โครงการ

การเรียนการสอนเพื่อเสริมประสบการณ์

ชื่อโครงการ	การเปรียบเทียบอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัมในการออกแบบพันธุกรรมผ่านการปรับความเหมาะสมแบบหลายวัตถุประสงค์
	Comparison between Conventional GA and Quantum-inspired GA on Genetics Design through Multi-Objective Optimization.
ชื่อนิสิต	นายภัทรพล คำมูล 603 36466 23
ภาควิชา	คณิตศาสตร์และวิทยาการคอมพิวเตอร์
	สาขาวิชา วิทยาการคอมพิวเตอร์
ปีการศึกษา	2563

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

การเปรียบเทียบอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัมในการออกแบบพันธุกรรมผ่านการปรับความเหมาะสมแบบหลายวัตถุประสงค์

นายภัทรพล คำมูล

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2563

Comparison between Conventional GA and Quantum-inspired GA on Genetics Design through Multi-Objective Optimization.

Phattharaphon Khammun

A Project Submitted in Partial Fulfillment of the Requirements

For the Degree of Bachelor of Science Program in Computer Science

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2020

หัวข้อโครงงาน การเปรียบเทียบอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัมในการออกแบบพันธุกรรมผ่านการปรับความเหมาะสมแบบหลายวัตถุประสงค์

โดย นายภัทรพล คำมูล

สาขาวิชา วิทยาการคอมพิวเตอร์

อาจารย์ที่ปรึกษาโครงงานหลัก อาจารย์ ดร.นฤมล ประทานวณิช

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับโครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาบัณฑิต ในรายวิชา 2301499 โครงงานวิทยาศาสตร์ (Senior Project)

[illegible] หัวหน้าภาควิชาคณิตศาสตร์

(ศาสตราจารย์ ดร. กฤษณะ เนียมมณี) และวิทยาการคอมพิวเตอร์

คณะกรรมการสอบโครงงาน

[illegible] อาจารย์ที่ปรึกษาโครงงานหลัก

(อาจารย์ ดร.นฤมล ประทานวณิช)

[illegible] กรรมการ

(ผู้ช่วยศาสตราจารย์ ศศิภา พันธุวดีธร)

กิติพร พลายมาศ กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. กิติพร พลายมาศ)

นายภัทรพล คำมูล: การเปรียบเทียบอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัมในการออกแบบพันธุกรรมผ่านการปรับความเหมาะสมแบบหลายวัตถุประสงค์. (Comparison between Conventional GA and Quantum-inspired GA on Genetics Design through Multi-Objective Optimization) อ.ที่ปรึกษาโครงงานหลัก: อาจารย์ ดร. นฤมล ประทานวณิช, 35 หน้า

โครงงานเรื่อง “การเปรียบเทียบอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัมในการออกแบบพันธุกรรมผ่านการปรับความเหมาะสมแบบหลายวัตถุประสงค์” เป็นโครงงานที่จัดทำขึ้นเพื่อศึกษาข้อได้เปรียบเสียเปรียบ (trade-off) ระหว่างการปรับปรุงสายพันธ์จุลินทรีย์โดยใช้อัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมกับอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัม ขั้นตอนการพัฒนาประกอบไปด้วย 3 ส่วนหลัก คือสร้างโมเดลจากอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิม, สร้างโมเดลจากอัลกอริทึมเชิงวิวัฒนาการแบบควอนตัม และศึกษาข้อได้เปรียบเสียเปรียบของทั้งสองวิธีการ เมื่อศึกษาข้อได้เปรียบเสียเปรียบของทั้งสองวิธีการเรียบร้อยแล้ว ได้ข้อสรุปว่า อัลกอริทึมเชิงวิวัฒนาการสามารถให้คำตอบที่เหมาะสม(optimal solution) ได้เช่นเดียวกับอัลกอริทึมเชิงวิวัฒนาการแบบดั้งเดิมและสามารถเพิ่มความหลายหลายให้กับคำตอบได้ แต่วิธีที่ใช้ในการสร้างโมเดล (implementation) ยังจำเป็นต้องได้รับการปรับปรุงโดยการเพิ่มวิธีที่จะทำให้คำตอบที่ถูกครอบงำ (dominated solution) กลายเป็นคำตอบที่ไม่ถูกครอบงำ (nondominated solution)

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์ ลายมือชื่อนิสิต.......ภัทรพล คำมูล.......

สาขาวิชา วิทยาการคอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษาโครงงานหลัก.......[illegible].......

ปีการศึกษา 2563

##6033646623: MAJOR COMPUTER SCIENCE

KEYWORD: GENETIC ALGORITHMS, QUANTUM-INSPIRED GENETIC ALGORITHMS, OPTIMIZATION, FLUX BALANCE ANALYSIS

PHATTHARAPHON KHAMMUN: COMPARISION BETWEEN CONVENTIONAL GA AND QUANTUM-INSPIRED GA ON GENETICS DESIGN THROUGH MULTI-OBJECTIVE OPTIMIZATION. ADVISOR: PROF. NARUEMON PRATANWANICH, Ph.D., 35 pp.

The objective of this project is to study trade-off between conventional genetic algorithms (GA) and quantum-inspired genetic algorithms (QGA) on genetic design through multi-objective optimization. First, we implemented a GA model based on a previous work. Second, we developed a QGA model for multi-objective optimization using the same strategies as in the GA model. We finally compared and analyzed the results obtained from both models. We conclude that the QGA approach can find optimal solutions as well as GA. Although the QGA solutions were more diverse, most of them were dominated. Therefore, the strategies used in the QGA method are still needed to be improved by adding some mechanisms to generate more nondominated solutions.

Department: Mathematics and Computer Science Student's Signature......[illegible]......

Field of Study: Computer Science Advisor's Signature......[illegible]......

Academic Year: 2020

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my advisor, Professor Naruemon Pratanwanich for her invaluable help and constant encouragement throughout the planning and development of this project. I am most grateful for her teaching and advice.

