Solving Hamiltonian Cycle Problem with Grover's Algorithm

Advanced Computing and Networking Laboratory
National Central University
Department of Computer Science & Information Engineering

研究生： 高子豪　　指導教授: 江振瑞 博士

ACAN Laboratory

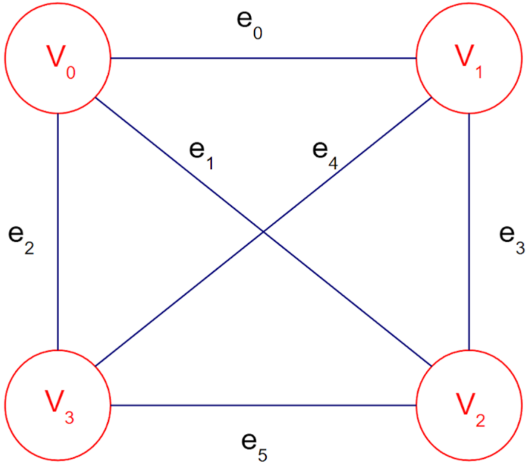
Outline

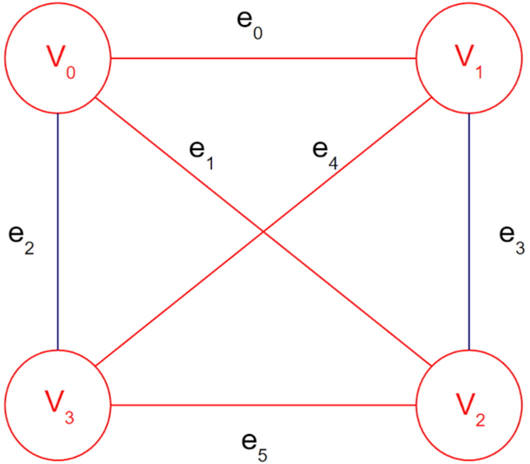
- Introduction
- Background and related knowledge
- Proposed Method
- Experiment and Result
- Conclusion

ACAN Laboratory

Hamiltonian Cycle Problem

- Definition : Given a graph G=(V,E), determine if there exists a Hamiltonian Cycle that passes through all vertices in V, except the starting vertex, exactly once, and returns to the starting vertex.



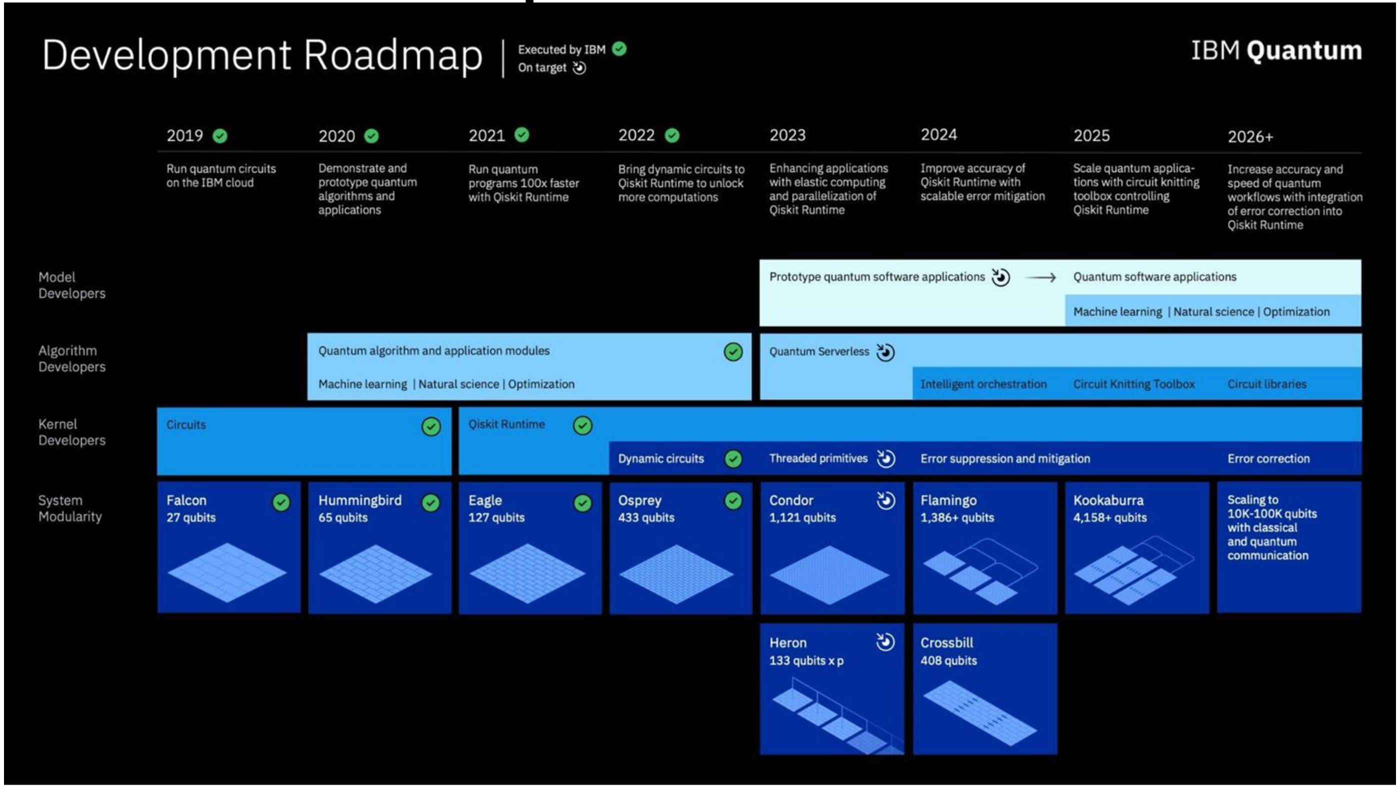


Hamiltonian Cycle Problem

- Hamiltonian Cycle Problem is an NP-hard problem and also an NP-complete problem, as proven by Richard Manning Karp in 1972.
- NP-hard problems lack efficient classical solutions, and algorithms solving them exhibit exponential time complexity in the worst case.
- Quantum computers leverage the superposition property of qubits to provide potential exponential speedup for solving NP-hard problems.

ACAN Laboratory

IBM Roadmap



Reference : https://www.ibm.com/quantum/roadmap

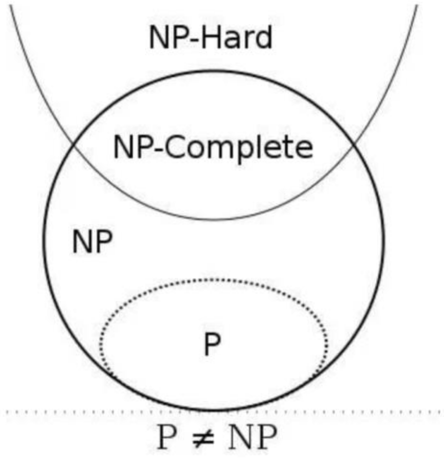
Outline

- Introduction
- Background and related knowledge
- Proposed Method
- Experiment and Result
- Conclusion

ACAN Laboratory

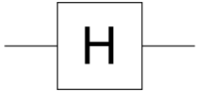
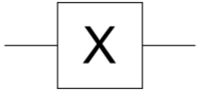
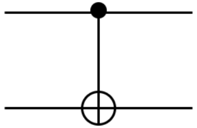
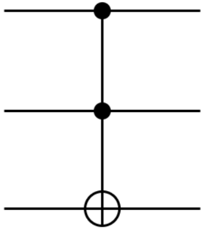
P、NP、NP-Hard、NP-Complete

- P : Problems that can be solved by deterministic algorithms in polynomial time.
- NP : Problems that can be solved by nondeterministic algorithms in polynomial time.
- NP-hard : Problems that can be polynomial time reduced to all NP problems.
- NP-complete : Problems that are both in P and NP-hard.



Reference : https://medium.com/@josh_dow/np-completeness-part-1-f3c0d4f2e0bd

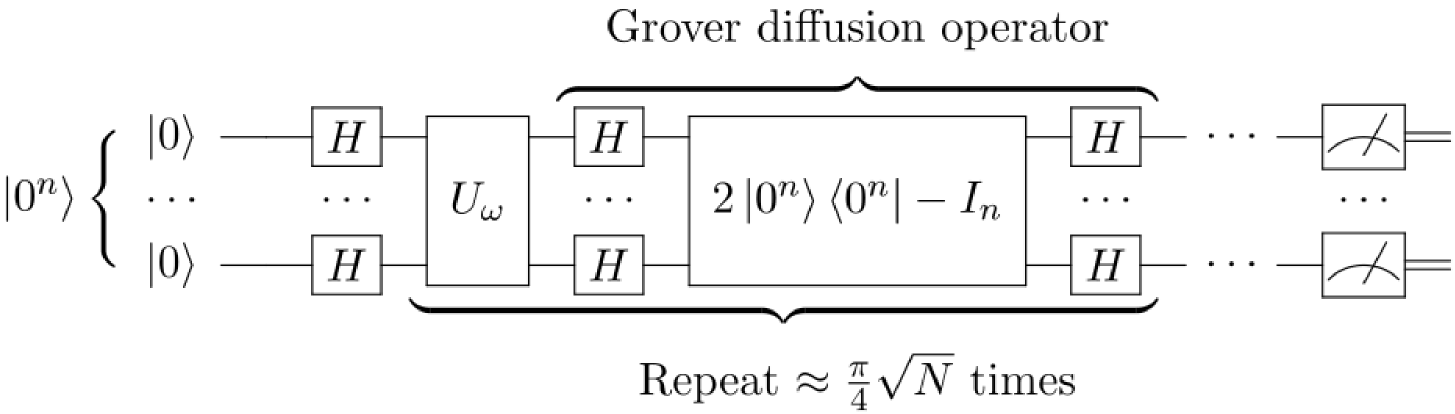
Quantum Logic Gate

Quantum Logic Gate	Icon	Matrix
Hadamard gate, H		$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Pauli-X gate, X		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Controlled not gate, CNOT		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Toffoli gate, CCX		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$

IBM Quantum Computer

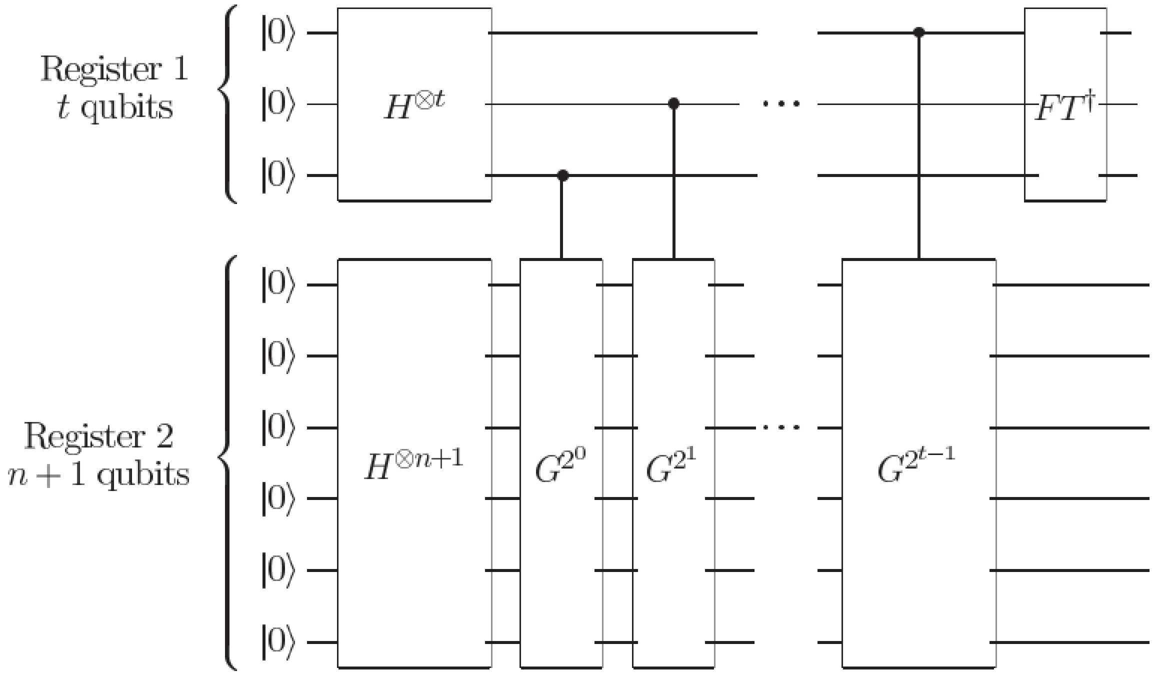
- IBM, Google, and others focus on developing quantum computers. IBM specializes in superconducting circuits, offering a high qubit count. IBM introduced Quantum Volume (QV) as a metric to measure computer capability, considering qubit count, error rates, and connectivity. Higher QV means more qubits and complex circuits.
- To achieve better results, in addition to choosing a good quantum computer, we can also obtain improved circuits by adjusting certain parameters during compilation, such as optimization levels and layout methods.
- Optimization levels range from 0 to 4. Common layout methods include Trivial Layout, Dense Layout, Noise Adaptive Layout, and Sabre Layout.

