Available online at www.sciencedirect.com
SCIENCE DIRECT®

Theoretical Computer Science 334 (2005) 259–274

Theoretical Computer Science

www.elsevier.com/locate/tcs

Majority-based reversible logic gates

Guowu Yang, William N.N. Hung*, Xiaoyu Song, Marek Perkowski

Department of Electrical and Computer Engineering, Portland State University, Portland, OR 97201, USA

Received 29 October 2003; received in revised form 7 July 2004; accepted 10 December 2004

Communicated by J. Watrous

Abstract

Reversible logic plays an important role in the synthesis of circuits for quantum computing. In this paper, we introduce families of reversible gates based on the majority Boolean function (MBF) and we prove their properties in reversible circuit synthesis. These gates can be used to synthesize reversible circuits of minimum "scratchpad register width" for arbitrary reversible functions. We show that, given a MBF f with $2k + 1$ inputs, f can be implemented by a reversible logic gate with $2k + 1$ inputs and $2k + 1$ outputs, i.e., without any constant inputs. We also demonstrate new gates from this family with very efficient quantum realizations for majority-based applications. They can be used to synthesize any reversible function of the same width in conjunction with inverters and Feynman (2-qubit controlled-NOT) gates. The gate universality problem is formulated in terms of elementary group theory and solved using the algebraic software GAP.
© 2005 Elsevier B.V. All rights reserved.

Keywords: Reversible logic; Quantum computing; Majority Boolean functions; Logic synthesis

1. Introduction

Reversible logic [5,6] circuits play an important role in application of quantum computing [12,14,35]. Majority (voting) functions are used in fault-tolerant computing [37] and other applications [25]. Various reversible gates that realize polarized majority function have relatively inexpensive quantum realizations [25]. There are extensive works in constructing reversible gates which have certain properties such as universality, symmetry,

* Corresponding author. Symplicity INC., 600 W. California Avenue, Sunnyvale, CA 94086, USA.
E-mail address: whung@synplicity.com (W.N.N. Hung).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.12.026

etc. [1,8,10,12,14–16,18,21,24,30,31,36]. In particular, there are synthesis algorithms by composition [24,27], decomposition [27], factorization [17], EXOR logic [14,19,20,27,33], group-theoretic methods [7,31,36], synthesis to regular structures [1,28–30], synthesis of various forms of reversible cascades [10,18,20–24,27] and spectral methods [22,23].

The Miller's gate [22] ($A = a\overline{b} \oplus ac \oplus \overline{b}c$, $B = \overline{a}b \oplus \overline{a}c \oplus bc$, $C = ab \oplus ac \oplus bc$, where inputs = abc, outputs = ABC) was proposed for quantum logic realizations or in emerging reversible technologies to compute the majority function. In this paper, we present a novel family of gates, of which the Miller's gate is a special case. As a motivation, we demonstrate in Section 2 some new reversible gates with lower cost than Miller's gate (through Toffoli [34] ($A = a$, $B = b$, $C = c \oplus (ab)$) or 2-qubit gates) in terms of quantum realization for majority function. In Section 3, we introduce our family of majority reversible gates and show that, given a majority Boolean function (MBF) f with $2k + 1$ inputs, f can be implemented by a majority-based reversible logic gate with $2k + 1$ inputs and $2k + 1$ outputs, i.e., without any constant inputs. In Section 4, we show that our low cost gates can be used in synthesis without constant inputs. The new gates can be used to synthesize reversible circuits of minimum "scratchpad register width" [12] (quantum register width [2]) for arbitrary reversible functions. This characteristic is of practical importance in realization of current quantum computers due to the small possible width of the scratchpad register (this width is so far limited to 7 [38]). In Section 5, we present a generalization of majority-based reversible logic gates. We formulate our problem in terms of group theory and solve it using the algebraic software GAP [31,32,36].