In addition, I most gratefully acknowledge my parent for all their support throughout the period of this project.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Background

One of the goals in synthetic biology is to increase a cell's production of certain substances. It is involved with permuting gene activation patterns to investigate the effect of gene expression that yield the most desirable substances. This process can be done by the combination of "metabolic engineering" and "genetic engineering" techniques in order to modify gene patterns in bacterial DNA.

In general, a bacterial DNA has more than 500 genes. Hence, it is challenging to explore all patterns of active/inactive genes, which delivers more than 2^{500} possible combinations. In addition, This process is costly and time-consuming. Currently, since metabolic networks, the seriers of biochemical reactions in a cell, are well known, it is possible to simulate a cell's mechanism *in vitro* in order to discover a small set of gene combinations that drives the cell to attain the desirable amount of objective products.

Despite using computer simulation, it's prohibitively expensive to try all possible experiments. Even if we perform the simulation process with modern computing technology, it is still impractical to be done with limited resources. From this point, a genetic algorithm (GA) plays an important role to address this challenge. GA is inspired by the evolutionary theory that the fittest will survive through generations. We apply this idea to generate, select, and inherit good answers to the next better answers until the goal-satisfied answers are obtained. While in principle effective, GA is challenging to trade-off between exploitation and exploration as it uses a binary representation in a combinatorial optimization. Here, we apply a quantum-inspired genetic algorithm (QGA) by substituting binary representations with quantum states and then study advantages and disadvantages between these two methods.

1.2 Objectives

To study advantages and disadvantages between genetic algorithms and quantum-inspired genetic algorithms in the combinatorial optimization problem.

1.3 Scope

- Optimize only acetate and biomass fluxes.
- Consider only an *E coli* model.

1.4 Project Activities

1. Study genetic algorithms (GA).
2. Study the concept of metabolic engineering and microbial model.
3. Study flux balance analysis and multi-objective optimization models.
4. Study non-domination sort genetic algorithm II (NSGA-II).
5. Reproduce genetic Design through multi-objective optimization in Python.
6. Study the basics of quantum mechanics.
7. Study quantum-inspired genetic algorithms (QGA) via the combinatorial optimization problems.
8. Implement quantum-inspired genetic algorithms in MATLAB.
9. Measure and compare performance and result given from QGA to GA.

Project Activities Gantt Chart

Processes	2021										
	Jan	Feb	Mar	Apr	May	Jun	Jul	Sep	Oct	Nov	Dec
1. Study genetic algorithms (GA).	■										
2. Study the concept of metabolic engineering and microbial model.		■									
3. Study flux balance analysis and multi-objective optimization models.		■	■								
4. Study non-domination sort genetic algorithm II (NSGA-II).			■								
5. Reproduce genetic Design through multi-objective optimization in Python.			■	■	■	■	■				
6. Study the basics of quantum mechanics.								■			
7. Study quantum-inspired genetic algorithms (QGA) via the combinatorial optimization problems.									■		
8. Implement quantum-inspired genetic algorithms in MATLAB.									■	■	

9. Measure and compare performance and result given from QGA to GA.											■

1.5 Benefits

a) <u>Benefit for the implementor</u>
   1. Learn and practice about genetic algorithms, specifically NSGA-II and QGA.
   2. Be able to solve combinatorial optimization problems.

b) <u>Benefit for users</u>
   1. Give information about computational methods in regard to comparing performance between GA and QGA.

1.6 Report Outlines

This report consists of five chapters as follows. Chapter I includes background, objectives, scope, project activities and benefits of this project are presented. Chapter II includes the theoretical background knowledge relating to the work in this project are reviewed. Chapter III includes the methodology and the comparison pros and cons between models. Chapter IV includes the results of our project. Chapter V includes discussion, conclusion, and future work.

CHAPTER II

THEORETICAL BACKGROUND

2.1.Optimization

Optimization problem is a problem that needs an optimal objective value under a set of predefined constraints to be a solution. The optimization problems often have common components as follows.

- Objective function: An equation or a method to measure the goodness of a solution. Different problems usually have different objective functions.
- Objective value: A value evaluated by the corresponding objective function. We need this to be as much as maximum/minimum as possible.
- Constraints: Criteria in which every solution must be satisfied such as a lowerbound and an upperbound of each parameter. A criterion can be defined by equations or inequations.

2.2.Single-Objective and Multi-Objective Optimization problem

We can superficially categorize optimization problems into two types, one is single-objective and another one is multiple- or multi-objective function. The core of difference between those is the conflict of objective function, objective for short, that we want to optimize. In single-objective, objectives are non conflict but in multi-objective, objectives are conflict.

To illustrate what conflict and non conflict objective is, if you want to buy a car with a low price and less luxury, this is considered to be single-objective because these objectives, price and luxury, are going to the same way, decrease price, decrease luxury. On the other hand, if you want to buy a car with low price but high luxury, this is considered to be a multi-objective because these two objectives do not go the same way, if you want a luxury car, you have to pay high, but you want to pay less, clearly conflict.

The solutions that come from multi-objective optimization are called tradeoff solutions which mean if you want one objective more, you receive another one less.

Another important topic in multi-objective optimization is Pareto front. Pareto front is a set containing solutions that cannot tell one better than another. We use the term ‘not dominate’ to describe a solution that cannot tell better than another while using ‘dominate’ for otherwise. Note that the meaning of 'better' is different across different problems.

2.3.Flux balance analysis (FBA)

In a cell's metabolism, metabolite is a substance that made by organism and a process to a turnover reactant(s) metabolite(s) to the product metabolite(s) is called reaction. We refer to the action that reactants converted to products as a forward process and vice versa as a backward process. A reaction can be either reversible or irreversible. The turnover rate of a reaction is quantified and described as a reaction flux. A map of a complete set of reactions and metabolites in a cell's metabolism is called a metabolic network.

When the total amount of any metabolite being produced must be equal to the amount being consumed, the system is in a steady-state (Orth JD *et al.* 2010). One of the 22 main factors influencing flux production is the presence or absence of a set of genes (Occhipinti *et al*. 2020). For example, disabling gene A can cause decreasing metabolite B in reaction C whereas increasing metabolite D in reaction E. In metabolic Engineering, we can benefit from the relationships between genes and metabolic networks in a cell’s steady state by modifying certain genes in order to obtain the maximum flux of desired metabolites. Such modification processes can be done systematically *in vitro* by changing knockout status of each gene and then investigating the consequence in the fluxes of desired metabolites until reaching the optimal amount.

