Quantum Algorithms for Biomolecular Solutions of the Satisfiability Problem on a Quantum Machine

Weng-Long Chang*, Ting-Ting Ren, Jun Luo, Mang Feng, Minyi Guo, *Senior Member, IEEE*, and Kawuu Weicheng Lin

***Abstract*—In this paper, we demonstrate that the logic computation performed by the DNA-based algorithm for solving general cases of the satisfiability problem can be implemented more efficiently by our proposed quantum algorithm on the quantum machine proposed by Deutsch. To test our theory, we carry out a three-quantum bit nuclear magnetic resonance experiment for solving the simplest satisfiability problem.**

***Index Terms*—Molecular algorithms, quantum algorithms, the NP-complete problems, the satisfiability problem.**

I. Introduction

SINCE the publication of Deutsch's [1] and Adleman's [2] seminal articles, various quantum algorithms and DNA-based algorithms have been, respectively, proposed for many computational problems. So far, the most frequently cited quantum algorithms are Shor's algorithms for solving factoring integers and discrete logarithm [3] and Grover's search algorithm [4] for unsorted databases. On the other hand, famous DNA-based algorithms are used to solve factoring integers [5] and the set-partition problem [6].

II. Quantum Algorithms for Biomolecular Solutions of the Satisfiability Problem

A. All of the Possible Solutions to the Satisfiability Problem

A clause is a formula of the form $u_n \vee u_{n-1} \cdots \vee u_2 \vee u_1$, where each u_k for $1 \le k \le n$ is a Boolean variable or its negation. In general, a satisfiability problem includes a Boolean formula of the form $C_1 \wedge C_2 \cdots \wedge C_m$, where each C_j for $1 \le j \le m$ is a clause. Then, the question is to find values of the variables so that the whole formula has the value 1.

Assume that U is a set of 2^n possible choices and equal to $\{u_n\, u_{n-1} \cdots u_2 u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$ and each element represents one of 2^n combinational states for n Boolean variables. For the sake 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. The jth element in U can be represented as a unique *computational state vector* $|u_1\rangle = \lfloor u_{1,1} \quad u_{1,2} \quad \cdots \quad u_{1,2^n} \rfloor^T_{1\times 2^n}$, where $u_{1,j} = 1\ \forall\ u_{1,h} = 0$ for $1 \le h \ne j \le 2^n$. The corresponding computational state vector for the *first* element, $u_n^0\ u_{n-1}^0 \cdots u_2^0\ u_1^0$, in U is $[1 \quad 0 \quad \cdots \quad 0]^T_{1\times 2^n}$, and the corresponding computational state vector for the *last* element, $u_n^1\ u_{n-1}^1 \cdots u_2^1\ u_1^1$, in U is $[0 \quad 0 \quad \cdots \quad 1]^T_{1\times 2^n}$. For the sake of presentation, we assume that B is the set of the corresponding computational state vectors to the elements in U and $B = \{[1 \quad 0 \quad \cdots \quad 0]^T_{1\times 2^n} \cdots [0 \quad 0 \quad \cdots \quad 1]^T_{1\times 2^n}\}$. Because each component in B is a coordinated vector, we span $B = C^{2^n}$ [1, 3, 4], where C^{2^n} is a Hilbert space. This implies that the set B is an *orthonormal* basis in a Hilbert space.

B. Computational Space of Molecules for the Satisfiability Problem

The following biomolecular operations cited from [2] will be used to construct computational space of molecules for solving the satisfiability problem with m clauses and n Boolean variables.

Definition 2.1: Given a set $U = \{u_n\, u_{n-1} \cdots u_2\, u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$ and a Boolean variable u_j, the biomolecular operation, "Append-Head," appends u_j onto the head of every element in the set U. The formal representation is written as Append-Head $(U,\ u_j) = \{u_j\ u_n\ u_{n-1} \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n$ and $u_j \in \{0,\ 1\}\}$.

Definition 2.2: Given a set $U = \{u_n\, u_{n-1} \cdots u_2\, u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$ and a Boolean variable u_j, the biomolecular operation, "Append-Tail," appends u_j onto the end of every element in the set U. The formal representation is written as Append-Tail$(U,\ u_j) = \{u_n\ u_{n-1} \cdots u_2\ u_1\ u_j | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n$ and $u_j \in \{0,\ 1\}\}$.

Definition 2.3: Given a set $U = \{u_n\, u_{n-1} \cdots u_2\, u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$, the biomolecular operation, "Discard (U)" sets U to be an empty set.

Definition 2.4: Given a set $U = \{u_n\, u_{n-1} \cdots u_2\, u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$, the biomolecular operation "Amplify $(U, \{U_i\})$" creates a number of identical copies, U_i, of the set U, and then discard(U).

Definition 2.5: Given a set $U = \{u_n\, u_{n-1} \cdots u_2\, u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$ and a Boolean variable, u_j, if the value of u_j is equal to one, then the biomolecular *extract* operation creates two new sets, $+(U,\ u_j^1) = \{u_n\ u_{n-1}\ \cdots\ u_j^1\ \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \ne j \le n\}$

Manuscript received April 23, 2007; revised May 13, 2008. Current version published August 29, 2008. This work was supported in part by the National Science Foundation of R.O.C. under Grant 96-2221-E-151-008 and Grant 96-2218-E-151-004, and in part by the National Natural Science Foundation of China under Grant 10774163 and Grant 2006CB921203. *Asterisk indicates corresponding author.*

*W.-L. Chang is with the Department of Computer Science and Information Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung City 80778, Taiwan, R.O.C. (e-mail: changwl@cc.kuas.edu.tw).

K. W. Lin is with the Department of Computer Science and Information Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung City 80778, Taiwan, R.O.C. (e-mail: linwc@cc.kuas.edu.tw).

T.-T. Ren, J. Luo, and M. Feng are with the State Key Laboratory of Magnetic Resonance and Atomic and Molecular Physics, Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan 430071, China (e-mail: ttren@wipm.ac.cn; jluo@wipm.ac.cn; mangfeng@wipm.ac.cn).

M. Guo is with the School of Computer Science and Engineering, University of Aizu, Fukushima 965-8580, Japan (e-mail: minyi@u-aizu.ac.jp).

Digital Object Identifier 10.1109/TNB.2008.2002286

and $-(U,\ u_j^1) = \{u_n\ u_{n-1}\ \cdots\ u_j^0\ \cdots\ u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \leq k \neq j \leq n\}$. Otherwise, it produces another two new sets, $+(U,\ u_j^0) = \{u_n\ u_{n-1} \cdots u_j^0 \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \leq k \neq j \leq n\}$ and $-(U,\ u_j^0) = \{u_n\ u_{n-1} \cdots u_j^1 \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \leq k \neq j \leq n\}$.

