THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 24, Number 3/2023, pp. 275–282
DOI: 10.59277/PRA-SER.A.24.3.09

TWO EXAMPLES OF IMPLEMENTATION OF THE IF/THEN/ELSE CONDITIONAL STATEMENT WITH QUANTUM COMPUTERS

Guang Ping HE

Sun Yat-sen University, School of Physics, Guangzhou 510275, China
E-mail: hegp@mail.sysu.edu.cn

Abstract. While quantum algorithms are known to be very powerful, there are also people who think that "quantum computers will never be able to run the if/then/else type of logic". On the other hand, though many experts know that such a claim is a misconception, in literature there is still lacking of a general description of the procedure for implementing the if/then/else conditional statements in the quantum way. In the current work, we present such a general method in details, explaining how the unitary transformation required for any if/then/else operation can be constructed from simpler unitary transformations and projective operators. Moreover, by using the well-known controlled-NOT gate, Fredkin gate and Toffoli gate as examples, it is elaborated that the structure of the existing conditional quantum operations agrees with our general method. An even more complicated example is also provided, showing that our method is convenient for solving real-world if/then/else problems with quantum computers. Thus the result not only refutes the above claim, but also helps to develop the skill on quantum programming.

***Key words**:* quantum computation, quantum algorithm, conditional statement.

1. INTRODUCTION

The success of Shor's algorithm for factoring large integers [1] and Grover's search algorithm [2] highlighted the potential power of quantum computers. Nowadays, quantum algorithms further find its place in the researches of neural networks and machine learning [3, 4], which contribute to the blooming of artificial intelligence technology. Nevertheless, there are still doubts whether quantum computers can handle all the tasks that classical computers are capable of. In a recent interview [5], Andy Stanford Clark, IBM CTO for UK and Ireland, claimed that "quantum computers will never be able to run the if/then/else type of logic that we're familiar with our traditional Von Neumann architecture computers, (where they are) sequentially going from step to step". But in fact, as it was already shown in Section 3.1.3 of our book [6], all classical if/then/else type of logic can be constructed in the quantum way. Unfortunately, the interview [5] gave no further explanation to the above claim. Thus it is hard to judge whether the "if/then/else type of logic" that A. S. Clark mentioned has a different meaning from what we usually know. But to avoid the confusion spreads, it seems necessary to demonstrate here that quantum computer can indeed implement the if/then/else conditional statement that we are familiar with when programming classical computers. Moreover, though in literature it is well known that many basic quantum gates are actually if/then/else operations [7], and some complicated implementations of the if/then/else conditional statement were introduced as the building blocks for certain quantum algorithms (e.g., Shor's quantum correction code [8]), to our best knowledge, there is still lacking of a concrete description on how to construct such quantum conditional statement from scratch in general. Therefore, it is expected to fill the gap in the current paper.

In the next section, the general method for accomplishing this task will be proposed. Then in Section 3.1, it will be shown that many known quantum gates actually fit this general method. More importantly, in Section 3.2, it will be elaborated how our general method can be applied on a complicated real-world if/then/else problem, which is a new example that has not appeared in our past papers before. Further applications of the result will be discussed briefly in Section 4. Finally, the conclusions of this paper will be summarized in Section 5.

2. THE GENERAL METHOD

As it is well known [9], the basic structure of the if/then/else conditional statement for a series of conditions and operations can be written as:

Classical if/then/else conditional statement:
If (the 1st Boolean condition is satisfied) Then (perform the 1st operation).
Else if (the 2nd Boolean condition is satisfied) Then (perform the 2nd operation).
.....
Else if (the i-th Boolean condition is satisfied) Then (perform the i-th operation).
.....

To construct its quantum correspondence, the general idea is as follows.

Quantum if/then/else conditional statement:
i) Introduce two quantum registers X and Y, where X serves as the control register, while Y is the target register. Note that each register can be a multi-level system, instead of being limited to a single qubit.

ii) Find a projective operator $P_i^{(X)}$ *(i.e., it satisfies the requirement* $\left(P_i^{(X)}\right)^2 = P_i^{(X)}$ *so that it represents a physically implementable measurement) on X, such that the projection will be successful if the i-th Boolean condition is satisfied.*

iii) Find a unitary operator $U_i^{(Y)}$ *(i.e., it satisfies the requirement* $\left(U_i^{(Y)}\right)^+ = U_i^{(Y)}$ *so that it is a physically implementable transformation) on Y for performing the i-th operation.*

iv) Finally, applying on the composite system $X \otimes Y$ *with the operation*

$$U \equiv \sum_i P_i^{(X)} \otimes U_i^{(Y)}. \tag{1}$$

This completes the implementation of the if/then/else conditional statement.

