Quantum Algorithms and Mathematical Representation of Bio-molecular Solutions for the Hitting-set Problem on a Quantum Computer

Weng-Long Chang

1Contact Author: Department of Computer Science and Information Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung City, Taiwan 807-78, Republic of China
E-mail: changwl@cc.kuas.edu.tw

Ting-Ting Ren2, Mang Feng3 and Jun Luo4

2, 3, 4State Key Laboratory of Magnetic Resonance and Atomic and Molecular Physics, Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan, 430071, People’s Republic of China
E-mail: ttren@wipm.ac.cn, mangfeng@wipm.ac.cn, jluo@wipm.ac.cn

Kawuu Weicheng Lin5

5Department of Computer Science and Information Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung City, Taiwan 807-78, Republic of China
E-mail: linwe@cc.kuas.edu.tw

Minyi Guo6

6Department of Computer Software, the University of Aizu, Aizu-Wakamatsu City, Fukushima 965-8580, Japan
E-mail: minyi@u-aizu.ac.jp

Lai Chin Lu7

7Department of Industrial and Management Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung City, Taiwan 807-78, Republic of China
E-mail: rachel@cc.kuas.edu.tw

Abstract—Assume that a finite set $S$ is $\{u_1, \ldots, u_n\}$ and a collection $C$ is the set of subsets $\{C_1, \ldots, C_m\}$ to the finite set $S$. Mathematically, a hitting set is to find whether there is a subset $S_1 \subseteq S$ such that $S_1$ contains at least one element from each subset in $C$. The hitting-set problem means finding a minimum-sized hitting-set for $S$ and $C$ and also is an NP-complete problem. In this paper, it is demonstrated that quantum implementation of bio-molecular solutions to compute the number of elements in each hitting-set in an instance of the hitting-set problem could be considered as the oracle work in Grover’s algorithm, i.e., the target state labeling, preceding Grover’s searching steps. From this fact, the answer of an instance of the hitting-set problem could be found by means of the standard steps of Grover’s algorithm. It is also estimated from our way the time and space complexity of solving an instance of the hitting-set problem. Furthermore, also it is demonstrated how for the hitting-set problem, molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space. Finally, for testing our theory, a three-qubit nuclear magnetic resonance (NMR) experiment of solving the simplest hitting-set problem is performed.

Index Terms—Data Structure and Algorithm, Quantum Algorithms, Molecular Algorithms, Nuclear Magnetic Resonance.
I. INTRODUCTION

The molecular computation was first proposed in 1961 by Feynman [1], while this idea had not been tested experimentally until 1994 when Adleman successfully solved an instance of the Hamiltonian path problem in a test tube by DNA strands [2]. On the other hand, Feynman [3] presented one of the most important problems in computation theory in 1982 that is whether quantum mechanically computing devices are able to finish computations faster than the standard Turing machines [4]. Following this idea, Benioff [5] offered something about the possibility of quantum computation, and Deutsch [6] proposed a general model of quantum computation — the quantum Turing machine. From [7], Farhi and his co-authors gave a quantum algorithm for solving instances of the satisfiability problem, based on adiabatic evolution which is a general model of adiabatic quantum computation — the adiabatic quantum Turing machine.

In this paper, it is shown that quantum implementation of bio-molecular solutions in [8] for figuring out the number of elements in each hitting-set in an instance of the hitting-set problem could be considered as the oracle work in Grover’s algorithm, i.e., the target state labeling, preceding Grover’s searching steps. Next, in light of this fact, the answer to an instance of the hitting-set problem could be found by means of the standard steps of Grover’s algorithm. Then, it is also estimated from our way the time and space complexity of solving an instance of the hitting-set problem. Furthermore, also it is demonstrated how for the hitting-set problem, molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space. Finally, for testing our theory, a three-qubit nuclear magnetic resonance (NMR) experiment of solving the simplest hitting-set problem is performed.

The rest of the paper is organized as follows: Section 2 gives motivation for developing quantum algorithm for the hitting-set problem with an *n*-element set and *an m*-subset collection. In Section 3 we will briefly review the development of molecular computing and quantum computing. We present our quantum algorithm of bio-molecular solutions [8] for solving an instance of the hitting-set problem with an *n*-element set and *an m*-subset collection in Section 4. In Section 5, the time complexity and the space complexity of our quantum algorithm are analyzed for solving an instance of the hitting-set problem with an *n*-element set and *an m*-subset collection. Section 6 is for a nuclear magnetic resonance (NMR) experiment to check our theory, in which we report an NMR experiment for the simplest example of the hitting-set problem. In Section 7, it is shown how for the hitting-set problem, molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space. We conclude with a brief discussion in Section 8.

II. MOTIVATION

Because of the publication of Deutsch’s [6] and Adleman’s [2] seminal articles, various quantum algorithms and DNA-based algorithms have been, respectively, presented to solve many computational problems [9, 10, 11]. In the field of molecular computing, one of the main open questions is to ask what mathematical forms of bio-molecular
solutions for any NP-Complete Problem are. Our motivation of writing the article is to try to find and demonstrate whether bio-molecular solutions for any NP-Complete Problem can be represented in term of a unit vector in a finite-dimensional Hilbert space.

III. BRIEF REVIEW OF MOLECULAR COMPUTING AND QUANTUM COMPUTING

Adleman and his co-authors [12] performed experiments to solve a 20-variable 24-clause three-conjunctive normal form (3-CNF) formula. Zhang and Winfree [13] presented an allosteric DNA molecule that, in its active configuration, catalyzes a noncovalent DNA reaction. Yin and his co-authors [14] programmed diverse molecular self-assembly and disassembly pathways using a ‘reaction graph’ abstraction to specify complementarity relationships between modular domains in a versatile DNA hairpin motif. Cook and his co-authors [15] showed how several common digital circuits (including de-multiplexers, random access memory, and Walsh transforms) could be built in a bottom-up manner using biologically inspired self-assembly.

Strengths and weaknesses of the quantum Turing machine for solving NP-complete problems were discussed [16]. So far, the most frequently cited quantum algorithms are Shor’s algorithms for solving factoring integers and discrete logarithm [17] and Grover’s search algorithm [18] for unsorted databases. Yang and his co-authors [19] simulated Grover’s searching process under the influence of the cavity decay and showed that their scheme could be achieved efficiently to find a marked state with high reliability. Peng and his co-authors [20] proposed an adiabatic quantum algorithm capable of factorizing numbers, using fewer qubits than Shor’s algorithm and implemented the algorithm in a NMR quantum information processor and experimentally factorize the number 21.

IV. QUANTUM ALGORITHMS OF BIO-MOLECULAR OF THE HITTING-SET PROBLEM

In this section, we will first introduce the definition of the hitting-set problem and the DNA-based algorithm for solving an instance of the hitting-set problem [8]. Then based on bio-molecular solutions for solving an instance of the hitting-set problem [8], its corresponding quantum algorithm is presented.

A. DEFINITION OF THE HITTING-SET PROBLEM

Assume that a finite set \( S \) is \( \{u_0, ..., u_i\} \), where \( u_i \) is the \( i \)th element in \( S \) with \( 1 \leq i \leq n \). \( |S| \) is denoted as the number of elements in \( S \) and \( |S| \) is equal to \( n \). Suppose that a collection \( C \) is a set of subsets of the finite set \( S \), denoted by \( \{C_1, ..., C_m\} \), where \( C_j \) is the \( j \)th element in \( C \) with \( 1 \leq j \leq m \). \( |C| \) is denoted as the number of subsets in \( C \) and \( |C| \) is equal to \( m \). Mathematically, a hitting-set is to find whether there is a subset \( S_i \subseteq S \) such that \( S_i \) contains at least one element from each subset in \( C \). **Definition 4-1** cited in [9] is applied to denote the hitting-set problem, which is an NP-complete problem [9].

**Definition 4-1:** The hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-element collection \( C \) of subsets for \( S \) means finding a minimum-sized hitting-set.
As an example, we consider a simple case in Figure 4-1, where the finite set $S$ is $\{2, 1\}$ and the collection $C$ is $\{\{1\}\}$. The two sets define a hitting-set problem. In Figure 4-1, there are two hitting sets that are, respectively, $\{1\}$ and $\{1, 2\}$. From **Definition 4-1**, the answer (a minimum-sized hitting-set) of the hitting-set problem for $S$ and $C$ in Figure 4-1 is $\{1\}$.

$$ S = \{2, 1\} \text{ and } C = \{\{1\}\} $$

Figure 4-1: A finite set $S$ and a collection $C$ of subsets for $S$.

B. ALL OF THE POSSIBLE SOLUTIONS TO THE HITTING-SET PROBLEM

From **Definition 4-1**, we know that all of the possible solutions to the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ contain $2^n$ possible choices. Each possible choice corresponds to a subset in $S$. Therefore, we assume that $U$ is a set of $2^n$ possible choices and equal to $\{u_n u_{n-1} \ldots u_1 u_0\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$, which implies that the length of each element in $U$ is $n$ bits and each element represents one of $2^n$ subsets for an $n$-element finite set $S$. For the purpose of presentation, we suppose that $u_k^0$ is used to denote the value of $u_k$ to be zero and $u_k^1$ means value of $u_k$ to be one. If the value of $u_k$ is equal to one in a subset, then the $k$th element in $S$ appears in the subset. Otherwise, the $k$th element in $S$ does not appear in the subset. **Definition 4-2** below will show how each element in $U$ is represented to be a unique computational state vector with $2^n$-tuples of binary numbers.

**Definition 4-2**: The $j$th element in $U$ can be represented as a unique computational state vector $\begin{bmatrix} u_1^j \end{bmatrix} =$

$$\begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,2^n} \\
\vdots & \vdots & \ddots & \vdots \\
u_{2^n,1} & u_{2^n,2} & \cdots & u_{2^n,2^n} \end{bmatrix}_{2^n \times 1}$$

$u_{1,1}^{j} = 1$, $u_{j,1} = 1$, $\forall h, 1 \leq h \neq j \leq 2^n$.

The corresponding computational state vector for the first element, $u_n^0 u_{n-1}^0 \ldots u_2^0 u_1^0$, in $U$ is $\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}_{1 \times 2^n}^T$, the corresponding computational state vector for the second element, $u_n^0 u_{n-1}^0 \ldots u_2^0 u_1^1$, in $U$ is $\begin{bmatrix} 0 & 1 & \cdots & 0 \end{bmatrix}_{1 \times 2^n}^T$, and the corresponding computational state vector for the last element, $u_n^1 u_{n-1}^1 \ldots u_2^1 u_1^1$, in $U$ is $\begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}_{1 \times 2^n}^T$. For the sake of presentation, we assume that $B$ is the set of the corresponding computational state vectors to the elements in $U$ and $B = \{ \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}_{1 \times 2^n}^T \}$.
Because each element in $B$ is a coordinated vector, we span $B = C^{2^n}$ [21, 22], where $C^{2^n}$ is a Hilbert space. This implies that the set $B$ consists of orthonormal bases in a Hilbert space.

C. SOLUTION SPACE OF BIO-MOLECULES FOR THE HITTING-SET PROBLEM

The following eight bio-molecular operations cited from [23] are briefly introduced. For solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-element collection $C$ of subsets for $S$, the following DNA-based algorithm cited from [8] in terms of the eight bio-molecular operations is described below.

**Definition 4-3:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$ and a Boolean variable $u_j$, the bio-molecular operation, “Append-Head”, appends $u_j$ onto the head of every element in the set $U$. The formal representation is written as $\text{Append-Head}(U, u_j) = \{u_{n-1} u_n \ldots u_2 u_1 u_j\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$ and $u_j \in \{0, 1\}$.

**Definition 4-4:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$ and a Boolean variable $u_j$, the bio-molecular operation, “Append-Tail”, appends $u_j$ onto the end of every element in the set $U$. The formal representation is written as $\text{Append-Tail}(U, u_j) = \{u_{n-1} u_n \ldots u_2 u_1 u_j\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$ and $u_j \in \{0, 1\}$.

