An Exclusive-Sum Form for Reversible Circuit Using Basic Quantum Gates

Shiou-An Wang[1]
Chin-Yung Lu[2]

[1]Assistant Professor, Department of Computer Science and Information Engineering, Delin Institute of Technology

[2]Instructor, Department of Electronic Engineering, Delin Institute of Technology

Abstract

The concern with quantum computing has been growing for the last several years. This results in that reversible circuit has been brought to public attention again since a reversible circuit can be synthesized by quantum gates. The most important point of using reversible circuits is to reduce the problem of power dissipation. If a circuit is reversible, it can reduce the energy consumption caused by information loss. We need a algorithm to synthesize a reversible circuit, but the classical synthesis algorithm is not directly applicable to the synthesis of reversible circuits because the basic classical gates, except the NOT gate, are not reversible gates. In this paper, we propose an exclusive-sum form which can be easily transformed into a reversible circuit by using the basic quantum gates including NOT, CN, and Toffoli gates. In fact, the resulting expression in exclusive-sum form generated by synthesis algorithms can be transformed into a more simplified reversible circuit. We have shown that a reversible circuit in exclusive-sum form has lower quantum cost than one in exclusive-sum-of- products form after the combination of terms. Moreover, if permutations are represented as an expression in exclusive-sum form, we can realize permutations to be reversible circuits with lower quantum cost and without unnecessary garbage bits. Similarly, we can also synthesize irreversible circuits by transforming into an expression in exclusive-sum form and adding qubits to make the circuits reversible.

Key word –Quantum computing, Circuit optimization, Reversible circuit, Logic synthesis

使用基本量子邏輯閘表示可逆電路的互斥和格式

王秀安 [1]

盧勤庸 [2]

[1] 德霖技術學院資訊工程系助理教授
[2] 德霖技術學院電子工程系專任講師

中文摘要

在最近幾年，對於量子計算的關注是越來越高，這是因為可逆電路可以用量子邏輯閘進行電路合成，所以這導致大眾又再度注意到可逆電路，應用可逆電路可以減輕功率消耗所帶來的問題，如果電路是可逆的，它可以降低因為訊息損失所造成的能源消耗，我們需要一個演算法執行合成可逆電路，但是傳統合成演算法無法直接適用在可逆電路，這是因為除了 NOT 閘之外，基本傳統邏輯閘都不是可逆邏輯閘。在這個論文中，我們提出一種互斥和格式，可以很容易地轉換成為可逆電路，只要使用基本量子邏輯閘，其中包括 NOT、CN 和 Toffoli 閘，事實上，由合成演算法所產生以互斥和格式表示的結果關係式，可以轉換成為更簡化的可逆電路，我們已經表明，一個以互斥和格式表示的可逆電路比以互斥和-積格式表示的可逆電路，具有更低的量子成本。此外，如果排列電路可以用以互斥和格式表示，我們就可以把排列電路轉換成為可逆電路，而且這個電路具有較低的量子成本和沒有不必要的無用量子位元。同樣地，我們也可以合成不可逆電路，只要轉換成為以互斥和格式表示的關係式，以及增加量子位元，就可以使得電路變成可逆。

關鍵字 – 量子計算，電路最佳化，可逆電路，邏輯合成

I. Introduction

For the future digital circuit design, energy consumption will be a more and more critical problem. As the process technology progresses, circuit speed becomes much faster while energy consumption problem is becoming more and more serious. Landauer [1] proved that traditional binary irreversible gates result in power dissipation in spite of implementation. Bennett [2] showed that for a binary circuit built from reversible gates no power is consumed due to information loss. It is tempting to consider that reversible circuits may be useful for the power dissipation problem. Moreover, quantum computing [3] is one of the important applications of reversible circuits. Several quantum algorithms which improve some traditional problems have been proposed [4], [5]. Thus, quantum computing becomes one of the most rapidly expanding research fields. In fact, quantum circuits must be reversible, so reversible circuits can be a special case of quantum circuits. In order to find simpler reversible circuits, some works [6-11] on developing algorithms for synthesis of reversible circuits have been proposed.

A *reversible circuit* that produces a unique output vector for each input vector has the same number of inputs and outputs. To build a reversible circuit, we need a set of reversible gates as a universal gate set to synthesize the circuit. But traditional gates such as AND, OR, and exclusive-OR are not reversible. Only the NOT gate is a reversible gate. Therefore, a set of reversible gates different from traditional gates need to be defined. In fact, several reversible gates, including the controlled NOT gate (called the CN gate) and the Toffoli gate [12], have been proposed. Therefore, a set of reversible gates including NOT, CN, and Toffoli gates are used to synthesize the reversible circuits.

For the classical circuit synthesis, we can use the truth table to generate the expression in sum-of-products form. This is based on that there are NOT, AND, and OR gates in the basic classical gate library. However, it is hard to synthesize a more simplified reversible circuit by using basic classical gate library because the basic quantum gate library has NOT, CN, and Toffoli gates different from the classical gate library on the gate function, except for the NOT gate. We should develop a new more suitable form for the reversible circuit design to synthesize a more simplified circuit.