Definition 2.6: Given m sets $U_1 \cdots U_m$, the biomolecular *merge* operation, $\cup(U_1, \ldots, U_m) = U_1 \cup \cdots \cup U_m$.

Definition 2.7: Given a set $U = \{u_n\ u_{n-1} \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \leq k \leq n\}$, the biomolecular operation "Detect (U)" returns a *true* if $U \neq \emptyset$. Otherwise, it returns a *false*.

Definition 2.8: Given a set $U = \{u_n\ u_{n-1} \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \leq k \leq n\}$, the biomolecular operation "Read(U)" performs an arbitrary element in U. Even if U contains many different elements, the biomolecular operation can give an explicit description of exactly one of them.

For solving the satisfiability problem with m clauses and n Boolean variables, the following biomolecular algorithm can be used to create all of the 2^n possible choices. A set U is an empty set and is regarded as the input set of the DNA-based algorithm. The second parameter n in *CombinationalStates*(U, n) is to represent the number of Boolean variables.

```
Procedure CombinationalStates($U$, $n$)
(0a)Append-Tail($U_1,\ u_n^1$).
(0b) Append-Tail($U_2,\ u_n^0$).
(0c) $U = \cup(U_1,\ U_2)$.
(1) For $k = n - 1$ downto 1
        (1a) Amplify($U, U_1, U_2$).
        (1b) Append-Tail($U_1,\ u_k^1$).
        (1c) Append-Tail($U_2,\ u_k^0$).
        (1d) $U = \cup(U_1,\ U_2)$.
    End For
End Procedure
```

Lemma 2.1: For solving the satisfiability problem with m clauses and n Boolean variables, 2^n possible choices created from the DNA-based algorithm, *CombinationalStates*(U, n), form an orthonormal basis of a Hilbert space (i.e., a complex vector space, C^{2^n}).

C. Computational Space of Quantum Mechanical Solution for the Satisfiability Problem

A quantum bit (*qubit*) has two "computational basis vectors" $|0\rangle$ and $|1\rangle$ of the 2-D Hilbert space corresponding to the classical bit values 0 and 1 [1], [3], [4], and an arbitrary state $|\varphi\rangle$ of a qubit is a linearly weighted combination of the computational basis vectors (2.1): $|\varphi\rangle = l_1 \cdot |0\rangle + l_2 \cdot |1\rangle$, where the weighted factors l_1 and $l_2 \in C^2$ are the so-called probability amplitudes, with $|l_1|^2 + |l_2|^2 = 1$. A collection of n qubits is called a quantum register (*qregister*) of size n. It may include any of the 2^n-D computational basis vectors, n qubits of size, or arbitrary superposition of these vectors [1], [3], [4].

D. Lipton's DNA-Based Algorithms for Solving the Satisfiability Problem

Lipton's DNA-based algorithm [7] for solving the satisfiability problem is described next. The symbol $|C_j|$ in the following algorithm is applied to represent the number of Boolean variables and their negations in the jth clause in a formula.

Algorithm 2.1: Lipton's DNA-based algorithm for solving the satisfiability problem.

```
(1) CombinationalStates($U, n$)
(2) For $j = 1$ to $m$ do begin
(3)     For $i = 1$ to $|C_j|$ do begin
(4)         If the $i$th element in the $j$th clause is one of $n$ Boolean
                variables $u_k$, Then
(5)             $U_i = +(U,\ u_k^1)$ and $U = -(U,\ u_k^1)$
(6)         Else
(7)             $U_i = +(U,\ u_k^0)$ and $U = -(U,\ u_k^0)$
(8)         End If
(9) End For
(10) Discard($U$)
(11) For $i = 1$ to $|C_j|$ do begin
(12) $U = \cup(U,\ U_i)$
(13) End For
(14) End For
(15) If (Detect($U$) = = true) Then
        (15a) Read($U$)
        End If
(16) End Algorithm
```

Lemma 2.2: Algorithm 2.1, Lipton's DNA-based algorithm, can be applied to solve the satisfiability problem with m clauses and n Boolean variables.

E. Introduction of Quantum Gates for Solving the Satisfiability Problem

The time evolution of the states of quantum registers can be modeled by means of unitary operators that are often referred to as quantum gates [1], [3], [4]. Therefore, a quantum gate can be regarded as an elementary quantum-computing device that performs a fixed unitary operation on selected qubits during a fixed period of time. The NOT gate is a one-qubit gate and sets the only (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 one. The CCNOT (*controlled-controlled*-NOT) gate is a three-qubit gate and flips the third qubit (the target qubit) if and only if the first qubit and second qubit (the two control qubits) are both one.

F. Constructing Quantum Networks for Solving the Satisfiability Problem

The operations OR and AND are implemented by quantum circuits in Figs. 1 and 2, respectively. For evaluating a clause with the form $u_n \vee u_{n-1} \cdots \vee u_2 \vee u_1$, three quantum registers $|u_n \cdots u_1\rangle$, $|y_n \cdots y_1\rangle$, and $|r_n r_{n-1} \cdots r_1 r_0\rangle$ are needed. Therefore, its evaluating computationis equal to (2.2)

$$|r_n r_{n-1} \cdots r_1\rangle|r_0\rangle|y_n \cdots y_1\rangle|u_n \cdots u_1\rangle \rightarrow |(r_n \oplus \bar{y}_n \bullet \bar{r}_{n-1}) \cdots (r_1 \oplus \bar{y}_1 \bullet \bar{r}_0)\rangle|r_0\rangle|y_n \cdots y_1\rangle|u_n \cdots u_1\rangle$$

where $\bullet$ denotes operation AND of their negations of two Boolean variables $\{\bar{y}_k, \bar{r}_{k-1}\}$ for $1 \leq k \leq n$. The first bit, r_0, in the third

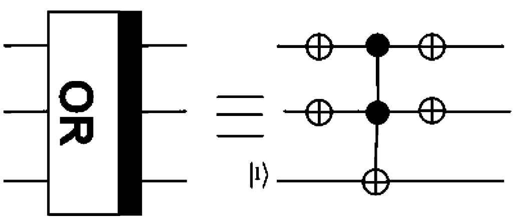


Fig. 1. OR operation of two Boolean variables.

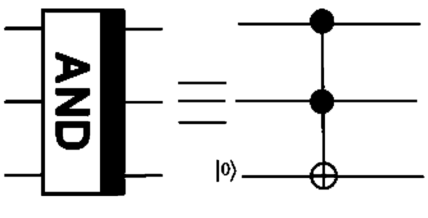


Fig. 2. AND operation of two Boolean variables.

quantum register is initially prepared in state $|0\rangle$, and the other n bits in the third quantum register are initially prepared in state $|1\rangle$. The $(n+1)$th quantum bit, r_n, in the third register is employed to store the final result of the evaluating computation.