2. Quantum realizations of reversible majority gates

Binary reversible logic gates and circuits have been proposed in quantum, optical, CMOS, nano-mechanical and DNA realizations. Universal systems of reversible gates include Feynman gate [11] ($A = a$, $B = a \oplus b$) and some other 3×3 gates, such as Toffoli or Fredkin [10] ($A = a$, $B = \overline{a}b + ac$, $C = \overline{a}c + ab$), shown in Fig. 1. The main practical question is this: what 3×3 gate(s) should be selected to a synthesis library to make the reversible synthesis practical and the corresponding circuits realizable. Toffoli and Fredkin gates are most often realized and used in synthesis, but this is perhaps mostly for historical reasons. For instance, in quantum computing the Peres gate [26] ($A = a$, $B = a \oplus b$, $C = c \oplus (ab)$) has similar functional behavior to Toffoli gate and has a simpler realization with 2-qubit quantum primitives (3-qubit or 3×3 gates are not realizable directly in quantum, they must be built from 1-qubit and 2-qubit gates) and with cost 4 (we will discuss the cost concept later in this section). Nobody proved, however, so far, what is the best realization of any 3-qubit universal quantum gate with a given set of quantum 1-qubit and 2-qubit primitives.

Not much is known about realizations of 3×3 reversible functions in optical, CMOS, nano-mechanical and DNA realizations, except for Toffoli and Fredkin gates, so the logic synthesis researchers can only speculate about the costs of future realizations. In this case it is important to analyze from the mathematical and logical points of view what would constitute good gate families and what should be their properties in general-purpose logic synthesis and design of self-repairable fault-tolerant systems. When such gates are well understood, the experimentalists can realize them in practical circuits.

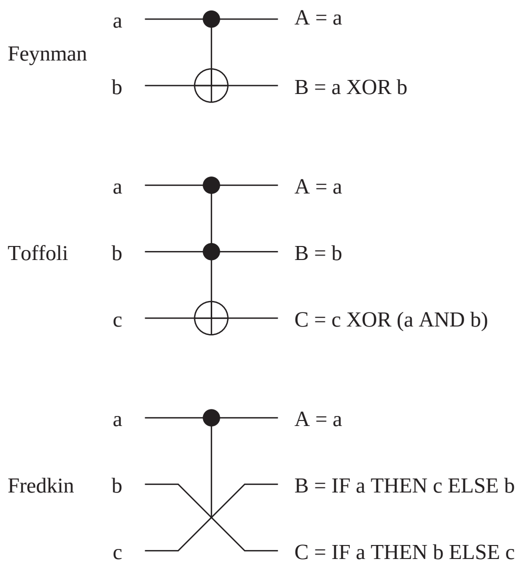

Fig. 1. Feynman, Toffoli and Fredkin gates.

Observe that so far, most researchers evaluated the complexity of reversible gates by the complexity of their binary counterparts in technologies like CMOS. The situation is different for quantum logic, in which the so-called pseudo-binary (binary permutation) circuits can be realized. Although the detailed costs depend on any particular realization technology for quantum logic circuits, where the cost relations between gates (e.g. Fredkin and Toffoli) can differ in NMR and ion trap realizations, the assumptions used in the previous work are far too approximate and should be replaced with more precise gate costs to be used in optimization and synthesis algorithms. Two assumptions were usually taken for reversible logic: (1) every $k \times k$ gate has the same cost, (2) the cost of a gate is proportional to the number of inputs/outputs. However another cost approximation has been proposed [9] and it is better suited to quantum realizations: the cost is the number of 2-qubit gates—realizable quantum primitives, regardless of their internal structures. We explain below with an example why this choice of cost is reasonable for future quantum circuit synthesis algorithms.

Using the well-known realization of Toffoli gate (in dotted rectangle from Fig. 2(a) [35], the cost of Toffoli gate in terms of 2-qubit quantum primitive gates [3,8,34] is 5. This design uses two controlled-V, one controlled-V^+ and two Feynman gates. The unitary matrix V is the “Square root of NOT”. The reader can check that since $V \times V =$ NOT and $V \times V^+ =$ I (identity), the circuit in dotted rectangle realizes the binary functions of a Toffoli gate: $A = a$, $B = b$, $C = (ab) \oplus c$.