**Definition 4-5:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$, the bio-molecular operation, “Discard($U$)” sets $U$ to be an empty set.

**Definition 4-6:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$, the bio-molecular operation “Amplify($U$, $U_m$)” creates a number of identical copies, $U_m$, of the set $U$, and then discard($U$).

**Definition 4-7:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$ and a Boolean variable, $u_j$, if the value of $u_j$ is equal to one, then the bio-molecular extract operation creates two new sets, $+(U, u_j) = \{u_{n-1} u_n \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \neq j \leq n$ and $-(U, u_j) = \{u_{n-1} u_n \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \neq j \leq n$. Otherwise, it produces another two new sets, $+(U, u_j^0) = \{u_{n-1} u_n \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \neq j \leq n$ and $-(U, u_j^0) = \{u_{n-1} u_n \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \neq j \leq n$.

**Definition 4-8:** Given $m$ sets $U_1 \ldots U_m$, the bio-molecular merge operation, $\cup(U_1, \ldots, U_m) = U_1 \cup \ldots \cup U_m$.

**Definition 4-9:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$, the bio-molecular operation “Detect($U$)” returns true if $U \neq \emptyset$. Otherwise, it returns false.

**Definition 4-10:** Given a set $U = \{u_n u_{n-1} \ldots u_2 u_1\} \forall u_k \in \{0, 1\}$ for $1 \leq k \leq n$, the bio-molecular operation
“Read(\(U\))” reads an arbitrary element in \(U\). Even if \(U\) contains many different elements, the bio-molecular operation can give an explicit description of exactly one of them.

**Algorithm 4-1**: Solving an instance of the hitting-set problem with an \(n\)-element finite set \(S\) and an \(m\)-element collection \(C\) of subsets for \(S\).

(0a) Append-Head\((T_1, u_1^1)\).
(0b) Append-Head\((T_2, u_1^0)\).
(0c) \(T_0 = \cup(T_1, T_2)\).

(1) **For** \(k = 2\ to \ n\)
   (1a) Amplify\((T_0, T_1, T_2)\).
   (1b) Append-Head\((T_1, u_k^1)\).
   (1c) Append-Head\((T_2, u_k^0)\).
   (1d) \(T_0 = \cup(T_1, T_2)\).

EndFor

(2) **For** \(j = 1\ to \ |C|\)

(3) **For** \(i = 1\ to \ |C_j|\)
   (3a) **If** (the \(i^{th}\) element in the \(j^{th}\) subset in \(C\) is the \(k^{th}\) element in \(S\)) **then**
   (3b) \(T_i = +(T_0, u_k^1)\) and \(T_0 = -(T_0, u_k^1)\).
   **EndIf**

EndFor

(4) Discard\((T_0)\).

(5) **For** \(i = 1\ to \ |C_j|\)
   (5a) \(T_0 = \cup(T_0, T_i)\).

EndFor

EndFor

(6) **For** \(i = 0\ to \ n - 1\)

(7) **For** \(j = i\ down\ to\ 0\)
   (7a) \(T_{j+1}^{\text{ON}} = +(T_j, u_{j+1}^1)\) and \(T_j = -(T_j, u_{j+1}^1)\).
   (7b) \(T_{j+1} = \cup(T_{j+1}, T_{j+1}^{\text{ON}})\).

EndFor

EndFor

(8) **For** \(k = 1\ to \ n\)
   (8a) **If** (Detect\((T_k)\)) **then**
   (8b) Read\((T_k)\) and terminate the algorithm.
   **EndIf**

EndFor
A finite set \( S = \{2, 1\} \) and a collection \( C = \{\{1\}, \{2\}\} \) in Figure 4-1 include two elements and two subsets. Thus, the values for \( n \) and \( m \) are both two. On the first execution of Steps (0a) and (0b), it generates \( T_1 = \{u_1^1 \} \) and \( T_2 = \{u_1^0 \} \). Next, the first execution of Step (0c) results in \( T_0 = \{u_1^1, u_1^0\} \), \( T_1 = \emptyset \), and \( T_2 = \emptyset \). The first execution of Step (1a) produces \( T_1 = \{u_1^1, u_1^0\} \), \( T_2 = \{u_1^1, u_1^0\} \) and \( T_0 = \emptyset \). Then, the first execution of Steps (1b) and (1c) obtains \( T_1 = \{u_2^1 u_1^1, u_2^1 u_1^0\} \) and \( T_2 = \{u_2^0 u_1^1, u_2^0 u_1^0\} \). The first execution of Step (1d) results in \( T_0 = \{u_2^1 u_1^1, u_2^0 u_1^1, u_2^1 u_1^0, u_2^0 u_1^0\} \), \( T_1 = \emptyset \), and \( T_2 = \emptyset \). Therefore, after each operation in Step 1 in Algorithm 4-1 is performed, \( T_0 \) contains \( 2^2 \) possible choices (consisting of legal and illegal hitting sets).

The first and second subsets in \( C \) in Figure 4-1 are \( \{1\} \) and \( \{2\} \), respectively. Hence, based on the first subset \( \{1\} \) in \( C \), the first execution of Step (3b) generates \( T_1 = \{u_2^1 u_1^1, u_2^0 u_1^1\} \) including hitting sets, \( \{2, 1\} \) and \( \{1\} \), and \( T_0 = \{u_2^1 u_1^0, u_2^0 u_1^0\} \) including illegal hitting sets, \( \{2\} \) and \( \emptyset \). Because each DNA strand in \( T_0 \) does not satisfy the definition of a hitting set, next, the first execution of Step (4) obtains \( T_0 = \emptyset \). Since each DNA strand in \( T_1 \) at least includes the first element, 1, the first execution of Step (5a) then results in \( T_0 = \{u_2^1 u_1^1, u_2^0 u_1^1\} \) and \( T_1 = \emptyset \). Based on the second subset \( \{2\} \) in \( C \), the second execution of Step (3b) obtains \( T_1 = \{u_2^1 u_1^1\} \) including hitting sets, \( \{2, 1\} \) and \( \{1\} \), and \( T_0 = \{u_2^0 u_1^1\} \) including illegal hitting sets, \( \{2\} \). Because the only DNA strand in \( T_0 \) does not satisfy the definition of a hitting set, the second execution of Step (4) then results in \( T_0 = \emptyset \). Since the only DNA strand in \( T_1 \) satisfies the definition of a hitting set, the second execution of Step (5a) obtains \( T_0 = \{u_2^1 u_1^1\} \) and \( T_1 = \emptyset \).

Because the value of \( n \) is equal to two, Steps (7a) and (7b) will be executed three times. The first execution of Step (7a) results in \( T_1^{ON} = \{u_2^1 u_1^1\} \) and \( T_0 = \emptyset \). The first execution of Step (7b) then produces \( T_1 = \{u_2^1 u_1^1\} \) and \( T_1^{ON} = \emptyset \). The second execution of Step (7a) generates \( T_2^{ON} = \{u_2^1 u_1^1\} \) and \( T_1 = \emptyset \). Next, the second execution of Step (7b) results in \( T_2 = \{u_2^1 u_1^1\} \) and \( T_2^{ON} = \emptyset \). The third execution of Step (7a) results in \( T_1^{ON} = \emptyset \) and \( T_0 = \emptyset \). The third execution of Step (7b) then obtains \( T_1 = \emptyset \) and \( T_1^{ON} = \emptyset \). Next, in the first execution and the second execution of Step (8a), a false and a true are returned. Therefore, after the first execution of Step (8b) is finished, the answer of the hitting-set problem for \( S \) and \( C \) in Figure 4-1 is \( \{2, 1\} \).

Lemma 4-1: Using the steps in Algorithm 4-1, an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-element collection \( C \) of subsets for \( S \) can be solved on a molecular computer.

D. COMPUTATIONAL SPACE OF QUANTUM MECHANICAL SOLUTION FOR THE HITTING-SET PROBLEM

A qubit (quantum bit) has two ‘computational basis vectors’ \( |0\rangle \) and \( |1\rangle \) of the two-dimensional Hilbert space corresponding to the classical bit values 0 and 1 [21, 22], and an arbitrary state \( |\psi\rangle \) of a qubit is a linearly
weighted combination of the computational basis vectors

\[
|\varphi\rangle = l_1 |0\rangle + l_2 |1\rangle = l_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix}_{2\times 1} + l_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}_{2\times 1}, \quad (4.1)
\]

where the weighted factors \( l_1 \) and \( l_2 \in \mathbb{C} \) are the so-called probability amplitudes, with \(| l_1 |^2 + | l_2 |^2 = 1 \). A collection of \( n \) qubits is called a qregister (quantum register) of size \( n \). It may include any of the \( 2^n \)-dimensional computational basis vectors, \( n \) qubits of size, or arbitrary superposition of these vectors [21, 22]. If the content of quantum bits in a quantum register is known, then the state of the quantum register can be computed by means of a tensor product in following way \(| \varphi \rangle = |\theta_1\rangle \otimes |\theta_2\rangle \otimes \ldots \otimes |\theta_n\rangle \otimes |\theta_{n+1}\rangle\), where \( \theta_k \in \{0, 1\} \) for \( 1 \leq k \leq n \). From [21, 22], the Hadamard gate \( H \) is a quantum gate of one quantum bit (a \( 2 \times 2 \) matrix), and \( H_{1,1} = \frac{1}{\sqrt{2}}, \ H_{1,2} = \frac{1}{\sqrt{2}}, \ H_{2,1} = \frac{1}{\sqrt{2}}, \ \text{and} \ H_{2,2} = -\frac{1}{\sqrt{2}} \). For a general input (4.1), it produces

\[
|\Phi\rangle = H |\varphi\rangle = \frac{l_1 + l_2}{\sqrt{2}} |0\rangle + \frac{l_1 - l_2}{\sqrt{2}} |1\rangle, \quad (4.2)
\]

From (4.2), \( H |0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \), and \( H |1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \). The generalization of (4.2) to \( n \)-qubit registers by an \( n \)-qubit Hadamard gate is for \( 2^n \) possible choices for the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \). For an all-zero input \(|\Phi\rangle = |00 \cdots 0\rangle\), the outcome is

\[
|\varphi\rangle = H^{\otimes n} |\Phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} i, \quad (4.3)
\]

where \( H^{\otimes n} \) stands for the \( n \)-qubit Hadamard gate. Equation (4.3) also gives computational space for the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).

E. INTRODUCTION OF QUANTUM GATES FOR SOLVING THE HITTING-SET PROBLEM

The time evolution of the states of quantum registers can be modeled by means of unitary operators, which are often referred to as quantum gates [21, 22]. Therefore, a quantum gate can be regarded as an elementary quantum-computing device that performs a fixed unitary operation on selected quantum bits during a fixed period of time. One-qubit and two-qubit quantum gates are elementary quantum gates. The NOT gate is a one-qubit gate and
sets only the (target) bit to its negation. The CNOT (*controlled*-NOT) gate is a two-qubit gate and flips the second qubit (the target qubit) if and only if the first qubit (the control qubit) is equal to one. The *controlled*-controlled-NOT (CCNOT) gate is a three-qubit gate and flips the third qubit (the target qubit) if and only if the first and second qubits (the two control qubits) are both one. Graphical representations for NOT, CNOT, and CCNOT are presented in Figures 4-2, 4-3, and 4-4, respectively. The control quantum bits are graphically represented by a dot, while the target quantum bits are graphically represented by a cross. The operation OR, $\lor$, is the “logical or” operation and the operation AND, $\land$, is the “logical and” operation. Therefore, $u_2 \lor u_1$ is 0 only if both $u_2$ and $u_1$ are 0; $u_2 \land u_1$ is 1 only if both $u_2$ and $u_1$ are 1. The operations OR and AND are implemented by quantum circuits in Figures 4-5 and 4-6, respectively.