Then, in order to evaluate the AND operation of the previous clause and the current clause, the fourth quantum register $|c_m c_{m-1} \cdots c_1 c_0\rangle$ is needed. The first bit, c_0, in the fourth quantum register is initially prepared in state $|1\rangle$, and the other m bits in the quantum register are initially in state $|0\rangle$. The $(m+1)$th quantum bit, c_m, in the fourth register is employed to store the final result of evaluating computation for all of the clauses. The full network, quantum evaluating circuit (*QEC*, for checking whether the current clause is true or not), is illustrated in Fig. 3 and can be understood as follows:

1) We compute the $(n+1)$th bit of the third quantum register to the final result of evaluating a clause. This step requires computing all the OR operations through the relation $r_k \leftarrow r_k \oplus (\bar{y}_k \bullet \bar{r}_{k-1})$ for $1 \le k \le n$. Then, we compute the AND operation of the previous clause and the current clause through the relation $c_j \leftarrow c_j \oplus (c_{j-1} \bullet r_n)$ for $1 \le j \le m$.
2) Subsequently, we reverse all those OR operations in order to restore every quantum bit of each quantum register to its initial state. This enables us to reuse the same quantum registers, should the problem, for example, require repeated OR operation.

Subsequently, we reverse all those operations (NOT or CNOT) on the second quantum register to restore every quantum bit of the second quantum register to its initial state. This enables us to reuse the same second quantum register to evaluate the next clause.

G. Quantum Algorithms of Lipton's DNA-Based Algorithms for Solving the Satisfiability Problem

Based on *Algorithm 2.1* in Section II-D, the following quantum algorithm is proposed to work on the physical quantum computer proposed by Deutsch [1]. For convenience of our following presentation, we suppose that c_j^1 denotes the value of c_j to be 1 and c_j^0 defines the value of c_j to be 0 for $1 \le j \le m$. We also assume that r_k^1 denotes the value of r_k to be 1 and r_k^0 defines the value of r_k to be 0 for $0 \le k \le n$. Similarly, y_k^1 denotes the value of y_k to be 1 and y_k^0 defines the value of y_k to be 0. Moreover, the notations used in *Algorithm 2.2* next have been denoted in previous sections.

Algorithm 2.2: Quantum algorithms of Lipton's DNA-based algorithm for solving the satisfiability problem with m clauses and n Boolean variables.

```
(1) For an input $|\Phi\rangle = (\otimes_{s=m}^{1} |c_s^0\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{q=n}^{1} |y_q^0\rangle) \otimes (\otimes_{q=n}^{1} |u_q^0\rangle)$, $2^n$ possible choices of $n$ bits (including all of the possible choices) are $|\varphi_{1,0}\rangle = (\otimes_{s=m}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes H^{\otimes} n |\Phi\rangle = \frac{1}{\sqrt{2^n}} (\otimes_{s=m}^{1} |c_s^0\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{q=n}^{1} |y_q^0\rangle) \otimes (\otimes_{q=n}^{1} (|u_q^0\rangle + |u_q^1\rangle))$.
(2) **For** $j = 1$ **to** $m$ **do begin**
(3)     **For** $i = 1$ **to** $|C_j|$ **do begin**
(4)         **If** the $i$th element in the $j$th clause is one of $n$ Boolean variables $u_k$, **Then**
(5) $|\varphi_{j,i}\rangle = (\otimes_{s=m}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{p=n}^{k+1} I_{2\times2}) \otimes (|y_k^0 \oplus (u_k^0 + u_k^1)\rangle) \otimes (\otimes_{p=k-1}^{1} I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) |\varphi_{j,i-1}\rangle = \frac{1}{\sqrt{2^n}} (\otimes_{s=m}^{j} |c_s^0\rangle) \otimes (\otimes_{s=j-1}^{1} |c_s\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{p=n}^{k+1} |y_p^0\rangle) \otimes (|y_k^0 + y_k^1\rangle) \otimes (\otimes_{p=k-1}^{1} |y_p^0\rangle) \otimes (\otimes_{q=n}^{1} (|u_q^0\rangle + |u_q^1\rangle))$, where $c_s = c_s \oplus (c_{s-1} \bullet r_n)$ for $j-1 \ge s \ge 1$.
(6)         **Else**
(7) $|\varphi_{j,i}\rangle = (\otimes_{s=m}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{p=n}^{k+1} I_{2\times2}) \otimes (|((\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix})_{2\times2} y_k^0) \oplus (u_k^0 + u_k^1)\rangle) \otimes (\otimes_{p=k-1}^{1} I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) |\varphi_{j,i-1}\rangle = \frac{1}{\sqrt{2^n}} (\otimes_{s=m}^{j} |c_s^0\rangle) \otimes (\otimes_{s=j-1}^{1} |c_s\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{p=n}^{k+1} |y_p^0\rangle) \otimes (|y_k^1 + y_k^0\rangle) \otimes (\otimes_{p=k-1}^{1} |y_p^0\rangle) \otimes (\otimes_{q=n}^{1} (|u_q^0\rangle + |u_q^1\rangle))$, where $c_s = c_s \oplus (c_{s-1} \bullet r_n)$ for $j-1 \ge s \ge 1$.
(8)         **End If**
(9)     **End For**
(10) $|\varphi_{j+1,-1}\rangle = \mathbf{QEC} \otimes (\otimes_{q=n}^{1} I_{2\times2}) |\varphi_{j,|C_j|}\rangle = \frac{1}{\sqrt{2^n}} (\otimes_{s=m}^{j+1} |c_s^0\rangle) \otimes (|c_j \oplus (c_{j-1} \bullet r_n)\rangle) \otimes (\otimes_{s=j-1}^{1} |c_s\rangle \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) (\otimes_{q=n}^{1} |Y_q\rangle) \otimes (\otimes_{q=n}^{1} (|u_q^0\rangle + |u_q^1\rangle))$, where **QEC** is for checking whether the current clause is true or not in Fig. 3, $c_s = c_s \oplus (c_{s-1} \bullet r_n)$ for $j-1 \ge s \ge 1$, and
```