FBA is an approach to find optimal fluxes in the steady-state based on all reactions, given a metabolic network and a set of active genes.

2.4.Mathematical representation of a metabolic model

In order to represent the entire metabolic network mathematically, a stoichiometric coefficient matrix (S) is used. Let m and n are a number of metabolites and reactions in a cell respectively. S is an m x n matrix where $\boldsymbol{S_{ij}}$ represents the number of molecules of a metabolite i used in the jth reaction. Moreover, this matrix also describes how metabolites are related to each reaction as follows.

- $\boldsymbol{S_{ij}} < \mathbf{0}$, the metabolite i is involved in the reaction j as a reactant.
- $\boldsymbol{S_{ij}} = \mathbf{0}$, the metabolite i is not used in the reaction j.
- $\boldsymbol{S_{ij}} > \mathbf{0}$, the metabolite i is involved in the reaction j as a product.

A set of genes is represented by a binary row vector (y) where 1 denotes that the gene is deactivated or knockout and 0 otherwise.

As stated before, the existence of a set of genes has an effect on biochemical behavior of a cell (Occhipinti *et al*. 2020). An L x n GPR mapping matrix (G) is used to describe such relationships where L and n are a number of genes and reactions in the cell, respectively. Each entry $\boldsymbol{G_{ij}}$ in the matrix can be either 0 if a reaction i has effect to a reaction j or 1 otherwise.

All of the reaction fluxes are denoted by a column vector v which has a number of elements equal to the number of reactions in a cell's metabolism (n). Notice that those elements in v is a real number.

Figure 1.1 is illustrating a general form of an optimization problem used in FBA.

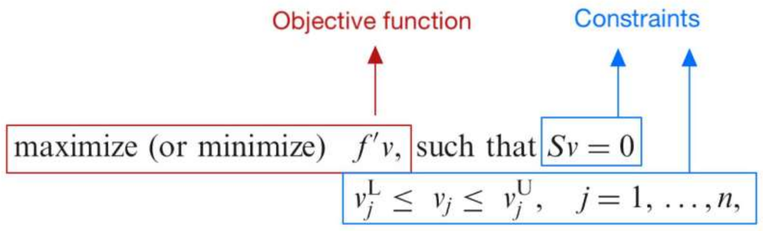
Objective function
Constraints
maximize (or minimize) $f'v$, such that $Sv = 0$
$v_j^L \leq v_j \leq v_j^U, \quad j = 1, \ldots, n,$

Figure 2.1 A general form of an optimization problem used in FBA

2.5.Classical algorithms for combinatorial optimization

Genetic algorithm (GA)

GA is a search heuristic that is inspired by the theory of natural evolution. This algorithm is analogous to the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation. Processes in GA are composed of

Population Initialization

We initialize a set of individuals, called population. The characteristics of individuals depend on what problems you want to solve with. Each individual is composed of genes, Each gene is embedded in binary (for this project), so, in some contexts, people often call individuals by chromosome instead.

Fitness Value Calculation

Fitness value or fitness score show how an individual is fit quantitatively. The fitness value is given by a fitness function that takes an individual as an input and returns a fitness value as an output.

Parent Selection

Parent selection is a process that uses fitness value to select individuals with predefined numbers to use as a parent to generate offspring, called crossover.

Crossover

After parent is acquired, we do crossing over among these parent to obtain offspring. Then, we add parent and offspring and set it to be individuals of next generation.

Mutation

To increase diversity among the population in the generation, GA has a mutation step to randomly change a gene value. Notice that changing gene value is done under a little amount of genes.

GA has many termination criteria. Here, we choose population convergence, offspring do not produce much differences in objective value relative to parents, to terminate algorithms.

2.6.Non-dominated sort genetic algorithm II (NSGA-II)

NSGA-II is the improved version of GA. It improves parent selection step by separate it to two step Non-dominated sorting and crowding distance, how far within the solution.

This modification yields a diversity among individuals in each generation and reduces sharing parameters from experimentally entered by the implementor (K. Deb *et al.* 2002).

2.7.Quantum-inspired algorithms for combinatorial optimization

Quantum Superposition

Quantum superposition is a principle of quantum mechanics. It states that any two (or more) pure quantum states, a mathematical entity that provides a probability distribution for the outcomes of each possible measurement on a system (Kuk-Hyun Han and Jong-Hwan Kim 2002), can be added together, called superposed, and the result will be another valid quantum state, says, that every quantum state can be represented as a sum of two or more other distinct states.

Quantum Computing

a) Quantum bit (Qubit)

Superposition is used as one of the quantum phenomena in quantum computing. In quantum computing, the smallest unit of information stored in a two-state quantum computer is called a quantum bit or qubit. A qubit may be in the “1” state, in the “0” state, or in any superposition of the two (Zhang G 2011). The state of a qubit can be represented as:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

where α and β are complex numbers that amplify the probability amplitudes of the "0" state and "1" state respectively. $|\alpha|^2$ gives the probability that the qubit will be found in the “0” state and $|\beta|^2$ gives the probability that the qubit will be found in the “1” state. Normalization of the state to unity guarantees

$$|\alpha|^2 + |\beta|^2 = 1.$$

b) Quantum gate (Qugate)

In the quantum circuit model of computation, a quantum logic gate (or qugate) is a basic quantum circuit operating on a small number of qubits. They are the building blocks of quantum circuits like classical logic gates are for conventional digital circuits. There are many qugates used in the quantum computing domain, but, here, we are considered only a rotation gate. The rotation gate can be exposition as:

$$U(\Delta\theta_i) = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix}$$

where $\Delta\theta_i$ is a rotation angle of each Q-bit toward either 0 or 1 state depending on its sign. We use this rotation gate together with the lookup table, table that specify angle rotate to, to escape a local optimum (A. Narayanan and M. Moore 1996).