The remainder of this paper is organized as follows. In Section II, we briefly describe the basic definitions of reversible logic theory. We propose an exclusive-sum form which can transform a truth table into a reversible circuit described in Section III. In Section IV, the comparison between exclusive-sum form and exclusive-sum-of-products form in the different criteria are presented. In Section V, we realize permutations and irreversible circuits to become reversible circuits by using the expressions in exclusive-sum form. Finally, we show the conclusion in Section VI.

II. Notations and Preliminaries

A. Basic Definitions

A literal is a variable or its complement, and a *minterm* of *n* variables is a product of *n* literals in which each variable appears exactly once in either true or complemented form, but not both. For example, the term $ab\bar{c}$ is a minterm of 3 variables. Clearly, a *permutation* which performs permutations of input vectors is a reversible circuit because it has the same number of input and output variables and each mapping to an output pattern from an input pattern is unique. For example, a truth table of a permutation is shown in Table 1(a).

A full adder whose truth table is in Table 1(b) is not a reversible circuit because the number of inputs is not equal to the number of outputs and the input vector can not map to a unique output vector. This kind of circuits is called *irreversible circuits*. But they can be easily transformed into reversible circuits by adding some qubits. For example, we can add one qubit to a full adder and it becomes a reversible circuit with 4 inputs and 4 outputs. The synthesis of a reversible circuit is to transform an irreversible circuit into a reversible one. This can reduce the energy consumption due to information loss. The number of inputs is equal to the number of outputs for a reversible circuit. So some outputs of a reversible circuit transformed from an irreversible circuit will be redundant. These outputs need not be synthesized. The main function of these outputs is to help generate a simpler reversible circuit. Therefore, for a reversible circuit, some of its outputs called *significant outputs* are functional, others are redundant. These redundant outputs are called *garbage bits*.

Table 1: Truth tables. (a) Permutation. (b) Irreversible Circuit.

a	*b*	*c*	*f*	*g*	*h*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

(a)

x_1	x_2	c_{in}	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

(b)

An *m*-input *k*-output Boolean function can become a reversible circuit with *n* inputs and *n* outputs ($n \geq m$ and $n \geq k$) after adding some qubits and the number of garbage bits is *n*-*k*. For example, a full adder has 3 inputs and 2 outputs as shown in Table 1(b). A full adder is first transformed into a circuit with 3 inputs and 3 outputs and now it is still an irreversible circuit. Then by adding one qubit makes it a reversible circuit with 4 inputs and 4 outputs. The number of garbage bits is 2 since n-k=4-2=2. The number of significant outputs is also 2.

Definition 1. A generalized Toffoli gate with n+1 inputs and n+1 outputs is a reversible gate performing the operation

$$(x_1, x_2, \ldots, x_n, x_{n+1}) \rightarrow (x_1, x_2, \ldots, x_n, (x_1 \cdot x_2 \cdot \ldots \cdot x_n) \oplus x_{n+1}),$$

where x_1, x_2,…, and x_n are controls, x_{n+1} is a target and the notation ⊕ denotes the exclusive-OR operation. □

According to Definition 1, a NOT gate which performs NOT operation is a generalized Toffoli gate with 1 input and 1 output. Its operation is $(x) \rightarrow (\bar{x})$. Similarly, a CN gate which performs exclusive-OR operation is a generalized Toffoli gate with 2 inputs and 2 outputs. Its operation is $(x_1, x_2) \rightarrow (x_1, x_1 \oplus x_2)$. A Toffoli gate with 2 controls is a generalized Toffoli gate with 3 inputs and 3 outputs. Its operation is $(x_1, x_2, x_3) \rightarrow (x_1, x_2, (x_1 \cdot x_2) \oplus x_3)$. It performs AND operation if input x_3 is equal to 0. These gates are shown in Fig. 1. In Fig. 1(b), a 2-input and 2-output reversible circuit has a CN gate. The vertical line represents a gate, where the notation ⊕ denotes a target and ● a control. The horizontal lines represent wires or qubits.

Our universal gate set contains three basic reversible gates, NOT, CN, and Toffoli gates. A reversible circuit can be realized by the universal gate set after adding the needed garbage bits. This set is the smallest complete set of gates. These gates were defined by Toffoli [12] and are all reversible gates.

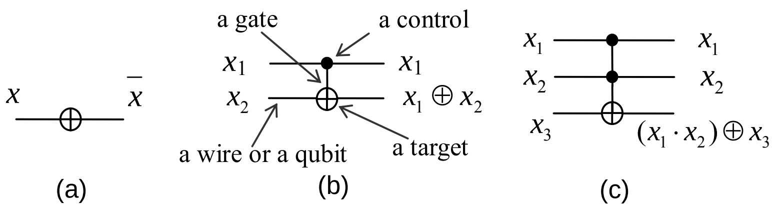

Fig. 1. Basic reversible gates. (a) NOT gate. (b) CN gate. (c) Toffoli gate.