F. INTRODUCTION OF QUANTUM NETWORKS FOR FINDING HITTING SETS

From definition of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ above, we know that it contains $2^n$ possible choices. If any one of $2^n$ possible choices at least contains one element from each subset in $C$, then it is a hitting set. Otherwise, it is not an answer. Next, among the legal choices, a hitting set with the minimum size of elements needs to be found. Therefore, it is assumed that all of the elements in the $j$th subset in $C$ for $1 \leq j \leq m$, are, subsequently, $u_n$, $u_{n-1}$, ..., $u_2$, $u_1$. Because a hitting set at least includes one element from the the
$j$th subset in $C$ for $1 \leq j \leq m$, this implies that the requested condition from the $j$th subset in $C$ for $1 \leq j \leq m$ can be regarded as finding choices from $2^n$ possible choices that satisfy each formula of the form, $u_n \lor u_{n-1} \ldots \lor u_2 \lor u_1$, i.e., the true value.

It is assumed that a quantum register of $n$ quantum bits, $|u_n \cdots u_1\rangle$, is applied to represent $2^n$ possible choices by means of $n$ Hadamard gates operating them. It is supposed that each quantum register among $m$ quantum registers has $(n + 1)$ quantum bits, and the $k^{th}$ quantum register $\{r_{k,n} \ r_{k,n-1} \cdots r_{k,1} \ r_{k,0}\}$ for $1 \leq k \leq m$ is used to store the results of evaluating the $k^{th}$ clause with the form $u_n \lor u_{n-1} \ldots \lor u_2 \lor u_1$. Because the OR operation is implemented by quantum circuits in Figures 4-5, for $1 \leq k \leq m$ and $1 \leq j \leq n$ the initial states for $|r_{k,0}\rangle$ and $|r_{k,j}\rangle$ are, respectively, prepared in state $|0\rangle$ and in state $|1\rangle$. Therefore, its evaluating computation for the $k^{th}$ clause with the form $u_n \lor u_{n-1} \ldots \lor u_2 \lor u_1$ is equal to

$$\left(\bigotimes_{q=n}^1 |r_{k,q}^1\rangle \right) \otimes \left(|r_{k,0}^0\rangle \right) \otimes \left(\bigotimes_{q=n}^1 |u_q\rangle \right) \rightarrow \left(\bigotimes_{q=n}^2 (r_{k,q}^1 \oplus \bar{u}_q \cdot \bar{r}_{k,q-1}^1)\right) \otimes \left(|r_{k,1}^1 \oplus \bar{u}_1 \cdot \bar{r}_{k,0}^0\rangle\right) \otimes \left(|r_{k,0}^0\rangle \right) \otimes \left(\bigotimes_{q=n}^1 |u_q\rangle \right)$$

where $\oplus$ denotes the AND operation of two Boolean variables $\{\bar{u}_q, \bar{r}_{k,q-1}\}$ for $1 \leq q \leq n$. The $(n + 1)^{th}$ quantum bit, $|r_{k,n}\rangle$, in $|r_{k,n} \ r_{k,n-1} \cdots r_{k,1} \ r_{k,0}\rangle$ is employed to store the final result of the evaluating computation for the $k^{th}$ clause with the form $u_n \lor u_{n-1} \ldots \lor u_2 \lor u_1$.

Then, in order to evaluate the AND operation of the current clause (the $k^{th}$ clause) and the previous clause (the $(k-1)^{th}$ clause) for $1 \leq k \leq m$, an additional quantum register $|c_m \ c_{m-1} \cdots c_1 \ c_0\rangle$ is needed. Because the AND operation is implemented by quantum circuits in Figures 4-6, the first quantum bit, $|c_0\rangle$, is initially prepared in state $|1\rangle$, and other $m$ quantum bits, $|c_m \ c_{m-1} \cdots c_1\rangle$, are initially in state $|0\rangle$. Hence, the $(k + 1)^{th}$ quantum bit, $|c_k\rangle$, in $|c_m \ c_{m-1} \cdots c_1 \ c_0\rangle$ is applied to store the result of evaluating computation for the current clause (the $k^{th}$ clause) and the previous clause (the $(k-1)^{th}$ clause). This is to say that its evaluating computation is equal to

$$\left(\bigotimes_{p=m}^{k+1} |c_p^0\rangle \right) \otimes \left(|c_k^0\rangle \right) \otimes \left(\bigotimes_{p=k-1}^1 |c_p\rangle \right) \otimes \left(|c_0^1\rangle \right) \otimes \left(\bigotimes_{q=n}^1 |r_{k,q}\rangle \right) \otimes \left(|r_{k,0}^0\rangle \right) \otimes \left(\bigotimes_{q=n}^1 |u_q\rangle \right) \rightarrow \left(\bigotimes_{p=m}^{k+1} |c_p^0\rangle \right) \otimes \left(|c_k^0 \oplus c_{k-1} \cdot r_{k,n}\rangle \right) \otimes \left(\bigotimes_{p=k-1}^1 |c_p\rangle \right) \otimes \left(|c_0^1\rangle \right) \otimes \left(\bigotimes_{q=n}^1 |r_{k,q}\rangle \right) \otimes \left(|r_{k,0}^0\rangle \right) \otimes$$
\[
\mathbf{\otimes}^1_{q=n} |u_q\rangle,
\]

(4.5)

where \( \bullet \) denotes the AND operation of two Boolean variables \( \{ c_{k-1}, r_{k,n} \} \) for \( 1 \leq k \leq m \), and for \( 1 \leq p \leq k - 1 \),

\[
c_p = c_p^0 \oplus c_{p-1} \bullet r_{p,n}.
\]

Therefore, the \((m + 1)\)th quantum bit, \( |c_m \oplus c_{m-1} \cdots c_1 c_0\rangle \), is applied to store the final result of evaluating computation for all of the clauses. **Lemma 4-2** is applied to show how Equations (4.4) and (4.5) perform the parallel logic computation completed by Steps (3a) through (5a) in **Algorithm 4-1** and judge which among the \(2^n\) choices are legal hitting sets and which are not answers.

**Lemma 4-2**: To solve the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-element collection \( C \) of subsets for \( S \), Equations (4.4) and (4.5) can be applied to complete the parallel logic computation finished by Steps (3a) through (5a) in **Algorithm 4-1** and to judge which among the \(2^n\) choices are legal hitting sets and which are not answers.

**Proof**:

A mathematical induction is employed to perform the proof. Step (2) in **Algorithm 4-1** is a nested loop, and the value of their loop index variables \( j \) and \( i \) are, respectively, from 1 through \( m \) and from 1 to \( n \). When the value of the loop index variable, \( j \), in Step (2) in **Algorithm 4-1** is equal to one, it is assumed that the first subset \( C_1 \) has at most \( n \) elements where each element comes from an \( n \)-element finite set \( S \). Also it is supposed that bits \( u_1 \) through \( u_n \) are applied to represent the first element through the \( n \)th element in \( C_1 \). The parallel logic computation finished by Steps (3a) through (3b) at loop iterations (1, 1) through (1, \( n \)) in Step (2) in **Algorithm 4-1** is to select those subsets at least containing one element in \( C_1 \) and those subsets not containing any element in \( C_1 \). This implies that the right choices satisfy the formula of the form, \( u_1 \vee u_{n-1} \cdots \vee u_2 \vee u_1 \), i.e., the \(\text{true}\) value. Therefore, from (4.4), the corresponding evaluating computation is equal to: 

\[
\left| r_{1,n}^1 \oplus (\bar{r}_{1,n-1} \bullet \bar{r}_{1,1} \cdots \bar{r}_{1,1} \bullet (\bar{r}_{0,0} \oplus \bar{r}_{1,1} \bullet \bar{r}_{0,0})) \right| \left| r_{1,0}^0 \right| \left| u_n \cdots u_1 \right|
\]

The parallel logic computation completed by Step (4) at loop iteration (1) in Step (2) in **Algorithm 4-1** is to discard those choices not including any an element in \( C_1 \), and the parallel logic computation completed by Step (5) at loop iterations (1, 1) through (1, \( n \)) in Step (2) and Step (5) in **Algorithm 4-1** is to reserve those choices at least including one element in \( C_1 \). This indicates that the reserved choices satisfy the previous clause and the current (first) clause, i.e., the \(\text{true}\) value. Hence, from (4.5), the corresponding evaluating computation is equal to:

\[
(\mathbf{\otimes}^2_{p=m} |c_p^0\rangle) \otimes \left( |c_1^0\rangle \otimes |c_0^1\rangle \otimes (\mathbf{\otimes}^1_{q=n} |r_q^0\rangle) \right) \otimes
\]

\[
(\mathbf{\otimes}^2_{p=m} |c_p^0\rangle) \otimes \left( |c_1^0\rangle \otimes c_0^1 \bullet r_{0,n} \right) \otimes \left( |c_0^1\rangle \otimes (\mathbf{\otimes}^1_{q=n} |r_q^0\rangle) \otimes (r_{1,0}^0) \right)
\]

\[
(\mathbf{\otimes}^1_{q=n} |u_q\rangle).
\]
When the value of the loop index variable, \( j \), in Step (2) in Algorithm 4-1 is equal to \( a \), it is assumed that the parallel logic computation finished by Steps (3a) through (3b) at loop iterations \((a, 1)\) through \((a, n)\) in Step (2) and Step (3) in Algorithm 4-1 can be performed by Equation (4.4). Similarly, it is also supposed that the parallel logic computation finished by Step (4) at loop iteration \((a, 1)\) in Step (2) in Algorithm 4-1 can be performed by Equation (4.5). It is assumed that the parallel logic computation finished by Step (5a) at loop iterations \((a, 1)\) through \((a, n)\) in Step (2) and Step (5) in Algorithm 4-1 can be performed by Equation (4.5).

When the value of the loop index variable, \( j \), in Step (2) in Algorithm 4-1 is equal to \((a + 1)\), it is assumed that the \((a + 1)\)th subset \( C_{a+1} \) at most has \( n \) elements where each element comes from an \( n \)-element finite set \( S \). Also it is supposed that bits \( u_1 \) through \( u_n \) are applied to represent the first element through the \( n \)th element in \( C_{a+1} \). The parallel logic computation completed by Step (4) at loop iteration \((a + 1)\) in Step (2) in Algorithm 4-1 is to discard those choices not including any an element in \( C_{a+1} \), and the parallel logic computation completed by Step (5) at loop iterations \((a + 1, 1)\) through \((a + 1, n)\) in Step (2) and Step (5) in Algorithm 4-1 is to reserve those choices at least including one element in \( C_{a+1} \). This implies that the reserved choices satisfy the \( a \)th clause and the \((a + 1)\)th clause, i.e., the true value. Hence, from (4.5), the corresponding evaluating computation is equal to:

\[
\begin{align*}
\left| r_{a+1,0}^0 \right| & \left| u_n \ldots u_1 \right| \\
\left| r_{a+1,1}^0 \right| & \left| u_n \ldots u_1 \right|
\end{align*}
\]

The parallel logic computation completed by Step (4) at loop iteration \((a + 1)\) in Step (2) in Algorithm 4-1 is to discard those choices not including any an element in \( C_{a+1} \), and the parallel logic computation completed by Step (5) at loop iterations \((a + 1, 1)\) through \((a + 1, n)\) in Step (2) and Step (5) in Algorithm 4-1 is to reserve those choices at least including one element in \( C_{a+1} \). This implies that the reserved choices satisfy the \( a \)th clause and the \((a + 1)\)th clause, i.e., the true value. Hence, from (4.5), the corresponding evaluating computation is equal to:

\[
\begin{align*}
\left( \bigotimes_{p=m}^{a+2} \left| c_p \right|^{0} \right) \otimes \left( \left| c_{a+1} \right|^{0} \right) \otimes \left( \bigotimes_{p=m}^1 \left| c_p \right| \right) \otimes \left( \left| c_0 \right|^{1} \right) \otimes \left( \bigotimes_{q=n}^{1} \left| r_{a+1,q} \right| \right) \otimes \left( \left| r_{a+1,0} \right|^{0} \right) \otimes \left( \bigotimes_{q=n} \left| u_q \right| \right)
\end{align*}
\]

Therefore, it is inferred that Equations (4.4) and (4.5) can be used to perform the parallel logic computation finished by Steps (3a) through (5a) in Algorithm 4-1 and to judge which among the \( 2^n \) choices are legal hitting sets and which are not answers.

