Quantum Speedup for the Maximum Cut Problem*

1st* Weng-Long Chang
Department of Computer Science and Information Engineering
National Kaohsiung University of Science and Technology
Kaohsiung, Taiwan
changwl@cc.kuas.edu.tw

2nd* Renata Wong
Physics Division
National Center for Theoretical Sciences
National Taiwan University
Taipei, Taiwan
https://orcid.org/0000-0001-5468-0716

3rd Wen-Yu Chung
Department of Computer Science and Information Engineering
National Kaohsiung University of Science and Technology
Kaohsiung, Taiwan
wychung@nkust.edu.tw

4th Yu-Hao Chen
Department of Physics
National Taiwan University
Taipei, Taiwan
r10222067@ntu.edu.tw

5th Ju-Chin Chen
Department of Computer Science and Information Engineering
National Kaohsiung University of Science and Technology
Kaohsiung, Taiwan
jc.chen@nkust.edu.tw

6th Athanasios V. Vasilakos
Center for AI Research (CAIR)
University of Agder
Grimstad, Norway
thanos.vasilakos@uia.no

Abstract—Given an undirected, unweighted graph with n vertices and m edges, the maximum cut problem is to find a partition of the n vertices into disjoint subsets V_1 and V_2 such that the number of edges between them is as large as possible. Classically, it is an NP-complete problem, which has potential applications ranging from circuit layout design, statistical physics, computer vision, machine learning and network science to clustering. In this paper, we propose a quantum algorithm to solve the maximum cut problem for any graph G with a quadratic speedup over its classical counterparts, where the temporal and spatial complexities are reduced to, respectively, $O(\sqrt{2^n/r})$ and $O(m^2)$. With respect to oracle-related quantum algorithms for NP-complete problems, we identify our algorithm as optimal. Furthermore, to justify the feasibility of the proposed algorithm, we successfully solve a typical maximum cut problem for a graph with three vertices and two edges by carrying out experiments on IBM's quantum computer.

Index Terms—data structures and algorithms, the maximum cut problem, quantum algorithms, quantum computing, quantum speedup

I. INTRODUCTION

Let $G = (V, E)$ be an undirected, unweighted graph with a set of vertices V is a set of edges E. Further, let $|V| = n$ and let $|E| = m$. A cut $\{V_1, V_2\}$ of G is defined as a partition of vertices into two disjoint subsets V_1 and V_2. The size of the cut is the number of the edges between V_1 and V_2.

Example: Consider an undirected unweighted graph G that contains three vertices $\{v_1, v_2, v_3\}$ and two edges $\{(v_1, v_2), (v_2, v_3)\}$ as shown in Fig. 1. If the cut is $V_1 = \{v_1\}$ and $V_2 = \{v_2, v_3\}$, or $V_1 = \{v_1, v_2\}$ and $V_2 = \{v_3\}$, the size of the cut is 1. If it is $V_1 = \{v_1, v_3\}$ and $V_2 = \{v_2\}$, then it is 2, which is also the maximum cut size for this graph.
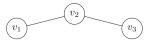

Fig. 1. Example graph.

In what follows, we assume that $X = \{x_n \cdots x_1 | x_d \in \{0, 1\}, 1 \leq d \leq n\}$ is a set of 2^n possible cuts. We further assume that x^0 indicates that $x = 0$, while x^1 indicates that $x = 1$. With this, each element in X is n bits long and represents one of the 2^n possible partitions of n vertices into two disjoint subsets V_1 and V_2. Furthermore, if an $x_d = 1$ in $x_n \cdots x_1 \in X$ then this indicates that the d-th vertex in graph G is in V_1. For the same partition, if $x_d = 0$, then this indicates that the d-th vertex is in V_2.