B. Comparison Criteria

It is possible that more than one minimization result can be generated, so we need a mechanism to estimate the cost for each selection. Actually, we apply two main criteria to determine which is a better result. One is the number of garbage bits. The synthesis method in [16] reduces the quantum cost by introducing a few additional garbage bits. The quantum costs of their simplified circuits are lower than others, but increasing the number of garbage bits lead to another problem. Since reversible gates are all quantum gates and the number of qubits is limited and very expensive to implement by today's technology, the number of garbage bits must be as few as possible. Thus it is clear that the number of garbage bits is an important criterion for the synthesis methods. Another criterion is the quantum cost. The quantum cost is based on that proposed in [13-15]. The quantum cost of a reversible gate is the number of one- or two-qubit controlled-V operations which implement the gate. Thus, the quantum cost of NOT and CN gates are all 1, but a Toffoli gate with 2 controls is 5.

Besides, the quantum cost of a Toffoli gate with fewer controls is much lower than one with more controls. For instance, two 3-control Toffoli gates have lower quantum cost than one 4-control Toffoli gate. Therefore, a circuit should reduce the number of Toffoli gates with more controls. According to the previous discussion, we found that the total number of gates is not an important criterion to determine which circuit is better. However some methods [17-19] still choose the total gate count as one of their important criteria. Therefore, our main criteria to evaluate the simplified circuits only include the number of garbage bits and the quantum cost. A good reversible circuit should have fewer garbage bits and lower quantum cost.

III. Exclusive-Sum Form

The form of the resulting expression by using the classical synthesis algorithms is a sum-of-products form such as $f = \bar{a}bc + a\bar{b}\bar{c}$. This is on the basis of the gate set including NOT, OR and AND gates. Our gate set is different because OR operation is replaced by exclusive-OR operation. Thus, the form should be changed into the exclusive-sum-of-products form. For example, $f = \bar{a}bc \oplus a\bar{b}\bar{c}$. But we found the expression in this form does not have the lowest cost for reversible circuits.

To compare the two forms, we use the quantum cost as a criterion. The quantum cost of an AND operation implemented by a Toffoli gate is more expensive than an exclusive-OR operation implemented by a CN gate. The quantum cost of a Toffoli gate become larger when the number of inputs increases. Thus, we should reduce the number of inputs of a Toffoli gate as well as the number of Toffoli gates. Although the number of CN gates will also increase, the total quantum cost will be lower. For example, let $f = \bar{a}bc \oplus a\bar{b}\bar{c} = (a \oplus b) \cdot (a \oplus c)$. Originally two 4-input Toffoli gates and 4 NOT gates are needed to realize output *f*. After changing the expression form, it only needs one 3-input Toffoli gate and two CN gates. Clearly, the total gate count and quantum cost is much lower. Thus, this expression form can reduce both the number of Toffoli gates and the number of inputs of a Toffoli gate. Therefore, the most suitable form is defined as follows:

***Definition 2*:** An *exclusive-sum* form of a reversible circuit is the exclusive-OR sum of product terms, in which a product term is composed of literals or subterms, where a subterm can be transformed into CN gates or a Toffoli gate.

For example, $f = e \oplus (a \oplus b)(a \oplus c) \oplus ad$ is the expression in exclusive-sum form. Actually, an exclusive-sum form is a variation of exclusive-sum-of-products form. It is a 3-level expression.

Lemma 1: $(x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_n) = (\overline{x_1 \oplus x_2})(\overline{x_1 \oplus x_3}) ... (\overline{x_1 \oplus x_i})(x_{i+1} x_{i+2} ... x_n)$

$= (\overline{x_1 \oplus x_2})(\overline{x_2 \oplus x_3}) ... (\overline{x_{i-1} \oplus x_i})(x_{i+1} x_{i+2} ... x_n)$, where $x_1, x_2, ..., x_n$ are variables or terms, and a term here is the exclusive-OR sum of variables.

Proof: First we will expand the left expression as follows:

$$
\begin{aligned}
&(x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_n) \\
&= [(\overline{x_1 x_2 ... x_i})(\bar{x}_1 \bar{x}_2 ... \bar{x}_i) + (x_1 x_2 ... x_i)(\overline{\bar{x}_1 \bar{x}_2 ... \bar{x}_i})](x_{i+1} x_{i+2} ... x_n) \\
&= [(\bar{x}_1 + \bar{x}_2 + ... + \bar{x}_i)(\bar{x}_1 \bar{x}_2 ... \bar{x}_i) + (x_1 x_2 ... x_i)(x_1 + x_2 + ... + x_i)](x_{i+1} x_{i+2} ... x_n) \\
&= (\bar{x}_1 \bar{x}_2 ... \bar{x}_i + x_1 x_2 ... x_i)(x_{i+1} x_{i+2} ... x_n) \qquad (1)
\end{aligned}
$$