Quantum-Inspired Genetic Algorithms (QGA)

QGA applies qugate and qubit to GA by modifies chromosome representation and mutation as describes following:

1. It substitutes binary embedded by qubit embedded as an information unit in the chromosome.

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

Figure 2.2 A gene unit in the chromosome

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{array}\right]$$

Figure 2.3 A chromosome m-length

2. It uses rotation by rotation gate to perform rotation instead of mutation.

CHAPTER III

METHODOLOGY

3.1. An overview of process

In GA, we have 6 steps. There are initialization, selection, crossover, and mutation. In the first generation, we generate population of chromosome in initialization. After the first generation to the last generation(as predefined, 1500), we select parent chromosome in selection, produce offspring from parent in crossover, variate the offspring in mutation, and select the population for the next generation. In QGA, we have 6 steps from GA plus 1 step, rotation, between mutation and population selection.

3.2. Strategy of implementation in each step

a. Initialization

In GA, first, we generate 1000 chromosomes as a population. The chromosome is a real-value vector. Each chromosome has 1041 genes plus 2 objective values, flux of acetate and biomass respectively. Second, we randomly select 50 genes from 1041 genes to knockout by set it to 1. Finally, we evaluate the objectives of each chromosome.

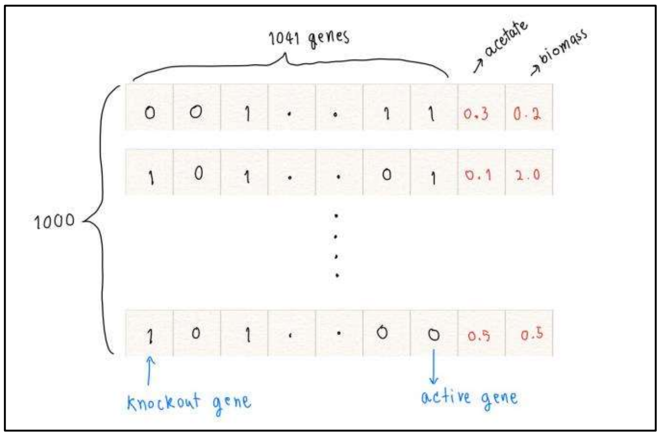

Figure 3.1 Initialization of GA

In QGA, we generate 1000 qubits by randomly initiate probability of state 0, α, and state 1, β, under superposition constraint, $\alpha^2 + \beta^2 = 1$. Second, we make chromosome by observing each qubit. There are 2 steps. First, we create vector r with 1041 elements. The element r_i, in r is a random value between 0 and 1. Then, If r_i greater than α_i^2, we assign 1 into the chromosome at the index i. otherwise, we assign 0.

To obtain objective value of each chromosome, we use the same step as used in GA.

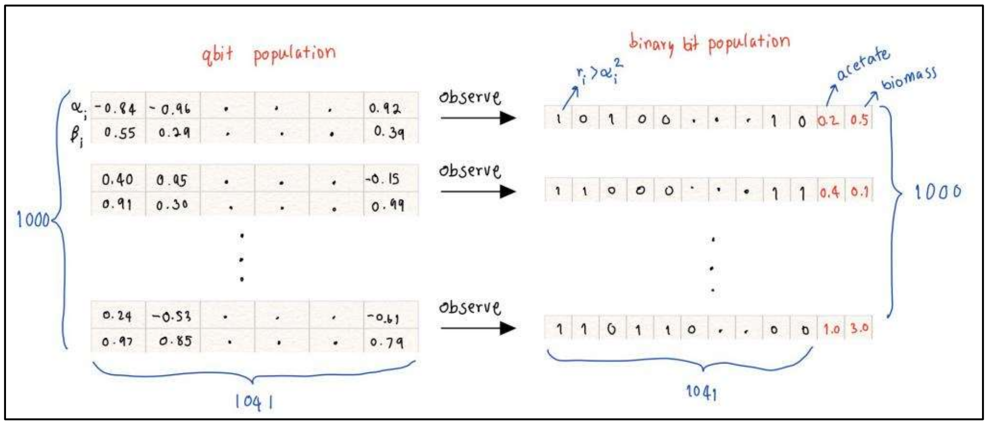

Figure 3.2 Initialization of QGA

b. Parent Selection

This step performs similarly both GA and QGA. We use tournament selection strategy to select the parent chromosome. In this project, we use half of population, 500 chromosomes, to be parents. Tournament selection is 2-nested loop. First loop is using for pooling chromosome. We pool chromosomes by randomly select 2 chromosomes from the population. After that, we select a chromosome which dominate each other (If chromosomes nondominated to each other, we select the one which have minimum rank, maximum distance, or random sequentially) in Inner loop.

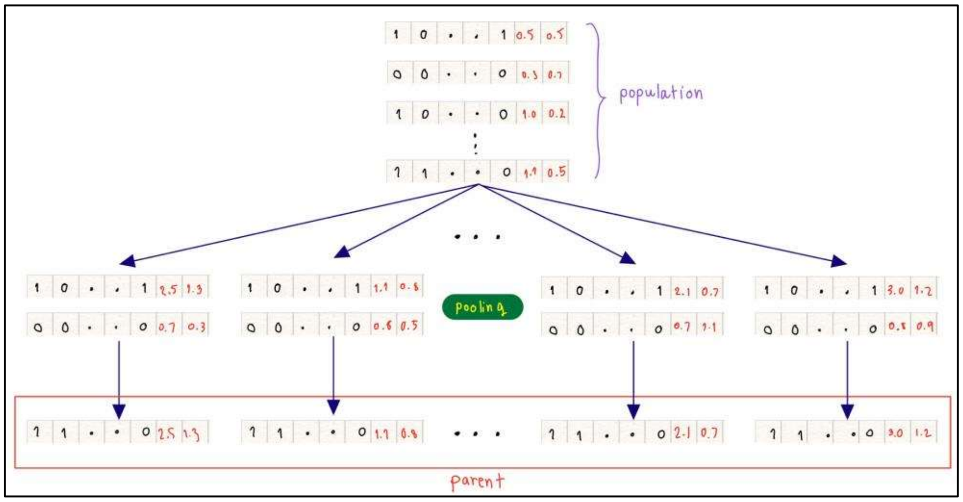

Figure 3.3 Parent selection

In QGA, we return index of parents along with chromosomes to using in select qubit population.

c. **Crossover**

This step performs almost similar in both GA and QGA. We use element-wise crossover method to produce offspring. This method is composed of 3 steps. First, we randomly select 1 pair of chromosomes from parent set from previous step, called dad and mom. Second, we generate vector r which element have a value given by random 0 to 1. If r_i is greater than 0.5, select bit i from dad, else select bit from mom. We repeat step 1 and 2 until we retrieve 500 offspring. Finally, we calculate acetate and biomass of offspring.