G. CONSTRUCTION OF QUANTUM NETWORKS FOR FINDING HITTING SETS

From Lemma 4-2, it is indicated that Equations (4.4) and (4.5) can be applied to test whether the \( k \)th clause with the form \( u_k \vee u_{k-1} \ldots \vee u_2 \vee u_1 \) and the \((k - 1)\)th clause (its result has been obtained) is true or not. Therefore, Lemma 4-3 is applied to show how quantum evaluating circuits (QEC) of Equations (4.4) and (4.5) are implemented by means of OR quantum gates and AND quantum gates. Next, Lemma 4-4 is applied to show how quantum networks (QN) of selecting legal hitting sets and identify illegal hitting sets among \( 2^n \) possible choices are
Lemma 4-3: Quantum evaluating circuits (QEC) for implementing the function of Equations (4.4) and (4.5) can be implemented by means of OR quantum gates and AND quantum gates, and are drawn in Figure 4-7.

Proof:

It is assumed that $u_q$ is the $q^{th}$ Boolean variable in the $k^{th}$ clause for $1 \leq k \leq m$ and $1 \leq q \leq n$. From Figure 4-5, one OR quantum gate consists of three quantum bits. The first and second quantum bits are both control bits, and the third quantum bit is the target bit. It is assumed that the evaluated result of the previous "∨" (logical or) for $u_{q-1}$ is represented as the first quantum bit in a OR quantum gate, the $q^{th}$ Boolean variable $u_q$ in the $k^{th}$ clause is represented as the second quantum bit in a OR quantum gate, and the evaluated result of the current "∨" (logical or) for $u_q$ is stored as the third quantum bit in a OR quantum gate.

From Figure 4-6, one AND quantum gate also consists of three quantum bits. The first and second quantum bits are both control bits, and the third quantum bit is the target bit. It is assumed that the evaluated result of the $(k-1)^{th}$ clause is represented as the first quantum bit in a AND quantum gate, the evaluated result of the last "∨" (logical or) for $u_n$ is represented as the second quantum bit in a AND quantum gate, and the evaluated result of the $k^{th}$ "∧" (logical and) between the $(k-1)^{th}$ clause and the $k^{th}$ clause is stored as the third quantum bit in a AND quantum gate.

Figure 4–7: The full network is QEC (the abbreviation of quantum evaluating circuits).
Therefore, through the relation \( r_{k,q} \leftarrow r_{k,q}^1 \oplus (\overline{U}_q \cdot \overline{F}_{k,q-1}) \) for \( 1 \leq k \leq m \) and \( 1 \leq q \leq n \), quantum evaluating circuits (QEC) in Figure 4-7 require computing \( n \) OR quantum gates for performing the function of Equation (4.4). Next, through the relation \( (c_k^0 \oplus c_{k-1} \cdot r_{k,n}) \), quantum evaluating circuits (QEC) in Figure 4-7 require computing one AND quantum gate for finishing the function of Equation (4.5). Therefore, it is at once inferred that the function of Equations (4.4) and (4.5) can be implemented by means of OR quantum gates and AND quantum gates and its corresponding quantum evaluating circuits (QEC) are drawn in Figure 4-7.

**Lemma 4-4**: For solving the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-element collection \( C \) of subsets for \( S \), quantum networks (QN) of selecting legal hitting sets and identify illegal hitting sets among \( 2^n \) possible choices can be implemented by means of QEC, and are drawn in Figure 4-8.

**Proof:**

A mathematical induction is used to complete the proof. It is assumed that the variable \( k \) is from 1 through \( m \), and \( C_k \) is used to represent the \( k \)th subset in \( C \) and has the formula of the form, \( u_n \lor u_{n-1} \ldots \lor u_1 \lor u_1 \). From Lemma 4-3, it is indicated that one QEC (quantum evaluating circuits in Figure 4-7) consists of \( (2 \times n + 3) \) input lines and \( (2 \times n + 3) \) output lines. When the value of \( k \) is equal to one, from Lemma 4-3, the first QEC (quantum evaluating circuits in Figure 4-7) to the first subset \( C_1 \) is applied to select the right choices and to identify the wrong choices among \( 2^n \) possible choices. So, \( (\left| c_{1}^0 \right> \otimes (\left| c_{1}^1 \right> \otimes (\otimes_{q=n}^{1} | r_{1,q}^1 \rangle) \otimes (\left| r_{1,0}^0 \right>) \otimes (\otimes_{q=n}^{1} | u_{q} \rangle) \rangle \) are regarded as the input lines of the first QEC (quantum evaluating circuits in Figure 4-7) and \( (\left| c_{0}^0 \oplus c_{0}^1 \cdot r_{1,0} \right> \otimes (\left| c_{0}^1 \right>) \rangle \) \( \otimes (\otimes_{q=n}^{2} (r_{1,q}^1 \oplus \overline{U}_q \cdot \overline{F}_{1,q-1}) \rangle \) \( \otimes \left( \left| r_{1,1}^1 \oplus \overline{U}_1 \cdot \overline{F}_{1,0}^0 \right> \rangle \otimes (\left| r_{1,0}^0 \right>) \otimes (\otimes_{q=n}^{1} | u_{q} \rangle \) \) are regarded as the output lines of the first QEC (quantum evaluating circuits in Figure 4-7).

When the value of \( k \) is equal to \( a \), from Lemma 4-3, the \( a \)th QEC (quantum evaluating circuits in Figure 4-7) to the \( a \)th subset \( C_a \) is used to select the right choices and to identify the wrong choices among \( 2^n \) possible choices. Therefore, \( (\left| c_{a}^0 \right> \otimes (\left| c_{a-1} \right> \otimes (\otimes_{q=n}^{1} | r_{a,q}^1 \rangle) \otimes (\left| r_{a,0}^0 \right>) \otimes (\otimes_{q=n}^{1} | u_{q} \rangle) \rangle \) are regarded as the input lines of the \( a \)th QEC (quantum evaluating circuits in Figure 4-7) and \( (\left| c_{a}^0 \oplus c_{a-1} \cdot r_{a,n} \right> \otimes (\left| c_{a-1} \right>) \rangle \) \( \otimes (\otimes_{q=n}^{2} (r_{a,q}^1 \oplus \overline{U}_q \cdot \overline{F}_{a,q-1}) \rangle \) \( \otimes \left( \left| r_{a,1}^1 \oplus \overline{U}_1 \cdot \overline{F}_{a,0}^0 \right> \rangle \otimes (\left| r_{a,0}^0 \right>) \otimes (\otimes_{q=n}^{1} | u_{q} \rangle \) \) are regarded as the output lines of the \( a \)th QEC (quantum evaluating circuits in Figure 4-7).

When the value of \( k \) is equal to \((a + 1)\), from Lemma 4-3, the \((a + 1)\)th QEC (quantum evaluating circuits in Figure 4-7) to the \((a + 1)\)th subset \( C_{a+1} \) is employed to select the right choices and to identify the wrong choices
among \(2^n\) possible choices. So, \(\left| c_{a+1}^0 \right\rangle \otimes \left| c_{a} \right\rangle \otimes \left( \otimes_{q=n}^{1} \left| r_{a+1,q}^1 \right\rangle \right) \otimes \left( \left| r_{a+1,0}^0 \right\rangle \right) \otimes \left( \otimes_{q=n}^{1} |r_q \rangle \right)\) are regarded as the input lines of the \((a + 1)\)th QEC (quantum evaluating circuits in Figure 4-7) and
\(\left( \left| c_{a+1}^0 \oplus c_q \cdot r_{a,n} \right\rangle \right) \otimes \left( \left| c_{a} \right\rangle \right) \otimes \left( \otimes_{q=n}^{2} \left( \left| r_{a+1,q}^1 \oplus \bar{H}_q \cdot \bar{r}_{a+1,q-1} \right\rangle \right) \right) \otimes \left( \left| r_{a+1,1}^1 \oplus \bar{H}_q \cdot \bar{r}_{a+1,0}^0 \right\rangle \right) \otimes \left( \left| r_{a+1,0}^0 \right\rangle \right) \otimes \left( \otimes_{q=n}^{1} |r_q \rangle \right)\) are regarded as the output lines of the \((a + 1)\)th QEC (quantum evaluating circuits in Figure 4-7).

Based the rules above, in Figure 4-8, quantum networks (QN) of selecting legal hitting sets and identify illegal hitting sets among \(2^n\) possible choices are drawn. Therefore, it is at once inferred that quantum networks (QN) of selecting legal hitting sets and identify illegal hitting sets among \(2^n\) possible choices can be implemented by means of QEC, and are drawn in Figure 4-8.

Figure 4–8: The full network is QN (the abbreviation of quantum networks).

H. INTRODUCING QUANTUM NETWORKS FOR FINDING MINIMUM-SIZED HITTING SETS
From **Definition 4-1** of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \), the answer is to find a hitting set with the minimum size of elements among legal hitting sets in \( S \) and \( C \). Since **Algorithm 4-1** is the DNA-based algorithm for solving the hitting-set problem, Steps (6) and (7) in **Algorithm 4-1** are the outer loop and the inner loop, respectively, of the nested loop and they are applied to find a minimum-sized hitting-set. The value of the outer loop index variable \( i \) is from 0 to \( n - 1 \) and the value of the inner loop index variable \( j \) is from \( i \) down to 0, so Steps (6) and (7) in **Algorithm 4-1** are mainly used to compute the influence of \( u_{i+1} \) for the number of ones in tubes (sets) \( T_0 \) through \( T_{j+1} \) for that the value of \( j \) is from \( i \) through 0. On each execution of Step (7a) in **Algorithm 4-1**, it applies the extract operation from tube (set) \( T_j \) to form two different tubes (sets), \( T_{j+1}^{ON} \) and \( T_j \). It is indicated that tube (set) \( T_{j+1}^{ON} \) includes those combinations that have \( u_{i+1} = 1 \) and tube (set) \( T_j \) consists of those combinations that have \( u_{i+1} = 0 \). Those combinations in tube (set) \( T_j \) have \( j \) ones, so those combinations in \( T_{j+1}^{ON} \) have \((j + 1)\) ones. Thus, next, on each execution of Step (7b) in **Algorithm 4-1**, it uses the merge operation to pour tube (set) \( T_{j+1}^{ON} \) into tube (set) \( T_{j+1} \). This implies that those combinations in tube (set) \( T_{j+1} \) have \((j + 1)\) ones. Repeat the execution of Steps (7a) and (7b) in **Algorithm 4-1** until the influence of \( u_n \) for the number of ones in tubes (sets) \( T_0 \) through \( T_n \) is processed. This is to say that those combinations in tube (set) \( T_i \) for \( 0 \leq i \leq n \) have \( i \) ones.

