mathematics

IMPACT FACTOR 2.2

CITESCORE 4.6

Article

Dicke State Quantum Search for Solving the Vertex Cover Problem

Jehn-Ruey Jiang

MDPI

https://doi.org/10.3390/math13183005

mathematics

MDPI

Article

Dicke State Quantum Search for Solving the Vertex Cover Problem

Jehn-Ruey Jiang

Department of Computer Science and Information Engineering, National Central University, Taoyuan 320317, Taiwan; jrjiang@csie.ncu.edu.tw

Abstract

This paper proposes a quantum algorithm, named Dicke state quantum search (DSQS), to set qubits in the Dicke state $|D_k^n\rangle$ of D states in superposition to locate the target inputs or solutions of specific patterns among 2^n unstructured input instances, where n is the number of input qubits and $D = \binom{n}{k} = O(n^k)$ for $\min(k, n-k) \ll n/2$. Compared to Grover's algorithm, a famous quantum search algorithm that calls an oracle and a diffuser $O(\sqrt{2^n})$ times, DSQS requires no diffuser and calls an oracle only once. Furthermore, DSQS does not need to know the number of solutions in advance. We prove the correctness of DSQS with unitary transformations, and show that each solution can be found by DSQS with an error probability less than 1/3 through $O(n^k)$ repetitions, as long as $\min(k, n-k) \ll n/2$. Additionally, this paper proposes a classical algorithm, named DSQS-VCP, to generate quantum circuits based on DSQS for solving the k-vertex cover problem (k-VCP), a well-known NP-complete (NPC) problem. Complexity analysis demonstrates that DSQS-VCP operates in polynomial time and that the quantum circuit generated by DSQS-VCP has a polynomial qubit count, gate count, and circuit depth as long as $\min(k, n-k) \ll n/2$. We thus conclude that the k-VCP can be solved by the DSQS-VCP quantum circuit in polynomial time with an error probability less than 1/3 under the condition of $\min(k, n-k) \ll n/2$. Since the k-VCP is NP-complete, NP and NPC problems can be polynomially reduced to the k-VCP. If the reduced k-VCP instance satisfies $\min(k, n-k) \ll n/2$, then both the instance and the original NP/NPC problem instance to which it corresponds can be solved by the DSQS-VCP quantum circuit in polynomial time with an error probability less than 1/3. All statements of polynomial algorithm execution time in this paper apply only to VCP instances and similar instances of other problems, where $\min(k, n-k) \ll n/2$. Thus, they imply neither NP $\subseteq$ BQP nor P = NP. In this restricted regime of $\min(k, n-k) \ll n/2$, the Dicke state subspace has a polynomial size of $D = \binom{n}{k} = O(n^k)$, and our DSQS algorithm samples from it without asymptotic superiority over exhaustive enumeration. Nevertheless, DSQS may be combined with other quantum algorithms to better exploit the strengths of quantum computation in practice. Experimental results using IBM Qiskit packages show that the DSQS-VCP quantum circuit can solve the k-VCP successfully.

Keywords: Dicke state; Grover's algorithm; NP-complete problem; oracle; quantum search; vertex cover problem

MSC: 68Q12

check for updates

Academic Editor: João Nuno Prata

Received: 9 August 2025
Revised: 12 September 2025
Accepted: 14 September 2025
Published: 17 September 2025

Citation: Jiang, J.-R. Dicke State Quantum Search for Solving the Vertex Cover Problem. *Mathematics* **2025**, *13*, 3005. https://doi.org/10.3390/math13183005

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Quantum computers manipulate quantum bits, or qubits, which harness quantum phenomena such as superposition, entanglement, and tunneling [1,2] for computations. Unlike classical bits, which are confined to a state of either 0 or 1, qubits can exist in a superposition, representing both 0 and 1 simultaneously until measured. This characteristic allows for n qubits to represent and process all 2^n possible states at once, whereas n classical bits are restricted to represent and process one of those 2^n states at a time. Thus, the computational power of quantum computers increases exponentially with the number of qubits. Quantum computers can therefore surpass classical computers, finishing computations that classical computers can never complete, achieving what is called quantum supremacy [3]. This has driven the development of numerous quantum algorithms, such as the Deutsch–Josza algorithm [4], Shor's algorithm [5], and Grover's algorithm [6].

Grover's algorithm [6] is a quantum search algorithm proposed by Grover in 1996. It is designed to identify a specific *target input* or *solution* from a set of N unstructured or unsorted inputs. Grover's algorithm uses the concept of amplitude amplification to search for the target input, employing an *oracle* to invert the phase of the target input state and a *diffuser* to amplify its amplitude to be much larger than others. It is shown that Grover's algorithm can identify the target input with high probability by repeating the oracle and the diffuser $\mathrm{O}(\sqrt{N})$ times. In 1998, Boyer et al. [7] demonstrated that if the number M $(1 \leq M \ll N)$ of target inputs or solutions is known in advance, then repeating the oracle and the diffuser $\left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor$ times enables Grover's algorithm to find all M solutions with high probability. In 1997, Bennett et al. showed that any oracle-based quantum algorithm needs to call an oracle at least $\mathrm{O}(\sqrt{N})$ times to identify a target input from N unstructured inputs with high probability [8]. This establishes that Grover's algorithm is theoretically optimal oracle-based quantum search algorithm.

This paper proposes a quantum algorithm, named Dicke state quantum search (DSQS), to set qubits in the Dicke state $|D_k^n\rangle$ of D states in superposition to identify all M target inputs or solutions of special patterns from 2^n unstructured input instances, where n is the number of input qubits, and $D = \binom{n}{k} = \mathrm{O}(n^k)$, provided that $\min(k, n-k) \ll n/2$. Compared to Grover's algorithm, which calls an oracle and a diffuser $\mathrm{O}(\sqrt{2^n})$ times, DSQS requires no diffuser and calls an oracle only once. Moreover, DSQS does not need to know the number M of solutions in advance. The correctness of DSQS is proven with unitary transformations. We also show that DSQS can locate every single solution with a probability of error less than $1/3$ through $\mathrm{O}(n^k)$ repetitions for $\min(k, n-k) \ll n/2$. This does not conflict with the assertion that Grover's algorithm is the optimal oracle-based quantum search algorithm. The reason for this is that DSQS sets qubits into the Dicke state $|D_k^n\rangle$, reducing the search space for the oracle to D states, where exactly k out of n qubits are in the state $|1\rangle$, and the remaining qubits are in the state $|0\rangle$. Grover's algorithm calls an oracle $\mathrm{O}(\sqrt{2^n})$ times to find solutions with high probability in a single repetition. (In practice, multiple repetitions are required to identify the solutions with a probability that approximates the theoretical value.) DSQS calls an oracle once to locate solutions with a probability of $1/D$ through one repetition. Afterwards, DSQS locates solutions with high probability through $\mathrm{O}(n^k)$ repetitions, the order of which is polynomial as long as $\min(k, n-k) \ll n/2$.

This paper also proposes a classical algorithm, named Dicke state quantum search–vertex cover problem (DSQS-VCP), to construct a quantum circuit based on DSQS to solve the k-vertex cover problem (k-VCP) [9]. The k-VCP is also referred to as the vertex cover problem (VCP) for simplicity. It is a known NP-complete (NPC) problem, which is a decision problem that determines whether at least one solution satisfies specified conditions [10]. For an NPC problem, it is unlikely that a classical algorithm exists to solve

the problem in polynomial time for all cases. DSQS-VCP addresses the NPC challenge and tries to solve the search version of the k-VCP by generating a quantum circuit, named the DSQS-VCP quantum circuit, to identify all solutions satisfying the given conditions of the k-VCP in polynomial time with bounded error for specific cases.

We analyze the time complexity of DSQS-VCP and further decompose the DSQS-VCP quantum circuit into components consisting solely of H, T, and CX gates, which are basis gates in a universal gate set. We also analyze the complexity of this decomposed circuit in terms of the qubit count, gate count, and circuit depth. The overall complexity analysis reveals that DSQS-VCP can generate a quantum circuit based on DSQS using $|D_k^n\rangle$ to solve the k-VCP in polynomial time with a probability of error less than $1/3$ for $\min(k, n-k) \ll n/2$. Since the k-VCP is NP-complete, NP and NPC problems can be polynomially reduced to the k-VCP. If the reduced k-VCP instance satisfies $\min(k, n-k) \ll n/2$, then both the instance and the original NP/NPC problem instance to which it corresponds can be solved by the DSQS-VCP quantum circuit in polynomial time with an error probability less than $1/3$. We further conduct two experiments using IBM Qiskit [11] packages to implement and run DSQS-VCP quantum circuits to identify all solutions to the example k-VCP instances successfully. The instances are for small n and k, such as $(n = 5, k = 1, 2, 3)$ and $(n = 7, k = 2, 3, 4)$, all of which satisfy the condition $\min(k, n-k) \ll n/2$.

Overall, this paper presents the DSQS algorithm and constructs DSQS-VCP quantum circuits to solve with polynomial complexity VCP instances and similar instances of other problems when $\min(k, n-k) \ll n/2$ holds. However, this implies neither NP $\subseteq$ BQP nor P=NP. In the restricted regime of $\min(k, n-k) \ll n/2$, the Dicke state subspace has a polynomial size of $D = \binom{n}{k} = \mathrm{O}(n^k)$, and DSQS samples from it without asymptotic superiority over exhaustive enumeration. Nevertheless, DSQS may be paired with other quantum algorithms to better exploit the strengths of quantum computation, which is not within the scope of this paper.

The remainder of this paper is organized as follows. Some background knowledge is introduced in Section 2. The concept of DSQS is elaborated, and its correctness is shown in Section 3. The DSQS-VCP algorithm is introduced, and its generated quantum circuit is analyzed in Section 4. The results of experiments based on IBM Qiskit packages are shown in Section 5. Finally, Section 6 concludes this paper.

2. Background

2.1. Grover's Algorithm

In 1996, Grover introduced a quantum search algorithm known as Grover's algorithm [6]. This algorithm is designed to locate a specific *target input* or *solution* within a set of N unstructured or unsorted input instances. By iteratively applying an oracle and a diffuser $O(\sqrt{N})$ times, Grover's algorithm achieves a high probability of successfully identifying the solution. In contrast, classical algorithms typically require $\mathrm{O}(N)$ oracle calls on average, and, in the worst case, to locate the target input within unstructured input instances. This highlights the quadratic speedup provided by Grover's algorithm in terms of oracle call efficiency.

The quantum circuit of Grover's algorithm is depicted in Figure 1 [12], which illustrates two primary components: the oracle and the diffuser. These two components are collectively referred to as the *Grover iterator*, which is executed $O(\sqrt{N})$ times throughout the algorithm.

The oracle, denoted U_f in Figure 1, plays a crucial role in identifying the target input by flipping its phase. Its definition is provided below.

$$U_f |x\rangle = \begin{cases} -|x\rangle & \text{if } |x\rangle = |x^*\rangle \\ |x\rangle & \text{otherwise} \end{cases} \quad (1)$$

In Equation (1), the target input is denoted $|x^*\rangle$. The oracle operates by flipping the phase of the state $|x\rangle$ if and only if $|x\rangle$ matches $|x^*\rangle$; otherwise, it leaves the state unchanged.

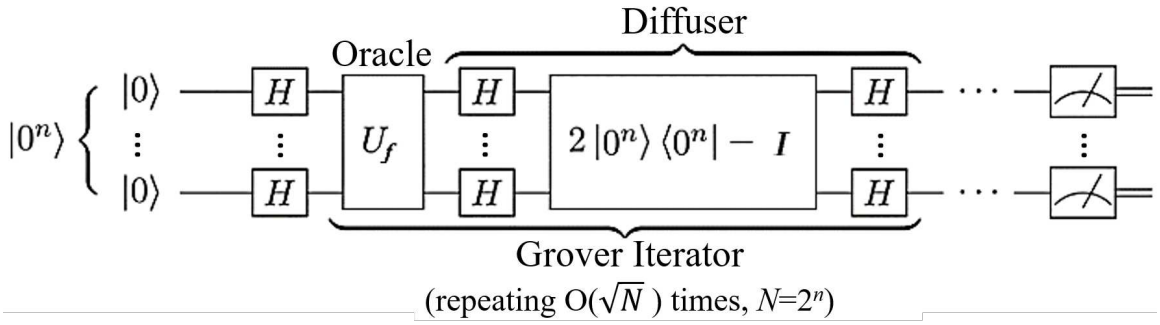


Figure 1. The quantum circuit of Grover's algorithm.

The diffuser, on the other hand, is responsible for adjusting the probability amplitudes of quantum states by inverting them about their average value; it is an inversion around amplitude mean operation. Grover's algorithm leverages the oracle and the diffuser to iteratively refine the quantum state of the system. Initially, the qubits are prepared in a uniform superposition state. When the oracle is applied, it flips the phase of the target input, effectively assigning it a negative probability amplitude. The diffuser then amplifies the amplitude of the target input and meanwhile reduces the amplitudes of all other inputs. Repeating the Grover iterator $O(\sqrt{N})$ times significantly increases the amplitude of the target input while diminishing those of non-target inputs, allowing for the algorithm to identify the solution with high probability. Grover's algorithm, which repeats the oracle and the diffuser $O(\sqrt{N})$ times, has been shown in [8] to be the optimal oracle-based quantum search algorithm.

Grover's algorithm can also handle scenarios involving multiple target inputs. In 1998, Boyer et al. demonstrated that when M target solutions exist and M is known beforehand, performing the oracle and the diffuser $\left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor$ times ensures that all solutions can be identified with high probability. To estimate the number of solutions M in a given problem, Brassard et al. introduced the quantum counting algorithm in [13]. This algorithm employs quantum phase estimation (QPE) with t counting qubits and a total of $2^t - 1$ Grover iterator repetitions [14]. However, this approach may result in large gate counts and deep quantum circuits, posing challenges for practical implementation.

Numerous research papers [12,15–24] are proposed in the literature to build quantum circuits based on Grover's algorithm to solve various NP-hard and NP-complete problems. The solved problems include the k-coloring problem, the maximum clique problem, the list coloring problem, the pure Nash equilibria finding problem in graphical games, the Hamiltonian cycle problem, the dominating set problem, the exact cover problem, and the vertex cover problem. Readers are referred to [12] for descriptions of all the above-mentioned research results.

2.2. The Vertex Cover Problem

The vertex cover problem (VCP) is a well-known decision problem. It is also known as the k-vertex cover problem (k-VCP). As shown in [9], the VCP is NP-complete (NPC),

meaning it is unlikely that a classical algorithm exists to solve it in polynomial time for any cases.

