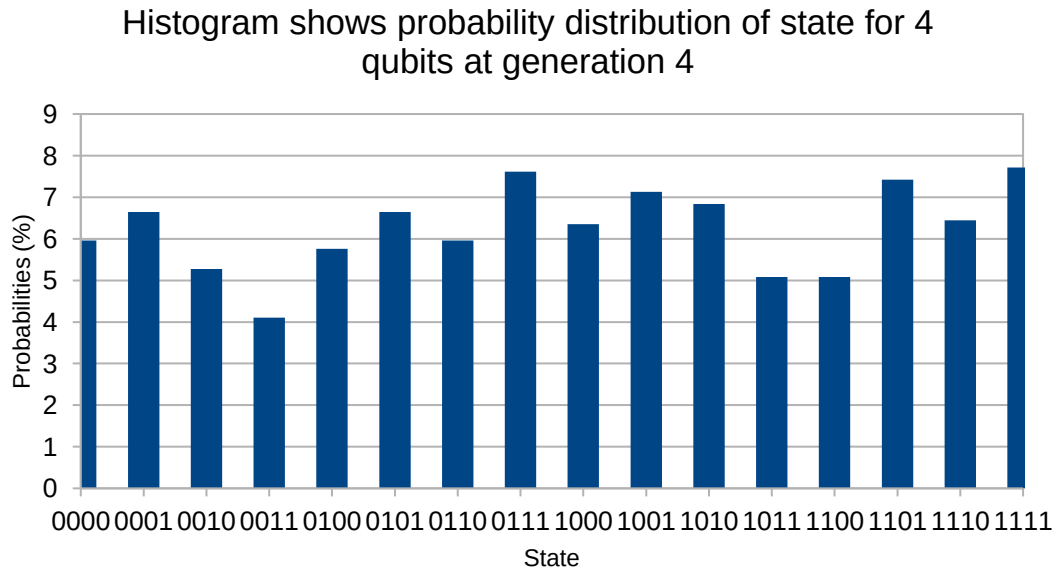
ภาพที่ 18 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 4 คิวบิต ที่ generation 1



ภาพที่ 19 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 4 คิวบิต ที่ generation 2

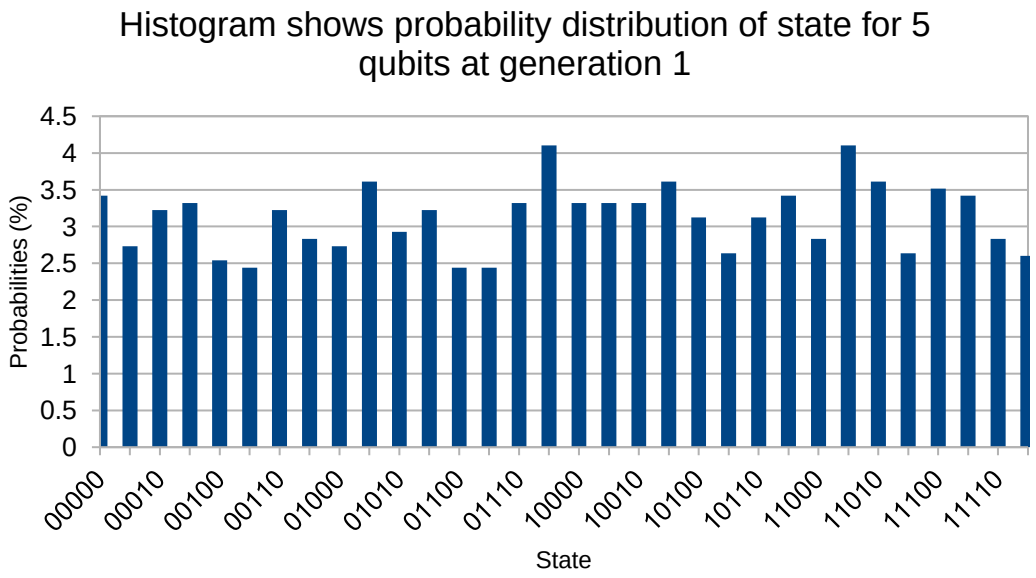


ภาพที่ 20 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 4 คิวบิต ที่ generation 3

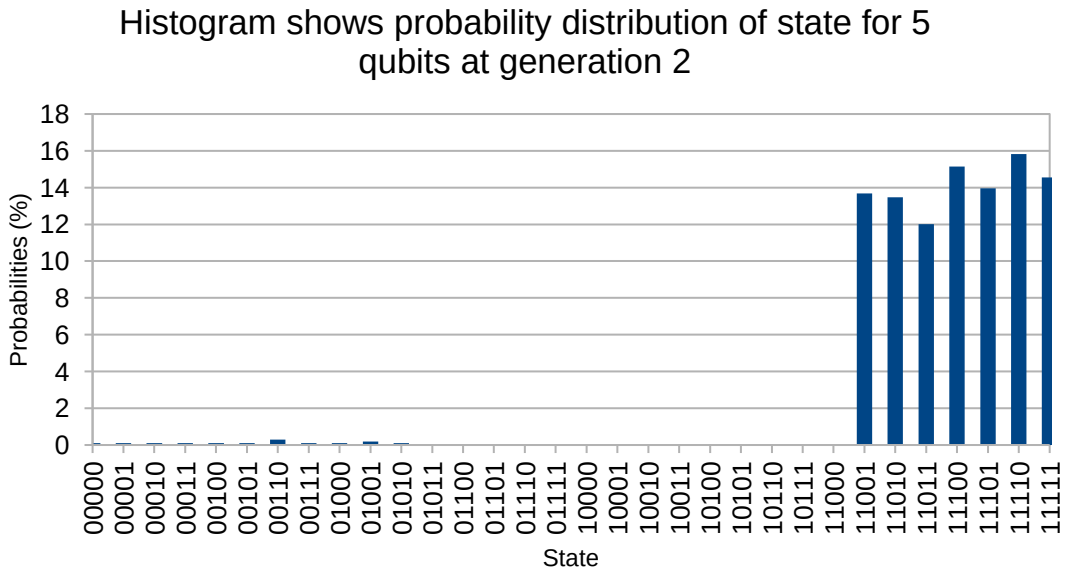


ภาพที่ 21 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 4 คิวบิต ที่ generation 4

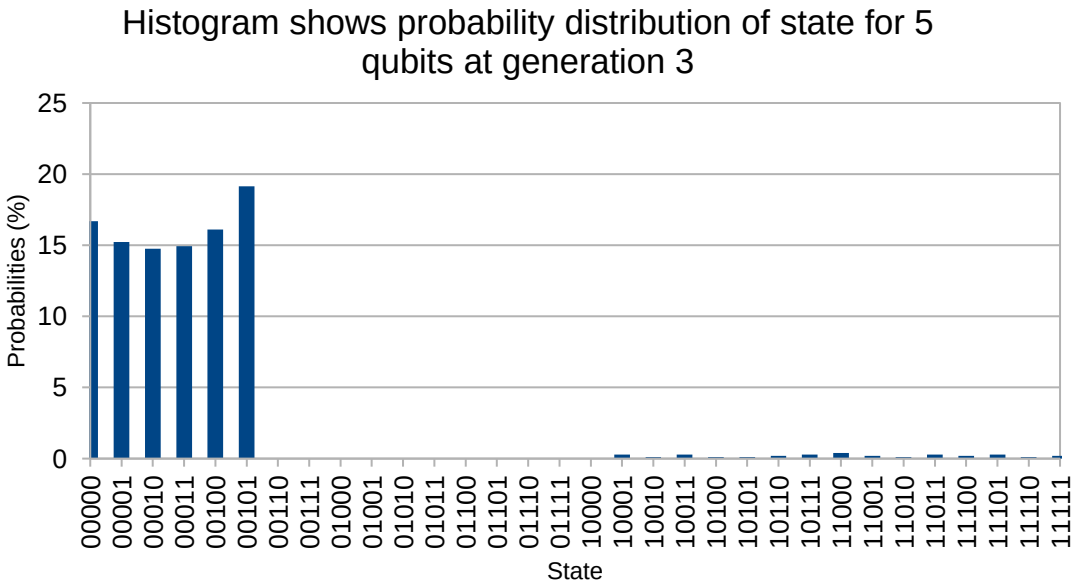
4.4.4.2 ตัวอย่างภาพ Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิตสำหรับ generation ที่ 1 - 6 แสดงดังภาพที่ 22 – 27



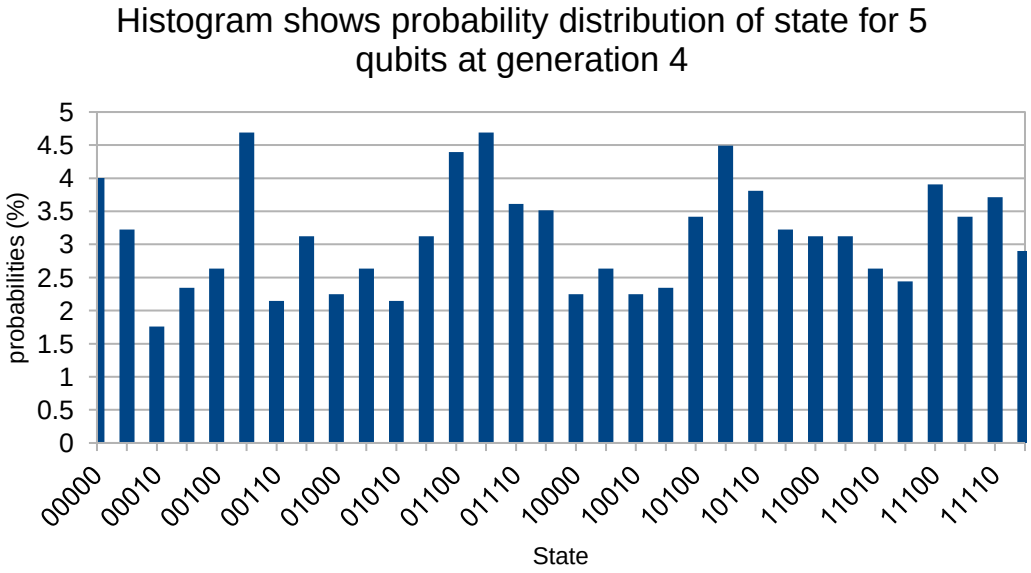
ภาพที่ 22 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 1



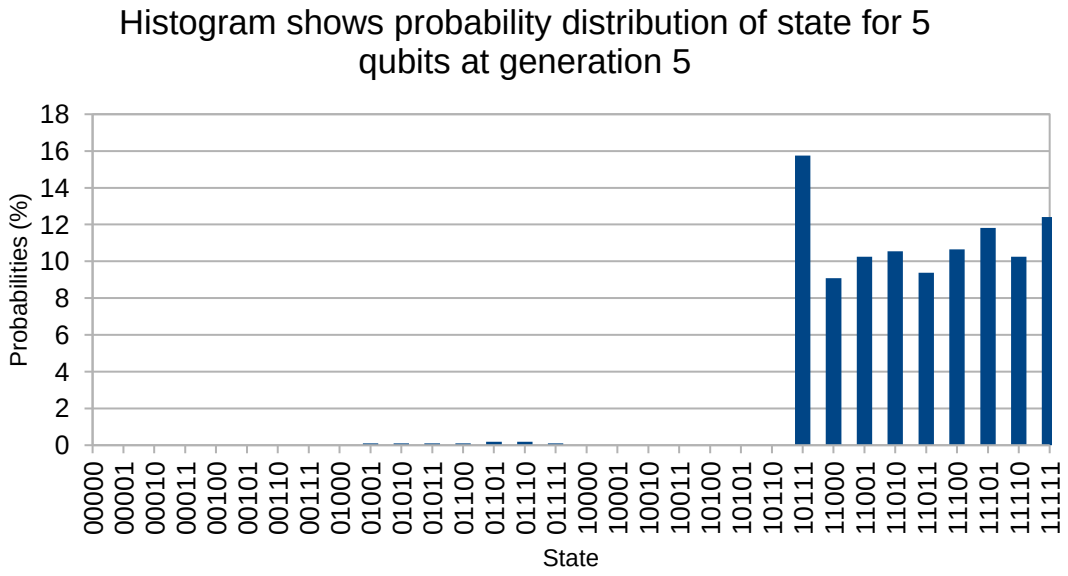
ภาพที่ 23 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 2



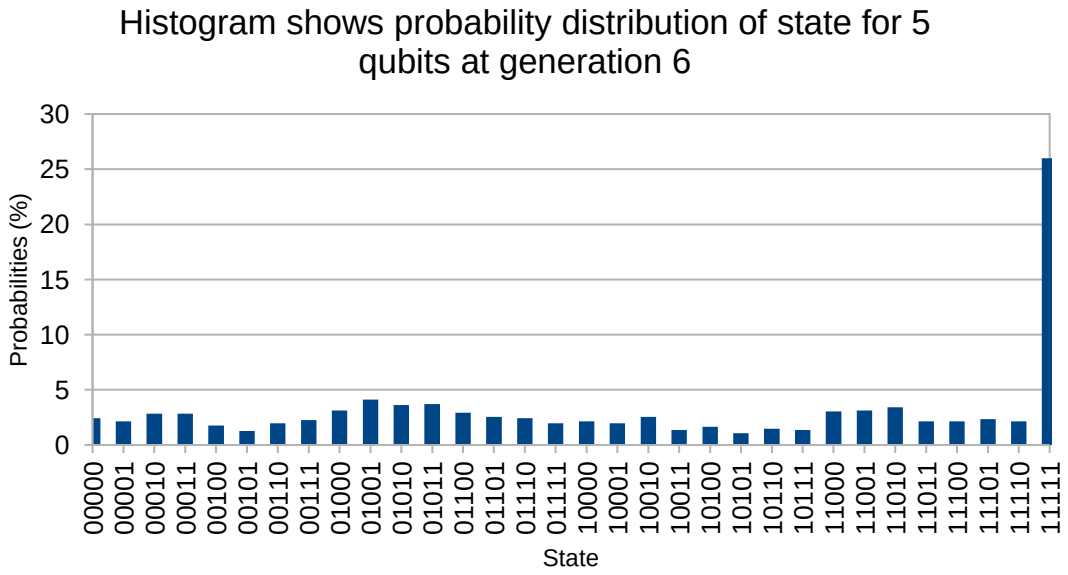
ภาพที่ 24 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 3



ภาพที่ 25 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 4

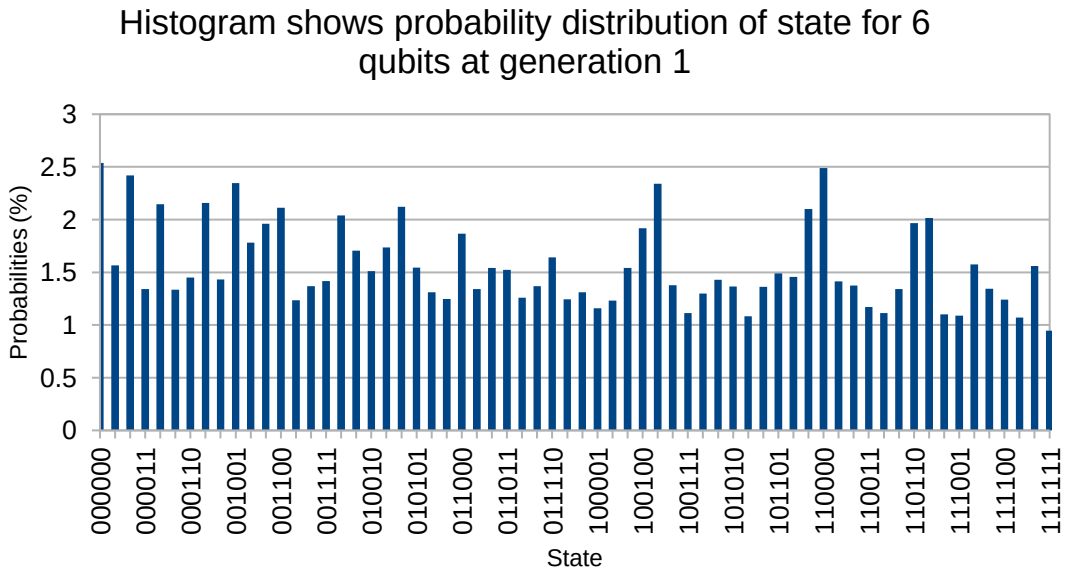


ภาพที่ 26 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 5

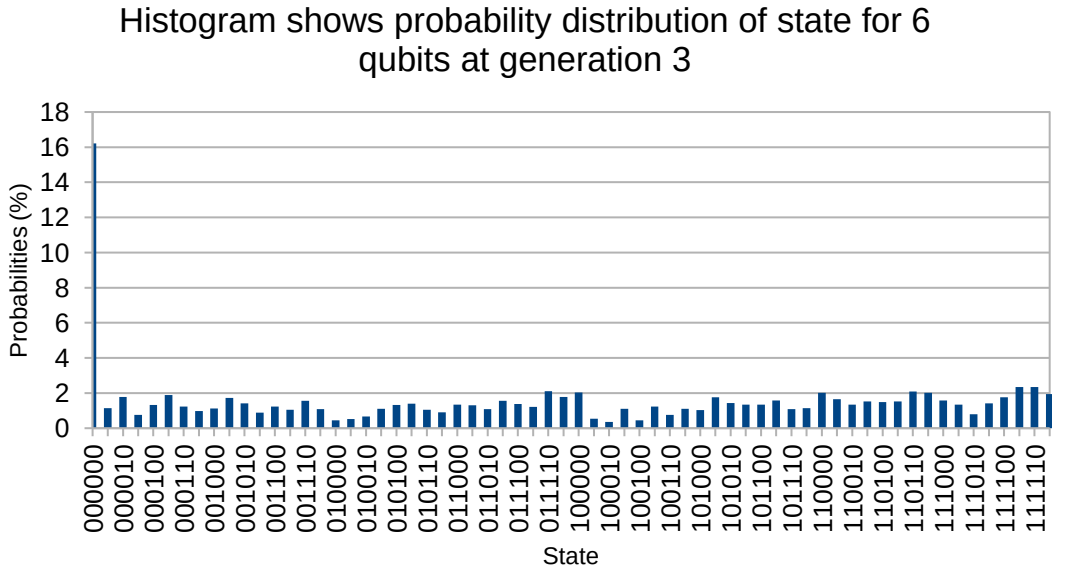


ภาพที่ 27 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 5 คิวบิต ที่ generation 6

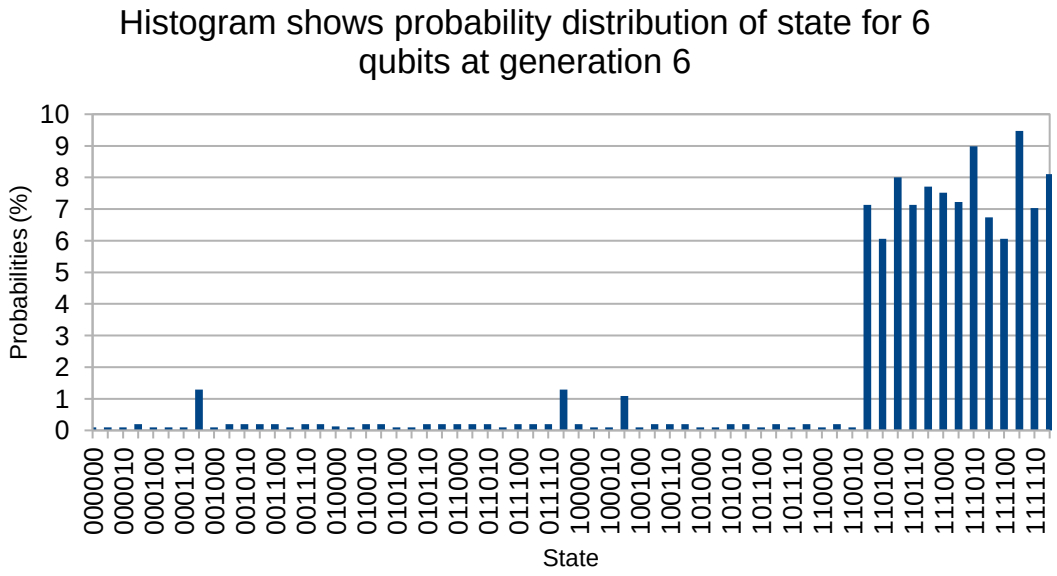
4.4.4.3 ตัวอย่างภาพ Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต สำหรับบาง generation แสดงดังภาพที่ 28 – 33



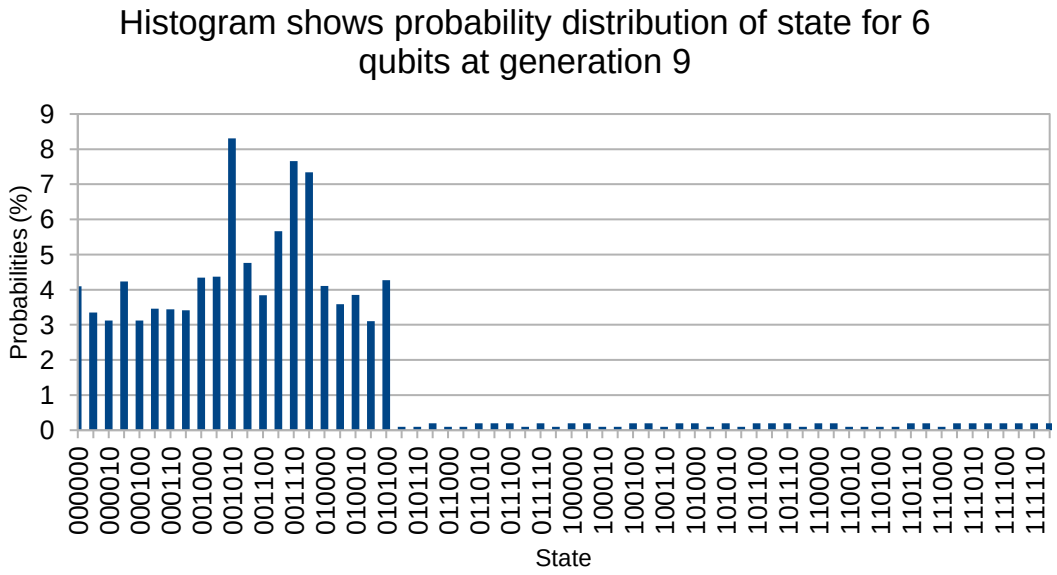
ภาพที่ 28 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 1



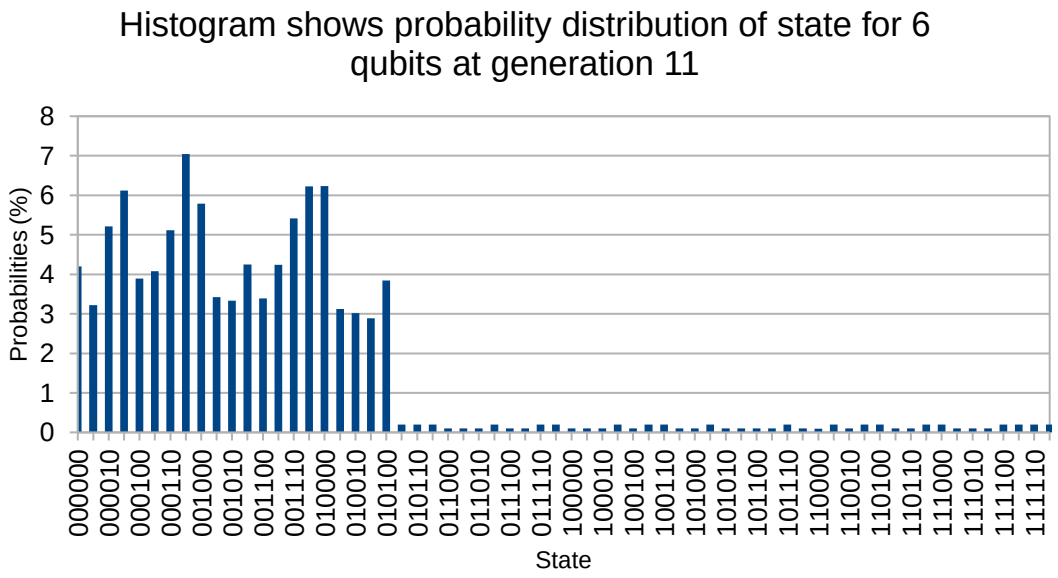
ภาพที่ 29 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 3



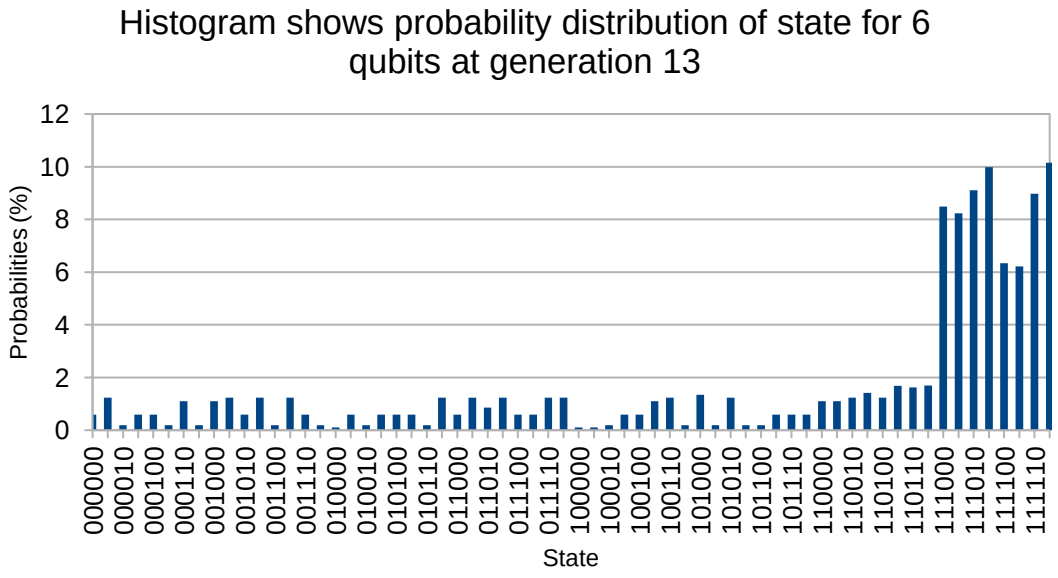
ภาพที่ 30 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 6



ภาพที่ 31 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 9



ภาพที่ 32 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 11



ภาพที่ 33 Histogram แสดงการกระจายตัวของค่าความน่าจะเป็นของเวกเตอร์สถานะ 6 คิวบิต ที่ generation 13

4.4.5. สรุปผลการเปรียบเทียบประสิทธิภาพในการประมวลผลระหว่างขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัมและขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดดั้งเดิมสำหรับปัญหาง่าย One-max

การเปรียบเทียบประสิทธิภาพในการประมวลผลระหว่างขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) และขั้นตอนวิธีทางพันธุกรรมแบบกระชับ (Classical cGA) แบ่งเป็น 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ และจำนวนครั้งในการประเมินค่าความเหมาะสม (function evaluation) โดยในเบื้องต้นผู้วิจัยได้ทำการทดลองกับปัญหาง่ายได้แก่ One-max จากผลการทดลองในเบื้องต้นพบว่าทั้งสองอัลกอริทึมสามารถหาผลเฉลยที่ดีที่สุดของปัญหา One-max ได้ จึงกล่าวได้ว่าประสิทธิภาพในการหาผลเฉลยที่ดีที่สุดของปัญหา One-max ของทั้งสองอัลกอริทึมเท่ากัน อย่างไรก็ตามเมื่อพิจารณาในแง่จำนวนครั้งของการประเมินค่าความเหมาะสม Classical cGA มีจำนวน function evaluation มากกว่า Quantum cGA ทั้งสำหรับ 4 คิวบิต 5 คิวบิต และ 6 คิวบิต โดยมากกว่าประมาณ 1.3 เท่า 1.3 เท่า และ 1 เท่า ตามลำดับ สาเหตุที่เป็นเช่นนี้เนื่องจากขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่นำเสนอมีการประยุกต์อัลกอริทึมการค้นหาของ Grover ซึ่งเป็นอัลกอริทึมสำหรับการค้นหาฐานข้อมูลที่ไม่เรียงลำดับ และสามารถค้นหาข้อมูลได้เร็วขึ้นโดยใช้หลักการคำนวณเชิงควอนตัม ได้แก่ สถานะซ้อนทับ (Superposition) ทำให้สถานะย่อยๆของคิวบิตสามารถถูกประมวลผลในเวลาเดียวกันบนรีจิสเตอร์ตัวเดียวกันได้ และใช้วิธีการขยายค่าแอมพลิจูดเพื่อเพิ่มค่าความน่าจะเป็นของสถานะคำตอบที่สนใจ จึงทำให้มีโอกาสที่จะได้คำตอบที่ต้องการสูงขึ้น ซึ่งถือเป็นการเร่งกระบวนการวิวัฒนาการของ Classical cGA ให้ลู่เข้าหาคำตอบที่เหมาะสมได้เร็วขึ้น จะสังเกตได้ว่าเมื่อจำนวนคิวบิตเพิ่มขึ้นประสิทธิภาพของ Quantum cGA ในแง่ของจำนวน function evaluation เมื่อเปรียบเทียบกับ Classical cGA มีอัตราส่วนที่ลดลง เนื่องจากเมื่อจำนวนคิวบิตเพิ่มขึ้น อัตราความผิดพลาดของเครื่องคอมพิวเตอร์เชิงควอนตัมอันเนื่องมาจากความผิดพลาดของควอนตัมเกต การวัด การสื่อสารข้ามอุปกรณ์ และประสิทธิภาพคอมไพเลอร์ของวงจรควอนตัม ก็เพิ่มขึ้นตามไปด้วย จึงทำให้ในกรณีที่จำนวนคิวบิตมากๆ Quantum cGA มีแนวโน้มที่จะใช้จำนวนครั้งในการประเมินค่าความเหมาะสม