For finishing the parallel logic computation generated by Steps (7a) and (7b) at the same iteration in Steps (6) and (7) in **Algorithm 4-1**, auxiliary quantum bits for \( 0 \leq i \leq n - 1 \) and \( 0 \leq j \leq i \), \( f_{i,j,0} \), \( h_{i,j,j-i+1} \), \( h_{i,j,0} \), and \( z_{0,0} \) are needed. For \( 0 \leq i \leq n - 1 \) and \( 0 \leq j \leq i \), each quantum bit in \( z_{i+1,j} \), \( z_{i+1,i+1} \), \( g_{i,j,0} \), \( f_{i,j,0} \), and \( h_{i,j,j-i+1} \) is initially prepared in state \( \{0\} \), and each quantum bit in \( h_{i,j,0} \) and \( z_{0,0} \) is initially prepared in state \( \{1\} \). It is supposed that for \( 0 \leq i \leq n - 1 \) and \( 0 \leq j \leq i \), \( z_{i+1,j+1} \) is employed to record the status of tube (set) \( T_{j+1} \) that has \((j + 1)\) ones, and \( z_{i+1,j} \) is applied to record the status of tube (set) \( T_j \) that has \( j \) ones after the influence of \( u_{i+1} \) to the number of ones is computed from the loop iteration \((i, j)\) in the two-level nested loop of Steps (6) and (7) in **Algorithm 4-1**. **Lemma 4-5** is applied to describe that the parallel logic computation is performed by Steps (7a) and (7b) at the same iteration \((i, j)\) in the two-level nested loop of Steps (6) and (7) in **Algorithm 4-1**.

**Lemma 4-5**: The parallel logic computation finished by Steps (7a) and (7b) at the same iteration \((i, j)\) in the two-level nested loop of Steps (6) and (7) in **Algorithm 4-1** is

\[
|z_{i+1,j+1}\rangle = |z_{i+1,j1}\rangle \oplus \left(|c_m\rangle \land (|u_{i+1}\rangle \land |z_{i,j}\rangle \land (\lor_{k=j+2}^{j+1} |\overline{z}_{i+1,k}\rangle)\right) \text{ and } |z_{i+1,j}\rangle = |z_{i+1,j}\rangle \oplus \left(|c_m\rangle \land |\overline{u}_{i+1}\rangle \land |z_{i,j}\rangle\right). \tag{4.6}
\]
Proof:

A mathematical induction is used to complete the proof. In the two-level nested loop of Steps (6) and (7) in Algorithm 4-1, the value of the first loop index variable $i$ is from 0 through $n - 1$, and the value of the second loop index variable $j$ is from $i$ down to 0. When the value of $i$ is equal to zero, the value of $j$ is only equal to zero. Therefore, the parallel logic computation finished by Steps (7a) and (7b) at the same iteration $(0, 0)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 consists of solutions (legal hitting sets) in a tube (a set) $T_1$ including the first element and solutions (legal hitting sets) in a tube (a set) $T_0$ not containing the first element. Based on the first and the second conditions of (4.6) and the iteration $(0, 0)$, $|z_{i,1}⟩ = |z_{i,1}⟩ ⊕ (|c_m⟩ \land u_1 \land |z_{0,0}⟩)$ and $|z_{i,0}⟩ = |z_{i,0}⟩ \oplus (|c_m⟩ \land \overline{u}_1 \land |z_{0,0}⟩)$ are obtained. Because the initial state for $|z_{i,1}⟩$ is zero, the value of $|z_{i,1}⟩$ is changed to one if and only if the values of three quantum bits ($|c_m⟩$, $u_1$, and $|z_{0,0}⟩$) are all one. The initial state for $|z_{i,0}⟩$ is also zero, so the value of $|z_{i,0}⟩$ is changed to one if and only if the values of three quantum bits ($|c_m⟩$, $\overline{u}_1$, and $|z_{0,0}⟩$) are all one. This is to say that if the value for $|z_{i,1}⟩$ is equal to one, then $|z_{i,1}⟩$ is used to indicate that solutions (legal hitting sets) in a tube (a set) $T_1$ include the first element and have one ones, and if the value for $|z_{i,0}⟩$ is equal to one, then $|z_{i,0}⟩$ is used to indicate that solutions (legal hitting sets) in a tube (a set) $T_0$ do not contain the first element and have zero ones. Hence, it is indicated that the parallel logic computation performed by Steps (7a) and (7b) at the same iteration $(0, 0)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 can be implemented by the first and the second conditions of (4.6).

Next, when the value of $i$ is equal to $p$ for $0 \leq p \leq n - 1$ and the value of $j$ is equal to $p$, it is assumed that the parallel logic computation performed by Steps (7a) and (7b) at iteration $(p, p)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 can be implemented by the first and second conditions of (4.6). When the value of $i$ is equal to $p$, the value of $j$ is then $p - 1$. Therefore, the parallel logic computation performed by Steps (7a) and (7b) at iteration $(p, p - 1)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 consists of solutions (legal hitting sets) in a tube (a set) $T_{p-1,p+1}$ containing the $(p + 1)$th element and have $p$ ones, and solutions (legal hitting sets) in a tube (a set) $T_{p-1}$ do not include the $(p + 1)$th element and have $(p - 1)$ ones. From the first and second conditions of (4.6) and the iteration $(p, p - 1)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1, $|z_{p+1,(p-1)+1}⟩ = |z_{p+1,(p-1)+1}⟩ \oplus (|c_m⟩ \land u_{p+1} \land |z_{p,p-1}⟩ \land |z_{p+1,p+1}⟩)$ and $|z_{p+1,p-1}⟩ = |z_{p+1,p-1}⟩ \oplus (|c_m⟩ \land \overline{u}_{p+1} \land |z_{p,p-1}⟩)$ are obtained.

The initial state for $|z_{p+1,(p-1)+1}⟩$ is zero, and if its value is changed to one at the previous iteration $(p, p)$ in the
two-level nested loop of Steps (6) and (7) in Algorithm 4-1, then the value for $\overline{u}_{p+1}$ is one and the value to $u_{p+1}$ is zero. Thus, under this condition, the value one of $\left| z_{p+1, (p-1)+1} \right|$ is preserved because the value of $u_{p+1}$ is zero. In other words, when the value for $\left| z_{p+1, (p-1)+1} \right|$ is not changed to one at the previous iteration $(p, p)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1, the value of $1 + pu$ is one and the value to $1 + pu$ is zero. Thus, under this condition, the value one of $1 + pu$ is preserved because the value of $1 + pu$ is zero. In other words, when the value for $\left| z_{p+1, (p-1)+1} \right|$ is not changed to one at the previous iteration $(p, p)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1, the value of $1 + pu$ is changed to one if and only if the values of quantum bits ($\left| c_m \right|$, $u_{p+1}$, $\left| z_{p+1,p} \right|$, and $\left| \overline{z}_{p+1,p} \right|$) are all one. The value of $\left| z_{p+1,p} \right|$ is changed to one if and only if the values of three quantum bits ($\left| c_m \right|$, $\overline{u}_{p+1}$, and $\left| z_{p+1} \right|$) are all one. This is to say that if the value for $\left| z_{p+1,(p-1)+1} \right|$ is equal to one, then $\left| z_{p+1,(p-1)+1} \right|$ is applied to indicate that solutions (legal hitting sets) in a tube (a set) $T_{(p-1)+1}$ contain the $(p+1)$th element and have $p$ ones, and if the value for $\left| z_{p+1,p} \right|$ is equal to one, then $\left| z_{p+1,p} \right|$ is applied to indicate that solutions (legal hitting sets) in a tube (a set) $T_{p-1}$ do not consist of the $(p+1)$th element and have $(p-1)$ ones. Hence, it is indicated that the parallel logic computation finished by Steps (7a) and (7b) at iteration $(p, p-1)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 can be implemented by the first and second conditions of (4.6). Thus, it is at once concluded that the parallel logic computation completed by Steps (7a) and (7b) at iteration $(i, j)$ in the two-level nested loop of Steps (6) and (7) in Algorithm 4-1 is (4.6):

$$\left| z_{i+1,j} \right| = \left| z_{i+1,j} \right| \oplus \left( \left| c_m \right| \wedge (u_{i+1} \wedge \left| z_{i,j} \right| \wedge (\wedge_{k=j+2}^{i+1} \left| \overline{z}_{i+1,k} \right|)) \right)$$

and

$$\left| z_{i+1,j} \right| = \left| z_{i+1,j} \right| \oplus \left( \left| c_m \right| \wedge \overline{u}_{i+1} \wedge \left| z_{i,j} \right| \right).$$

I. CONSTRUCTING QUANTUM NETWORKS FOR FINDING MINIMUM-SIZED HITTING SETS

From Lemma 4-5, Equation (4.6) can be used to determine which legal hitting sets own the minimum size of elements. Therefore, Lemma 4-6 is applied to show how complete quantum networks (CQN) of Equation (4.6), computing the influence of $u_{i+1}$ to the number of ones among legal hitting sets, are implemented by means of NOT gates, AND gates and CCNOT gates.

Lemma 4-6: Complete quantum networks (CQN) for implementing the function of Equation (4.6) can be implemented by means of NOT gates, AND gates and CCNOT gates, and are drawn in Figure 4-9.

Proof:

The first condition of Equation (4.6) in Lemma 4-5 is $\left| z_{i+1,j+1} \right| = \left| z_{i+1,j+1} \right| \oplus \left( \left| c_m \right| \wedge (u_{i+1} \wedge \left| z_{i,j} \right| \wedge (\wedge_{k=j+2}^{i+1} \left| \overline{z}_{i+1,k} \right|)) \right)$. Therefore, performing the first condition of Equation (4.6) is to compute $\left| z_{i+1,j+1} \right|$ for $0 \leq i \leq$
The task requires computing $(i − j)$ NOT gates and $(i − j + 2)$ AND gates through (1) the relation $h_{i,j,a} \leftarrow h_{i,j,a} \oplus (h_{i,j,a−1} \cdot \mathbb{Z}_{j+1} \cdot k_{a})$ for $1 \leq a \leq i$ and $j+2 \leq k \leq i+1$, (2) the relation $h_{i,j,i−j+1} \leftarrow h_{i,j,i−j+1} \oplus (h_{i,j,i−j} \cdot \mathbb{Z}_{j+2})$, and (3) the relation $f_{i,j,0} \leftarrow f_{i,j,0} \oplus (h_{i,j,i−j+1} \cdot u_{i+1})$. Next, the task requires computing one CCNOT gates through the relation $z_{i+1,j+1} \leftarrow z_{i+1,j+1} \oplus (c_m \cdot f_{i,j,0})$. The quantum bit $\left| z_{i+1,j+1} \right>$ is applied to store the evaluating result of the first condition of (4.6) in Lemma 4-5. Subsequently, the NOT gates on $\mathbb{Z}_{i+1,j+1}$ for $j + 2 \leq k \leq i + 1$ are reversed to restore each quantum bit $\mathbb{Z}_{i+1,j+1}$ to its previous state. This enables to reuse $\mathbb{Z}_{i+1,j+1}$ for $j + 2 \leq k \leq i + 1$. 
Because the second condition of Equation (4.6) in Lemma 4-5 is \( z_{i+1,j} = z_{i+1,j} \oplus (c_m \land \overline{u}_{i+1} \land z_{i,j}) \), performing the second condition of (4.6) in Lemma 4-5 is to compute \( z_{i+1,j} \) for \( 0 \leq i \leq n - 1 \) and \( 0 \leq j \leq i \). This task requires computing one NOT gate on \( u_{i+1} (\overline{u}_{i+1}) \), one AND gate and one CCNOT gate through the relation \( g_{i,j,0} \leftarrow g_{i,j,0} \oplus (z_{i,j} \cdot \overline{u}_{i+1}) \) and the relation \( z_{i+1,j} \leftarrow z_{i+1,j} \oplus (c_m \cdot g_{i,j,0}) \). Subsequently, all the NOT gates on \( u_n \cdots u_1 \) are reversed to restore every quantum bit in \( u_n \cdots u_1 \) to its superposition state. This enables to preserve the superposition in \( u_n \cdots u_1 \) and to reuse the superposition in \( u_n \cdots u_1 \).

Based on the rules above, complete quantum networks (CQN) for implementing the function of Equation (4.6) are drawn in Figure 4-9. Therefore, it is at once inferred that complete quantum networks (CQN) for implementing the function of Equation (4.6) can be implemented by means of NOT gates, AND gates and CCNOT gates, and are drawn in Figure 4-9.

