Final Project Report

1. Info

Group : 5

Members :

(左) B08901056 盧弘偉

(右) B08901069 黃政勛

(中) B08901198 顏柏傑

2. Final project problem

Solving travelling salesman problem using phase estimation

(1) travelling salesman problem

A salesman is required to visit once and only once each of n different cities starting from a base city, and returning to this city. What path minimizes the total distance travelled by the salesman?

(2) phase estimation algorithm

In quantum computing, the quantum **phase estimation** algorithm (also referred to as quantum **eigenvalue estimation**

algorithm), is a quantum algorithm to estimate the phase (or eigenvalue) of an eigenvector of a unitary operator.

3. Setup

Utilize Qiskit environment (IBMQ) with Python

4. Demonstration result

Case1 : n = 4 (number of vertices), t = 6 (number of counting bits)

path	eigen state	output phase	distance	check
(1, 2, 3, 4)	11000110	100100 -> 36	3.5342917352885173	3.5342917352885173
(1, 2, 4, 3)	10001101	100000 -> 32	3.141592653589793	3.141592653589793
(1, 3, 2, 4)	11100001	011100 -> 28	2.748893571891069	2.748893571891069
(1, 3, 4, 2)	01110010	100000 -> 32	3.141592653589793	3.141592653589793
(1, 4, 2, 3)	10110100	011100 -> 28	2.748893571891069	2.748893571891069
(1, 4, 3, 2)	01101100	100100 -> 36	3.5342917352885173	3.5342917352885173

The min. distance path would be (1, 3, 2, 4, 1) or (1, 4, 2, 3, 1)

Case2 : n = 8 (number of vertices), t = 6 (number of counting bits)

```
(1, 6, 4, 8, 7, 3, 2, 5) 10001011010100100011101 1 011100 -> 28    2.748893571891069 2.7488935718910694
(1, 6, 4, 8, 7, 3, 5, 2) 00110011010101000011101 1 010111 -> 23    2.2580197197676637 2.258019719767664
(1, 6, 4, 8, 7, 5, 2, 3) 01010000110111000011101 1 100001 -> 33    3.2397674240144743 3.2397674240144743
(1, 6, 4, 8, 7, 5, 3, 2) 00101010010111000011101 1 100000 -> 32    3.141592653589793 3.141592653589793
(1, 6, 5, 2, 3, 4, 7, 8) 11110000101010100011110 1 011110 -> 30    2.945243112740431 2.945243112740431
(1, 6, 5, 2, 3, 4, 8, 7) 11010000101010100011101 1 011010 -> 26    2.552544031041707 2.552544031041707
(1, 6, 5, 2, 3, 7, 4, 8) 11110000111010100010011 1 011110 -> 30    2.945243112740431 2.945243112740431
(1, 6, 5, 2, 3, 7, 8, 4) 01110000111110100010110 1 011011 -> 27    2.650718801466388 2.650718801466388
(1, 6, 5, 2, 3, 8, 4, 7) 11010000111110100011010 1 011001 -> 25    2.454369260617026 2.454369260617026
(1, 6, 5, 2, 3, 8, 7, 4) 01110000111010100011010 1 010000 -> 16    1.5707963267948966 1.570796326794897
(1, 6, 5, 2, 4, 3, 7, 8) 11110001100110100010110 1 010011 -> 19    1.8653206380689396 1.8653206380689396
(1, 6, 5, 2, 4, 3, 8, 7) 11010001100110100011010 1 001001 -> 9     0.8835729338221293 0.8835729338221293
(1, 6, 5, 2, 4, 7, 3, 8) 11110011000110100011010 1 010010 -> 18    1.7671458676442586 1.7671458676442588
(1, 6, 5, 2, 4, 7, 8, 3) 01010011100110100011110 1 010101 -> 21    2.061670178918302 2.061670178918302
(1, 6, 5, 2, 4, 8, 3, 7) 11010011100110100010011 1 010111 -> 23    2.2580197197676637 2.258019719767664
(1, 6, 5, 2, 4, 8, 7, 3) 01010011000110100011011 1 010010 -> 18    1.7671458676442586 1.7671458676442588
(1, 6, 5, 2, 7, 3, 4, 8) 11110011001010100001011 1 011011 -> 27    2.650718801466388 2.6507188014663883
(1, 6, 5, 2, 7, 3, 8, 4) 01110011011110100001010 1 010010 -> 18    1.7671458676442586 1.7671458676442586
(1, 6, 5, 2, 7, 4, 3, 8) 11110001111010100001010 1 010000 -> 16    1.5707963267948966 1.570796326794897
```

```
In [16]: print(m*2*pi)
         print(res)

         0.8835729338221293
         (1, 6, 5, 2, 4, 3, 8, 7)
```

The min. distance path would be (1, 6, 5, 2, 4, 3, 8, 7, 1)

5. Code instruction

First, we build unitary gates Ui and then combine them into a U gate.

In the second part, we show how to construct eigen states with a given number of vertices/cities.

In the next part, we can run the phase estimation algorithm, get the measurement outcome, and then do some calculation on it to get the final result.

In the final part, we demonstrate the whole procedure for solving this kind of problem with the number of vertices/cities being 4 and 8.

6. Work distribution

Every member has participated in every part of work.