The fact that an edge $(x_k, x_p) \in V_1 \times V_2$ can be verified by formula (1), while the fact that an edge is not shared between V_1 and V_2 can be verified by formula (2).

$$f(x_k, x_p) = (x_k \wedge \overline{x_p}) \vee (\overline{x_k} \wedge x_p) \tag{1}$$

$$g(x_k, x_p) = (\overline{x_k} \wedge \overline{x_p}) \vee (x_k \wedge x_p) \tag{2}$$

Here, $\wedge$ stands for the logic AND, and $\vee$ stands for the logic OR operation.

II. RELATED WORKS AND MOTIVATION

Quantum computing promises to solve certain hard problems more efficiently than classical algorithms. This holds especially for the case when the input size is too large for classical algorithms to process. Some of the best known quantum algorithms that offer such a speedup are quantum integer factorization [2] which runs exponentially faster than any known classical algorithm, and quantum search algorithm [3] that offers a generic square-root speedup over classical algorithms. It has been shown [1] that classical algorithms would require $\Omega(2^n)$ queries where Grover's algorithm only requires $O(2^{n/2})$. Shor's algorithm is problem-specific. On the other hand, Grover's algorithm finds a wide range of applications as a subroutine in quantum algorithms, such as in [4], [5], [7], [9], [11].

While the opposite problem of finding a minimum cut in a graph is efficiently solvable by the Ford–Fulkerson algorithm [8], the maximum cut problem is known to be NP-hard. This means that there are no known polynomial classical algorithms for the problem in general graphs. The max-cut problem in planar graphs can be however solved in polynomial time [10]. As planar graphs constitute only a small subset of the graph family, it is vital to research on finding ways to solve the maximum cut problem efficiently.

There exist classical approximation algorithms, the best of which runs in polynomial time and has an approximation ratio of ≈ 0.878 [6]. In contrast to that, the algorithm presented in the present work is exact.

III. ALGORITHM FOR THE MAXCUT PROBLEM

A. Computing number of edges in a cut

In order to compute the number of edges in each cut, we introduce auxiliary Boolean variables $z_{i+1,j}$ and $z_{i+1,j+1}$, $1 \leq i \leq m, 0 \leq j \leq i$. $z_{i+1,j+1}$ stores the number of edges in a cut after determining the influence of bits (x_k, x_p) encoding the $(i+1)$-th edge (v_k, v_p) on the number of edges (this corresponds to the number of 1s). Hence, $z_{i+1,j+1} = 1$ indicates that there are $j+1$ edges in the cut. Likewise, $z_{i+1,j}$ stores the number of edges in a cut after determining the influence of bits (x_k, x_p) encoding the $(i+1)$-th edge (v_k, v_p) on the number of edges. $z_{i+1,j} = 1$ indicates that there are j edges in the cut.

Computing the number of edges in a cut is straightforward. To that end, an edge must satisfy the Boolean formula in (1) and the cut must have currently j edges. If both conditions are met, i.e. if the formula

$$[(x_k \wedge \overline{x_p}) \vee (\overline{x_k} \wedge x_p)] \wedge z_{i,j} \tag{3}$$

is satisfied, the number of edges in a cut is increased by 1.

Similarly, if an edge satisfies the following formula, the number of edges remains intact:

$$[(\overline{x_k} \wedge \overline{x_p}) \vee (x_k \wedge x_p)] \wedge z_{i,j} \tag{4}$$

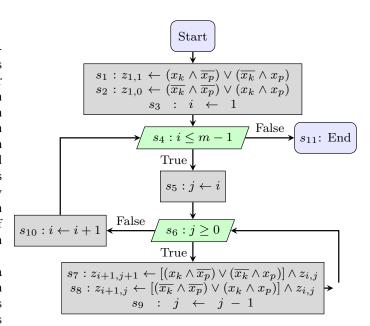A flowchart on the calculation of the number of edges is given in Fig. 2.

Fig. 2. Flowchart for calculating the number of edges in a cut.