The problem definition is as follows. Given an undirected graph $G(V, E)$ with n vertices and m edges, the k-VCP or the VCP associated with k $(1 \leq k \leq n)$ asks whether there exists a vertex subset $V' \subseteq V$ with $|V'| \leq k$ such that every edge $(u, v) \in E$ is covered by V', i.e., $u \in V'$ and/or $v \in V'$. In this context, a vertex subset V' is called a k-vertex cover (or a vertex cover with size k) if its size is k and covers all edges in the graph. Simply put, the k-VCP determines whether a vertex cover of size k or smaller exists for a given graph. If so, it answers "yes"; otherwise, it outputs "no".

For example, Figure 2 illustrates two vertex covers with size 2 for an undirected graph with 5 vertices and 4 edges. In the case of $k = 1$, the output of the k-VCP for this graph is "no". However, in the case of $k \geq 2$, the output of the k-VCP for this graph is "yes". For another example, Figure 3 illustrates three vertex covers with size 3 for an undirected graph with 7 vertices and 7 edges. In the cases of $k = 1$ and $k = 2$, the output of the k-VCP for this graph is "no". However, in the case of $k \geq 3$, the output of the k-VCP for this graph is "yes".

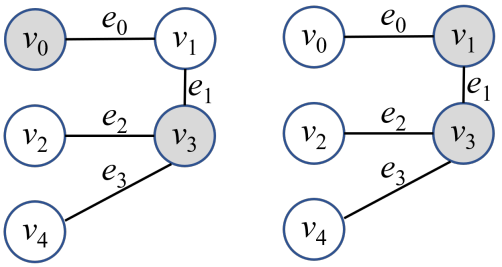


Figure 2. Two vertex covers for an undirected graph with five vertices and four edges: (**left**) the vertex subset $\{v_0, v_3\}$ is a 2-vertex cover; (**right**) the vertex subset $\{v_1, v_3\}$ is also a 2-vertex cover.

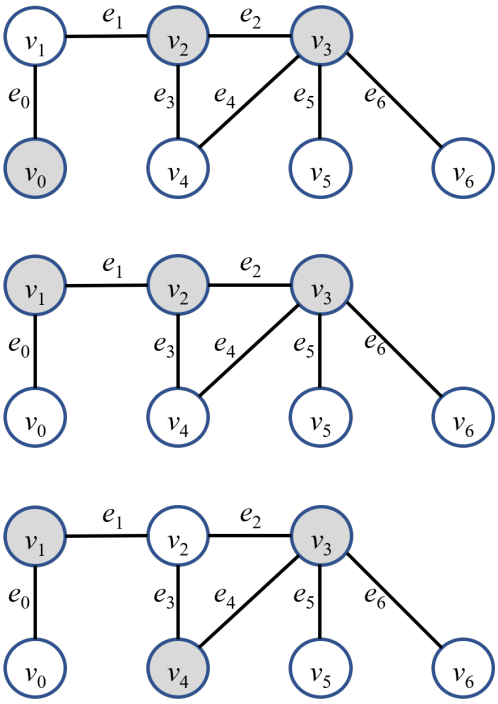


Figure 3. Three vertex covers for an undirected graph with seven vertices and seven edges: (**top**) the vertex subset $\{v_0, v_2, v_3\}$ is a 3-vertex cover; (**middle**) the vertex subset $\{v_1, v_2, v_3\}$ is another 3-vertex cover; (**bottom**) the vertex subset $\{v_1, v_3, v_4\}$ is yet another 3-vertex cover.

The original VCP or k-VCP is a decision problem that determines whether a given graph has a vertex cover of size k or smaller. However, this paper also considers the search-version k-VCP, which outputs all vertex covers of size k (i.e., all k-vertex covers) for a given graph. Throughout this work, the term "k-VCP" may refer to either the decision version k-VCP or the search version k-VCP, depending on the context.

2.3. The Dicke State

The concept of Dicke states was introduced by Dicke in 1954 [25]. A Dicke state, denoted $|D_k^n\rangle$, is a special type of entangled quantum state of n qubits, where exactly k qubits are in $|1\rangle$ and the rest are in $|0\rangle$. In other words, it is the equal (or symmetric) superposition of all n-qubit states $|x\rangle$ with Hamming weight $\mathrm{wt}(x) = k$, where $\mathrm{wt}(x)$ stands for the number of bits set to 1 in binary string x [26]. The Dicke state is defined as follows:

$$|D_k^n\rangle = \sqrt{\frac{1}{\binom{n}{k}}} \sum_{x=\{0,1\}^n, \mathrm{wt}(x)=k} |x\rangle . \tag{2}$$

The number D of possible in-superposition states in $|D_k^n\rangle$ is $D = \binom{n}{k} = \frac{n!}{k!(n-k)!}$. For example, $|D_3^4\rangle = \sqrt{\frac{1}{4}}(|0111\rangle + |1011\rangle + |1101\rangle + |1110\rangle)$ is an equal superposition of $D = \binom{4}{3} = 4$ states.

For $1 \leq k \ll n/2$, $k!$ can be considered as a constant. We therefore obtain the following Equation (3):

$$D = \binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k!} = cn^k = \mathrm{O}(n^k) \tag{3}$$

In Equation (3), $c = \frac{n(n-1)\cdots(n-k+1)}{k!n^k} < 1$ for $1 \leq k \ll n/2$. By Equation (3), $D = \mathrm{O}(n^k)$ grows polynomially with n for $1 \leq k \ll n/2$. However, for k near $n/2$, $D = \binom{n}{k}$ reaches its maximum value. By using Stirling's approximation for factorials (i.e., $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$) [27], we have the following Equation (4):

$$\binom{n}{n/2} = \frac{n!}{(n/2)!(n/2)!} = \mathrm{O}\left(\frac{2^n}{\sqrt{n}}\right) \tag{4}$$

In Equation (4), $D = \mathrm{O}\left(\frac{2^n}{\sqrt{n}}\right)$ grows exponentially in n, with a polynomial correction factor of $\frac{1}{\sqrt{n}}$ for $k = n/2$. In summary, D grows polynomial with n, as long as $\min(k, n-k) \ll n/2$, i.e., either $k = \mathrm{O}(n/\ln n)$ or $(n-k) = \mathrm{O}(n/\ln n)$. In contrast, D grows exponentially with n when $k = n/2$. For intermediate k, D transits between polynomial and exponential growth, depending on k/n. It is notable that $\min(k, n-k) \ll n/2$ is required for the proposed DSQS algorithm to solve problems in polynomial time with bounded error. Therefore, we assume $\min(k, n-k) \ll n/2$ in this paper when k is not specified.

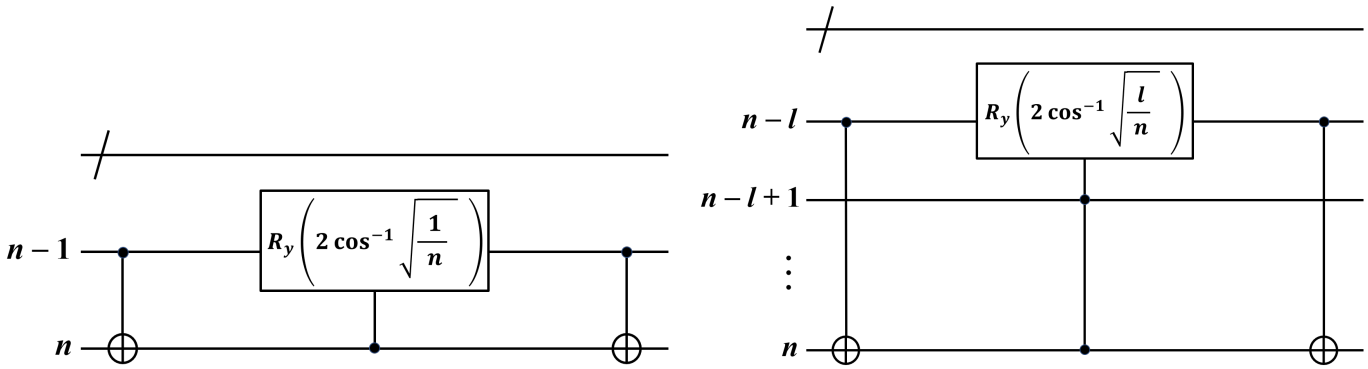
Generating Dicke states deterministically is a challenging task. In [26], Bartschi and Eidenbenz proposed a quantum circuit capable of efficiently preparing the Dicke state $|D_k^n\rangle$. This quantum circuit requires $\mathrm{O}(kn)$ quantum gates, has a depth of $\mathrm{O}(n)$, does not rely on ancillary qubits, and is compatible with the linear nearest neighbor (LNN) architecture. The overall circuit is based on a recursive decomposition based on an initial state $|0\rangle^{\otimes(n-k)}|1\rangle^{\otimes k}$ to produce the final state $|D_k^n\rangle$. The recursive decomposition is described in Equation (5) as follows.

$$|D_k^n\rangle = \sqrt{\frac{n-k}{n}}|D_k^{n-1}\rangle|0\rangle + \sqrt{\frac{k}{n}}|D_{k-1}^{n-1}\rangle|1\rangle. \tag{5}$$

Equation (5) indicates that the preparation of $|D_k^n\rangle$ can be achieved by combining the state $|D_k^{n-1}\rangle$ along with $|0\rangle$ and the state $|D_{k-1}^{n-1}\rangle$ along with $|1\rangle$, while applying appropriate Y-rotation R_y transforms to adjust the probability amplitudes.

By the recursive decomposition, Bartschi and Eidenbenz showed that the Dicke state $|D_k^n\rangle$ preparation quantum circuit can be built based on two basic constructs, as depicted in Figure 4. Note that the boundary condition of the Dicke state decomposition is $D_1^1 = |1\rangle$ and $D_0^1 = |0\rangle$. Also note that a Y-rotation $R_y\left(2\cos^{-1}\sqrt{\frac{l}{n}}\right)$ gate transforms the state $|0\rangle$ to the state $\sqrt{\frac{l}{n}}|0\rangle + \sqrt{\frac{n-l}{n}}|1\rangle$ to make qubits have proper probability amplitudes obeying the Dicke state superposition.

Figure 5 shows the quantum circuit using the two basic constructs recursively for Dicke state $|D_3^4\rangle$ preparation. Figure 6 shows the quantum circuit implemented with IBM Qiskit packages to generate the Dicke state $|D_3^4\rangle$. Figure 7 shows the quantum circuit to measure qubits in the Dicke state $|D_3^4\rangle$. Figure 8 shows the measurement results of the quantum circuit of the Dicke state $|D_3^4\rangle$. From the measurement results, we can observe that qubits are in equal (or symmetric) superposition of $D = \binom{4}{3} = 4$ states $|0111\rangle$, $|1011\rangle$, $|1101\rangle$, and $|1110\rangle$ with the probability amplitude of $\sqrt{\frac{1}{D}} = \sqrt{\frac{1}{4}}$ and the probability density of $\frac{1}{D} = \frac{1}{4}$.



(**a**) The first basic construct.

(**b**) The second basic construct.

Figure 4. The two basic constructs used to build the Dicke state $|D_k^n\rangle$ preparation quantum circuit.

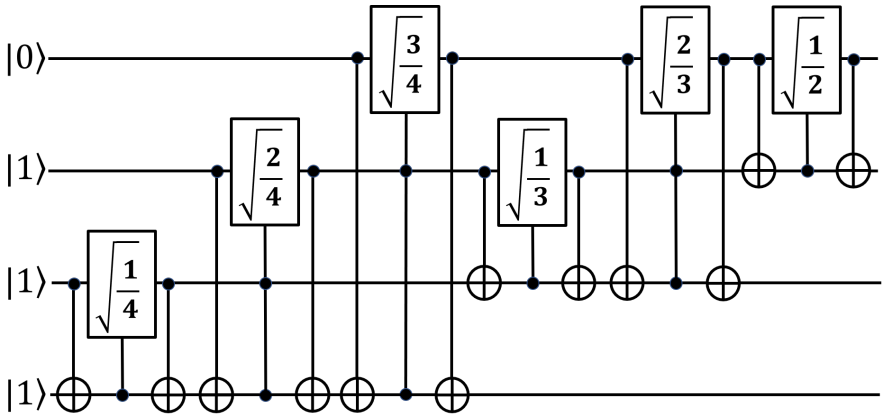


Figure 5. An example of the quantum circuit for the Dicke state $|D_3^4\rangle$ preparation, where a $\sqrt{\frac{l}{n}}$ gate is a shorthand for a Y-rotation $R_y\left(2\cos^{-1}\sqrt{\frac{l}{n}}\right)$ gate.

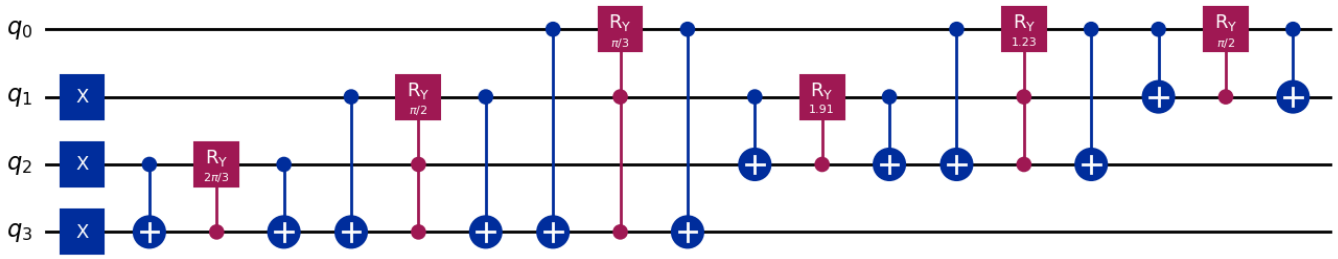


Figure 6. The quantum circuit implemented with IBM Qiskit packages to generate the Dicke state $|D_3^4\rangle$.

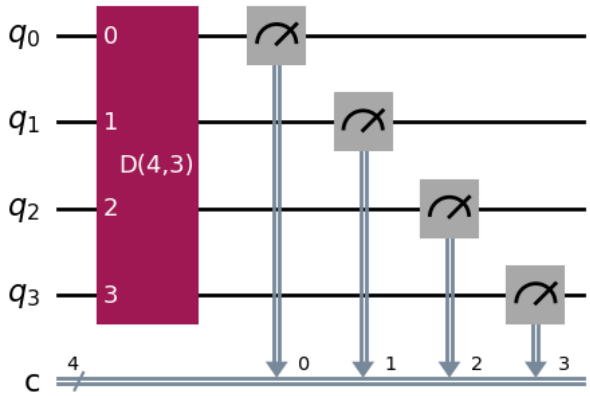


Figure 7. The quantum circuit implemented with IBM Qiskit packages to measure qubits in the Dicke state $|D_3^4\rangle$.