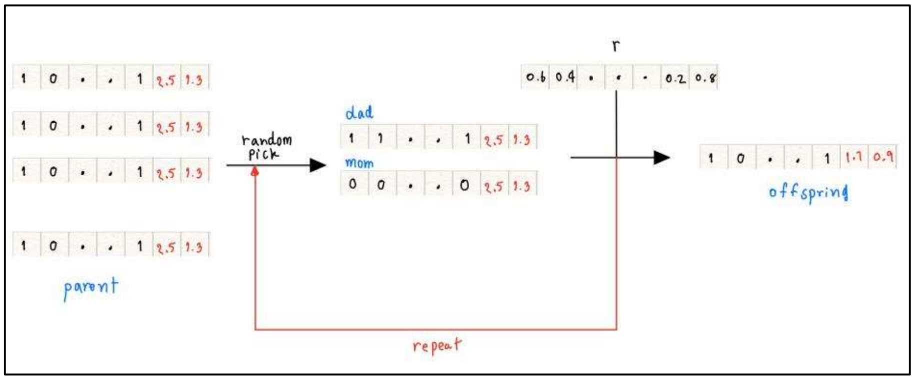

Figure 3.4 Crossover of GA

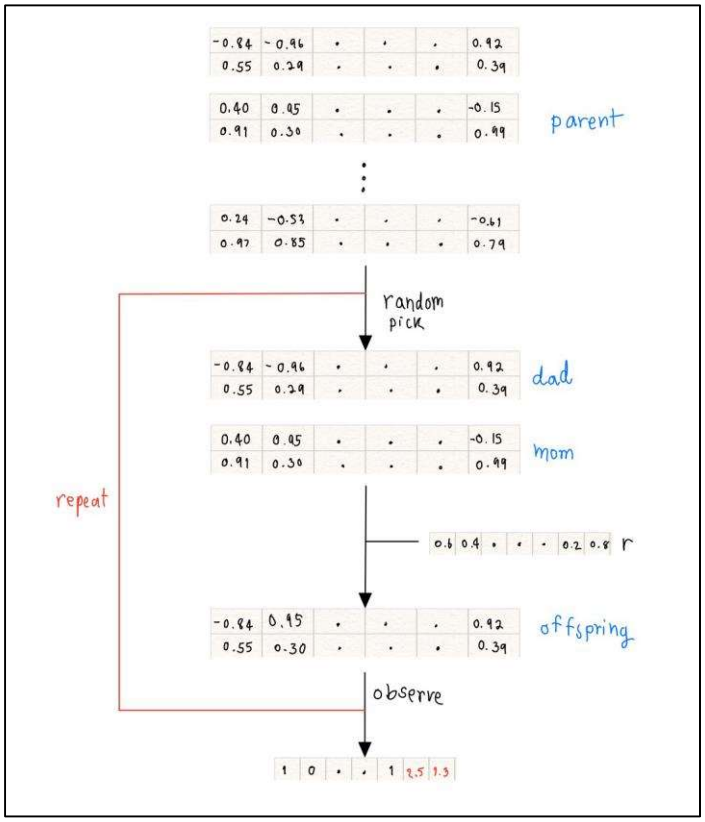

Figure 3.5 Crossover of QGA

d. **Mutation**

Mutation step create vector r with 1041 elements. The element r_i, in r is a random value between 0 and 1. Then, if r_i is less than mutation rate, which we set it to 0.2, we inverse knockout state of gene i. In GA, we inverse ith bit. In QGA, we switch probability of state-0 and state-1.

e. **Rotation**

This step performs only QGA. We rotate qubit with rotation rate 0.2. The rotation is defined by perform following operation:

$$R(\Delta\theta) = \begin{bmatrix} \cos(\Delta\theta) & \sin(-\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix}$$

The $\Delta\theta$ is defined by following lookup table from Bin-Bin Li,2002:

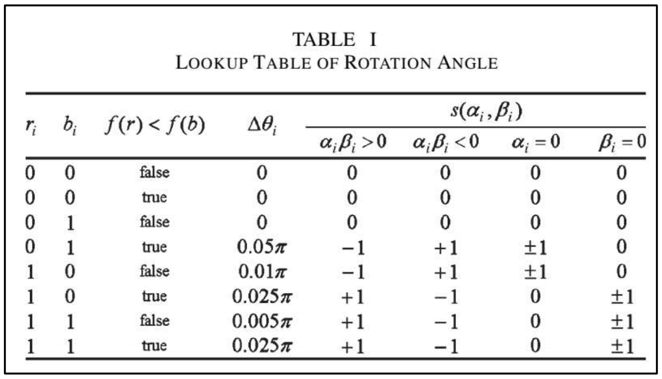

TABLE I
LOOKUP TABLE OF ROTATION ANGLE

r_i	b_i	$f(r) < f(b)$	$\Delta\theta_i$	$s(\alpha_i, \beta_i)$			
				$\alpha_i\beta_i > 0$	$\alpha_i\beta_i < 0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	false	0	0	0	0	0
0	0	true	0	0	0	0	0
0	1	false	0	0	0	0	0
0	1	true	0.05π	-1	$+1$	±1	0
1	0	false	0.01π	-1	$+1$	±1	0
1	0	true	0.025π	$+1$	-1	0	±1
1	1	false	0.005π	$+1$	-1	0	±1
1	1	true	0.025π	$+1$	-1	0	±1

Table 3.1 Lookup table

In this table, r_i is the binary bit observed from qubit at index i. b_i is the best bit, the first chromosome of the first rank from previous generation. $f(r), f(b)$ is the objective value of r and b respectively.

f. **Population selection**

This last step combining 1000 current population and 500 offspring from crossover. After that, perform nondomination sort to combined population, select only the first 1000 chromosomes to be the parent of next generation (iteration).