J. QUANTUM ALGORITHMS FOR CALCULATING THE NUMBER OF ONES TO LEGAL HITTING SETS

The following quantum algorithm is proposed to work on the physical quantum computer proposed by Deutsch [6] and is applied to figure out the number of ones to legal hitting sets in the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \). For convenience of presentation, it is supposed that \( u_1 = (z_1, 0, 1) \), \( r_1 = (z_1, 1) \), \( c_1 = (z_1, 1) \), \( h_1 = (z_1, 0) \), \( f_1 = (z_0, 1) \), \( g_1 = (z_0, 1) \), \( z_1 = (z_0, 1) \), and \( z_{i+1,i+1} = (z_0, 1) \) for \( 1 \leq q \leq n \), \( 0 \leq k \leq m \), \( 0 \leq i \leq n - 1 \), \( 0 \leq j \leq i \), and \( 0 \leq a \leq i - j + 1 \), subsequently, denote the value of their corresponding quantum bits to be 1. Also it is assumed that \( u_q = (z_0, 0) \), \( r_q = (z_0, 0) \), \( c_q = (z_0, 0) \), \( h_q = (z_0, 0) \), \( f_q = (z_0, 0) \), \( g_q = (z_0, 0) \), \( z_q = (z_0, 0) \), and \( z_{i+1,i+1} = (z_0, 0) \) for \( 1 \leq q \leq n \), \( 0 \leq k \leq m \), \( 0 \leq i \leq n - 1 \), \( 0 \leq j \leq i \), and \( 0 \leq a \leq i - j + 1 \), subsequently, denote the value of their corresponding quantum bits to be 0. Moreover, the notations used in Algorithm 4-2 below have been denoted in previous subsections. The first parameter \( w \), in Algorithm 4-2 is employed to represent the minimum size of elements among legal answers, and its value is passed from the execution of Step (1a) in Algorithm 4-3 in next subsection. To increase the probability of success on measuring the answer from among \( 2^n \) choices, Grover’s algorithm [18] is integrated into the proposed quantum algorithm and is used to significantly increase the amplitudes of those answers. Grover’s operator is diffusion transform \( G \), which is defined by matrix \( G \) as follows: \( G_{i,j} = \frac{2}{2^n} \) if \( i \neq j \) and \( G_{i,i} = (-1+\frac{2}{2^n}) \).

Algorithm 4-2: Quantum algorithms of figuring out the number of ones (the number of elements) to legal hitting sets in the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).
(1) For an initial input \( \Phi \) = \( |1\rangle \otimes (\otimes_{i=n}^{0} \otimes_{j=i}^{0} |z_{i,j}^{0}\rangle ) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} |g_{i,j,0}\rangle ) \otimes (\otimes_{i=i-1}^{0} \otimes_{j=i}^{0} |f_{i,j,0}\rangle ) \), \( 2^n \) possible choices of \( n \) bits (containing all of the possible choices) are

\[
|\varphi_{0,0}\rangle = (H) \otimes (\otimes_{i=n}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (I_{2x2}) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (\otimes_{i=i-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (\otimes_{i=m}^{1} I_{2x2}) \otimes (I_{2x2})
\]

(2) \( \varphi_{0,0} \) = \( (I_{2x2}) \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{0} I_{2x2}) \otimes (I_{2x2}) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} I_{2x2}) \otimes \) QN \otimes \( \varphi_{0,0} \) = \( (\otimes_{i=n}^{0} \otimes_{j=i}^{0} |z_{i,j}^{0}\rangle ) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} |f_{i,j,0}\rangle ) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} |g_{i,j,0}\rangle ) \otimes (\otimes_{i=m}^{1} I_{2x2}) \otimes (I_{2x2}) \)

QN is the quantum circuit in Figure 4-8 in Lemma 4-4.

(3) For \( i = 0 \) to \( n - 1 \)

(4a) \( \varphi_{i,j} \) (i.e. \( i = \sum_{k=0}^{1}(a_k+1)+(i-j)+1, 0 \)) = \( (I_{2x2}) \otimes \) QN \otimes \( \varphi_{i,j} \) = \( (\otimes_{i=n}^{0} \otimes_{j=i}^{0} |z_{i,j}^{0}\rangle ) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} |g_{i,j,0}\rangle ) \otimes (\otimes_{i=i-1}^{0} \otimes_{j=i}^{0} |f_{i,j,0}\rangle ) \otimes (\otimes_{i=n-1}^{0} \otimes_{j=i}^{0} |h_{i,j,0}\rangle ) \otimes (\otimes_{i=k}^{1} c_{k} \otimes (c_{k-1} \otimes r_{k,n})) \otimes (\otimes_{i=q}^{1} (u_{q}^{0} + u_{q}^{1})) \), where

\( z_{i,j,l} \) is the quantum circuit in Figure 4-8 in Lemma 4-4.
\[
(\otimes^0_{i=j-1} |z_{s,l}^0\rangle) \otimes (\otimes^0_{s=i} (\otimes^0_{l=s} z_{l,1}^0)) \otimes (|z_{0,0}^1\rangle) \otimes (\otimes^0_{d=n-1} \otimes^0_{e=d} g_{d,e,0}^0) \otimes \\
(\otimes^i_{d=i} ((\otimes^0_{e=d} g_{d,e,0})) \otimes \left|g_{i,j,0}^0 + (u_{i,j+1} \cdot z_{i,j})\right\rangle \otimes (\otimes^0_{e=j-1} g_{d,e,0}^0) \otimes (\otimes^0_{e=d-i} (\otimes^0_{e=d} g_{d,e,0})) \otimes \\
(\otimes^i_{d=n-1} \otimes^0_{e=d} f_{d,e,0}^0) \otimes (\otimes^i_{d=i} ((\otimes^0_{e=d} f_{d,e,0}) \otimes \left|f_{i,j,0}^0 + (u_{i,j+1} \cdot h_{i,j-j+1})\right\rangle \otimes (\otimes^0_{e=j-1} f_{d,e,0}^0) \otimes \\
(\otimes^0_{d=i} \otimes^0_{e=d} f_{d,e,0}^0) \otimes (\otimes^i_{d=n-1} \otimes^0_{e=d} ((\otimes^1_{a=d-e+1} h_{d,e,a}^0) \otimes (h_{d,e,0}^1))) \otimes \\
(\otimes^i_{a=i-j} (h_{i,j,a}^0 \otimes (z_{i,j+1} \cdot h_{i,j-1}))) \otimes (\left|h_{i,j,0}^1\right\rangle) \otimes (\otimes^0_{e=j-1} ((\otimes^1_{a=d-e+1} h_{d,e,a}^0) \otimes (h_{d,e,0}^1))) \otimes \\
(\otimes^0_{d=i} \otimes^0_{e=d} ((\otimes^1_{a=d-e+1} h_{d,e,a}^0) \otimes (h_{d,e,0}^1))) \otimes (\left|h_{i,j,0}^1\right\rangle) \otimes (\otimes^1_{k=m} c_k^0) \otimes (\left|c_0^1\right\rangle) \otimes \\
(\otimes^1_{k=n} ((\otimes^0_{q=n} r_{k,q}^0)) \otimes (r_{k,0}^0))) \otimes (\otimes^1_{q=n} (|u_q^0\rangle + |u_q^1\rangle)),
\]
where CQN is the quantum circuit in Figure 4-9 in Lemma 4-6, and for \(m \geq k \geq 1\) and \(n \geq q \geq 1\) quantum bits \(|c_k\rangle\) and \(|r_{k,q}\rangle\) are all the results generated in Step (2).

\textbf{End For}

\textbf{End For}

\[(5) \quad \left|\varphi_{1, n(n+1)+1, 0}^{\otimes n(n+1)+1, 0}\right\rangle = \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}} \otimes (\otimes^0_{i=n} \otimes^0_{j=i} I_{2x2}) \otimes \left(I_{2x2}\right) \otimes (\otimes^0_{i=n-1} \otimes^0_{j=i} I_{2x2}) \otimes \\
(\otimes^0_{i=n-1} \otimes^0_{j=i} I_{2x2}) \otimes (\otimes^0_{i=n-1} \otimes^0_{j=i} ((\otimes^1_{a=i-j+1} h_{i,j,a}^0) \otimes (I_{2x2}))) \otimes (\otimes^1_{k=m} I_{2x2}) \otimes \left(I_{2x2}\right) \otimes \\
(\otimes^1_{k=m} (\otimes^1_{q=n} I_{2x2}) \otimes \left(I_{2x2}\right)) \otimes (\otimes^1_{q=n} I_{2x2}) \mid \varphi_{1, n(n+1)+1, 0}^{\otimes n(n+1)+1, 0}\left\rangle = \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}} \otimes \\
\frac{1}{\sqrt{2^n}} \otimes \left(|z_{i,j}\rangle \otimes (\otimes^0_{i=n} \otimes^0_{j=i} \mid z_{i,j}\rangle) \otimes \left|z_{0,0}^1\right\rangle \otimes (\otimes^0_{i=n-1} \otimes^0_{j=i} g_{i,j,0}^0) \otimes (\otimes^0_{i=n-1} \otimes^0_{j=i} f_{i,j,0}^0) \otimes \\
(\otimes^0_{i=n-1} \otimes^0_{j=i} (\otimes^1_{a=i-j+1} h_{i,j,a}^0) \otimes (h_{i,j,0}^1))) \otimes (\otimes^1_{k=m} \mid c_k\rangle) \otimes (c_0^1) \otimes (\otimes^1_{k=n} (\otimes^1_{q=n} r_{k,0}^0)) \otimes \\
(\otimes^1_{q=n} (\mid u_q^0\rangle + \mid u_q^1\rangle)).\]

(6) Since quantum operations are naturally reversible, the auxiliary quantum bits can be restored to their initial states by reversing the operations from Steps (4a) to (2).

(7) Apply Grover’s operator in Grover’s algorithm to the quantum state vector generated in Step (6).
(8) At most repeat to execute from Step (2) to Step (7) of \( O\left(\sqrt{\frac{2^n}{R}}\right) \) times, where the value of \( R \) is the number of solutions and can be efficiently computed from quantum counting algorithm in [22].

(9) The answer is obtained with a successful probability of at least \( \frac{1}{2} \) after a measurement is finished.

End Algorithm

Lemma 4-7: Algorithm 4-2 is used to calculate the number of ones (the number of elements) among legal hitting sets in the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).

Proof:

Since there are \( 2^n \) possible choices (including all possible hitting sets) for the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \), a quantum register of \( n \) bits \((1 \otimes_q\ldots\otimes_q u_q)\) is used to represent \( 2^n \) choices with initial state \((0\ldots0)\). The hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) requires finding a minimum-sized hitting-set, so those auxiliary quantum registers are needed. From the execution of Step (1), an initial vector \( |\Phi\rangle = |1\rangle \otimes (\otimes_{i=n}^0 z_{i,j}^0) \otimes (z_{0,0}^1) \otimes (\otimes_{i=n-1}^0 g_{i,j,0}^0) \otimes (\otimes_{i=n-1}^0 f_{i,j,0}^0) \otimes (\otimes_{i=n-1}^0 h_{i,j,a}^0) \otimes (h_{1,j,0}^1) \otimes (c_{k}^0) \otimes (c_{k}^1) \otimes (\otimes_{k=m}^1 r_{k,q}^1) \otimes (r_{0,0}^0) \otimes (u_q^0) \) starts the quantum computation of the hitting-set problem. \( H^{\text{on}} \) that stands for the joined \( n \)-qubit Hadamard gate is applied to the part of choices of the initial vector \( |\Phi\rangle \), and then the resulting state vector becomes \( |\varphi_{0,0}\rangle \) with \( 2^n \) choices. This is to say that the function finished by Step (0a) through Step (1d) in Algorithm 4-1 can be implemented by Step (1) in Algorithm 4-2.