Let us consider a function of three majorities investigated by Miller (example 2 in [23]). We realized this function with one Toffoli and four Feynman gates, found it useful in other designs and thus worthy to be a stand-alone quantum gate. We call it *the Miller gate*. As seen in Fig. 2(a), the Miller gate requires 4 Feynman gates and a Toffoli gate, which would suggest a cost of $5 + 2 \times 2 = 9$ in terms of 2-qubit quantum primitives. However,

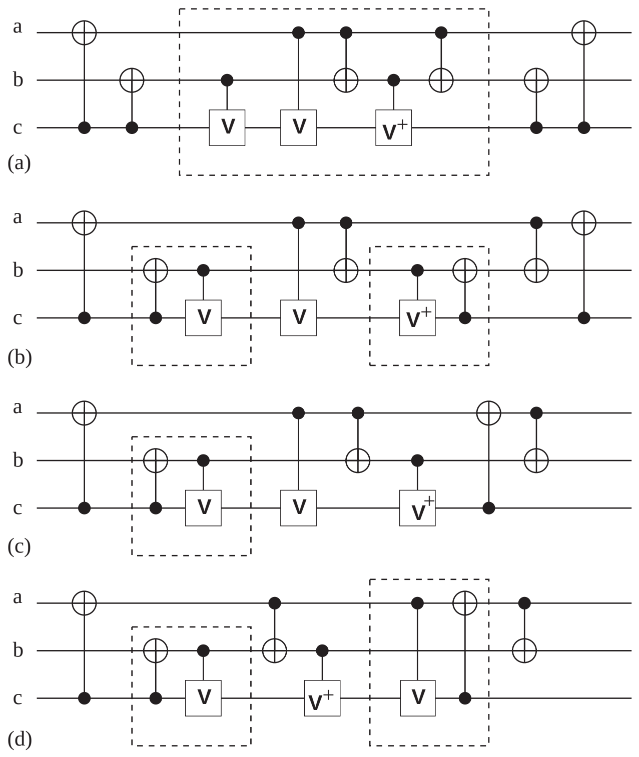

Fig. 2. Quantum realizations of Miller gate: (a) with integrated Toffoli gate—cost 9, (b) with integrated 2-qubit gates—cost 7, (c) with one integrated 2-qubit gate—cost 7, (d) with two integrated 2-qubit gates—cost 6.

using transformations and cost evaluations introduced by Smolin [35], as in Fig. 2(b), we obtain a solution with cost 7 because each dotted 2×2 subgate in Fig. 2(b) has a cost of 1 [35]. Another solution with cost 7 is shown in Fig. 2(c). It is also based on simple EXOR transformation. Further optimization [13] reveals a solution with cost 6 shown in Fig. 2(d). All the solutions from Figs. 2(b)–(d) are practically realizable and their exact costs would depend on a particular quantum circuit realization technology. So in first approximation we can assume that the smallest cost is 6. Again, the Miller gate looks initially much more complicated than the Toffoli gate (cost 5), as interpreted in a binary logic. But a closer inspection in quantum logic proves that it is just slightly more expensive with a cost of 6. In fact, it has been proved in [13] that the minimum cost of Toffoli gate is 5 and the minimum cost of Miller is 6. This example shows that it is worthy to realize pseudo-binary gates from truly quantum 2×2 primitives to evaluate more accurate costs of such gates for synthesizing circuits. This applies to all gates considered below. Such costs can be used, for instance, to approximately compare quantum realization costs of two variants of a reversible circuit—one using Toffoli and another one using Miller gates.

Let us observe, in Fig. 3(a), that by removing the three rightmost Feynman gates from the gate shown in Fig. 2(b), we obtain a new reversible gate M_2 that realizes the following

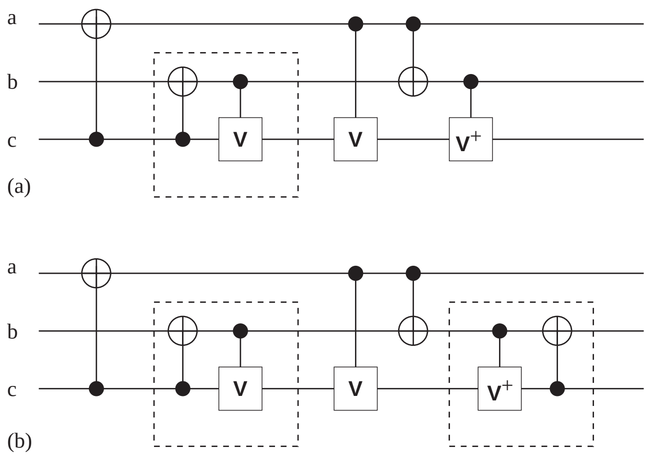

Fig. 3. Quantum realizations of majority gates of cost 5: (a) new gate M_2, (b) new gate M_3.

functions:

$$A = a \oplus c,$$
$$B = a \oplus b,$$
$$C = ab \oplus ac \oplus bc.$$

This gate has the same cost of 5 as the Toffoli gate, but has more behavior changes (outputs A and B are not simple wires from a and b anymore). Such gates should be used together with *Feynman*, *NOT* and *Toffoli* gates to obtain exact minimum quantum circuit.