จนได้ผลเฉลยที่ดีที่สุดของปัญหาที่สนใจมากกว่า Classical cGA อย่างไรก็ตามในปัจจุบันมีงานวิจัยทางด้านควอนตัมของ IBM ในการพยายามลดอัตราส่วนความผิดพลาดดังกล่าวข้างต้น[42] และทำให้ Quantum processor รองรับการประมวลผลจำนวนหลายคิวบิตและหลายตัวดำเนินการ ซึ่งจะมีผลในการเพิ่มประสิทธิภาพการคำนวณเชิงควอนตัมในอนาคต

4.5 การปรับปรุงขั้นตอนวิธีเชิงพันธุกรรมแบบกระซับชนิดควอนตัมสำหรับทดสอบกับปัญหายากบนเครื่องคอมพิวเตอร์เชิงควอนตัมจำลองของ IBM

การปรับปรุงอัลกอริทึมขั้นตอนวิธีเชิงพันธุกรรมแบบกระซับชนิดควอนตัม (Quantum cGA) เพื่อให้สามารถหาคำตอบของปัญหายาก ได้แก่ TSP ซึ่งปัจจุบันขั้นตอนวิธีเชิงพันธุกรรมแบบกระซับชนิดดั้งเดิม (Classical cGA) ยังไม่สามารถหาคำตอบของปัญหาดังกล่าวได้ภายใน Polynomial time[11] TSP เป็นปัญหาจำพวก NP-hard คือปัญหาที่ยากและยังไม่มีวิธีการที่ใช้เวลาแบบโพลีโนเมียลในการแก้ปัญหาได้ โดยปัญหา TSP นี้เป็นปัญหาที่ทำการตัดสินใจหาเส้นทางการเดินทางเมื่อมีเมืองหรือสถานที่ที่ต้องเดินทางไปจำนวน *N* เมือง การเดินทางจะเริ่มเดินทางจากเมืองใดเมืองหนึ่งในจำนวน N เมือง โดยเส้นทางการเดินทางนั้นๆ จะต้องเดินทางผ่านเมืองทุกเมืองโดยไม่เดินทางซ้ำเมืองเดิม และกลับมาที่เมืองที่เริ่มต้นเดินทาง ลักษณะการเดินเหมือนการเดินวนรอบ เช่น พนักงานขายเดินทางไปขายสินค้าให้กับลูกค้าจานวน 10 รายได้แก่เมือง A ถึงเมือง J โดยเมือง G เป็นที่ตั้งของศูนย์กระจายสินค้าของพนักงานขายรายนี้ พนักงานขายรายนี้จะเดินทางเริ่มต้นจากเมือง G แล้วเดินไปตามเส้นทางดังนี้ G-A-C-F-E-D-B-J-I-H-G ซึ่งเป็นการเดินทางจากเมือง G ต่อด้วยการเดินทางไปเมือง A และ C ไปเรื่อยๆ จนกระทั่งลูกค้าทุกรายในจำนวน 10 เมืองได้รับการเยี่ยมจากพนักงานขายแล้วพนักงานขายก็ย้อนกลับมาที่เมือง G เช่นเดิม เราสามารถเขียนสมการอธิบายปัญหา TSP ได้ดังนี้

$$X_{ij} = \begin{cases} 1, & the\ path\ goes\ from\ city\ i\ to\ city\ j \\ 0, & otherwise \end{cases} \tag{28}$$

โดยที่ $X_{ij} = 1$ ถ้ามีเส้นทางระหว่างเมือง *i* ไปเมือง *j* และ $X_{ij} = 0$ ถ้าไม่มีเส้นทางระหว่างเมือง *i* ไปเมือง *j* สำหรับ *N* เมือง กำหนดให้ U_i เป็นตัวแปรเทียม โดยที่ $i = 0, 1, \dots, N$ และให้ C_{ij} คือ

ระยะทางจากเมือง i ไปเมือง j ดังนั้นสามารถเขียนฟังก์ชันวัตถุประสงค์ (Objective function) สำหรับปัญหา TSP ได้ดังนี้

$$
\begin{aligned}
& min \sum_{i=0}^{N} \sum_{j \neq i, j=0}^{N} C_{i,j} X_{i,j} \\
& 0 \leq X_{i,j} \leq 1 && i, j = 0, \dots, N \\
& U_i \in Z && i = 0, \dots, N \\
& \sum_{i=0, i \neq j}^{N} X_{i,j} = 1 && j = 0, \dots, N \\
& \sum_{j=0, j \neq i}^{N} X_{i,j} = 1 && i = 0, \dots, N \\
& U_i - U_j + N X_{i,j} \leq N - 1 && 1 \leq i \neq j \leq N
\end{aligned}
\tag{29}
$$

ถ้าใช้วิธีการตรวจสอบทุกความเป็นไปได้ของเส้นทางการเดินทางของพนักงานขาย (Brute-force method) เราจะต้องตรวจสอบมากถึง (N – 1)! ครั้ง ในกรณีที่มีการกำหนดจุดเริ่มต้นของเมืองเป็นเมืองที่ 1 เสมอ เมื่อประมาณความซับซ้อนของการคำนวณ (Time complexity) เทียบเท่า $O(N!)$ ครั้ง ถ้าจำนวนเมืองที่พนักงานขายต้องแวะมีทั้งหมด 11 เมือง โดยเริ่มต้นจากเมืองที่ 1 เสมอเส้นทางที่เป็นไปได้ทั้งหมดที่พนักงานขายสามารถเดินทางได้คือ 10! หรือประมาณ 39,916,800 เส้นทาง และเมื่อจำนวนเมืองเพิ่มมากขึ้น จำนวนเส้นทางที่เป็นไปได้ทั้งหมดจะเพิ่มขึ้นแบบเอกซ์โพเนนเชียล และไม่สามารถใช้วิธี Brute-force ในการค้นหาเส้นทางที่สั้นที่สุดได้ภายในเวลา Polynomial ดังนั้นปัญหา TSP จึงเป็นปัญหาที่ถูกจัดให้อยู่ในกลุ่มปัญหายาก (NP-hard)

4.6 เปรียบเทียบประสิทธิภาพในการประมวลผลระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับปัญหายาก

การเปรียบเทียบประสิทธิภาพในการประมวลผลของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Quantum cGA) สำหรับปัญหายาก เปรียบเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (Classical cGA) โดย Quantum cGA ถูกประมวลผลบนคอมพิวเตอร์เชิงควอนตัมจำลองของ IBM (IBM QASM simulator) เนื่องด้วยข้อจำกัดของจำนวนคิวบิตที่อนุญาตให้ใช้ในเครื่องคอมพิวเตอร์ควอนตัมจริงของ IBM มีไม่เกิน 10 คิวบิต ซึ่งปัญหา TSP จำเป็นต้องใช้

จำนวนคิวบิตในการประมวลผลมากกว่า 10 คิวบิต สำหรับ Classical cGA ถูกประมวลผลบนเครื่องคอมพิวเตอร์ของผู้วิจัยคือคอมพิวเตอร์ยี่ห้อ ThinkPad CPU AMD Ryzen 5 PRO 3500U w/ Radeon Vega Mobile Gfx 2.10 GHz Ram 8.00 GB เพื่อวิเคราะห์ความแตกต่างของประสิทธิภาพในการประมวลผลระหว่างสองขั้นตอนวิธีดังกล่าว

บทที่ 5
การพัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับที่ทำงานบน IBM QASM simulator

ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับที่ทำงานบนเครื่องคอมพิวเตอร์ดั้งเดิมนั้น ใช้ตัวแบบความน่าจะเป็น (Probabilistic model) แทนการใช้กลุ่มประชากร ดังนั้นการแทนคำตอบของขั้นตอนวิธีดังกล่าวจะอยู่ในรูปแบบของเวกเตอร์ความน่าจะเป็น (Probability vector) ซึ่งจะใช้เป็นตัวแบบในการหาการกระจายตัวของคำตอบ โดยแต่ละมิติของเวกเตอร์เป็นค่าความน่าจะเป็นที่แต่ละบิตจะเป็น 1 การพัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับที่สามารถทำงานได้บนเครื่องคอมพิวเตอร์เชิงควอนตัมนั้นจะแตกต่างออกไป ตัวแบบในการหาการกระจายคำตอบจะอยู่ในรูปแบบของเวกเตอร์ความน่าจะเป็นซึ่งจะถูกใช้เป็นข้อมูลอ้างอิงสำหรับกำหนดสถานะของคิวบิตที่ถูกเก็บไว้ในควอนตัมรีจิสเตอร์ เพื่อนำคุณสมบัติที่สำคัญของการประมวลผลเชิงควอนตัม ได้แก่ สถานะซ้อนทับของคิวบิต (Superposition) มาใช้เป็นประโยชน์สำหรับการประมวลผลเชิงควอนตัมแบบคู่ขนาน (Quantum parallelism) โดยเป็นการประมวลผลของทุกสถานะซ้อนทับของคิวบิตในเวลาเดียวกัน และใช้การวัดสถานะของคิวบิตทุกตัวในควอนตัมรีจิสเตอร์เพื่อดูผลลัพธ์สุดท้ายของแต่ละคิวบิตว่าเป็น 0 หรือ 1 โดยความน่าจะเป็นที่แต่ละคิวบิตจะเป็น 0 หรือ 1 นั้นขึ้นอยู่กับค่าแอมพลิจูดของแต่ละสถานะของคิวบิต ณ ขณะที่อยู่ในสถานะซ้อนทับ การวัดจะทำให้สถานะซ้อนทับถูกยุบ (Collapse) เหลือเพียงสถานะเดียวของแต่ละคิวบิต ส่วนการพิจารณาค่าความเหมาะสมของคำตอบ และการปรับค่าเวกเตอร์ความน่าจะเป็นตามคำตอบที่ดีกว่า ในเบื้องต้นของงานวิจัยยังคงพิจารณาค่าความเหมาะสม และปรับค่าเวกเตอร์ความน่าจะเป็นตามคำตอบที่ดีกว่าบนเครื่องคอมพิวเตอร์แบบดั้งเดิม เนื่องด้วยการเปรียบเทียบค่าความเหมาะสมบนเครื่องคอมพิวเตอร์เชิงควอนตัม ณ ปัจจุบันจำเป็นต้องใช้ Quantum comparator มาช่วยในการเปรียบเทียบ ซึ่งผู้วิจัยจะอธิบายส่วนการพัฒนา Quantum comparator ในบทขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (เวอร์ชันปรับปรุง) ของงานวิจัยนี้ จากนั้นจึงใช้การหมุนคิวบิต (Qubit rotation) ไปตามเวกเตอร์ความน่าจะเป็นในแต่ละมิติด้วยค่ามุมที่คำนวณได้จากเวกเตอร์ความน่าจะเป็น ดังนั้นโอกาสที่คิวบิตนั้นจะถูกสร้างในรอบถัดไปแล้วมีค่าใกล้เคียงกับคำตอบที่ดีก็จะมีมากขึ้น เมื่อผ่านกระบวนการวิวัฒนาการไประยะหนึ่งเวกเตอร์ความน่าจะเป็นจะมีรูปแบบการกระจายตัวของคำตอบที่ดี อย่างไรก็ตามผู้วิจัยทำการออกแบบและพัฒนาขั้นตอนวิธีการเชิงพันธุกรรมแบบกระชับสำหรับทำงานบนเครื่อง IBM QASM simulator เท่านั้น เนื่องด้วยเครื่อง IBM QASM simulator มีจำนวนคิวบิตให้ใช้เพียงพอกับการแก้ปัญหา TSP ที่ผู้วิจัยนำมาใช้เป็นปัญหาต้นแบบสำหรับงานวิจัยนี้ และเครื่อง simulator ดังกล่าวเทียบได้กับเครื่องคอมพิวเตอร์เชิงควอนตัมในอุดมคติที่แทบจะไม่มี Noise ดังนั้นงานวิจัยนี้จึงไม่ได้กล่าวถึงสัญญาณรบกวน (Noise) ที่อาจเกิดขึ้นเมื่อใช้งานบนเครื่องคอมพิวเตอร์ควอนตัมจริง และ

วิธีการตรวจจับและแก้ไขข้อผิดพลาด (Error correction) ของคิวบิตอันเนื่องมาจากสัญญาณรบกวนดังกล่าว

5.1 การลดรูปปัญหา TSP ให้อยู่ในรูปแบบการตัดสินใจของแบบจำลอง Ising (Ising model)

การนำปัญหาการหาค่าเหมาะสม (optimization problem) มาหาคำตอบโดยใช้การประมวลผลด้วยเทคโนโลยีเชิงควอนตัม จำเป็นต้องมีการแปลงปัญหาที่สนใจให้อยู่ในรูปแบบที่สามารถสร้างวงจรควอนตัมเพื่อแทนกลุ่มคำตอบของปัญหาดังกล่าวได้ รูปแบบหนึ่งที่ค่อนข้างได้รับความนิยมเป็นอย่างมาก คือ แบบจำลอง Ising (Ising model) เนื่องจากแบบจำลองดังกล่าวเป็นแบบจำลองทางสถิติที่ช่วยให้สามารถระบุการเปลี่ยนเฟสได้ ซึ่งเป็นขั้นตอนหนึ่งที่สำคัญสำหรับการควบคุมคิวบิตให้ได้ผลลัพธ์ตามที่ต้องการ ปี 2014 จึงปรากฏงานวิจัยที่รวบรวมสูตรการแปลงปัญหา NP ให้อยู่ในรูปแบบแบบจำลอง Ising (Ising formulation)[43] ดังนั้นงานวิจัยนี้จึงนำ Ising formulation สำหรับปัญหา TSP มาใช้ในการแปลงปัญหา TSP ให้อยู่ในรูปแบบการตัดสินใจของแบบจำลอง Ising เพื่อให้สามารถประมวลผลได้บนเครื่องคอมพิวเตอร์เชิงควอนตัม นอกจากนี้ Ising formation ยังเป็นฟังก์ชันวัตถุประสงค์ในการคำนวณหาค่าความเหมาะสมของคำตอบแต่ละรูปแบบ

แบบจำลอง Ising ของปัญหา TSP ใช้สปินทั้งหมด $(N-1)^2$ สปิน เมื่อ *N* คือจำนวนเมือง และผู้วิจัยกำหนดให้พนักงานขายเริ่มต้นเดินทางจากเมืองที่ 1 เสมอ ดังนั้นเมืองที่ 1 จะปรากฏเป็นเมืองแรกในวัฏจักรแบบฮามิลตันเสมอ เส้นทางการเดินทางของพนักงานขายสามารถเขียนแทนได้ด้วยเมทริกซ์ $(N-1)\times(N-1)$ จึงต้องใช้จำนวนคิวบิตทั้งหมด $(N-1)^2$ คิวบิต เพื่อแทนกลุ่มของคำตอบทั้งหมด ตัวอย่างเช่น เส้นทางการเดินทาง 1 -> 2 -> 3 -> 4 -> 1 เราสนใจเฉพาะลำดับของเมือง 2 3 และ 4 เนื่องจากเรากำหนดให้พนักงานขายเริ่มเดินทางจากเมือง 1 เท่านั้น ดังนั้นเราจึงสนใจเฉพาะเส้นทาง 2 -> 3 -> 4 โดยลำดับของเมืองจะเปลี่ยนแปลงภายใน 3 เมืองนี้เท่านั้น จากตัวอย่างดังกล่าวสามารถแปลงจากเมทริกซ์ $(N-1)\times(N-1)$ ให้อยู่ในรูปของเวกเตอร์ [1, 0, 0, 0, 1, 0, 0, 0, 1] แต่ละค่าในเมทริกซ์คือตัวแปร $X_{i,p}$ เมื่อ *i* คือ เมือง และ *p* คือลำดับของเมืองในวัฏจักร $X_{i,p}=1$ เมื่อเมือง *i* ถูกผ่านที่ลำดับที่ *p* ในวัฏจักร และ$X_{i,p}=0$ เมื่อเมือง *i* ไม่ถูกผ่านที่ลำดับที่ *p* ในวัฏจักร ดังนั้นจากตัวอย่างรูปแบบเวกเตอร์ดังกล่าวจึงหมายถึง

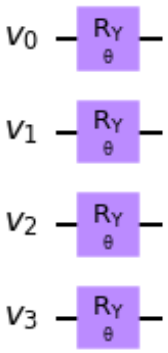
เมือง 2 ลำดับที่ 1 เมือง 3 ลำดับที่ 2 และเมือง 4 ลำดับที่ 3 นั่นเอง เนื่องจากเราต้องการให้แต่ละเมืองปรากฏในวัฏจักรแค่ 1 ครั้ง และในแต่ละลำดับมีเพียงเมืองเดียวที่ปรากฏ

5.2 การกำหนดสถานะเริ่มต้นของคิวบิต

โดยปกติคิวบิตทุกตัวจะเริ่มต้นที่สถานะ $|0\rangle$ เสมอ ซึ่งเราต้องการกำหนดให้เวกเตอร์ความน่าจะเป็นในแต่ละมิติเริ่มต้นที่ 0.5 ในทุกมิติ เพื่อให้ตัวอย่างประชากรที่สุ่มสร้างขึ้นมา จะมีโอกาสเป็นบิต 0 หรือ 1 เท่าๆกัน ดังนั้นในวงจรควอนตัม เราจึงใส่เกต R_y (Rotation around Y-axis) เพื่อกำหนดสถานะเริ่มต้นของคิวบิตให้เป็นสถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ อย่างละเท่าๆกัน เพื่อให้ความน่าจะเป็นที่วัดสถานะของคิวบิตแล้วได้ค่าเป็น 0 หรือ 1 เท่ากัน โดยค่ามุมที่ใช้สำหรับการหมุนคิวบิตรอบแกน Y เพื่อให้คิวบิตอยู่ในสถานะซ้อนทับ สามารถคำนวณได้จากสมการ (30) โดยค่าเริ่มต้นของ probability (p) คือ 0.5 สำหรับทุกมิติในเวกเตอร์ความน่าจะเป็น

$$angle(\theta) = 2 \times \cos^{-1}\sqrt{1 - probability(p)} \tag{30}$$

แผนภาพวงจรควอนตัมสำหรับกำหนดสถานะเริ่มต้นของคิวบิตเป็นดังภาพที่ 34



ภาพที่ 34 แผนภาพวงจรควอนตัมสำหรับกำหนดสถานะเริ่มต้นของคิวบิต

5.3 การดำเนินการกับสถานะของคิวบิตก่อนการวัด

ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับใช้การคัดเลือกโครโมโซมแบบการแข่งขัน (tournament selection) โดยงานวิจัยนี้ใช้ tournament selection เท่ากับ 2 คือ สุ่มสร้างคำตอบขึ้นมา 2 คำตอบ และทำการเปรียบเทียบค่าความเหมาะสมระหว่าง 2 คำตอบ เพื่อหาคำตอบที่ดีกว่า ผู้วิจัยได้ใช้ตัวดำเนินการกับสถานะของคิวบิตก่อนการวัดค่าสถานะของคิวบิตสำหรับคำตอบแรก (First candidtate) และคำตอบที่สอง (Second candidate) แตกต่างกัน First candidate ไม่มีการดำเนินการเพิ่มเติมกับสถานะของคิวบิต นอกเหนือจากการหมุนคิวบิตตามที่ระบุในข้อ 5.2 ส่วน Second candidate ผู้วิจัยได้จำกัดพื้นที่การค้นหา (Search space) เพื่อให้ได้คำตอบสำหรับปัญหา TSP ที่เป็นรูปแบบเส้นทางที่เป็นไปได้ (Feasible path) เท่านั้น โดยใช้ประโยชน์จากอัลกอริทึมควอนตัม คือ อัลกอริทึมการค้นหาของโกรเวอร์ (Grover's search algorithm) ซึ่งขั้นตอนการทำงานของอัลกอริทึมการค้นหาของโกรเวอร์ มีดังนี้

5.3.1. <u>การกำหนดสถานะเริ่มต้นของคิวบิตในอัลกอริทึมการค้นหาของโกรเวอร์</u>

อัลกอริทึมการค้นหาของโกรเวอร์แบบดั้งเดิมจะกำหนดสถานะเริ่มต้นของคิวบิตให้เป็นสถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ อย่างละเท่าๆกันเสมอ แต่เนื่องจากงานวิจัยนี้ได้นำเวกเตอร์ความน่าจะเป็นที่ได้รับการปรับปรุงเมื่อสิ้นสุดแต่ละรอบวิวัฒนาการมาใช้เป็นข้อมูลอ้างอิงสำหรับกำหนดสถานะของคิวบิตในรอบวิวัฒนาการครั้งต่อไป ดังนั้นสถานะเริ่มต้นของคิวบิตในอัลกอริทึมการค้นหาของโกรเวอร์สำหรับงานวิจัยนี้จึงไม่ได้เริ่มต้นที่สถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ อย่างละเท่าๆกันเสมอ โดยขึ้นอยู่กับค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น ณ ขณะนั้น ดังนั้นการกำหนดสถานะเริ่มต้นของคิวบิตในอัลกอริทึมการค้นหาของโกรเวอร์จะเหมือนกันกับข้อ 5.1 คือ ใส่เกต R_y (Rotation around Y-axis) เพื่อกำหนดสถานะเริ่มต้นของคิวบิตให้เป็นสถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ ตามค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ โดยการหมุนคิวบิตไปตามค่ามุมที่คำนวณได้จากสมการ (30) ซึ่งค่าเริ่มต้นของ probability (p) คือ 0.5 สำหรับทุกมิติในเวกเตอร์ความน่าจะเป็น ดังนั้นแผนภาพวงจรควอนตัมสำหรับสถานะเริ่มต้นของคิวบิตในอัลกอริทึมการค้นหาของโกรเวอร์ เป็นดังภาพที่

5.3.2. <u>การสร้างฟังก์ชันโอราเคิล (Oracle) สำหรับปัญหา TSP ในอัลกอริทึมการค้นหาของโกรเวอร์</u>

แนวคิดหลักสำหรับการกำหนดฟังก์ชันโอราเคิลคือการกำหนดฟังก์ชัน $f(x) = 1$ เมื่อ $y(x)$ เจอคำตอบของปัญหาการค้นหาที่สนใจ ในทางกลับกัน $f(x) = 0$ เมื่อ $y(x)$ ไม่เจอคำตอบของปัญหาการค้นหาที่สนใจ ซึ่งปัญหาการค้นหาที่แตกต่างกันก็มีการกำหนดฟังก์ชันโอราเคิลที่แตกต่างกัน ผู้วิจัยได้นำปัญหา TSP ซึ่งเป็นหนึ่งในปัญหายากที่ค่อนข้างได้รับความนิยมจากนักวิจัยในหลากหลายสาขาวิชา มาเป็นปัญหาต้นแบบสำหรับงานวิจัยนี้ โดยปัญหา TSP นี้เป็นปัญหาที่ทำการตัดสินใจหาเส้นทางการเดินทางของพนักงานขาย เมื่อมีเมืองที่ต้องเดินทางไปจำนวน N เมือง การเดินทางจะเดินทางจากเมืองใดเมืองหนึ่งในจำนวน N เมือง โดยจะต้องเดินทางผ่านครบทุกเมืองอย่างละ 1 ครั้ง เดินทางไม่ซ้ำเมืองเดิม และกลับมาที่เมืองแรกที่เริ่มต้นเดินทาง โดยได้ระยะทางรวมที่สั้นที่สุด การกำหนดฟังก์ชันโอราเคิลสำหรับงานวิจัยนี้จะเป็นการจำกัดพื้นที่การค้นหา (Search space) เพื่อให้ได้คำตอบสำหรับปัญหา TSP ที่เป็นรูปแบบเส้นทางที่เป็นไปได้ (Feasible path) เท่านั้น กล่าวคือ มีการเริ่มต้นเดินทางจากเมืองหนึ่ง เดินทางไปจนครบทุกเมืองโดยไม่ซ้ำเมืองเดิม และเดินทางกลับมาที่เมืองเริ่มต้น โดยไม่ได้สนใจว่าเส้นทางที่เดินทางนั้นเป็นเส้นทางที่ระยะทางรวมสั้นที่สุดหรือไม่ คอมพิวเตอร์แบบดั้งเดิมสามารถหาคำตอบของปัญหา TSP ได้โดยการใช้วิธี Brute-force ของทุกๆเส้นทางทั้งที่เป็น Feasible path และไม่ใช่ Feasuble path และทำการเปรียบเทียบไปเรื่อยๆจนกว่าจะได้ระยะทางที่สั้นที่สุดที่เป็น Feasible path จากเส้นทางทั้งหมด ดังนั้นถ้ามี N เมือง จะมีรูปแบบการเดินทางทั้งหมดคือ 2^N รูปแบบ ซึ่งรวมถึงรูปแบบที่ไม่ใช่ Feasuble path ด้วย จะเห็นได้ว่าคอมพิวเตอร์แบบดั้งเดิมต้องใช้เวลาในการหาคำตอบเป็นเอ็กโปเน็นเชียล และเวลาในการหาคำตอบก็เพิ่มขึ้นเป็นเอ็กโปเน็นเชียลตามจำนวนเมืองที่เพิ่มขึ้น นอกเหนือจากวิธี Brute-force การใช้วิธีการเขียนโปรแกรมแบบไดนามิกก็ช่วยลดเวลาในการหาคำตอบลงได้ แต่ก็ยังคงใช้เวลาเป็นเอ็กโปเน็นเชียลอยู่ เช่น การกำหนดให้เส้นทางที่เป็นไปได้ทั้งหมดเป็น Feasible path เท่านั้น จะมีรูปแบบเส้นทางทั้งหมด $N!$ รูปแบบ แต่ถ้าเรากำหนดให้พนักงานขายเริ่มต้นเดินทางที่เมืองใดเมืองหนึ่งเสมอ รูปแบบเส้นทางทั้งหมดที่ต้องค้นหาจะเหลือ $(N - 1)!$ รูปแบบ การกำหนดเมืองเริ่มต้นสามารถช่วยลดพื้นที่การค้นหาคำตอบ แต่ก็ยังคงใช้เวลาในการค้นหาคำตอบมากอยู่ อย่างไร

ก็ตามด้วยคุณสมบัติ quantum superposition ทำให้สามารถประมวลผลทุกสถานะควอนตัมได้พร้อมกัน (quantum parallelism) ทำให้ประมวลผลได้อย่างรวดเร็วกว่าคอมพิวเตอร์ดั้งเดิมมาก

อัลกอริทึมการค้นหาของโกรเวอร์เป็นอัลกอริทึมควอนตัมสำหรับการค้นหาข้อมูลที่ไม่เรียงลำดับ อัลกอริทึมดังกล่าวได้นำคุณสมบัติทางกลศาสตร์ควอนตัมของคิวบิต นั่นก็คือสภาวะซ้อนทับของคิวบิต รวมกับเทคนิคการขยายแอมพลิจูด (Amplitude amplification) ที่ถูกนำเสนอในอัลกอริทึมนี้ มาช่วยเร่งความเร็วในการค้นหาคำตอบด้วยความเร็วกำลังสอง ดังนั้นเราสามารถค้นหาคำตอบที่ต้องการด้วยจำนวนครั้งโดยประมาณ $\sqrt{N}$ เมื่อมีคำตอบขนาดใหญ่ *N* คำตอบ ซึ่งในคำตอบเหล่านี้มี 1 คำตอบที่มีคุณสมบัติที่เป็นเอกลักษณ์ที่เราต้องการค้นหา ซึ่งทำให้ประหยัดเวลาเป็นอย่างมากสำหรับการค้นหาคำตอบที่ต้องการจากจำนวนคำตอบทั้งหมดที่มีขนาดใหญ่มาก

การกำหนดฟังก์ชันโอราเคิลสำหรับปัญหา TSP เพื่อจำกัดพื้นที่การค้นหาคำตอบเฉพาะรูปแบบเส้นทางที่เป็น Feasible path นั้น จากข้อ 5.1 ผู้วิจัยได้แปลงปัญหา TSP ให้อยู่ในรูปของปัญหาวัฏจักรแบบฮามิลตัน (Hamiltonian cycles problem) เพื่อลดรูปปัญหาให้อยู่ในรูปแบบการตัดสินใจของแบบจำลอง Ising (Ising model) โดยใช้สปินทั้งหมด $(N-1)^2$ สปิน เมื่อ *N* คือจำนวนเมือง และผู้วิจัยกำหนดให้พนักงานขายเริ่มต้นเดินทางจากเมืองที่ 1 เสมอ ดังนั้นต้องใช้จำนวนคิวบิตทั้งหมด $(N-1)^2$ คิวบิต เพื่อแทนกลุ่มของคำตอบทั้งหมด แต่ละค่าในเมทริกซ์คือตัวแปร $X_{i,p}$ เมื่อ *i* คือ เมือง และ *p* คือ ลำดับของเมืองในวัฏจักร $X_{i,p}=1$ เมื่อเมือง *i* ถูกผ่านที่ลำดับที่ *p* ในวัฏจักร และ$X_{i,p}=0$ เมื่อเมือง *i* ไม่ถูกผ่านที่ลำดับที่ *p* ในวัฏจักร เนื่องจากเราต้องการให้แต่ละเมืองปรากฏในวัฏจักรแค่ 1 ครั้ง ยกเว้นเมืองเริ่มต้น และในแต่ละลำดับมีเพียงเมืองเดียวที่ปรากฏ ดังนั้นจึงมีการกำหนด 2 ข้อจำกัด เพื่อให้เงื่อนไขดังกล่าวเป็นจริง ดังสมการ (31)

$$\sum_i x_{i,p}=1 \quad \forall p, \quad \sum_p x_{i,p}=1 \quad \forall i \tag{31}$$