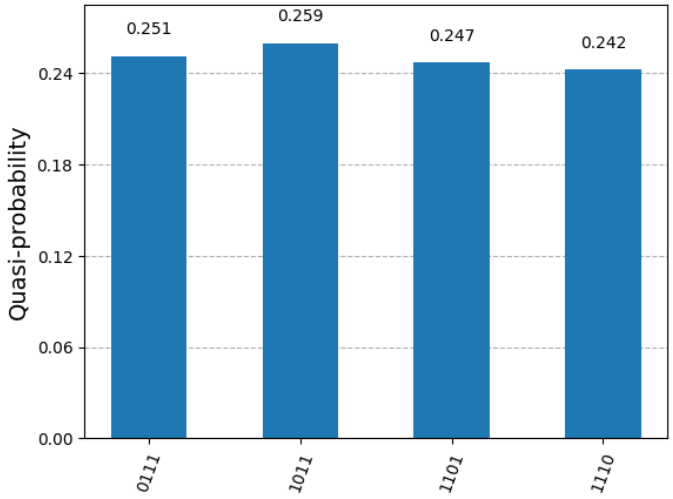


Figure 8. The measurement results of the quantum circuit of the Dicke state $|D_3^4\rangle$.

3. DSQS

3.1. The DSQS Quantum Circuit

Figure 9 shows the quantum circuit of the proposed DSQS algorithm. As shown in Figure 9, the quantum circuit uses $2n+1$ qubits, a Dicke state $|D_k^n\rangle$ preparation gate and an oracle U_f. All $2n+1$ qubits are in $|0\rangle$ initially. They include n working qubits (or input qubits) $x_0, \ldots, x_{n-1}$ grouped as register x, one response qubit (or decision qubit) y as register y, and n mirror qubits $x'_0, \ldots, x'_{n-1}$ grouped as register x' to reflect the states of qubits in register x.

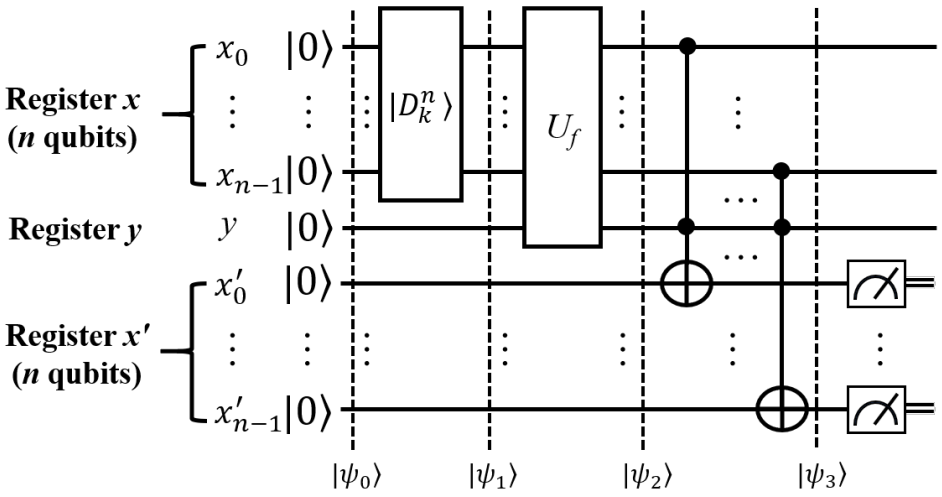


Figure 9. The quantum circuit of the proposed Dicke state quantum search (DSQS) algorithm.

Initially, n working qubits $x_0, \ldots, x_{n-1}$ in $|0\rangle$ are used to span the search space. They are transformed into the Dicke state $|D_k^n\rangle$ with a Dicke state gate. The resulting state is an equal superposition of states where exactly k qubits are in $|1\rangle$, and the remaining $n-k$ qubits are in $|0\rangle$. The total number of such possible in-superposition states is $D = \binom{n}{k}$. Subsequently, the oracle is applied to the working qubits $x_0, \ldots, x_{n-1}$ in register x and the qubit y in register y, performing the unitary transformation U_f of the oracle as defined in the next subsection.

3.2. The DSQS Oracle

The DSQS oracle U_f is a unitary transform that operates on the n working qubits (or input qubits) $x_0, \ldots, x_{n-1}$ in register x and the decision qubit y in register y. Let T be the set of all target inputs or solutions to a given problem, and let $f(x)$ be the function corresponding to the oracle U_f for the given problem to satisfy the following definition:

$$f(x) = \begin{cases} 1, & \text{if } x \in T \text{ (i.e., } x \text{ is a target input)} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

The oracle U_f is then defined as

$$U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle = \begin{cases} |x\rangle|y \oplus 1\rangle, & \text{if } x \in T \\ |x\rangle|y \oplus 0\rangle, & \text{otherwise} \end{cases} \tag{7}$$

When $y = 0$, the equation simplifies to

$$U_f\,|x\rangle\,|y\rangle = U_f\,|x\rangle\,|0\rangle = |x\rangle\,|f(x)\rangle = \begin{cases} |x\rangle|1\rangle, \text{ if } x \in T \\ |x\rangle|0\rangle, \text{ otherwise} \end{cases} \tag{8}$$

Specifically, the initial state of qubit y is $|0\rangle$. If the quantum state of the working qubits $x_0, \ldots, x_{n-1}$ in register x corresponds to a target input or solution in set T, then the oracle U_f sets qubit y to $|y \oplus f(x)\rangle = |1\rangle$. Otherwise, qubit y remains $|y \oplus f(x)\rangle = |0\rangle$.

Let the total number of solutions be M. Therefore, there are M states of register x matching with solution states corresponding to target inputs in T. Before the oracle operation U_f, qubits $x_0, \ldots, x_{n-1}$ are in the Dicke state $|D_k^n\rangle$ in equal superposition of a total of $D = \binom{n}{k}$ possible states, and qubit y is $|0\rangle$. Therefore, the oracle U_f flips the state of

y to $|1\rangle$ for M solution states out of the D states of $|D_k^n\rangle$, while leaving y in state $|0\rangle$ for the remaining $D - M$ non-solution states. We therefore have the following Equation (9):

$$y = \sqrt{\frac{D-M}{D}}\,|0\rangle + \sqrt{\frac{M}{D}}\,|1\rangle \tag{9}$$

In Equation (9), we have that the probability amplitude of qubit y being $|1\rangle$ is $\sqrt{M/D}$. Thus, the probability density P of qubit y being $|1\rangle$ is M/D. We then obtain the following equation:

$$M = PD \tag{10}$$

We can measure register y to estimate the probability density P of y being $|1\rangle$. Let the estimated probability density be P'. According to Equation (10), we can estimate the total number of solutions M by multiplying P' with D, as shown in Equation (11).

$$M = P'D \tag{11}$$

In Equation (11), we can see that if the DSQS quantum circuit consists only of register x with n qubits and register y with a single qubit, it can also function as a quantum circuit for estimating the total number of solutions M.

However, DSQS performs more tasks, as described below. DSQS takes a qubit x_i from register x and the qubit y as the two control qubits of a Toffoli, or controlled–controlled-X (CCX, or C^2X) gate, with one qubit x'_i from register x' as the target qubit, where $0 \leq i \leq n-1$. This setup ensures that the solution states of register x of n working qubits are reflected on register x' of n mirror qubits $x'_0, \ldots, x'_{n-1}$, whereas the non-solution states are all reflected as $|0\rangle^{\otimes n}$ on mirror qubits.

By measuring the results of register x', we can identify the solutions to the given search problem. If the measurement outcome is the all-zero state with a probability of 1, it indicates that the problem has no solution. Conversely, non-all-zero outcomes, each occurring with a probability of $1/D$, correspond to valid solutions to the problem. Meanwhile, the all-zero state occurs with a probability of $(D - M)/D$, where M is the number of solutions and $D = \binom{n}{k}$ is the number of possible states in the Dicke state $|D_k^n\rangle$. The correctness of DSQS will be demonstrated in the next subsection.

3.3. The DSQS Correctness

The correctness for the DSQS quantum circuit using the Dicke state $|D_k^n\rangle$ with $D = \binom{n}{k}$ states in superposition to identify all solutions to a given problem is proven by the unitary transformations below. Refer to Figure 9 for the placement of quantum states $|\psi_0\rangle, \ldots, |\psi_3\rangle$ within the quantum circuit. Note that we use $x \in DS_k^n$ as the shorthand for $x = \{0,1\}^n$, $\mathrm{wt}(x) = k$.

$$|\psi_0\rangle = |0\rangle^{\otimes n}\,|0\rangle\,|0\rangle^{\otimes n} \tag{12}$$

$$\begin{aligned} |\psi_1\rangle &= |D_k^n\rangle\,|0\rangle\,|0\rangle^{\otimes n} \\ &= \left(\sqrt{\frac{1}{D}}\sum_{x\in DS_k^n}|x\rangle\right)|0\rangle\,|0\rangle^{\otimes n} \end{aligned} \tag{13}$$

$$|\psi_2\rangle = U_f\left(\left(\sqrt{\frac{1}{D}}\sum_{x\in DS_k^n}|x\rangle\right)|0\rangle\right)|0\rangle^{\otimes n}$$

$$= \left(\sqrt{\frac{1}{D}} \sum_{x \in DS_k^n} |x\rangle \right) |f(x)\rangle \, |0\rangle^{\otimes n} \tag{14}$$

$$|\psi_3\rangle = \prod_{i=n-1}^{0} \mathrm{CCX}_i \left(\left(\sqrt{\frac{1}{D}} \sum_{x \in DS_k^n} |x\rangle \right) |f(x)\rangle \, |0\rangle^{\otimes n} \right) \tag{15}$$

$$|\psi_3\rangle_{x'} = \sqrt{\frac{1}{D}} \left(\sum_{x \notin T} |0\rangle^{\otimes n} + \sum_{x \in T} |x\rangle \right)$$

$$= \sqrt{\frac{D-M}{D}} \, |0\rangle^{\otimes n} + \sqrt{\frac{1}{D}} \sum_{x \in T} |x\rangle \tag{16}$$

In Equation (15), CCX_i stands for a CCX gate with qubits x_i and y as control qubits and with x_i' as the target qubit. In Equation (15), different registers are entangled. Therefore, measuring only a subset of registers (e.g., $x_0', \ldots, x_{n-1}'$) and discarding the rest is formally equivalent to taking a partial trace over the unmeasured registers. In Equation (16), for register x' in state $|\psi_3\rangle$ (denoted $|\psi_3\rangle_{x'}$), its probability amplitude of $|0\rangle^{\otimes n}$ is $\sqrt{(D-M)/D}$, and its probability amplitude of $|x\rangle$ is $\sqrt{1/D}$, where $x \in T$, $0 \leq x \leq 2^n - 1$, and T is the set of all M target inputs or solutions. Therefore, DSQS makes register x' have a probability density of $1/D$ for each solution state corresponding to a solution, and a probability density of $(D-M)/D$ for the all-zero state $|0\rangle^{\otimes n}$. If the problem has no solution, i.e., $M = 0$, the all-zero state $|0\rangle^{\otimes n}$ has a probability density of $(D-M)/D = D/D = 1$.

3.4. The DSQS Error Probability

Bernstein and Vazirani introduced the complexity class of bounded-error quantum polynomial time (BQP) [28] in 1993. Specifically, BQP is defined as the class of decision problems solvable by a quantum Turing machine (QTM) [29] in polynomial time with an error probability that is less than $1/3$. The QTM [29] was proposed by Deutsch in 1985; it is a quantum version of the classical Turing machine (TM) [30] proposed by Turing in 1936. A classical computer can be considered a TM, whereas a quantum computer is a QTM. Note that the error bound of $1/3$ in BQP originates from the convention adopted by the classical probabilistic complexity class BPP (bounded-error probabilistic polynomial time) [31]. The choice of $1/3$ is not mandatory. In practice, any constant less than $1/2$ can be used, as it guarantees a success probability greater than $1/2$. This is important because when the success probability exceeds $1/2$, it can be boosted arbitrarily close to 1 by repeating the algorithm and taking a majority vote.

The Chernoff bound shows that the probability of majority error decreases exponentially in the number of repetitions.

Some algorithms, such as Shor's algorithm, are believed to place problems, such as integer factorization and discrete logarithms, in BQP because the algorithms run in polynomial time to solve problems with bounded error on a quantum computer. While Grover's algorithm also operates with bounded error, it does not place problems in BQP. This is because Grover's algorithm finds all solutions to a problem among 2^n input instances by calling the oracle and diffuser $\mathrm{O}(\sqrt{2^n})$ times. The number of calls is exponential in the input size n, where n is the number of qubits defining the search space.

With the help of Dicke state $|D_k^n\rangle$, the proposed DSQS algorithm can locate all solutions of specific patterns to a given problem among 2^n input instances by calling only the oracle just once. However, it still does not place problems in BQP. This is because the probability of error not to locate a single solution is $\frac{D-1}{D}$, which is not less than $1/3$, where $D = \binom{n}{k}$. Nonetheless, we show below that DSQS can locate all solutions with a probability of error

less than $1/3$ by performing $O(n^k)$ repetitions, the order of which is polynomial, as long as $\min(k, n-k) \ll n/2$.

As previously mentioned, the proposed DSQS does not utilize a diffuser for amplitude amplification. Instead, it maintains the probability amplitude of a solution state corresponding to a solution to be $\sqrt{\frac{1}{D}}$. This results in a probability density of $\frac{1}{D}$ for a solution state. Therefore, all solution states have the probability amplitude of $\sqrt{\frac{M}{D}}$ with the probability density of $\frac{M}{D}$. In contrast, a non-solution state corresponding to a non-solution has a probability amplitude and a probability density of 0, with the exception that the all-zero state has a probability amplitude of $\sqrt{\frac{D-M}{D}}$ and the probability density of $\frac{D-M}{D}$. Here, we assume the all-zero state is also a non-solution state without loss of generality. In practice, DSQS completely shifts the probability amplitudes of all non-solution states to the all-zero state. Thus, when we run DSQS and perform a single measurement, each solution state has, independently, a $\frac{1}{D}$ probability of being observed, whereas the all-zero state has a $\frac{D-M}{D}$ probability of being observed and the other non-solution states have no probability of being observed. Consequently, in the absence of noise, any measured result other than the all-zero combination can be considered a reflection of a solution.

Due to the Dicke state, the size of the search space for oracle processing is reduced from 2^n to $D = \binom{n}{k} = O(n^k)$ for $\min(k, n-k) \ll n/2$. This reduction allows for DSQS to ensure that the probability of not observing a single solution is smaller than $\frac{1}{3}$ after a polynomial number of repetitions. We present the following lemma and theorem:

Theorem 1. *The probability that a single solution is not observed by DSQS using the Dicke state* $|D_k^n\rangle$ *becomes less than* $\frac{1}{3}$ *after* $O(n^k)$ *repetitions for* $\min(k, n-k) \ll n/2$.