Then expand the expression in the middle as follows:

$$
\begin{aligned}
&(\overline{x_1 \oplus x_2})(\overline{x_1 \oplus x_3}) ... (\overline{x_1 \oplus x_i})(x_{i+1} x_{i+2} ... x_n) \\
&= (\bar{x}_1 \bar{x}_2 + x_1 x_2)(\bar{x}_1 \bar{x}_3 + x_1 x_3) ... (\bar{x}_1 \bar{x}_i + x_1 x_i)(x_{i+1} x_{i+2} ... x_n) \\
&= (\bar{x}_1 \bar{x}_2 ... \bar{x}_i + x_1 x_2 ... x_i)(x_{i+1} x_{i+2} ... x_n) \qquad (2)
\end{aligned}
$$

Since the expanded results of Eq. (1) and (2) are equal, so the equality is proved. Similarly, we can prove the following equality $(x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_n)$

$= (\overline{x_1 \oplus x_2})(\overline{x_2 \oplus x_3}) ... (\overline{x_{i-1} \oplus x_i})(x_{i+1} x_{i+2} ... x_n)$. □

Lemma 2: $(x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_{n-1}) = (\overline{x_1 \oplus x_2})(\overline{x_1 \oplus x_3}) ... (\overline{x_1 \oplus x_i})(x_{i+1} ... x_{n-1})$

$\oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n)$, where $x_1, x_2, ..., x_n$ are variables or terms, and a term here is the exclusive-OR sum of variables.

Proof: We can add redundant terms $(x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n) \oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n)$.

$$
\begin{aligned}
&(x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_{n-1}) \\
&= (x_1 x_2 ... x_i x_{i+1} ... x_n) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_{n-1}) \oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n) \oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n) \\
&= (x_1 x_2 ... x_i x_{i+1} ... x_{n-1}) \oplus (\bar{x}_1 \bar{x}_2 ... \bar{x}_i x_{i+1} ... x_{n-1}) \oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n)
\end{aligned}
$$

Applying Lemma 1, we can obtain

$(\overline{x_1 \oplus x_2})(\overline{x_1 \oplus x_3}) ... (\overline{x_1 \oplus x_i})(x_{i+1} ... x_{n-1}) \oplus (x_1 x_2 ... x_i x_{i+1} ... x_{n-1} \bar{x}_n)$ □

***Lemma 3*:** $(zx_1x_2...x_m)\oplus(x_1x_2...x_my_1y_2...y_n)=(x_1x_2...x_m)(z\oplus y_1y_2...y_n)$, where z, x_i, and y_i are variables.

Proof: Applying the distributive law $X(Y\oplus Z)=XY\oplus XZ$, the equality is proved. □

For finding a more simplified result, the truth table can be synthesized to generate an expression in exclusive-sum-of-products form by using synthesis algorithms, and then the expression is transformed into one in exclusive-sum form by applying Lemma 1-3. Applying Lemma 1-3 can make the quantum cost of the expression lower. The detailed discussion is described in Section IV:

IV. Comparison by the Different Criteria

First we will discuss the performance comparison of two forms in Lemma 1 by comparing the criteria of their reversible circuits. Assume that an expression synthesized by a synthesis algorithm has two product terms $x_1x_2...x_n$ and $\bar{x}_1\bar{x}_2...\bar{x}_ix_{i+1}...x_n$ which differ in i variables, where i>1. The reversible circuit realized from them is in Fig. 2(a). The gate count N_{a1} and the quantum cost Q_{a1} are shown below:

$$\begin{cases} N_{a1}=i+2 \\ Q_{a1}=Q(2T(n+1)+iNOT) \end{cases}$$

where $T(n+1)$, NOT, and Q(gates) denote a Toffoli gate with n+1 inputs, a NOT gate, and the quantum cost of the gates, respectively. They can be combined to become a new term $(\bar{x}_1\oplus x_2)(\bar{x}_1\oplus x_3)...(\bar{x}_1\oplus x_i)(x_{i+1}x_{i+2}...x_n)$ by Lemma 1 and $\overline{X\oplus Y}=\overline{X}\oplus Y$. Its reversible circuit is shown in Fig. 2(b). The gate count N_{b1} and the quantum cost Q_{b1} are shown below:

$$\begin{cases} N_{b1}=i+1 \\ Q_{b1}=Q(T(n)+(i-1)CN+NOT) \end{cases}$$

where CN denotes a CN gate. The comparison between two circuits is described as follow:

$$\begin{cases} N_{a1}>N_{b1} \\ Q_{a1}>Q_{b1} \\ G_{a1}=G_{b1} \end{cases}$$

where G_{a1} and G_{b1} are the garbage bits of two reversible circuits, respectively. Thus, an expression in exclusive-sum-of-products form is transformed into one in exclusive-sum form by Lemma 1. Its gate count and quantum cost become lower and its garbage bit is not changed after transformation.