สำหรับลำดับของเมืองแต่ละเมืองในวัฏจักรแบบฮามิลตัน ถ้า $X_{i,p}$ และ $X_{j,p+1}$ เป็น 1 ทั้งคู่ แต่ไม่มีเส้นทางเชื่อมระหว่างเมือง *i* ไปเมือง *j* จำเป็นต้องกำหนดบทลงโทษ (Energy penalty) อย่างไรก็ตาม สำหรับงานวิจัยนี้ถือว่าทุกเมืองมีเส้นทางเชื่อมถึงกัน (Fully connected) ดังนั้นจึงไม่ได้เพิ่มบทลงโทษสำหรับกรณีดังกล่าวสำหรับการคำนวณค่าพลังงานของแบบจำลอง Ising

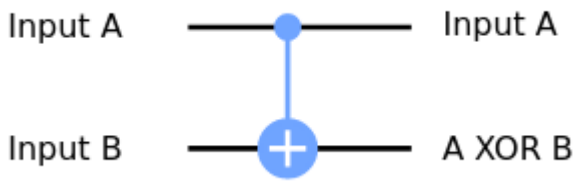
การกำหนดค่าพลังงานของแบบจำลอง Ising เพื่อให้ได้ค่าพลังงานต่ำสุดซึ่งแทนคำตอบที่เหมาะสมที่สุดของปัญหา TSP ต้องพิจารณาเส้นทางที่ทำให้ได้ระยะทางที่สั้นที่สุด โดยที่เส้นทางดังกล่าวเป็น Feasible path ตามเงื่อนไขของปัญหา TSP ดังนั้นสมการคำนวณ ค่าพลังงานของแบบจำลอง Ising ที่แทนคำตอบของปัญหา TSP มีดังนี้

$$Energy = \sum_{i,j} w_{i,j} \sum_p x_{i,p} x_{j,p+1} + B \sum_p \left(1 - \sum_i x_{i,p}\right)^2 + B \sum_i \left(1 - \sum_p x_{i,p}\right)^2 \quad (32)$$

เมื่อ *B* คือค่าน้ำหนักของการลงโทษ ซึ่งต้องมีขนาดมากพอที่จะทำให้เงื่อนไขของ Feasible path ของปัญหา TSP เป็นจริง สำหรับค่าในวงเล็บที่มีการยกกำลังสองเพื่อให้ค่าต่ำสุดของส่วนการลงโทษ (Penalty term) มีค่าเป็น 0 ในกรณีที่เงื่อนไข Feasible path เป็นจริง ดังนั้นค่าความเหมาะสมของคำตอบของปัญหา TSP สามารถหาได้จากค่าพลังงานของแบบจำลอง Ising จากสมการ (32) โดยการหาค่าพลังงานที่ต่ำที่สุดเพื่อให้ได้คำตอบที่เหมาะสมที่สุด ซึ่งก็คือเส้นทางที่เป็น Feasible path และเป็นระยะทางที่สั้นที่สุด วิธีการแปลงปัญหา TSP เป็นแบบจำลอง Ising ดังกล่าวข้างต้น ทำให้ผู้วิจัยสามารถสร้างวงจรควอนตัมได้โดยง่าย[43]

การตรวจสอบสถานะของคิวบิตที่ให้คำตอบที่เป็น Feasible path มีแนวคิดมาจากฟังก์ชันที่ทำงานบนคอมพิวเตอร์แบบดั้งเดิม และแปลงฟังก์ชันดังกล่าวให้สามารถทำงานบนเครื่องคอมพิวเตอร์เชิงควอนตัมได้ ฟังก์ชันนี้จะตรวจสอบในแต่ละคอลัมน์และในแต่ละแถวว่ามีค่า 1 ปรากฏเพียงที่เดียวหรือไม่ โดยลำดับของคอลัมน์แทนลำดับเมือง และลำดับของแถวแทนลำดับการเดินทางไปยังเมืองนั้น เนื่องจากเราต้องการให้แต่ละเมืองปรากฏในวัฏ

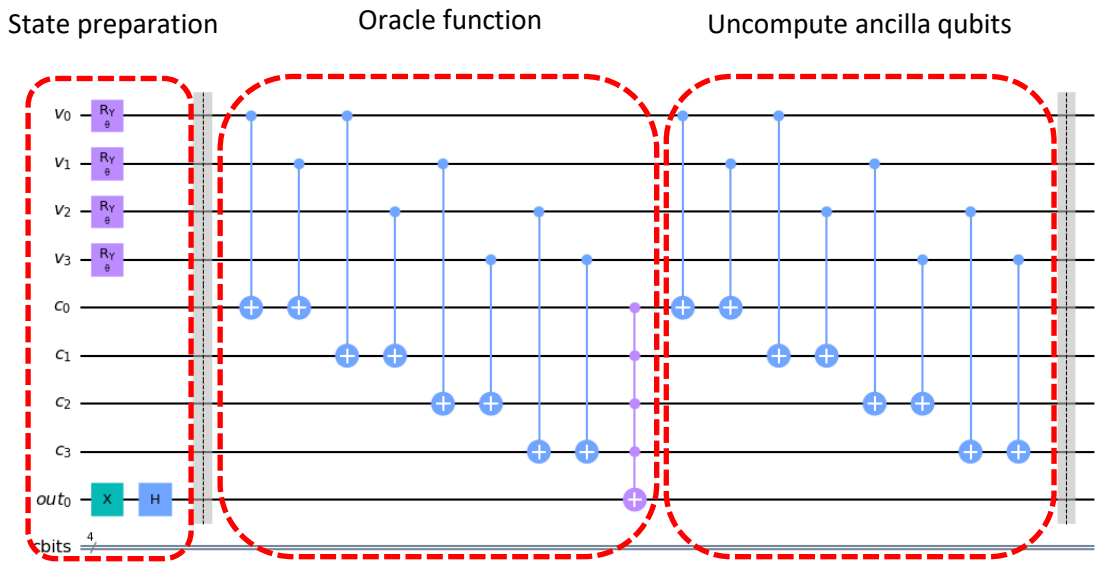
จักรแค่ 1 ครั้ง และในแต่ละลำดับมีเพียงเมืองเดียวที่ปรากฏ ส่วนเมืองแรกที่เริ่มต้นเดินทางจะไม่ได้นำมาคิด เนื่องจากมีการกำหนดให้เป็นเมืองที่ 1 เสมอ ดังนั้นในแบบจำลอง Ising จะเริ่มเช็คตั้งแต่เมืองที่ 2 ไปจนถึงเมืองที่ *N* ผู้วิจัยได้สร้างรายการที่ต้องเปรียบเทียบทั้งหมดลงในรายการที่ต้องตรวจสอบว่าเป็น Feasible path หรือไม่ และตรวจสอบ Feasible path โดยใช้เกต XOR เราสามารถสร้างวงจรควอนตัมที่มีการทำงานคล้ายกันกับ XOR gate ของคอมพิวเตอร์แบบดั้งเดิมโดยใช้เกต controlled-NOT หรือ CNOT แผนผังวงจรควอนตัมสำหรับเกต CNOT เป็นดังภาพที่ 35



ภาพที่ 35 แผนผังวงจรควอนตัมแสดงเกต CNOT

เกต CNOT นำมาใช้กับคิวบิต 2 คิวบิต โดยจุดกลมของเกต CONT อยู่ตรงกับคิวบิตใด คิวบิตนั้นจะทำงานเป็นคิวบิตควบคุม (Control qubit) และวงกลมที่มีเครื่องหมายบวกตรงกลางจะทำงานเป็นคิวบิตเป้าหมาย (Target qubit) โดยค่าของคิวบิตเป้าหมายจะเปลี่ยนแปลง เมื่อค่าของคิวบิตควบคุมเป็น 1 ดังนั้นจากภาพที่ 35 ถ้า Input A และ Input B ค่าเหมือนกัน A XOR B จะมีค่าเป็น 0 ในทางกลับกัน ถ้าค่าต่างกัน A XOR B จะมีค่าเป็น 1 เราจะใช้ คุณสมบัติของเกต CNOT ในการตรวจสอบทุกเงื่อนไขที่ต้องเปรียบเทียบ เช่น กรณี 3 เมือง ตัวแปร $X_{i,p}$ และ $X_{j,p+1}$ ที่ต้องตรวจสอบมีทั้งหมด 4 คู่ ได้แก่ $(X_{2,2}, X_{2,3}), (X_{2,2}, X_{3,2}), (X_{2,3}, X_{3,3}), (X_{3,2}, X_{3,3})$ เพื่อตรวจสอบว่าเป็น Feasible path หรือไม่ โดยใช้เกต CNOT ดังภาพที่ 35 และทำซ้ำจนครบทุกคู่ ผลลัพธ์ของการ XOR ในแต่ละคู่นั้นจะเป็นคิวบิตควบคุมของคิวบิตเป้าหมายสุดท้ายที่เป็นคิวบิตผลลัพธ์ (Output qubit) โดยใช้เกต Multi-controlled ซึ่งคิวบิตผลลัพธ์จะถูกกำหนดค่าเริ่มต้นให้อยู่ในสถานะ $|-\rangle$ ถ้าทุกคิวบิตควบคุมเป็น 1 จะทำให้คิวบิตเป้าหมายถูกเปลี่ยนค่าจาก 0 เป็น 1 ซึ่งทำให้เฟสของคิวบิตเป้าหมายที่เป็นเฟสติดลบ (Negative phase) มีผลต่อการกลับเฟส (Phase

kickback) ในอัลกอริทึมค้นหาของโกรเวอร์โดยทันที วงจรส่วนฟังก์ชันโอราเคิลแสดงดังภาพที่ 36 ทำให้การทำ Grover iteration ในท้ายที่สุดจะไปขยายสัญญาณแอมพลิจูดของสถานะที่ต้องการ หรือสถานะที่ให้คำตอบที่เป็น Feasible path ดังจะได้อธิบายในหัวข้อถัดไป ซึ่งจะช่วยเพิ่มความน่าจะเป็นที่เมื่อวัดสถานะของคิวบิตแล้วได้คำตอบที่เป็น Feasible path นั่นเอง



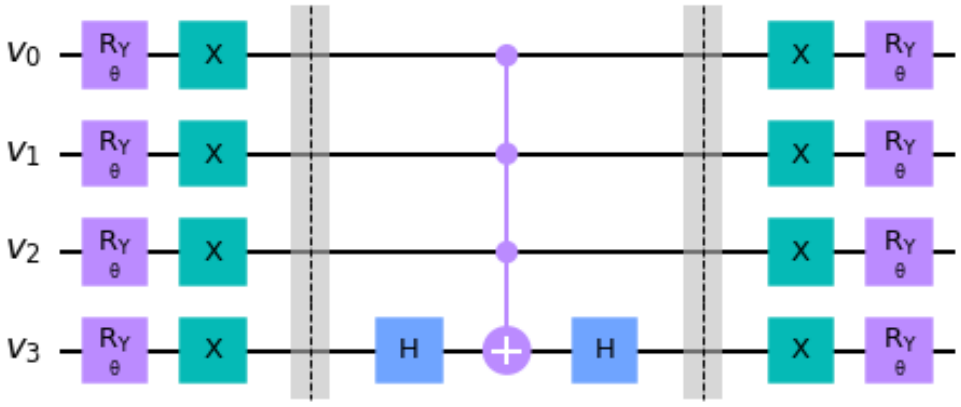
ภาพที่ 36 วงจรควอนตัมแสดงฟังก์ชันโอราเคิลสำหรับตรวจสอบสถานะควอนตัมที่แทนรูปแบบคำตอบที่เป็น feasible path สำหรับปัญหา TSP ขนาด 3 เมือง

5.3.3. <u>การขยายค่าแอมพลิจูดของสถานะที่สนใจในอัลกอริทึมการค้นหาของโกรเวอร์</u>

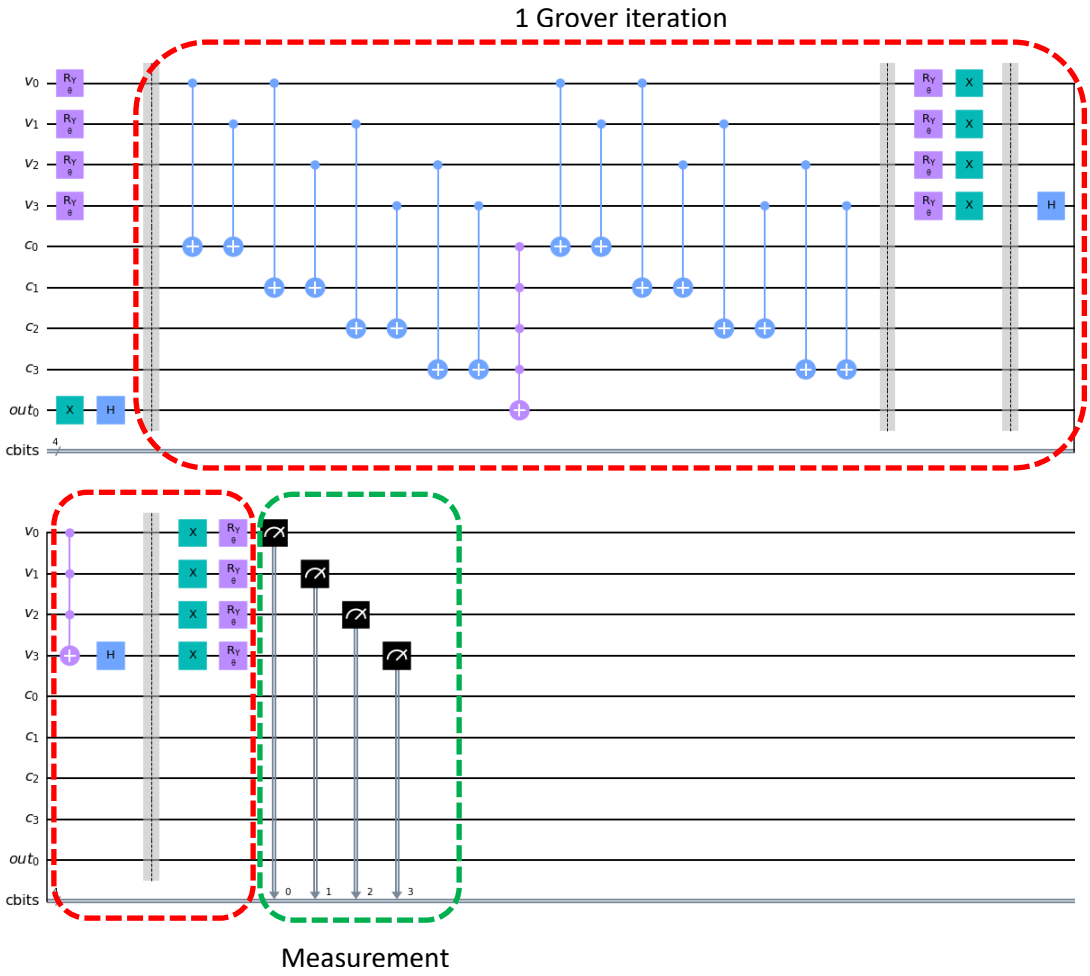
จากข้อ 5.3.1 สถานะเริ่มต้นของคิวบิตในอัลกอริทึมการค้นหาของโกรเวอร์สำหรับงานวิจัยนี้ไม่ได้เริ่มต้นที่สถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ อย่างละเท่าๆกันเสมอ โดยขึ้นอยู่กับค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น ณ ขณะนั้น เนื่องจากผู้วิจัยได้นำเวกเตอร์ความน่าจะเป็นที่ได้รับการปรับปรุงเมื่อสิ้นสุดแต่ละรอบวิวัฒนาการ มาใช้เป็นข้อมูลอ้างอิงสำหรับกำหนดสถานะของคิวบิตในรอบวิวัฒนาการครั้งต่อไป เพื่อให้ความน่าจะเป็นที่วัดคิวบิตแล้วได้สถานะที่ต้องการมีมากขึ้น ดังนั้นส่วนของการดำเนินการ Diffusion operator เพื่อสะท้อนสถานะของคิวบิตผลลัพธ์กับสถานะคิวบิตเริ่มต้น โดยใช้หลักการทางเรขาคณิตเรื่องการสะท้อน 2 ครั้งสำหรับการหมุนในระนาบสอง

มิติ ซึ่ง Diffusion operator ของอัลกอริทึมการค้นหาของโกรเวอร์แบบดั้งเดิมนั้น คือการสะท้อนกับแกนที่เป็นเวกเตอร์สถานะเริ่มต้นของคิวบิตที่เป็นสถานะซ้อนทับระหว่างสถานะ $|0\rangle$ และ สถานะ $|1\rangle$ อย่างละเท่าๆกัน แต่เวกเตอร์สถานะเริ่มต้นของคิวบิตสำหรับงานวิจัยนี้จะเปลี่ยนแปลงไปตามค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น ทำให้ต้องปรับเปลี่ยนวงจรควอนตัมส่วนของ Diffusion operator ในอัลกอริทึมการค้นหาของโกรเวอร์เล็กน้อย เพื่อให้สะท้อนเวกเตอร์สถานะเริ่มต้นของคิวบิตได้อย่างถูกต้อง[44] ดังนั้นวงจรควอนตัมส่วนของ Diffusion operator สำหรับงานวิจัยนี้เป็นดังภาพที่ 37

การทำซ้ำของ Grover (Grover iteration) คือขั้นตอนสำคัญที่ใช้สำหรับขยายแอมพลิจูดของสถานะที่สนใจ ซึ่งประกอบด้วย 2 ขั้นตอน ได้ Oracle reflection และ Diffuser 2 ขั้นตอนดังกล่าวสามารถถูกวนซ้ำได้เป็นจำนวนครั้งสูงสุดคือ $\frac{\pi}{4}\sqrt{\frac{N}{T}}$ ครั้ง เมื่อ N คือจำนวนสถานะของคิวบิตทั้งหมด และ T คือจำนวนคำตอบที่เป็นผลลัพธ์ของฟังก์ชันโอราเคิล เพื่อขยายค่าแอมพลิจูดของสถานะที่สนใจให้มีค่าสูงสุด[41] ซึ่งจะทำให้ค่าความน่าจะเป็นที่เมื่อวัดสถานะคิวบิตแล้วได้สถานะที่สนใจสูงสุดตามไปด้วย ดังนั้นวงจรควอนตัมแสดงส่วนการทำงานของอัลกอริทึมการค้นหาของโกรเวอร์ทั้งหมดแสดงดังภาพที่ 38



ภาพที่ 37 ตัวอย่างวงจรควอนตัมส่วนของ Diffusion operator ในอัลกอริทึมการค้นหาของโกรเวอร์ สำหรับปัญหา TSP ขนาด 3 เมือง



ภาพที่ 38 วงจรควอนตัมแสดงส่วนการทำงานของอัลกอริทึมการค้นหาของโกรเวอร์ สำหรับปัญหา TSP ขนาด 3 เมือง

5.4 การวัดสถานะของคิวบิต

การวัดสถานะคิวบิตทำให้สถานะซ้อนทับ (Superposition) ของคิวบิตแต่ละตัวถูกยุบเหลือเพียงสถานะ 0 หรือ 1 เท่านั้น โดย First candidate จะมีค่าสถานะคิวบิตสอดคล้องตามค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น ณ ขณะนั้น ในขณะที่ Second candidate จะมีค่าสถานะคิวบิตสอดคล้องตามค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น และนำอัลกอริทึมควอนตัม ได้แก่ อัลกอริทึมการค้นหาของโกรเวอร์ มาช่วยในการจำกัดพื้นที่การค้นหาเฉพาะเส้นทางที่เป็น Feasible path และช่วยลดระยะเวลาในการค้นหาเป็นกำลังสองของอัลกอริทึมแบบดั้งเดิมทั่วไป

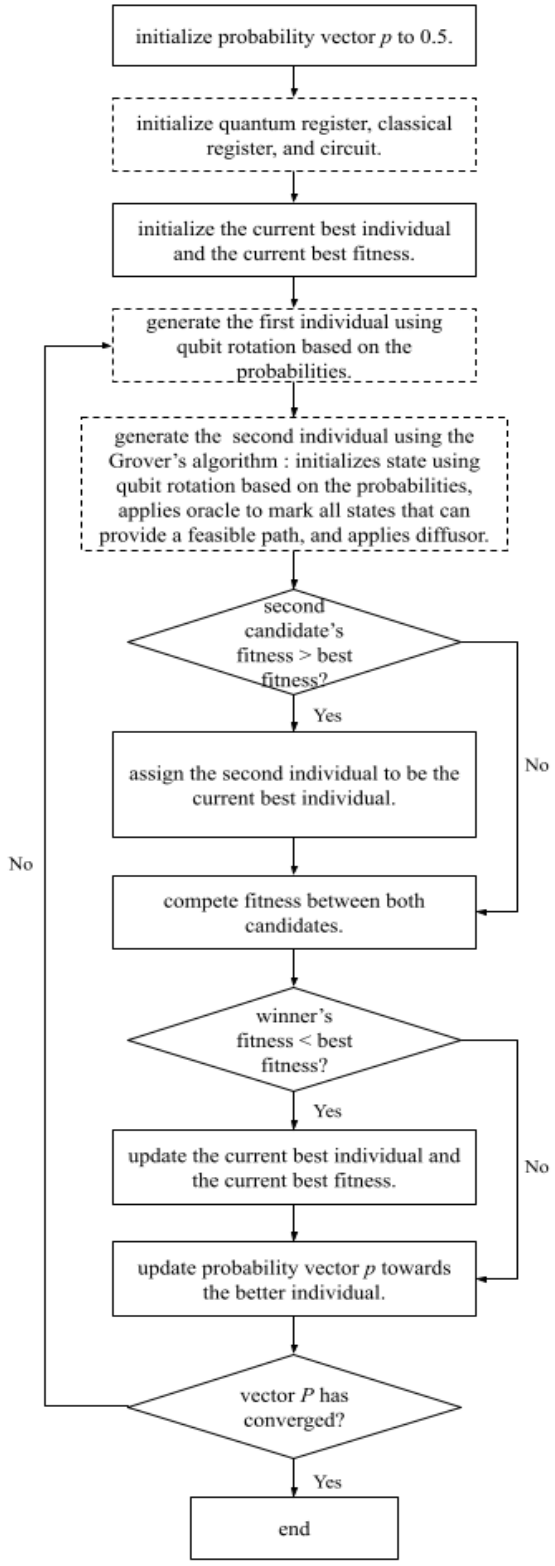
5.5 การดำเนินการของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม

ในขั้นแรกจะมีการสุ่มคำตอบขึ้นมา 1 คำตอบ และเก็บเป็นรูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบันหลังจากนั้นจึงเริ่มกระบวนการตามข้อ 5.2 – 5.4 เมื่อได้รูปแบบคำตอบ 2 คำตอบจากข้อ 5.4 แล้ว ต้องทำการประเมินค่าความเหมาะสมของรูปแบบคำตอบทั้ง 2 คำตอบตามฟังก์ชันจุดประสงค์ที่ระบุในสมการ (32) เฉพาะคำตอบที่ 2 ที่จะมีเงื่อนไขเพิ่มเติม คือ นำมาเปรียบเทียบกับค่าความเหมาะสมของคำตอบที่ดีที่สุด ณ ปัจจุบัน และค่อยเลือกคำตอบที่ดีกว่ามาเปรียบเทียบกับคำตอบแรกเพื่อหาคำตอบที่ดีกว่าอีกครั้ง และใช้คำตอบที่ดีกว่ามาเป็นต้นแบบในการปรับปรุงค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็น วิธีการดังกล่าวจะใกล้เคียงกับ compact genetic algorithm with an elite[12] ซึ่งเป็นการปรับปรุงขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ โดยการเลือกคำตอบจากกลุ่มคำตอบที่ดีที่สุดที่เคยบันทึกไว้ มาใช้ในการเปรียบเทียบกัน ก่อนจะหาคำตอบที่ดีกว่าเพื่อใช้เป็นต้นแบบในการปรับปรุงค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็น ขั้นตอนดังกล่าวจะทำงานบนเครื่องคอมพิวเตอร์แบบดั้งเดิม เนื่องด้วยข้อจำกัดในการเปรียบเทียบค่าในเครื่องคอมพิวเตอร์เชิงควอนตัมจำเป็นต้องสร้างวงจรเปรียบเทียบเชิงควอนตัม (Quantum comparator) มาใช้ในการเปรียบเทียบค่า ผู้วิจัยจึงทำการประเมินค่าความเหมาะสม และเปรียบเทียบค่าความเหมาะสมบนเครื่องคอมพิวเตอร์แบบดั้งเดิมแทน จากนั้นจึงทำการปรับปรุงค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นตามคำตอบที่ดีกว่า โดยทำการเปรียบเทียบบิตของผู้ชนะและผู้แพ้ทีละบิต ถ้าบิตของผู้ชนะ ไม่เหมือนกันกับบิตของผู้แพ้ และบิตของผู้ชนะเท่ากับ 1 จะทำการปรับเพิ่มความน่าจะเป็นด้วยอัตราส่วน 1 : ขนาดของประชากร ในทางกลับกัน ถ้าบิตของผู้ชนะเท่ากับ 0 จะทำการปรับลดความน่าจะเป็นด้วยอัตราส่วน 1 : ขนาดของประชากร เช่นกัน จากนั้นจึงตรวจสอบว่าคำตอบที่เป็นผู้ชนะดีกว่าคำตอบที่ดีที่สุด ณ ปัจจุบันหรือไม่ ถ้าใช่จึงค่อยทำการปรับปรุงรูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบันตามรูปแบบคำตอบของผู้ชนะสุดท้ายตรวจสอบว่าค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นนั้นลู่เข้าแล้วหรือยัง ซึ่งเป็นเงื่อนไขสำหรับตรวจสอบการจบขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ ภาพที่ 39 แสดงโค้ดเทียมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่นำเสนอในงานวิจัยนี้ ภาพที่ 40 แสดงแผนผังขั้นตอนการทำงานของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมโดยกรอบเส้นประคือประมวลผลบนเครื่อง IBM QASM simulator และกรอบเส้นทึบคือประมวลผลบนเครื่องคอมพิวเตอร์ดั้งเดิม

Algorithm 2: Grover-assisted cGA*

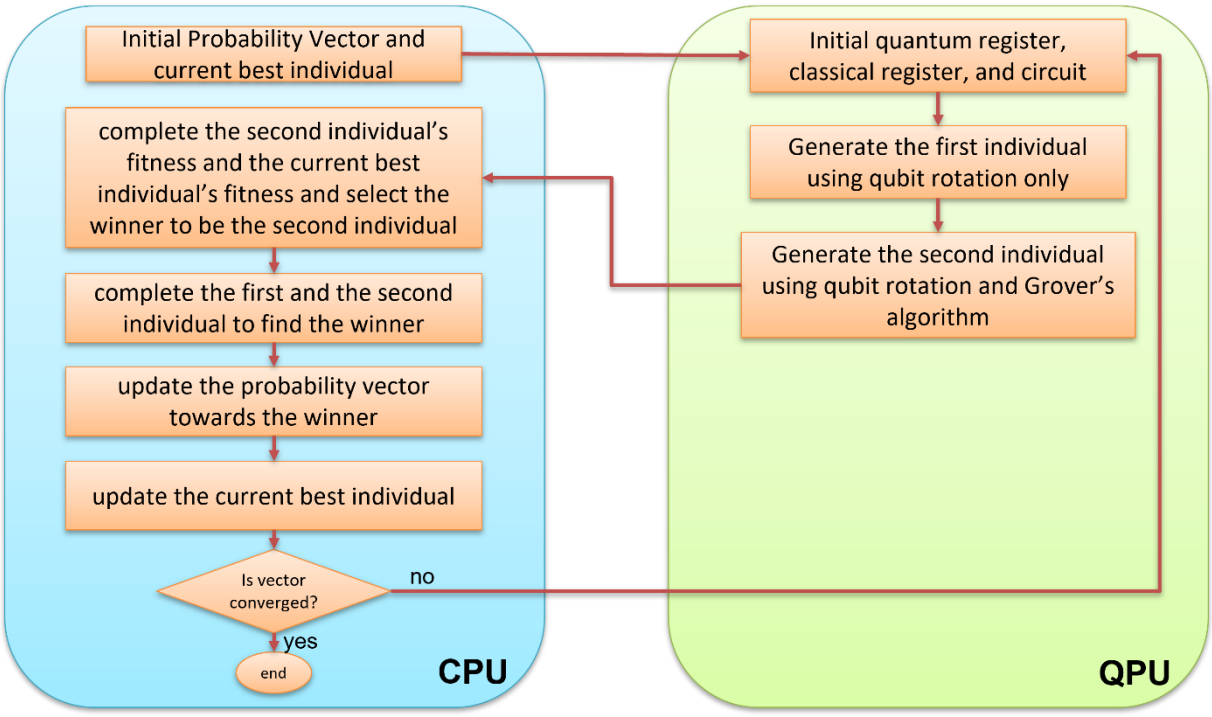
1) initialize probability vector:
for $i \leftarrow 1$ *to* l **do**
  $p[i] \leftarrow 0.5$;
end
2) initialize the current best individual and Grover iteraton:
$curBestIdv \leftarrow 000..00$, $t \leftarrow numGroverIteration$;
3) initialize quantum register, classical register, circuit:
$circuit \leftarrow QuantumCircuit(qr, cr)$;
4) generate the first individual using qubit rotation based on the probabilities:
$a \leftarrow generateFirstIdv(p)$;
5) generate the second individual using the adjusted Grover's algorithm with oracle of the objective function:
$circuit \leftarrow initialize(p)$;
for $i \leftarrow 1$ *to* t **do**
  $circuit \leftarrow feasibleSolution(circuit)$;
  $circuit \leftarrow inverseFeasibleSol(circuit)$;
  $circuit \leftarrow diffuser(circuit)$;
end
$b \leftarrow measure(circuit)$;
6) evaluate the second individual's fitness:
if $curBestIdv.fitness > b.fitness$ **then**
  $b \leftarrow curBestIdv$;
end
7) let them compete:
$winner, loser \leftarrow compete(a, b)$;
8) update probability vector towards winner:
for $i \leftarrow 1$ *to* l **do**
  if $winner[i] \neq loser[i]$ **then**
    if $winner[i] = 1$ **then**
      $p[i] \leftarrow p[i] + 1/n$;
    else
      $p[i] \leftarrow p[i] - 1/n$;
    end
  end
end
9) update the current best individual:
if $curBestIdv.fitness < winner.fitness$ **then**
  $curBestIdv \leftarrow winner$;
end
10) check if the vector has converged:
for $i \leftarrow 1$ *to* l **do**
  if $p[i] > 0$ *and* $p[i] < 1$ **then**
    return to step 3;
  end
end
l is a chromosome length, n is a population size, and t is a number of Grover iterations.