B. Quantum circuits for computing the number of edges

Let the initial state of our quantum system be given as $|\phi_0\rangle = \bigotimes_{d=n}^{1} |x_d^0\rangle$, indicating that all qubits are initialized to 0. We apply a Hadamard gate to each of the n qubits, thereby obtaining the state:

$$|\phi_1\rangle = H^{\otimes n} |\phi_0\rangle = \frac{\bigotimes_{d=n}^{1} (|x_d^0\rangle + |x_d^1\rangle)}{\sqrt{2^n}} = \sum_{x=0}^{2^n-1} \frac{|x\rangle}{\sqrt{2^n}} \tag{5}$$

The state $|x_n^0 \cdots x_1^0\rangle$ in $|\phi_1\rangle$ encodes the partition of the n vertices into subsets $V_1 = \emptyset$ and $V_2 = \{v_n, \cdots, v_1\}$. The state $|x_n^0 \cdots x_2^0 x_1^1\rangle$ encodes the partition into $V_1 = \{v_1\}$ and $V_2 = \{v_n, \cdots, v_2\}$, etc.

C. Calculating cut membership for edges

In order to determine to which cut an edge belongs, we introduce auxiliary qubits $|r_{j,s}\rangle$, $1 \leq j \leq m(m+1)/2$, $1 \leq s \leq 4$ that will store intermediate evaluation results for $x_k \wedge \overline{x_p}$, $\overline{x_k} \wedge x_p$, $\overline{x_k} \wedge \overline{x_p}$, and $x_k \wedge x_p$. Further auxiliary qubits $|s_{j,1}\rangle$, $|s_{j,2}\rangle$, $1 \leq j \leq m(m+1)/2$, initialized to 1, are used to evaluate $|r_{j,1}\rangle \vee |r_{j,2}\rangle$ and $|r_{j,3}\rangle \vee |r_{j,4}\rangle$, respectively.

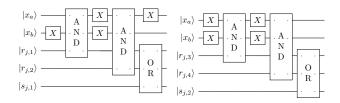The two circuits in Fig. 3 implement, respectively, (1) and (2).

Fig. 3. Left: quantum circuit EIIAC calculates formula (1). Right: quantum circuit EINIAC calculates formula (2).

D. Calculating the number of edges in each cut

The results of executing the operations s_1 and s_2 in Fig. 2 are stored in auxiliary variables $|z_{1,1}\rangle$ and $|z_{1,0}\rangle$, respectively. This fact is shown in Fig. 4 (left).

Similarly, the results of executing the circuits in operations s_7 and s_8 are stored in auxiliary qubits $|z_{i+1,j}\rangle$ and $|z_{i+1,j+1}\rangle$, where $1 \leq i \leq m$ and $0 \leq j \leq i$. This fact is shown in Fig. 4 (right).
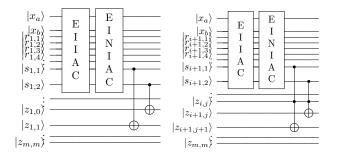

Fig. 4. Left: quantum circuit CFE calculates s_1 and s_2 in Fig. 2. Right: quantum circuit CSE calculates s_7 and s_8 in Fig. 2.

E. Putting the algorithm together

The pieces of our quantum algorithm described above need to be put together and combined with Grover's algorithm for amplification of solutions. This is shown by the pseudo-code in Algorithm 1. We note that R in line (3) is the number of max-cuts. R can be determined by the quantum counting algorithm [1].

The outer loop in line (1) is to be understood in such a way that if the highest value of t doesn't produce a solution, then a value decreased by 1 is tested, and so on, until a solution is found.

In Algorithm 1 the initial state is

$$|\psi_0\rangle = |1\rangle \bigotimes_{i=m}^{1} \bigotimes_{j=i}^{0} |z_{i,j}^0\rangle \bigotimes_{i=\frac{m(m+1)}{2}}^{1} \bigotimes_{b=2}^{1} |s_{i,b}^1\rangle$$

$$\bigotimes_{i=\frac{m(m+1)}{2}}^{1} \bigotimes_{k=4}^{1} |r_{i,k}^0\rangle \bigotimes_{d=n}^{1} |x_d^0\rangle$$

The first register ($|1\rangle$) is a standard auxilliary register used in Grover's routine for both the oracle and the diffusion operator. We will refer to it as *aux*.