Proof. The probability that a single solution is observed in one repetition of DSQS using Dicke state $|D_k^n\rangle$ is $\frac{1}{D}$, where $D = \binom{n}{k} = O(n^k)$ for $\min(k, n-k) \ll n/2$. The probability of error that a single solution is not observed in one repetition of DSQS is therefore $\frac{D-1}{D}$. Since $D = O(n^k)$ for $\min(k, n-k) \ll n/2$, we assume $D \leq cn^k$ with c being a positive constant. Without loss of generality, we assume $D = cn^k$ for $c > 0$.

Let ϵ be the probability of error that a single solution is not observed in t consecutive repetitions of DSQS. We obtain

$$\epsilon = \left(\frac{D-1}{D}\right)^t$$

We aim for $\epsilon < \frac{1}{3}$, leading to

$$\left(\frac{D-1}{D}\right)^t < \frac{1}{3}$$

Taking the natural logarithm on both sides, we obtain

$$\ln\left(\left(\frac{D-1}{D}\right)^t\right) < \ln\frac{1}{3}$$

$$t \cdot \ln\left(\frac{D-1}{D}\right) < \ln\frac{1}{3}$$

Substituting $D = cn^k$ into $\ln\left(\frac{D-1}{D}\right)$, we obtain

$$\ln\left(\frac{D-1}{D}\right) = \ln\left(\frac{cn^k-1}{cn^k}\right) = \ln\left(1 - \frac{1}{cn^k}\right) \approx -\frac{1}{cn^k}$$

The above equation derivation uses the Taylor expansion approximation, $\ln(1-z) = -z - \frac{z^2}{2} - \frac{z^3}{3} - \cdots \approx -z$, where we assume $z = \frac{1}{cn^k} \to 0$ for large n and higher-order terms are negligible. Thus, we obtain

$$t \cdot \left(-\frac{1}{cn^k}\right) < \ln\frac{1}{3}$$

$$t > \frac{\ln\frac{1}{3}}{-\frac{1}{cn^k}} = c'n^k$$

Here, $c' = -c\ln\frac{1}{3} = 1.0986c$ for $c > 0$ and $\min(k, n-k) = \mathrm{O}(n/\ln n)$.

The above derivation shows that after $c'n^k = \mathrm{O}(n^k)$ repetitions of DSQS using the Dicke state $|D_k^n\rangle$, the probability that a single solution is not observed is less than $\frac{1}{3}$ for $\min(k, n-k) = \mathrm{O}(n/\ln n)$. □

4. DSQS-VCP

In this section, we present an algorithm to generate a quantum circuit based on DSQS using Dicke states to solve the k-VCP. The algorithm is named DSQS-VCP, and the generated quantum circuit is referred to as the DSQS-VCP quantum circuit. Below, we first introduce the DSQS-VCP algorithm and then present its complexity analysis.

4.1. The DSQS-VCP Algorithm

The DSQS-VCP algorithm, whose pseudo-code is shown in Algorithm 1 below, can generate a quantum circuit based on DSQS using the Dicke state to solve the k-VCP by finding all k-vertex covers for a given undirected graph $G = (V, E)$, where the vertex set $V = \{v_0, \ldots, v_{n-1}\}$ is of size n and the edge set $E = \{e_0, \ldots, e_{m-1}\}$ is of size m. The quantum circuit is designed to identify all vertex covers of size k for the graph G.

The DSQS-VCP algorithm first prepares a quantum circuit QC containing $2n+1$ qubits in $|0\rangle$ initially. The qubits consist of n working qubits $x_0, \ldots, x_{n-1}$, grouped into a register x, with each qubit corresponding to a vertex in V. QC also contains $2m$ qubits $f_0, \ldots, f_{2m-1}$ serving as qubits associated with the controlled quantum flag (CQF) gates used by the m edges, where each edge uses two qubits for its associated quantum flag gates, as will be explained later. In addition, QC contains a qubit y as a register y, and n mirror qubits $x'_0, \ldots, x'_{n-1}$, grouped into a register x', as well as n classical bits grouped as cb to store the measurement results of mirror qubits.

First of all, QC applies a $|D_k^n\rangle$ gate to working qubits to make them in the Dicke state $|D_k^n\rangle$. Note that the $|D_k^n\rangle$ gate may be labeled as $\mathrm{D}(n,k)$ in practical implementations later. Afterwards, QC employs many CQF gates, as introduced below. The CQF gate is inspired by the concept of controlled quantum semaphore (CQS) gate proposed in [22]. The function of the CQF gate, which has two flag qubits, is to verify whether a given vertex subset S, specified by the state of register x, covers a specific edge $e = (u, v)$, that is, to check whether $u \in S$ and/or $v \in S$. If so, the first flag qubit, which is initially $|0\rangle$, will be flipped to $|1\rangle$; otherwise, it remains in $|0\rangle$. Note that the second flag qubit functions merely as an ancillary qubit to assist the first flag qubit to be set properly.

Figure 10 shows the block diagram of the CQF gate along with its detailed quantum circuit. The CQF gate, denoted *Flag* in Figure 10, has one control qubit, *ctrl*, and two flag qubits, f_h and f_{h+1}, all initially in $|0\rangle$. Table 1 shows all the possible combinations of gate inputs (i.e., *ctrl*, f_h, and f_{h+1}) and gate outputs (i.e., *ctrl*′, f'_h, and f'_{h+1}) of the gate. Note that $f_{h+1} = |0\rangle$ initially and always remains in $|0\rangle$, so there is no input combination of $f_{h+1} = |1\rangle$ in Table 1. On the contrary, f_h remains unchanged if *ctrl* is $|0\rangle$, and it is definitely

$|1\rangle$ if *ctrl* is $|1\rangle$. The CQF gate is synthesized directly from the truth table with the fewest possible qubits and MCX gates.

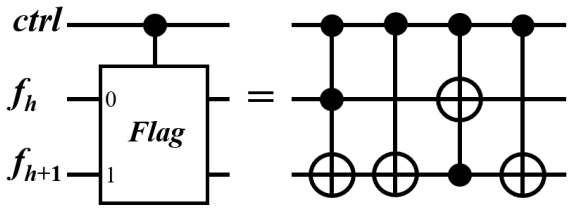


Figure 10. (**left**) Block diagram of a CQF gate, denoted *Flag*, and (**right**) its detailed quantum circuit.

Table 1. State transition of a CQF gate, $(ctrl, f_h, f_{h+1}) \rightarrow (ctrl', f'_h, f'_{h+1})$, with *ctrl* as the control qubit.

$ctrl$	f_h	f_{h+1}	$ctrl'$	f'_h	f'_{h+1}
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	1	0
1	1	0	1	1	0

An edge $e = (v_i, v_j)$ uses two consecutive CQF gates with flag qubits f_h and f_{h+1}, as well as control qubits x_i and $x_j, 0 \leq i, j \leq n-1$, as shown in Figure 11. Note that the two flag qubits are in $|0\rangle$ initially. The possible states of f_h and f_{h+1} are shown in Table 2. The reader can verify that if the control qubit x_i is $|0\rangle$, then both qubits f_h and f_{h+1} remain in $|0\rangle$. In contrast, if the control qubit x_i is $|1\rangle$, then both qubits f_h and f_{h+1} are first set to $|1\rangle$. Afterwards, qubit f_{h+1} is reset to $|0\rangle$ by an X gate. Consequently, if the CQF gate is applied again with the control qubit x_j in $|1\rangle$, then qubit f_h is set to $|1\rangle$ and f_{h+1} is first set to $|1\rangle$ and then reset to $|0\rangle$. Specifically, starting from the initial state of $|f_{h+1}\, f_h\rangle = |00\rangle$, the flag qubit f_h finally remains in $|0\rangle$ only if $x_i = |0\rangle$ and $x_j = |0\rangle$. For the case of $x_i = |1\rangle$ and $x_j = |1\rangle$ and the cases of either $x_i = |1\rangle$ or $x_j = |1\rangle$, the flag qubit f_h is definitely in $|1\rangle$. Note that $x_i = |1\rangle$ (resp., $x_j = |1\rangle$) means that vertex v_i (resp., v_j) is included in the vertex subset corresponding to the state of the working qubits. So, we just check whether the flag qubit f_h is $|1\rangle$ or not. If f_h is $|1\rangle$, then the edge e is covered by v_i and/or v_j.

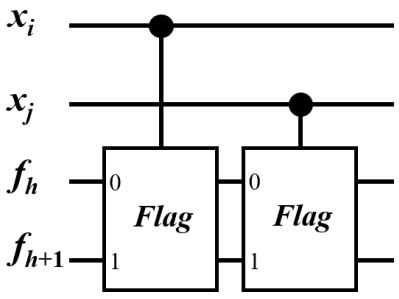


Figure 11. Two consecutive CQF (or *Flag*) gates using two flag qubits f_h and f_{h+1} are employed to check if an edge $e = (v_i, v_j)$ is covered by vertex v_i and/or vertex v_j, where x_i (resp., x_j) of $|1\rangle$ indicates that v_i (resp., v_j) is included in the vertex subset corresponding to the state of working qubits.

After using $2m$ CQF gates to check if each of m edges is covered by at least one vertex, *QC* employs an MCX gate to check if all edges are covered by at least one vertex. The MCX gate takes the flag qubits $f_0, f_2, \ldots, f_{2m-2}$ as control qubits and takes the response qubit y as the target qubit. Therefore, if all edges are covered by at least one vertex, then the response qubit y is $|1\rangle$, meaning that the associated quantum state corresponds to a solution (i.e., a k-vertex cover) of the given k-VCP. Afterwards, n CCX gates are used to reflect the state of

qubit x_i on qubit x'_i with the ith CCX gate taking qubit x_i and qubit y as the control qubits and taking qubit x'_i as the target qubit, where $0 \leq i \leq n-1$.

Table 2. State transition of two consecutive CQF gates, $(x_i, x_j, f_h, f_{h+1}) \rightarrow (x'_i, x'_j, f'_h, f'_{h+1})$, with x_i and x_j as the first and the second control qubits, respectively.

x_i	x_j	f_h	f_{h+1}	x'_i	x'_j	f'_h	f'_{h+1}
0	0	0	0	0	0	0	0
0	1	0	0	0	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	1	1	0

Finally, measuring the n mirror qubits $x'_0, \ldots, x'_{n-1}$ can reveal the solutions to the given k-VCP. If the measurement results are all zeros with probability 1, then the k-VCP has no solution. Otherwise, any non-all-zero result is a solution to the given k-VCP. As described earlier, if there are M solutions, then each non-all-zero measurement result is supposed to have a probability of $1/D$ to be observed, and the all-zero measurement result is obtained, $(D-M)/D$, where $D = \binom{n}{k} = \mathrm{O}(n^k)$ for $\min(k, n-k) = \mathrm{O}(n/\ln n)$.

Algorithm 1 DSQS-VCP

Input: A k-vertex cover problem (k-VCP) instance:
a given positive integer k and a given undirected graph $G = (V, E)$, where the vertex set $V = \{v_0, \ldots, v_{n-1}\}$ is of size n and the edge set $E = \{e_0, \ldots, e_{m-1}\}$ is of size m.
Output: QC: a quantum circuit based on DSQS using the Dicke state $|D^n_k\rangle$ to solve the k-VCP

$QC \leftarrow$ a quantum circuit with
  n working qubits $x_0, \ldots, x_{n-1}$ in $|0\rangle$,
  $2m$ flag qubits $f_0, \ldots, f_{2m-1}$ in $|0\rangle$,
  1 decision qubit y in $|0\rangle$,
  n ancilla qubits $x'_0, \ldots, x'_{n-1}$ in $|0\rangle$, and
  n classical bits grouped as cb
Add a $|D^n_k\rangle$ gate on n qubits $x_0, \ldots, x_{n-1}$ to make them in the Dicke state $|D^n_k\rangle$
$l \leftarrow 0$
for each edge $e = (v_i, v_j) \in E$ **do**
  Add a CQF gate with x_i as the control qubit, and f_l and f_{l+1} as the flag qubits
  Add a CQF gate with x_j as the control qubit, and f_l and f_{l+1} as the flag qubits
  $l \leftarrow l + 2$
end for
Add an MCX gate with $f_0, f_2, \ldots, f_{2m-2}$ as control qubits and y as the target qubit
for $i \leftarrow 0$ to $n-1$ **do**
  Add a CCX gate with x_i and y as control qubits and x'_i as the target qubit
end for
Measure qubits $x'_0, \ldots, x'_{n-1}$ and store the results on classical bits grouped in cb
return QC

4.2. The DSQS-VCP Complexity Analysis

Below, we analyze the time complexity of the DSQS-VCP algorithm. First of all, DSQS-VCP applies a $|D^n_k\rangle$ gate to working qubits. According to [26], a $|D^n_k\rangle$ gate has $\mathrm{O}(kn)$ basis gates, which implies that the time complexity to add the $\mathrm{O}(kn)$ basis gates in the quantum circuit to realize the $|D^n_k\rangle$ gate is also $\mathrm{O}(kn)$. Then, we analyze the other parts of DSQS-VCP. The most computationally intensive part of DSQS-VCP lies in two main loops. The first loop adds two consecutive CQF gates for every edge of totally m edges. The time complexity of the first loop is $\mathrm{O}(m)$. The second loop adds a CCX gate for every vertex of totally n vertices. The time complexity of the second loop is $\mathrm{O}(n)$. Thus, the total time

complexity of the two DSQS-VCP main loops is $O(m+n)$. Consequently, the overall time complexity of DSQS-VCP to generate a quantum circuit is $O(m+kn)$, which is polynomial.

Below, we analyze the circuit complexity of the DSQS-VCP quantum circuit QC in terms of the qubit count, gate count, and circuit depth. In general, QC uses $2m+2n+1$ qubits, including n working qubits $x_0,\ldots,x_{n-1}$ of register x, a total of $2m$ flag qubits $f_0,\ldots,f_{2m-1}$ of CQF gates, a qubit y of register y, and n mirror qubits $x'_0,\ldots,x'_{n-1}$ of register x'.

We first analyze the number of quantum gates and the circuit depth. The high-level gates are a $|D^n_k\rangle$ gate, $2m$ CQF gates, an m-control-qubit Toffoli (C^mX) gate, and n CCX (C^2X) gates. According to [26], a $|D^n_k\rangle$ gate costs $O(kn)$ gates and $O(n)$ depth. Based on Figure 10, a CQF gate uses two CX gates and two X gates with constant depth inside the CQF gate quantum circuit. Thus, $2m$ CQF gates cost $O(m)$ gates and $O(m)$ depth.