Grover’s Algorithm



Reference：https://en.wikipedia.org/wiki/Grover%27s_algorithm#/media/File:Grover's_algorithm_circuit.svg

Quantum Counting



Reference :
Nielsen, M.A. and Chuang, I.L. Quantum computation and quantum information. Cambridge University Press, 2000. Chapter 6.

Related work

Several researches have been conducted based on Grover's algorithm to design quantum circuits for implementing quantum algorithms to solve different problems. Below, we list these problems.

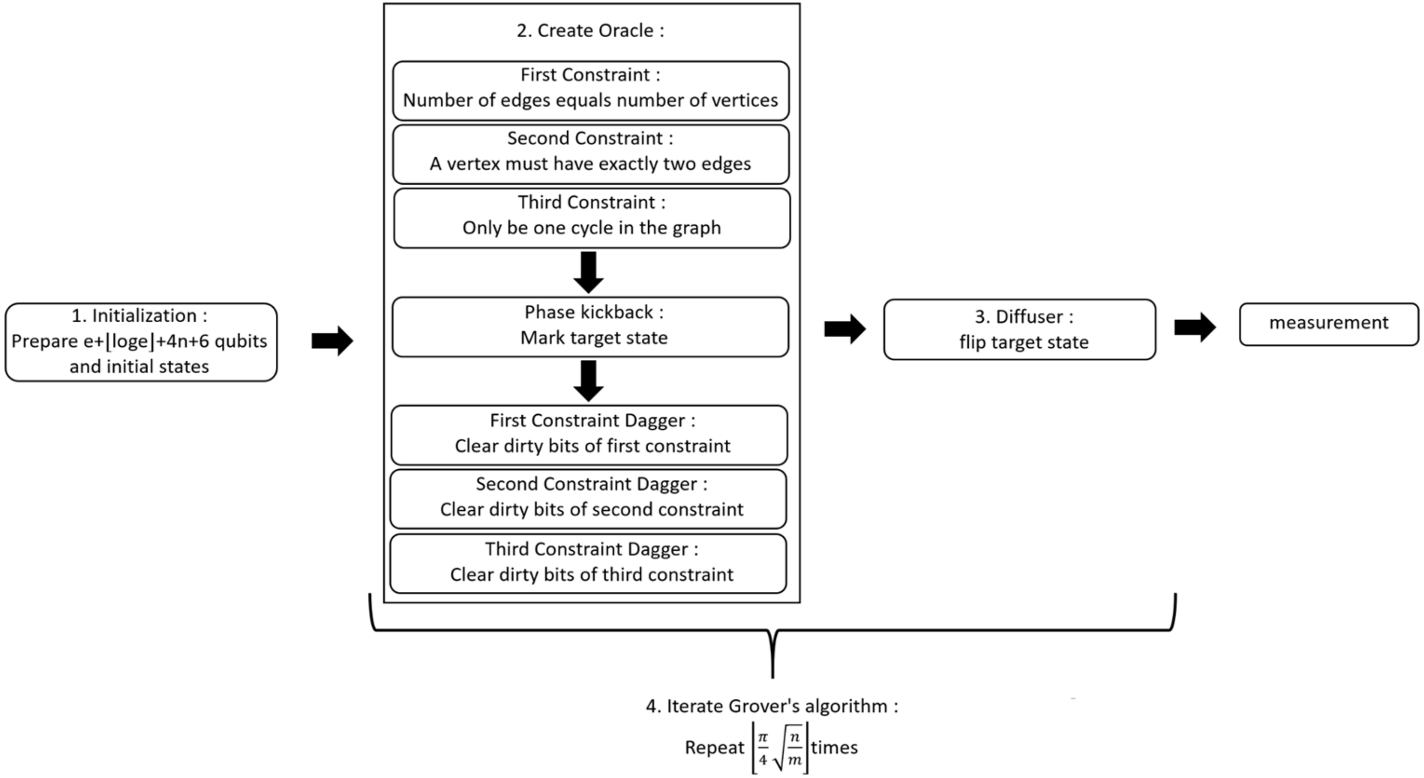
1. K-coloring problem[1]
2. Chromatic Number problem[2]
3. Maximum Clique Problem[3]
4. List Coloring Problem[4]
5. Pure Nash Equilibria in Graphical Games[5]
6. Maximum Satisfiability[6]
7. Hamiltonian Cycle Problem[7]
8. Pharmaceutical Patent Search[8]

ACAN Laboratory

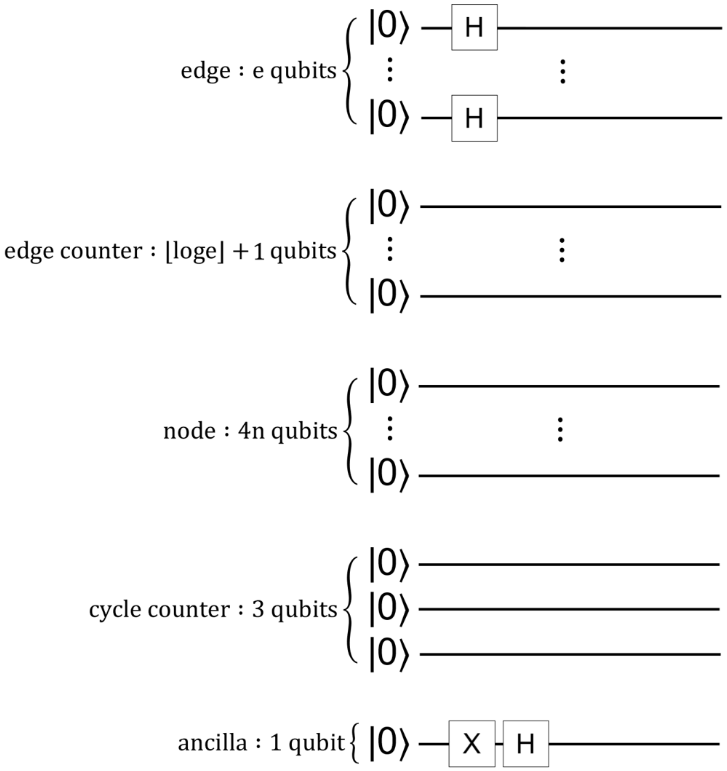
Outline

- Introduction
- Background and related knowledge
- Proposed Method
- Experiment and Result
- Conclusion

Proposed Method



Initialize Qubits

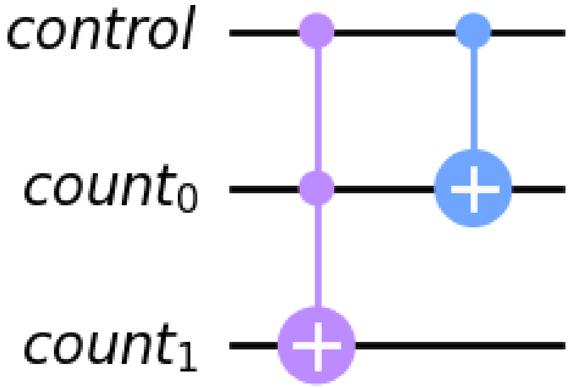


Qubits : e+⌊log e⌋+4n+5

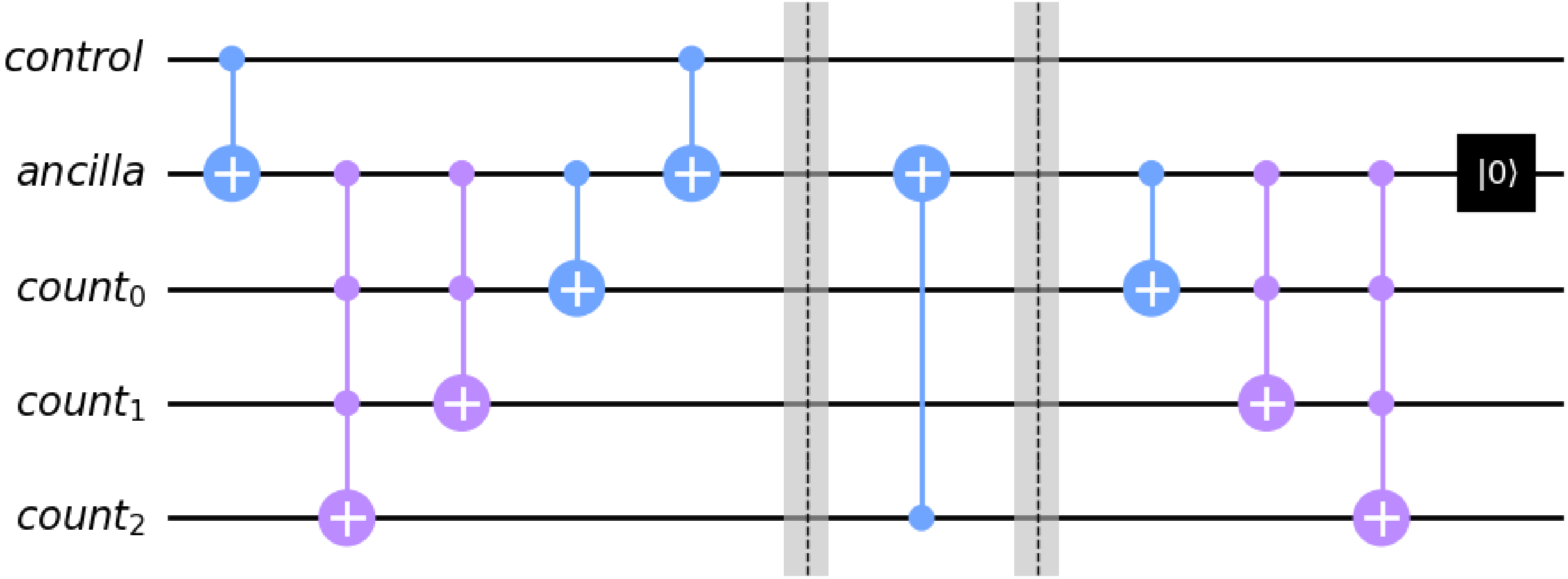
Three Constraints

1. Number of edges equals number of vertices
2. A vertex must have exactly two edges
3. Only one loop

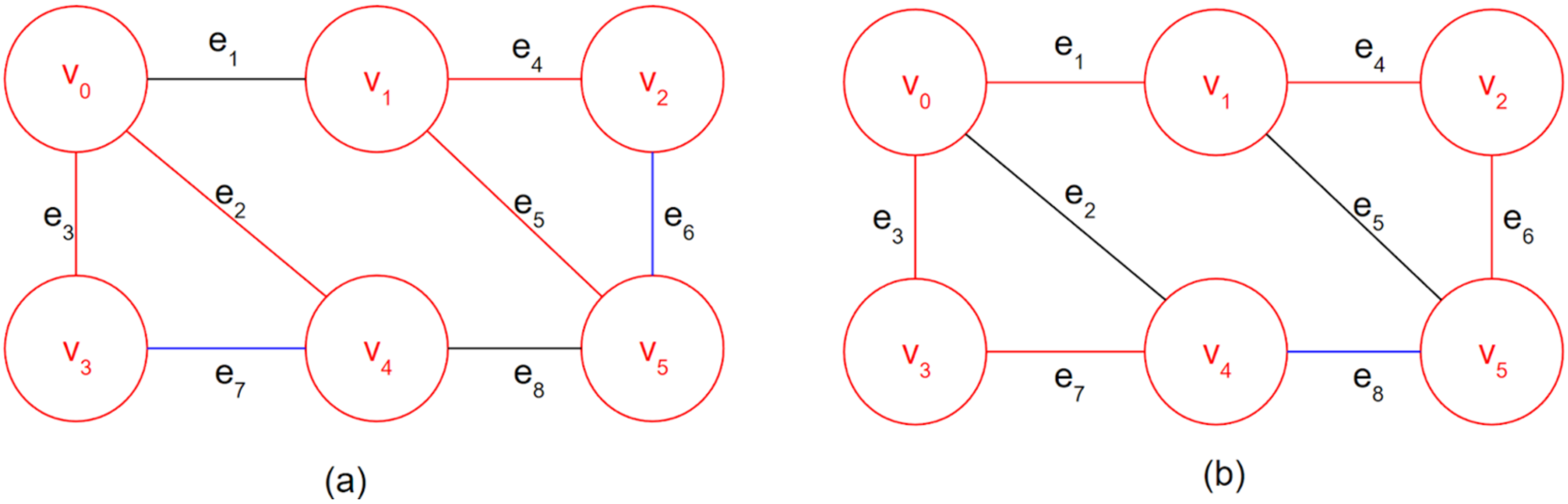
First Constraint (counter)