The above examples show that, for quantum realization of the majority function, there is a lower cost using our new reversible logic gates M_2 and M_3 than using the Miller gate (implemented through Toffoli or other gates in Fig. 2). These new gates are examples of a new family of gates that we will introduce in Section 3. Furthermore, we will show in Section 4 that these new gates M_2 or M_3 can be used together with Feynman gates and inverters (without the need for Toffoli) to synthesize any reversible 3×3 circuit.

3. Majority based reversible logic gates

Given any n-input, n-output, (i.e., $n \times n$) reversible logic gate, we denote its inputs by $B_1, B_2, \dots, B_n$. Similarly, we denote its outputs by $P_1, P_2, \dots, P_n$. We start with some basic notions on majority.

Definition 1 (*Majority boolean function*). Given a Boolean function f_{MB} with an odd number of inputs, (i.e., $2k + 1$ inputs, where k is a natural number), f_{MB} is called a MBF when

f_{MB} returns TRUE if and only if more than half (i.e., $k+1$ or more) of its inputs are TRUE:

$$f_{MB}(B_{2k+1}, \ldots, B_2, B_1) = \sum_{1 \leqslant j_1 < j_2 < \ldots < j_k < j_{k+1} \leqslant 2k+1} Q_{j_1 j_2 \ldots j_k j_{k+1}}, \tag{1}$$

where $Q_{j_1 j_2 \ldots j_k j_{k+1}} = B_{j_1} B_{j_2} \ldots B_{j_k} B_{j_{k+1}}$.

Because we want to use majority gates repeatedly in quantum circuit design, and at the same time we do not want to increase the width, an interesting question to ask is whether a MBF can be implemented in reversible logic using the same number of inputs used by the function, i.e., without introducing any constant inputs and garbage outputs (the technique of adding constant inputs and garbage (useless) outputs is popularly used in reversible logic design as a "last resort" method). To answer this question, we first establish a necessary and sufficient condition for any Boolean function to be realizable using reversible logic without any constant inputs (also called ancilla bits or garbage inputs).

Lemma 1. *Given the truth table of any $n \times n$ reversible logic gate, the output rows must be a permutation of the input rows in the truth table.*

Proof. There are 2^n rows (patterns) for inputs, and they are all unique. There are also 2^n rows (patterns) for outputs. Since the logic gate is reversible, all output rows must be distinct. Hence we have 2^n unique rows for all n outputs as well. Thus the 2^n output rows of the truth table must be a permutation of the 2^n input rows. □

Lemma 2. *Given any single-output Boolean function f with n inputs, f can be implemented by a $n \times n$ reversible logic gate (where one of the gate outputs equals f) if-and-only-if the number of 1's and 0's are the same in the output column of the truth table for f.*

Proof. (IF) The input columns of a truth table must have the same number of 1's and 0's. If f has the same number of 1's and 0's in the output column of its truth table, we can construct a logic gate with a truth table such that the output rows are a permutation of the input rows and one of the output column is the same as the output of function f. Since the output rows are a permutation of the input rows, the logic gate is reversible. This truth table is the $n \times n$ reversible logic gate that implements f in one of its outputs.

(ONLY-IF) Let us construct a truth table with all n inputs and all n outputs of the logic gate. Using Lemma 1, the 2^n output rows of the truth table must be a permutation of the 2^n input rows. The inputs of a truth table have the same number of 1's and 0's in each column. Hence, the outputs must have the same number of 1's and 0's in each column. □

It has been shown [4] that a class of product ciphers that produce k-functions can be implemented in reversible logic without any constant inputs (a.k.a. ancilla bits). Unfortunately, these k-functions does not include MBF's. We now show that MBF's can be implemented in reversible logic without constant inputs.