Below, we analyze the C^mX gate and the n C^2X gates. Numerous studies have tried to decompose the C^mX gate into gates in a selected universal basis gate set. Note that a universal basis gate set is a finite set of quantum gates that can be combined to approximate any quantum operation (or gate). For example, the set of an H gate, a T gate, and a CX gate (i.e., H, T, CX) is a widely used universal basis gate set. The authors in [32] selected H, T, and CX gates as the basis gates to decompose a C^mX gate into $32m-96=O(m)$ T gates and $24m-72=O(m)$ CX gates with the circuit depth $216m-648=O(m)$, along with an extra ancilla qubit. The authors in [33] show that a CCX gate can be decomposed into a constant number of H, T, and CX gates without any extra ancilla qubit with a constant depth. Therefore, the C^mX gate and the n C^2X gates cost $O(m+n)$ basis gates and $O(m+n)$ depth with an extra ancilla qubit.

From all of the above-mentioned analysis results, we conclude that the DSQS-VCP quantum circuit QC uses totally $2m+2n+2=O(m+n)$ qubits and $O(m+kn)$ gates with $O(m+n)$ depth. We can see that the qubit count, the gate count, and the circuit depth of QC are all polynomial.

Constructing quantum circuits to solve intractable problems, particularly NPC problems such as the k-VCP, is challenging due to the need to encode combinatorial constraints into oracle operations, often resulting in high complexity in terms of qubit usage, gate count, and circuit depth. The DSQS-VCP algorithm proposed in this section addresses this challenge by systematically generating a DSQS-based quantum circuit for solving instances of the k-VCP. Our analysis demonstrates that the resulting quantum circuit, including the oracle component that enforces the problem constraints, has a polynomial complexity with respect to the problem size in terms of construction time complexity, qubit usage, gate counts, and circuit depth. Notably, a significant portion of the overall circuit corresponds to the oracle implementation. This highlights the practical advantage of the DSQS algorithm, which requires only a single oracle invocation per execution, effectively reducing the cumulative cost associated with complex oracle constructions and their repetitions.

5. Experimental Results

This section shows two experiments using IBM Qiksit packages to implement and run DSQS-VCP quantum circuits based on DSQS using the Dicke state $|D^n_k\rangle$ to successfully find all solutions to given k-VCP instances. The experiments rely on IBM Aer Simulator with n^k or 5000 shots (repetitions) per experiment, where n is the number of input qubits. Since this paper primarily addresses the proof of principle and complexity analysis of DSQS and DSQS-VCP, we do not conduct experiments to execute DSQS-VCP quantum circuits on real quantum computers. The details of the experiments are described in the following subsections.

5.1. The First Experiment

The first experiment is to implement and run the DSQS-VCP quantum circuit to find all solutions to the k-VCP defined below. Given an undirected graph $G = (V, E)$ with $n = 5$ vertices $v_0, \ldots, v_4$ in V and $m = 4$ edges $e_0, \ldots, e_3$ in E, as shown in Figure 2, the k-VCP is associated with different k values, such as 1, 2, and 3.

Figure 12 shows the DSQS-VCP quantum circuit to find all solutions to the k-VCP with $k = 1$ in the first experiment. The quantum circuit has $n = 5$ working qubits $x_0, \ldots, x_4$ for vertices $v_0, \ldots, v_4$ in vertex set V. With a Dicke state $|D_k^n\rangle = |D_1^5\rangle$ preparation gate, the working qubits are set to $|D_1^5\rangle$ state to form the space of $D = \binom{n}{k} = \binom{5}{1} = 5$ possible input instances. Note that the quantum circuits for $k = 2$ and $k = 3$ are all the same, except that they use different Dicke state preparation gates. As mentioned earlier, an edge $e = (v_i, v_j)$ uses two consecutive CQF gates (denoted "Flag" gates in Figure 12) having two flag qubits and two different control qubits, x_i and x_j, to check whether the edge e is covered by v_i and/or v_j. For example, in the DSQS-VCP quantum circuit of the first experiment, two flag qubits f_0 and f_1 in $|0\rangle$ are initially used to check the coverage of the edge $e_0 = (v_0, v_1)$ by vertices v_0 and/or v_1. It is required to check whether the flag qubit f_0 is $|1\rangle$ or not to determine if edge e_0 is covered by v_0 and/or v_1. The coverage of edges $e_1, \ldots, e_3$ can be checked similarly.

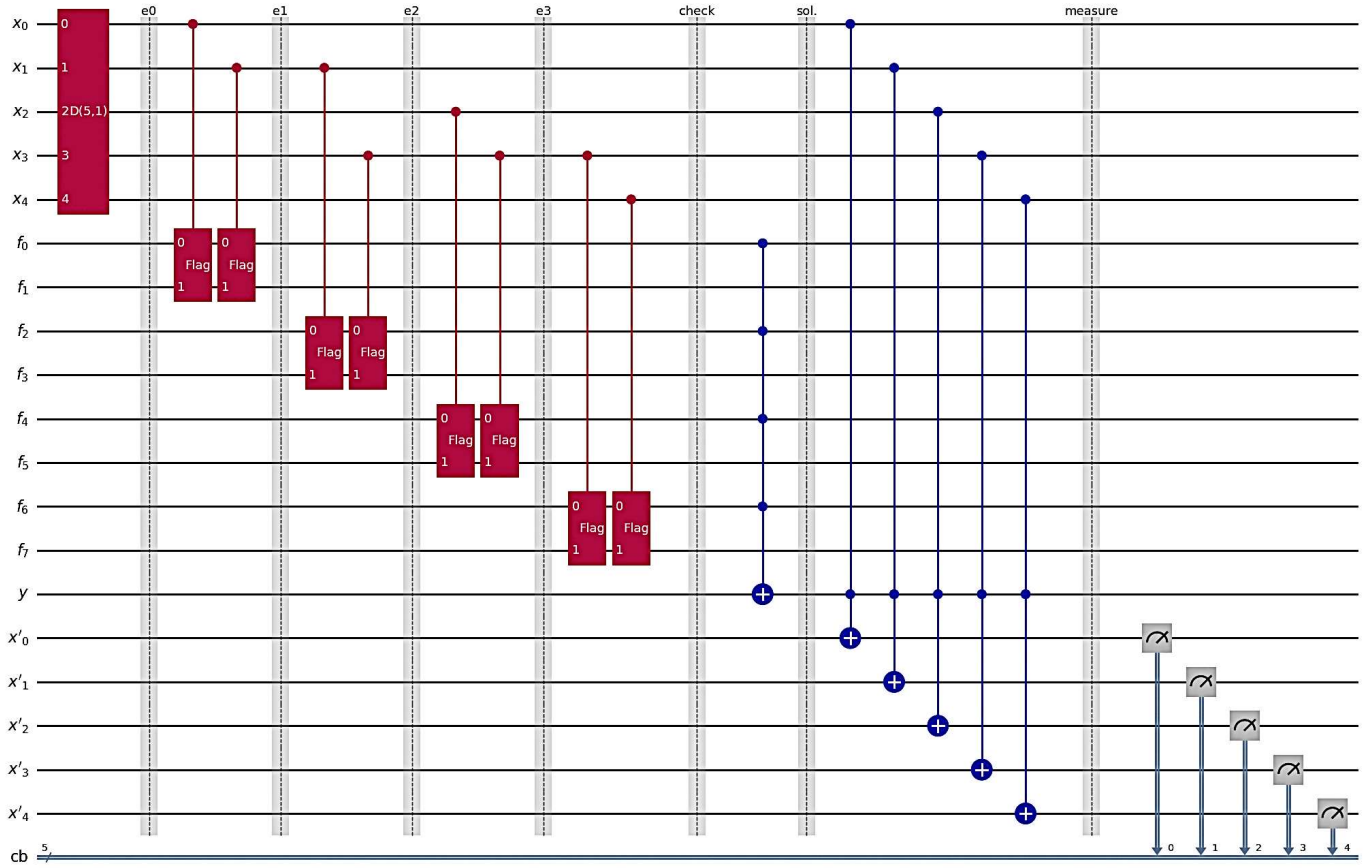


Figure 12. The DSQS-VCP quantum circuit to find all solutions to the k-VCP in the first experiment.

The decision qubit y, which is initially $|0\rangle$, then serves as the target qubit of an MCX gate that takes the first flag qubit of all $2m = 8$ quantum flag gates as its control qubits. Therefore, if all of the first flag qubits are $|1\rangle$, then qubit y is flipped from $|0\rangle$ to $|1\rangle$. This means that the bit combination corresponding to the quantum state is a solution to the given k-VCP.

Afterwards, $n = 5$ CCX gates, each of which takes qubits y and x_i as two control qubits and qubit x_i' as the target qubit, are appended to the quantum circuit. This is intended to reflect the quantum states of working qubits $x_0, \ldots, x_{n-1}$ corresponding to solutions

on mirror qubits $x'_0, \ldots, x'_{n-1}$, where $0 \leq i \leq n-1 = 4$. Therefore, the mirror qubits are in the superposition of all quantum states corresponding to all M solutions and the state $|0\rangle^{\otimes n}$. It is notable that the quantum state corresponding to a solution has a probability amplitude of $\sqrt{\frac{1}{D}}$ and the quantum state $|0\rangle^{\otimes n}$ has a probability amplitude of $\sqrt{\frac{D-M}{D}}$, where $D = \binom{n}{k} = \binom{5}{2} = 10$. It is easy to check whether the given k-VCP has no solution. If so, the mirror qubits $x'_0, \ldots, x'_{n-1}$ are definitely in $|0\rangle^{\otimes n}$ with a probability amplitude of 1.

Figures 13–15 show the histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for k = 1, 2, and 3 in the first experiment. In Figure 13, there is only one outcome 00000 with a probability of 1 for $k = 1$ with either $n^k = 5^1 = 5$ shots or 5000 shots, which means the k-VCP has no solution for the case of k = 1. Note that we only present one diagram in Figure 13, as diagrams with 5 shots and 5000 shots are the same.

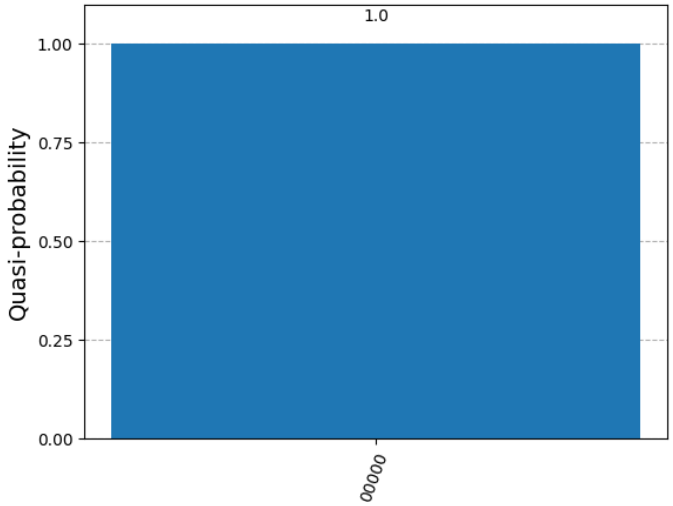
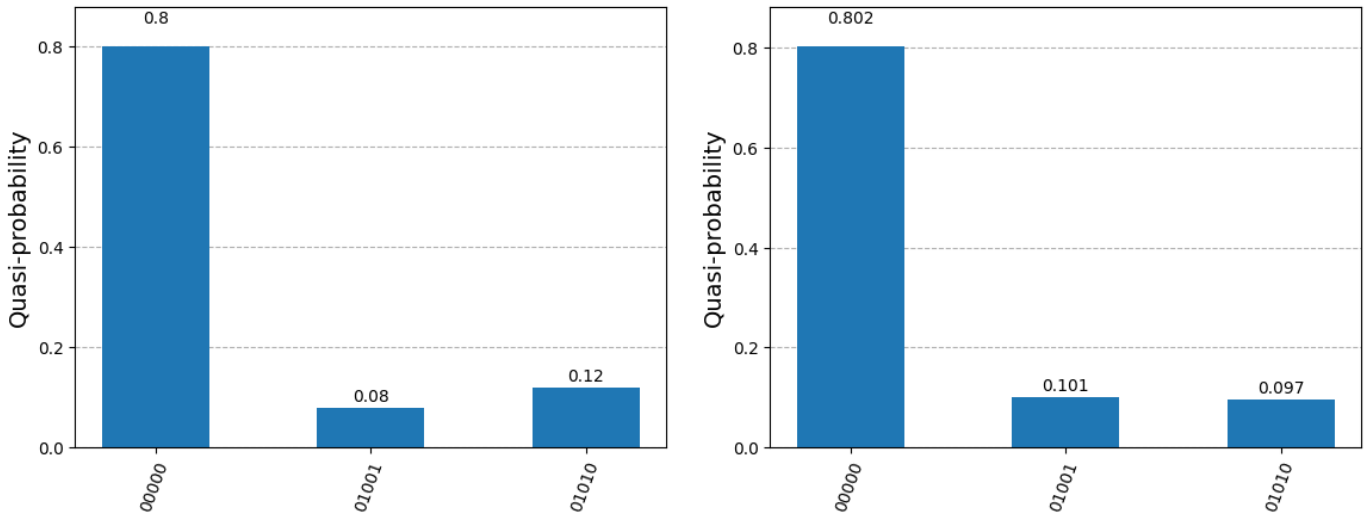


Figure 13. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 1$ with $n^k = 5^1 = 5$ and 5000 shots in the first experiment.



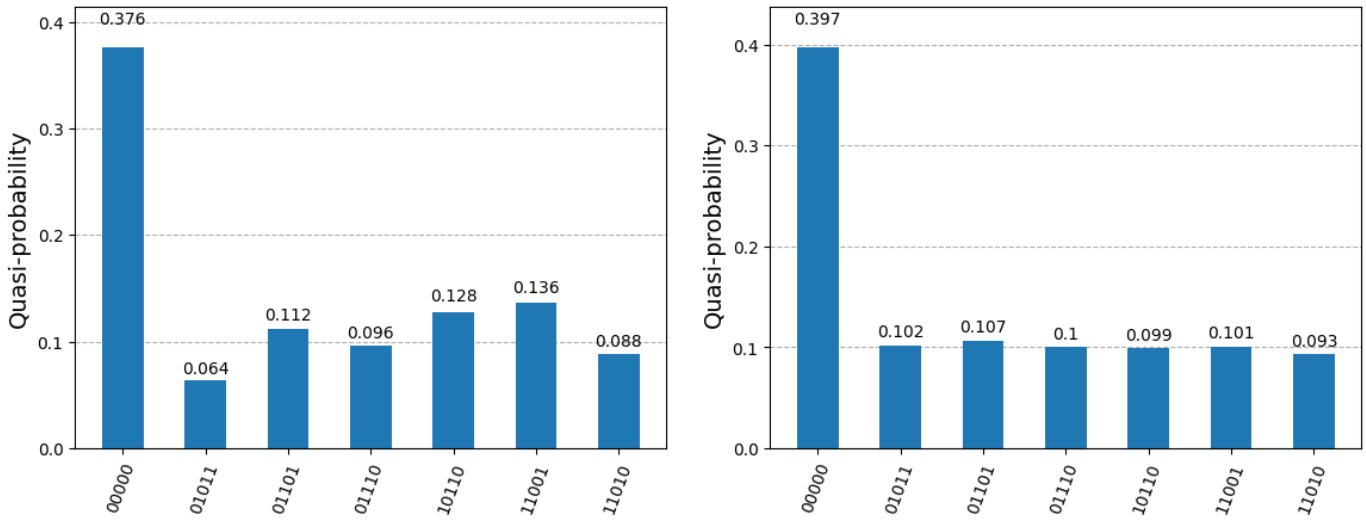
(**a**) $n^k = 5^2 = 25$ shots (**b**) 5000 shots

Figure 14. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 2$ in the first experiment.