ภาพที่ 39 โค้ดเทียมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA)*



ภาพที่ 40 แผนผังแสดงขั้นตอนการทำงานของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม กรอบเส้นประคือประมวลผลบนเครื่อง IBM QASM simulator

Application architecture ของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมแสดงดังภาพที่ 41 โดยแสดงการเชื่อมโยงการทำงานระหว่างการประมวลผลบนเครื่องคอมพิวเตอร์ดั้งเดิมและการประมวลผลบนเครื่องคอมพิวเตอร์เชิงควอนตัมของขั้นตอนวิธีที่นำเสนอ งานวิจัยนี้ได้ประยุกต์การประมวลผลเชิงควอนตัมและการหมุนคิวบิตเพื่อกำหนดสถานะเริ่มต้นของคิวบิตที่แทนรูปแบบคำตอบของปัญหาที่สนใจ โดยค่ามุมที่ใช้ในการหมุนคิวบิตขึ้นอยู่กับค่าความน่าจะเป็นของแต่ละ element ในเวกเตอร์ความน่าจะเป็นที่ถูกกำหนดจากคอมพิวเตอร์ดั้งเดิม ก่อนจะทำการวัดค่าสถานะคิวบิตเพื่อสร้างโครโมโซมตัวแรก นอกจากนี้ได้นำประโยชน์ของการประมวลผลเชิงควอนตัมแบบขนาน (Quantum parallelism) จากอัลกอริทึมการค้นหาของโกรเวอร์ ซึ่งเป็นอัลกอริทึมที่ทำงานบนพื้นฐานกลศาสตร์ควอนตัมมาใช้ในขั้นตอนการสร้างโครโมโซมตัวที่สองของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม เพื่อให้ได้โครโมโซมหรือรูปแบบคำตอบของปัญหาที่ดี และช่วยเพิ่มประสิทธิภาพในการค้นหาคำตอบที่เหมาะสมมากขึ้น ในขณะที่การประเมินค่าความเหมาะสมของรูปแบบคำตอบ และการปรับปรุงเวกเตอร์ความน่าจะเป็นตามรูปแบบคำตอบที่ดียังคงทำงานอยู่บนเครื่องคอมพิวเตอร์ดั้งเดิม



ภาพที่ 41 Application architecture ของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม

บทที่ 6
ผลการวิจัยเบื้องต้นสำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม

6.1 การจัดเตรียมสภาพแวดล้อมสำหรับการทดลองเปรียบเทียบประสิทธิภาพระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับชนิดควอนตัม

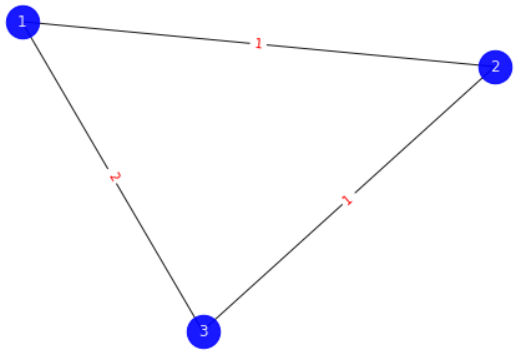
เนื่องจากขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่นำเสนอในงานวิจัยนี้ ได้มีการปรับปรุงขั้นตอนการคัดเลือก (Selection) คำตอบเพื่อนำมาใช้ในการเปรียบเทียบ โดยมีการเปรียบเทียบ Second candidate กับรูปแบบคำตอบที่ดีที่สุด ณ เวลานั้น จากนั้นจึงเลือกคำตอบที่ดีกว่ามาใช้สำหรับเปรียบทียบกับ First Candidate ต่อไป ผู้วิจัยจึงได้ทำการปรับปรุงขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมให้มีขั้นตอนการคัดเลือกคำตอบที่นำมาใช้ในการเปรียบเทียบ (Compact genetic algorithm with an elite) แบบเดียวกัน เพื่อให้สามารถเปรียบเทียบประสิทธิภาพการทำงานระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับชนิดควอนตัมได้ ดังนั้นต่อจากนี้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมในงานวิจัยนี้หมายถึง “cGA*” โดยภาพรวมการทำงานของ cGA* แสดงดังภาพที่ 42 ส่วนขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมในงานวิจัยนี้เรียกว่า “Grover-assisted cGA*” อย่างไรก็ตามการเข้ารหัสปัญหา TSP โดยรูปแบบการเข้ารหัสของโครโมโซมเป็นเลขฐานสอง (Binary bit string) ในคอมพิวเตอร์แบบดั้งเดิมนั้นจะใช้วิธีการแตกต่างกับในคอมพิวเตอร์เชิงควอนตัม เนื่องจากเราสามารถปรับปรุงการเข้ารหัสบนคอมพิวเตอร์แบบดั้งเดิมเพื่อให้ใช้จำนวนบิตน้อยที่สุดได้ โดยไม่ต้องสนใจว่าวงจรคลาสสิกจะมีหน้าตาเป็นอย่างไร ซึ่งแตกต่างกันกับการเข้ารหัสของปัญหา TSP บนวงจรควอนตัม

ปัญหา TSP ที่ใช้เป็นต้นแบบสำหรับงานวิจัยนี้เป็นปัญหา TSP ขนาดเล็กคือ 3 เมือง และ 4 เมือง เนื่องจากจำนวนคิวบิตที่คอมพิวเตอร์เชิงควอนตัมจำลองของไอบีเอ็มมีให้ใช้ค่อนข้างจำกัด จึงสามารถทดลองได้กับปัญหา TSP ขนาดเล็กเท่านั้น ภาพที่ 43 และ 44 แสดงปัญหา TSP 3 เมือง และ 4 เมือง ที่ใช้สำหรับงานวิจัยนี้ตามลำดับ

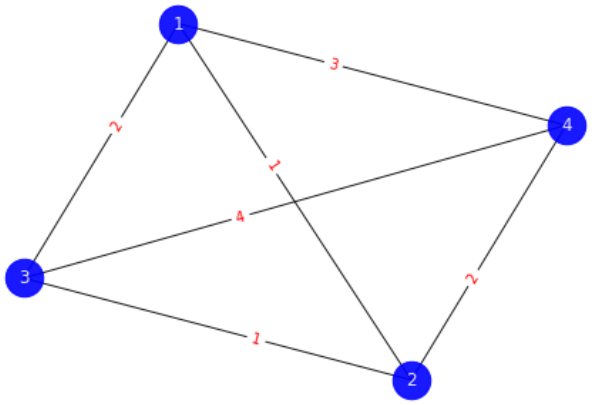
Algorithm 1: The cGA*

1) initialize probability vector:
for $i \leftarrow 1$ *to* l **do**
  $p[i] \leftarrow 0.5$;
end
2) initialize the current best individual:
$curBestIdv \leftarrow 000..00$;
3) generate two individuals from the vector:
$a \leftarrow generate(p)$ $b \leftarrow generate(p)$;
4) evaluate the second individual's fitness:
if $curBestIdv.fitness > b.fitness$ **then**
  $b \leftarrow curBestIdv$;
end
5) let them compete:
$winner, loser \leftarrow compete(a, b)$;
6) update probability vector towards winner:
for $i \leftarrow 1$ *to* l **do**
  if $winner[i] \neq loser[i]$ **then**
    if $winner[i] = 1$ **then**
      $p[i] \leftarrow p[i] + 1/n$;
    else
      $p[i] \leftarrow p[i] - 1/n$;
    end
  end
end
7) update the current best individual:
if $curBestIdv.fitness < winner.fitness$ **then**
  $curBestIdv \leftarrow winner$;
end
8) check if the vector has converged:
for $i \leftarrow 1$ *to* l **do**
  if $p[i] > 0$ *and* $p[i] < 1$ **then**
    return to step 3;
  end
end
l is a chromosome length and n is a population size.

ภาพที่ 42 โค้ดเทียมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกคำตอบ (cGA)*



ภาพที่ 43 ปัญหา TSP ขนาด 3 เมือง ที่ใช้เป็นต้นแบบในการทดลองสำหรับงานวิจัยนี้



ภาพที่ 44 ปัญหา TSP ขนาด 4 เมือง ที่ใช้เป็นต้นแบบในการทดลองสำหรับงานวิจัยนี้

6.1.1. <u>การเข้ารหัสปัญหา TSP บนเครื่องคอมพิวเตอร์แบบดั้งเดิม</u>

ค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็นแทนความน่าจะเป็นของแต่ละเส้นเชื่อมระหว่าง 2 เมือง เช่น 4 เมือง ได้แก่ A B C D จะมีเส้นเชื่อมระหว่างเมืองที่เป็นไปได้ทั้งหมด 6 เส้นเชื่อม ได้แก่ A->B, A->C, A->D, B->C, B->D, C->D ดังนั้นจำนวนบิตที่จำเป็นต้องใช้คือจำนวนเส้นเชื่อมทั้งหมดที่เป็นไปได้ เมื่อ N คือจำนวนเมือง สามารถเขียนสมการแสดงจำนวนเส้นเชื่อมทั้งหมดที่เป็นไปได้ ดังนี้

$$\text{จำนวนเส้นเชื่อมทั้งหมดที่เป็นไปได้} = \frac{(N-1)}{2} \times N \qquad (33)$$

ค่าความน่าจะเป็นของแต่ละมิติในเวกเตอร์ความน่าจะเป็น $\left(P_{i,j}\right)$ แทนความน่าจะเป็นของเส้นเชื่อมระหว่างเมือง *i* กับเมือง *j*

6.1.2. <u>สภาพแวดล้อมและค่าพารามิเตอร์ต่างๆที่ใช้สำหรับงานวิจัย</u>

ผู้วิจัยได้กำหนดขนาดของประชากร (Population size) สูงสุดสำหรับปัญหา TSP 3 เมือง คือ 50 และขนาดของประชากรสูงสุดสำหรับปัญหา TSP 4 เมือง คือ 100 โดยเริ่มต้นที่ขนาดประชากรเท่ากับ 4 และค่อยๆเพิ่มขนาดประชากรทีละ 2 จนถึงขนาดประชากรสูงสุด โดยขนาดประชากรในที่นี้ก็คือส่วนกลับของ Step size หรือความละเอียดในการค้นหา ซึ่งนำมาใช้ปรับปรุงค่าความน่าจะเป็นในแต่ละ element ของเวกเตอร์ความน่าจะเป็น เนื่องด้วยผลลัพธ์การทำงานของวงจรควอนตัมคือค่าความน่าจะเป็น ทำให้การวัดแต่ละครั้งอาจได้ผลลัพธ์แตกต่างกัน จึงจำเป็นต้องมีการกำหนดจำนวนช็อต ซึ่งก็คือจำนวนครั้งที่วงจรควอนตัมจะถูกทำงานซ้ำ เพื่อให้ได้การกระจายตัวของคำตอบที่มีความแม่นยำมากยิ่งขึ้น ดังนั้นผู้วิจัยจึงได้กำหนดจำนวนช็อตสำหรับขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับแก้ปัญหา TSP ขนาด 3 เมือง คือ 1 ช็อต และจำนวนช็อตสำหรับขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับแก้ปัญหา TSP ขนาด 4 เมือง คือ 1 ช็อต 10 ช็อต และ 20 ช็อต จำนวนช็อตที่ผู้วิจัยใช้ในการทดลองมาจากการทดลองเริ่มจากใช้จำนวนช็อตน้อยๆ และค่อยๆเพิ่มจนได้การกระจายตัวของคำตอบที่มีความแม่นยำที่เหมาะสม ซึ่งผู้วิจัยค้นพบว่าสำหรับปัญหา TSP ขนาด 3 เมือง จำนวนช็อตเพียง 1 ช็อตก็เพียงพอสำหรับการกระจายตัวของคำตอบที่มีความแม่นยำ แต่สำหรับปัญหา TSP ขนาด 4 เมือง การเพิ่มจำนวนช็อตมีผลต่อการการกระจายตัวของคำตอบให้มีความแม่นยำมากยิ่งขึ้น นอกจากนี้ผู้วิจัยได้กำหนดจำนวนรอบการวนซ้ำของโกรเวอร์ (Grover iteration) สำหรับ 3 เมือง คือ 2 รอบ เนื่องจากการเข้ารหัสปัญหา TSP ขนาด 3 เมืองบนวงจรควอนตัมใช้จำนวนคิวบิตทั้งหมด 4 คิวบิต ดังนั้นจำนวนรอบของโกรเวอร์สูงสุดที่ทำให้ความน่าจะเป็นของสถานะคิวบิตที่สนใจมีค่าสูงสุด คือ $\frac{\pi}{4}\sqrt{\frac{2^4}{2}} \approx 2$ ครั้ง ในขณะที่การเข้ารหัสสำหรับ 4 เมืองบนวงจรควอนตัมใช้จำนวนคิวบิตทั้งหมด 9 คิวบิต ดังนั้นจำนวนรอบของโกรเวอร์สูงสุดที่จะทำให้ความน่าจะเป็นของสถานะคิวบิตที่สนใจมีค่าสูงสุด คือ $\frac{\pi}{4}\sqrt{\frac{2^9}{6}} \approx 7$ ครั้ง ขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมมีการใช้จำนวนช็อตและจำนวนรอบของโกรเวอร์

แตกต่างกันเพื่อสังเกตผลกระทบของจำนวนช็อตและจำนวนรอบของโกรเวอร์ต่อประสิทธิภาพการทำงานของขั้นตอนวิธีที่นำเสนอ

ผู้วิจัยทำการรันขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (cGA*) และขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) ทั้งหมด 25 ครั้ง เพื่อหาค่าเฉลี่ยและนำมาพล็อตกราฟ และใช้การคัดเลือกโครโมโซมแบบการแข่งขัน (tournament selection) โดยกำหนดให้เท่ากับ 2 ขั้นตอนวิธีเชิงพันธุกรรมทั้งสองแบบจะรันจนกว่าค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นลู่เข้าค่า 0 หรือ 1

6.2 ผลการทดลองเปรียบเทียบประสิทธิภาพระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับชนิดควอนตัม

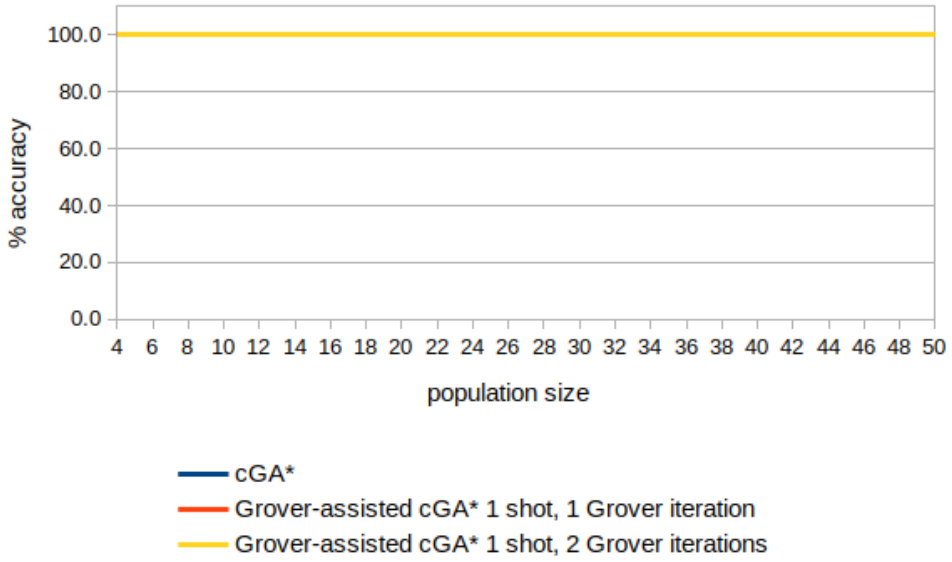
งานวิจัยนี้เปรียบเทียบประสิทธิภาพระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (cGA*) กับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) ใน 2 ด้าน ได้แก่ ความถูกต้องของคำตอบหน่วยเป็นเปอร์เซ็นต์ และจำนวนครั้งในการประเมินค่าความเหมาะสม (function evaluation) โดยวิธีการคำนวณจำนวน function evaluation ต่อหนึ่งรอบ generation สำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) คิดจากจำนวนช็อตคูณด้วยจำนวนรอบของโกรเวอร์ เนื่องจากทุกๆ 1 รอบของโกรเวอร์ ก็คือการประเมินค่าความเหมาะสม 1 ครั้งโดยที่ยังคงสถานะซ้อนทับของคิวบิตอยู่เพราะยังไม่ได้มีการอ่านค่าสถานะควอนตัม และบวกหนึ่งสำหรับการประเมินค่าความเหมาะสมของ first candidate จากวงจรควอนตัมโดยการอ่านค่าสถานะควอนตัม ดังสมการ (34)

$$\text{จำนวนครั้งที่ประเมินค่าความเหมาะสมต่อรอบ} = (\text{จำนวนช็อต} \times \text{จำนวนรอบโกรเวอร์}) + 1 \quad (34)$$

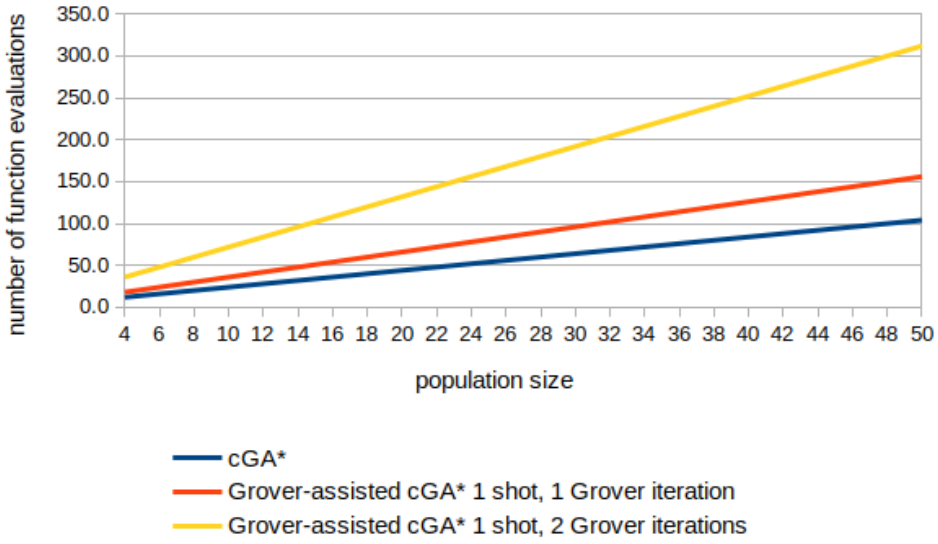
6.2.1 การวิเคราะห์ผลการทดลอง

ผลการทดลองแก้ปัญหา TSP ขนาด 3 เมืองด้วยขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) เปรียบเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (cGA*) ในด้านความถูกต้องของคำตอบหน่วยเป็นเปอร์เซ็นต์ เป็นดังภาพที่ 45 จากกราฟในภาพที่ 45 แสดงให้เห็นว่าทั้งสองอัลกอริทึมมีประสิทธิภาพเทียบเท่ากันในแง่ของความถูกต้องของคำตอบ โดยสามารถหาคำตอบที่เหมาะสมที่สุดได้ แต่เมื่อ

พิจารณาจำนวนครั้งในการประเมินค่าความเหมาะสมจากภาพที่ 46 cGA* จะใช้จำนวนครั้งในการประเมินค่าความเหมาะสมของคำตอบน้อยกว่า Grover-assisted cGA* เนื่องจากจำนวนครั้งในการประเมินค่าความเหมาะสมของ Grover-assisted cGA* จะเพิ่มขึ้นตามจำนวนช็อต และจำนวนรอบของโกรเวอร์ที่ใช้ในแต่ละช็อต โดยจำนวนรอบของโกรเวอร์ที่เพิ่มขึ้น ทำให้จำนวนครั้งที่ประเมินค่าความเหมาะสมของ Grover-assisted cGA* เพิ่มขึ้นเป็นเท่าตัวตามจำนวนรอบของโกรเวอร์ ซึ่งปัญหา TSP ขนาด 3 เมืองใช้จำนวนรอบของโกรเวอร์สูงสุดคือ 2 รอบ โดยที่จำนวนช็อต เพียง 1 ช็อตของ Grover-assisted cGA* ก็เพียงพอสำหรับการค้นหาคำตอบที่เหมาะสมที่สุด เนื่องจากปัญหา TSP ขนาด 3 เมือง ใช้จำนวนคิวบิตเพียง 4 คิวบิต แทนค่าสถานะของคิวบิตทั้งหมดที่เป็นไปได้ 16 สถานะควอนตัม จึงทำให้การกระจายตัวของคำตอบค่อนข้างมีความแม่นยำ แม้ว่าจะใช้จำนวนช็อตเพียง 1 ช็อตเท่านั้น กราฟแสดงจำนวนครั้งในการประเมินค่าความเหมาะสมไม่สามารถระบุได้ว่าขั้นตอนวิธีใดลู่เข้าหาคำตอบที่ถูกต้องได้เร็วกว่ากัน เนื่องจากวิธีวัดจำนวนครั้งที่แตกต่างกัน อย่างไรก็ตามเมื่อพิจารณากราฟแสดงจำนวนครั้งในการประเมินค่าความเหมาะสม พบว่าสำหรับปัญหา TSP ขนาด 3 เมือง ควรเลือกใช้ cGA* เนื่องจากใช้จำนวนครั้งประเมินค่าความเหมาะสมน้อยกว่า Grover-assisted cGA* มาก แต่ยังได้คำตอบที่เหมาะสมที่สุด

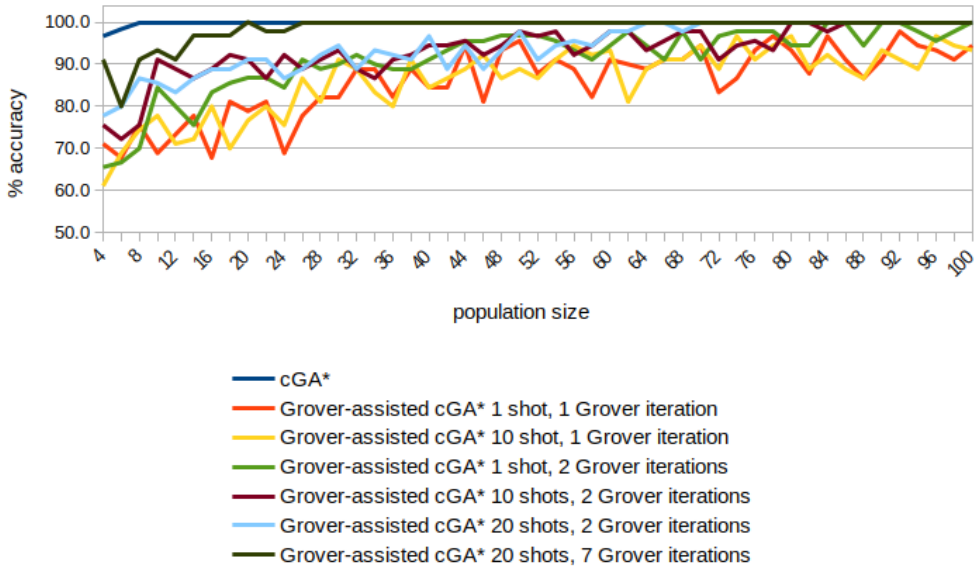


ภาพที่ 45 กราฟแสดงผลการเปรียบเทียบความถูกต้องของคำตอบ (จำนวนบิตที่ถูกต้องเป็นเปอร์เซ็นต์เมื่อสิ้นสุดการทำงาน) ระหว่าง cGA และ Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 3 เมือง*

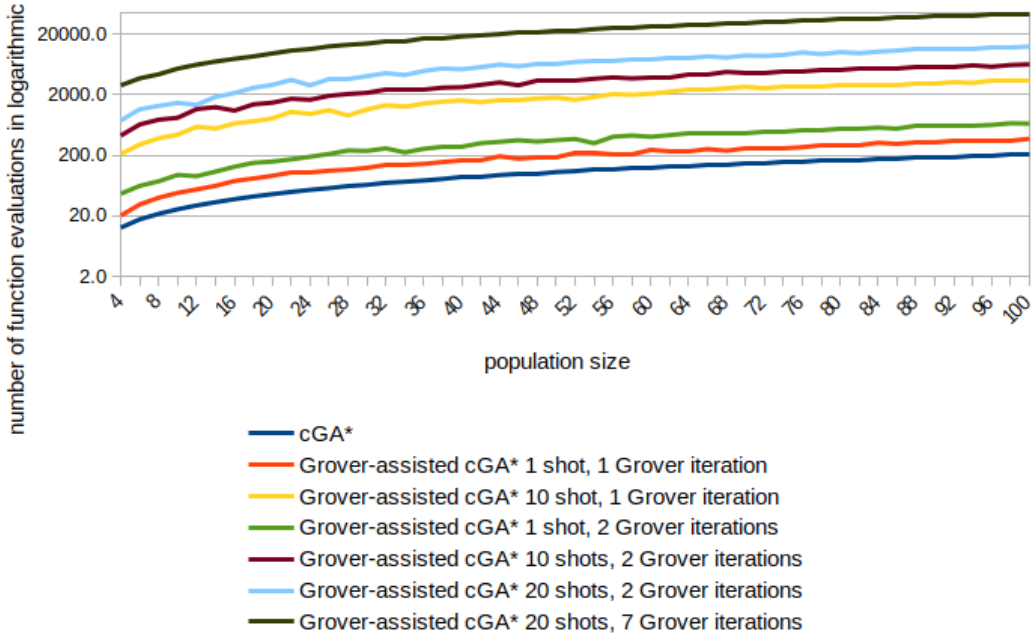


ภาพที่ 46 กราฟแสดงผลการเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสมจนกระทั่งสิ้นสุดการทำงาน ระหว่าง cGA และ Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 3 เมือง*

สำหรับปัญหา TSP ขนาด 4 เมือง cGA* สามารถหาคำตอบที่เหมาะสมที่สุดโดยใช้ขนาดของประชากรเพียง 8 และ ใช้จำนวนครั้งในการประเมินค่าความเหมาะสม 22 ในขณะที่ Grover-assisted cGA* ใช้ขนาดของประชากรประมาณ 26 จำนวนช็อต 20 ช็อต และใช้จำนวนรอบของโกรเวอร์สูงสุดคือ 7 รอบ จึงสามารถหาคำตอบที่เหมาะสมที่สุดได้ ดังภาพที่ 47 จะเห็นได้ว่า Grover-assisted cGA* ใช้ขนาดประชากรที่มากกกว่า cGA* นอกจากนี้เนื่องด้วยจำนวนช็อตและจำนวนรอบของโกรเวอร์ที่มาก ทำให้จำนวนครั้งในการประเมินค่าความเหมาะสมของ Grover-assisted cGA* มากตามไปด้วยดังภาพที่ 47 ซึ่งเป็นไปตามสมการ (34) อย่างไรก็ตามเมื่อพิจารณาผลการรัน Grover-assisted cGA* เปรียบเทียบกับ cGA* ตามภาพที่ 47 และ 48 จะเห็นได้ว่าการเพิ่มจำนวนช็อตไม่ได้มีผลต่อการเพิ่มประสิทธิภาพของอัลกอริทึมโดยตรง แต่ช่วยทำให้การกระจายตัวของคำตอบมีความแม่นยำมากขึ้น ทำให้คำตอบที่ได้มีความน่าเชื่อถือ ในขณะที่การเพิ่มจำนวนรอบของโกรเวอร์ ทำให้ประสิทธิภาพในการหาคำตอบของอัลกอริทึมที่นำเสนอดีขึ้น เนื่องจากจำนวนรอบของโกรเวอร์ที่สูงสุดจะช่วยเพิ่มความน่าจะเป็นที่วัดสถานะของคิวบิตแล้วได้คำตอบที่สนใจมากยิ่งขึ้น แต่ก็ทำให้จำนวนครั้งในการประเมินค่าความเหมาะสมมากด้วยเช่นกัน จากกราฟที่ 47 และ 48 จะเห็นได้ว่า cGA* ยังคงมีความเหมาะสมมากกว่าในการหาคำตอบปัญหา TSP ขนาด 4 เมือง เนื่องจากใช้จำนวนครั้งประเมินค่าความเหมาะสมน้อยกว่า Grover-assisted cGA* มาก แต่ยังได้คำตอบที่เหมาะสมที่สุด