CHAPTER IV
RESULTS

In this chapter, we will describe the results obtained from the last generation used as final solutions and the results across generations in order to see how population was evolved across generations.

4.1 Results from the last generation

a. <u>Solutions at the Pareto-front of the 1500th generation</u>

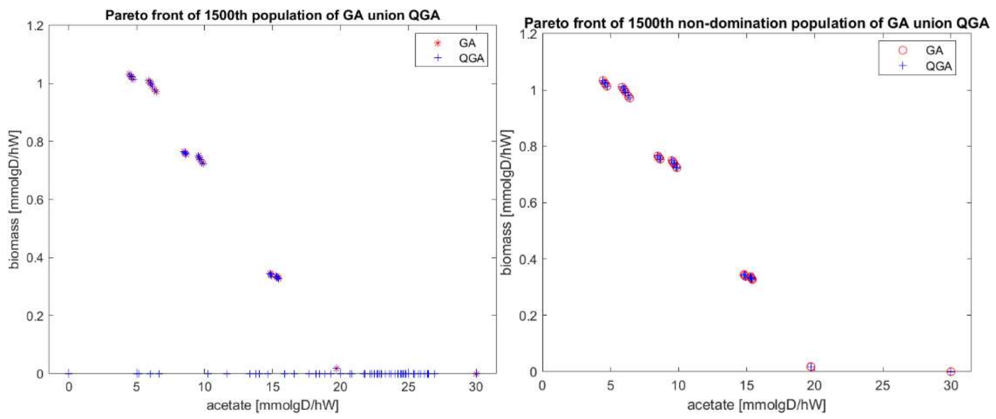

Figure 4.1 The entire population generated by GA and QGA (left). The final solutions at the first front of the last generation (right).

The results show that GA and QGA have exactly the same solution set in the first front. Unfortunately, QGA have many dominated solutions around 15-30 mmolgD/hW of acetate. Another point that is worth to mention is that even though GA does not have any dominated solutions, it contains fewer unique solutions (in here, unique solutions are referred to the solutions that produce the different amount of acetate and biomass) as illustrated in Figure 4.2. GA had only 74 unique solutions out of 1,000 (population size per generation). In addition, even though QGA produced more unique solutions than GA, most of them (1000-131 = 869) were dominated by others, resulting in not sitting at the first front.

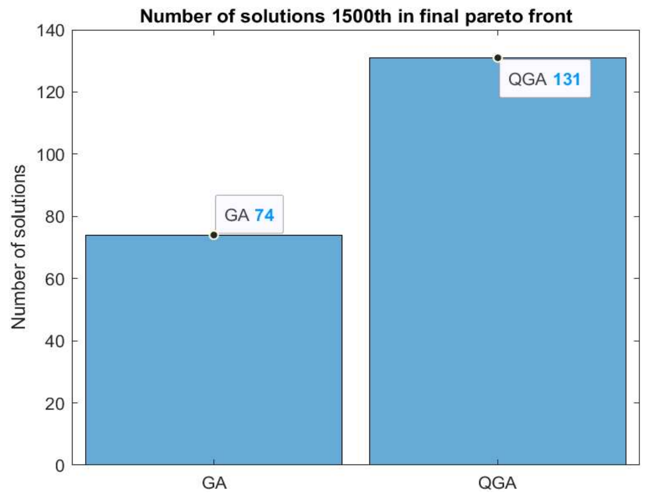

Figure 4.2 Number of unique solutions among the entire population in the 1500th generation.

b. <u>Top 15 most knockout genes</u>

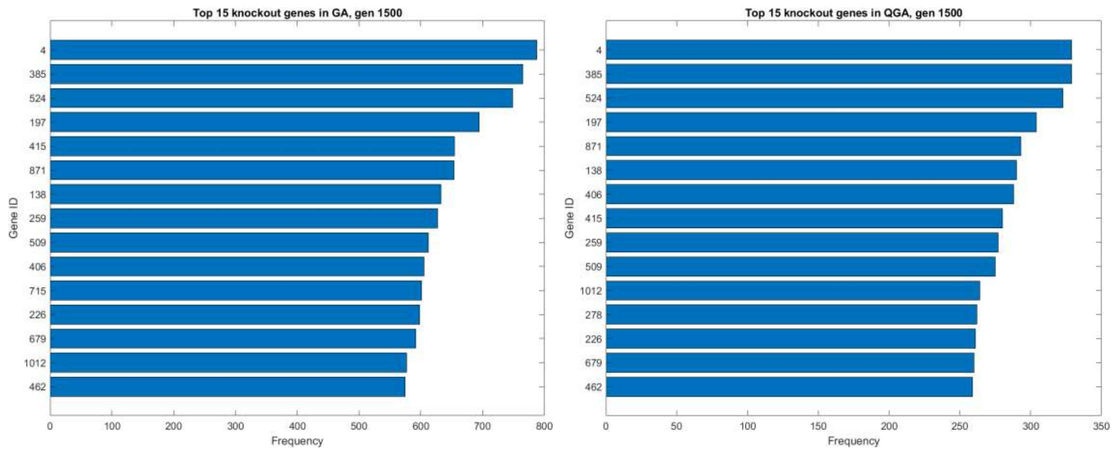

Figure 4.3 The most selected knockout genes in GA (left) and QGA (right).

First, there are the genes that GA and QGA tend to prioritize to be knocked out, such as Gene ID 4, 385, 524, and 197. Second, there are some genes that both GA and QGA selected to knockout, but with different frequency, as shown below in the rank after Gene ID 197. Finally, if we look at all genes, as illustrated in figure 4.4, we found that GA never picked the other 901 genes to knockout while QGA explored all genes despite by fewer individuals. In my point of view, this may cause GA to get stuck at local optima because GA had more redundant solutions, the solution that produce exactly the same objective values regardless knockout patterns, than QGA. I assume that this characteristics of QGA may come from the uncertainty of qubit state and the rotation operations.