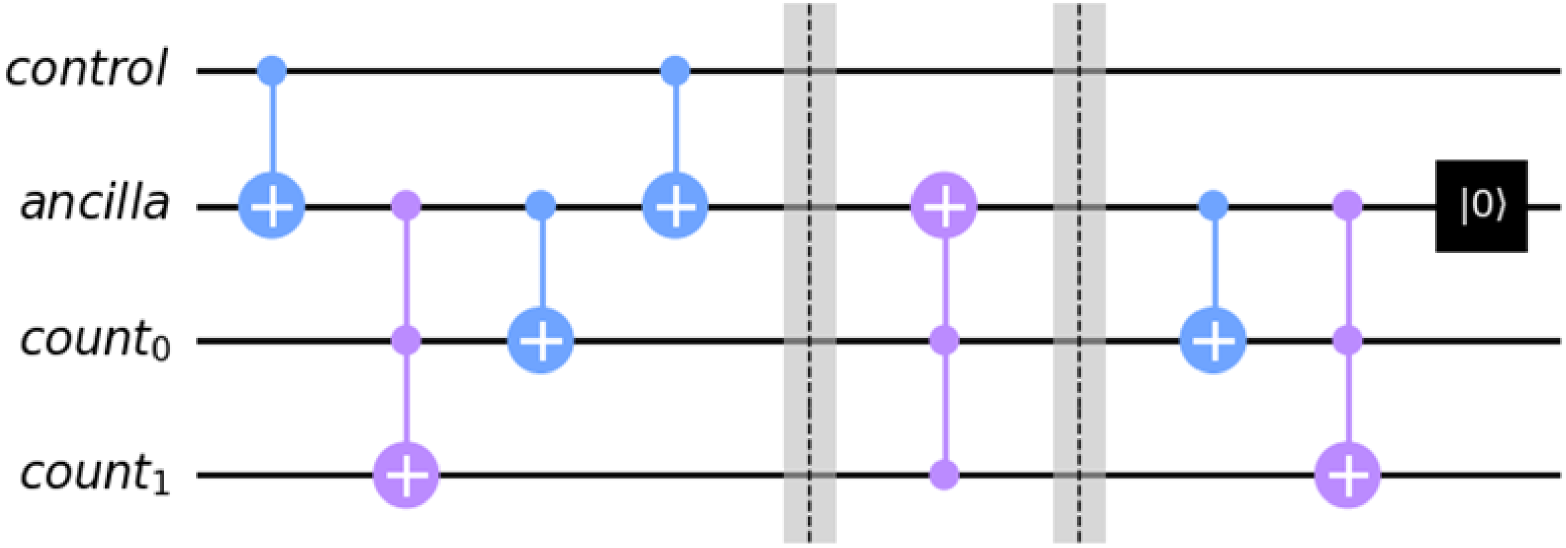
Second Constraint (max=3 range counter)



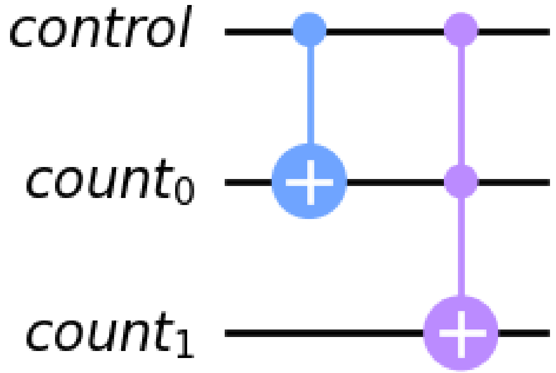
Third Constraint



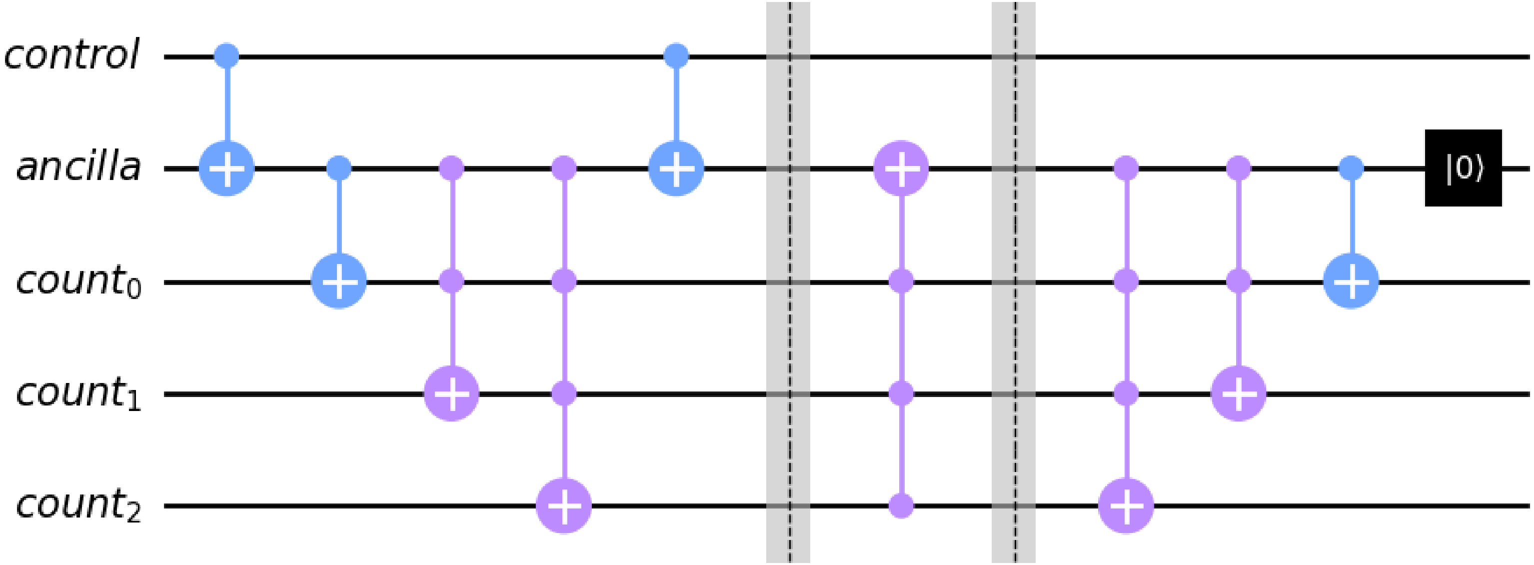
Third Constraint (max=2 counter)



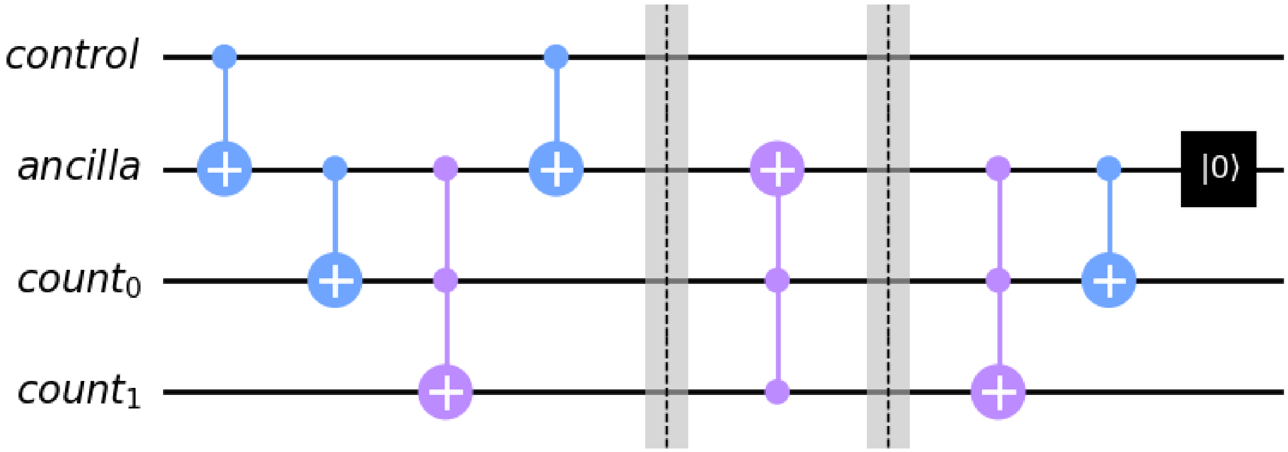
Inverse First Constraint (counter)



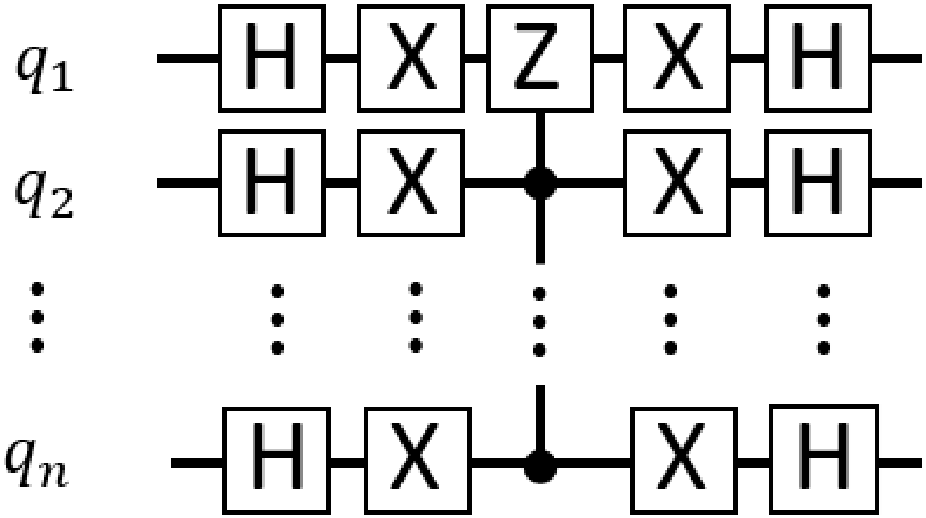
Inverse Second Constraint (3-bit min = 0 range counter)