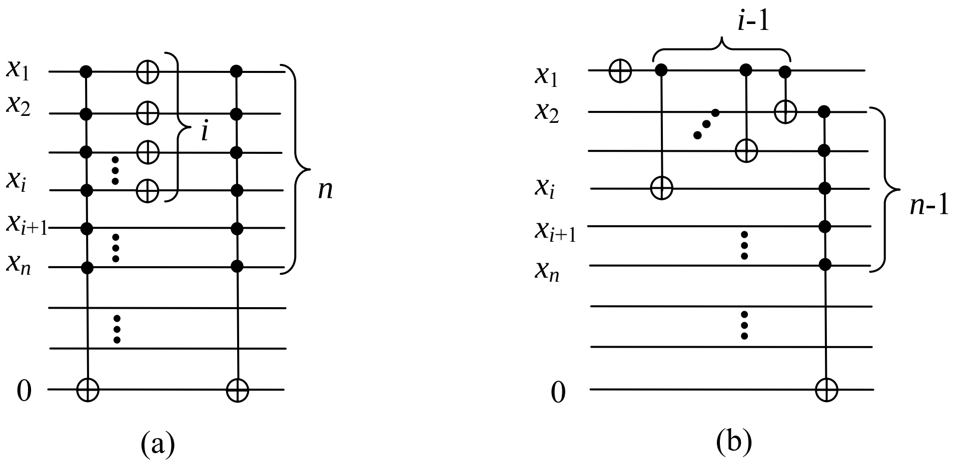

Fig. 2. Two equivalent reversible circuits transformed by Lemma 1.

Similarly, if an expression has two terms $x_1x_2...x_n$ and $\overline{x_1}\,\overline{x_2}...\overline{x_i}x_{i+1}...x_{n-1}$. This case can evaluate the performance comparison of two forms in Lemma 2 by comparing the criteria of their reversible circuits. The reversible circuit realized from them is in Fig. 3(a). The gate count N_{a2} and the quantum cost Q_{a2} are shown below:

$$\begin{cases} N_{a2} = i+2 \\ Q_{a2} = Q(T(n+1)+T(n)+iNOT) \end{cases}$$

They can be combined to become $(\overline{x_1} \oplus x_2)(\overline{x_1} \oplus x_3)...(\overline{x_1} \oplus x_i)(x_{i+1}x_{i+2}...x_{n-1}) \oplus x_1x_2...x_{n-1}\overline{x_n}$ by Lemma 2 and $\overline{X \oplus Y} = \overline{X} \oplus Y$. Its reversible circuit is in Fig. 3(b). The gate count N_{b2} and the quantum cost Q_{b2} are shown below:

$$\begin{cases} N_{b2} = i+3 \\ Q_{b2} = Q(T(n+1)+T(n-1)+(i-1)CN+2NOT) \end{cases}$$

If an expression has two terms $x_1x_2...x_{n-1}\overline{x_n}$ and $\overline{x_1}\,\overline{x_2}...\overline{x_i}x_{i+1}...x_{n-1}$, N_{a2}=i+3 and N_{b2}=i+2. This means that N_{a2} is almost equal to N_{b2}, which is dependent on the input value. The comparison between two circuits is described as follow:

$$\begin{cases} N_{a2} \cong N_{b2} \\ Q_{a2} > Q_{b2} \\ G_{a2} = G_{b2} \end{cases}$$

Thus, an expression is transformed into one in exclusive-sum form by Lemma 2. Its quantum cost becomes lower and others are not changed after transformation.

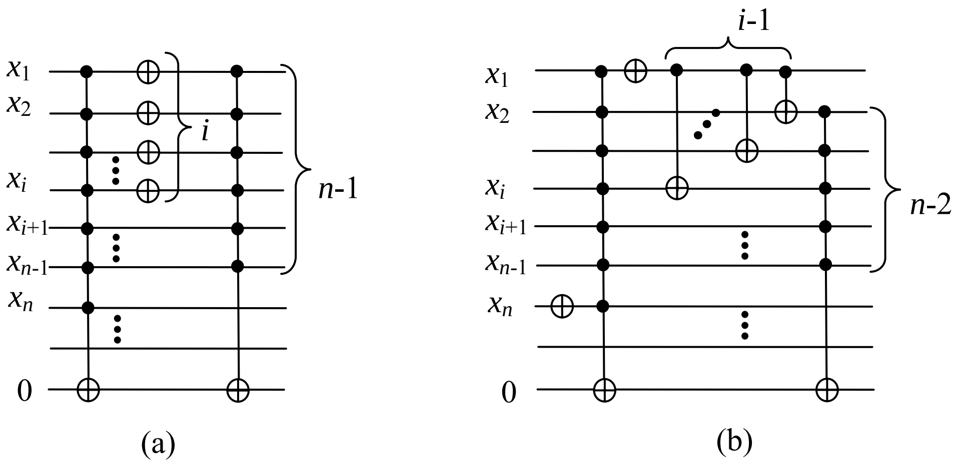

Fig. 3. Two equivalent reversible circuits transformed by Lemma 2.