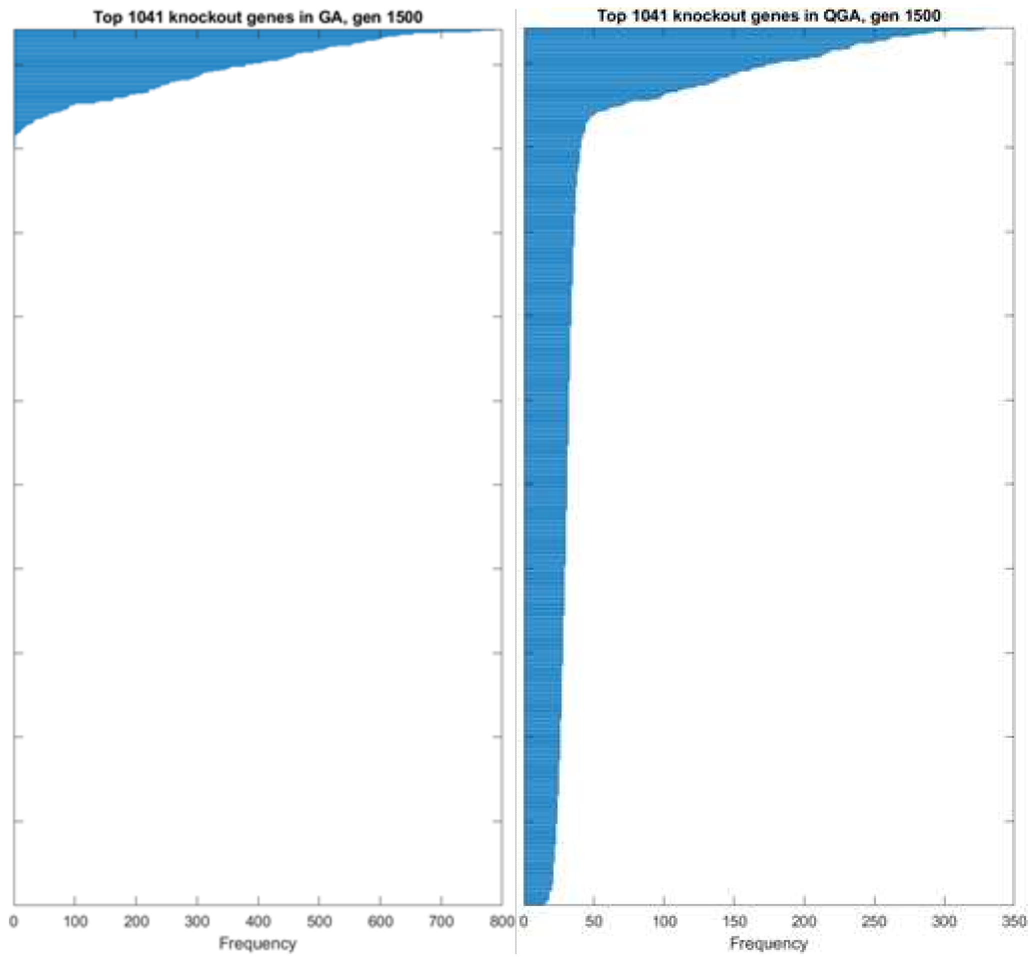

Figure 4.4 The number of knockout genes from GA (left) and QGA (right).

4.2 Results across generations

a. <u>Number of unique solutions at the first front in each generation</u>

Figure 4.5 illustrates the number of unique solutions in the generation 1, 100, 200, 300, 400, 500, 1000, and 1500. We can see that the number of unique solutions of GA and QGA is close to each other at the first 100th generations, then significantly increases from the 100th generation to the 500th generation, and finally reaching a plateau after 1000 generations

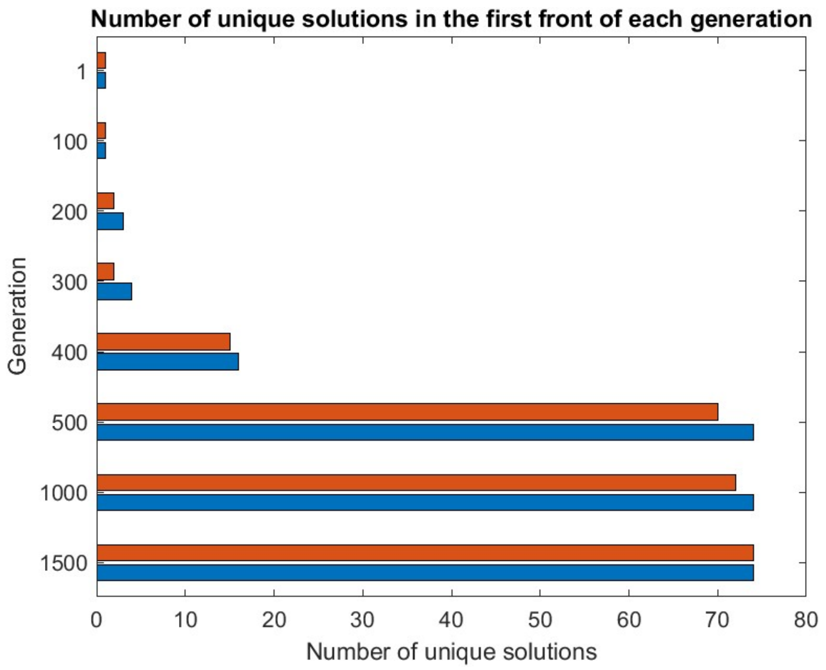

Figure 4.5 The number of unique solutions sitting at the first front of each generation.

According to Figure 4.6, the pareto fronts from the 500th generation and those from the 1000th generation are almost the same. Hence, we can stop the model in at the 500th generation around to save more computational cost.