Theorem 1. *Given a MBF f with $2k+1$ inputs, f can be implemented by a reversible logic gate with $2k+1$ inputs and $2k+1$ outputs, i.e., without any constant inputs.*

Proof. Since f is a MBF, its output is 1 if-and-only-if there are k or less inputs that are assigned to 0. The number of entries in the truth table with output 1 is

$$N = C_0^{2k+1} + C_1^{2k+1} + C_2^{2k+1} + \cdots + C_k^{2k+1}. \tag{2}$$

Since $C_r^n = C_{n-r}^n$, we have

$$N = C_{2k+1}^{2k+1} + C_{2k}^{2k+1} + C_{2k-1}^{2k+1} + \cdots + C_{k+1}^{2k+1}. \tag{3}$$

By adding (2) and (3), we have

$$2N = C_0^{2k+1} + \cdots + C_k^{2k+1} + C_{k+1}^{2k+1} + \cdots + C_{2k+1}^{2k+1}.$$

This is simply the sum of powers of binomial coefficients. Using the binomial theorem, we have

$$2N = \sum_{j=0}^{2k+1} C_j^{2k+1}(1)^j = (1+1)^{2k+1} = 2^{2k+1}.$$

Since there are $2k+1$ inputs, there are 2^{2k+1} entries in the truth table. From the above equation, half of these entries have output equal to 1. Thus, the other half have output equal to 0. Hence there are equal number of 1's and 0's in the output column for f. Using Lemma 2, we know that f can be implemented by a $(2k+1) \times (2k+1)$ reversible logic gate. $\square$

Now that we know a MBF can be implemented in reversible logic without constant input, we introduce a special reversible logic gate for this function.

Definition 2 (*Majority-based reversible logic gate*). A reversible logic gate is called a *majority-based reversible logic gate* (MBRLG) if it has an odd number of inputs and outputs, (i.e., $2k+1$ inputs and $2k+1$ outputs), such that at least one output is a MBF of all its inputs.

We use n-MBRLG to denote an n-input, n-output, (i.e., $n \times n$) MBRLG, where n is an odd number. For example, a 3×3 MBRLG is called a 3-MBRLG.

Theorem 2. *There are* 576 3-*MBRLG's where the last output* P_3 *is the MBF of all* 3 *inputs*.

Proof. The last output is $P_3 = B_1B_2 + B_1B_3 + B_2B_3$. Its truth table is shown in Table 1. There are $(2^3)! = 8!$ ways to permute the input rows to form the output rows in the truth table of any 3×3 reversible logic gate. The output of the MBF has four 0's and four 1's. So, there are C_4^8 combinations to arrange these four 0's and 1's. Only one of these combinations would correctly implement the MBF. Hence, the number of 3-MBRLG's with the last output being the MBF is

$$(8!) \cdot \frac{1}{C_4^8} = (4!)^2 = 576 \qquad \square$$

Theorem 3. *There are* $[(2^{2k})!]^2$ *distinct* $(2k+1)$-*MBRLG's for every natural number* k, *where the last output is the MBF*.

Table 1
Truth table of 3-input MBF

B_3	B_2	B_1	P_3
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Proof. We construct a reversible logic gate such that the last output is the MBF of all $2k+1$ inputs

$$P_{2k+1} = \sum_{1 \leqslant j_1 < j_2 < \cdots < j_k < j_{k+1} \leqslant 2k+1} Q_{j_1 j_2 \ldots j_k j_{k+1}},$$

where $Q_{j_1 j_2 \ldots j_k j_{k+1}} = B_{j_1} B_{j_2} \ldots B_{j_k} B_{j_{k+1}}$. From Lemma 2 and Theorem 1, we know that the numbers of 1's and 0's for the output of this MBF are the same. There are 2^{2k+1} entries in the truth table of this MBF. So we need to place 2^{2k} 0's in 2^{2k+1} output entries. The total number of different placements is

$$\frac{\left(2^{2k+1}\right)!}{C_{2^{2k}}^{2^{2k+1}}} = \frac{\left(2^{2k+1}\right)!}{\left[\frac{(2^{2k+1})!}{(2^{2k})!(2^{2k+1}-2^{2k})!}\right]} = \left[\left(2^{2k}\right)!\right]^2 \qquad \square$$

As a sanity check, we can use Theorem 3 when $k = 1$. With this instantiation, there are $[(2^{2(1)})!]^2 = 576$ distinct 3-MBRLG's, which agrees with Theorem 2.