In this process, two things should be taken extra care of:

a) $\left\{P_i^{(X)}\right\}$ (when all possible values of i are taken) should be a complete measurement (i.e., $\sum_i P_i^{(X)} = I^{(X)}$ where $I^{(X)}$ denotes the identity matrix having the same dimension as that of X) on the control register X. That is, when X is an n-level system, there is a specific measurement basis (generally it can be taken as the computational basis) $\left\{|0\rangle, |1\rangle, ..., |n-1\rangle\right\}$, in which $\left\{P_i^{(X)}\right\}$ can be expressed as $\left\{P_i^{(X)} = |i\rangle_X \langle i|,\ \ i = 0, ..., n-1\right\}$. Meanwhile, it is worth emphasizing again that all $U_i^{(Y)}$ ($i = 0, ..., n-1$) need to be unitary.

b) If the task "if (the i-th condition is satisfied) then (perform the i-th operation)" merely gives explicit description of the condition and operation for one or few of the i values instead of all $i = 0, ..., n-1$, then we should provide the description of the condition and operation for all the rest i values ourselves. A simple way of completing these missing description is: when none of the explicitly given conditions is met, the corresponding operation can be taken as the identity operator I (whose matrix form is the identity matrix). That is, no change is made to the target register Y. Especially, in traditional Von Neumann architecture computers where the operations are performed sequentially from step to step, when implementing the "if (*Boolean condition*) then (*consequent*) else (*alternative*)" conditional statement, sometimes the *alternative* operation has not been defined by the time the *consequent* operation is performed. Instead, it is preferred to decide what *alternative* operation to perform only after the *Boolean condition* was evaluated and the result turned out to be false. Then in the quantum case, to maintain the freedom on choosing the *alternative* operation, the *alternative* operation can be taken as the identity operator I temporarily when constructing the operation U in Eq. (1). After U was applied, a measurement can be made on the control register X to see whether the result of the *Boolean condition* is false, then it can be decided whether another *alternative*

operation needs to be performed on the target register *Y*. Or a second unitary operation *U* corresponding to another if/then conditional statement can be constructed, and applied on the composite system $X \otimes Y$ without measuring *X*.

These points are important because once they are satisfied, combining with the fact that

$$(P_{i'}^{(X)})^{+} P_i^{(X)} = |i'\rangle_X \langle i'|i\rangle_X \langle i| = \delta_{ii'} P_i^{(X)}, \tag{2}$$

it can be proven that the operation *U* in Eq. (1) satisfies

$$\begin{aligned} U^{+}U &= \left(\sum_{i'=0}^{n-1} P_{i'}^{(X)} \otimes U_{i'}^{(Y)}\right)^{+} \left(\sum_{i=0}^{n-1} P_i^{(X)} \otimes U_i^{(Y)}\right) \\ &= \sum_{i=0}^{n-1} \left(P_i^{(X)} \otimes \left(U_i^{(Y)}\right)^{+} U_i^{(Y)}\right) = \sum_{i=0}^{n-1} \left(P_i^{(X)} \otimes I^{(Y)}\right) = I^{(X\otimes Y)}, \end{aligned} \tag{3}$$

where the identity matrix $I^{(Y)}$ has the same dimension as that of the target register *Y*, while $I^{(X\otimes Y)}$ has the same dimension as that of $X \otimes Y$. This result guarantees that *U* is unitary. As it is well-known, all unitary operations can be realized physically in principle. Thus, with the advance of technology, they can be implemented sooner or later.

3. EXAMPLES

3.1. Multi-qubit quantum gates

Here it will be shown that some common multi-qubit quantum logic gates themselves already have the form of the if/then/else conditional statement, and their construction agrees with the above process.

3.1.1. Controlled NOT (CNOT) gate [6,7]

CNOT gate is defined as an operation on two registers *A* and *B*, such that if the content of *A* is 0 then leave the content of *B* unchanged, else if the content of *A* is 1 then invert the content of *B* (i.e., change 0 to 1 and 1 to 0). Note that in this case, the content of either one of *A* and *B* is limited to a single bit.