$$Y_q = \begin{cases} y_q^0 + y_q^1 & \text{if it is a Boolean variable} \\ y_q^1 + y_q^0 & \text{if it is a negation.} \end{cases}$$

```
(11) $|\varphi_{j+1,0}\rangle = (\otimes_{s=m}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes (I_{2\times2}) \otimes (\otimes_{q=n}^{1} V_q) \otimes (\otimes_{q=n}^{1} I_{2\times2}) \otimes |\varphi_{j+1,-1}\rangle = \frac{1}{\sqrt{2^n}} (\otimes_{s=m}^{j+1} |c_s^0\rangle) \otimes (\otimes_{s=j}^{1} |c_s\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1} |r_q^1\rangle) \otimes (|r_0^0\rangle) (\otimes_{q=n}^{1} |y_q^0\rangle) \otimes (\otimes_{q=n}^{1} (|u_q^0\rangle + |u_q^1\rangle))$, where $c_s = c_s \oplus (c_{s-1} \bullet r_n)$ for $j \ge s \ge 1$ and
```

$$V_q = \begin{cases} I_{2\times2} \text{ : no appearance in the clause} \\ Y_q \oplus (u_q^0 + u_q^1) \text{ : a Boolean variable} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}_{2\times2} (Y_q \oplus (u_q^0 + u_q^1)) \text{ : its negation.} \end{cases}$$

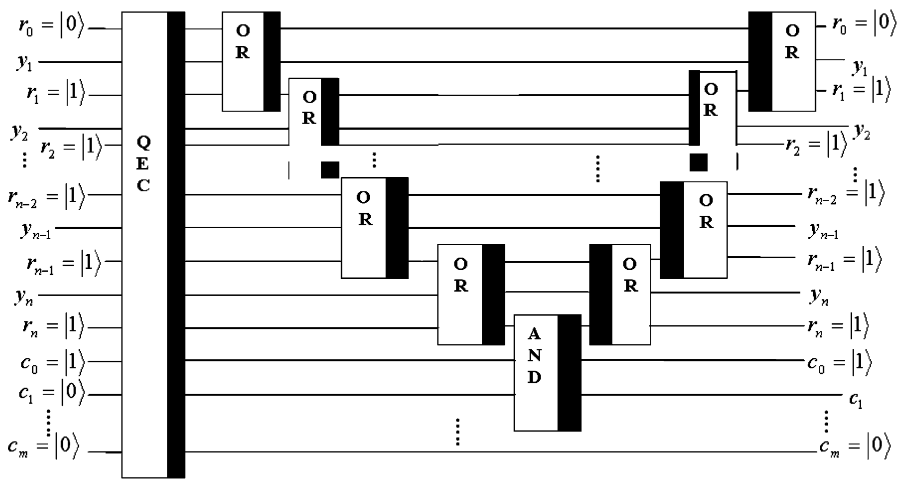


Fig. 3. Full network is the QEC to check whether the current clause is true or not.

(12) **End For**
(13) We obtain the final result, $|\varphi_{m+1,1}\rangle$, after a measurement of $|\varphi_{m+1,0}\rangle$.

End Algorithm

Lemma 2.3: Algorithm 2.2 is a quantum treatment for Lipton's DNA-based algorithm, which can be used to solve the satisfiability problem with m clauses and n Boolean variables.

Proof:

Because there are 2^n possible choices for the satisfiability problem with m clauses and n Boolean variables, a quantum register of n bits $(\otimes_{q=n}^{1}|u_k\rangle)$ is employed to represent 2^n choices with initial state $(\otimes_{q=n}^{1}|u_q^0\rangle)$. The satisfiability problem with m clauses and n Boolean variables is to evaluate whether the whole formula has the value 1, so three auxiliary quantum registers are needed. The initial states of these auxiliary quantum registers are $(\otimes_{q=n}^{1}|y_q^0\rangle)$, $(\otimes_{q=n}^{1}|r_q^1\rangle) \otimes (|r_0^0\rangle)$ and $(\otimes_{s=m}^{1}|c_m^0\rangle) \otimes (|c_0^1\rangle)$, respectively. Therefore, the total Hilbert space is $(\otimes_{s=m}^{1}\boldsymbol{C}^2) \otimes \boldsymbol{C}^2 \otimes (\otimes_{q=n}^{1}\boldsymbol{C}^2) \otimes \boldsymbol{C}^2 \otimes (\otimes_{q=n}^{1}\boldsymbol{C}^2) \otimes (\otimes_{q=n}^{1}\boldsymbol{C}^2)$.

From the execution of Step (1), an initial vector $|\Phi\rangle = (\otimes_{s=m}^{1}|c_m^0\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1}|r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{q=n}^{1}|y_q^0\rangle) \otimes (\otimes_{q=n}^{1}|u_q^0\rangle)$ starts the quantum computation of the satisfiability problem. $H^{\otimes n}$ that stands for the joined n-qubit Hadamard gate is applied to the part of choices of the initial vector $|\Phi\rangle$. Then, the resulting state vector becomes $|\varphi_{1,0}\rangle = \frac{1}{\sqrt{2^n}}(\otimes_{s=m}^{1}|c_m^0\rangle) \otimes (|c_0^1\rangle) \otimes (\otimes_{q=n}^{1}|r_q^1\rangle) \otimes (|r_0^0\rangle) \otimes (\otimes_{q=n}^{1}|y_q^0\rangle) \otimes (\otimes_{q=n}^{1}(|u_q^0\rangle + |u_q^1\rangle))$.

Step (2) is the outer loop of the main loop and is to judge whether each clause in m clauses is true or not. Step (3) is the first inner loop of the main loop and is used to test whether the jth clause with $1 \le j \le m$ is true or not. On each execution of Step (4), if the ith element in the jth clause is one of n Boolean variables u_k, then this implies that those choices with the value 1 of u_k satisfy the jth clause, and thus, *the operation* CNOT (i.*e*., $|y_k^0 \oplus (u_k^0 + u_k^1)\rangle$) corresponds to the implementation of Step (5) in *Algorithm 2.1*. Otherwise, from each execution of Step (6), the ith element in the jth clause is its negation and this implies that those choices with the value 0 of u_k satisfy the jth clause, and hence, *the operation* NOT and CNOT

$$(\text{i .e.}, |((\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix})_{2\times 2} y_k^0) \oplus (u_k^0 + u_k^1)\rangle)$$

corresponds to the implementation of Step (7) in *Algorithm 2.1*. After repeating the Steps (4)–(7), we have $|\varphi_{j,|C_j|}\rangle$, including those choices that satisfy the jth clause and those illegal choices that dissatisfy the jth clause.