After the application of Hadamard gates, the state becomes

$$|\psi_1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \frac{1}{\sqrt{2^n}} \bigotimes_{i=m}^{1} \bigotimes_{j=i}^{0} |z_{i,j}^0\rangle \bigotimes_{i=\frac{m(m+1)}{\sqrt{2}}}^{1} \bigotimes_{b=2}^{1} |s_{i,b}^1\rangle$$

$$\bigotimes_{i=\frac{m(m+1)}{\sqrt{2}}}^{1} \bigotimes_{k=4}^{1} |r_{i,k}^0\rangle \bigotimes_{d=n}^{1} \left(|x_d^0\rangle + |x_d^1\rangle \right)$$

IV. COMPLEXITY ASSESSMENT

As Grover's algorithm has the complexity of $O(2^{n/2})$, this is also the complexity of the present algorithm.

Algorithm 1: Overview of the quantum algorithm for the maximum cut problem

Data: quantum system of size n qubits in state $|\psi_0\rangle$
Result: a maximum cut

1 **for** $t = m$ *down to* 0 **do**
2   Apply Hadamard gates to *aux* and $|x\rangle$ to obtain $|\psi_1\rangle$;
3   **for** $k = 1$ *to* $\frac{\pi}{4} \frac{\sqrt{2^n}}{R}$ **do**
4   Apply CFE to the first edge (Fig.3 left);
5   **for** $i = 1$ *to* $m(m+1)/2$ **do**
6   **for** $j = i$ *down to* 0 **do**
7   Apply CSE to all other edges (Fig.3 right);
8   **end**
9   **end**
10   Apply CNOT gate with control qubit $|z_{m,t}\rangle$ and target qubit *aux*;
11   Reverse steps in lines 4-9;
12   Apply diffusion operator to register x;
13   **end**
14   **if** *output shows spikes with probability* $\geq 1/2$ **then**
15   break;
16   **end**
17 **end**

V. EXPERIMENTAL VALIDATION

We have coded [13] and executed our algorithm on IBM Quantum qasm simulator for the example graph given in Fig. 1. The statistical outcome is shown in Fig. 5. The maximum cut $V_1 = \{v_1, v_3\}, V_2 = \{v_2\}$ (or *vice versa*) is measured with a probability greater than 1/2.

Fig. 6 shows the circuit used to produce the outcome in Fig. 5. The gates applied before the first barrier belong to the initialization step. The circuit is run once only in accordance with the formula $\frac{\pi}{4}\sqrt{\frac{2^n}{R}}$, where $R = 2$ since the maximum cut sets are counted twice, as indicated in Fig. 5. The single run consists of a block of one CFE and two CSE circuits, followed by a CNOT gate that flips the phase of the oracle qubit *aux* for the case where $aux = 1$. After the CNOT gate, the CFE-CSE block needs to be uncomputed to free qubits for eventual further runs in the case such runs are specified by the formula $\frac{\pi}{4}\sqrt{\frac{2^n}{R}}$. And lastly, we have a diffusion block that amplifies the solution in each iteration/run of the Grover routine.