To accomplish the task in quantum computer, the registers *A* and *B* are taken as two qubits. The content 0 (1) is presented as the quantum state $|0\rangle$ ($|1\rangle$). Following the above general method, the condition "if the content of *A* is 0 (or 1)" is implemented by applying on the control register *A* with the projective operator $P_0^{(A)} \equiv |0\rangle_A \langle 0|$ (or $P_1^{(A)} \equiv |1\rangle_A \langle 1|$). Meanwhile, the unitary operator on the target register *B* is taken as $U_0^{(B)} \equiv I^{(B)}$ (or $U_1^{(B)} \equiv \sigma_x^{(B)}$), where $I^{(B)}$ is the identity operation on *B*, and $\sigma_x^{(B)}$ is the Pauli matrix σ_x acting on *B*, whose matrix form in the computational basis $\left\{ |0\rangle = [1\ 0]^T,\ |1\rangle = [0\ 1]^T \right\}$ is

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{4}$$

It is obvious that $\sigma_x |0\rangle = |1\rangle$ and $\sigma_x |1\rangle = |0\rangle$, i.e., it can indeed invert the content of the register.

The unitary quantum operation that can implement the if/then/else conditional statement corresponding to CNOT gate is then constructed as

$$U_{CNOT} \equiv P_0^{(A)} \otimes U_0^{(B)} + P_1^{(A)} \otimes U_1^{(B)} = |0\rangle_A \langle 0| \otimes I^{(B)} + |1\rangle_A \langle 1| \otimes \sigma_x^{(B)}, \tag{5}$$

whose matrix form is

$$U_{CNOT}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}. \tag{6}$$

Applying it on all possible states $|x\rangle_A\otimes|y\rangle_B$ ($|x\rangle_A,|y\rangle_B\in\{|0\rangle,|1\rangle\}$) of the composite system $A\otimes B$, and it can be verified that U_{CNOT} can indeed accomplish the desired task

$$U_{CNOT}\left(|x\rangle_A\otimes|y\rangle_B\right)=|x\rangle_A\otimes|x\oplus y\rangle_B. \tag{7}$$

3.1.2. Controlled swap (CSWAP) gate [6,7]

CSWAP gate is also known as the Fredkin gate. It applies on three registers *A*, *B* and *C*, where *A* is the control register while both *B* and *C* are the target registers. The goal is: if the content of *A* is 0 then keep both *B* and *C* unaltered, else if the content of *A* is 1 then swap the content of *B* and *C*.

When constructing the quantum form of CSWAP gate, the condition "if the content of *A* is 0 (or 1)" is also implemented by the projective operators $P_0^{(A)}\equiv|0\rangle_A\langle 0|$ and $P_1^{(A)}\equiv|1\rangle_A\langle 1|$ acting on *A*. The corresponding unitary operators on $B\otimes C$ are $U_0^{(BC)}\equiv I^{(B)}\otimes I^{(C)}$ (i.e., the identity operation) and $U_1^{(BC)}\equiv U_{SWAP}$. Here U_{SWAP} denotes the operation for swapping the states of *B* and *C*. When both *B* and *C* are qubits, the matrix form of U_{SWAP} in the computational basis is

$$U_{SWAP}=\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}. \tag{8}$$

Then the entire unitary operation on $A\otimes B\otimes C$ is

$$\begin{aligned}U_{CSWAP}&=P_0^{(A)}\otimes U_0^{(BC)}+P_1^{(A)}\otimes U_1^{(BC)}\\&=|0\rangle_A\langle 0|\otimes\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}_{BC}+|1\rangle_A\langle 1|\otimes\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}_{BC}\\&=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\end{bmatrix}_{ABC}.\end{aligned} \tag{9}$$

We can verify that

$$U_{CSWAP}\left(|0\rangle_A\otimes|x\rangle_B\otimes|y\rangle_C\right)=|0\rangle_A\otimes|x\rangle_B\otimes|y\rangle_C \tag{10}$$

and

$$U_{CSWAP}\left(|1\rangle_A \otimes |x\rangle_B \otimes |y\rangle_C\right) = |1\rangle_A \otimes |y\rangle_B \otimes |x\rangle_C, \tag{11}$$

i.e., U_{CSWAP} is exactly what we want.

When both *B* and *C* are multi-level systems so that their content are not limited to bits, all we need is to find the corresponding matrix form of U_{SWAP} to replace Eq. (8), and substitute it into Eq. (9) to obtain U_{CSWAP}.