We now construct a $(2k+1)$-MBRLG, MG_{2k+1}, with the following output functions:

$$\begin{aligned} P_1 &= B_1 \oplus B_{2k+1} \\ P_2 &= B_2 \oplus B_{2k+1} \\ &\vdots \\ P_{2k} &= B_{2k} \oplus B_{2k+1} \\ P_{2k+1} &= f_{MB}(B_{2k+1}, \ldots, B_1) \end{aligned}$$

In this construct, the last output is a MBF. For the first $2k$ outputs, notice that for any $1 \leqslant i, j \leqslant 2k$ such that $i \neq j$:

$$P_i \oplus P_j = (B_i \oplus B_{2k+1}) \oplus (B_j \oplus B_{2k+1}) = B_i \oplus B_j.$$

Hence XOR operations on the outputs of the above gate is the same as XOR operations on the inputs of the above gate.

Theorem 4. *MG_{2k+1} is a $(2k+1)$-MBRLG.*

Proof. The last output P_{2k+1} matches Definition 1, which means it is a MBF. There are 2^{2k+1} entries in the truth table. For each entry i, let U_i be the Boolean number encoding of the input pattern, $B_{2k+1}, B_{2k}, \dots, B_2, B_1$, in that entry, and let V_i be the Boolean number encoding of the output pattern, $P_{2k+1}, P_{2k}, \dots, P_2, P_1$, in that entry. To show reversibility, we only need to prove that $V_1, V_2, \dots, V_{2^{2k+1}}$ are different numbers. We place these numbers into two sets: $S_1 = \{V_1, \dots, V_{2^{2k}}\}$, $S_2 = \{V_{2^{2k}+1}, \dots, V_{2^{2k+1}}\}$. For S_1, the last $2k$ bits of each number $(P_{2k}, \dots, P_2, P_1)$ is a counting from $(0, 0, \dots, 0)$ to $(1, 1, \dots, 1)$. All these numbers are different. For S_2, the last $2k$ bits of each number $(P_{2k}, \dots, P_2, P_1)$ is a counting from $(1, 1, \dots, 1)$ to $(0, 0, \dots, 0)$. These numbers are also different. Thus the numbers within each set are distinct. Now we only need to prove that S_1 and S_2 are disjoint, i.e., $\forall V_i \in S_1.\ \forall V_j \in S_2.\ V_i \neq V_j$. If the last $2k$ bits of V_i and V_j are different, then they are different of course. Otherwise, the last $2k$ bits of V_i and V_j are the same. We have

$$V_i = (Q_{2k+1}, P_{2k}, P_{2k-1}, \dots, P_2, P_1),$$
$$V_j = (R_{2k+1}, P_{2k}, P_{2k-1}, \dots, P_2, P_1).$$

We will show that $Q_{2k+1} \neq R_{2k+1}$ for this case. The corresponding inputs for V_i and V_j are

$$U_i = (0, P_{2k}, P_{2k-1}, \dots, P_2, P_1),$$
$$U_j = \left(1, \overline{P_{2k}}, \overline{P_{2k-1}}, \dots, \overline{P_2}, \overline{P_1}\right).$$

Thus, for the majority function outputs, we have

$$Q_{2k+1} = f_{MB}(0, P_{2k}, P_{2k-1}, \dots, P_2, P_1),$$
$$R_{2k+1} = f_{MB}\left(1, \overline{P_{2k}}, \overline{P_{2k-1}}, \dots, \overline{P_2}, \overline{P_1}\right).$$

Let C_{ONE} be a function that counts the number of ones in its arguments. If $Q_{2k+1} = 1$, then

$$\begin{aligned} & C_{\mathrm{ONE}}\,(P_{2k}, \dots, P_2, P_1) \geqslant k+1 \\ & \Rightarrow C_{\mathrm{ONE}}\left(\overline{P_{2k}}, \dots, \overline{P_2}, \overline{P_1}\right) < (2k+1) - (k+1) = k \\ & \Rightarrow 1 + C_{\mathrm{ONE}}\left(\overline{P_{2k}}, \dots, \overline{P_2}, \overline{P_1}\right) < 1 + k \\ & \Rightarrow R_{2k+1} = 0. \end{aligned}$$

Similarly, $(Q_{2k+1} = 0) \Rightarrow (R_{2k+1} = 1)$. Hence, V_i and V_j are different. $\square$