Now we will discuss the performance comparison of two forms in Lemma 3. Assume that an expression has two terms $zx_1x_2...x_m$ and $x_1x_2...x_my_1y_2...y_n$. The reversible circuit realized from them is in Fig. 4(a). The gate count N_{a3} and the quantum cost Q_{a3} are shown below:

$$\begin{cases} N_{a3} = 2 \\ Q_{a3} = Q(T(m+2)+T(m+n+1)) \end{cases}$$

They can be combined to become $(x_1x_2...x_m)(z \oplus y_1y_2...y_n)$ by Lemma 3. Its circuit is in Fig. 4(b). The gate count N_{b3} and the quantum cost Q_{b3} are shown below:

$$\begin{cases} N_{b3} = 2 \\ Q_{b3} = Q(T(n+1)+T(m+2)) \end{cases}$$

The comparison between two circuits is described as follow:

$$\begin{cases} N_{a3} = N_{b3} \\ Q_{a3} > Q_{b3} \\ G_{a3} = G_{b3} \end{cases}$$

Thus, an expression is transformed into one in exclusive-sum form by Lemma 3. Its quantum cost becomes lower and others are not changed after transformation.

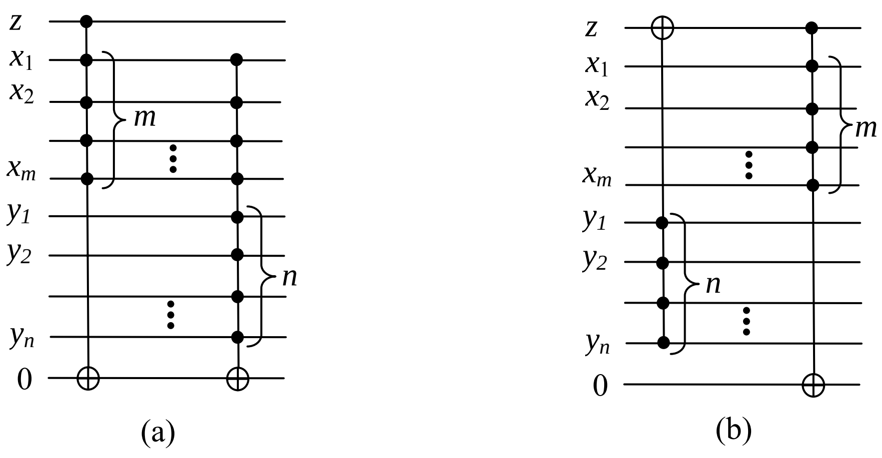

Fig. 4. Two equivalent reversible circuits transformed by Lemma 3.

V. Realizing Permutation and Irreversible Circuit

The expression in exclusive-sum-of-products form is still too complicated and can be further simplified. Actually, the expression initially generated by some of synthesis algorithms is in exclusive-sum-of-products form. After performing the process of expression transformation, the resulting expression really becomes an exclusive-sum form. There are two steps to do the process of expression transformation as shown below:

First, we can combine two product terms of the expression to generate a more simplified term by Lemma 1-3. According to the discussion of the previous section, we can find that the quantum cost of the simplified expression after expression transformation becomes lower.

Second, we should do our best to reduce the number of NOT gates since the quantum cost of a NOT gate is equal to a CN gate. Thus, in order to reduce the gate count and quantum cost, the number of NOT gates should be further reduced by the following two equalities

$$\begin{cases} \overline{(X \oplus Y)} = \overline{X} \oplus Y = X \oplus \overline{Y} \\ \overline{X}\,\overline{Y} = \overline{X} \oplus \overline{X}Y = \overline{Y} \oplus X\overline{Y} \end{cases}$$

where *X* and *Y* are variables or terms, and a term here is the exclusive-OR sum of variables. This reveals that the NOT gates can be shared or eliminated to reduce the number of NOT gates. For example, the function $f = a \oplus \overline{c} \oplus a\overline{(b \oplus c)}$ can move the position of a NOT operation by $\overline{X \oplus Y} = X \oplus \overline{Y}$. The new function $a \oplus \overline{c} \oplus a(b \oplus \overline{c})$ has lower quantum cost since the NOT operation $\overline{c}$ is shared.

Although an expression can now be transformed into a more simplified one by the process of expression transformation, the synthesis work for a single output is not complete yet. We still need to realize a reversible circuit from the expression in exclusive-sum form. The algorithm to realize a reversible circuit from the expression is as shown below:

First, add parentheses in the expression: According to the function of gates, add appropriate parentheses into the expression. Each set of parentheses represents a CN gate or a Toffoli gate excluding a NOT gate. The direction of adding parentheses can be classified into three modes. If the resulting expression is $f = x_1 \oplus x_2 \oplus ... \oplus x_n$, where x_i is a variable or a term. The modes are described as follows:

a. From left to right: $f = (((x_1 \oplus x_2) \oplus x_3)... \oplus x_n)$.

b. From right to left: $f = (x_1 \oplus (x_2 \oplus ...(x_{n-1} \oplus x_n)))$.

c. From two sides to the center: $f = (((x_1 \oplus x_2) \oplus x_3) \oplus ... \oplus (x_{n-2} \oplus (x_{n-1} \oplus x_n)))$.