Next, Step (2) in Algorithm 4-2 works as the unitary operator \( QN \) that is the quantum circuit in Figure 4-8 in Lemma 4-4. On each execution of Step (2), it is used to find choices among \( 2^n \) possible choices satisfying each formula of the form \( u_n \lor u_{n-1} \ldots \lor u_2 \lor u_1 \). After Step (2) is executed, the resulting state vector \( |\varphi_{1,0}\rangle \) is obtained including those legal choices with \( |c_{m}^1\rangle \) that include at least one element from each subset in \( C \) and those illegal choices with \( |c_{m}^0\rangle \) that do not satisfy the condition. This implies that the function completed by Steps (3a) through (5a) in Algorithm 4-1 can be implemented by Step (2) in Algorithm 4-2.
Then, Step (4a) is embedded in the first loop in Algorithm 4-2 and works as the unitary operator $CQN$ which is the quantum circuit in Figure 4-9 in Lemma 4-6. The step is employed to figure out the number of ones (the number of elements) among the legal choices. After repeating Step (4a), the resulting state vector $\left| \varphi_{\frac{\sqrt{m(n+1)}}{2}, 0} \right\rangle$ is obtained in which the number of elements in each legal hitting set is computed. This indicates that the function finished by Steps (7a) through (7b) in Algorithm 4-1 can be implemented by Step (4a) in Algorithm 4-2.

Next, one CNOT gate, $(\frac{0}{\sqrt{2}} - i) \otimes z_{n,w}$, in the execution of Step (5) is applied to carry out the oracle work (in the language of Grover’s algorithm), that is, the target state labeling preceding Grover’s searching step. The resulting state vector $\left| \varphi_{\frac{\sqrt{m(n+1)}}{2}, -1, 0} \right\rangle$ consists of the part of the answer with phase $-1$ and the other part with phase $+1$. Since quantum operations are reversible by nature, those auxiliary quantum bits can be restored to their initial states by simply applying the reverse operation, and then they can be repeated for safe use. Therefore, the execution of Step (6) is used to reverse all those operations performed by Steps (4a) and (2).

Next, each execution of Step (7) in Algorithm 4-2 applies Grover’s operator in Grover’s Algorithm to perform the task that is to increase the probability of success on measuring the answer. After repeating Steps (2) through (7) for $O\left(\frac{2^n}{R}\right)$ times, a maximum successful probability is generated. Then, from the execution of Step (9) in Algorithm 4-2, a measurement is used to obtain the answer(s) and the answer(s) is/are returned to Algorithm 4-3. Therefore, it is at once inferred that Algorithm 4-2 can be used to calculate the number of ones among legal hitting sets in the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$. ■

K. QUANTUM ALGORITHMS OF SOLVING THE HITTING-SET PROBLEM

The following quantum algorithm is used to solve the hitting problem with an $n$-element finite set $S$ and an $m$-subset collection $C$. The notations used in Algorithm 4-3 below have been denoted in previous subsections.

Algorithm 4-3: Quantum algorithms of solving the hitting problem with an $n$-element finite set $S$ and an $m$-subset collection $C$.

(1) For $w = 1$ to $n$
   (1a) Call Algorithm 4-2($w$).
   (1b) If the answer is obtained from the $w$th execution of Step (1a) then
       (1c) Terminate Algorithm 4-3.
   End If
Lemma 4-8: Algorithm 4-3 is the quantum implementation of Algorithm 4-1 (the DNA-based algorithm) which is equivalent to the oracle work (in the language of Grover’s Algorithm), that is, the target state labeling preceding Grover’s searching step, and is used to solve an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).

Proof:

In each execution of Step (1a) in Algorithm 4-3, Algorithm 4-2 is used to perform the two main tasks. The first main task is to compute the number of elements in each legal hitting set. This is to say that from Lemma 4-7 the oracle work in the language of Grover’s algorithm, that is, the target state labeling preceding Grover’s searching step, can be implemented by Algorithm 4-2. The second main task is to call for Grover’s algorithm that increases the probability of success in measuring the answer from the minimum-sized hitting sets. Next, in each execution of Step (1b) in Algorithm 4-3, if the answer is found from the \( w \)th execution of Step (1a) in Algorithm 4-3, then the \( w \)th execution of Step (1c) in Algorithm 4-3 is applied to terminate Algorithm 4-3. Otherwise, Steps (1a) through (1c) are executed until the answer for solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is found.

V. COMPLEXITY ASSESSMENT

The following lemmas are used to show the time complexity and the space complexity of Algorithm 4-3 for solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).

Lemma 5-1: The best case for time complexity of solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is \( O(n + 1) \) Hadamard gates, \( O(\frac{n}{2^2} \times (4 \times m \times n + (\frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3})) \) NOT gates, \( O(\frac{n}{2^2}) \) CNOT gates, \( O(\frac{2^2}{2^2} \times (2 \times m \times n + 2 \times m) + (\frac{n^3 + 15 \times n^2 + 14 \times n}{3})) \) CCNOT gates, \( O(2^2) \) Grover’s operators, and \( O(1) \) measurements.

Proof:

Step (1) is the main loop in Algorithm 4-3 and contains \( n \) iterations to execute those steps embedded in the main loop. Hence, on the first execution of Step (1a), it calls for Algorithm 4-2. From the execution of Step (1) in Algorithm 4-2, \( (n + 1) \) Hadamard gates are performed. Then, from the execution of Step (2) in Algorithm 4-2, (2
$m \times n$) NOT gates and $(m \times n + m)$ CCNOT gates are implemented. Next, Step (4a) of Algorithm 4-2 is embedded in the first loop, and Step (4a) of Algorithm 4-2 produces $\left(\frac{n \times (n+1) \times (n+2)}{3}\right)$ NOT gates and $\left(\frac{n \times (n+1) \times (n+14)}{6}\right)$ CCNOT gates. Then, from the execution of Step (5) in Algorithm 4-2, one CNOT gate is carried out. Next, Step (6) of Algorithm 4-2 is applied to restore the auxiliary quantum bits back to their original status. Therefore, Step (6) of Algorithm 4-2 results in $((2 \times m \times n) + \left(\frac{n \times (n+1) \times (n+2)}{3}\right))$ NOT gates and $(m \times n + m) + \left(\frac{n \times (n+1) \times (n+14)}{6}\right)$ CCNOT gates. This is to say that Steps (2) through (6) are employed to perform one oracle work. Step (7) of Algorithm 4-2 is then used to invoke Grover’s operator in Grover’s algorithm to significantly increase the amplitudes of those answers. This indicates that one Grover’s operator is implemented from the execution of Step (7) in Algorithm 4-2. Next, from Step (8) of Algorithm 4-2, it is indicated that at most repeat to execute from Step (2) to Step (7) of $(\tilde{O}(\sqrt{\frac{2^n}{R}}))$ times. Because the value of $R$ is equal to one and this case is the worst case, $(\sqrt{2^n})$ oracle works and $(\sqrt{2^n})$ Grover’s operators are implemented. Next, from the execution of Step (9) in Algorithm 4-2, a measurement is finished. Thus, after the first call of Algorithm 4-2 is performed, it is derived at once that $O(n+1)$ Hadamard gates, $O(2^n \times (4 \times m \times n + (\frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3})))$ NOT gates, $O(2^n)$ CNOT gates, $O(2^n \times ((2 \times m \times n + 2 \times m) + (\frac{n^3 + 15 \times n^2 + 14 \times n}{3})))$ CCNOT gates, $O(2^n)$ Grover’s operators, and $O(1)$ measurements are obtained.

Next, after the first call of Algorithm 4-2 is finished, from the first execution of Step (1b) in Algorithm 4-3, and if the answer is obtained from the first execution of Step (1a) in Algorithm 4-3, then Algorithm 4-3 is terminated in the first execution of Step (1c) in Algorithm 4-3. Hence, the best case for time complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is $O(n+1)$ Hadamard gates, $O(2^n \times (4 \times m \times n + (\frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3})))$ NOT gates, $O(2^n)$ CNOT gates, $O(2^n \times ((2 \times m \times n + 2 \times m) + (\frac{n^3 + 15 \times n^2 + 14 \times n}{3})))$ CCNOT gates, $O(2^n)$ Grover’s operators, and $O(1)$ measurements.

**Lemma 5-2:** The worst case for time complexity of solving an instance of the hitting-set problem with an $n$-element
finite set $S$ and an $m$-subset collection $C$ is $O(n \times (n + 1))$ Hadamard gates, $O(n \times (\frac{n^2}{2} \times (4 \times m \times n + \frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3} )))$ NOT gates, $O(n \times 2^{\frac{n}{2}})$ CNOT gates, $O(n \times (\frac{n^2}{2} \times ((2 \times m \times n + 2 \times m) + \frac{n^3 + 15 \times n^2 + 14 \times n}{3} )))$) CCNOT gates, $O(n \times (\sqrt{2^n}))$ Grover's operators, and $O(n \times 1)$ measurements.

Proof:

Algorithm 4-3 indicates that the worst case for solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is to find the answer after a measurement on the result generated from the $n$th execution of Step (1a) in Algorithm 4-3 is performed. This is to say that each step in Algorithm 4-3 is executed $n$ times. Thus, the worst case for time complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is $O(n \times (n + 1))$ Hadamard gates, $O(n \times (\frac{n^2}{2} \times (4 \times m \times n + \frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3} )))$ NOT gates, $O(n \times 2^{\frac{n}{2}})$ CNOT gates, $O(n \times (\frac{n^2}{2} \times ((2 \times m \times n + 2 \times m) + \frac{n^3 + 15 \times n^2 + 14 \times n}{3} )))$) CCNOT gates, $O(n \times (\sqrt{2^n}))$ Grover’s operators, and $O(n \times 1)$ measurements.

Lemma 5-3: The average case for time complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is $O((\frac{n+1}{2}) \times (n+1))$ Hadamard gates, $O((\frac{n+1}{2}) \times 2^{\frac{n}{2}} \times (4 \times m \times n + \frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3} )))$ NOT gates, $O((\frac{n+1}{2}) \times 2^{\frac{n}{2}})$ CNOT gates, $O((\frac{n+1}{2}) \times 2^{\frac{n}{2}} \times ((2 \times m \times n + 2 \times m) + \frac{n^3 + 15 \times n^2 + 14 \times n}{3} )))$ CCNOT gates, $O((\frac{n+1}{2}) \times (\sqrt{2^n}))$ Grover’s operators, and $O(\frac{n+1}{2})$ measurements.

Proof:

Assume that the time complexity of one time for executing each step in Algorithm 4-3 is $\Delta$. Thus, the average case for time complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is $((1 \times \Delta) + (2 \times \Delta) + \ldots + (n \times \Delta)) \div (n) = (\frac{n+1}{2}) \times \Delta$ since $\Delta$ is equal to the best case for time complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset
collection \( C \). Therefore, it is at once derived that the average case for time complexity of solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is \( \Theta((\frac{n+1}{2}) \times (n+1)) \) Hadamard gates, \( \Theta((\frac{n+1}{2}) \times (2^n \times (4 \times m \times n + (\frac{2 \times n^3 + 6 \times n^2 + 4 \times n}{3})))) \) NOT gates, \( \Theta((\frac{n+1}{2}) \times \frac{n}{2^2}) \) CNOT gates, \\
\( \Theta((\frac{n+1}{2}) \times (2^n \times ((2 \times m \times n + 2 \times m) + (\frac{n^3 + 15 \times n^2 + 14 \times n}{3})))) \) CCNOT gates, \( \Theta((\frac{n+1}{2}) \times (\sqrt{2^n})) \) Grover’s operators, and \( \Theta((\frac{n+1}{2})) \) measurements. 

**Lemma 5-4**: The best case for space complexity of solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is \( \Theta((n \times m + 2 \times m + 3) + (\frac{n^3 + 15 \times n^2 + 26 \times n}{6})) \) quantum bits.