3.1.3. Controlled-controlled NOT (CCNOT) gate [6,7]

It is also called Toffoli gate. Similar to the CSWAP gate, it also applies on three registers, and keeps both *B* and *C* unaltered if the content of *A* is 0. The difference is when the content of *A* is 1, *B* is taken as the control register and *C* is taken as the target register, and a CNOT operation is performed on $B \otimes C$.

For simplicity, suppose that the content of all three registers are limited to bits. Following the above procedure, the quantum operation for this gate can be found immediately as

$$\begin{aligned} U_{CCNOT} &= |0\rangle_A\langle 0| \otimes (I^{(B)} \otimes I^{(C)}) + |1\rangle_A\langle 1| \otimes U_{CNOT}^{(BC)} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}_{ABC}. \end{aligned} \tag{12}$$

3.2. A more complicated example

Now it will be shown how to construct the quantum form of the following conditional statement:

If $i < 3$ *then* $i \leftarrow i^2$,
else if $i = 3$ *then* $i \leftarrow 7$,
else if $i > 3$ *then* $i \leftarrow i - 3$.
(Here $i \in \{0,1,\ldots,7\}$*.)*

At first glance, the quantum implementation of this conditional statement may appear difficult because it requires the comparison operation of *i*. But in fact there are only three possibilities that satisfy the condition $i < 3$, that is: $i = 0$, $i = 1$, and $i = 2$. Therefore, for the control register *A* where *i* is stored, only three projective operators $P_0^{(A)} \equiv |0\rangle_A\langle 0|$, $P_1^{(A)} \equiv |1\rangle_A\langle 1|$ and $P_2^{(A)} \equiv |2\rangle_A\langle 2|$ are needed to accomplish the test of the condition $i < 3$. Likewise, the condition $i > 3$ can be implemented with projective operators $P_4^{(A)} \equiv |4\rangle_A\langle 4|$, $P_5^{(A)} \equiv |5\rangle_A\langle 5|$, $P_6^{(A)} \equiv |6\rangle_A\langle 6|$ and $P_7^{(A)} \equiv |7\rangle_A\langle 7|$, while $i = 3$ is represented by $P_3^{(A)} \equiv |3\rangle_A\langle 3|$.

Then another target register *B* is introduced for storing the corresponding calculation results i^2, 7 and $i - 3$. The initial state of *B* is prepared as $|0\rangle_B$. Taking $i = 2$ as an example. In this case the state of *B* needs to be turned from $|0\rangle_B$ into $|2^2\rangle_B = |4\rangle_B$. Obviously, applying $|4\rangle_B\langle 0|$ on *B* can do the job. But $|4\rangle_B\langle 0|$ is

not a unitary operator. More terms need to be added to make it unitary while maintaining the ability of turning $|0\rangle_B$ into $|4\rangle_B$. The idea of finding other required terms is as follows. For any state orthogonal to $|0\rangle_B$, these terms should be able to turn them into a state orthogonal to $|4\rangle_B$. Also, any two orthogonal states should be turned into orthogonal states. There exist many choices satisfying these requirements. A simple choice is to select the states among the computational basis $\{|0\rangle,|1\rangle,...,|7\rangle\}$, e.g., to construct the operator

$$\begin{aligned} U_2^{(B)} &\equiv |4\rangle_B\langle 0|+|5\rangle_B\langle 1|+|6\rangle_B\langle 2|+|7\rangle_B\langle 3|+|0\rangle_B\langle 4|+|1\rangle_B\langle 5|+|2\rangle_B\langle 6|+|3\rangle_B\langle 7| \\ &= \sum_{j=0}^{7}|(4+j)\bmod 8\rangle_B\langle j|. \end{aligned} \tag{13}$$

It can be verified that

$$\left(U_2^{(B)}\right)^{+} U_2^{(B)} = \sum_{j=0}^{7}|j\rangle_B\langle j| = I, \tag{14}$$

i.e., it is indeed unitary, and

$$U_2^{(B)}|0\rangle_B = |4\rangle_B, \tag{15}$$

i.e., it can indeed turn $|0\rangle_B$ into $|4\rangle_B$. When the state of register B is not $|0\rangle_B$, $U_2^{(B)}$ will turn it into something else. But it does not matter to us, because the initial state of B was already taken as $|0\rangle_B$, so that other results will never occur.