If parentheses can not be added to the expression, the sequence of the terms in the expression should be changed excluding the first term which can locate the output at the default qubit. For example, $f = (((c \oplus a) \oplus b) \oplus ab)$. There are 3 set of parentheses in the expression *f*. This means the reversible circuit has 3 gates which are 2 CN gates and 1 Toffoli gate.

Second, transform each set of parentheses into a corresponding gate one by one. If parentheses are contained within other parentheses, start with the innermost group and work outward to transform into corresponding gates. If parentheses are at the same level, the sequence of the transformation may need to be adjusted to realize the circuit with a lower gate count. For example, $f=((c\oplus ab)\oplus(b\oplus a))$. Since $(c\oplus ab)$ and $(b\oplus a)$ are at the same level, $(c\oplus ab)$ needs to be transformed first, then transform $(b\oplus a)$. This way we can generate a simplified circuit with only 3 gates. On the contrary, the circuit can not be generated without adding a qubit if $(b\oplus a)$ is transformed first.

If the circuit can not be realized, it is necessary to add qubits into the circuit. The right time to add a qubit is described as follows. First, if there is only a product term $(y_1y_2...y_m)$ in a set of parentheses, where y_i is a variable or a term, add a qubit to convert the product term into an AND gate. Second, if a term in a set of parentheses is $(xz\oplus xy_1y_2...y_m)$, where x is a variable or a term, and z is a 1 or a variable, then add a qubit to copy the function needed to implement the circuit. Third, a circuit must be added a qubit, or it can not be realized.

Furthermore, we can adjust the sequence of the terms in the expression to find the equivalent circuit. For example, the sequence of the expression $f=c\oplus a\oplus b\oplus ab$ is changed into $c\oplus ab\oplus b\oplus a$. We can find the equivalent circuits for the function f as shown in Fig. 5(a). In fact, f also can be transformed into $((c\oplus a)\oplus b(a\oplus b))$ and $((c\oplus b)\oplus a(a\oplus b))$ by $X\oplus XY=X(X\oplus Y)$. These equivalent circuits for the function f are shown in Fig. 5(b). Moreover, f can be converted to $((c\oplus b)\oplus a\bar{b})$ by $\bar{X}\bar{Y}=\bar{X}\oplus\bar{X}Y$, where $X=\bar{a}$ and $Y=b$. Similarly, f can be changed to $((c\oplus a)\oplus \bar{a}b)$ by $\bar{X}\bar{Y}=\bar{X}\oplus\bar{X}Y$, where $X=\bar{b}$ and $Y=a$. These equivalent circuits are shown in Fig. 5(c).

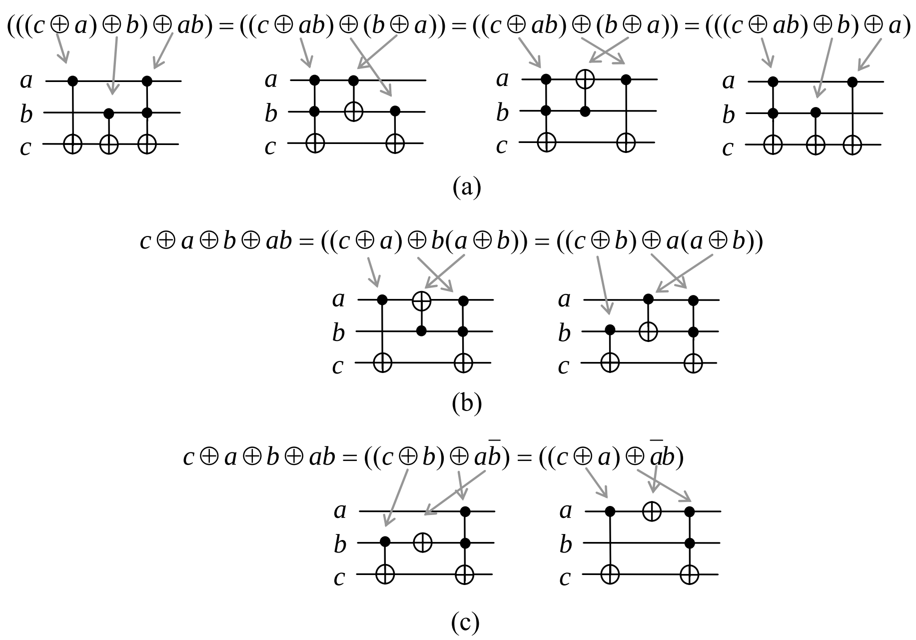

Fig. 5. The equivalent circuits for the function $f=c\oplus a\oplus b\oplus ab$.