**Proof:**

Since there are \( 2^n \) possible choices (including all possible hitting sets) for solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \), a quantum register with \( n \) bits \( (\otimes_{q=n}^{1} |u_q\rangle) \) is applied to represent \( 2^n \) choices with initial states \( (\otimes_{q=n}^{1} |u_q\rangle) \). The hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is to find a minimum-sized hitting set from those legal hitting sets, so those auxiliary quantum registers are needed. The initial states of those auxiliary quantum registers are \( |1\rangle \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{0} |z_{i,j}\rangle) \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{1} |z_{i,j}^1\rangle) \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{0} |g_{i,j,0}\rangle) \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{0} |f_{i,j,0}\rangle) \otimes (\otimes_{i=n}^{1} \otimes_{j=i}^{0} |h_{i,j,a}\rangle) \otimes (\otimes_{k=m}^{1} |c_k|) \otimes (\otimes_{q=n}^{1} |r_{k,q}|) \). It is indicated from **Algorithm 4-3** that the best case for space complexity of solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is to find the answer after **Algorithm 4-2** is only invoked once. Hence, the best case for space complexity of solving an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) is \( \Theta((n \times m + 2 \times m + 3) + (\frac{n^3 + 15 \times n^2 + 26 \times n}{6})) \) quantum bits. 

**Lemma 5-5**: The worst case for space complexity of solving an instance of the hitting-set problem with an
$n$-element finite set $S$ and an $m$-subset collection $C$ is $O((n \times m + 2 \times m + 3) + \left(\frac{n^3 + 15 \times n^2 + 26 \times n}{6}\right))$ quantum bits.

**Proof:**

From Lemma 5-2, Lemma 5-4 and Algorithm 4-3, quantum bits can be reused, so it is inferred that the worst case of space complexity is $O((n \times m + 2 \times m + 3) + \left(\frac{n^3 + 15 \times n^2 + 26 \times n}{6}\right))$ quantum bits. ■

**Lemma 5-6:** The average case for space complexity of solving an instance of the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$ is $O((n \times m + 2 \times m + 3) + \left(\frac{n^3 + 15 \times n^2 + 26 \times n}{6}\right))$ quantum bits.

**Proof:**

From Lemma 5-3 and Algorithm 4-3, quantum bits can be reused, so it is inferred at once that the average case for space complexity is $O((n \times m + 2 \times m + 3) + \left(\frac{n^3 + 15 \times n^2 + 26 \times n}{6}\right))$ quantum bits. ■

**VI. AN EXAMPLE OF THREE-QUBIT SOLUTION FOR THE SIMPLEST HITTING-SET PROBLEM**

Consider the simplest case of the hitting-set problem with a finite set $S \{1\}$ and a collection $C$ of subsets for $S \{1\}$. Figure 6-1 is the corresponding quantum circuit for the reduced version of Algorithm 4-3 that is to find the answer of the hitting-set problem of the finite set $S \{1\}$ and the collection $C$ of subsets for $S \{1\}$.

![Figure 6-1: The corresponding quantum circuit of the example above](image)
Our experiment is carried out on a Varian INOVA 600 NMR spectrometer. The sample is $^{13}\text{C}-\text{labelled}$ alanine with formula $^{13}\text{C}CH_3 - ^{13}\text{C}\text{H(NH}_2) - ^{13}\text{C}\text{COOH}$, where the three carbons $^{13}\text{C}, ^{13}\text{C}, ^{13}\text{C}$ correspond to the qubits $I_1, I_2, I_3$, respectively. The J-coupling constants are $J_{12} = 34.79\,\text{Hz}, J_{23} = 54.01\,\text{Hz}, J_{13} = 1.20\,\text{Hz}$. Soft pulses are used to achieve the selective excitation. The experiment has three main steps as follows.

**Step 1** is for initialization. Before the algorithm is carried out, the initial state, i.e., the pseudo-pure state, must be well prepared. There have many methods to do this job, among which the spatial averaging method proposed by [24] is most commonly used. So, in our experiment, we have also employed this technique to prepare the three-qubit pseudo-pure state $|000\rangle$ for which the detailed pulse sequence can be found in [24]. The states of the input qubits can be written in the form of the product operations as follows:

$$E + I_{1z} + I_{2z} + I_{3z} + 2I_{1z}I_{2z} + 2I_{1z}I_{3z} + 2I_{2z}I_{3z} + 4I_{1z}I_{2z}I_{3z},$$

where $E$ is the unity operator with the form of $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and $I_{iz} = \frac{1}{2} \sigma_i$, with $i = 1, 2, 3$, being the $i$th spin angular momentum operator in the $z$-direction, and $\sigma_i$ is the Pauli matrix $\sigma = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$.

**Step 2** translates the quantum gates into NMR pulses. We had to connect and optimize the pulses to construct the total NMR pulse sequence. The Hadamard gate can be achieved by a single $\pi/2$ pulse with phase $x$. The CNOT gate can be implemented by NMR pulses as follows [25]:

$$\left[\frac{\pi}{2}\right]_y \rightarrow (1/4J) \rightarrow \left[\frac{\pi}{2}\right]_z \rightarrow (1/4J) \rightarrow \left[\frac{\pi}{2}\right]_y \rightarrow \left[\frac{\pi}{2}\right]_z,$$

where the flip angle of the pulse and the time of delay are written in square brackets and in round brackets, respectively. The subscripts are the phases (i.e., along the $x$ or $y$ axis) of the pulse, and the superscripts are the nuclei to which the pulses are applied. Then we could obtain the total pulse sequence by connecting and optimizing the aforesaid pulses according to the quantum circuit.

**Step 3** is the measurement, where a readout pulse is applied to each qubit to obtain the spectra.

Note that in NMR measurements, the frequencies and phases of NMR signals could clearly indicate the state the system evolved to after the readout pulses had been applied. In our experiment, the phases of the reference of $^{13}\text{C}$ spectra from a thermal equilibrium were adjusted to be in absorption (i.e., to be positive), and then the same phase corrections were used to determine the absolute phases of the experimental spectra of $^{13}\text{C}$ after the algorithm was accomplished. In our case, the final state was $\left(\frac{|000\rangle_{123} + |111\rangle_{123}}{\sqrt{2}}\right)$ which means the three qubits are
entangled. As the readout by NMR is a weak measurement, we have no state collapse after the measurement. Besides, only single quantum coherence can be detected in NMR. As a result, we have to employ some additional operations to disentangle them for detecting the output state \( \frac{(|000\rangle_{123} + |111\rangle_{123})}{\sqrt{2}} \). For this end, we apply a CNOT gate on the second and first qubits to get the state \( \frac{(|000\rangle_{123} + |011\rangle_{123})}{\sqrt{2}} \). The second qubit is control qubit and the first qubit is the target qubit. Then the first qubit can be read out by a single \( \frac{\pi}{2} \) pulse along the \( x \)-axis, as shown in Figure 6-2 (a). Similar steps applied to the second and third quits, respectively, result in the spectrum in Figure 6-2 (b) and Figure 6-2 (c). It’s evident that the experimental results are in good agreement with our theoretical prediction.
VII. MATHEMATICAL REPRESENTATION OF MOLECULAR SOLUTIONS

The following lemma is applied to show how for solving the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$, molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space.

**Lemma 7-1**: For solving the hitting-set problem with an $n$-element finite set $S$ and an $m$-subset collection $C$, molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space.

**Proof**:

On each execution from Step (0) through Step (1d) in **Algorithm 4-1**, $2^n$ choices encoded by $2^n$ DNA strands are generated, and are also encoded by means of $n$ Hadamard gates operating $n$ initial quantum bits in Step (1) in **Algorithm 4-2**. This implies that $2^n$ choices encoded by $2^n$ DNA strands are represented in term of a vector unit in a finite-dimensional Hilbert space. Next, on each execution from Step (3a) through Step (5a) in **Algorithm 4-1**, legal choices and illegal choices among $2^n$ choices encoded by $2^n$ DNA strands are decided. The same task is also completed by means of unitary operators from Step (2) in **Algorithm 4-2**. This indicates that legal choices and
illegal choices among \(2^n\) choices encoded by \(2^n\) DNA strands are still represented in term of a unit vector in a finite-dimensional Hilbert space. Then, on each execution from Step (7a) through Step (7b) in Algorithm 4-1, legal choices among \(2^n\) choices encoded by \(2^n\) DNA strands are classified according to the number of vertex, and are also performed by means of unitary operators from Step (4a) in Algorithm 4-2. This is to say that legal choices classified among \(2^n\) choices encoded by \(2^n\) DNA strands are still represented in term of a unit vector in a finite-dimensional Hilbert space. Next, on each execution from Step (8a) through Step (8b) in Algorithm 4-1, the answer encoded by DNA strands with the minimum number of vertices is read, and is also read by means of a measurement after the Grover algorithm is applied to increase the amplitude of the answer. This implies that the answer encoded by DNA strands is still represented in term of a unit vector in a finite-dimensional Hilbert space. Therefore, it is at once derived that for solving the hitting-set problem with an \(n\)-element finite set \(S\) and an \(m\)-subset collection \(C\), molecular solutions are represented in term of a unit vector in a finite-dimensional Hilbert space.

VIII. CONCLUSIONS

From [22], many computing problems and many information processing can be traced back to search the extreme value of a database or a cost function. Unfortunately, if the database is unsorted or, equivalently, the cost function has many local minimum/maximum points, then classical solutions cause very high computational complexity. A rather useful extension of Grover’s algorithm is to find the minimum/maximum point of an unsorted database or a cost function. From [26], Durr and Hoyer proposed the first statistical method and bound to solve the problem. Later, based on this result, from [27] the bounds were improved. The two papers determine the estimation of the expected number of iterations described in [28]. Unfortunately all these algorithms provide the extreme value efficiently in terms of expected value thus no reasonable upper bound for the number of required elementary steps can be given [22, 28]. A special case of quantum counting, the so-called quantum existence testing algorithm was proposed from [29]. It also was offered from [29] how to combine classical binary search with quantum existence testing to design an extreme value searching algorithm for unsorted databases/cost functions.

The DNA-based algorithm [8] mainly includes four phases to solve an instance of the hitting-set problem with an \(n\)-element finite set \(S\) and an \(m\)-subset collection \(C\). The first phase is used to result in \(2^n\) combinations choices (states) encoded by \(2^n\) DNA strands and can be implemented by means of arbitrary superposition of \(n\) quantum bits (\(n\) Hadamard gates operating on \(n\) quantum bits). The second phase is employed to figure out legal hitting sets and to remove illegal hitting sets. It can be implemented by NOT gates and CCNOT gates (the quantum circuit in Figure 4-8). The third phase is applied to compute how many elements each legal hitting set contains. This calculates the number of ones for each legal hitting set. This phase can be also implemented by NOT gates, CNOT gates, and CCNOT gates (the quantum circuit in Figure 4-9). The fourth phase is used to read a minimum-sized hitting set that consists of the minimum elements. It can be implemented by Grover’s algorithm. This indicates from Lemma 4-8 that if Grover’s algorithm is employed to accomplish the readout step in the DNA-based algorithm, the quantum implementation of the DNA-based algorithm is equivalent to the oracle work (in the language of Grover’s algorithm), that is, the target state labeling preceding Grover’s searching steps. Compared to
the DNA-based algorithm [8], the quantum algorithm could really save some resources. It assures having a faster labeling of the target state, which also implies a speedy solution for an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \).

A quantum algorithm has been proposed to solve an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \). As the quantum algorithm makes use of the state superposition and quantum parallelism, it is argued that an instance of the hitting-set problem with an \( n \)-element finite set \( S \) and an \( m \)-subset collection \( C \) could be solved with much reduced difficulty.

ACKNOWLEDGEMENTS

This work is partly supported by the National Natural Science Foundation of China under Grants No. 10774163, partly by the National Fundamental Research Program of China under Grants No. 2006CB921203, and also partly supported by the National Science Foundation of Republic of China under Grants No. 96-2221-E-151-008- and 96-2218-E-151-004-.

REFERENCE