Similarly, for $i=0$ and $i=1$, following the construction of Eq. (13), the operators on register B can be taken as

$$U_0^{(B)} \equiv \sum_{j=0}^{7}|(0^2+j)\bmod 8\rangle_B\langle j| \tag{16}$$

and

$$U_1^{(B)} \equiv \sum_{j=0}^{7}|(1^2+j)\bmod 8\rangle_B\langle j|, \tag{17}$$

respectively. They are both unitary, and can turn $|0\rangle_B$ into $|i^2\rangle_B$ ($i=0,1$).

More generally, for any function $f(i)$, if $|0\rangle_B$ is expected to be turned into $|f(i)\bmod(\dim B)\rangle_B$ (with $\dim B$ denoting the dimension of B), the operator on B can be taken as

$$U_i^{(B)} \equiv \sum_{j=0}^{(\dim B)-1}|(f(i)+j)\bmod(\dim B)\rangle_B\langle j|. \tag{18}$$

Therefore, for any $i>3$, as $|0\rangle_B$ needs to be turned into $|i-3\rangle_B$, the corresponding unitary operator can be taken as

$$U_i^{(B)} \equiv \sum_{j=0}^{7}|(i-3+j)\bmod 8\rangle_B\langle j|. \tag{19}$$

The unitary operator for $i=3$ that turns $|0\rangle_B$ into $|7\rangle_B$ is

$$U_3^{(B)} \equiv \sum_{j=0}^{7} \left|(7+j) \bmod 8\right\rangle_B \langle j|. \tag{20}$$

Finally, substituting them into Eq. (1), and we obtain the unitary operation

$$\begin{aligned} U &\equiv \sum_{i=0}^{7} P_i^{(A)} \otimes U_i^{(B)} \\ &= \sum_{i=0}^{2} |i\rangle_A \langle i| \otimes \left(\sum_{j=0}^{7} \left|(i^2+j) \bmod 8\right\rangle_B \langle j| \right) \\ &\quad + |3\rangle_A \langle 3| \otimes \left(\sum_{j=0}^{7} \left|(7+j) \bmod 8\right\rangle_B \langle j| \right) \\ &\quad + \sum_{i=4}^{7} |i\rangle_A \langle i| \otimes \left(\sum_{j=0}^{7} \left|(i-3+j) \bmod 8\right\rangle_B \langle j| \right). \end{aligned} \tag{21}$$

Applying it on $|i\rangle_A \otimes |0\rangle_B$, and we yield

$$U|i\rangle_A \otimes |0\rangle_B = \begin{cases} |i\rangle_A \otimes |i^2\rangle_B, & (i<3) \\ |i\rangle_A \otimes |7\rangle_B, & (i=3) \\ |i\rangle_A \otimes |i-3\rangle_B. & (i>3) \end{cases} \tag{22}$$

This completes the implementation of the conditional statement "if $i<3$ then $i \leftarrow i^2$, else if $i=3$ then $i \leftarrow 7$, else if $i>3$ then $i \leftarrow i-3$", and the result is stored in register *B*. For example, applying Eq. (21) on the initial state $|4\rangle_A \otimes |0\rangle_B$ (i.e., inputting $i=4$) will result in

$$\begin{aligned} U\left(|4\rangle_A \otimes |0\rangle_B\right) &= \sum_{i=0}^{2} |i\rangle_A \langle i|4\rangle_A \otimes \left(\sum_{j=0}^{7} \left|(i^2+j) \bmod 8\right\rangle_B \langle j|0\rangle_B \right) \\ &\quad + |3\rangle_A \langle 3|4\rangle_A \otimes \left(\sum_{j=0}^{7} \left|(7+j) \bmod 8\right\rangle_B \langle j|0\rangle_B \right) \\ &\quad + \sum_{i=4}^{7} |i\rangle_A \langle i|4\rangle_A \otimes \left(\sum_{j=0}^{7} \left|(i-3+j) \bmod 8\right\rangle_B \langle j|0\rangle_B \right) \\ &= 0+0+|4\rangle_A \langle 4|4\rangle_A \otimes \left(0+\left|(4-3+0) \bmod 8\right\rangle_B \langle 0|0\rangle_B\right) = |4\rangle_A \otimes |1\rangle_B. \end{aligned} \tag{23}$$

If the result is further required to be stored in register *A*, then all it needs is simply to apply an operation U_{SWAP} on $A \otimes B$ like the one shown in Eq. (8) (but needed to be expanded to *8*×*8*-dimensional systems) to swap the states of registers *A* and *B*.