From Figure 14a,b, we can observe that there are $M = 2$ outcomes, 10001 and 10100, with probabilities of approximately $1/D = 1/10$ for $n^k = 5^2 = 25$ shots and 5000 shots,

where $D = \binom{n}{k} = \binom{5}{2} = 10$. More shots lead to probabilities that are closer to $1/D = 1/10$. Specifically, they correspond to two solutions, $\{v_0, v_1\}$ and $\{v_2, v_4\}$, to the given k-VCP for $k = 2$. The solutions correspond to two-vertex covers of the graph in the given k-VCP. There are totally $D - 2 = 8$ non-solutions among $D = 10$ input instances, indicated by the outcome 00000 with a probability that is approximately 8/10.

From Figure 15a,b, we can observe that there are $M = 6$ outcomes, 01110, 10011, 10101, 10110, 11001, and 11100, with probabilities of approximately $1/D = 1/10$ for $n^k = 5^3 = 125$ shots and 5000 shots, where $D = \binom{n}{k} = \binom{5}{3} = 10$. More shots lead to probabilities that are closer to $1/D = 1/10$. Specifically, they correspond to all the six three-vertex covers of the graph in the given k-VCP. There are, in total, $D - M = 4$ non-solutions among $D = 10$ input instances, indicated by the outcome 00000, with a probability of approximately 4/10.



(**a**) $n^k = 5^3 = 125$ shots (**b**) 5000 shots

Figure 15. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 3$ in the first experiment.

5.2. The Second Experiment

The second experiment is to implement and run the DSQS-VCP quantum circuit to find all solutions to the k-VCP defined below. Given a undirected graph $G = (V, E)$ with $n = 7$ vertices $v_0, \ldots, v_6$ in V and $m = 7$ edges $e_0, \ldots, e_6$ in E, as shown in Figure 3, the k-VCP is associated with different k values, such as 2, 3, and 4.

Figure 16 shows the DSQS-VCP quantum circuit to find all solutions to the k-VCP with $k = 2$ in the second experiment. The quantum circuit has $n = 7$ working qubits $x_0, \ldots, x_6$ for vertices $v_0, \ldots, v_6$ in vertex set V. With a Dicke state $|D_k^n\rangle = |D_2^7\rangle$ preparation gate, the working qubits are set to $|D_2^7\rangle$ state to form the space of $D = \binom{n}{k} = \binom{7}{2} = 21$ possible input instances. Note that the quantum circuits for $k = 3$ and $k = 4$ are all the same, except that they use different Dicke state preparation gates. As mentioned earlier, an edge $e = (v_i, v_j)$ uses two consecutive CQF gates (denoted "Flag" gates in Figure 16) having two flag qubits and two different control qubits, x_i and x_j, to check whether the edge e is covered by v_i and/or v_j, where $0 \le i, j \le n - 1$. For example, in the DSQS-VCP quantum circuit of the second experiment, two flag qubits, f_0 and f_1 in $|0\rangle$, initially are used to check the coverage of edge $e_0 = (v_0, v_1)$ by vertices v_0 and/or v_1. It is required to check whether the flag qubit f_0 is $|1\rangle$ or not to determine if edge e_0 is covered by v_0 and/or v_1. The coverage of edges $e_1, \ldots, e_6$ can be checked similarly.

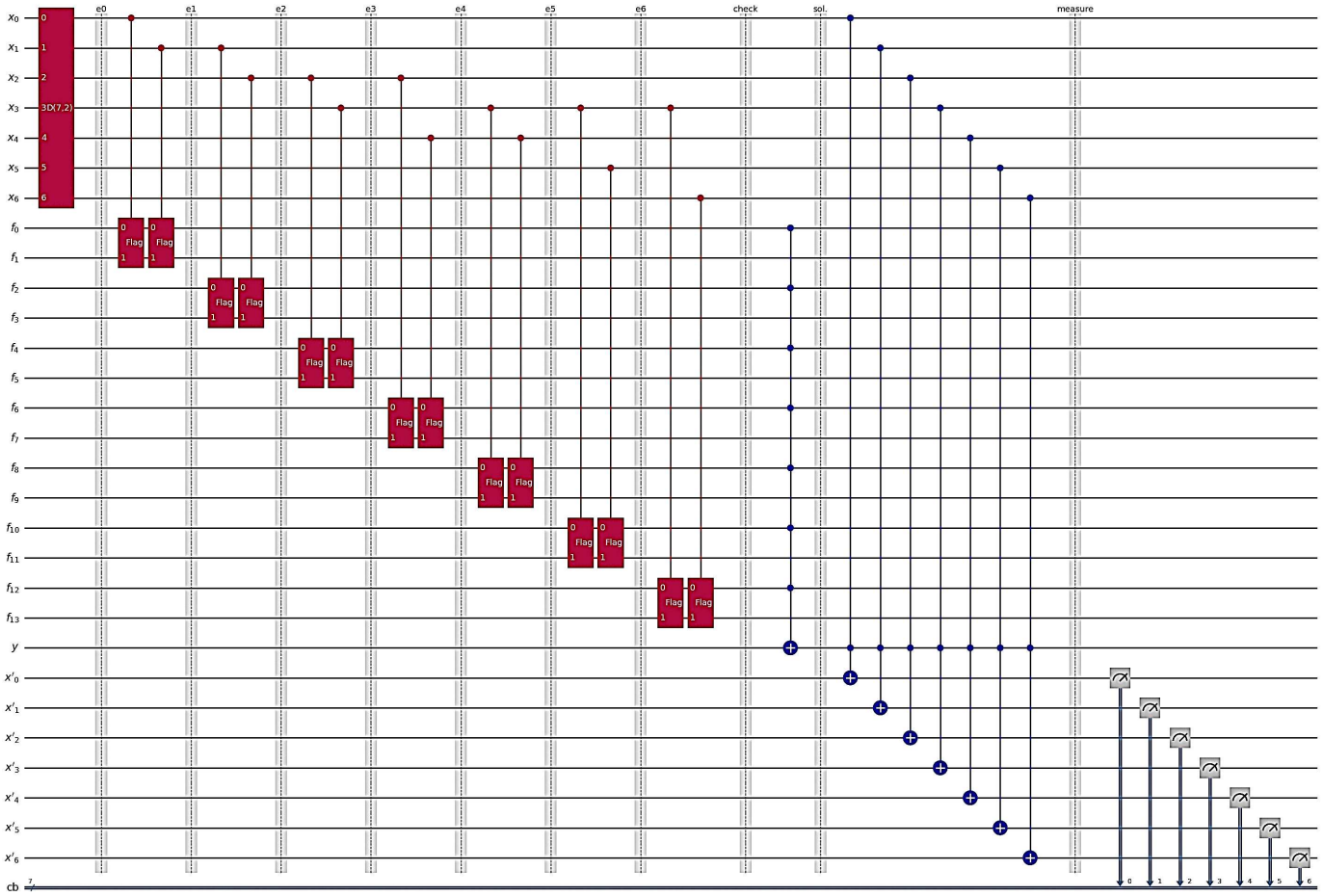


Figure 16. The DSQS-VCP quantum circuit to find all solutions to the k-VCP in the second experiment.

The decision qubit y, which is initially $|0\rangle$, then serves as the target qubit of an MCX gate that takes the first flag qubit of all $2m = 14$ quantum flag gates as its control qubits. Therefore, if all of the first flag qubits are $|1\rangle$, then qubit y is flipped from $|0\rangle$ to $|1\rangle$. This means that the bit combination corresponding to the quantum state is a solution to the given k-VCP.

Afterwards, $n = 7$ CCX gates, each of which takes qubits y and x_i as two control qubits and qubit x'_i as the target qubit, are appended into the quantum circuit. This is intended to reflect the quantum states of working qubits $x_0, \ldots, x_{n-1}$ corresponding to solutions into mirror qubits $x'_0, \ldots, x'_{n-1}$, where $0 \leq i \leq n-1 = 6$. Therefore, the mirror qubits are in the superposition of all quantum states corresponding to all M solutions and the state $|0\rangle^{\otimes n}$. It is notable that the quantum state corresponding to a solution has a probability amplitude of $\sqrt{\frac{1}{D}}$ and the quantum state $|0\rangle^{\otimes n}$ has a probability amplitude of $\sqrt{\frac{D-M}{D}}$, where $D = \binom{n}{k} = \binom{7}{2} = 21$. It is easy to check whether the given k-VCP has no solution. If so, the mirror qubits $x'_0, \ldots, x'_{n-1}$ are definitely in $|0\rangle^{\otimes n}$ with a probability amplitude of 1.

Figures 17–19 show histograms of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for k = 2, 3, and 4 in the second experiment. In Figure 17, there is only one outcome, 0000000, with a probability of 1 for $k = 2$ with either $n^k = 7^2 = 49$ shots or 5000 shots, which means the k-VCP has no solution for the case of k = 2. Note that we only present one diagram in Figure 17, as diagrams with 49 shots and 5000 shots are the same.

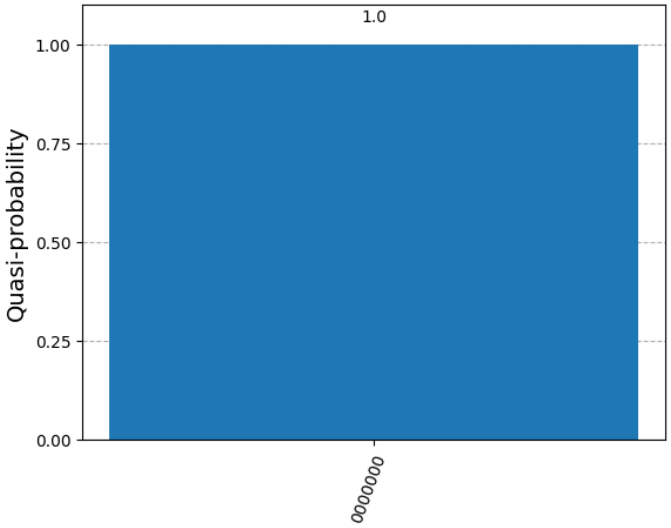
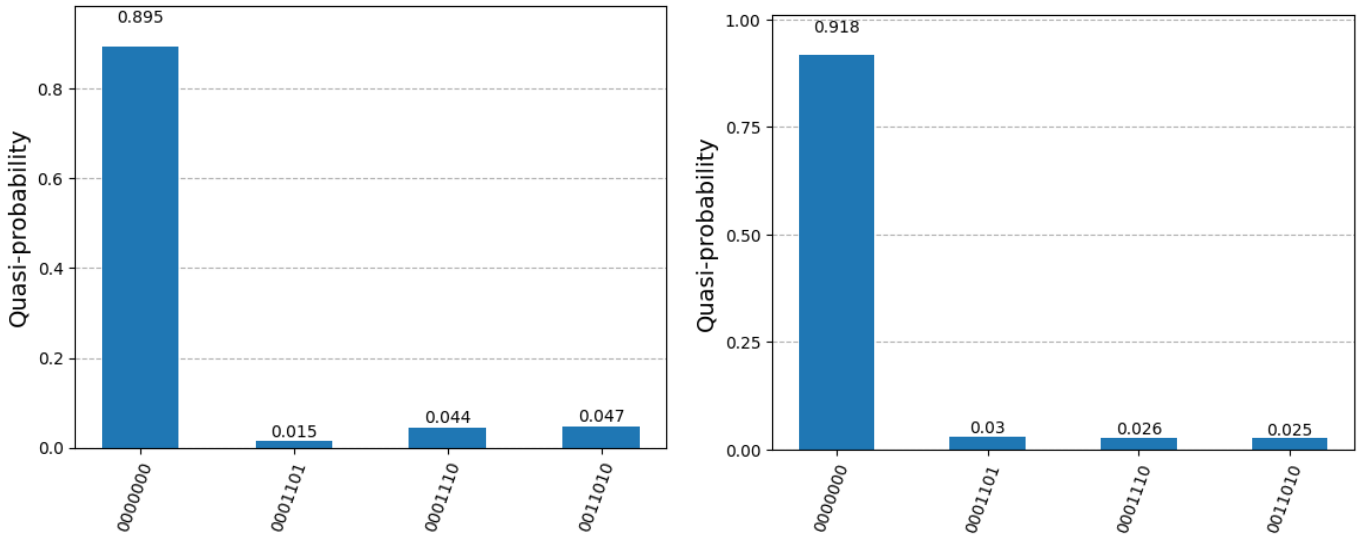


Figure 17. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 2$ with $n^k = 7^2 = 49$ and 5000 shots in the second experiment.

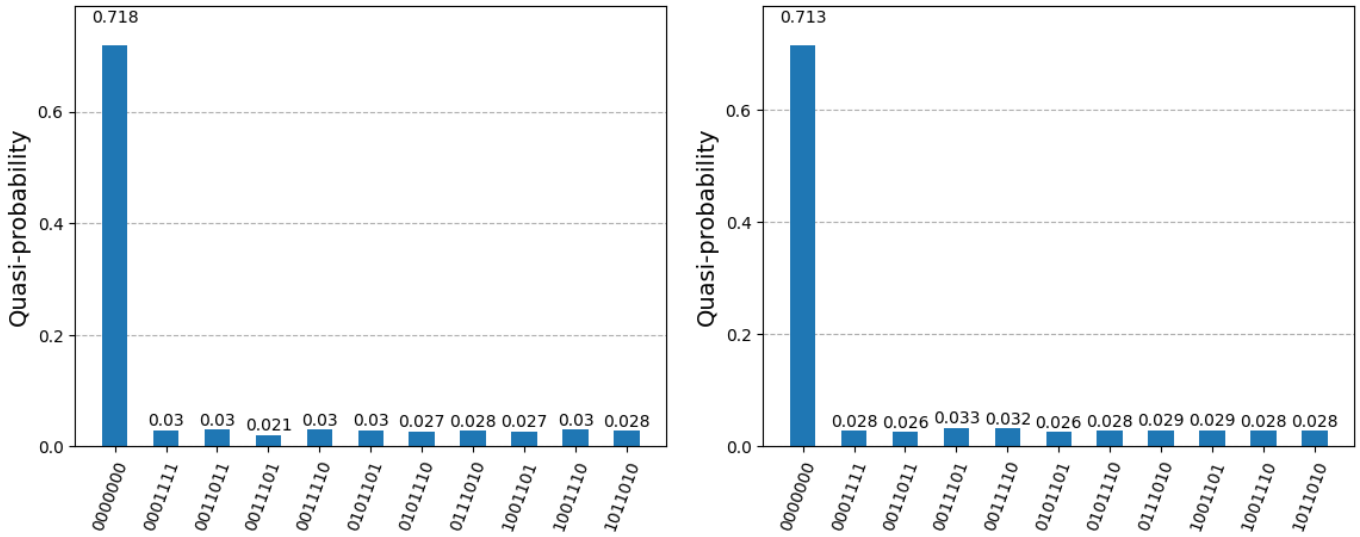


(**a**) $n^k = 7^3 = 343$ shots (**b**) 5000 shots

Figure 18. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 3$ in the second experiment.