ภาพที่ 47 กราฟแสดงผลการเปรียบเทียบความถูกต้องของคำตอบ (จำนวนบิตที่ถูกต้องเป็นเปอร์เซ็นต์เมื่อสิ้นสุดการทำงาน) ระหว่าง cGA และ Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 4 เมือง*



ภาพที่ 48 กราฟแสดงผลการเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสมจนกระทั่งสิ้นสุดการทำงาน ระหว่าง cGA และ Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 4 เมือง*

6.2.2 การวิเคราะห์ความซับซ้อนในเชิงควอนตัม

การวิเคราะห์ความซับซ้อนในเชิงควอนตัมจะแบ่งออกเป็น 2 ด้าน ได้แก่ 1: การวิเคราะห์จำนวนคิวบิต และจำนวนของคิวบิตทด (Ancilla qubits) ที่ต้องการที่สอดคล้องกับขนาดของปัญหาการหาค่าเหมาะสม และ 2: ต้นทุนของวงจรควอนตัมจากจำนวนเกต CNOT ที่ใช้ กับความลึกของวงจร (Circuit depth) ตามจำนวนรอบของโกรเวอร์ที่ใช้ใน Grover-assisted cGA* ซึ่งเกตควอนตัมส่วนมากประกอบจากเกตพื้นฐานของ arbitrary single-qubit และเกต CNOT อย่างไรก็ตามงานวิจัยนี้ ผู้วิจัยสนใจเฉพาะจำนวนเกต CNOT ที่ใช้ เนื่องจากเกต CNOT เป็นเกตที่มีต้นทุนสูง เพราะใช้เวลาในการประมวลผลนาน และมีโอกาสเกิดข้อผิดพลาดได้มากกว่าเกตพื้นฐานอื่นๆ นอกจากนี้จำนวนเกต CNOT ที่ใช้ในวงจรควอนตัมมักถูกนำมาใช้ในการทบทวนวรรณกรรมเพื่อเปรียบเทียบการออกแบบวงจรควอนตัมที่ดีสำหรับคอมพิวเตอร์เชิงควอนตัมโดยทั่วไป[45-47] นอกเหนือจากจำนวนเกต CNOT ที่ใช้แล้ว ความลึกของวงจรก็มีผลต่อต้นทุนและความซับซ้อนของวงจรควอนตัม ความลึกของวงจร (Circuit depth) มีความสำคัญเนื่องจากคิวบิตมีระยะเวลาในการเชื่อมโยงกันระหว่างคิวบิต หรือระยะเวลาที่คิวบิตยังมีคุณสมบัติทางกลศาสตร์ควอนตัม (Decoherence time) Decoherence time ที่จำกัดนี้ทำให้ความลึกของวงจรควอนตัมที่ยังคงสามารถทำงานได้โดยคงคุณสมบัติทางกลศาสตร์ควอนตัมถูกจำกัดตามไปด้วย

6.2.1.1 จำนวนของคิวบิตและคิวบิตทดที่ต้องใช้

จำนวนคิวบิตที่จำเป็นต้องใช้สำหรับเข้ารหัสปัญหา TSP ในรูปแบบแบบจำลอง Ising คือ $(n-1)^2$ คิวบิต เมื่อ n คือจำนวนเมือง นอกจากนี้ยังจำเป็นต้องใช้คิวบิตทดระหว่างการคำนวณนอกเหนือจากคิวบิตที่เป็น input และ output เพื่อเร่งกระบวนการคำนวณโดยการลดจำนวนเกตรวมที่ต้องใช้หรือความลึกของวงจร สำหรับจำนวนคิวบิตทดที่ต้องใช้ เราจำเป็นต้องพิจารณาแบบจำลอง Ising ของปัญหา TSP ตามที่ระบุในข้อ 5.1 ตัวอย่างเส้นทางการเดินทาง 1 -> 2 -> 3 -> 4 -> 1 เราสนใจเฉพาะลำดับของเมือง 2 3 และ 4 เนื่องจากเรากำหนดให้พนักงานขายเริ่มเดินทางจากเมือง 1 เท่านั้น โดยลำดับของเมืองอาจปลี่ยนแปลงภายใน 3 เมืองนี้ ภาพที่ 49 แสดงค่าในเมทริกซ์ของเส้นทางการเดินทาง 2 -> 3 -> 4 โดยที่แถวของเมทริกซ์แทนเมือง และคอลัมน์ของเมทริกซ์แทนลำดับของเมืองนั้นในเส้นทางการเดิน ซึ่งแถวที่

1 คือเมือง 2 แถวที่ 2 คือเมือง 3 และแถวที่ 3 คือเมือง 4 จากเมทริกซ์ดังกล่าวจะเห็นได้ว่าแต่ละเมืองจะมีลำดับของเมืองที่แตกต่างกัน โดยเมือง 2 อยู่ลำดับที่ 1 เมือง 3 อยู่ลำดับที่ 2 และเมือง 4 อยู่ลำดับที่ 3 ในวัฏจักรฮามิลตัน ดังนั้นเราสามารถตรวจสอบว่าแต่ละแถวในแต่ละคอลัมน์มี 1 ปรากฏอยู่โดยไม่ซ้ำกันหรือไม่ จึงมีเงื่อนไขทั้งหมด 6 เงื่อนไขที่ตองตรวจสอบสำหรับตัวอย่าง TSP ขนาด 4 เมืองข้างต้น เงื่อนไขที่ 1: ตรวจสอบแถวบนสุดกับคอลัมน์ที่ 1 ถึงคอลัมน์ที่ 3 เงื่อนไขที่ 2: ตรวจสอบแถวกลางกับคอลัมน์ที่ 1 ถึงคอลัมน์ที่ 3 เงื่อนไขที่ 3: ตรวจสอบแถวล่างสุดกับคอลัมน์ที่ 1 ถึงคอลัมน์ที่ 3 เงื่อนไขที่ 4: ตรวจสอบคอลัมน์ซ้ายสุดกับแถวที่ 1 ถึงแถวที่ 3 เงื่อนไขที่ 5: ตรวจสอบคอลัมน์กลางกับแถวที่ 1 ถึงแถวที่ 3 และเงื่อนไขที่ 6: ตรวจสอบคอลัมน์ซ้ายสุดกับแถวที่ 1 ถึงแถวที่ 3 ดังนั้นจำนวนคิวบิตทดทั้งหมดที่จำเป็นต้องใช้คือ $2(n-1)$ คิวบิต เมื่อ *n* คือจำนวนเมือง และต้นทุนทั้งหมดในการกำหนดสถานะเริ่มต้นของคิวบิตคือผลรวมของจำนวนคิวบิตที่ใช้สำหรับการเข้ารหัสปัญหา TSP และจำนวนคิวบิตทดสำหรับตรวจสอบเงื่อนไขดังกล่าวข้างต้นคือ $O(n^2-1)$ ตารางที่ 5 แสดงจำนวนคิวบิตที่ต้องใช้ในการกำหนดสถานะควอนตัมเริ่มต้นของรูปแบบคำตอบของปัญหา จำนวนเกต CNOT ที่ใช้ในการคำนวณ และความลึกของวงจร ซึ่งจะได้อธิบายในหัวข้อถัดไป

	1	2	3
1	1	0	0
2	0	1	0
3	0	0	1

ภาพที่ 49 ตัวอย่างเมทริกซ์ของเส้นทางการเดินทาง 1 -> 2 -> 3 -> 4 ->

จำนวนเมือง	จำนวนคิวบิต	จำนวนคิวบิตทด	จำนวนเกต CNOT	ความลึกของวงจร
3	$(n-1)^2$	$2(n-1)$	$57t$	$2+99t$
4	$(n-1)^2$	$2(n-1)$	$315t$	$1+549t$

ตารางที่ 5 สรุปต้นทุนวงจรควอนตัม จำแนกตามจำนวนคิวบิต จำนวนคิวบิตทด จำนวนเกต CNOT และความลึกของวงจร สำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม

6.2.1.2 ต้นทุนของวงจร

โดยทั่วไปวงจรควอนตัมที่ยิ่งมีจำนวนเกตและจำนวนคิวบิตที่มาก ก็ยิ่งเพิ่มความไม่เสถียรในการทำงานของวงจร เนื่องด้วยอัตราความผิดพลาดของเครื่องคอมพิวเตอร์เชิงควอนตัมอันเนื่องมาจากความผิดพลาดของควอนตัมเกต การวัด การสื่อสารข้ามอุปกรณ์ และประสิทธิภาพคอมไพเลอร์ของวงจรควอนตัม โดยอัตราความผิดพลาดเพิ่มขึ้นตามจำนวนเกตและจำนวนคิวบิตที่ใช้ การสร้างวงจรควอนตัมที่สามารถทำงานได้บนเครื่องคอมพิวเตอร์เชิงควอนตัมจึงเป็นเรื่องที่ท้าทายอย่างมาก ยิ่งไปกว่านั้นเราไม่สามารถสร้างอัลกอริทึมที่ซับซ้อนบนวงจรควอนตัมได้เนื่องจากข้อจำกัดในเรื่อง Decoherence time ของสถานะควอนตัม

งานวิจัยนี้ใช้ประโยชน์จากอัลกอริทึมควอนตัมนั่นคือ อัลกอริทึมค้นหาของโกรเวอร์ในการค้นหาคำตอบที่เหมาะสมที่สุดของปัญหา TSP โดยใช้เวลาในการค้นหาเร็วขึ้นเป็นกำลังสองของอัลกอริทึมแบบดั้งเดิมทั่วไป ซึ่งอัลกอริทึมการค้นหาของโกรเวอร์ประกอบด้วยวงจรส่วนที่กำหนดค่าเริ่มต้นของสถานะคิวบิต ส่วนการทำงานของฟังก์ชันโอราเคิล ส่วนการทำงานของ Diffusor และส่วนสุดท้ายคือการวัดสถานะคิวบิตเพื่อดูผลลัพธ์ ส่วนกำหนดค่าเริ่มต้นของสถานะคิวบิตและการวัดสถานะคิวบิตนั้นมีจำนวนเกตที่แน่นอน เนื่องจาก 2 ส่วนนี้เป็นเพียงการเตรียมข้อมูลขาเข้า และการนำข้อมูลออก ในขณะที่ส่วนการทำงานของฟังก์ชันโอราเคิลและ Diffusor คือส่วนสำคัญของอัลกอริทึมค้นหาของโกรเวอร์ เนื่องจากเป็นส่วนที่คำนวณฟังก์ชันโอราเคิล และทำการขยายแอมพลิจูดของสถานะควอนตัมที่สนใจซึ่งถูกกำหนดจากฟังก์ชันโอราเคิล สองส่วนนี้สามารถถูกวนซ้ำได้เพื่อเพิ่มค่าความน่าจะเป็นสูงสุดให้กับสถานะควอนตัมที่สนใจ ทำให้เมื่อวัดสถานะสุดท้ายแล้วได้ค่าสถานะควอนตัมที่ต้องการ การวนซ้ำของสองส่วนนี้ในอัลกอริทึมค้นหาของโกรเวอร์เรียกว่า “Grover iteration” ดังนั้นความลึกของวงจรควอนตัมจะพิจารณาจากจำนวนครั้งของ Grover iteration ที่ใช้ ยิ่งวนซ้ำมากความลึกของวงจรก็มากตามจำนวนรอบที่วนซ้ำเป็นเท่าตัว สำหรับปัญหา TSP ขนาด 3 เมือง จำนวนเกต CNOT ทั้งหมดที่ต้องใช้คือ $57t$ และความลึกของวงจร คือ $2 + 99t$ โดย t คือจำนวนครั้งของ Grover iteration ส่วนปัญหา TSP ขนาด 4 เมือง จำนวนเกต CNOT ทั้งหมดที่ต้องใช้คือ $315t$ และความลึกของวงจร คือ $1 + 549t$ จะเห็นได้ว่าการ

เพิ่มจำนวนเมืองมาแค่ 1 เมือง แต่จำนวนคิวบิต และจำนวนคิวบิตทดที่ต้องใช้เพิ่มขึ้นเป็นกำลังสอง และจำนวนคิวบิตที่เพิ่มขึ้นทำให้พื้นที่การค้นหาของอัลกอริทึมการค้นหาของโกรเวอร์เพิ่มขึ้นเป็นเอ็กโปเน็นเชียล (Exponential) นั่นหมายถึงจำนวนครั้งของ Grover iteration ก็เพิ่มขึ้นเป็นเอ็กโปเน็นเชียลเช่นกัน จึงทำให้จำนวนเกต CNOT และความลึกของวงจรสำหรับปัญหา TSP ขนาด 4 เมือง เพิ่มขึ้นอย่างมากเมื่อเทียบกับปัญหา TSP ขนาด 3 เมือง จึงสามารถสรุปได้ว่าความซับซ้อนในเชิงควอนตัมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) ที่นำเสนอในงานวิจัยนี้เท่ากับ $O\left(I\frac{\pi}{4}\sqrt{\frac{N}{T}}\right)$ เมื่อ I คือความลึกของวงจรควอนตัมสำหรับ Grover iteration ครั้งแรก N คือจำนวนสถานะควอนตัมทั้งหมด และ T คือจำนวนสถานะควอนตัมที่เป็นคำตอบของฟังก์ชันโอราเคิล

บทที่ 7
การปรับปรุงฟังก์ชันโอราเคิล

แม้ว่าการนำประโยชน์ของอัลกอริทึมการค้นหาของโกรเวอร์มาใช้กับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับตามที่นำเสนอในงานวิจัยนี้จะสามารถลดจำนวนครั้งในการค้นหาคำตอบของปัญหาการค้นหาข้อมูลที่ไม่มีโครงสร้างเป็นกำลังสอง เมื่อเทียบกับอัลกอริทึมการค้นหาแบบดั้งเดิม (classic search algorithm) แต่เนื่องจากขนาดของปัญหา TSP ที่มีจำนวนเมืองน้อย จึงทำให้ผลการวิจัยไม่ได้แสดงถึงประโยชน์ของการนำอัลกอริทึมการค้นหาของโกรเวอร์มาเพิ่มประสิทธิภาพในการค้นหาคำตอบของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับ อีกทั้งฟังก์ชันโอราเคิลที่ใช้ในอัลกอริทึมการค้นหาของโกรเวอร์ เป็นเพียงการกำหนดลักษณะคำตอบที่ฟังก์ชันโอราเคิลจะให้ผลลัพธ์คำตอบที่เป็น feasible path เท่านั้น ทำให้รูปแบบเส้นทางทั้งหมดที่เป็นไปได้ในพื้นที่การค้นหาคือ $N!$ รูปแบบ เมื่อ N คือจำนวนเมือง และเนื่องจากผู้วิจัยได้กำหนดให้พนักงานขายเริ่มต้นเดินทางที่เมืองใดเมืองหนึ่งเสมอ รูปแบบเส้นทางทั้งหมดที่ต้องค้นหาจึงเหลือ $(N - 1)!$ รูปแบบ ซึ่งพื้นที่ค้นหายังคงเพิ่มขึ้นตามจำนวนเมืองที่เพิ่มขึ้นเป็นเอ็กโปเน็นเชียลอยู่ ผู้วิจัยจึงทำการปรับปรุงฟังก์ชันโอราเคิลที่ใช้ในขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) โดยใช้ประโยชน์จาก quantum parallelism ในการเปรียบเทียบเส้นทางที่ดีที่สุด ณ ปัจจุบัน กับเส้นทางทั้งหมดในพื้นที่การค้นหาซึ่งอยู่ในรูปแบบสถานะ quantum superposition โดยเป็นเปรียบเทียบกับทุกสถานะควอนตัมในเวลาเดียวกัน เพื่อให้ฟังก์ชันโอราเคิลตรวจสอบเฉพาะสถานะควอนตัมของรูปแบบคำตอบที่เป็น feasible path และระยะทางรวมไม่มากกว่าระยะทางรวมที่สั้นที่สุด ณ ปัจจุบัน

7.1 รายละเอียดการปรับปรุงฟังก์ชันโอราเคิล

การใช้ประโยชน์จาก quantum parallelism ในการเปรียบเทียบเส้นทางทั้งหมดในพื้นที่การค้นหากับเส้นทางที่ดีที่สุด ณ ปัจจุบัน ผู้วิจัยจำเป็นต้องปรับปรุงฟังก์ชันโอราเคิลให้สามารถคำนวณระยะทางจากรูปแบบของคำตอบทั้งหมดในพื้นที่การค้นหาได้ และฟังก์ชันโอราเคิลจะต้องสามารถเปรียบเทียบระยะทางรวมของแต่ละรูปแบบคำตอบทั้งหมดในพื้นที่การค้นหา กับระยะทางรวมที่สั้นที่สุด ณ ปัจจุบันได้ด้วย ด้วยเหตุนี้ผู้วิจัยจึงประยุกต์ใช้วงจรบวกที่ใช้หลักการของควอนตัมฟูริเยร์ทรานสฟอร์ม (quantum Fourier transform) มาใช้สำหรับหาผลรวมระยะทางของแต่ละ

รูปแบบคำตอบในพื้นที่การค้นหา และใช้ quantum comparator ในการเปรียบเทียบระยะทางของแต่ละรูปแบบคำตอบกับระยะทางที่สั้นที่สุด ณ ปัจจุบัน

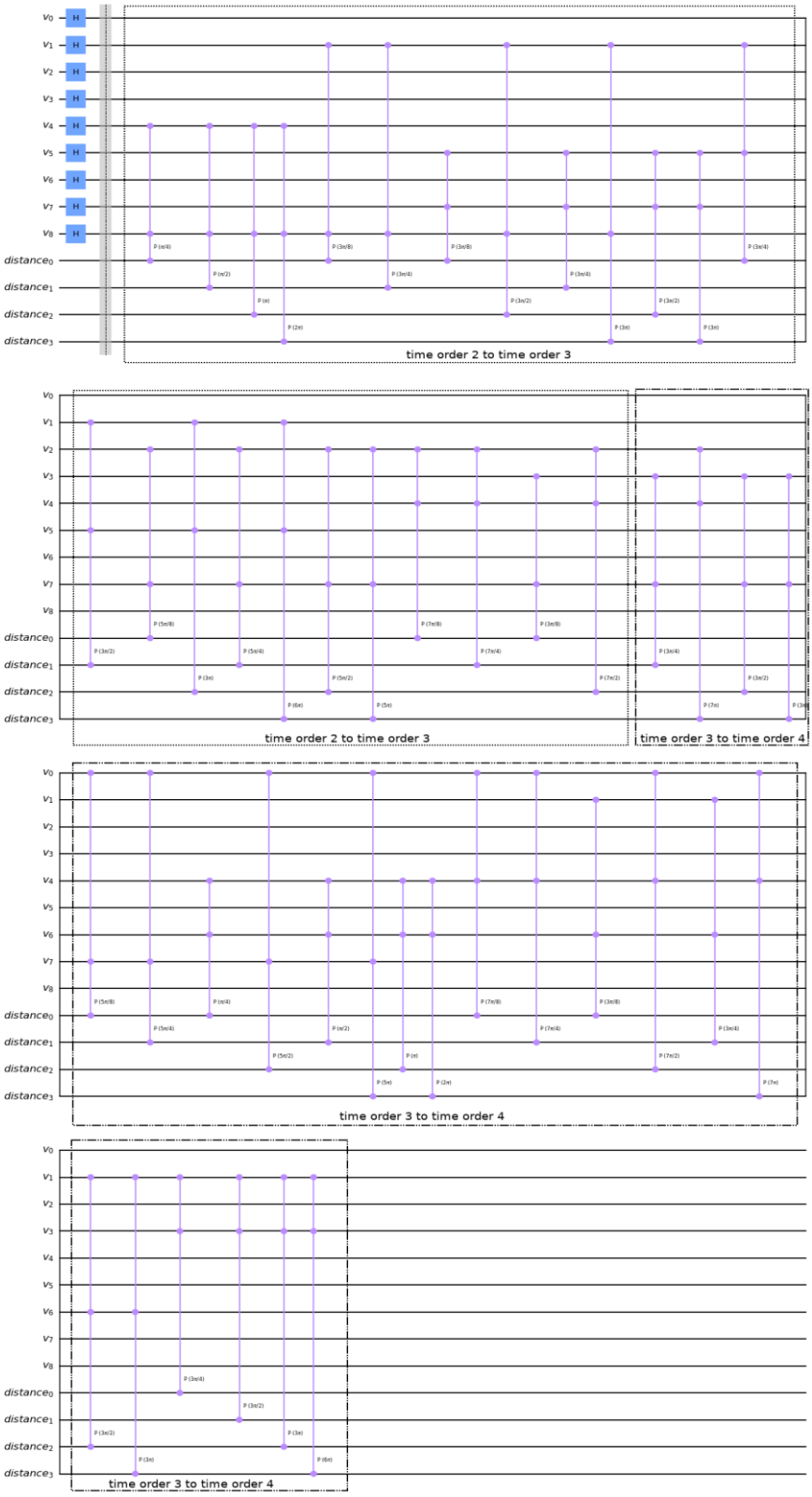
7.1.1. <u>การหาผลรวมระยะทางโดยใช้วงจรบวกที่ใช้ควอนตัมฟูริเยร์ทรานสฟอร์ม</u>

แนวคิดการบวกบนวงจรควอนตัมถูกนำเสนอครั้งแรกโดย Draper[9] ซึ่งใช้แนวคิดของการแปลงฟูริเยร์เชิงควอนตัม (quantum Fourier transform) ดังสมการต่อไปนี้

$$|j_1, \cdots, j_n\rangle \rightarrow \frac{(|0\rangle+e^{2\pi i 0.j_n}|1\rangle)\ldots(|0\rangle+e^{2\pi i 0.j_1\cdots j_n}|1\rangle)}{2^{n/2}} \quad (35)$$

มาใช้ในการสร้างวงจรบวกควอนตัม โดยใช้คุณสมบัติที่สำคัญของพื้นฐานการคำนวณฟูริเยร์ คือผลรวมของการแปลงฟูริเยร์ของสองสัญญาณเท่ากับการแปลงฟูริเยร์ของผลรวมของสองสัญญาณนั้น เราจึงสามารถนำการแปลงฟูริเยร์มาใช้สำหรับสร้างวงจรบวกควอนตัมได้ โดยประกอบด้วยขั้นตอนดังนี้ ขั้นแรกคือการแปลงข้อมูลที่เป็นพื้นฐานการคำนวณทั่วไป (Computational basis) ให้กลายเป็นการพื้นฐานการคำนวณฟูริเยร์ (Fourier basis) โดยใช้ direct Fourier transformation ดังภาพที่ 4 เป็นตัวอย่างการแปลงจาก computation basis ไปเป็น Fourier basis สำหรับ 3 คิวบิต ขั้นตอนถัดมาสร้างวงจรบวกเชิงควอนตัมสำหรับทุกๆเส้นเชื่อมระหว่าง 2 เมืองที่เป็นไปได้ทั้งหมดโดยใช้ multi-controlled-Phase gate (MCP gate) เนื่องจากงานวิจัยนี้ถือว่าทุกเมืองมีเส้นทางเชื่อมถึงกัน (Fully connected) โดย MCP gate จะทำให้เกิดเฟสบนสถานะของคิวบิตเป้าหมายตามค่ามุมที่กำหนด เมื่อคิวบิตควบคุมมีสถานะเป็น $|1\rangle$ ตัวอย่างเช่น ปัญหา TSP ขนาด 4 เมือง เนื่องจากงานวิจัยนี้ ผู้วิจัยได้กำหนดให้เมืองเริ่มต้นคือเมือง 1 เสมอ ดังนั้นเราจำเป็นต้องหาเมืองที่พนักงานขายเดินทางเป็นลำดับที่ 2⟶3 และ 3⟶4 ตามลำดับ ผลลัพธ์ที่เป็นไปได้ทั้งหมดสำหรับเมืองที่พนักงานขายเดินทางเป็นลำดับที่ 2 ⟶ 3 ได้แก่ เมือง 2⟶3 เมือง 2⟶4 เมือง 3⟶2 เมือง 3⟶4 เมือง 4⟶3 และเมือง 4⟶2 เช่นเดียวกันกับเมืองที่พนักงานเดินทางเป็นลำดับที่ 3 ⟶ 4 จะมีเส้นทางที่เป็นไปได้ทั้งหมด 6 เส้นทางดังกล่าวข้างต้น โดยไม่ได้สนใจว่าลำดับที่ 2 ⟶ 3 พนักงานเดินทางเส้นใดไปแล้ว นอกจากนี้ปัญหา

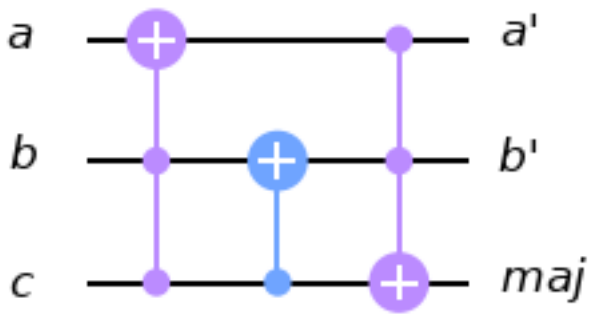
TSP ถูกแปลงให้อยู่ในรูปแบบจำลอง Ising ที่ใช้จำนวนคิวบิตทั้งหมด $(N-1)^2$ คิวบิต เมื่อ *N* คือจำนวนเมือง ซึ่งแต่ละคิวบิตคือค่าไบนารี่ $X_{i,p}$ และ $X_{j,p+1}$ มีค่าเป็น 1 ทั้งคู่ก็ต่อเมื่อพนักงานขายเดินทางจากเมือง *i* ไปเมือง *j* เพราะเมือง *i* ถูกแวะที่ลำดับที่ *p* และเมือง *j* ถูกแวะที่ลำดับที่ *p*+1 ดังนั้นเราจึงสามารถใช้ 2 คิวบิตดังกล่าวเป็นคิวบิตควบคุม ในกรณีที่มีการเดินทางจากเมือง *i* ไปเมือง *j* คิวบิตควบคุมทั้งสองจะมีค่าเป็น 1 ทั้งคู่ ซึ่งจะทำให้คิวบิตเป้าหมายถูกปรับเฟสด้วยค่ามุมที่กำหนด ในที่นี้ก็คือค่าระยะทางจากเมือง *i* ไปเมือง *j* $(w_{i,j})$ นั่นเอง โดยจะเป็นการหมุนคิวบิตเริ่มจากคิวบิตซ้ายสุดด้วยจำนวนรอบ $\frac{w_{i,j}}{2^n} \times 2\pi\, radians$ และค่ามุมจะเพิ่มขึ้นเป็นเท่าตัวสำหรับคิวบิตลำดับถัดไป เป็นเช่นนี้ไปเรื่อยๆจนถึงคิวบิตขวาสุด อย่างไรก็ตามสำหรับลำดับของเมืองลำดับที่ 2⟶3 นอกจากปรับเฟสด้วยระยะทางจากเมือง *i* ไปเมือง *j* $(w_{i,j})$ แล้ว ต้องเพิ่มระยะทางจากเมือง 1 ไปเมือง *i* ด้วย เนื่องจากพนักงานขายเริ่มเดินทางจากเมือง 1 เป็นเมืองแรก ค่ามุมที่ใช้ในการหมุนคิวบิตสำหรับเกต MCP จะกลายเป็น $\frac{w_{1,i}+w_{i,j}}{2^n} \times 2\pi\, radians$ เช่นเดียวกันกับลำดับของเมืองลำดับที่ N-1⟶N ก็ต้องเพิ่มระยะทางจากเมือง *N* ไปเมือง *1* $(w_{N,1})$ ด้วย เนื่องจากพนักงานขายต้องเดินทางกลับมาที่เมืองเริ่มต้นเสมอ ดังนั้นจำนวนเกต MCP ที่ต้องใช้ทั้งหมดสำหรับการหาผลรวมระยะทางโดยใช้วงจรบวกที่ใช้ควอนตัมฟูริเยร์ทรานสฟอร์มคือ $(N-1)!\,(N-2)$ เกต เมื่อ *N* คือจำนวนเมือง ตัวอย่างวงจรบวกสำหรับปัญหา TSP ขนาด 4 เมือง แสดงดังภาพที่ 50 จากนั้นจึงแปลงจากพื้นฐานการคำนวณฟูริเยร์กลับไปเป็นพื้นฐานการคำนวณทั่วไปเช่นเดิม เพื่อให้สามารถเปรียบเทียบกับระยะทางที่สั้นที่สุดซึ่งอยู่ในรูปแบบพื้นฐานการคำนวณทั่วไปได้ โดยใช้ inverse Fourier transform ดังตัวอย่างในภาพที่ 5 ซึ่งเป็น inverse Fourier transform ของ 3 คิวบิต