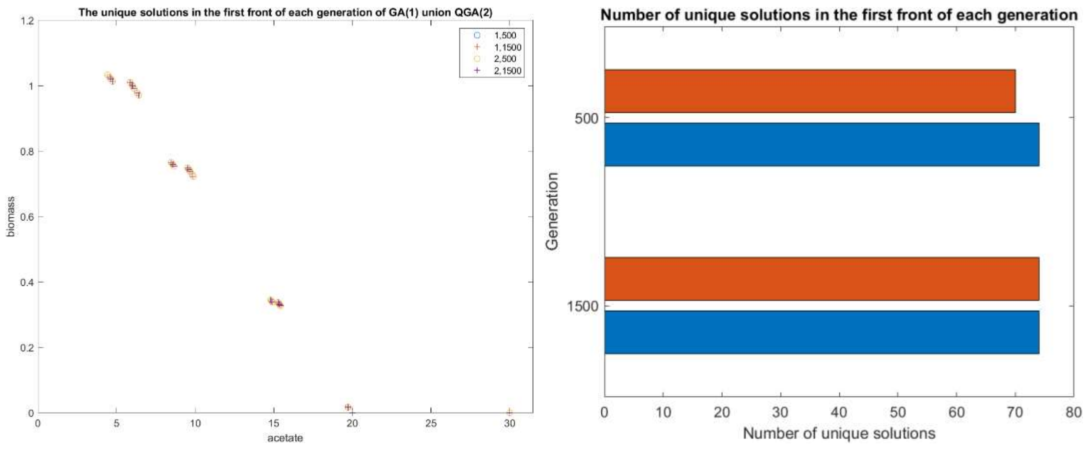

Figure 4.6 (left) the first front of unique solution in generation 500 and 1500. (right) the number of unique solutions in first front of gen 500 and 1500.

b. <u>Run time per generation</u>

We changed the strategies of GA implementation from Costanza, 2012 due to the limitation of run time. The details can be found in the appendix at the end of the report. As Table 4.1 shows, QGA spent as twice as GA per one generation, partly because of the rotation operation, which called an additional GLPK, a linear optimization solver. In total, the optimization solver was called twice before and after the rotation operation was performed in QGA, compared to only once in GA.

Approach	**Average run time per generation (second)**
GA (Costanza, 2012)	1,588.35
QGA	125.336
GA	58.69

Table 4.1 The average run time per generation spent by different approaches.

CHAPTER V
DISCUSSION

5.1 Discussion

We have three topics to discuss here. First, we found that not many genes directly affect the objective values. So, it is better if we have a preprocessing method to weighting relationship between genes and objective values. Second, the strategies implemented here do not give as good final results as those proposed in Costanza, 2012, in terms of the production of diverse and nondominated offspring. However, the computational time and resources are more expensive. Finally, QGA can increase variations as we expected, but most of them are dominated by the solutions which are similar to those generated by GA.

5.2 Conclusion

The QGA approach can find optimal solutions as well as GA and also improve diversity among the solutions. Unfortunately, the strategies used in QGA still need to be improved by adding some mechanisms to enhance them to be nondominated solutions.

5.3 Future work

The future work includes the implementation of GA with redundant solutions handler, implementation QGA with mechanism that enhance solutions to be nondominated and performing more experiments with different population size and number of generations.

REFERENCES

A. Narayanan and M. Moore, "Quantum-inspired genetic algorithms," Proceedings of IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 1996, pp. 61-66, doi: 10.1109/ICEC.1996.542334.

Costanza J, Carapezza G, Angione C, Lió P, Nicosia G. “Robust design of microbial strains”. Bioinformatics. 2012 Dec 1;28(23):3097-104. doi: 10.1093/bioinformatics/bts590. Epub 2012 Oct 7. PMID: 23044547.

K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, April 2002, doi: 10.1109/4235.996017.

Kuk-Hyun Han and Jong-Hwan Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 6, pp. 580-593, Dec. 2002, doi:10.1109/TEVC.2002.804320.

Occhipinti, A., Hamadi, Y., Kugler, H., Wintersteiger, C., Yordanov, B., & Angione, C. (2020). Discovering Essential Multiple Gene Effects through Large Scale Optimization: an Application to Human Cancer Metabolism. IEEE/ACM Transactions on Computational Biology and Bioinformatics. https://ieeexplore.ieee.org/document/9055142

Orth JD, Thiele I, Palsson BØ. What is flux balance analysis? Nat Biotechnol. 2010 Mar;28(3):245-8. doi: 10.1038/nbt.1614. PMID: 20212490; PMCID: PMC3108565.

Satvik Tiwari. (2019). Genetic Algorithm: Part 1 -Intuition. access 19 February 2020, https://medium.com/koderunners/genetic-algorithm-part-1-intuition-fde1b75bd3f9

Satvik Tiwari. (2019). Genetic Algorithm: Part 2 — Implementation. access 11 March 2020,https://medium.com/koderunners/genetic-algorithm-part-2-implementation-69d77cf668bf

Satvik Tiwari. (2019). Genetic Algorithm: Part 3 — Knapsack Problem. access 11 March 2020, https://medium.com/koderunners/genetic-algorithm-part-3-knapsack-problem-b59035ddd1d6

Zhang, G. Quantum-inspired evolutionary algorithms: a survey and empirical study. J Heuristics 17, 303–351 (2011). https://doi.org/10.1007/s10732-010-9136-0

APPENDIX

Strategy of implementation of Costanza, 2012

Costanza, 2012 use different crossover and mutation strategy and add redundant handling to the GA process

a) <u>Genetic operator (Crossover and Mutation)</u>

This step is composed of 3-leveled nested loop. The outmost for-loop loop through every individual to use it as a parent. The first inner-loop is while-loop which iterating to generate 10 candidate offspring by which randomly switch binary state 1 position. The second inner-lop is while-loop that iterating to ensure whether no chromosome has knocked out exceed limit. if so, randomky un-knockout it.

After finish inner-loop, we have 10 candidate offspring. So, they measure the objective values of every offspring. Then, they nondomination sort them and append the offspring that is at the first individual at the first rank to the offspring set.

b) <u>Delete redundant</u>

This step iterates through offspring and randomly un-knockout 1 position. Then evaluating objective values and compare to the offspring before un-knockout. If the difference of acetate is less than 10^{-10}, unknockout.

BIOGRAPHY

Mr. Phattharaphon Khammun

Department of Mathematics and Computer Science

Faculty of Science, Chulalongkorn University

Email: k.pattara@outlook.com