4. DISCUSSION

From the above examples, it can be seen that the trick is: the initial state of the control register is not prepared as a specific state in the computational basis. On the contrary, it is a superposition of the states corresponding to different Boolean conditions (e.g., $\left(|0\rangle_A + |1\rangle_A\right)/\sqrt{2}$). With this method, the unitary operation *U* in the form of Eq. (1) actually accomplishes a quantum parallel computation of multiple Boolean conditions. Also, even though *U* contains the projective operator $P_i^{(X)}$ on the control register *X*, the state in *X*

will not actually be measured and collapsed because all $P_i^{(X)}$ form a complete measurement on X. Instead, the control register X and the target register Y together experience a unitary transformation. Then the calculation results of the if/then/else logic are also stored in Y as a quantum superposition.

Although this does not necessarily imply quantum speed-up over classical algorithms (it still depends on the specific problem to be solved, just like other quantum algorithms), the superposition still leaves more freedom for the operations at a later time. This is very useful in both quantum computation and quantum cryptography. One of such examples can be found in Ref. [10], whose Eq. (2) is exactly an implementation of quantum parallel computation of the if/then/else logic, which enables an attack strategy against a class of quantum oblivious transfer protocols. In brief, an honest participant Bob in such a protocol is supposed to choose a random classical bit b and perform a series of operations T_b ($b \in \{0,1\}$) accordingly. But a dishonest Bob can prepare a control qubit $|b\rangle = \left(|0\rangle + |1\rangle\right)/\sqrt{2}$, then implement the conditional statement "if $b = 0$ ($b = 1$) then perform T_0 (T_1)" at the quantum level. Since the value of b is not fixed, at a later time, rather than using the basis $\{|0\rangle, |1\rangle\}$, Bob can measure $|b\rangle$ with another basis, so that he can obtain an extra amount of mutual information which is unavailable when using the classical if/then/else operations.

5. CONCLUSIONS

In summary, the above examples clearly showed that the if/then/else type of logic in classical computers can also be run in quantum computer, and it was already a common practice in quantum logic gates, thus the claim in Ref. [5] is refuted. A general method is also provided for constructing the unitary transformation required for any if/then/else conditional statement, and it is illustrated how to apply it for solving more complicated and valuable problems.

ACKNOWLEDGMENTS

This work was supported in part by Guangdong Basic and Applied Basic Research Foundation under Grant No. 2019A1515011048.

REFERENCES

1. P.W. SHOR, *Algorithms for quantum computation: discrete log and factoring*, DIMACS Technical Report 94-37, June 1994.
2. L.K. GROVER, *A fast quantum mechanical algorithm for database search*, Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, ACM Press, 1996, pp. 212–219.
3. A.W. HARROW, A. HASSIDIM, S. LLOYD, *Quantum algorithm for linear systems of equations*, Physical Review Letters, **103**, *15*, art. 150502, 2009.
4. E. FARHI, J. GOLDSTONE, S. GUTMANN, *A quantum approximate optimization algorithm*, Report number: MIT-CTP/4610, arXiv preprint arXiv: 1411.4028, 2014.
5. N. HEATH, *Quantum computing: Seven truths you need to know*, https://www.techrepublic.com/article/quantum-computing-seven-truths-you-need-to-know/ and https://www.onlyinfotech.com/2018/07/25/quantum-computing-seven-truths-you-need-to-know-info-innovation/, accessed online February 2023.
6. G.P. HE, *Quantum information made easy* (in Chinese), Science Press, Beijing, 2012.
7. M.A. NIELSEN, I.L. CHUANG, *Quantum computation and quantum information*, Cambridge University Press, 2000.
8. P.W. SHOR, *Scheme for reducing decoherence in quantum computer memory*, Physical Review A, **52**, *4*, pp. R2493–R2496, 1995.
9. K. MOCK, *Boolean conditions, If-Then*, CS A201: Programming Concepts I, University of Alaska Anchorage, 2012. http://www.cse.uaa.alaska.edu/~afkjm/cs201/handouts/IfStatements.pdf
10. G.P. HE, *Can relativistic bit commitment lead to secure quantum oblivious transfer?*, European Physical Journal D, **69**, art. 93, 2015.

Received February 6, 2023