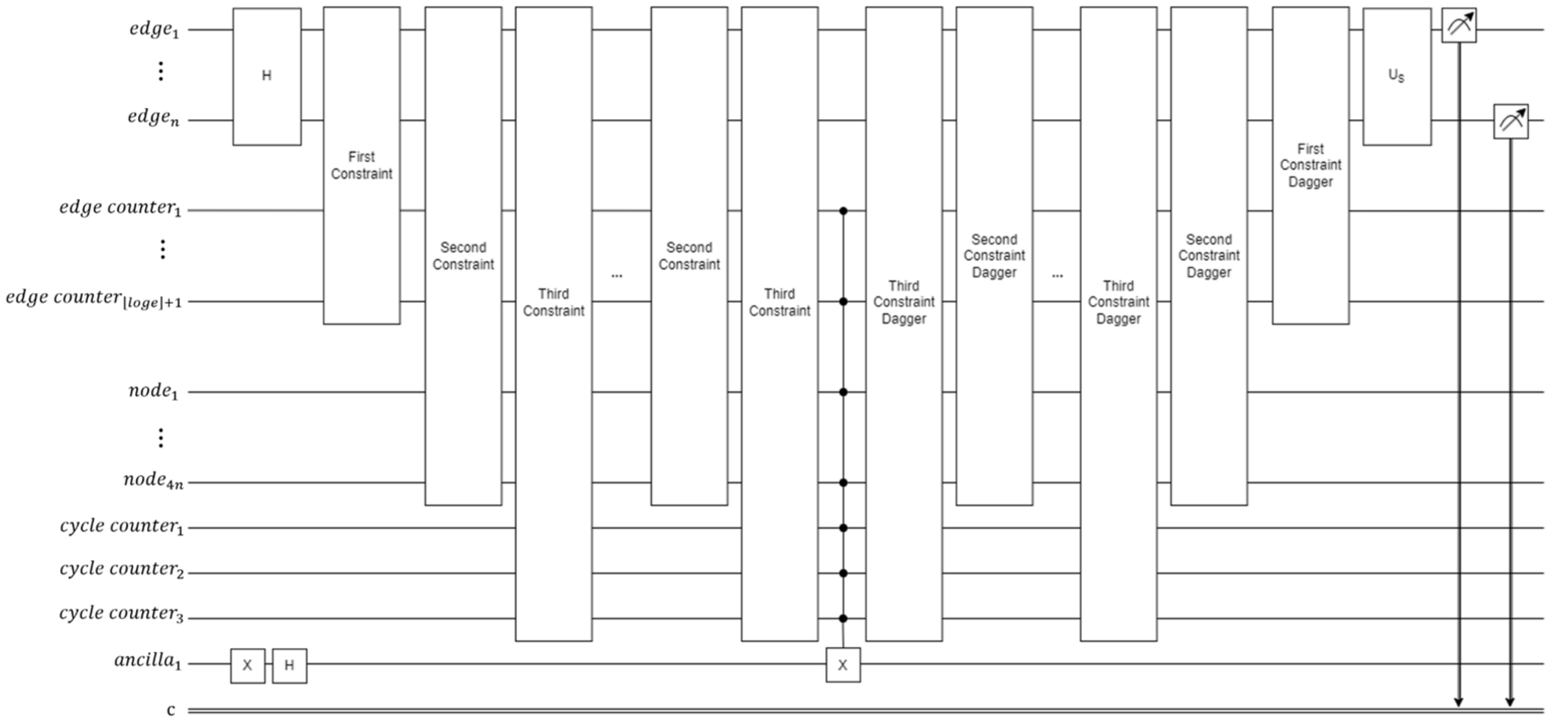
Inverse Third Constraint (2-bit min = 0 range counter)