Next, each execution of Step (10) works as the unitary operator, **QEC**, i.e., the quantum circuit in Fig. 3 to judge whether the jth clause is true or not and whether both the $(j-1)$th clause and the jth clause are true or not. On each execution of Step (11), the initial state of the first auxiliary quantum register is reset to be $(\otimes_{q=n}^{1}|y_q^0\rangle)$. This implies that Step (10) through Step (13) in *Algorithm 2.1* have been done. After repeating each operation embedded in the main loop, we get the resulting state vector $|\varphi_{m+1,0}\rangle$, including those choices that satisfy the whole formula and those illegal choices that dissatisfy the whole formula.

Finally, based on $|c_m^1\rangle$, the execution of Step (13) is for a measurement to find the answer to the satisfiability problem with m clauses and n Boolean variables. Therefore, *Algorithm 2.2*, i.e., the quantum algorithm of Lipton's DNA-based algorithm, can also be used to solve the satisfiability problem with m clauses and n Boolean variables. ■

III. Quantum Implementation of Biomolecular Computing on a Quantum Machine

The following lemmas are introduced to show that the biomolecular computer proposed by Adleman [2] can be implemented on the quantum machine proposed by Deutsch [1].

Lemma 3.1: For any NP-complete problem, 2^n possible solutions on the biomolecular computer proposed by Adleman [2] can be implemented on the quantum machine proposed by Deutsch [6], where n is the number of bits for input size of the problem.

Proof:

From *Lemma 2.1*, 2^n possible solutions, $U = \{u_n\ u_{n-1} \cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$, for any NP-complete problem are created from the DNA-based algorithm, *CombinationalStates*(U, n). From *Lemma 2.1*, 2^n possible solutions form an orthonormal basis of a Hilbert space (a complex vector space, C^{2^n}). Similarly, 2^n possible quantum solutions for the same problem are

$$|\phi\rangle = H^{\otimes n}|00\cdots 0\rangle = \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle$$

and also form an orthonormal basis of a Hilbert space (a complex vector space, C^{2^n}). Therefore, from the previous statements, we may conclude that for any NP-complete problem, 2^n possible solutions on the biomolecular computer proposed by Adleman [2] can be implemented on the quantum machine proposed by Deutsch [1]. ■

Lemma 3.2: The biomolecular operation, Append-Head $(U,\ u_j) = \{u_j\ u_n\ u_{n-1}\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n$ and $u_j \in \{0,\ 1\}\}$, denoted in *Definition 2.1* can be implemented by the corresponding quantum operator $|\Phi\rangle = |0\rangle \otimes (H^{\otimes n}|000\cdots 0\rangle) = |0\rangle \otimes (\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle)$ or $|\Phi\rangle = |1\rangle \otimes (H^{\otimes n}|000\cdots 0\rangle) = |1\rangle \otimes (\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle)$.

Proof:

From *Definition 2.1*, for 2^n possible solutions, $U = \{u_n\ u_{n-1}\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$, and a Boolean variable u_j, the biomolecular operation, "Append-Head," appends u_j onto the head of the elements in the set U, that is, Append-Head$(U,\ u_j) = \{u_j\ u_n\ u_{n-1}\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n$ and $u_j \in \{0,\ 1\}\}$. If the value for u_j is equal to 0, then from *Lemma 3.1*, the action of the biomolecular operation "Append-Head" can be implemented by

$$|\Phi\rangle = |0\rangle \otimes (H^{\otimes n}|000\cdots 0\rangle) = |0\rangle \otimes \left(\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle\right).$$

Otherwise, the action of the biomolecular operation "Append-Head" can be implemented by

$$|\Phi\rangle = |1\rangle \otimes (H^{\otimes n}|000\cdots 0\rangle) = |1\rangle \otimes \left(\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle\right).$$

■

Lemma 3.3: The biomolecular operation, Append-Tail $(U,\ u_j) = \{u_n\ u_{n-1}\cdots u_2\ u_1\ u_j | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n$ and $u_j \in \{0,\ 1\}\}$, denoted in *Definition 2.2* can be implemented by $|\Phi\rangle = (H^{\otimes n}|000\cdots 0\rangle) \otimes |0\rangle = (\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle) \otimes |0\rangle$ or $|\Phi\rangle = (H^{\otimes n}|000\cdots 0\rangle) \otimes |1\rangle = (\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle) \otimes |1\rangle$.

Proof: Refer to *Lemma 3.2*. ■

Lemma 3.4: The biomolecular *extract* operation, $+(U, u_j^1) = \{u_n\ u_{n-1}\ \cdots\ u_j^1\ \cdots\ u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \ne j \le n\}$ and $-(U,\ u_j^1) = \{u_n\ u_{n-1}\cdots u_j^0\ \cdots\ u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \ne j \le n\}$, denoted in *Definition 2.5* can be implemented by CNOT.

Proof: Refer to *Algorithm 2.2*. ■

Lemma 3.5: The biomolecular *extract* operation, $+(U, u_j^0) = \{u_n\ u_{n-1}\ \cdots\ u_j^0\ \cdots\ u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \ne j \le n\}$ and $-(U,\ u_j^0) = \{u_n\ u_{n-1}\cdots u_j^1\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \ne j \le n\}$, denoted in *Definition 2.5* can be implemented by NOT and CNOT.

Proof: Refer to *Algorithm 2.2*. ■

Lemma 3.6: Assume that $U = \{u_n\ u_{n-1}\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$ and W is a subset of U and U_a is also a subset of U for $1 \le a \le m$. From *Definitions 2.3* and *2.6*, the biomolecular *merge* operation and *discard* operation, i.e., $\cup(U_1, \ldots, U_m) = U_1 \cup \cdots \cup U_m$ and Discard$(W) = \oslash$, perform logic computation of the satisfiability problem that can be implemented by NOT, CNOT, and CCNOT.

Proof: Refer to *Algorithm 2.2*. ■

Lemma 3.7: Assume that $U = \{u_n\ u_{n-1}\cdots u_2\ u_1 | \forall u_k \in \{0,\ 1\}$ for $1 \le k \le n\}$. From *Definitions 2.7* and *2.8*, the biomolecular *detect* operation and *read* operation, i.e., Detect(U) and Read(U), to find the answer to the satisfiability problem can be implemented by means of measurements.

Proof: Refer to *Algorithm 2.2*. ■

Theorem 3.1: The biomolecular computer proposed by Adleman [2] is a *subset* of the quantum machine proposed by Deutsch [1] and can be implemented on the quantum machine.

Proof:

From *Lemmas 3.1* through *3.7*, it is very clear that biomolecular computational space for any NP-complete problem can be constructed on a quantum machine, and the biomolecular operations proposed by Adleman [2] can also be implemented by quantum gates (NOT, CNOT, and CCNOT). Therefore, it is inferred that the biomolecular computer proposed by Adleman [2] is a subset of the quantum machine proposed by Deutsch [1] and can be implemented on the quantum machine. ■

Theorem 3.2: For those famous NP-complete problems that had been solved on a biomolecular computer, their corresponding DNA-based algorithms can be fully implemented on a quantum machine.