In Figure 18a,b, we can observe that there are $M = 3$ outcomes, 0001101, 0001110, and 0011010, with probabilities of approximately $1/D = 1/\binom{7}{3} = 1/35 \approx 0.0285$. More shots lead to probabilities that are closer to $1/D = 1/35$. Specifically, they correspond to three solutions, $\{v_0, v_2, v_3\}$, $\{v_1, v_2, v_3\}$, and $\{v_1, v_3, v_4\}$, to the given k-VCP for $k = 3$. The solutions correspond to three-vertex covers of the graph in the given k-VCP. There are a total of $D - 3 = 32$ non-solutions among $D = 35$ input instances, indicated by the outcome 0000000 with a probability of approximately $32/35 \approx 0.9142$.

In Figure 19a,b, we can observe that there are $M = 10$ outcomes, 0001111, 0011011, 0011101, 0011110, 0101101, 0101110, 0111010, 1001101, 1001110, and 1011010, with probabilities of approximately $1/D = 1/\binom{7}{4} = 1/35 \approx 0.0285$. More shots lead to probabilities closer to $1/D = 1/35$. Specifically, they correspond to all the ten four-vertex covers of the graph in the given k-VCP. There are, in total, $D - M = 25$ non-solutions among $D = 35$ input instances, indicated by the outcome 0000000, with a probability of approximately $25/35 \approx 0.7142$.



(**a**) $n^k = 7^4 = 2401$ shots (**b**) 5000 shots

Figure 19. Histogram of the measurement results of the DSQS-VCP quantum circuit to find all solutions to the k-VCP for $k = 4$ in the second experiment.

6. Conclusions

This paper first proposes a quantum search algorithm, named DSQS, as a general quantum algorithm to set qubits in the Dicke state $|D_k^n\rangle$ to identify all target inputs or solutions of specific patterns from unstructured input instances by calling an oracle only once. DSQS improves Grover's algorithm, which needs to repeat an oracle and a diffuser $\left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rfloor = \mathrm{O}(\sqrt{2^n})$ times to find all M solutions out of $N = 2^n$ inputs with high probability, where n is the number of input qubits to span the search space. Furthermore, Grover's algorithm requires prior knowledge of the number M of solutions, whereas DSQS operates without this prerequisite. We prove the correctness of DSQS by unitary transformations and show that DSQS can locate every single solution with an error probability less than $1/3$ through $\mathrm{O}(n^k)$ repetitions for $\min(k, n-k) \ll n/2$.

Our research result does not conflict with the assertion that Grover's algorithm is the optimal oracle-based quantum search algorithm. This is because that DSQS sets qubits into the Dicke state $|D_k^n\rangle$ of D in superposition states, reducing the number of search states for the oracle to be $D = \binom{n}{k} = \mathrm{O}(n^k)$ under the condition of $\min(k, n-k) \ll n/2$. Grover's algorithm calls an oracle and the diffuser $\mathrm{O}(\sqrt{2^n})$ times to find solutions with high probability through one repetition. In contrast, DSQS calls an oracle once to locate solutions with a probability of $1/D$ through one repetition. Afterwards, DSQS can locate solutions with high probability through $\mathrm{O}(n^k)$ repetitions for $\min(k, n-k) \ll n/2$.

This paper also proposes a classical algorithm, named DSQS-VCP, to generate a quantum circuit based on DSQS using the Dicke state $|D_k^n\rangle$ to solve the search version k-VCP by locating all solutions to the k-VCP. We analyze the time complexity of DSQS-VCP and decompose its generated quantum circuit into components consisting of only H, T, and CX gates, which are basis gates of a universal gate set. This decomposed circuit is shown to have a polynomial qubit count, gate count, and circuit depth. The overall complexity analysis reveals that DSQS-VCP can generate a quantum circuit based on DSQS using the Dicke state $|D_k^n\rangle$ to find all k-VCP solutions in polynomial time with a probability of error less than $1/3$ for $\min(k, n-k) \ll n/2$. Since the k-VCP is NP-complete, NP and NPC problems can be polynomially reduced to the k-VCP. If the reduced k-VCP instance satisfies $\min(k, n-k) \ll n/2$, then both the instance and the original NP/NPC problem instance to

which it corresponds can be solved by the DSQS-VCP quantum circuit in polynomial time with an error probability less than $1/3$.

We conducted experiments using IBM Qiskit packages to implement and run the proposed QSDS-VCP quantum circuits, successfully identifying all solutions to the example k-VCP instances. The instances are for small n and k, such as $(n = 5, k = 1, 2, 3)$ and $(n = 7, k = 2, 3, 4)$, all of which satisfy the condition $\min(k, n-k) \ll n/2$. However, these experiments were performed using IBM Qiskit simulators and do not consider real-device conditions such as qubit decoherence, gate noise, or connectivity constraints. As this paper focuses on theoretical aspects and proof of principle, experiments on real quantum computers were not conducted. In the future, we plan to extend this work with implementations on real quantum computers and conduct detailed error resilience analysis to assess the algorithm's performance under realistic noise models. Additionally, we intend to continue exploring theoretical aspects, particularly investigating how to use the first half of the DSQS quantum circuit that contains only registers x and y to estimate the total number M of solutions. We aim to derive the error bound of the estimation of M and then determine the minimum number of repetitions required for DSQS to identify all M solutions with high probability. Furthermore, we plan to explore whether certain NP and NPC problems can be reduced to k-VCP satisfying $\min(k, n-k) \ll n/2$, so that they can be solved in polynomial time with an error probability less than $1/3$ via the DSQS-VCP quantum circuit. We also seek to design algorithms similar to DSQS-VCP to directly generate quantum circuits based on DSQS using the Dicke state $|D_k^n\rangle$ for solving certain NP and NPC problems in polynomial time with an error probability less than $1/3$ for $\min(k, n-k) \ll n/2$.

The paper [34] proposes an improved Grover's algorithm by using a diffuser tailored for the Dicke state to exploit quantum interference for amplifying the probability amplitudes of the solution states. In the Dicke state $|D_k^n\rangle$, the number of in-superposition states is given by $D = \binom{n}{k}$, which is of $O(n^k)$ under the condition of $\min(k, n-k) \ll n/2$. Therefore, a total of $R = \lfloor \frac{\pi}{4}\sqrt{D/M} \rfloor$ Grover iterators, each of which is composed of an oracle and a diffuser, are required to find all M solutions with high probability. This reduces the required repetitions of the Grover iterator from $O(\sqrt{2^n})$ in the original Grover's algorithm to only $O(\sqrt{n^k})$ as long as $\min(k, n-k) \ll n/2$. However, the improved Grover's algorithm still requires prior knowledge of the number M of solutions to accurately determine the optimal number R of Grover iterator repetitions. According to [7], Grover's algorithm suffers from the "overshooting problem", in which repeating the Grover iterator over R times causes the probability amplitudes of the solution states to decrease.

In the future, we plan to integrate the improved Grover's algorithm proposed in [34] with the DSQS algorithm proposed in this paper to leverage the advantages of both methods. The basic idea of the integrated method is to initially assume an estimated number $\hat{M} = \beta D$ of solutions, where $0 < \beta < 1$. With $\hat{M} = \beta D$, we can then assume an estimated optimal number $\hat{R} = \lfloor \frac{\pi}{4}\sqrt{D/\hat{M}} \rfloor$ of Grover iterator repetitions without knowing the exact value of M. When $\hat{M} \geq M$, we have $\hat{R} \leq R$. Thus, we can repeat the Grover iterator only $\hat{R}$ times, where $\hat{R} \leq R$, not only to reduce the quantum circuit depth and the number of quantum gates used, but also to avoid the overshooting problem. After that, we apply the special oracle of the DSQS algorithm and then determine all solutions through a suitable number of measurements. Because the diffuser amplifies the amplitudes of the solution states, the number of measurements or shots required is reduced. We will analytically derive the appropriate number of measurements by examining the amplitude amplification induced by the diffuser and conduct experiments to validate our analysis. We will also design an adaptive algorithm that assumes successive estimated values of the number of solutions, $\hat{M}$, from larger to smaller. Consequently, the corresponding values of $\hat{R}$ increases, but the

required number of measurements decreases gradually. The adaptive algorithm stops when the number of identified solutions exceeds $\hat{M}$, enabling the integrated method to adaptively and efficiently identify solutions. This aims to pursue a resource-efficient optimization objective: reduce the number of Grover iterator repetitions as much as possible and, at the same time, minimize the number of shots required by the DSQS quantum circuit, all without prior knowledge of the number M of solutions.

While we have outlined the integration of DSQS with the improved Grover's algorithm proposed in [34], we note that a rigorous theoretical or empirical comparison between the two standalone approaches has not yet been conducted. Such a comparison—especially in terms of the qubit count, gate count, circuit depth, and number of shots required to locate solutions—will be pursued in future work to better position the advantages and trade-offs of each algorithm.

Knowing the exact number M of solutions can benefit the improved Grover's algorithm proposed in [34], as it allows for determining the appropriate number of Grover iterator repetitions to maximize the amplitude of the target states without causing amplitude overshooting. However, in many practical scenarios, M is unknown. One way to estimate M is by using the quantum counting algorithm [13], which requires $2^t - 1$ Grover iterator repetitions when t counting qubits are used. Alternatively, we may estimate M by using only a portion of the DSQS quantum circuit proposed in this paper—specifically, the part consisting of the n-qubit register x, the single-qubit register y, one Dicke state preparation gate, one oracle, and the final measurement on the y register. Although a formal proof has not yet been established, it is expected that this partial DSQS circuit can estimate the value of M with low error probability after a number of measurements proportional to $D = \binom{n}{k}$, which is $\mathrm{O}(n^k)$ and remains polynomial as long as the condition $\min(k, n-k) \ll n/2$ is satisfied. In the future, we plan to integrate this partial circuit with the amplitude amplification mechanism of the improved Grover's algorithm proposed in [34] to further reduce the number of measurements required by the partial circuit for the solution count estimation. We will conduct an in-depth analysis of the relationship between the number of repetitions of the Grover iterator and the number of measurements needed to accurately determine the solution count. The goal of this integration is to perform a small number of Grover iterations while allowing for the partial circuit to accurately estimate the solution count with significantly fewer measurements.

We briefly recall that BQP (Bounded-error Quantum Polynomial time) is the class of decision problems solvable by a family of polynomial complexity quantum Turing machines (quantum circuits) with error bounded by a constant (e.g., a constant $\epsilon \leq 1/3$) [14,28]. It is noted that all statements of polynomial complexity in this paper apply only to VCP instances and similar instances of other problems for which the feasible solution set lies in a fixed-Hamming-weight Dicke state subspace with a polynomial size of $D = \binom{n}{k} = \mathrm{O}(n^k)$ under the condition of $min(k, n-k) \ll n/2$. DSQS samples uniformly from that subspace of size D, yielding per-shot solution probability $1/D$ to observe a single solution. Consequently, the $D = \mathrm{O}(n^k)$ repetition bound is required, which is not asymptotically better than exhaustive enumeration methods. The required shot budgets are therefore probabilistic demonstrations rather than certificates that all D candidates are examined. Nevertheless, DSQS may be combined with amplitude amplification algorithms confined to the Dicke state subspace, such as the improved Grover's algorithm [34], to boost single-shot success probabilities of observing a solution. Appendix A shows shows the results of a preliminary experiment of the method combining DSQS with the improved Grover algorithm [34] using Dicke states. This experiment, using the same setting as Experiment 1 in Section 5, indicates that amplitude amplification can significantly enhance the probability of finding target solutions. Since the correctness of the combined method cannot be validated

solely by this preliminary experiment, nor can the improvement in success probability be analytically established, we will conduct a comprehensive exploration covering theoretical derivations, probability analyses, and experiments across various scenarios. Our goal is to verify that DSQS can enhance the effectiveness of quantum search algorithms in locating target solutions.

Funding: This research was supported in part by the National Central University (NCU) and in part by the National Science and Technology Council (NSTC) under Grant 113-2622-E-008-019 and Grant 114-2119-M-006-006.

Data Availability Statement: Since the related technology described in the paper is currently under patent application, the data presented in this study are available on request from the corresponding author.

Acknowledgments: We express our gratitude to the National Taiwan University—IBM Quantum Computing Center (IBM Q Hub at NTU) for providing us with access to the IBM Q system.

Conflicts of Interest: The author declares no conflicts of interest.

Appendix A

This appendix shows the results of a preliminary experiment of the method combining DSQS with the Dicke state-based Grover's algorithm [34] to find all solutions to the k-VCP. This preliminary experiment uses the same experimental setting as Experiment 1 in Section 5. Specifically, the experiment is to solve the k-VCP for an undirected graph $G = (V, E)$ with $n = 5$ vertices $v_0, \ldots, v_4$ in V and $m = 4$ edges $e_0, \ldots, e_3$ in E, as shown in Figure 2, with $k = 2$.

Figure A1 shows the quantum circuit to find all solutions to the k-VCP with $k = 2$. The quantum circuit has $n = 5$ working qubits $x_0, \ldots, x_4$ for vertices $v_0, \ldots, v_4$ in vertex set V. With a Dicke state $|D_k^n\rangle = |D_2^5\rangle$ preparation gate, the working qubits are set to $|D_2^5\rangle$ state to form the space of $D = \binom{n}{k} = \binom{5}{2} = 10$ possible input instances. An edge $e = (v_i, v_j)$ uses two consecutive CQF gates (denoted "F" gates in Figure A1) having two flag qubits and two different control qubits, x_i and x_j, to check whether the edge e is covered by v_i and/or v_j. An ancilla qubit z, which is set to $|-\rangle$ by an X gate and an H gate, then serves as the target qubit of an MCX gate that takes the first flag qubit of all $2m = 8$ quantum flag gates as its control qubits. Therefore, if all of the first flag qubits are $|1\rangle$, then qubit z is forced to trigger the phase kickback effect. The phase of the quantum state that makes the first flag qubit of all $2m = 8$ quantum flag gates is then reversed.