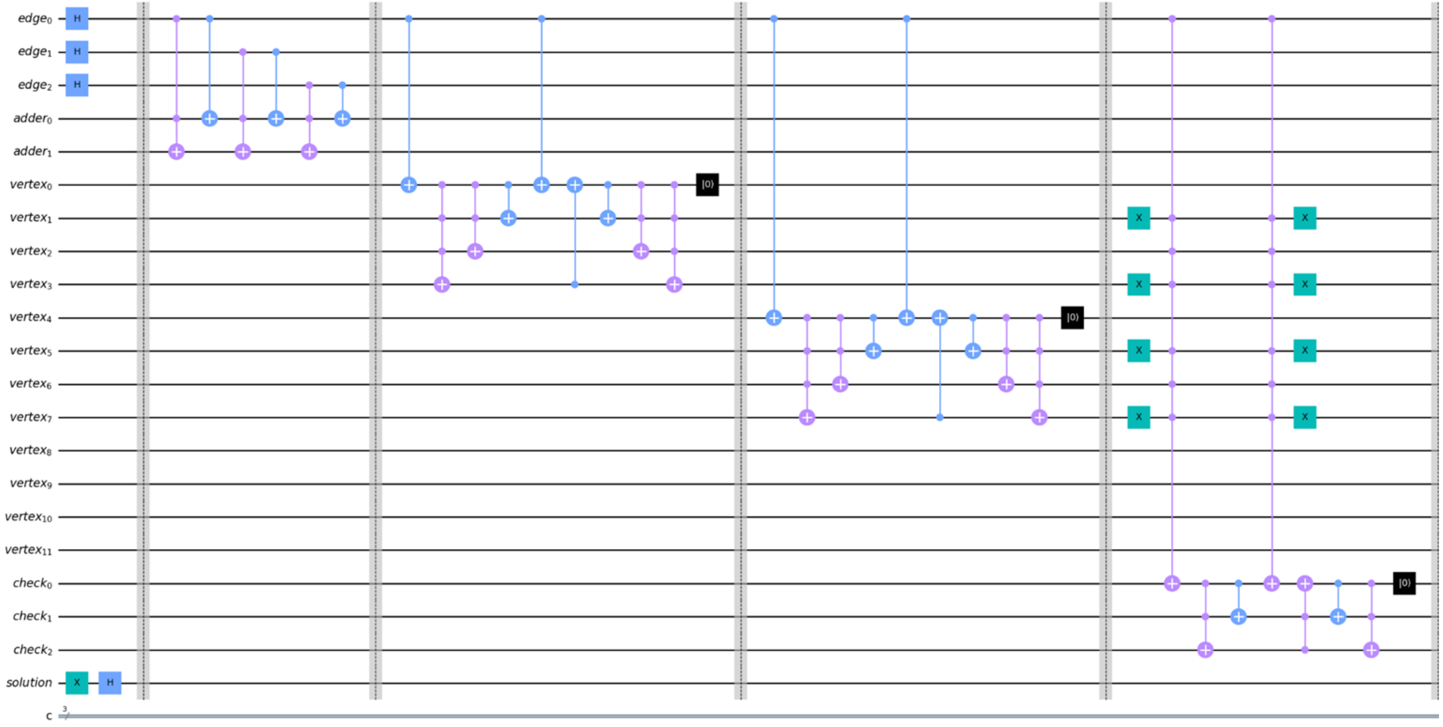
General Diffuser



Circuit Diagram

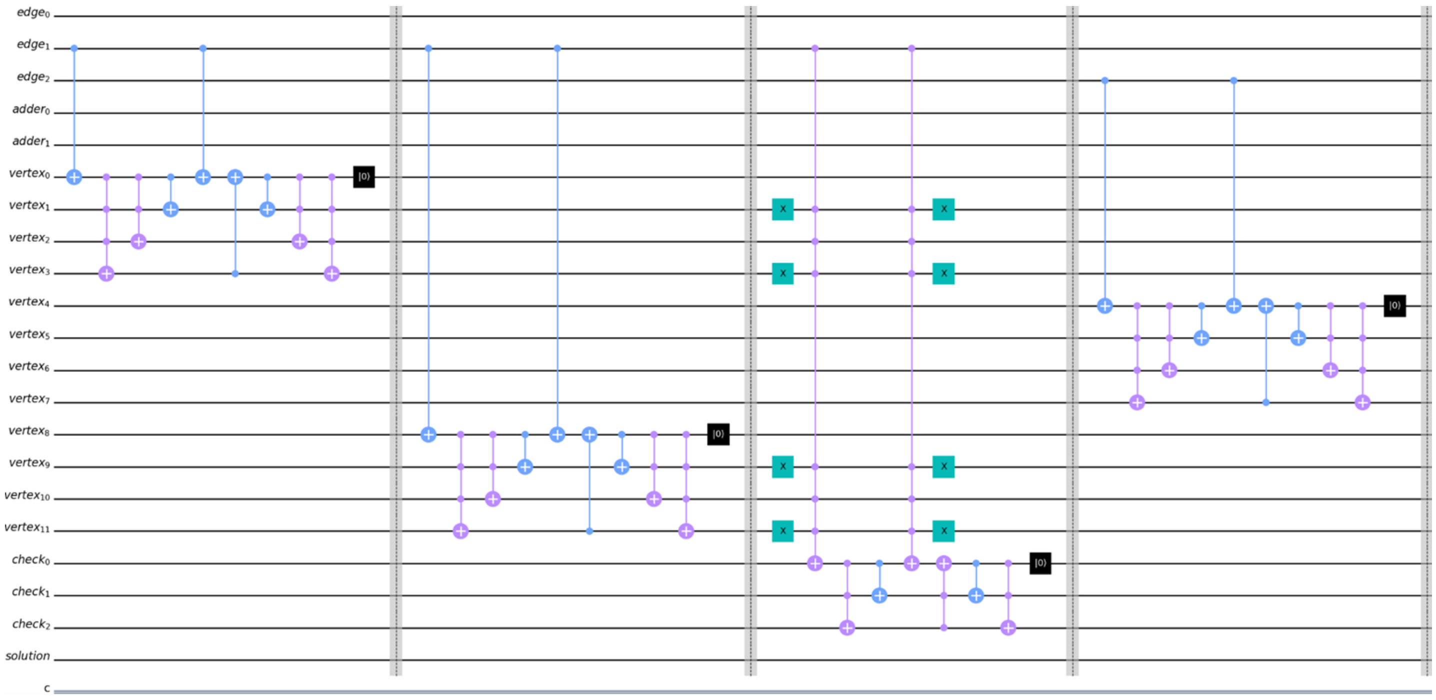


Grover's Algorithm Solves K_3 in a single iteration circuit

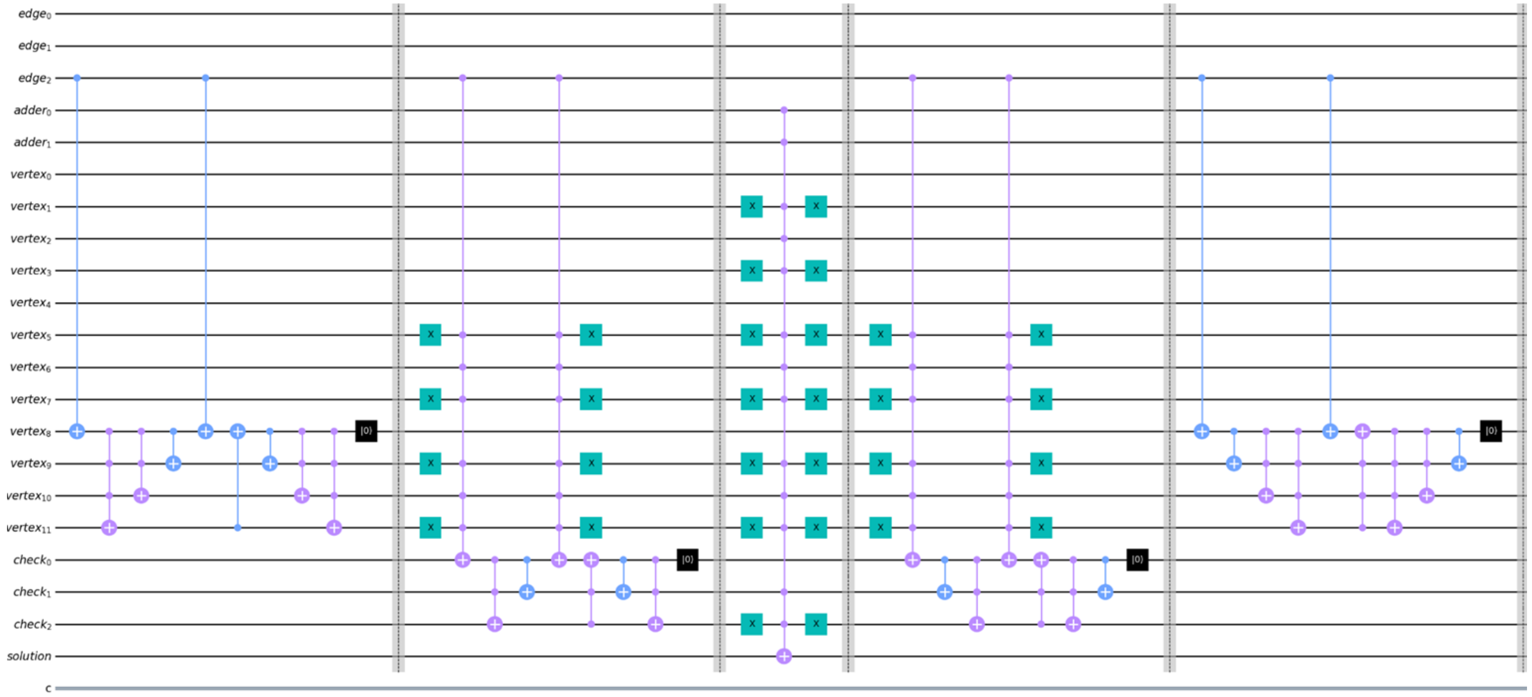


ACAN Laboratory

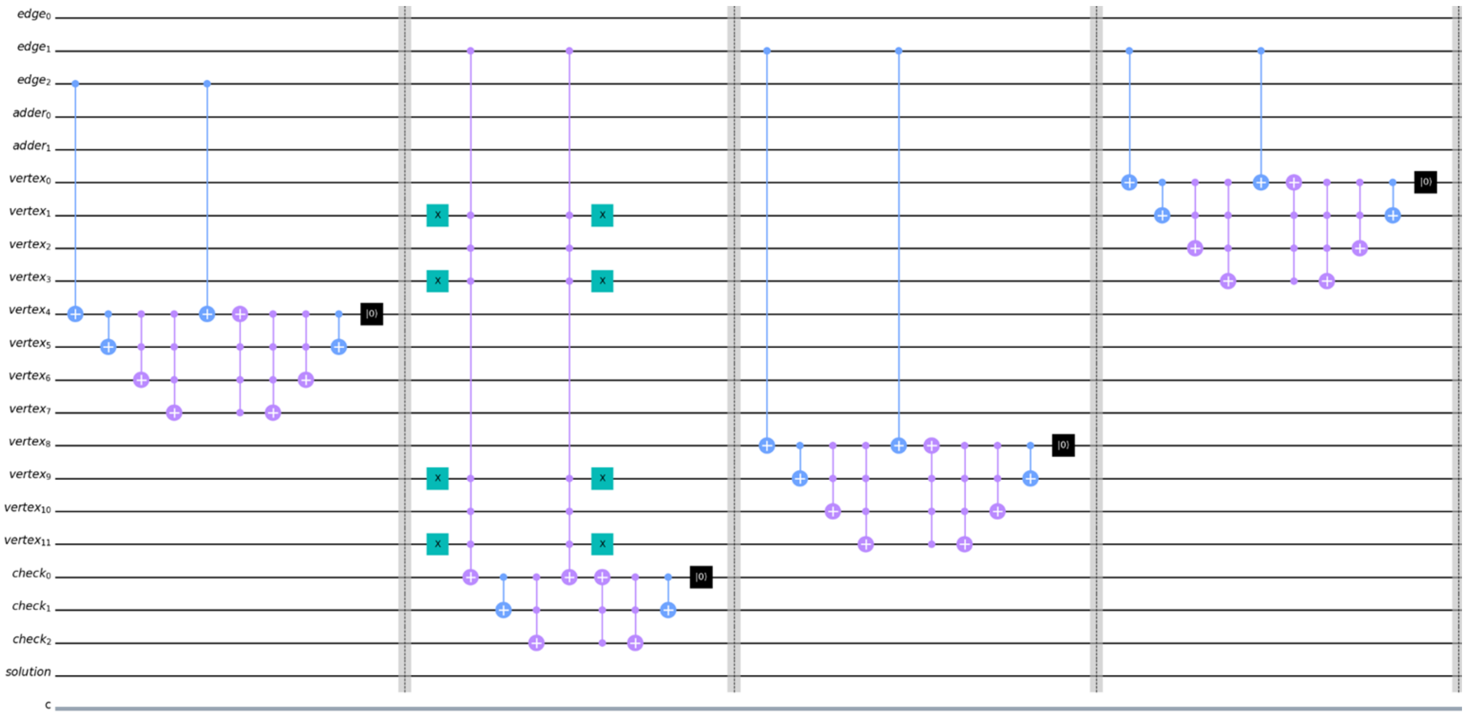
Grover's Algorithm Solves K_3 in a single iteration circuit



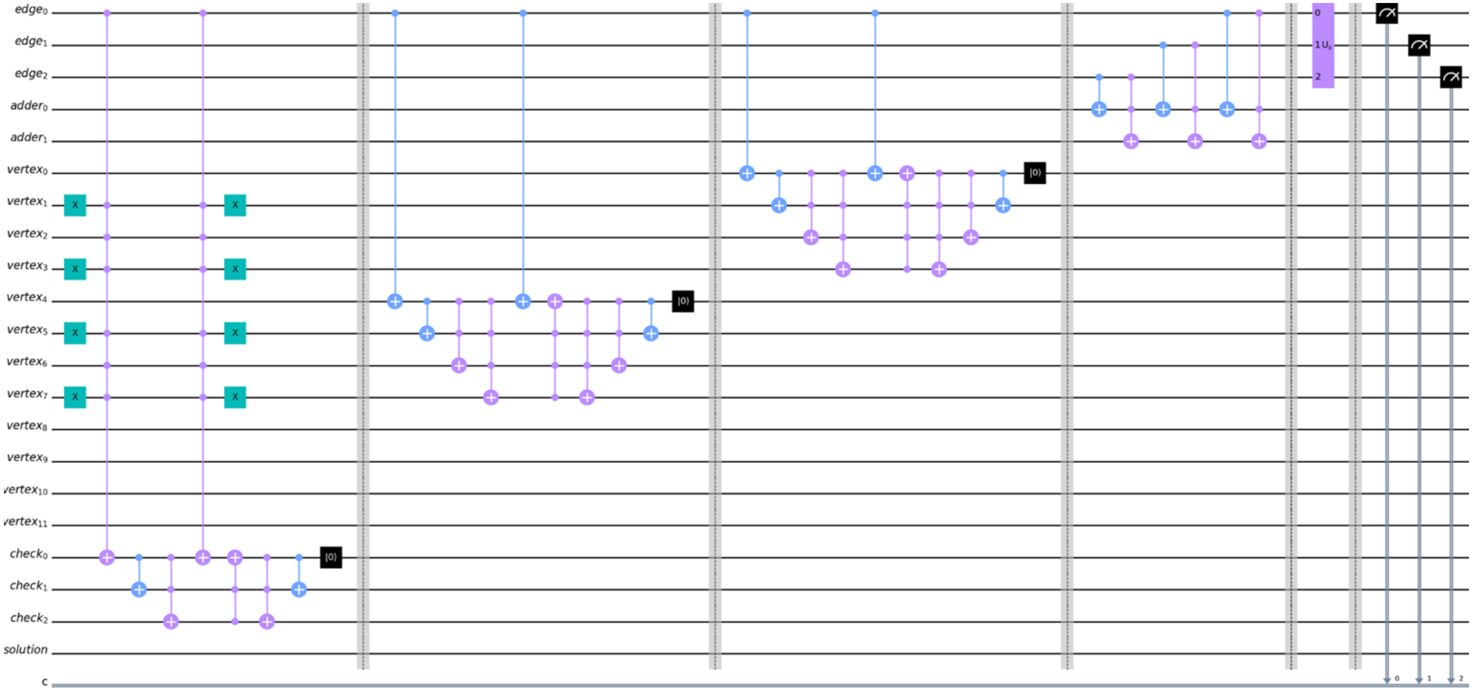
Grover's Algorithm Solves K_3 in a single iteration circuit



Grover's Algorithm Solves K_3 in a single iteration circuit



Grover's Algorithm Solves K_3 in a single iteration circuit

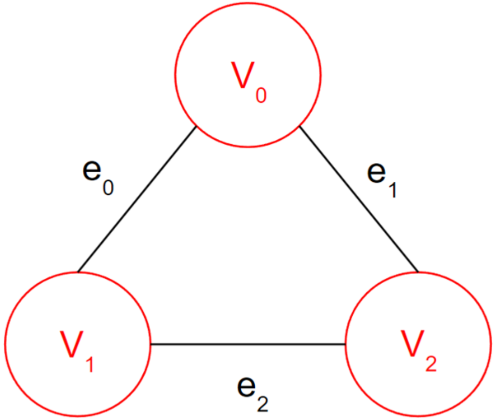


Outline

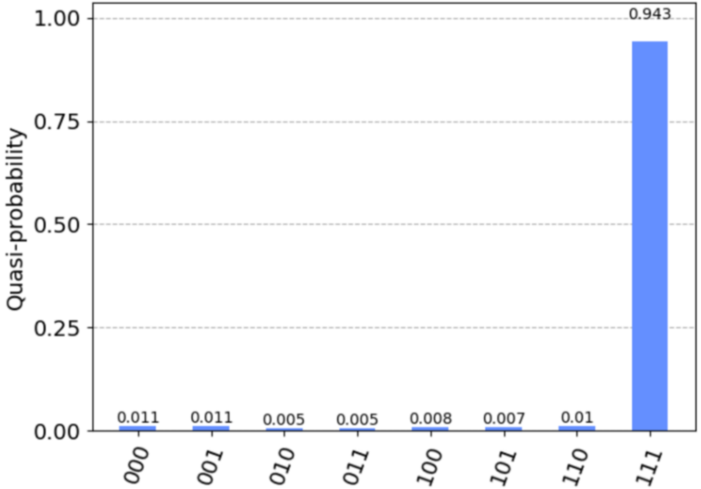
- Introduction
- Background and related knowledge
- Proposed Method
- Experiment and Result
- Conclusion

ACAN Laboratory

K_3 Using Aer Simulator

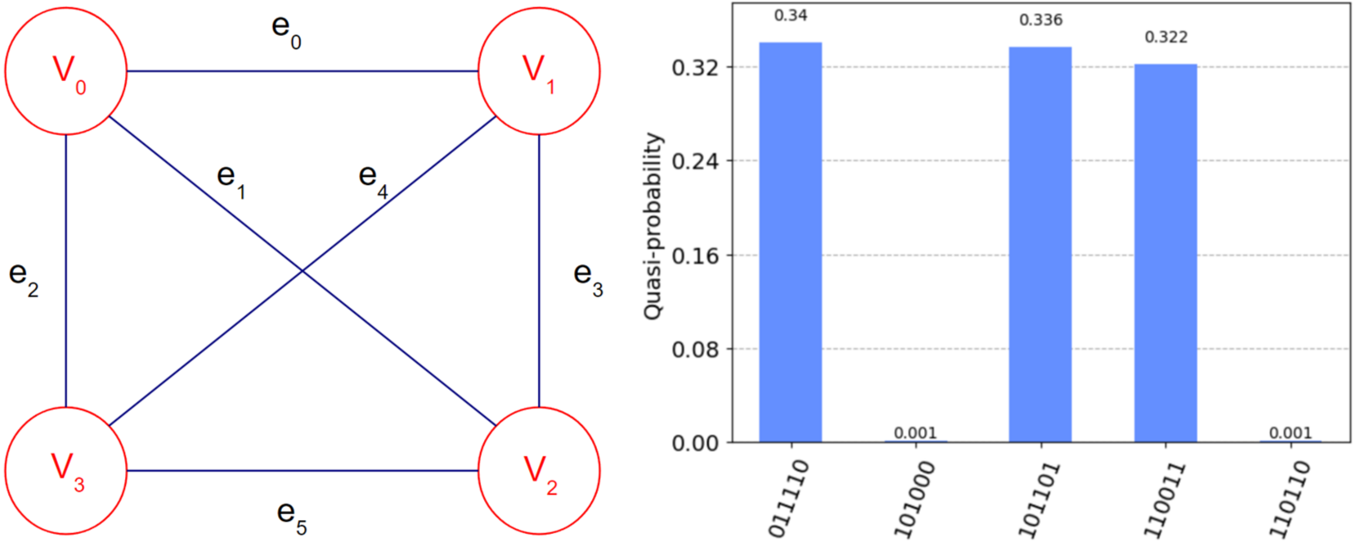


(a)