Proof:

From *Theorem 3.1*, it is very obvious that a biomolecular computer can be implemented on a quantum machine. Therefore, it is derived at once that the DNA-based algorithms for solving those famous NP-complete problems can all be implemented on a quantum machine. ■

IV. Complexity Assessment

The following lemmas could be used to show the time complexity and space complexity of *Algorithm 2.2* for solving the satisfiability problem with m clauses and n Boolean variables

Lemma 4.1: Time complexity for solving the satisfiability problem with m clauses and n Boolean variables is O(n) Hadamard gates, O($10 \times m \times n$) NOT gates, O($2 \times m \times n$) CNOT gates, O($2 \times m \times n + m$) CCNOT gates, and O(1) projective operators.

Proof:

In Step (1) of *Algorithm 2.2*, n Hadamard gates are performed. Next, Steps (5) and (7) of *Algorithm 2.2* are embedded in the main loop, corresponding to the implementation of at most $(m \times n)$ NOT gates and $(m \times n)$ CNOT gates. From Step (10) of *Algorithm 2.2*, we have $(8 \times m \times n)$

NOT gates and $(2 \times m \times n + m)$ CCNOT gates performed and Step (11) leads to at most $(m \times n)$ NOT gates and $(m \times n)$ CNOT gates. Finally, Step (13) of *Algorithm 2.2* is a one projective operator. So, time complexity is O(n) Hadamard gates, O($10 \times m \times n$) NOT gates, O($2 \times m \times n$) CNOT gates, O($2 \times m \times n + m$) CCNOT gates, and O(1) projective operators. ■

Lemma 4.2: Space complexity of solving the satisfiability problem with m clauses and n Boolean variables is O($m + 3 \times n + 2$) quantum bits.

Proof:

From Step (1) in *Algorithm 2.2*, it is indicated that the total Hilbert space is $(\otimes_{s=m}^{1} C^2) \otimes C^2 \otimes (\otimes_{q=n}^{1} C^2) \otimes C^2 \otimes (\otimes_{q=n}^{1} C^2) \otimes (\otimes_{q=n}^{1} C^2)$. So, the space complexity is O($m + 3 \times n + 2$) quantum bits. ■

V. Example of Three-Qubit Solution for the Satisfiability Problem

Consider the formula: $F = (u_1)$, the simplest case of the satisfiability problem. It contains one clause "(u_1)," and one Boolean variable u_1. The following algorithm, i.e., *Algorithm 5.1*, is the reduced version of *Algorithm 2.2* in Section II-F and is employed to find the answer to the satisfiability problem of the clause, $F = (u_1)$.

Algorithm 5.1: Solving the satisfiability problem of the clause, $F = (u_1)$.

(1) For an input $|\Phi\rangle = |c_1^0\rangle \otimes |y_1^0\rangle \otimes |u_1^0\rangle$, two choices are $|\varphi_{1,0}\rangle = (I_{2\times 2}) \otimes (I_{2\times 2}) \otimes (H)\,|\Phi\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle) \otimes (|y_1^0\rangle) \otimes (|u_1^0\rangle + |u_1^1\rangle)$.

(2) $|\varphi_{1,1}\rangle = (I_{2\times 2}) \otimes (|y_1^0 \oplus (u_1^0 + u_1^1)\rangle) \otimes (I_{2\times 2})\,|\varphi_{1,0}\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle) \otimes (|y_1^0 + y_1^1\rangle) \otimes (|u_1^0\rangle + |u_1^1\rangle)$.

(3) $|\varphi_{2,-1}\rangle = (|c_1^0 \oplus (y_1^0 + y_1^1)\rangle) \otimes (I_{2\times 2}) \otimes (I_{2\times 2})\,|\varphi_{1,1}\rangle = \frac{1}{\sqrt{2}}(|(c_1^0 + c_1^1)\rangle) \otimes (|y_1^0 + y_1^1\rangle) \otimes (|u_1^0\rangle + |u_1^1\rangle)$.

(4) $|\varphi_{2,0}\rangle = (I_{2\times 2}) \otimes (|(y_1^0 \oplus u_1^0) + (y_1^1 \oplus u_1^1)\rangle) \otimes (I_{2\times 2})\,|\varphi_{2,-1}\rangle = \frac{1}{\sqrt{2}}(|(c_1^0 + c_1^1)\rangle) \otimes (|y_1^0\rangle) \otimes (|u_1^0\rangle + |u_1^1\rangle)$.

(5) We obtain the final result, $|\varphi_{2,1}\rangle$, after a measurement of $|\varphi_{2,0}\rangle$.

End Algorithm

So, from Step (1) of *Algorithm 5.1*, for an input $|\Phi\rangle = |c_1^0\rangle \otimes |y_1^0\rangle \otimes |u_1^0\rangle$, 2^1 possible choices are $|\varphi_{1,0}\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle\,|y_1^0\rangle\,|u_1^0\rangle + |c_1^0\rangle\,|y_1^0\rangle\,|u_1^1\rangle)$. Then, in the first clause, the first Boolean variable is u_1. Therefore, after Step (2) of *Algorithm 5.1* is performed, we have $|\varphi_{1,1}\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle\,|y_1^0\rangle\,|u_1^0\rangle + |c_1^0\rangle\,|y_1^1\rangle\,|u_1^1\rangle)$. Because only a clause and a Boolean variable are involved in the example, we could use the CNOT gate to replace *QEC* in Fig. 3 to evaluate whether the only clause with the only Boolean variable is true or not. Hence, the implementation of Steps (3) and (4) in *Algorithm 5.1* yields $|\varphi_{2,-1}\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle\,|y_1^0\rangle\,|u_1^0\rangle + |c_1^1\rangle\,|y_1^1\rangle\,|u_1^1\rangle)$ and $|\varphi_{2,0}\rangle = \frac{1}{\sqrt{2}}(|c_1^0\rangle\,|y_1^0\rangle\,|u_1^0\rangle + |c_1^1\rangle\,|y_1^0\rangle\,|u_1^1\rangle)$. Step (5) of *Algorithm 5.1* is for a measurement on $|\varphi_{2,0}\rangle$ to find $|c_1^1\rangle$, i.e., the answer to the satisfiability problem for the formula: $F = (u_1)$. The measurement yields $|\varphi_{2,1}\rangle = |c_1^1\rangle\,|y_1^0\rangle\,|u_1^1\rangle$, and the corresponding quantum circuit of the aforesaid example is shown in Fig. 4.