VI. CONCLUSIONS

In the present paper, we have devised a quantum algorithm for the max-cut problem that offers a quadratic speedup over its classical, exact counterparts. We have further successfully executed an instance of the proposed quantum algorithm using IBM's Qiskit SDK [12].
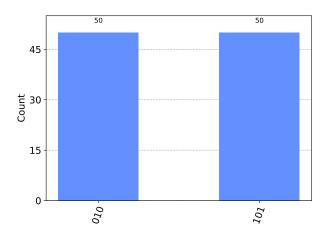
Fig. 5. The maximum cut found for the example graph in Fig.1. Note that both 010 and 101 indicate equivalent maximum cuts. Hence, the maximum cut for Fig.1 is obtained with probability 1.
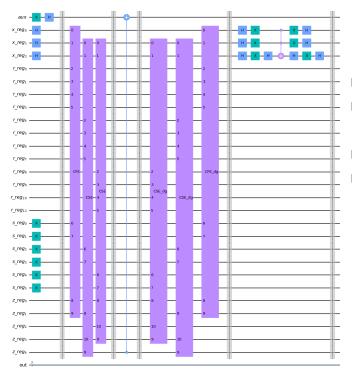

Fig. 6. The circuit for the example graph in Fig.1.

CODE AVAILABILITY

The Python/Qiskit code for the proposed algorithm can be obtained from R. Wong's GitHub repository https://github.com/renatawong/quantum-maxcut [13].

REFERENCES

[1] G. Brassard, P. Hoyer, A. Tapp, "Quantum counting", in: Larsen, K.G., Skyum, S., Winskel, G. (eds) Automata, Languages and Programming. ICALP 1998. Lecture Notes in Computer Science, vol 1443. Springer, Berlin, Heidelberg, DOI: 10.1007/BFb0055105.

[2] P.W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," Proceedings 35th Annual Symposium on Foundations of Computer Science. IEEE Comput. Soc. Press: 124–134, 1994, DOI:10.1109/sfcs.1994.365700.

[3] L.K. Grover, "A fast quantum mechanical algorithm for database search", Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing. STOC '96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery: 212–219, 1996, DOI: 10.1145/237814.237866.

[4] W.-L. Chang et al., "Quantum Speedup for Inferring the Value of Each Bit of a Solution State in Unsorted Databases Using a Bio-Molecular Algorithm on IBM Quantum's Computers," IEEE Transactions on NanoBioscience 21(2):286-293, 2021, DOI: 10.1109/TNB.2021.3130811.

[5] W.-L. Chang et al., "Quantum Speedup and Mathematical Solutions from Implementing Bio-molecular Solutions for the Independent Set Problem on IBM's Quantum Computers," IEEE Transactions on NanoBioscience 20(3):354-376, 2021, DOI: 10.1109/TNB.2021.3075733.

[6] M.X. Goemans and D.P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming", Journal of the ACM, 42 (6): 1115–1145, 1995, DOI: 10.1145/227683.227684.

[7] R. Wong, W.-L. Chang, W.-Y. Chung, and A.V. Vasilakos, "Biomolecular and quantum algorithms for the dominating set problem in arbitrary networks," Scientific Reports 13:4205, 2023, DOI: 10.1038/s41598-023-30600-4.

[8] L.R. Ford and D.R. Fulkerson, "Maximal flow through a network", Canadian Journal of Mathematics 8: 399–404, 1956, DOI: 10.4153/CJM-1956-045-5.

[9] R. Wong, and W.-L. Chang, "Quantum speedup for protein structure prediction," IEEE Transactions on NanoBioscience 20(3):323-330, 2021, DOI: 10.1109/TNB.2021.3065051.

[10] F. Hadlock, "Finding a Maximum Cut of a Planar Graph in Polynomial Time", SIAM J. Comput., 4 (3): 221–225, 1975, DOI: 10.1137/0204019.

[11] R. Wong, and W.-L. Chang, "Fast quantum algorithm for protein structure prediction in hydrophobic-hydrophilic model," Journal of Parallel and Distributed Computing 164:178-190, 2022, DOI: 10.1016/j.jpdc.2022.03.011.

[12] Qiskit contributors, "Qiskit: An Open-source Framework for Quantum Computing," 2023, DOI: 10.5281/zenodo.2573505.

[13] R. Wong, "Quantum maximum cut algorithm", Zenodo, 2023, DOI: 10.5281/zenodo.7790804.