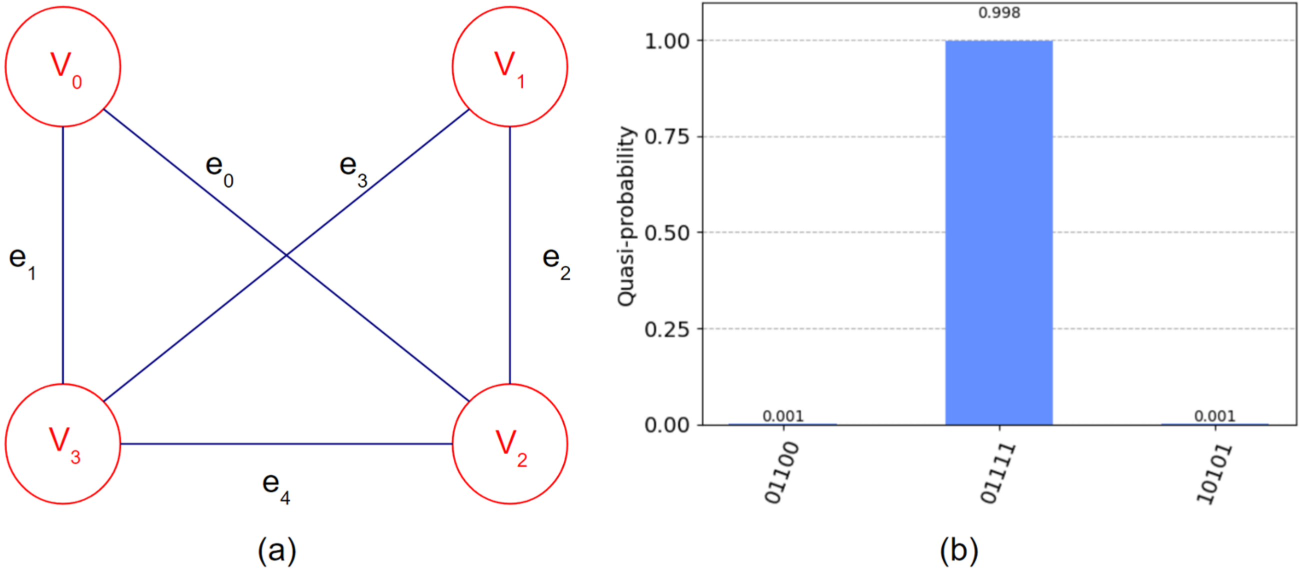
(b)

K_4 Using Aer Simulator

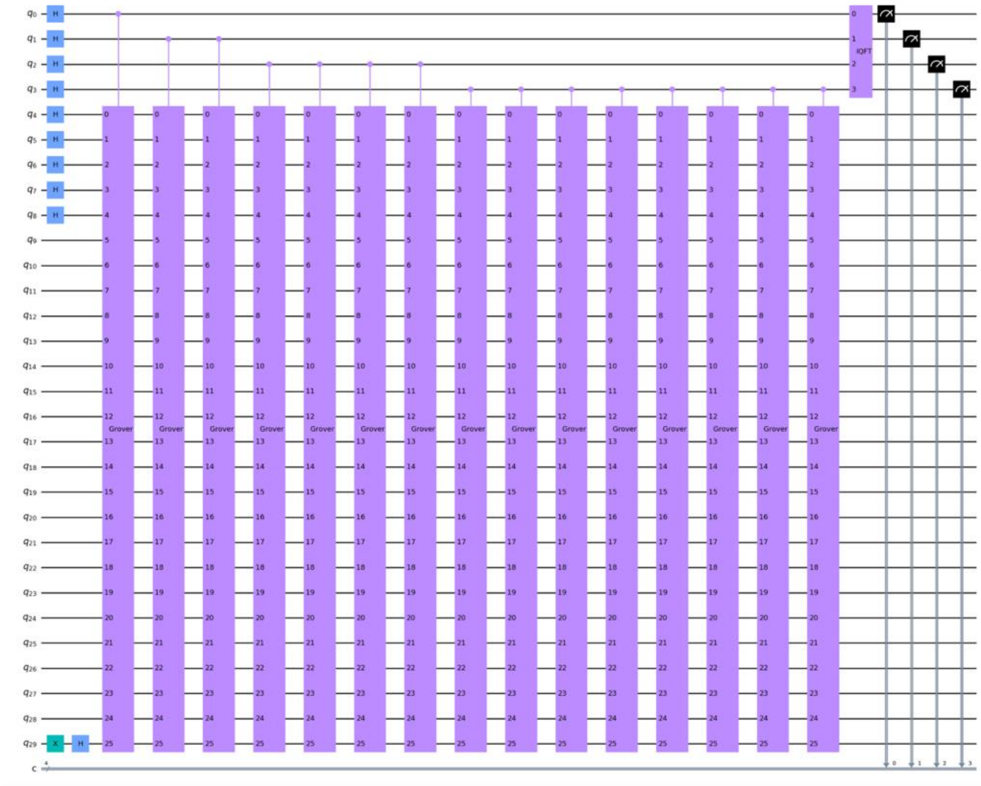


(a) (b)

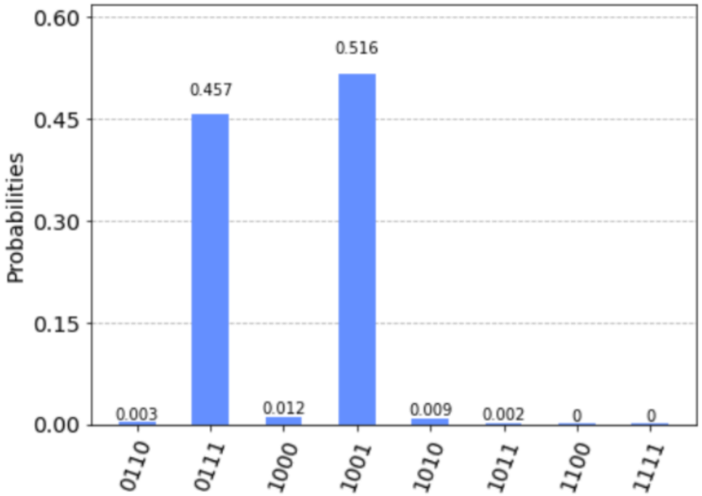
H_4 Using Aer Simulator



Quantum counting with H_4



Quantum counting with H_4

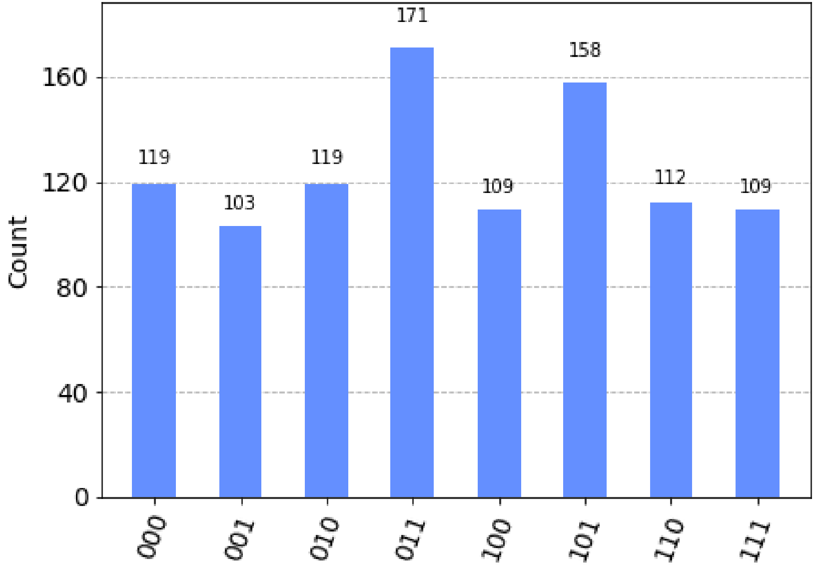


$$\theta = measured\ value * \frac{2\pi}{2^t}$$

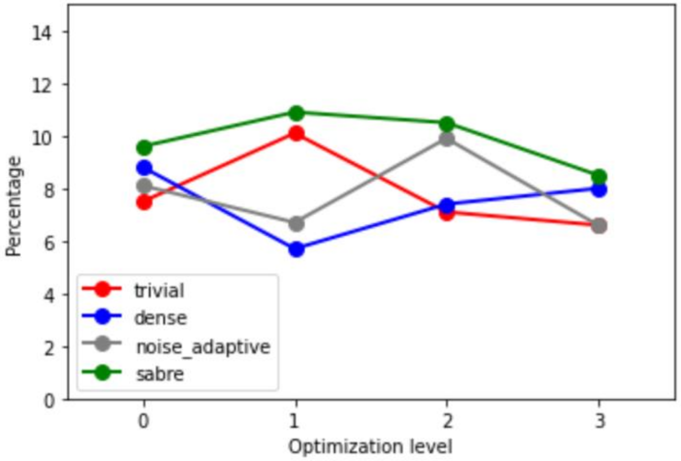
$$N\cos^2\frac{\theta}{2} = M$$

K_3 Using ibmq_mumbai

Quantum computer	ibmq_mumbai
Qubits	27
QV	128
Processor type	Falcon r5.10
Features	OpenQASM3
Basis gates	CX, ID, RZ, SX, X



K_3 with different optimization_level & layout method



optimization level	depth	cx	rz	sx	x	reset	measure
0	155701	142870	60880	412	141	36	3
1	156004	143967	60505	340	128	36	3
2	155990	143981	60448	340	128	36	3
3	161353	120393	90074	43758	5361	36	3

ACAN Laboratory

Compare with classical algorithms

Method	Time complexity	Difference
Dynamic Programming[9]	$O(n^2 2^n)$	All Hamiltonian paths can be found and listed, and whether there is a Hamiltonian cycle can be judged
Parity check[10]	$O(1.619^n)$	Only applicable to directed graphs, several solutions can be found, but no combination of solutions can be found
Improved Eppstein's Algorithm[11]	$O(1.251^n)$	Only applicable to graphs with Max degree < 3
Monte-Carlo[12]	$O(1.657^n)$ $o(1.415^n)$(bipartite graph)	It can only determine whether a Hamiltonian cycle exists in the graph, but not the number of solutions.
Proposed Method	$O(\sqrt{2^e})$	Can find and list all Hamiltonian cycles

Comparisons with the classical dynamic programming algorithm

Method	Time complexity	Space complexity	Difference
Dynamic Programming[9]	$O(n^2 2^n)$	$O(n2^n)$	All Hamiltonian paths can be found and listed, and whether there is a Hamiltonian cycle can be judged.
Proposed Method	$O(\sqrt{2^e})$	$O(e+\lfloor \log e \rfloor+4n+5)$	Can find and list all Hamiltonian cycles.

ACAN Laboratory

Outline

- Introduction
- Background and related knowledge
- Proposed Method
- Experiment and Result
- Conclusion

ACAN Laboratory

Conclusion

- We proposed a quantum circuit for solving the Hamilton cycle problem using Grover's algorithm.
- Three constrains are reflected in the oracle of Grover’s algorithm
  - Number of edges equals number of vertices: with quantum counter
  - A vertex must have exactly two edges: with a quantum range counter of max=3
  - Only one loop: with a quantum range counter of max=2
- The experimental results show that our method can find the solution of the Hamilton cycle problem with a time complexity of O($\sqrt{2^e}$) of oracle inquiries in the Aer simulator, but in the quantum computer, due to the low fidelity of the current quantum computer, it cannot find the correct solution.

Future Work

- Reduce the error rate of quantum computers through techniques such as quantum error correction codes.

- Use Qiskit Runtime Estimator to estimate expectation values of quantum circuits and observables.

- Find methods to reduce circuit depth, enabling the proposed approach in this paper to achieve accurate results in quantum computers.

ACAN Laboratory

Q & A
THANK YOU FOR LISTENING !

Appendix

Dirac notation

bra	ket
$\langle\psi\| = (\psi_1^*,\ \psi_2^*,\ \psi_3^*, \dots ,\ \psi_4^*)$	$\|\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_n \end{pmatrix}$

Optimization Level

- Optimization Level 0 : just maps the circuit to the backend, with no explicit optimization (except whatever optimizations the mapper does).
- Optimization Level 1: maps the circuit, but also does light-weight optimizations by collapsing adjacent gates.
- Optimization Level 2 : medium-weight optimization, including a noise-adaptive layout and a gate-cancellation procedure based on gate commutation relationships.
- Optimization Level 3: heavy-weight optimization, which in addition to previous steps, does resynthesis of two-qubit blocks of gates in the circuit.

Reference :
https://qiskit.org/documentation/tutorials/circuits_advanced/04_transpiler_passes_and_passmanager.html#Preset-Pass-Managers

Layout Method

- Trivial Layout : Map each virtual qubit to the same numbered physical qubit on the device, i.e. [0,1,2,3,4] -> [0,1,2,3,4].
- Dense Layout : Find the sub-graph of the device with greatest connectivity that has the same number of qubits as the circuit.
- Noise Adaptive Layout : It dynamically adjusts the layout based on the strength of connections between qubits and the level of noise, aiming to minimize the impact of noise on computations to the maximum extent possible.
- Sabre Layout : Selects a layout by starting from an initial random layout and then repeatedly running a routing algorithm (by default SabreSwap) both forwards and backwards over the circuit using the permutation caused by swap insertions to adjust that initial random layout.