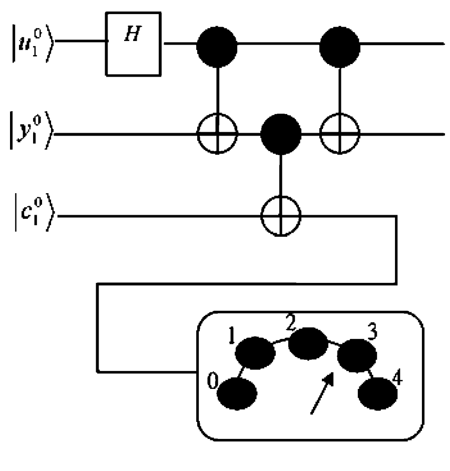


Fig. 4. Quantum circuit of the aforesaid example.

NMR approach has been widely employed for quantum information processing over the past several years due to its mature and well-controllable technology. Although the quantum information processing by NMR is made on ensembles of nuclear spins, instead of individual spins, NMR has remained to be the most convenient experimental tool to demonstrate quantum information processing. We here also adopt this technology to check our theory.

Our experiment is carried out on a Varian INOVA 600 NMR spectrometer. The sample is ^{13}C-*labeled* alanine with formula $_1^{13}$CH$_3$–$_2^{13}$CH(NH$_2$)–$_3^{13}$COOH, where the three carbons $_1^{13}$C, $_2^{13}$C, $_3^{13}$C correspond to the qubits I_1, I_2, I_3, respectively. The J-coupling constants are $J_{12} = 34.79\,\text{Hz}$, $J_{23} = 54.01\,\text{Hz}$, $J_{13} = 1.20\,\text{Hz}$. Selective excitation was achieved by using soft pulses.

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 pseudopure state, must be well prepared. There have been many methods to do this job, among which the spatial averaging method proposed by Cory *et al.* is most commonly used. So, in our experiment, we have also employed this technique to prepare the three-qubit pseudopure state for which the detailed pulse sequence can be found in [8]. The states of the input qubits are prepared by following operation $E + I_{1z} + I_{2z} + I_{3z} + 2I_{1z}I_{2z} + 2I_{2z}I_{3z} + 2I_{1z}I_{3z} + 4I_{1z}I_{2z}I_{3z}$, where E is the unity operator with the form of $E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and $I_{iz} = \frac{1}{2}\sigma_z$, with $i = 1$, 2, and 3, being the ith spin angular momentum operator in the z-direction, and σ_z is the Pauli matrix $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$.

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

$$[\pi/2]_y^2 \to (1/4J) \to [\pi]_x^{1,2} \to (1/4J) \to [\pi]_x^{1,2} \to [\pi/2]_x^2$$

where the flip angle of the pulse and the time of delay are written in square brackets and in round brackets, respectively.

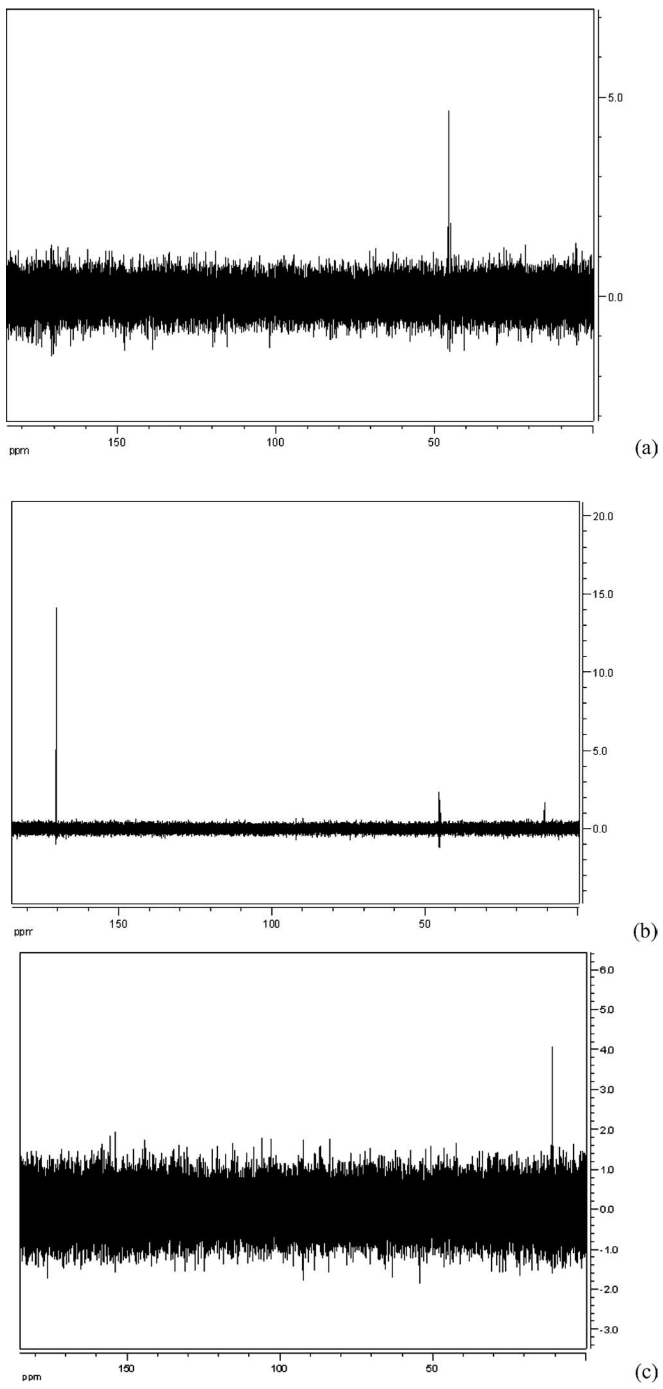


Fig. 5. Experimental spectra (a)–(c) of the three-qubit solution for the satisfiability problem after the readout on the first, second, and third qubits, respectively.

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}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}C after the algorithm was accomplished. In our case, the final state was $(|000\rangle_{123} + |101\rangle_{123})/\sqrt{2} = (|00\rangle_{13} + |11\rangle_{13})|0\rangle_2/\sqrt{2}$ which means the first and the third 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 for detecting the output state $(|000> +|101>)/\sqrt{2}$. We may detect the second qubit directly by applying a $\pi/2$ readout pulse along the x-axis, yielding Fig. 5 (b). But for the first and third qubits, we need to disentangle them before measurement. To this end, we apply a CNOT gate, respectively, on the first and second qubits followed by another CNOT gate, respectively, on the second and first qubits to get the state $(|000> +|011>)/\sqrt{2}$. Then the first qubit can be read out by a single π/2 pulse along the x-axis, as shown in Fig. 5 (a). Similar steps applied to the third qubit result in the spectrum in Fig. 5 (c). It is evident that the experimental results are in good agreement with our theoretical prediction.