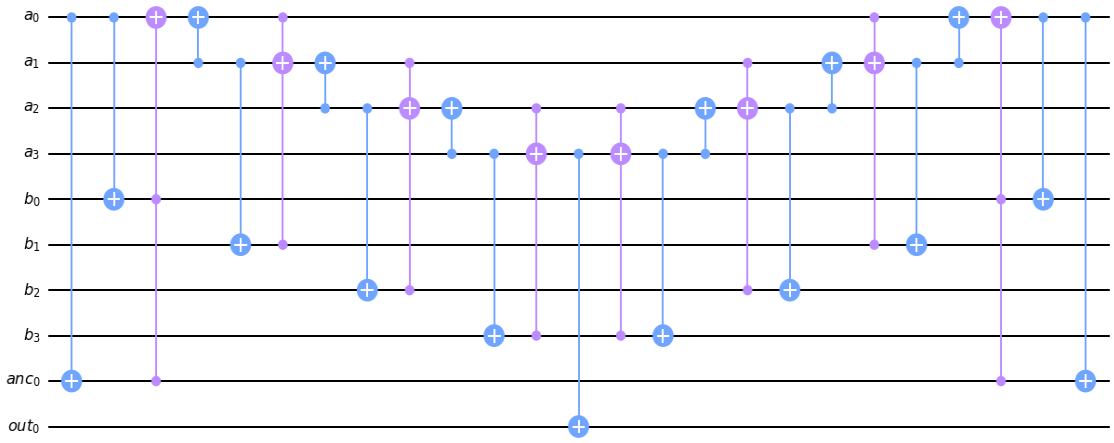
ภาพที่ 50 การหาระยะทางของปัญหา TSP 4 เมือง โดยใช้วงจรบวกที่ใช้แนวคิดควอนตัมฟูริเยร์ทรานสฟอร์ม

7.1.2. การเปรียบเทียบระยะทางโดยใช้ quantum comparator

เมื่อได้ผลรวมระยะทางของแต่ละรูปแบบคำตอบที่อยู่ในสถานะซ้อนทับ (superposition) เรียบร้อยแล้ว เราสามารถทำการเปรียบเทียบระยะทางของคำตอบดังกล่าวกับระยะทางที่สั้นที่สุด ณ ปัจจุบันที่มีการบันทึกค่าไว้ โดยใช้ quantum comparator ในการระบุว่า B น้อยกว่า A หรือไม่ งานวิจัยนี้ใช้วงจรบวกควอนตัมในการเปรียบเทียบที่เรียกว่า “Cuccaro adder” ถูกนำเสนอในปี 2004 โดย Cuccaro และทีมงาน[48] ซึ่งเป็น quantum comparator ที่นำมาใช้ในการตรวจสอบว่าระยะทางของรูปแบบคำตอบใด (B) ที่น้อยกว่าระยะทางที่สั้นที่สุด ณ ปัจจุบัน (A) quantum comparator ดังกล่าวเริ่มต้นโดยการกำหนด B ให้เป็น inverse B (B’) จากนั้นจึงบวก A และ B’ เข้าด้วยกัน ขั้นตอนถัดมาคือตรวจสอบ carry out bit เนื่องจากมันสามารถใช้ระบุได้ว่า B น้อยกว่า A หรือไม่ วงจร MAJ หรือ เกต MAJ แสดงดังภาพที่ 51 คือส่วนสำคัญของวงจรบวกควอนตัมนี้ โดยเกต MAJ รับอินพุต 3 คิวบิต และให้เอ้าท์พุต 3 คิวบิตเช่นกัน เกต MAJ จะตรวจสอบ majority ของอินพุตทั้งหมด และเปลี่ยนเอ้าท์พุตให้เป็นค่า majority ซึ่งอาจเป็นทั้ง 0 หรือ 1 เราสนใจเพียงแค่บิตสูงสุดของวงจรบวกโดยไม่ได้สนใจค่าผลรวม ดังนั้นเราสามารถสร้าง quantum comparator โดยกระจาย majority ของทั้ง 3 คิวบิตที่เป็นอินพุตไปยัง most significant bit ซึ่งเป็นคิวบิตเอ้าท์พุต และทำการย้อนตัวดำเนินการทั้งหมดเพื่อรีเซ็ตคิวบิตทดทั้งหมดกลับไปเป็นค่าเริ่มต้นก่อนทำการวัดคิวบิตเอ้าท์พุต วงจร quantum comparator สำหรับ 4 คิวบิตโดยใช้ Cuccaro adder แสดงดังภาพที่ 52



ภาพที่ 51 เกต majority (MAJ)



ภาพที่ 52 วงจร quantum comparator สำหรับ 4 คิวบิต โดยใช้ Cuccaro adder[48]

ในการตรวจสอบว่าระยะทางของรูปแบบคำตอบใดๆนั้น (B) น้อยกว่าระยะทางที่สั้นที่สุด ณ ปัจจุบัน (A) หรือไม่ เราจำเป็นต้องเข้ารหัสระยะทางที่สั้นที่สุด ณ ปัจจุบันในรูปแบบเลขฐานสอง และทำการใส่ complement ของระยะทางของรูปแบบคำตอบใดๆ (B) ในพื้นที่การค้นหาของอัลกอรึทึมการค้นหาของโกรเวอร์ จากนั้นจึงหาผลรวมระหว่าง A และ B’ และตรวจสอบคิวบิตเอ้าท์พุต (carry output qubit) ผลลัพธ์คิวบิตเอ้าท์พุตจะมีค่าเป็น “1” ถ้าระยะทางของรูปแบบคำตอบนั้นๆมีค่าน้อยกว่าระยะทางที่สั้นที่สุด ณ ปัจจุบัน และเป็น “0” ถ้าไม่ใช่

7.2 ภาพรวมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง

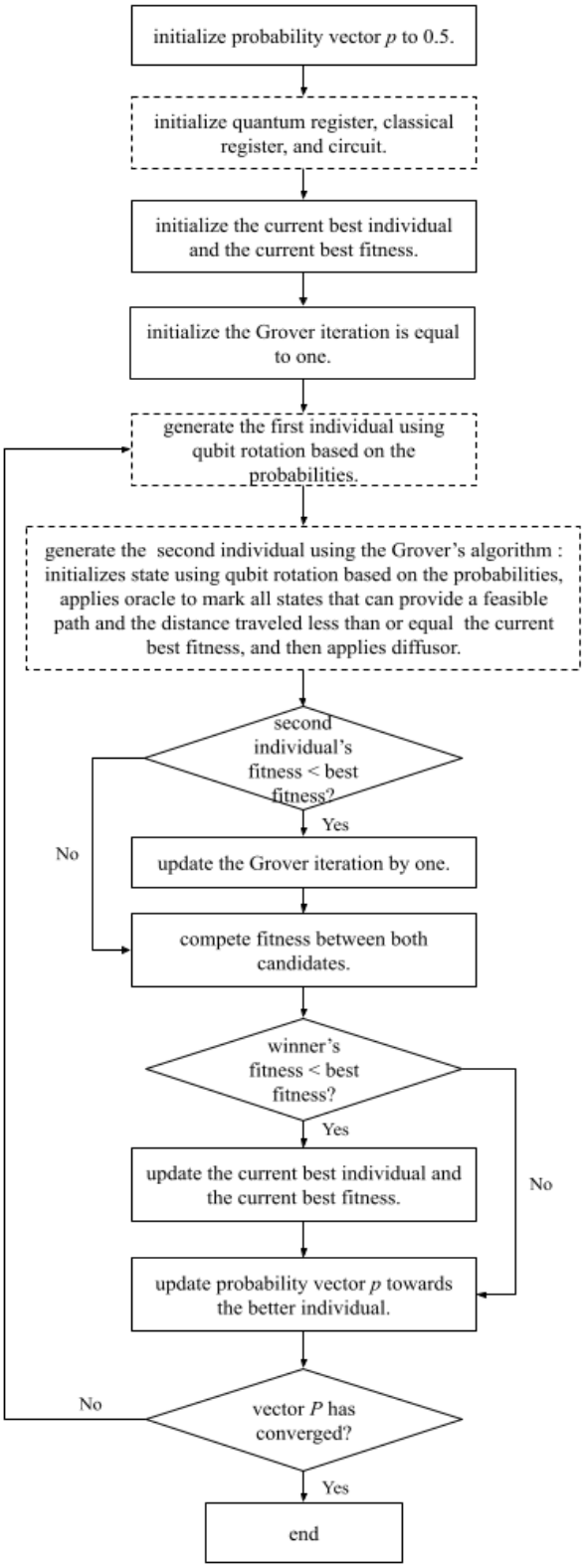
ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง นั้นมีโครงสร้างการทำงานของอัลกอริทึมคล้ายกันกับของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่ได้กล่าวไปในบทที่ 5 ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม แตกต่างกันตรงที่**การคำนวณระยะทางของรูปแบบคำตอบทั้งหมดจากเดิมที่คำนวณที่เครื่องคอมพิวเตอร์คลาสสิก เปลี่ยนมาทำการคำนวณที่เครื่อง IBM QASM simulation แทน โดยคำนวณในฟังก์ชันโอราเคิลเพื่อให้สามารถใช้ quantum comparator ในการเปรียบเทียบกับระยะทางที่สั้นที่สุด ณ ปัจจุบันได้ โดยเป็นการเปรียบเทียบโดยใช้ประโยชน์จาก quantum parallelism กล่าวคือสามารถเปรียบเทียบ**

ระยะทางที่สั้นที่สุดกับสถานะควอนตัมที่เป็น superposition ซึ่งแทนสถานะคำตอบทุกรูปแบบในเวลาเดียวกันได้ ซึ่งจะทำให้ผลลัพธ์ของการค้นหาของโกรเวอร์ได้คำตอบที่ไม่แย่กว่าเดิม โดยไม่จำเป็นต้องทำการเปรียบเทียบกับระยะทางที่สั้นที่สุด ณ ปัจจุบันบนคอมพิวเตอร์ดั้งเดิมเหมือนขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมก่อนหน้านี้ และยังคงได้ผลลัพธ์เหมือนเดิมโดยเป็นการคำนวณที่ฝั่งคอมพิวเตอร์ควอนตัมทั้งหมด นอกจากนี้**ผู้วิจัยไม่สามารถกำหนดจำนวนรอบของโกรเวอร์ที่แน่นอนเหมือนขั้นตอนวิธีเดิมได้เนื่องจากเราไม่ทราบจำนวนคำตอบที่เป็นผลลัพธ์ของฟังก์ชันโอราเคิลที่แน่นอนเพราะเราไม่ทราบว่ารูปแบบระยะทางที่ไม่มากกว่าระยะทางที่สั้นที่สุด ณ ปัจจุบันมีกี่รูปแบบ จึงจำเป็นต้องใช้การกำหนดจำนวนรอบของโกรเวอร์ แบบ adaptive** คือ เริ่มต้นจำนวนรอบของโกรเวอร์ ที่ 1 เสมอ แสดงดังภาพที่ 53 ขั้นตอนที่ 2 t คือจำนวนรอบของโกรเวอร์ (Grover iteration) ซึ่งถูกกำหนดค่าเริ่มต้นให้เป็น 1 เสมอ จากนั้นจึงค่อยๆเพิ่มจำนวนรอบของโกรเวอร์ ทีละ 1 ดังขั้นตอนที่ 7 เมื่อทำการวัดสถานะคิวบิตแล้วได้รูปแบบคำตอบที่ไม่ดีกว่ารูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน เพราะนั่นแสดงว่าจำนวนรอบของโกรเวอร์อาจยังไม่มากพอสำหรับเพิ่มความน่าจะเป็นของคำตอบที่เหมาะสมให้มีค่ามากที่สุด ซึ่งจำนวนรอบของโกรเวอร์ที่เพิ่มขึ้นมานี้จะถูกนำไปใช้สำหรับการสร้าง Second candidate ในรอบถัดไป ในทางกลับกัน ถ้าคำตอบที่วัดได้ดีกว่าหรือเท่ากับรูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน ก็จะไม่เพิ่มจำนวนรอบของโกรเวอร์ อย่างไรก็ตามส่วนการคำนวณค่าความเหมาะสมระหว่าง first individual และ second individual เพื่อหาผู้ชนะ รวมถึงการอัพเดตค่าความน่าจะเป็นในแต่ละ element ของเวกเตอร์ความน่าจะเป็นยังเป็นการคำนวณด้วยคอมพิวเตอร์ดั้งเดิม ภาพที่ 54 แสดงขั้นตอนการทำงานของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (เวอร์ชันปรับปรุงฟังก์ชันโอราเคิล) โดยกรอบเส้นประคือประมวลผลบนเครื่อง IBM QASM simulator และกรอบเส้นทึบคือประมวลผลบนเครื่องคอมพิวเตอร์ดั้งเดิม

Algorithm 3: new Grover-assisted cGA*

1) initialize the probability vector:
for $i \leftarrow 1$ *to* l **do**
  $p[i] \leftarrow 0.5$;
end
2) initialize the current best individual and Grover iteration:
$curBestIdv \leftarrow 000..00, t \leftarrow 1$;
3) initialize quantum and classical register, circuit:
$circuit \leftarrow QuantumCircuit(qr, cr)$;
4) generate the first individual using qubit rotation based on the probability:
$a \leftarrow generateFirstIdv(p)$;
5) generate the second individual using the Grover's algorithm with the specific oracle:
$c \leftarrow convertBinary(curBestIdv.fitness + 1)$;
$circuit \leftarrow initialize(p)$;
for $i \leftarrow 1$ *to* t **do**
  $circuit \leftarrow feasibleSolution(circuit)$;
  $circuit \leftarrow qAdder(circuit)$;
  $circuit \leftarrow qComparator(c, circuit)$;
  $circuit \leftarrow inverseQComparator(c, circuit)$;
  $circuit \leftarrow inveseQAdder(circuit)$;
  $circuit \leftarrow inverseFeasibleSol(circuit)$;
  $circuit \leftarrow diffuser(circuit)$;
end
$b \leftarrow measure(circuit)$;
6) let them compete:
$winner, loser \leftarrow compete(a, b)$;
7) check if the Grover iteration is highest:
if $curBestIdv.fitness > b.fitness$ **then**
  $t \leftarrow t + 1$;
end
8) update probability vector towards winner:
for $i \leftarrow 1$ *to* l **do**
  if $winner[i] \neq loser[i]$ **then**
    if $winner[i] \neq 1$ **then**
      $p[i] \leftarrow p[i] + 1/n$;
    else
      $p[i] \leftarrow p[i] - 1/n$;
    end
  end
end
9) update the current best individual:
if $curBestIdv.fitness < winner.fitness$ **then**
  $curBestIdv \leftarrow winner$;
end
10) check if the vector has converged:
for $i \leftarrow 1$ *to* l **do**
  if $p[i] > 0$ *and* $p[i] < 1$ **then**
    return to step 3;
  end
end

ภาพที่ 53 โค้ดเทียมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (เวอร์ชันปรับปรุง)



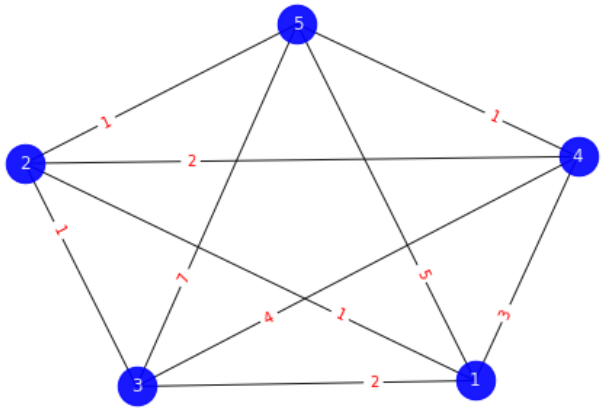
ภาพที่ 54 แผนผังแสดงขั้นตอนการทำงานของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (เวอร์ชันปรับปรุงฟังก์ชันโอราเคิล) กรอบเส้นประคือประมวลผลบนเครื่อง IBM QASM simulator

บทที่ 8

ผลการทดสอบขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง

8.1 การจัดเตรียมสภาพแวดล้อมสำหรับการทดลองเปรียบเทียบประสิทธิภาพระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับชนิดควอนตัม (เวอร์ชันปรับปรุง)

ผู้วิจัยได้กำหนดขนาดของประชากร (Population size) สูงสุดสำหรับปัญหา TSP 3 เมือง คือ 50 ขนาดของประชากรสูงสุดสำหรับปัญหา TSP 4 เมือง คือ 100 และขนาดของประชากรสูงสุดสำหรับปัญหา TSP 5 เมือง คือ 300 โดยเริ่มต้นที่ขนาดประชากรเท่ากับ 4 และค่อยๆเพิ่มขนาดประชากรทีละ 2 จนถึงขนาดประชากรสูงสุด เนื่องด้วยผลลัพธ์การทำงานของวงจรควอนตัมคือค่าความน่าจะเป็น ทำให้การวัดแต่ละครั้งอาจได้ผลลัพธ์แตกต่างกัน จึงจำเป็นต้องมีการกำหนดจำนวนช็อต ซึ่งก็คือจำนวนครั้งที่วงจรควอนตัมจะถูกทำงานซ้ำ เพื่อให้ได้การกระจายตัวของคำตอบที่มีความแม่นยำมากยิ่งขึ้น ดังนั้นผู้วิจัยจึงได้กำหนดจำนวนช็อตสำหรับขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง สำหรับแก้ปัญหา TSP ขนาด 3 เมือง คือ 1 ช็อต จำนวนช็อตสำหรับขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง สำหรับแก้ปัญหา TSP ขนาด 4 เมือง คือ 1 ช็อต 10 ช็อต และ 20 ช็อต และจำนวนช็อตสำหรับขั้นตอนวิธิเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง สำหรับแก้ปัญหา TSP ขนาด 5 เมือง คือ 1,024 ช็อต 2,000 ช็อต 3,000 ช็อต และ 4,000 ช็อต จำนวนช็อตที่ผู้วิจัยใช้ในการทดลองมาจากการทดลองเริ่มจากใช้จำนวนช็อตน้อยๆ และค่อยๆเพิ่มจนได้การกระจายตัวของคำตอบที่มีความแม่นยำที่เหมาะสมสำหรับแต่ละขนาดของปัญหา TSP ซึ่งผู้วิจัยค้นพบว่าสำหรับปัญหา TSP ขนาด 3 เมือง จำนวนช็อตเพียง 1 ช็อตก็เพียงพอสำหรับการกระจายตัวของคำตอบที่มีความแม่นยำ แต่สำหรับปัญหา TSP ขนาด 4 เมือง และ 5 เมือง การเพิ่มจำนวนช็อตมีผลต่อการการกระจายตัวของคำตอบให้มีความแม่นยำคำตอบมีความน่าเชื่อถือ โดยเฉพาะอย่างยิ่ง ปัญหา TSP ขนาด 5 เมือง ซึ่งมีรูปแบบคำตอบที่เป็นไปได้ในพื้นที่การค้นหาทั้งหมด 2^{16} รูปแบบ เนื่องจากใช้จำนวนคิวบิตทั้งหมด 16 คิวบิตในการแทนรูปแบบคำตอบ ทำให้ผู้วิจัยจำเป็นต้องเริ่มต้นใช้จำนวนช็อตตั้งแต่ 1,024 ช็อต เป็นต้นไป เพื่อให้การกระจายตัวของคำตอบมีความน่าเชื่อถือเพียงพอที่จะนำคำตอบนั้นมาใช้ ตัวอย่างปัญหา TSP ขนาด 3 เมือง และ 4 เมือง ที่นำมาใช้ในการทดลองเป็นไปดังภาพที่ 43 และ 44 ตามลำดับ สำหรับตัวอย่างปัญหา TSP ขนาด 5 เมือง แสดงดังภาพที่ 55



ภาพที่ 55 ปัญหา TSP ขนาด 5 เมือง ที่ใช้เป็นต้นแบบในการทดลองสำหรับงานวิจัยนี้

สำหรับงานวิจัยนี้ผู้วิจัยจะขอเรียกวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุงนี้ว่า “new Grover-assisted cGA*” เนื่องจากฟังก์ชันโอราเคิลของวิธี new Grover-assisted cGA* ไม่สามารถระบุจำนวนคำตอบที่เป็นผลลัพธ์ของฟังก์ชันโอราเคิลได้อย่างแน่นอน เพราะเราไม่สามารถรู้ได้ว่ามีจำนวนคำตอบกี่จำนวนที่น้อยกว่าคำตอบที่ดีที่สุด ณ ปัจจุบันในพื้นที่การค้นหาของโกรเวอร์ ดังนั้นผู้วิจัยจึงได้กำหนดวิธีการปรับจำนวนรอบของโกรเวอร์ เป็นแบบ adaptive โดยกำหนดจำนวนรอบของโกรเวอร์ เริ่มต้นที่ 1 เสมอ และจำนวนรอบของโกรเวอร์ จะถูกปรับเพิ่มทีละ 1 เมื่อทำการวัดคำตอบออกมาแล้วคำตอบแย่กว่าคำตอบที่ดีที่สุด ณ ปัจจุบัน ซึ่งจำนวนรอบของโกรเวอร์ที่เพิ่มขึ้นจะถูกนำไปใช้ในการกำหนดจำนวนรอบของโกรเวอร์สำหรับการสร้าง second candidate ในรอบถัดไป ในทางกลับกันถ้าคำตอบที่วัดได้ดีกว่าหรือเท่ากับรูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน ก็จะไม่เพิ่มจำนวนรอบของโกรเวอร์ ทำให้การสร้าง second candidate ในรอบถัดไป จะใช้จำนวนรอบของโกรเวอร์เท่าเดิม

ผู้วิจัยได้ทำการรันขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (cGA*) และขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชั่นปรับปรุง (new Grover-assisted cGA*) ทั้งหมด 25 ครั้ง เพื่อหาค่าเฉลี่ยและนำมาพล็อตในกราฟ และใช้การคัดเลือกโครโมโซมแบบการแข่งขัน (tournament selection) โดยกำหนด tournament selection เท่ากับ 2 ขั้นตอนวิธีเชิงพันธุกรรมทั้ง 2 แบบจะรันจนกว่าค่าความน่าจะเป็นในแต่ละมิติของเวกเตอร์ความน่าจะเป็นลู่เข้าค่า 0 หรือ 1

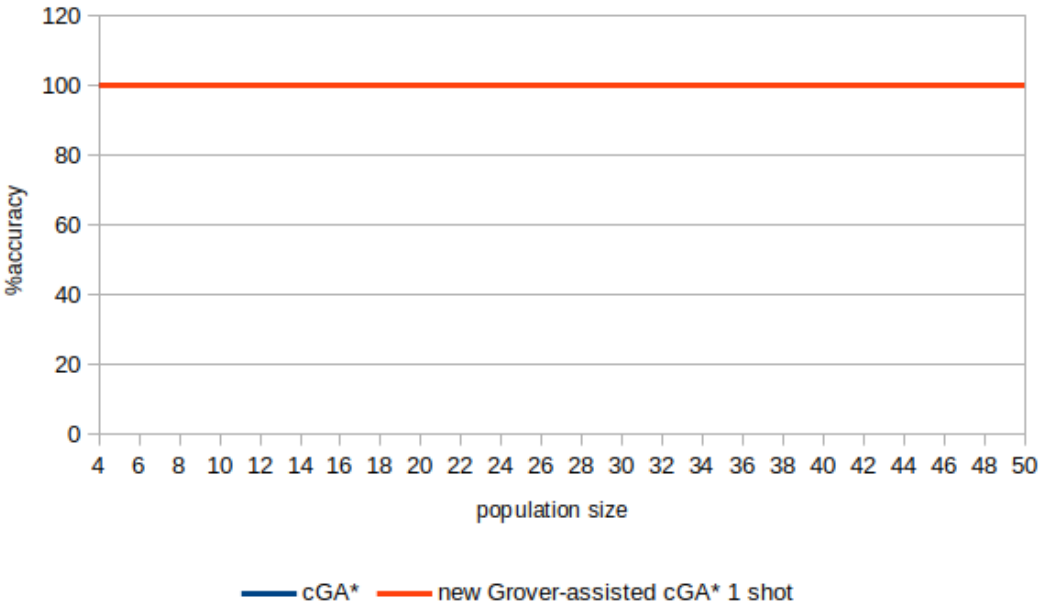
8.2 ผลการทดลองเปรียบเทียบประสิทธิภาพระหว่างขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมกับชนิดควอนตัมเวอร์ชันปรับปรุง

การเปรียบเทียบประสิทธิภาพในการประมวลผลระหว่าง classical cGA* และ new Grover-assisted cGA แบ่งเป็น 2 ด้าน ได้แก่ ความถูกต้องของคำตอบ และจำนวนครั้งในการประเมินค่าความเหมาะสม (function evaluation) โดยวิธีการคำนวณจำนวน function evaluation ต่อหนึ่งรอบ generation สำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง (new Grover-assisted cGA*) เป็นดังสมการที่ (34)

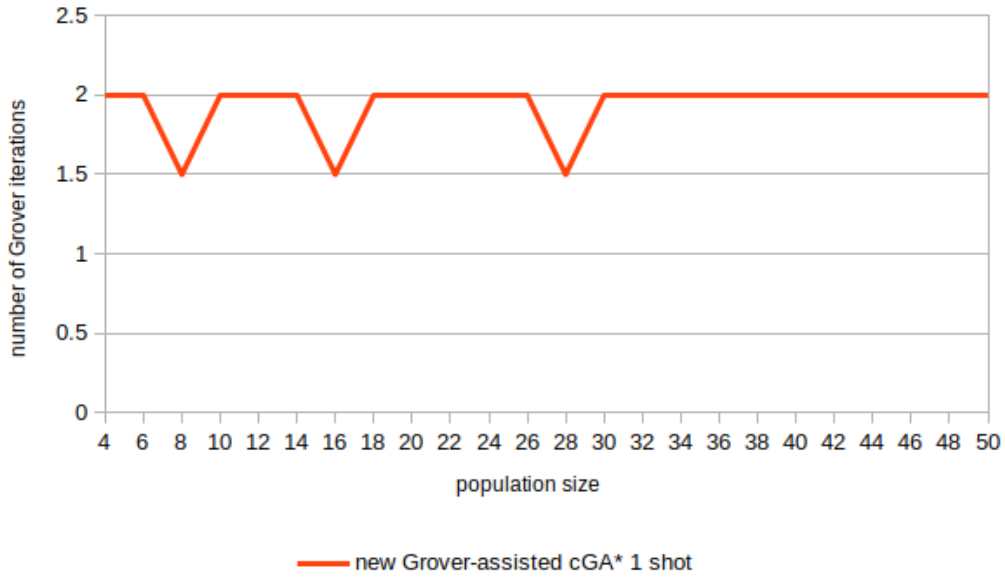
8.2.1. <u>การวิเคราะห์ผลการทดลอง</u>

ผลการทดลองแก้ปัญหา TSP ขนาด 3 เมืองด้วยขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง (new Grover-assisted cGA*) เปรียบเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิม (cGA*) ในด้านความถูกต้องของคำตอบหน่วยเป็นเปอร์เซ็นต์ เป็นดังภาพที่ 56 จากกราฟในภาพที่ 56 แสดงให้เห็นว่าทั้งสองอัลกอริทึมมีประสิทธิภาพเทียบเท่ากันในแง่ความถูกต้องของคำตอบ โดยสามารถหาคำตอบที่เหมาะสมที่สุดได้ตั้งแต่ขนาดประชากรเท่ากับ 4 เนื่องด้วย new Grover-assisted cGA* ใช้วิธีการกำหนดจำนวนรอบของโกรเวอร์แบบ adaptive โดยเริ่มต้น generation ที่จำนวนรอบของโกรเวอร์เท่ากับ 1 ผู้วิจัยจึงทำการพล็อตค่าเฉลี่ยจำนวนรอบของโกรเวอร์ที่ใช้ในแต่ละขนาดของประชากร (population size) ซึ่งเป็นค่าเฉลี่ยจากการรันทั้งหมด 25 ครั้งดังที่กล่าวไปข้างต้น รายละเอียดดังภาพที่ 57 จะเห็นได้ว่าช่วงขนาดประชากรที่ 4 ถึง 30 จำนวนรอบของโกรเวอร์ที่ใช้จะอยู่ระหว่าง 1 ถึง 2 รอบ และตั้งแต่ขนาดประชากรมากกว่า 30 เป็นต้นไป จำนวนรอบของโกรเวอร์ที่ใช้จะคงที่อยู่ที่ 2 รอบ เมื่อพิจารณาจำนวนครั้งในการประเมินค่าความเหมาะสมจากภาพที่ 58 new Grover-assisted cGA* ใช้จำนวนครั้งในการประเมินค่าความเหมาะสมของคำตอบน้อยกว่า cGA* ประมาณ 12% เทียบกับ cGA* เนื่องจากปัญหา TSP ขนาด 3 เมือง ใช้จำนวนคิวบิตเพียง 4 คิวบิต แทนค่าสถานะของคิวบิตทั้งหมดที่เป็นไปได้ 16 สถานะควอนตัม ประกอบกับฟังก์ชันโอราเคิลสามารถเปรียบเทียบสถานะควอนตัมทั้ง 16 สถานะพร้อมกัน และเลือกเฉพาะสถานะควอนตัมที่ให้คำตอบที่ระยะทางไม่ยาวกว่าระยะทางที่สั้นที่สุด ณ ปัจจุบัน ซึ่งใช้จำนวนรอบของโกรเวอร์ โดยเฉลี่ย