Reference : https://qiskit.org/documentation/apidoc/transpiler.html?fbclid=IwAR3JeeAVoxWwbn-GpNK-qtAVgIMOHkD0YX2va_ahS0lZS8sCxKBUfC-Ik2o#layout-stage

ACAN Laboratory

Grover’s Algorithm

Grover's algorithm can be divided into the following four steps :

1. Initialize qubits : Initialize the system to the uniform superposition over all states

$$|\psi\rangle = \frac{1}{\sqrt{n}}(\psi_1, \psi_2, \psi_3, \dots, \psi_n)$$

2. Create oracle : An oracle is a black box that can determine if an input is the target or not. If the input is the target, the oracle marks it by flipping its phase to negative. If the input is not the target, the oracle leaves it unchanged.

$$|\psi\rangle = U_f|\psi\rangle$$

ACAN Laboratory

Grover’s Algorithm

3. Amplitude amplification : Amplification of the amplitude is achieved by phase inverting the states of all qubits, so that objects flipped to negative phase by the oracle are amplified by twice the amplitude, while other non-target amplitudes are reduced. Amplitude amplification can be achieved through a phase transformation (diffusion operator). The phase transformation formula is as follows : $U_s = 2|\psi\rangle\langle\psi| - I$

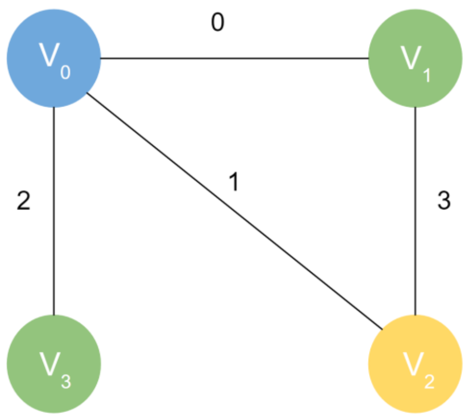
$|\psi\rangle = U_s U_f |\psi\rangle$

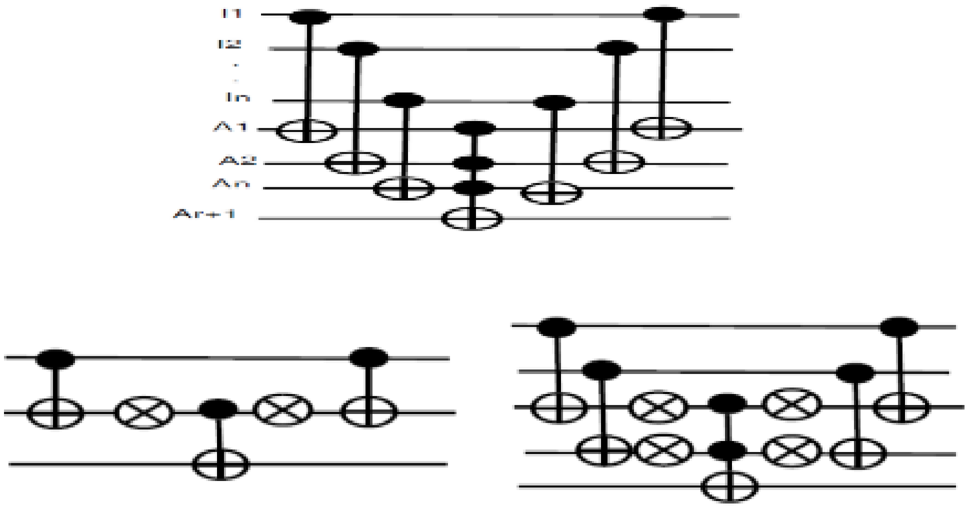
4. Iterate Grover’s algorithm : In the Grover's algorithm, iterations of the second and third steps are performed until the highest measurement result for the target is achieved. The number of iterations can be calculated using the formula $k = \lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \rfloor$

$|\psi_k\rangle = U_s U_f U_s U_f \dots U_s U_f |\psi\rangle$

ACAN Laboratory

K-coloring problem

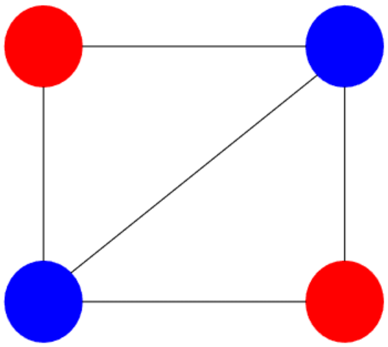


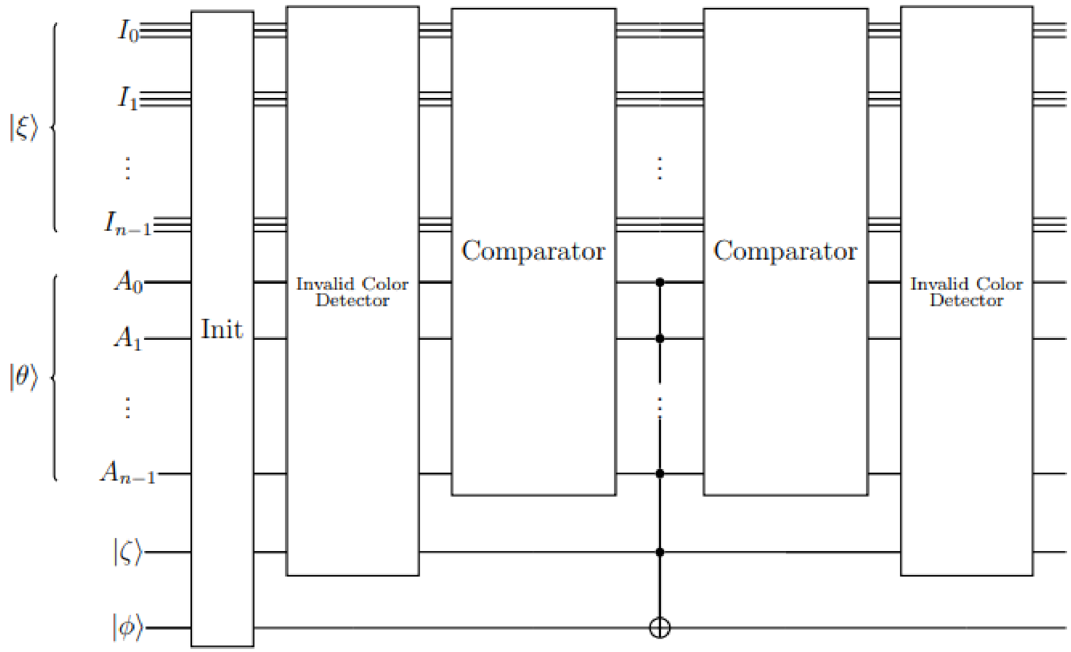


(a)

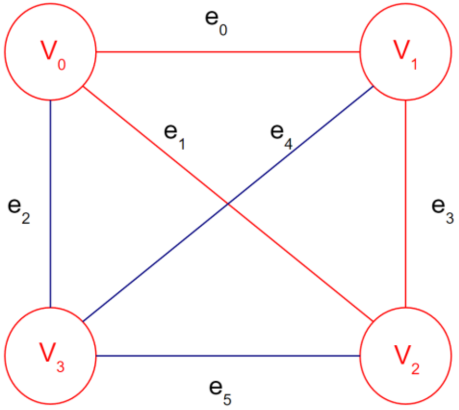
(b)