The previous discussion is all about the synthesis of a reversible circuit. At the same time, these reversible circuits are all the circuits of permutations. The problem we have to consider next is the synthesis of an irreversible circuit. For the truth table of a general circuit, if the number of 1's of a single output is not one half of all minterms, this is not a reversible circuit. Further, if the truth table of a single output has one half of 1's, but each input vector can not map to a unique output vector, this is still not a reversible circuit. In fact, it is easy to transform an irreversible circuit into a reversible one. Usually, adding variables into the truth table can make it reversible. Then we can generate the expression in exclusive-sum form from the truth table.

VI. Conclusions

In this paper, we propose a new exclusive-sum form to represent an expression generated from a truth table. The exclusive-sum form is more applicable than other forms to generating a reversible circuit. The exclusive-sum form is designed to synthesize a more simplified reversible circuit assembled by a quantum gate library including NOT, CN, and Toffoli gates. According to our comparison information by the three different criteria, we found that a reversible circuit transformed from an expression in exclusive-sum form has a lower quantum cost and equal number of garbage bits. This means that the exclusive-sum form has better performance than other forms in the reversible circuit design. To use an expression in the exclusive-sum form, we can easily synthesize permutations to be a reversible circuit with a lower quantum cost. We can also convert irreversible circuits by adding needed qubits to make the circuits reversible.

References

[1] R. Landauer, “Irreversibility and heat generation in the computing process, “IBM J. Research and Development, pp. 5:183-191, 1961.

[2] C. H. Bennett, “Logical reversibility of computation, “IBM J. Research and Development, pp. 17:525-532, November, 1973.

[3] M. Nielsen and I. Chuang, “Quantum Computation and Quantum Information,” Cambridge, U.K.: Cambridge Univ. Press, 2000.

[4] P. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in Proc. of the 35th Annual IEEE Symposium on the Foundations of Computer Science, pp. 124-134, 1994.

[5] L. Grover, “A fast quantum mechanical algorithm for database search,” in Proc. of the 28th Annual ACM symposium on the Theory of Computing, pp. 212-219, 1996.

[6] A. Mishchenko and M. Perkowski, “Logic synthesis of reversible wave cascades, “ in International Workshop on Logic and Synthesis, pp. 197-202, June, 2002.

[7] D. Voudouris, S. Stergiou, and G. Papakonstantinou, “Minimization of Reversible Wave Cascades, “ IEICE Trans. Fundamentals, Vol. E88-A, No. 4, pp. 1015-1023, April, 2005.

[8] V. V. Shende, A. K. Prasad, K. N. Patel, I. L. Maslov, and J. P. Hayes, “Scalable simplification of reversible logic circuits, “ in IWLS, May, 2003.

[9] D. M. Miller and G. W. Dueck, “Spectral techniques for reversible logic synthesis, “ in 6th International Symposium on Representations and Methodology of Future Computing Technologies, pp. 56-62, March, 2003.

[10] P. Kerntopf, “A new heuristic algorithm for reversible logic synthesis, “ in Design Automation Conference, pp. 834-837, June, 2004.

[11] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes, “Synthesis of reversible logic circuits,” IEEE Trans. On CAD of Integrated Circuits and Systems, Vol. 22, No. 6, pp. 710-722, June, 2003.

[12] T. Toffoli, “Reversible computing, “ Tech. memo MIT/LCS/TM-151, MIT Lab for Comp. Sci., 1980.

[13] D. Maslov and G. W. Dueck, “Improved quantum cost for n-bit Toffoli gates, “ Electronics Letters, Vol. 39, No. 25, pp. 1790-1791, December, 2003.

[14] D. Maslov, G. W. Dueck, and N. Scott, “Reversible logic synthesis benchmarks page, “ [Online] Available: www.cs.uvic.ca/~dmaslov/, February, 2005.

[15] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, “Elementary gates for quantum computation,” Physical Review A, 50: pp. 3457-3467, Nov. 1995.

[16] D. Maslov, “Dynamic Programming algorithms as reversible circuits: symmetric function realization,” TR03-161, UNB, August, 2003.

[17] D. Maslov, D. M. Miller, and G. W. Dueck, “Techniques for the synthesis of reversible Toffoli networks, “ in quant-ph/0607166, July, 2006.

[18] D. M. Miller, D. Maslov, and G. W. Dueck, “A transformation based algorithm for reversible logic synthesis, “ in Design Automation Conference, pp. 318-323, June, 2003.

[19] A. Agrawal and N. K. Jha, “Synthesis of reversible logic, “ in Design, Automation and Test in Europe, pp.1384-1385, February, 2004.

[20] D. Maslov and G. W. Dueck, “Reversible cascades with minimal garbage, “ IEEE Trans. on CAD of Integrated Circuits and Systems, Vol. 23, No. 11, pp. 1497-1509, November, 2004.