Therefore, due to the fact that NMR quantum operations are not made on individual nuclear spins, but on spin ensemble, there is a difference in the operation between Fig. 4 and our experiment. Some remarks must be addressed. First of all, the three-qubit NMR experiment that we have carried out suffices to make a comprehensive test for our theory, because we have achieved the key aspects of our theory. Although the simple case with three qubits did not reflect the efficiency of quantum computation for the SAT problem, we argue that, with more variables and clauses involved, the quantum computing efficiency would be more and more evident, which has been reflected in our previous discussion about the computational complexity. Second, DNA computation does not involve entanglement, whereas entanglement does appear in our quantum treatment. The necessity of additional operations to disentangle the output qubits is not the intrinsic characteristic of our quantum mechanical treatment, but due to the unique feature of the NMR technique. Anyway, those additional operations have not changed the essence of our implementation.

VI. Conclusion

By using the mature technique of NMR, we have carried out a solution for the simplest satisfiability problem. The experimental results are in good agreement with the theoreticalprediction.

References

[1] D. Deutsch, “Quantum theory, the Church–Turing principle and the universal quantum computer,” in *Proc. Roy. Soc. Lond. Ser. A*, 1985, vol. 400, pp. 97–117.
[2] L. Adleman, “Molecular computation of solutions to combinatorial problems,” *Science*, vol. 266, pp. 1021–1024, 1994.
[3] P. W. Shor, “Algorithm for quantum computation: Discrete logarithm and factoring algorithm,” in *Proc. 35th Annu. IEEE Symp. Found. Comput. Sci.*, 1994, pp. 124–134.

[4] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. Twenty-Eighth Annu. ACM Symp. Theory Comput.*, 1996, pp. 212–219.

[5] W.-L. Chang, M. Ho, and M. Guo, "Fast parallel molecular algorithms for DNA-based computation: factoring integers," *IEEE Trans. Nanobiosci.*, vol. 4, no. 2, pp. 149–163, Jun. 2005.

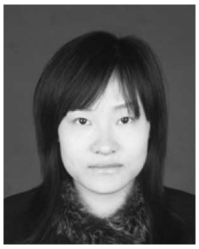
[6] W.-L. Chang, "Fast parallel DNA-based algorithms for molecular computation: The set-partition problem," *IEEE Trans. Nanobiosci.*, vol. 6, no. 1, pp. 346–353, Dec. 2007.

[7] R. Lipton, "DNA solution of hard computational problems," *Science*, vol. 268, pp. 542–545, 1995.

[8] D. G. Cory, M. D. Price, and T. F. Havel, "Nuclear magnetic resonance spectroscopy: An experimentally accessible paradigm for quantum computing," *Phyisca D*, vol. 120, no. 1–2, pp. 82–101, 1998.

Weng-Long Chang received the Ph.D. degree in computer science and information engineering from the National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1999.

He is currently an Associated Professor at the National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan. His current research interests include DNA-based algorithms, quantum algorithms, and languages and compilers for parallel computing.



Ting-Ting Ren is currently working toward the Ph.D. degree at Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan, China.

Her current research interests include quantum computation, quantum algorithm, and nuclear magnetic resonance (NMR) techniques.

Jun Luo received the Ph.D. degree in physics at Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan, China.

He is currently an Associate Professor of Physics at the State Key Laboratory of Magnetic Resonance and Atomic and Molecular Physics, Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences. His current research interests include experimental fields of nuclear magnetic resonance (NMR) quantum computing and the enhancement of the sensitivity of NMR signals with spin-exchange optical pumping methods.

Mang Feng received the Ph.D. degree in condensed matter physics at the Centre for Fundamental Physics, University of Science and Technology of China, Hefei, China.

He is currently a Full Professor at the Laboratory of Magnetic Resonance and Atomic and Molecular Physics, Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan, China. His current research interests include quantum computation and quantum communication with atoms and solid-state materials, entanglement and decoherence: quantum information processing with decoherence-free subspace, quantum algorithms and their realization, investigation of quantum optics problem, such as quantum treatment of interaction between matter and radiation field (based on Jaynes–Cummings model), and other aspects concerning quantum mechanics and mathematical physics.

Minyi Guo (M'02–SM'07) received the Ph.D. degree in computer science from the University of Tsukuba, Tsukuba, Japan.

Before 2000, he had been a research scientist of NEC Corporation, Japan. He is currently a Professor in the School of Computer Science and Engineering, University of Aizu, Fukushima, Japan. He was also a Visiting Professor of Georgia State University, USA, Hong Kong Polytechnic University, University of Hong Kong, National Sun Yet-Sen University in Taiwan, University of Waterloo, Canada, and University of New South Wales, Australia, and an Adjunct Professor at Shanghai Jiao Tong University, China. He is the Editor-in-Chief of the *Journal of Embedded Systems*. He is also on the Editorial Boards of the *Journal of Pervasive Computing and Communications, International Journal of High Performance Computing and Networking, Journal of Embedded Computing, Journal of Parallel and Distributed Scientific and Engineering Computing, and International Journal of Computer and Applications*. His current research interests include parallel and distributed processing, parallelizing compilers, pervasive computing, embedded software optimization, molecular computing, and software engineering. He is the author or coauthor of more than 150 research papers published in international journals and conferences.

Dr. Guo was the General Chair, and the Program Committee or Organizing Committee Chair for many international conferences. He is the founder of the International Conference on Parallel and Distributed Processing and Applications (ISPA) and the International Conference on Embedded and Ubiquitous Computing (EUC). He is a member of the Association for Computing Machinery (ACM), the Information Processing Society of Japan (IPSJ), and the Institute of Electrical, Information and Communication Engineers (IEICE).

Kawuu Weicheng Lin received the B.S. degree from the Department of Computer Science and Information Engineering, National Taiwan University (NTU), Taipei, Taiwan, R.O.C., in 1999 and the Ph.D. degree from the Department of Computer Science and Information Engineering, National Cheng Kung University (NCKU), Tainan, Taiwan, 2006.

Since August 2007, he has been an Assistant Professor in the Department of Computer Science and Information Engineering, National Kaohsiung University of Applied Sciences (KUAS), Kaohsiung, Taiwan. His current research interests include data mining and its applications, sensor technologies, and parallel and distributed processing.

Dr. Lin is a member of the Phi Tau Phi honorary society, and has won the Phi Tau Phi Scholastic Honor in 2006.