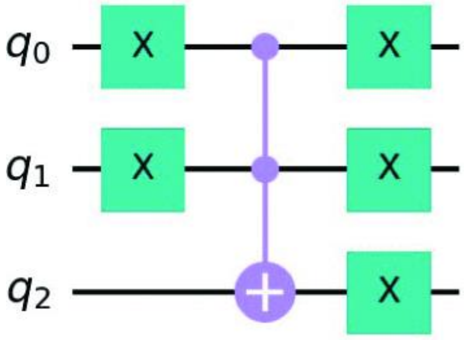
Chromatic Number problem

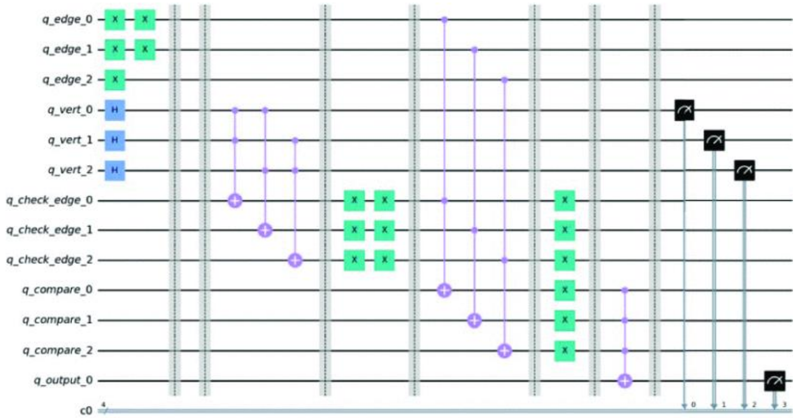




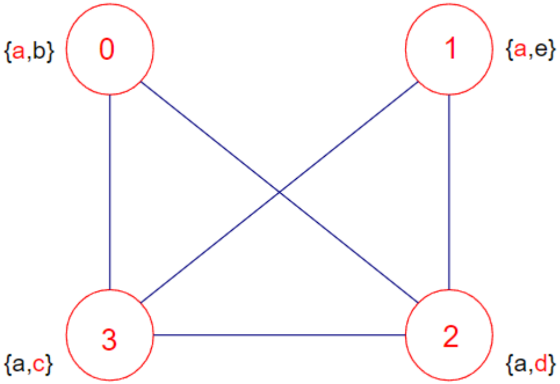
Maximum Clique Problem

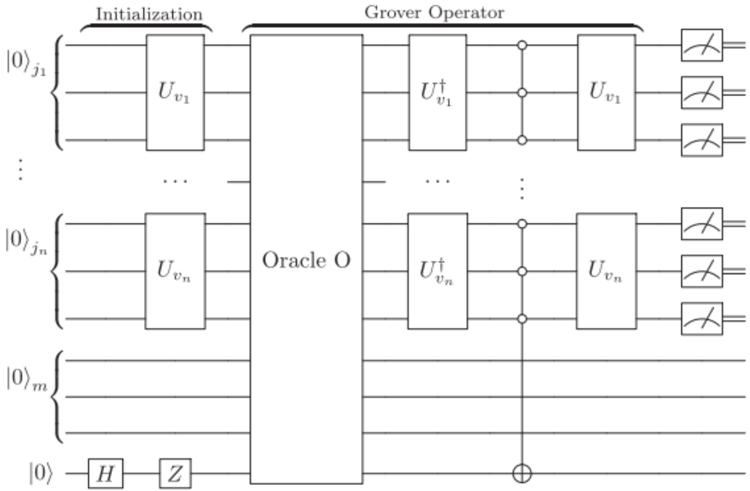




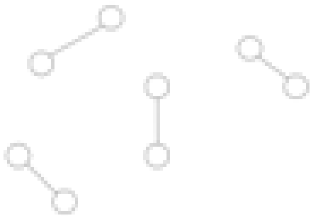


List Coloring Problem





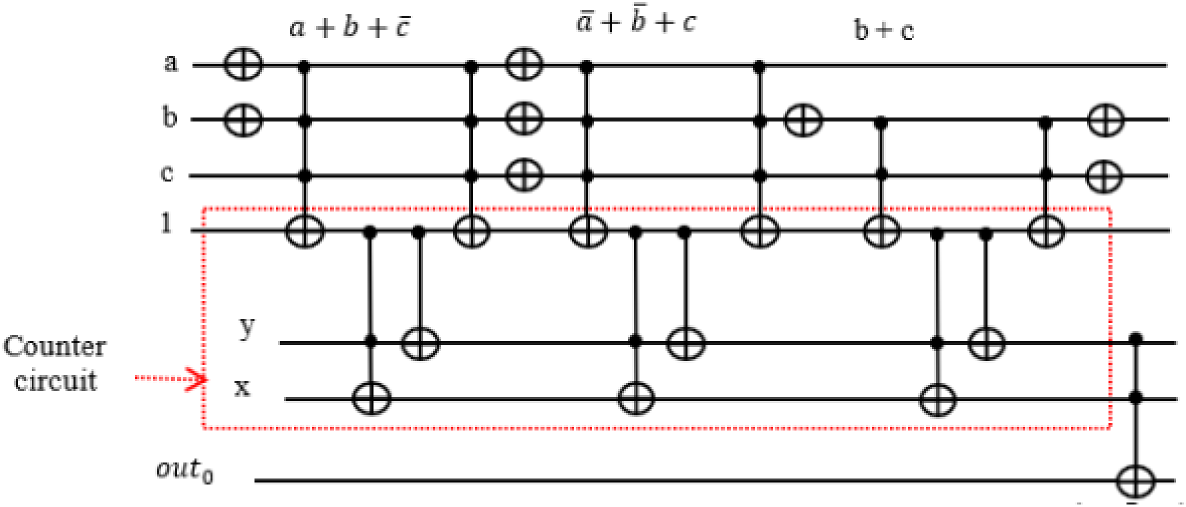
Pure Nash Equilibria in Graphical Games



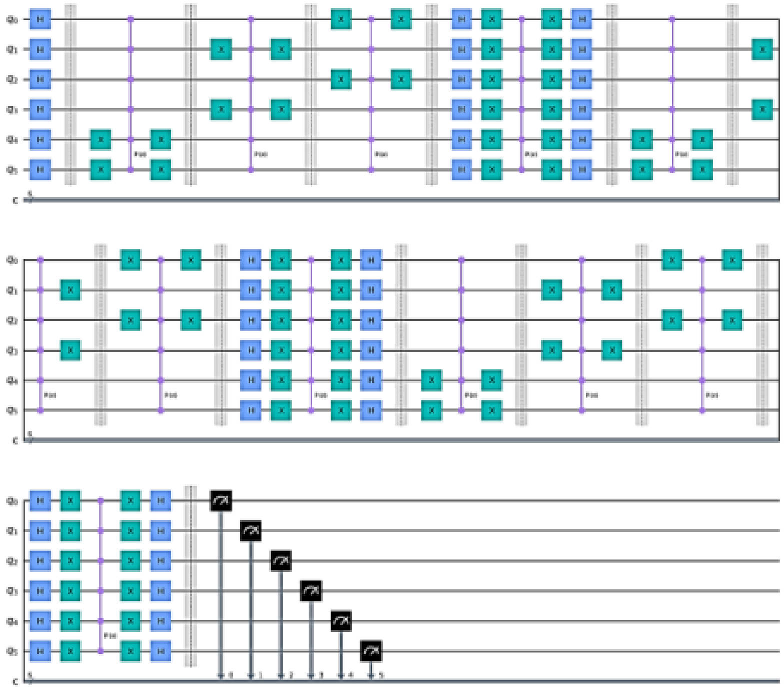
$$O_{oracle} = O_{intra} \wedge O_{inter}$$

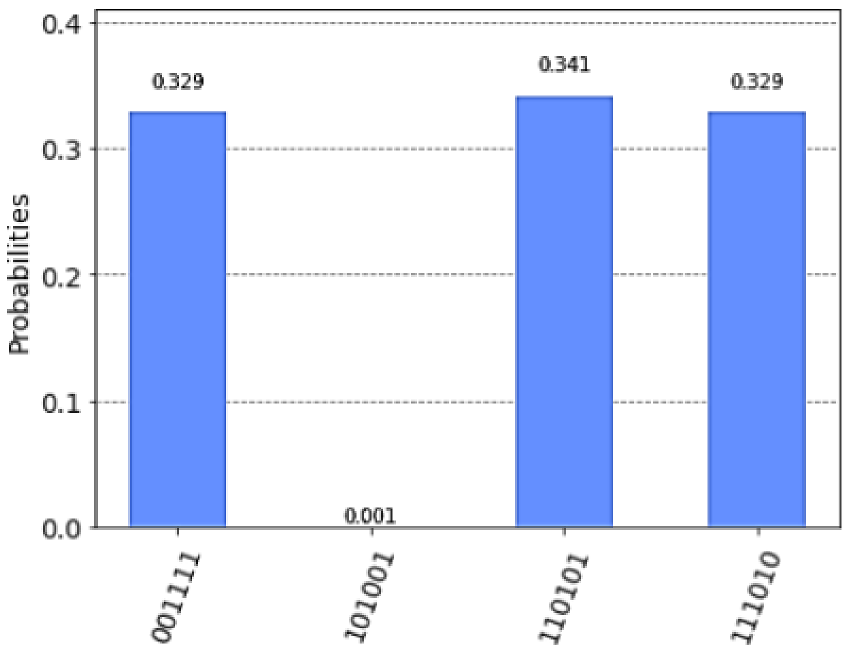
$$O_{intra} = O^{A}_{intra} \wedge O^{B}_{intra} \wedge \ldots O^{N}_{intra}$$

Maximum Satisfiability

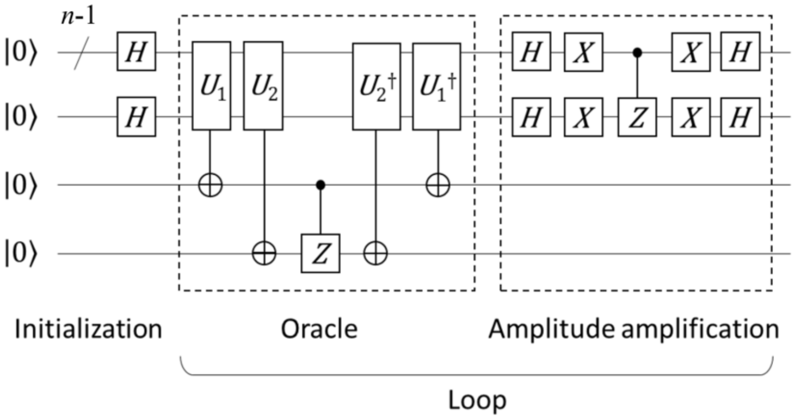


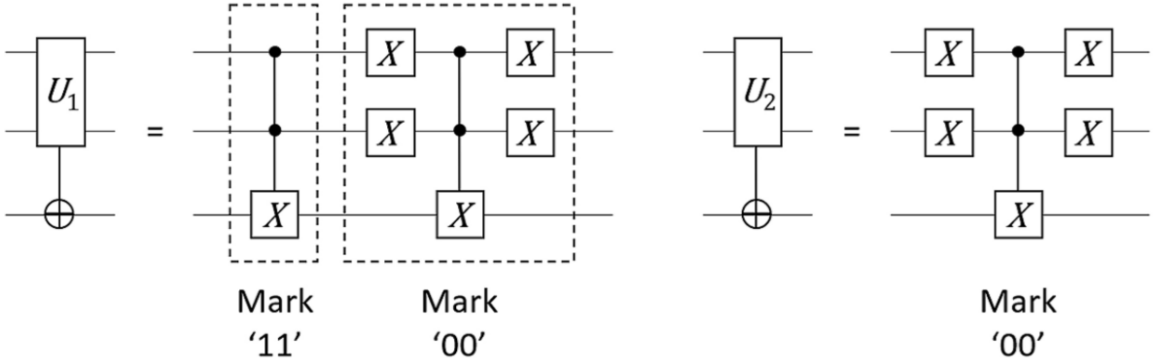
Hamiltonian Cycle Problem



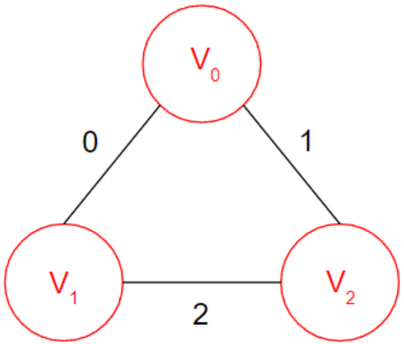


Pharmaceutical Patent Search





Dynamic Programming



vertex/ mask	0	1	2	3	4	5	6	7
0	0	1	0	1	0	1	0	1
1	0	0	1	1	0	0	1	1
2	0	0	0	0	1	1	1	1

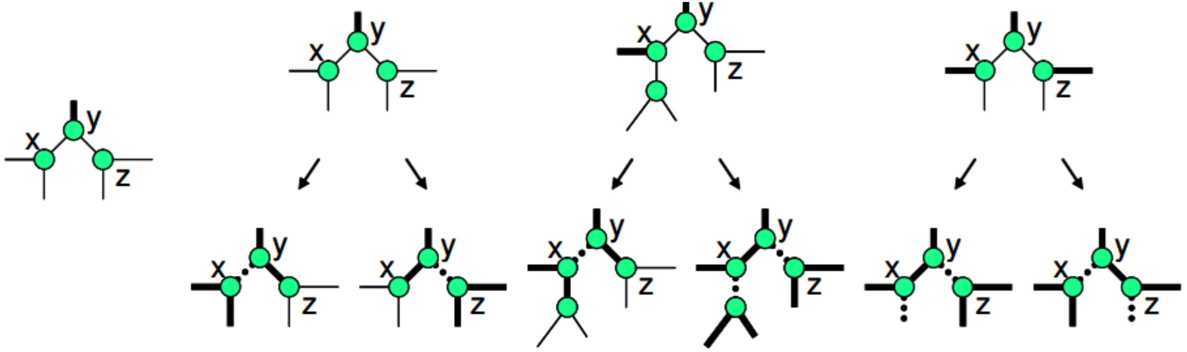
Parity check

$$\oplus \mathscr{H} = \frac{1}{2} \sum_{X,Y,Z} \left(\prod_{x \in X} d_x(X) \right) \left(\prod_{y \in Y} d_y(Y) \right) \left(\prod_{z \in Z} d_z(\overline{Z}) \right) \pmod 2$$

Algorithm P (Parity.) Given a directed graph G = (V, E), computes $\oplus$ H.
P1 [Initialise.] Set s = 0.
P2 [Locate.] List every $X \subseteq V$ such that $\prod_{x \in X} d_x(X)$ is odd.
P3 [Contribute.] Compute f(X) (mod 2) for every such X and add it to s.
P4 [Report.] Return s (mod 2).

ACAN Laboratory

Improved Eppstein's Algorithm



(a) (b) (c) (d)

$$T(n) \leq 2T(n-3)$$
$$T(n) \leq T(n-2) + T(n-5)$$
$$T(n) \leq 2T(n-4)$$

$$T(n) \leq 2T(n-1)$$
$$T(n) \leq 3\,T(n-2)$$

Monte-Carlo

I $\Lambda(D, V_2 \cup L_m, f) = \sum_{H \in hc^m_{V_2}(G)} \left(\sum_{\sigma:\mathcal{U}(H) \to L_m} \prod_{uv \in \mathcal{U}(H)} x_{uv,\sigma(uv)} \right) \left(\prod_{uv \in \mathcal{L}(H)} x_{uv} \right)$

with σ *one-to-one.*

II $\Lambda(D, V_2 \cup L_m, f)$ *is the zero polynomial if and only if* $hc^m_{V_2}(G) = \emptyset$.

$$\Lambda(D, V_2 \cup L_m, f) = \sum_{H \in hc^m_{V_2}(G)} \left(\sum_{\sigma:\mathcal{U}(H) \to L_m} \prod_{uv \in \mathcal{U}(H)} x_{uv,\sigma(uv)} \right) \left(\prod_{uv \in \mathcal{L}(H)} x_{uv} \right)$$

ACAN Laboratory

IBM Q Hub at NTU account and common account comparison

Account	Quantum computer	Max shots	Max circuits
IBM Q Hub at NTU	ibm_sherbrooke、ibm_brisbane、ibm_nazca、ibm_algiers、ibmq_kolkata、ibmq_mumbai、ibm_cairo、ibm_auckland、ibm_hanoi、ibmq_guadalupe、ibm_perth、ibm_lagos、ibm_nairobi、ibmq_jakarta、ibmq_manila、ibmq_quito、ibmq_belem、ibmq_lima	100000	300
Normal	ibm_perth、ibm_lagos、ibm_nairobi、ibmq_jakarta、ibmq_manila、ibmq_quito、ibmq_belem、ibmq_lima	20000	100

ACAN Laboratory

Quantum Superposition