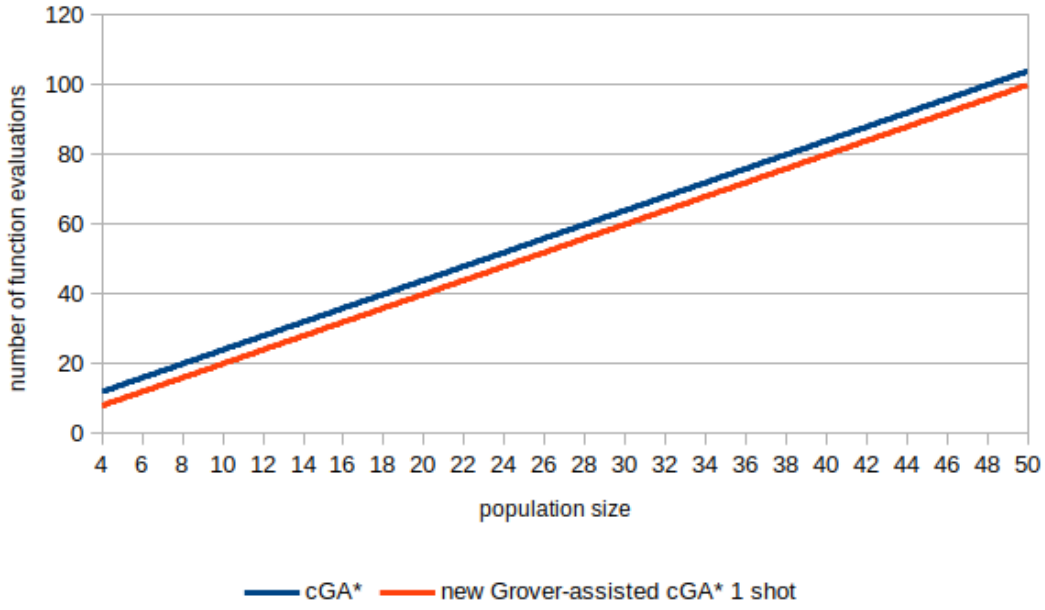
ประมาณ 2 รอบ และใช้จำนวนช็อตเพียง 1 ช็อต ดังภาพที่ 57 ก็เพียงพอที่จะทำให้การกระจายตัวของคำตอบมีความแม่นยำ และสามารถค้นหาคำตอบที่เหมาะสมที่สุดได้ จำนวนครั้งในการประเมินค่าความเหมาะสมของ new Grover-assisted cGA* ที่เพิ่มขึ้นตามจำนวนช็อต และจำนวนรอบของโกรเวอร์ที่ใช้ในแต่ละช็อตจึงมีค่าน้อยกว่าจำนวนครั้งในการประเมินค่าความเหมาะสมของ cGA* ดังภาพที่ 58 ดังนั้นสำหรับปัญหา TSP ขนาด 3 เมือง new Grover-assisted cGA* ที่ใช้ขนาดประชากรเท่ากับ 4 ก็เพียงพอในการหาคำตอบ ซึ่งวิธีการดังกล่าวยังใช้จำนวนครั้งในการประเมินค่าความเหมาะสมน้อยกว่า cGA* ด้วย



ภาพที่ 56 กราฟแสดงผลการเปรียบเทียบความถูกต้องของคำตอบ (จำนวนบิตที่ถูกต้องเป็นเปอร์เซ็นต์เมื่อสิ้นสุดการทำงาน) ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 3 เมือง*



ภาพที่ 57 กราฟแสดงผลจำนวนรอบของโกรเวอร์ที่ใช้ของ new Grover-assisted cGA จำนวน 1 ช็อต จนกระทั่งสิ้นสุดการทำงานในแต่ละขนาดของประชากร สำหรับปัญหา TSP ขนาด 3 เมือง*



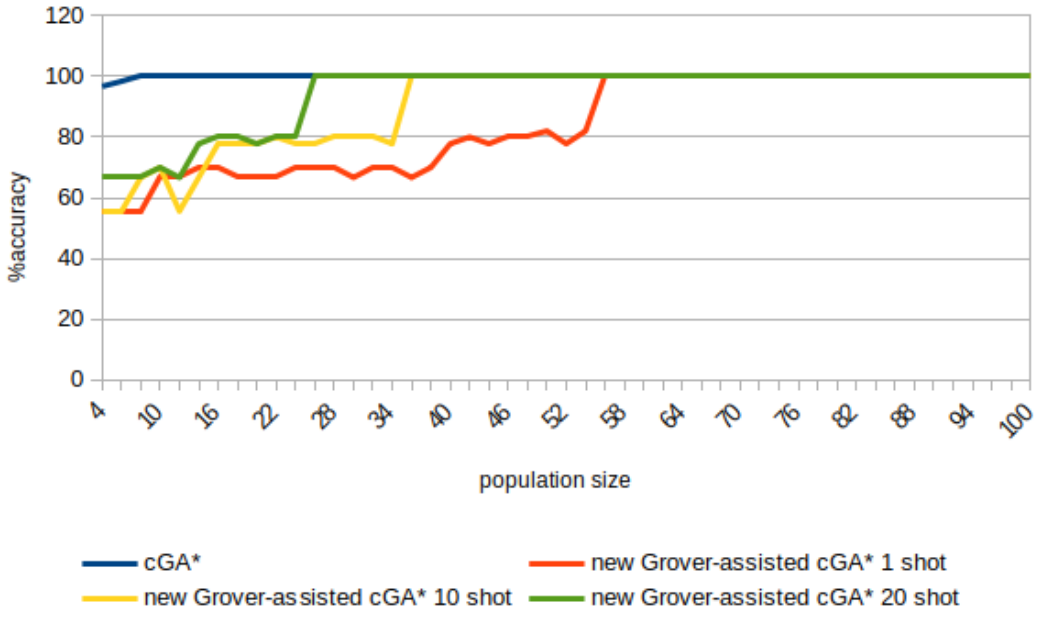
ภาพที่ 58 กราฟแสดงผลการเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสมจนกระทั่งสิ้นสุดการทำงาน ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 3 เมือง*

สำหรับปัญหา TSP ขนาด 4 เมือง จากภาพที่ 59 cGA* สามารถหาคำตอบที่เหมาะสมที่สุดโดยใช้ขนาดของประชากรเพียง 8 ในขณะที่ new Grover-assisted cGA* ที่ใช้จำนวนช็อต 1 ช็อต จะเจอคำตอบที่เหมาะสมที่สุดที่ขนาดของประชากรประมาณ 56 และใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยคือ 7.2 รอบ ดังภาพที่ 60 เมื่อเพิ่มจำนวนช็อตเป็น 10 ช็อต new Grover-assisted cGA* จะเจอคำตอบที่เหมาะสมที่สุดที่ขนาดของประชากร 36 โดยใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยคือ 4.5 รอบ สุดท้ายเมื่อเพิ่มจำนวนช็อตเป็น 20 ช็อต new Grover-assisted cGA* จะเจอคำตอบที่เหมาะสมที่สุดที่ขนาดของประชากร 26 โดยใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยคือ 5 รอบ จะเห็นได้ว่า new Grover-assisted cGA* ใช้ขนาดประชากรที่มากกว่า cGA* สาเหตุหนึ่งมาจากการเข้ารหัสปัญหา TSP ของทั้งสองขั้นตอนวิธีแตกต่างกัน โดย cGA* ใช้จำนวนบิตทั้งหมดในการแทนรูปแบบคำตอบคือ 6 บิต ในขณะที่ new Grover-assisted cGA* ใช้จำนวนคิวบิตทั้งหมด 9 คิวบิตในการแทนรูปแบบคำตอบ ดังนั้นพื้นที่การค้นหาคำตอบของ cGA* จึงน้อยกว่า new Grover-assisted cGA* ทำให้ใช้ขนาดประชากรที่น้อยกว่าในการหาคำตอบที่เหมาะสมที่สุด นอกจากนี้เมื่อพิจารณากราฟที่ 58 ดูเหมือนว่าการเพิ่มจำนวนช็อตให้กับ new Grover-assisted cGA* ทำให้เจอคำตอบได้เร็วขึ้นในขนาดประชากรที่น้อยลง แต่การเพิ่มจำนวนช็อตไม่ได้มีผลต่อการเพิ่มประสิทธิภาพของอัลกอริทึมโดยตรง แต่ช่วยทำให้การกระจายตัวของคำตอบมีความแม่นยำมากขึ้น และช่วยลดความผิดพลาดจากการอ่านคำตอบจากสถานะซ้อนทับของสถานะควอนตัม new Grover-assisted cGA* ที่ใช้จำนวนช็อต 1 ช็อต มีความเป็นไปได้สูงที่จะเกิดความผิดพลาดจากการอ่านคำตอบจากสถานะซ้อนทับของสถานะควอนตัม เนื่องจากทำการรันอัลกอริทึมเพียง 1 ครั้งและอ่านคำตอบทันที

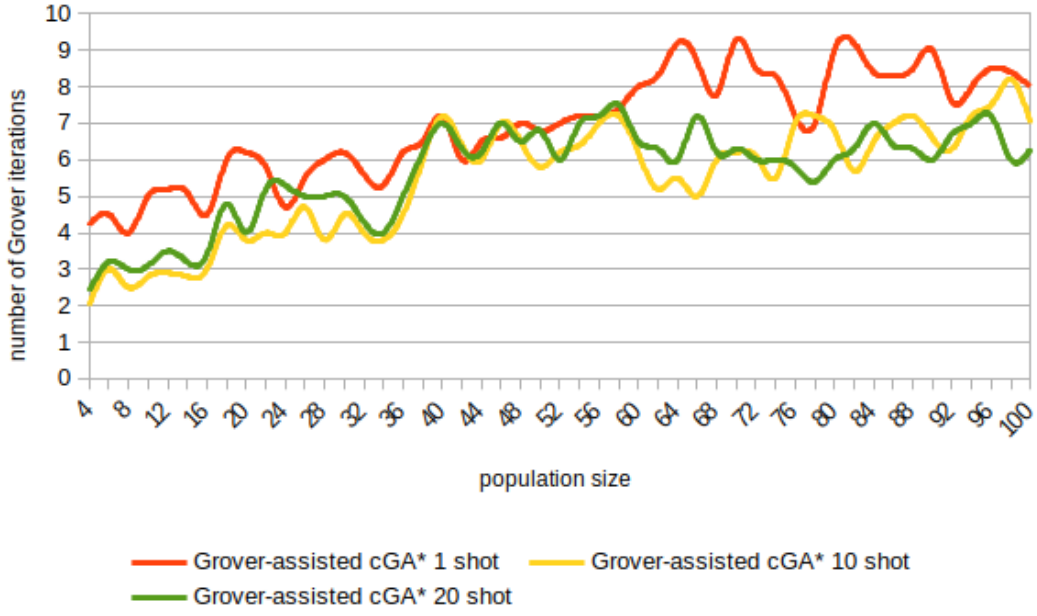
เมื่อพิจารณาจำนวนรอบของโกรเวอร์ที่ใช้จากกราฟที่ 59 พบว่าการใช้จำนวนช็อตน้อยๆเพียง 1 ช็อต ทำให้จำนวนรอบของโกรเวอร์ที่ใช้มากกว่าการใช้จำนวนช็อต 10 ช็อต และ 20 ช็อต เนื่องจากการเพิ่มจำนวนช็อตมีผลต่อการเพิ่มความน่าเชื่อถือของคำตอบในกรณีที่รูปแบบคำตอบในพื้นที่การค้นหาค่อนข้างมาก เช่นกรณี 4 เมือง รูปแบบคำตอบทั้งหมดคือ 2^9 รูปแบบ ซึ่งเพิ่มจาก 3 เมืองค่อนข้างมาก ดังนั้นการวัดค่าคำตอบในแต่ละรอบสำหรับกรณีที่ใช้จำนวนช็อตเพียง 1 ช็อต อาจเจอคำตอบที่ไม่ถูกต้องในบางครั้ง ทำให้

จำนวนรอบของโกรเวอร์ที่ต้องใช้ใน generation ถัดไปเพิ่มสูงขึ้น จึงทำให้ค่าเฉลี่ยของจำนวนรอบของโกรเวอร์ที่ใช้กรณี 1 ช็อต มากกว่า 10 ช็อต และ 20 ช็อต เกือบเท่าตัว ส่วนจำนวนรอบของโกรเวอร์ที่ใช้สำหรับกรณี 10 ช็อต และ 20 ช็อตค่อนข้างใกล้เคียงกัน นั่นเป็นเพราะจำนวนช็อต 10 ช็อตก็เพียงพอสำหรับการกระจายตัวของคำตอบที่มีความแม่นยำ การเพิ่มจำนวนช็อตที่เกินกว่า 10 ช็อต จึงให้ผลลัพธ์ที่ไม่แตกต่างกันอย่างมีนัยสำคัญ นี่เป็นการสะท้อนให้เห็นว่าถึงแม้จะใช้จำนวนช็อตเพียง 1 ช็อต แต่ถ้าจำนวนรอบของโกรเวอร์ มีค่ามากพอที่จะช่วยเพิ่มความน่าจะเป็นที่วัดสถานะของคิวบิตแล้วได้คำตอบที่สนใจมากยิ่งขึ้น ก็สามารถหาคำตอบที่เหมาะสมที่สุดได้เช่นกัน ดังนั้นการเพิ่มจำนวนรอบของโกรเวอร์จึงมีผลต่อการเพิ่มประสิทธิภาพในการหาคำตอบของอัลกอริทึมที่นำเสนอให้ดีขึ้น

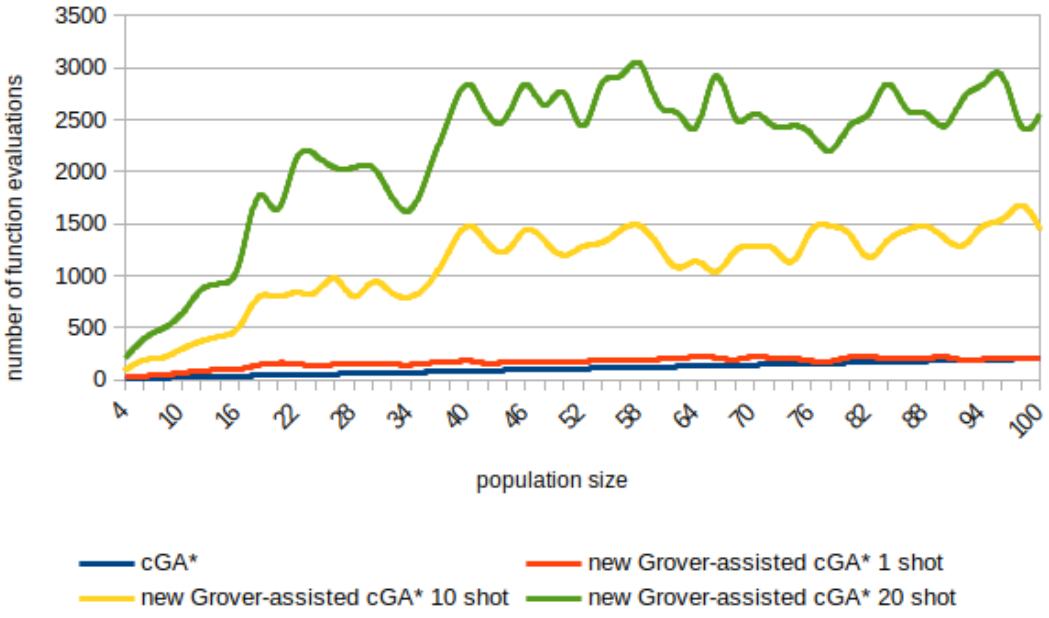
เมื่อพิจารณาจำนวนครั้งในการประเมินค่าความเหมาะสมของทั้งสองขั้นตอนวิธี จากภาพที่ 61 สำหรับ new Grover-assisted cGA* ที่ใช้จำนวนช็อต 1 ช็อต จะมีจำนวนครั้งในการประเมินค่าความเหมาะสมที่ค่อนข้างใกล้เคียงกับ cGA* ถึงแม้ว่าจะเจอคำตอบที่เหมาะสมที่ขนาดประชากรสูงกว่า นั่นคือ new Grover-assisted cGA* ต้องการความละเอียดในการค้นหาที่มากกว่า cGA* แต่จำนวนครั้งที่ใช้ในการประเมินค่าความเหมาะสมค่อนข้างใกล้เคียงกัน ส่วน new Grover-assisted cGA* 10 ช็อต และ 20 ช็อตนั้นใช้จำนวนครั้งในการประเมินค่าความเหมาะสมมากกว่า 1 ช็อตค่อนข้างมาก เนื่องจากจำนวนครั้งในการประเมินค่าความเหมาะสมของ new Grover-assisted cGA* เพิ่มขึ้นตามจำนวนช็อต และจำนวนรอบของโกรเวอร์ที่ใช้ ซึ่งเป็นไปตามสมการ (34) ดังนั้นวิธี new Grover-assisted cGA* ที่ใช้จำนวนช็อต 1 ช็อต จึงเป็นวิธีที่เหมาะสมที่สุดเมื่อเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมด้วยกันเนื่องจากสามารถหาคำตอบที่เหมาะสมที่สุดได้โดยใช้ขนาดประชากรที่ไม่มาก และใช้จำนวนครั้งในการประเมินค่าความเหมาะสมน้อยที่สุด



ภาพที่ 59 กราฟแสดงผลการเปรียบเทียบความถูกต้องของคำตอบ (จำนวนบิตที่ถูกต้องเป็นเปอร์เซ็นต์เมื่อสิ้นสุดการทำงาน) ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 4 เมือง*



ภาพที่ 60 กราฟแสดงผลการเปรียบเทียบจำนวนรอบของโกรเวอร์ที่ใช้ของ new Grover-assisted cGA จำนวน 1 ช็อต 10 ช็อต และ 20 ช็อต จนกระทั่งสิ้นสุดการทำงานในแต่ละขนาดของประชากร สำหรับปัญหา TSP ขนาด 4 เมือง*



ภาพที่ 61 กราฟแสดงผลการเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสมจนกระทั่งสิ้นสุดการทำงาน ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 4 เมือง*

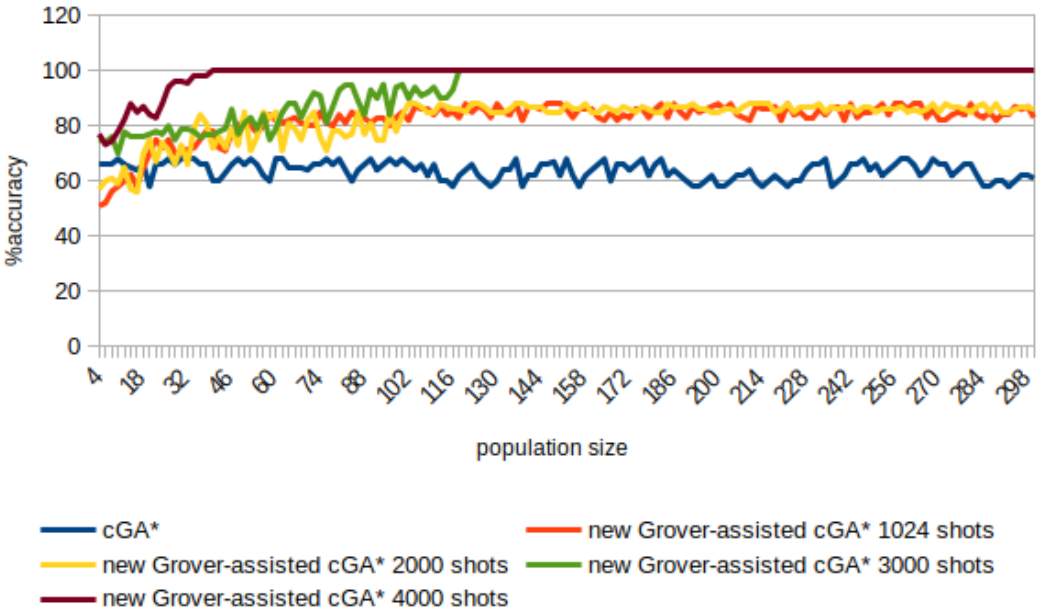
เมื่อพิจารณาผลการทดลองสำหรับปัญหา TSP ขนาด 5 เมือง ดังภาพที่ 62 cGA* ไม่สามารถหาคำตอบที่เหมาะสมที่สุดได้ ซึ่งเปอร์เซ็นต์ความถูกต้องของบิตคำตอบอยู่ที่ 58-62 เปอร์เซ็นต์โดยประมาณ แม้ว่าจะเพิ่มขนาดประชากรสูงสุดเป็น 300 ในขณะที่ new Grover-assisted cGA* ที่ใช้จำนวนช็อต 3,000 ช็อต และ 4,000 ช็อต เจอคำตอบที่เหมาะสมที่สุดที่ขนาดของประชากร 118 และ 40 ตามลำดับ โดย new Grover-assisted cGA* 3,000 ช็อต ใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยคือ 4.2 รอบ และ new Grover-assisted cGA* 4,000 ช็อต ใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยคือ 3 รอบ จะเห็นได้ว่าการเพิ่มจำนวนช็อตมากขึ้นอีก 1,000 ช็อต ทำให้การกระจายตัวของคำตอบมีความแม่นยำมากยิ่งขึ้น และช่วยลดข้อผิดพลาดในการอ่านคำตอบจากสภาวะซ้อนทับของสถานะควอนตัม ทำให้สามารถหาคำตอบโดยใช้ขนาดของประชากรเพียงแค่ 40 อย่างไรก็ตามเมื่อพิจารณาจำนวนรอบของโกรเวอร์สำหรับ new Grover-assisted cGA* 3,000 ช็อต และ 4,000 ช็อต ดังภาพที่ 63 จะเห็นว่าจำนวนรอบของโกรเวอร์ไม่ต่างกันมาก เนื่องจากจำนวนช็อตที่ใช้ค่อนข้างมากทั้งคู่ และเนื่องด้วยพื้นที่ในการค้นหาข้อมูลของฟังก์ชันโอราเคิลเป็นพื้นที่

ขนาดใหญ่เพราะมีรูปแบบคำตอบทั้งหมด 2^{16} รูปแบบ (เข้ารหัสปัญหาด้วยจำนวนคิวบิต 16 คิวบิต) ซึ่งจำนวนคิวบิตที่มากขึ้นส่งผลกับเปอร์เซ็นต์ความผิดพลาดในการอ่านค่าคิวบิตที่มากขึ้นด้วย ผู้วิจัยจึงทำการทดลองเพื่อหาจำนวนช็อตที่เหมาะสมที่ควรนำมาใช้เพื่อลดผลกระทบจากข้อผิดพลาดดังกล่าว และเพิ่มความแม่นยำในการกระจายตัวของคำตอบ จำนวนช็อตเริ่มต้นที่ใช้จึงเป็น 1,024 ช็อต ผู้วิจัยพบว่า new Grover-assisted cGA* 1,024 ช็อต และ 2,000 ช็อต ยังไม่สามารถหาคำตอบที่เหมาะสมที่สุดได้ แม้ว่าจะใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยมากถึง 7-8 ครั้ง ดังภาพที่ 63 ในขณะที่ new Grover-assisted cGA* 3,000 ช็อต และ 4,000 ช็อต สามารถหาคำตอบที่เหมาะสมที่สุดได้ โดยใช้จำนวนรอบของโกรเวอร์น้อยกว่าครึ่งหนึ่ง นี่จึงแสดงให้เห็นว่าจำนวนคิวบิตที่มากขึ้นส่งผลกระทบต่อประสิทธิภาพการทำงานของอัลกอริทึมเป็นอย่างมาก เนื่องจากความผิดพลาดต่างๆที่เกิดขึ้นจากการดำเนินการระหว่างคิวบิตที่เพิ่มมากขึ้น และความผิดพลาดจากการอ่านค่าสถานะคิวบิตแต่ละตัว ดังนั้นการใช้จำนวนช็อตที่เหมาะสมในการรันอัลกอริทึมเพื่อหาคำตอบก็เป็นสิ่งที่สำคัญเช่นกัน แม้ว่าจำนวนช็อตไม่ได้มีผลในการเพิ่มประสิทธิภาพการทำงานของอัลกอริทึม แต่ช่วยทำให้ประสิทธิภาพการทำงานของอัลกอริทึมไม่ลดลงเนื่องจากข้อผิดพลาดดังกล่าว

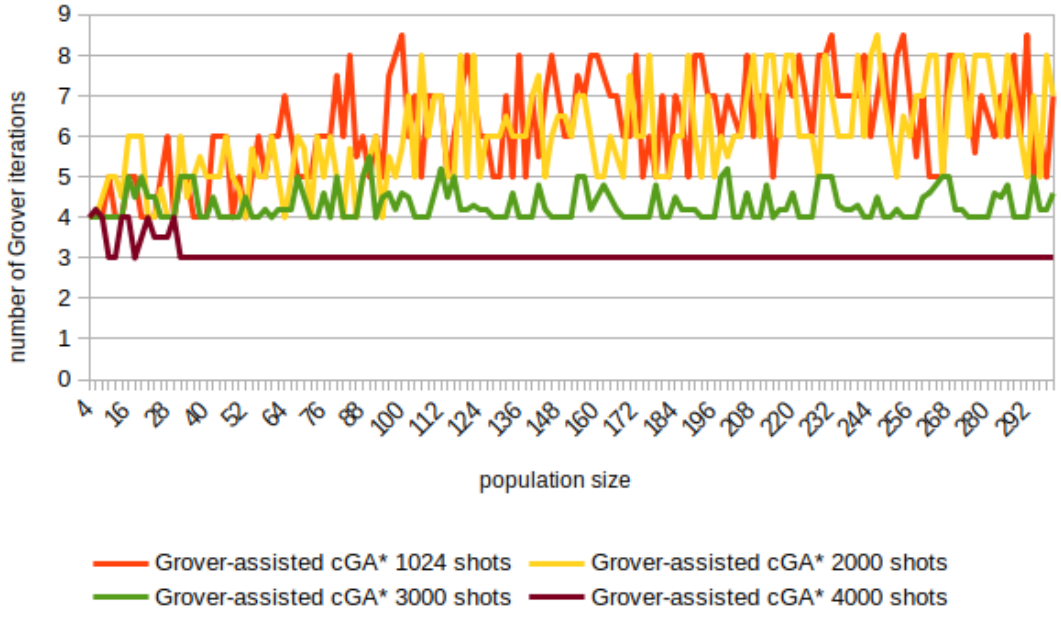
เนื่องด้วยจำนวนช็อตที่ต้องใช้ในการรันอัลกอริทึมมากขึ้นจากเหตุผลดังกล่าวข้างต้น ทำให้จำนวนครั้งในการประเมินค่าความเหมาะสมเพิ่มขึ้นตามจำนวนช็อตที่ใช้ และจำนวนรอบของโกรเวอร์ ดังสมการ (34) จากภาพที่ 64 จะเห็นได้ว่า new Grover-assisted cGA* 4,000 ช็อต new Grover-assisted cGA* 3,000 ช็อต และ new Grover-assisted cGA* 2,000 ช็อต ใช้จำนวนครั้งในการประเมินค่าความเหมาะสมค่อนข้างใกล้เคียงกัน เนื่องจาก new Grover-assisted cGA* 4,000 ช็อต แม้ใช้จำนวนช็อตมากกว่า แต่ใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยน้อยกว่า new Grover-assisted cGA* 2,000 ช็อต และ 3,000 ช็อต ในขณะที่ new Grover-assisted cGA* 2,000 ช็อต และ 3,000 ใช้จำนวนช็อตน้อยกว่า แต่ใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยมากกว่า 4,000 ช็อต และ new Grover-assisted cGA* 2,000 ช็อต ก็ใช้จำนวนรอบของโกรเวอร์โดยเฉลี่ยมากกว่า 3,000 ช็อต เช่นกัน ทั้งสามขั้นตอนวิธีจึงใช้จำนวนครั้งในการประเมินค่าความเหมาะสมค่อนข้างใกล้เคียงกัน ส่วน

cGA* ใช้จำนวนครั้งในการประเมินค่าความเหมาะสมน้อยที่สุด แต่เปอร์เซ็นต์ความถูกต้องของคำตอบอยู่ระหว่าง 58 – 62 เปอร์เซ็นต์