Afterwards, a series of inverse CQF gates (denoted "iF" gates in Figure A1) are applied in the reverse order of their corresponding CQF gates. The iF gates reset the flag qubits by uncomputing the auxiliary information, restoring them to $|0\rangle$ and eliminating residual entanglement, thereby ensuring that only the edge cover constraints contribute to the subsequent amplitude amplification.

In the subsequent circuit, a diffuser confined to the Dicke state subspace is applied. According to [34], the diffuser is constructed by an iD(5,2) gate, n X gates, a $C^{n-1}Z$ gate, n X gates, and a D(5,2) gate, where iD(5,2) denotes the inverse Dicke state construction gate of $|D_2^5\rangle$. The diffuser performs a reflection about the mean of all states in the Dicke state subspace, ensuring that the amplitude amplification process is confined to the space spanned by $|D_2^5\rangle$.

The final part in the quantum circuit is the DSQS algorithm, which ensures that the measurement yields the states corresponding to the target solutions (whose amplitudes have already been amplified by the diffuser), while the non-solution states are all mapped to the all-zero state.

Figure A2 shows the histogram of the measurement results of the quantum circuit to find all solutions to the k-VCP with $k = 2$. We observe that, by leveraging the amplitude amplification of the improved Grover's algorithm, the probability of finding a target solution can be significantly boosted.

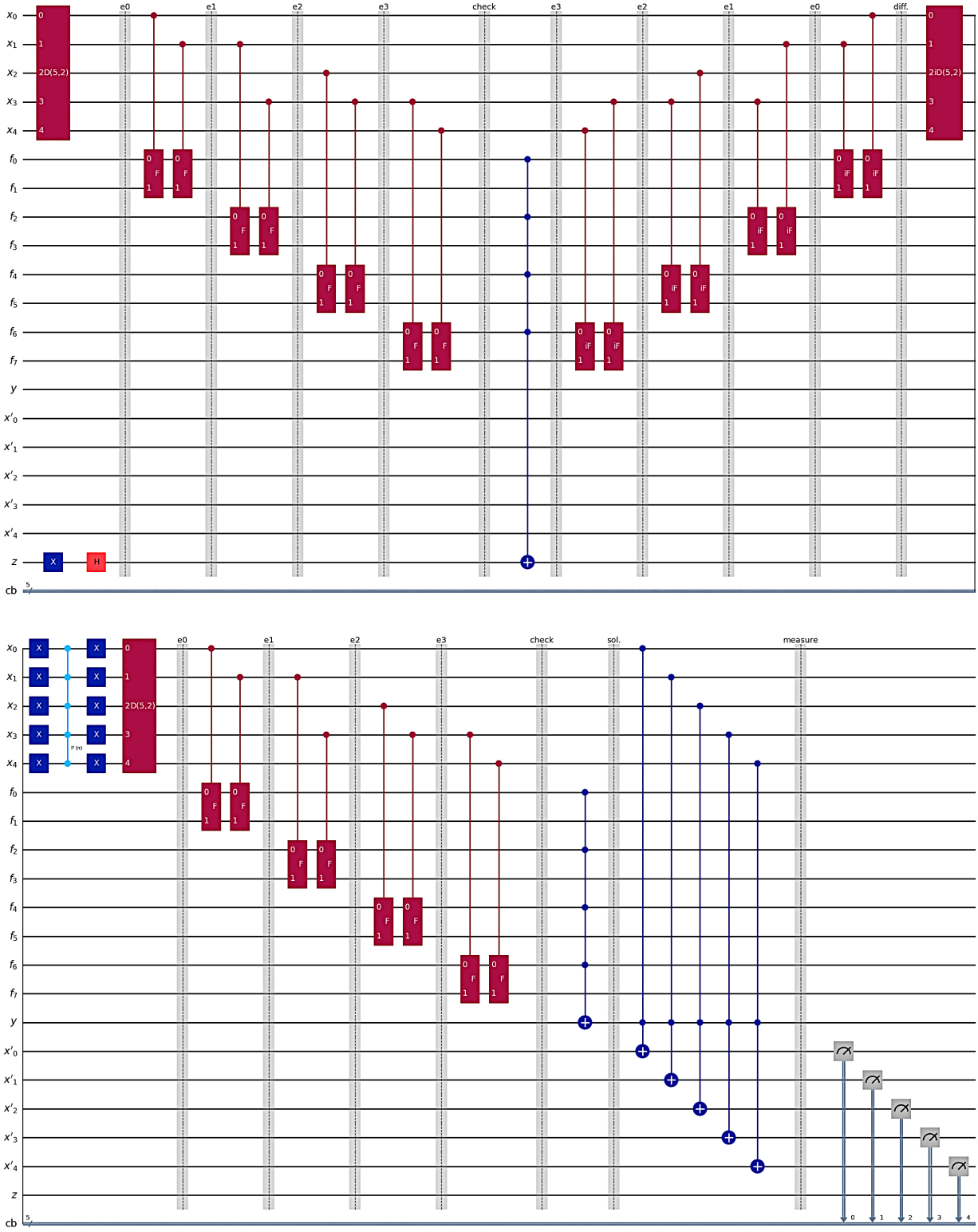
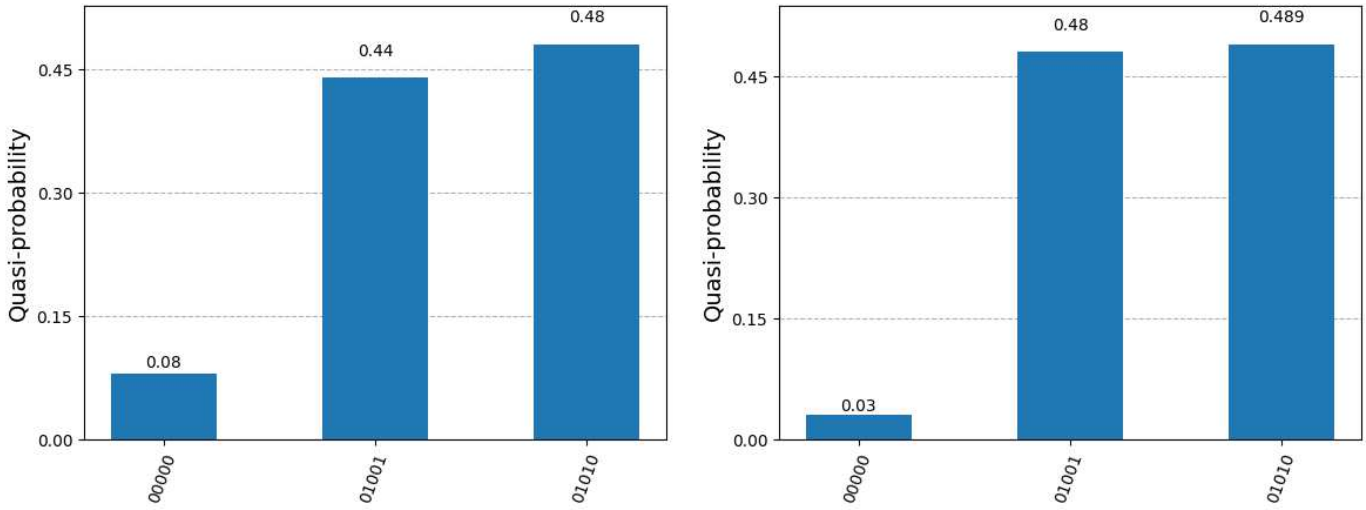


Figure A1. Quantum circuit of combining DSQS with the improved Grover's algorithm using Dicke states to find all solutions to the k-VCP in the preliminary experiment.



(**a**) $n^k = 5^2 = 25$ shots (**b**) 5000 shots

Figure A2. Histogram of the measurement results of the method that combining DSQS and improved Grover's algorithm using Dicke states to find all solutions to the k-VCP for $k = 2$ in the preliminary experiment.

References

1. Jiang, J.R.; Chu, C.W. Classifying and benchmarking quantum annealing algorithms based on quadratic unconstrained binary optimization for solving NP-hard problems. *IEEE Access* **2023**, *11*, 104165–104178. [CrossRef]
2. Jiang, J.R.; Shu, Y.C.; Lin, Q.Y. Benchmarks and Recommendations for Quantum, Digital, and GPU Annealers in Combinatorial Optimization. *IEEE Access* **2024**, *12*, 125014–125031. [CrossRef]
3. Arute, F.; Arya, K.; Babbush, R.; Bacon, D.; Bardin, J.C.; Barends, R.; Biswas, R.; Boixo, S.; Brandao, F.G.; Buell, D.A.; et al. Quantum supremacy using a programmable superconducting processor. *Nature* **2019**, *574*, 505–510. [CrossRef] [PubMed]
4. Deutsch, D.; Jozsa, R. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. Ser. A Math. Phys. Sci.* **1992**, *439*, 553–558.
5. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the Proceedings 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
6. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.
7. Boyer, M.; Brassard, G.; Høyer, P.; Tapp, A. Tight bounds on quantum searching. *Fortschritte Phys. Prog. Phys.* **1998**, *46*, 493–505. [CrossRef]
8. Bennett, C.H.; Bernstein, E.; Brassard, G.; Vazirani, U. Strengths and weaknesses of quantum computing. *SIAM J. Comput.* **1997**, *26*, 1510–1523. [CrossRef]
9. Karp, R.M. Reducibility among Combinatorial Problems. In Proceedings of the Complexity of Computer Computations: Proceedings of a Symposium on the Complexity of Computer Computations, New York, NY, USA, 20–22 March 1972; pp. 85–103.
10. Cook, S.A. The complexity of theorem-proving procedures. In Proceedings of the Third Annual ACM Symposium on Theory of Computing, Shaker Heights, OH, USA, 3–5 May 1971; pp. 151–158.
11. Fadillah, M.H.A.Z.; Idrus, B.; Hasan, M.K. Introduction to IBM Quantum & Qiskit. 2021. Available online: https://ftsm.ukm.my/v5/file/research/technicalreport/LP-CAIT-FTSM-2021-010.pdf (accessed on 9 September 2025).
12. Jiang, J.R.; Wang, Y.J. Using a Simplified Quantum Counter to Implement Quantum Circuits Based on Grover's Algorithm to Tackle the Exact Cover Problem. *Mathematics* **2024**, *13*, 90. [CrossRef]
13. Brassard, G.; Høyer, P.; Tapp, A. Quantum counting. In Proceedings of the Automata, Languages and Programming: 25th International Colloquium, ICALP'98, Aalborg, Denmark, 13–17 July 1998; Proceedings 25; Springer: Berlin/Heidelberg, Germany, 1998; pp. 820–831.
14. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2010.
15. Saha, A.; Saha, D.; Chakrabarti, A. Circuit design for k-coloring problem and its implementation on near-term quantum devices. In Proceedings of the 2020 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS), Chennai, India, 14–16 December 2020; pp. 17–22.

16. Saha, A.; Saha, D.; Chakrabarti, A. Circuit design for *k*-coloring problem and its implementation in any dimensional quantum system. *SN Comput. Sci.* **2021**, *2*, 427. [CrossRef]
17. Haverly, A.; López, S. Implementation of Grover's algorithm to solve the maximum clique problem. In Proceedings of the 2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Tampa, FL, USA, 7–9 July 2021; pp. 441–446.
18. Mukherjee, S. A Grover search-based algorithm for the list coloring problem. *IEEE Trans. Quantum Eng.* **2022**, *3*, 3101008. [CrossRef]
19. Roch, C.; Castillo, S.L.; Linnhoff-Popien, C. A Grover based quantum algorithm for finding pure Nash equilibria in graphical games. In Proceedings of the 2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C), Honolulu, HI, USA, 12–15 March 2022; pp. 147–151.
20. Jiang, J.R. Quantum circuit based on Grover algorithm to solve Hamiltonian cycle problem. In Proceedings of the 2022 IEEE 4th Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 28–30 October 2022; pp. 364–367.
21. Jiang, J.R.; Kao, T.H. Solving Hamiltonian Cycle Problem with Grover's Algorithm Using Novel Quantum Circuit Designs. In Proceedings of the 2023 IEEE 5th Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 27–29 October 2023; pp. 796–801.
22. Jiang, J.R.; Yan, W.H. Novel Quantum Circuit Designs for the Oracle of Grover's Algorithm to Solve the Vertex Cover Problem. In Proceedings of the 2023 IEEE 5th Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 27–29 October 2023; pp. 652–657.
23. Jiang, J.R.; Wang, Y.J. Quantum circuit based on Grover's algorithm to solve exact cover problem. In Proceedings of the 2023 VTS Asia Pacific Wireless Communications Symposium (APWCS), Tainan, Taiwan, 23–25 August 2023; pp. 1–5.
24. Jiang, J.R.; Lin, Q.Y. Utilizing Novel Quantum Counters for Grover's Algorithm to Solve the Dominating Set Problem. *arXiv* **2023**, arXiv:2312.09388.
25. Dicke, R.H. Coherence in spontaneous radiation processes. *Phys. Rev.* **1954**, *93*, 99. [CrossRef]
26. Bärtschi, A.; Eidenbenz, S. Deterministic preparation of Dicke states. In Proceedings of the International Symposium on Fundamentals of Computation Theory, Copenhagen, Denmark, 12–14 August 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 126–139.
27. Knuth, D.E. *The Art of Computer Programming: Fundamental Algorithms*; Addison-Wesley Professional: Boston, MA, USA, 1997; Volume 1.
28. Bernstein, E.; Vazirani, U. Quantum complexity theory. In Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, 16–18 May 1993; pp. 11–20.
29. Deutsch, D. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A Math. Phys. Sci.* **1985**, *400*, 97–117.
30. Turing, A.M. On computable numbers, with an application to the Entscheidungs problem. *J. Math* **1936**, *58*, 5.
31. Gill III, J.T. Computational complexity of probabilistic Turing machines. In Proceedings of the Sixth Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, 30 April–2 May 1974; pp. 91–95.
32. He, Y.; Luo, M.X.; Zhang, E.; Wang, H.K.; Wang, X.F. Decompositions of n-qubit Toffoli gates with linear circuit complexity. *Int. J. Theor. Phys.* **2017**, *56*, 2350–2361. [CrossRef]
33. Barenco, A.; Bennett, C.H.; Cleve, R.; DiVincenzo, D.P.; Margolus, N.; Shor, P.; Sleator, T.; Smolin, J.A.; Weinfurter, H. Elementary gates for quantum computation. *Phys. Rev. A* **1995**, *52*, 3457. [CrossRef] [PubMed]
34. Jiang, J.R.; Lin, Q.Y. Improving Grover's Algorithm with Dicke States. In Proceedings of the IEEE International Conference on Consumer Electronics—Taiwan (ICCE-TW), Kaohsiung, Taiwan, 16–18 July 2025.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.