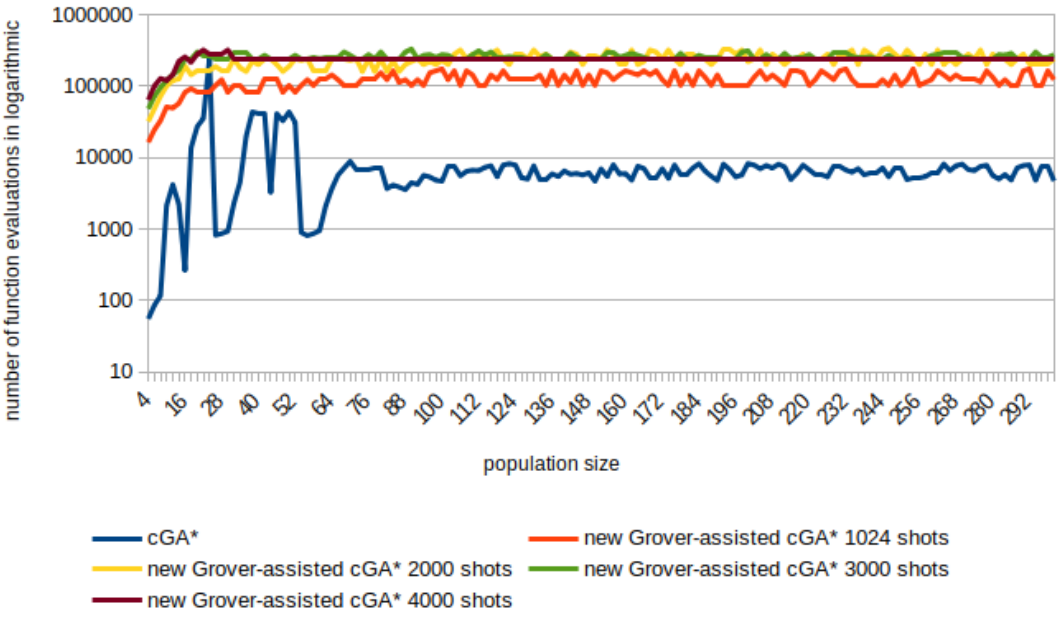
ถ้าพิจารณาความถูกต้องของคำตอบที่ 80 เปอร์เซ็นต์ จากกราฟที่ 61 พบว่าขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมตั้งแต่ 1,024 ช็อตขึ้นไปสามารถหาคำตอบที่ถูกต้อง 80 เปอร์เซ็นต์ได้ โดย new Grover-assisted cGA* 1,024 ช็อต ใช้ขนาดประชากรประมาณ 70 ซึ่งมากกว่า new Grover-assisted cGA* 2,000 ช็อต 3,000 ช็อต และ 4,000 ช็อต ซึ่งใช้ขนาดประชากรประมาณ 46, 32 และ 48 ตามลำดับ แต่ new Grover-assisted cGA* 1,024 ช็อต ใช้จำนวนครั้งที่ในการประเมินค่าความเหมาะสมน้อยที่สุด เมื่อเพิ่มความถูกต้องของคำตอบเป็น 90 เปอร์เซ็นต์ จะต้องใช้จำนวนช็อต ตั้งแต่ 3,000 ช็อตขึ้นไป ดังนั้นถ้าต้องการคำตอบที่ถูกต้องไม่น้อยกว่า 80 เปอร์เซ็นต์ new Grover-assisted cGA* 1,024 ช็อต ก็เพียงพอสำหรับการหาคำตอบ โดยใช้จำนวนครั้งในการประเมินค่าความเหมาะสมน้อยที่สุดเมื่อเทียบกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมด้วยกัน



ภาพที่ 62 กราฟแสดงผลการเปรียบเทียบความถูกต้องของคำตอบ (จำนวนบิตที่ถูกต้องเป็นเปอร์เซ็นต์เมื่อสิ้นสุดการทำงาน) ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 5 เมือง*



ภาพที่ 63 กราฟแสดงผลการเปรียบเทียบจำนวนรอบของโกรเวอร์ที่ใช้ของ new Grover-assisted cGA จำนวน 1,024 ช็อต 2,000 ช็อต 3,000 ช็อต และ 4,000 ช็อต จนกระทั่งสิ้นสุดการทำงานในแต่ละขนาดของประชากร สำหรับปัญหา TSP ขนาด 5 เมือง*



ภาพที่ 64 กราฟแสดงผลการเปรียบเทียบจำนวนครั้งในการประเมินค่าความเหมาะสมจนกระทั่งสิ้นสุดการทำงาน ระหว่าง cGA และ new Grover-assisted cGA* สำหรับปัญหา TSP ขนาด 5 เมือง*

8.2.2. <u>การวิเคราะห์ความซับซ้อนในเชิงควอนตัม</u>

จะใช้หลักเกณฑ์เดียวกันในข้อ 6.2.2 สำหรับการวิเคราะห์ความซับซ้อนในเชิงควอนตัมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง (new Grover-assisted cGA*) ได้แก่ 1: การวิเคราะห์จำนวนคิวบิต และจำนวนของคิวบิตทด (Ancilla qubits) ที่ต้องการที่สอดคล้องกับขนาดของปัญหาการหาค่าเหมาะสม และ 2: ต้นทุนของวงจรควอนตัมจากจำนวนเกต CNOT ที่ใช้ กับความลึกของวงจร (Circuit depth) ตามจำนวนรอบของโกรเวอร์ที่ใช้ใน new Grover-assisted cGA*

8.2.2.1. จำนวนคิวบิตและคิวบิตทดที่ต้องใช้

ในเบื้องต้นจำนวนคิวบิตที่จำเป็นต้องใช้จะเป็นไปตามที่ระบุในหัวข้อ 6.2.2.1 แต่จำนวนคิวบิตทดที่ต้องใช้จะเพิ่มเติมขึ้นมาจากหัวข้อ 6.2.2.1 เนื่องจากฟังก์ชันโอราเคิลของ new Grover-assisted cGA* มีการปรับปรุงให้สามารถคำนวณระยะทางของแต่ละรูปแบบคำตอบ และทำการเปรียบเทียบกับระยะทางที่สั้นที่สุด ณ ปัจจุบันได้ ซึ่งจำเป็นต้องใช้จำนวนคิวบิตทดเพิ่มสำหรับการดำเนินการดังกล่าวข้างต้น โดยในส่วนของการเปรียบเทียบระยะทางจำเป็นต้องเพิ่มจำนวนคิวบิตทดขึ้นมาเท่ากับระยะทางรวมสูงสุดที่เป็นไปได้ของการเดินทางครบทุกเมืองในรูปแบบเลขฐานสอง เช่น ถ้าระยะทางรวมสูงสุดของการเดินทางคือ 25 ซึ่งแปลงเป็นเลขฐานสองได้ 11001 ดังนั้นจำเป็นต้องใช้คิวบิตทดเพิ่มอีก 5 ตัว สำหรับเก็บผลรวมระยะทางที่สั้นที่สุด ณ ปัจจุบัน เพื่อสามารถนำมาเปรียบเทียบกับผลรวมระยะทางของแต่ละรูปแบบคำตอบในฟังก์ชันโอราเคิลได้ ดังนั้นจำนวนคิวบิตทดทั้งหมดที่จำเป็นต้องใช้คือ $2(n-1)+binary(\max\_totalRoute)$ คิวบิต เมื่อ n คือจำนวนเมือง และ *max_totalRoute* คือระยะทางรวมที่ยาวที่สุดที่เป็นไปได้สำหรับปัญหานั้นๆ ต้นทุนทั้งหมดสำหรับการกำหนดสถานะเริ่มต้นของคิวบิตคือผลรวมของจำนวนคิวบิตที่ใช้สำหรับการเข้ารหัสปัญหา TSP และจำนวนคิวบิตทดสำหรับการคำนวณคือ $O(n^2-1)+binary(\max\_totalRoute)$

8.2.2.2. ต้นทุนของวงจร

ความลึกของวงจรควอนตัมจะพิจารณาจากจำนวนครั้งของ Grover iteration ที่ใช้ ตามที่อธิบายรายละเอียดไปในข้อ 6.2.2.2 ยิ่งวนซ้ำมากความลึกของ

วงจรก็มากตามจำนวนรอบที่วนซ้ำเป็นเท่าตัว สำหรับปัญหา TSP ขนาด 3 เมือง จำนวนเกต CNOT ทั้งหมดที่ต้องใช้คือ $239t$ และความลึกของวงจร คือ $2+415t$ โดย t คือจำนวนครั้งของ Grover iteration ส่วนปัญหา TSP ขนาด 4 เมือง จำนวนเกต CNOT ทั้งหมดที่ต้องใช้คือ $1107t$ และความลึกของวงจร คือ $2+1905t$ ส่วนปัญหา TSP ขนาด 5 เมือง จำนวนเกต CNOT ทั้งหมดที่ต้องใช้คือ $5304t$ และความลึกของวงจร คือ $2+9616t$ จะเห็นได้ว่าการเพิ่มจำนวนเมืองมาแค่ 1 เมือง แต่จำนวนคิวบิต และจำนวนคิวบิตทดที่ต้องใช้เพิ่มขึ้นเป็นกำลังสอง และจำนวนคิวบิตที่เพิ่มขึ้นทำให้พื้นที่การค้นหาของอัลกอริทึมการค้นหาของโกรเวอร์เพิ่มขึ้นเป็นเอ็กโปเน็นเชียล (Exponential) นั่นหมายถึงจำนวนครั้งของ Grover iteration ก็เพิ่มขึ้นเป็นเอ็กโปเน็นเชียลเช่นกัน จึงทำให้จำนวนเกต CNOT และความลึกของวงจรสำหรับปัญหา TSP ขนาด 4 เมือง เพิ่มขึ้นอย่างมากเมื่อเทียบกับ ปัญหา TSP ขนาด 3 เมือง และความลึกของวงจรสำหรับปัญหา TSP ขนาด 5 เมือง เพิ่มขึ้นอย่างมากเมื่อเทียบกับ ปัญหา TSP ขนาด 4 เมือง จึงสามารถสรุปได้ว่าความซับซ้อนในเชิงควอนตัมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) ที่นำเสนอในงานวิจัยนี้เท่ากับ $O\left(I\frac{\pi}{4}\sqrt{\frac{N}{T}}\right)$ เมื่อ I คือความลึกของวงจรควอนตัมสำหรับ Grover iteration ครั้งแรก N คือจำนวนสถานะควอนตัมทั้งหมด และ T คือจำนวนสถานะควอนตัมที่เป็นคำตอบของฟังก์ชันโอราเคิล

จำนวนเมือง	จำนวนคิวบิต	จำนวนคิวบิตทด	จำนวนเกต CNOT	ความลึกของวงจร
3	$(n-1)^2$	$2(n-1)$ $+ binary(\max\_totalRoute)$	$239t$	$2+415t$
4	$(n-1)^2$	$2(n-1)$ $+ binary(\max\_totalRoute)$	$1107t$	$2+1905t$
5	$(n-1)^2$	$2(n-1)$ $+ binary(\max\_totalRoute)$	$5304t$	$2+9616t$

ตารางที่ 6 สรุปต้นทุนวงจรควอนตัม จำแนกตามจำนวนคิวบิต จำนวนคิวบิตทด จำนวนเกต CNOT และความลึกของวงจร สำหรับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมเวอร์ชันปรับปรุง

บทที่ 9
สรุปผล

9.1 สรุปผลการวิจัย

งานวิจัยนี้ต้องการนำเสนอขั้นตอนวิธีทางพันธุกรรมแบบกระชับชนิดควอนตัมสำหรับปัญหายาก โดยเป็นการนำข้อได้เปรียบจากการประมวลผลเชิงควอนตัม ได้แก่ สภาวะซ้อนทับของสถานะควอนตัม (Quantum superposition) และการประมวลผลควอนตัมแบบขนาน (Quantum parallelism) ในอัลกอริทึมการค้นหาของโกรเวอร์ (Grover's search algorithm) มาประยุกต์ใช้ในกระบวนการคัดเลือกโครโมโซมของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกโครโมโซมที่ดี (Compact genetic algorithm with an elite) เพื่อให้ได้ขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่มีประสิทธิภาพดีขึ้นในแง่ของความถูกต้องของคำตอบ และสามารถนำมาใช้แก้ปัญหาการเดินทางของพนักงานขาย (traveling salesman problem) ซึ่งเป็นปัญหา NP-hard ได้บนเครื่องจำลองคอมพิวเตอร์เชิงควอนตัม

ผู้วิจัยได้ทำการศึกษาและออกแบบวิธีการเข้ารหัสปัญหา TSP โดยใช้แบบจำลอง Ising เพื่อทำการแปลงข้อมูลอินพุตให้อยู่ในสถานะควอนตัม จากนั้นจึงกำหนดฟังก์ชันโอราเคิล โดยในเบื้องต้นของการออกแบบขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม (Grover-assisted cGA*) ผู้วิจัยสนใจเฉพาะการกำหนดให้ฟังก์ชันโอราเคิลสามารถตรวจสอบรูปแบบคำตอบที่เป็นเส้นทางที่เป็นไปได้ของคำตอบของปัญหา TSP (Feasible path) ผลการทดลองเปรียบเทียบประสิทธิภาพของขั้นตอนวิธีดังกล่าวกับขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดดั้งเดิมที่มีการคัดเลือกโครโมโซมที่ดี (cGA*) พบว่า Grover-assisted cGA* สามารถหาคำตอบของปัญหา TSP สำหรับเมืองขนาดเล็ก คือ 3 เมือง และ 4 เมืองได้ แต่ใช้จำนวนครั้งในการประเมินค่าความเหมาะสมมากกว่า cGA* มาก เนื่องจากจำนวนครั้งในการประเมินค่าความเหมาะสมของ Grover-assisted cGA* เพิ่มขึ้นตามจำนวนช็อตและจำนวนรอบของโกรเวอร์ที่ใช้ ดังนั้นยิ่งจำนวนช็อต และจำนวนรอบของโกรเวอร์ที่ใช้มีค่ามากจำนวนครั้งในการประเมินค่าความเหมาะสมจะเพิ่มขึ้นเป็นเท่าตัว นอกจากนี้ผู้วิจัยพบว่าการเพิ่มจำนวนช็อตไม่ได้ส่งผลต่อการเพิ่มประสิทธิภาพของขั้นตอนวิธีที่นำเสนอโดยตรง แต่ช่วยทำให้การกระจายตัวของคำตอบมีความแม่นยำมากขึ้นเท่านั้น ในขณะที่การเพิ่มจำนวนรอบของโกรเวอร์ ทำให้

ประสิทธิภาพในการหาคำตอบของอัลกอริทึมที่นำเสนอดีขึ้น เนื่องจากจำนวนรอบของโกรเวอร์ที่เหมาะสมจะช่วยเพิ่มความน่าจะเป็นที่วัดสถานะของคิวบิตแล้วได้คำตอบที่สนใจมากยิ่งขึ้น

นอกจากนี้ผู้วิจัยได้ทำการปรับปรุงฟังก์ชันโอราเคิลของขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัม เรียกว่า “new Grover-assisted cGA*” โดยผู้วิจัยได้กำหนดให้ฟังก์ชันโอราเคิลสามารถคำนวณผลรวมระยะทางของรูปแบบคำตอบทั้งหมดในพื้นที่การค้นหาของโกรเวอร์ และสามารถเปรียบเทียบกับผลรวมระยะทางที่สั้นที่สุด ณ ปัจจุบันได้ เพื่อให้ผลลัพธ์ของฟังก์ชันโอราเคิลได้รูปแบบคำตอบที่ไม่แย่กว่ารูปแบบคำตอบที่ดีที่สุด ณ ปัจจุบัน การปรับปรุงดังกล่าวช่วยลดการคำนวณระยะทางบนเครื่องคอมพิวเตอร์แบบดั้งเดิม โดยเปลี่ยนไปคำนวณระยะทางด้วยการประมวลผลเชิงควอนตัมแทน ซึ่งสามารถทำได้เร็วกว่าบนเครื่องคอมพิวเตอร์แบบดั้งเดิม เพราะสามารถประมวลผลระยะทางของทุกรูปแบบคำตอบได้ในเวลาเดียวกัน จากผลการทดลองเปรียบเทียบประสิทธิภาพของ new Grover-assisted cGA* กับ cGA* พบว่าจำนวนครั้งในการประเมินค่าความเหมาะสมลดลงมากเมื่อเทียบกับ Grover-assisted cGA* เนื่องจากใช้จำนวนช็อตและจำนวนรอบของโกรเวอร์น้อยลงในการหาคำตอบที่เหมาะสมที่สุด อย่างไรก็ตามเมื่อผู้วิจัยเพิ่มขนาดของปัญหา TSP ที่ใช้ทดสอบเป็น 5 เมือง พบว่าจำนวนคิวบิตที่ต้องใช้เพิ่มขึ้นมาก นอกจากนี้จำนวนช็อตที่ต้องใช้ในการรันอัลกอริทึมก็เพิ่มมากตามไปด้วย เนื่องจากจำนวนคิวบิตที่มากขึ้น ความผิดพลาดต่างๆที่เกิดขึ้นจากการดำเนินการระหว่างคิวบิตก็เพิ่มมากขึ้น และยังมีความผิดพลาดจากการอ่านค่าสถานะคิวบิตแต่ละตัวด้วย ดังนั้นการใช้จำนวนช็อตที่เหมาะสมในการรันอัลกอริทึมเพื่อหาคำตอบจึงเป็นสิ่งสำคัญ แม้ว่าจำนวนช็อตไม่ได้มีผลในการเพิ่มประสิทธิภาพการทำงานของอัลกอริทึมโดยตรง แต่ช่วยทำให้ประสิทธิภาพการทำงานของอัลกอริทึมไม่ลดลงเนื่องจากข้อผิดพลาดดังกล่าว

9.2 งานวิจัยในอนาคต

เนื่องด้วยข้อจำกัดของจำนวนคิวบิตที่มีให้ใช้ ณ ปัจจุบันค่อนข้างน้อย ประกอบกับอัตราความผิดพลาดของเครื่องคอมพิวเตอร์เชิงควอนตัมอันเนื่องมาจากความผิดพลาดของควอนตัมเกต การวัด การสื่อสารข้ามอุปกรณ์ และประสิทธิภาพคอมไพเลอร์ของวงจรควอนตัม รวมถึงเรื่องของสัญญาณรบกวน (Noise) จากสภาพแวดล้อม ทำให้การวิจัยเพื่อพัฒนาให้การประมวลผลข้อมูลควอนตัมขนาดใหญ่สามารถทำได้จริงยังคงเป็นเรื่องที่ท้าทายอย่างมาก ผู้วิจัยเองจึงถูกจำกัดการทดลองให้สามารถรันขั้นตอนวิธีที่นำเสนอกับปัญหา TSP ที่มีขนาดเล็กได้เท่านั้น เนื่องจากจำนวนคิวบิตที่มีให้ใช้อย่างจำกัด อย่างไรก็ตามงานวิจัยนี้ยังไม่ได้มุ่งเน้นในเรื่องการปรับปรุงวงจรให้มี

ประสิทธิภาพที่ดีโดยใช้ต้นทุกของวงจรให้น้อยที่สุด ซึ่งผู้วิจัยคิดว่าสามารถปรับปรุงวงจรควอนตัมได้ตั้งแต่วิธีการเข้ารหัสข้อมูลของปัญหา TSP เพื่อลดจำนวนคิวบิตที่ต้องใช้ในการเข้ารหัสให้ได้มากที่สุด รวมถึงลดจำนวนเกตสำหรับการคำนวณระยะทางของรูปแบบคำตอบ โดยยังสามารถประมวลผลบนวงจรควอนตัมได้ นอกจากนี้ถ้าในอนาคตมีการพัฒนา Quantum processor ที่สามารถรองรับการประมวลผลจำนวนหลายคิวบิต และหลายตัวดำเนินการ โดยมีความผิดพลาดในการประมวลผลน้อย ก็น่าจะสามารถนำงานวิจัยนี้ไปต่อยอดสำหรับพัฒนาขั้นตอนวิธีเชิงพันธุกรรมแบบกระชับชนิดควอนตัมที่สามารถแก้ปัญหา TSP ขนาดใหญ่ได้

บรรณานุกรม

1. Ltd, T.C.M.G. *The future of electronics is light*. 2016 Feb 2, 2019]; Available from: https://theconversation.com/the-future-of-electronics-is-light-68903.
2. DiVincenzo, D.P. *Principles of quantum computing*. in *Proceedings ISSCC '95 - International Solid-State Circuits Conference*. 1995.
3. Denchev, V.S., et al., *What is the Computational Value of Finite-Range Tunneling?* Physical Review X, 2016. **6**(3): p. 031015.
4. LLC., F.M. *6 Practical Examples Of How Quantum Computing Will Change Our World*. 2017 Feb 10, 2019]; Available from: https://www.forbes.com/sites/bernardmarr/2017/07/10/6-practical-examples-of-how-quantum-computing-will-change-our-world/#4f9d1da780c1.
5. Malossini, A., E. Blanzieri, and T. Calarco, *Quantum Genetic Optimization.* IEEE Transactions on Evolutionary Computation, 2008. **12**(2): p. 231-241.
6. Yingchareonthawornchai, S., C. Aporntewan, and P. Chongstitvatana. *An implementation of compact genetic algorithm on a quantum computer*. in *2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE)*. 2012.
7. K. Grover, L., *Fast quantum mechanical algorithm for database search*. 1996.
8. contributors, W. *Grover's algorithm. Wikipedia, The Free Encyclopedia*. 2019 March 5, 2019]; Available from: https://en.wikipedia.org/w/index.php?title=Grover%27s_algorithm&oldid=903654732.
9. Draper, T.G., *Addition on a quantum computer.* arXiv preprint quant-ph/0008033, 2000.
10. Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning*. 1989: Addison-Wesley Longman Publishing Co., Inc. 372.
11. Harik, G.R., F.G. Lobo, and D.E. Goldberg, *The compact genetic algorithm.* IEEE Transactions on Evolutionary Computation, 1999. **3**(4): p. 287-297.
12. Musnjak, M. and M. Golub. *Using a set of elite individuals in a genetic algorithm*.

in *26th International Conference on Information Technology Interfaces, 2004.* 2004. IEEE.

13. Zurek, W.H., *Quantum Darwinism.* Nature Physics, 2009. **5**: p. 181.
14. Demmer, M., R. Fonseca, and F. Koushanfar, *RICHARD FEYNMAN: SIMULATING PHYSICS WITH COMPUTERS*. 2008.
15. Narayanan, A. and M. Moore. *Quantum-inspired genetic algorithms*. in *Proceedings of IEEE International Conference on Evolutionary Computation*. 1996.
16. Rylander, B., et al., *Quantum Genetic Algorithms*. 2000. 373.
17. Kuk-Hyun, H. and K. Jong-Hwan, *Quantum-inspired evolutionary algorithm for a class of combinatorial optimization.* IEEE Transactions on Evolutionary Computation, 2002. **6**(6): p. 580-593.
18. Meter, R.V., K. Nemoto, and W. Munro, *Communication Links for Distributed Quantum Computation.* IEEE Transactions on Computers, 2007. **56**(12): p. 1643-1653.
19. Zhou, S., et al., *A novel quantum genetic algorithm based on particle swarm optimization method and its application*. Vol. 34. 2006. 897-901.
20. Huang, J., R.A. Berry, and M.L. Honig, *Auction-based spectrum sharing.* Mob. Netw. Appl., 2006. **11**(3): p. 405-418.
21. Wang, H., et al., *The Improvement of Quantum Genetic Algorithm and Its Application on Function Optimization*. Vol. 2013. 2013.
22. Ying, M., *Quantum computation, quantum theory and AI.* Artificial Intelligence, 2010. **174**(2): p. 162-176.
23. King, J., et al., *Quantum-assisted genetic algorithm.* arXiv preprint arXiv:1907.00707, 2019.
24. Supasil, J., P. Pathumsoot, and S. Suwanna. *Simulation of implementable quantum-assisted genetic algorithm*. in *Journal of Physics: Conference Series*. 2021. IOP Publishing.
25. Laboudi, Z. and S. Chikhi. *Evolving cellular automata by parallel quantum genetic algorithm*. in *2009 First International Conference on Networked Digital Technologies*. 2009. IEEE.

26. Layeb, A. and D.-E. Saidouni, *Quantum genetic algorithm for binary decision diagram ordering problem.* International Journal of Computer Science and Network Security, 2007. **7**(9): p. 130-135.
27. Tkachuk, V., *Quantum genetic algorithm based on qutrits and its application.* Mathematical Problems in Engineering, 2018. **2018**.
28. Kuk-Hyun, H. and K. Jong-Hwan, *Quantum-inspired evolutionary algorithms with a new termination criterion, H/sub /spl epsi// gate, and two-phase scheme.* IEEE Transactions on Evolutionary Computation, 2004. **8**(2): p. 156-169.
29. Zhang, G., *Quantum-inspired evolutionary algorithms: A survey and empirical study*. Vol. 17. 2011. 303-351.
30. Nayek, U.R.S.R.a.S., *Article: Optimization with Quantum Genetic Algorithm.* International Journal of Computer Applications, 2014. **102**: p. 1-7.
31. Xiong, Y.S.a.Y.G.a.H., *Function Optimization Based on Quantum Genetic Algorithm.* Res. J. Appl. Sci. Eng. Technol, 2014. **7**: p. 144–149.
32. Talbi, H. and A. Draa, *A new real-coded quantum-inspired evolutionary algorithm for continuous optimization.* Applied Soft Computing, 2017. **61**: p. 765-791.
33. Wang, H., et al., *Improved Quantum Genetic Algorithm in Application of Scheduling Engineering Personnel*. Vol. 2014. 2014. 1-10.
34. Lee, J.-C., et al., *Quantum genetic algorithm for dynamic economic dispatch with valve-point effects and including wind power system.* International Journal of Electrical Power & Energy Systems, 2011. **33**(2): p. 189-197.
35. Talbi, H., A. Draa, and M. Batouche, *A Novel Quantum-Inspired Evolutionary Algorithm for Multi-Sensor Image Registration*. 2004.
36. Hu, W., *Cryptanalysis of TEA Using Quantum-Inspired Genetic Algorithms*. Vol. 3. 2010. 50-57.
37. Feynman, R.P., *Simulating physics with computers.* Int. j. Theor. phys, 1982. **21**(6/7).
38. Deutsch, D., *Quantum theory, the Church–Turing principle and the universal quantum computer.* Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 1985. **400**(1818): p. 97-117.

39. Shor, P.W. *Algorithms for quantum computation: discrete logarithms and factoring*. in *Proceedings 35th annual symposium on foundations of computer science*. 1994. Ieee.

40. Zurek, W.H., *Decoherence and the transition from quantum to classical—revisited*, in *Quantum Decoherence*. 2006, Springer. p. 1-31.

41. team, I.R.a.t.I.Q. *Grover's Algorithm*. 2017 Feb 25, 2019]; Available from: https://quantumexperience.ng.bluemix.net/proxy/tutorial/full-user-guide/004-Quantum_Algorithms/070-Grover's_Algorithm.html.

42. Blog, I.R. *Cramming More Power Into a Quantum Device*. 2019 March 10, 2019]; Available from: https://www.ibm.com/blogs/research/2019/03/power-quantum-device.

43. Brusco, M., C.P. Davis-Stober, and D. Steinley, *Ising formulations of some graph-theoretic problems in psychological research: models and methods.* Journal of Mathematical Psychology, 2021. **102**: p. 102536.

44. Brassard, G., et al., *Quantum amplitude amplification and estimation.* Contemporary Mathematics, 2002. **305**: p. 53-74.

45. Cheng, C.S., A.K. Singh, and L. Gopal, *Efficient three variables reversible logic synthesis using mixed-polarity Toffoli gate.* Procedia Computer Science, 2015. **70**: p. 362-368.

46. Muñoz-Coreas, E. and H. Thapliyal, *T-count and qubit optimized quantum circuit design of the non-restoring square root algorithm.* ACM Journal on Emerging Technologies in Computing Systems (JETC), 2018. **14**(3): p. 1-15.

47. Nam, Y., et al., *Automated optimization of large quantum circuits with continuous parameters.* npj Quantum Information, 2018. **4**(1): p. 1-12.

48. Cuccaro, S.A., et al., *A new quantum ripple-carry addition circuit.* arXiv preprint quant-ph/0410184, 2004.

ประวัติผู้เขียน

ชื่อ-สกุล	กมลลักษณ์ สุขเสน
วัน เดือน ปี เกิด	23 เมษายน 2533
สถานที่เกิด	จังหวัดลำปาง ประเทศไทย
วุฒิการศึกษา	จบการศึกษาระดับมัธยมศึกษา สายการเรียนวิทย์-คณิต จากโรงเรียนบุญวาทย์วิทยาลัย จังหวัดลำปาง จบการศึกษาระดับปริญญาตรีจากคณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย จังหวัดกรุงเทพมหานคร จบการศึกษาระดับปริญญาโทจากคณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย จังหวัดกรุงเทพมหานคร
ที่อยู่ปัจจุบัน	112/71 หมู่บ้านเดอะแกรนด์ พระราม 2 ถนนพระราม 2 ตำบลพันท้ายนรสิงห์ อำเภอเมืองสมุทรสาคร จังหวัดสมุทรสาคร 74000
ผลงานตีพิมพ์	“Program Development Tools: Debugging by Reverse Computing” โดย กมลลักษณ์ สุขเสน และ ประภาส จงสถิตย์วัฒนา ในงานประชุมวิชาการ “International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC) 2014” ณ โรงแรมภูเก็ต เกรซแลนด์ รีสอร์ท แอนด์ สปา จังหวัดภูเก็ต ในระหว่างวันที่ 1-4 กรกฎาคม 2557 “Exploiting Building Blocks in Hard Problems with Modified Compact Genetic Algorithm” โดย กมลลักษณ์ สุขเสน และ ประภาส จงสถิตย์วัฒนา ในงานประชุมวิชาการ “The 15th International Joint Conference on Computer Science and Software Engineering (JCSSE2018)” ณ คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล วิทยาเขตศาลายา จังหวัดนครปฐม ในระหว่างวันที่ 11-13 กรกฎาคม 2561