4. Reversible logic synthesis using MBRLG

We consider logic synthesis of reversible circuits using a collection of reversible logic gates. Let us consider the following three 3-MBRLG's:

$$\begin{aligned} M_1\colon\ & P_3 = B_1B_2 + B_1B_3 + B_2B_3, \\ & P_2 = \overline{B_1}B_2 + \overline{B_1}B_3 + B_2B_3, \\ & P_1 = B_1\overline{B_2} + B_1B_3 + \overline{B_2}B_3, \\ M_2\colon\ & P_3 = B_1B_2 + B_1B_3 + B_2B_3, \\ & P_2 = B_1 \oplus B_2, \\ & P_1 = B_1 \oplus B_3, \end{aligned}$$

Table 2
Feynman (Fe_{12}) where $P_1 = B_1 \oplus B_2$ and $Fe_{12} = (3,4)(7,8)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	0	1
0	0	1	2	0	0	1	2
0	1	0	3	0	1	1	4
0	1	1	4	0	1	0	3
1	0	0	5	1	0	0	5
1	0	1	6	1	0	1	6
1	1	0	7	1	1	1	8
1	1	1	8	1	1	0	7

$$\begin{aligned} M_3\colon\ & P_3 = B_1B_2 + B_1B_3 + B_2B_3, \\ & P_2 = B_1B_2 + B_1\overline{B_3} + B_2\overline{B_3}, \\ & P_1 = B_1 \oplus B_3. \end{aligned}$$

M_1 is the Miller gate mentioned in Section 2. M_2 and M_3 are the new majority-based reversible logic gates that we originally introduced in Section 2 with lower quantum realization costs than the Miller gate. We are interested in the synthesis of reversible logic circuits using these gates.

Theorem 5. *Any* 3-*input reversible logic gate can be synthesized without input constants using Feynman gates*, *NOT gates* (*inverters*) *and gates of the type* M_1.

Theorem 6. *Any* 3-*input reversible logic gate can be synthesized without input constants using Feynman gates*, *NOT gates* (*inverters*) *and gates of the type* M_2.

Theorem 7. *Any* 3-*input reversible logic gate can be synthesized without input constants using Feynman gates*, *NOT gates* (*inverters*) *and gates of the type* M_3.

Proof. (Theorems 5–7): Using Lemma 1, the output rows of any reversible logic gate must be a permutation of all input rows. Thus, we can establish a bijective (one-to-one) mapping of all 3-input reversible logic gates (using their truth table) onto the permutation group $S_{2^3} = S_8$. We can also establish bijective mappings of each gate onto their corresponding elements in the permutation group. In the following discussion, we will use permutation elements to express the reversible gates. For example, $Fe_{12} = (3,4)(7,8)$ means that Feynman gate Fe_{12} corresponds to (3,4)(7,8). This notation (from group theory) means that output rows 3 and 4 (counting from 1 to 8) are interchanged from their corresponding input rows and similarly for rows 7 and 8. For rows that are not mentioned in the parentheses, their outputs are the same as their inputs. Tables 2–7 list the permutations of various reversible gate configurations.

Table 3
Feynman (Fe_{31}) where $P_3 = B_1 \oplus B_3$ and $Fe_{31} = (2,6)(4,8)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	0	1
0	0	1	2	1	0	1	6
0	1	0	3	0	1	0	3
0	1	1	4	1	1	1	8
1	0	0	5	1	0	0	5
1	0	1	6	0	0	1	2
1	1	0	7	1	1	0	7
1	1	1	8	0	1	1	4

Table 4
NOT gate for input 1: $n_1 = (1,2)(3,4)(5,6)(7,8)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	1	2
0	0	1	2	0	0	0	1
0	1	0	3	0	1	1	4
0	1	1	4	0	1	0	3
1	0	0	5	1	0	1	6
1	0	1	6	1	0	0	5
1	1	0	7	1	1	1	8
1	1	1	8	1	1	0	7

Table 5
3-MBRLG: $M_1 = (4,5)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	0	1
0	0	1	2	0	0	1	2
0	1	0	3	0	1	0	3
0	1	1	4	1	0	0	5
1	0	0	5	0	1	1	4
1	0	1	6	1	0	1	6
1	1	0	7	1	1	0	7
1	1	1	8	1	1	1	8

We can generate subgroups using the above defined elements as generators:
g_1 = Group generated by $Fe_{12}, Fe_{31}, n_1, M_1$.
g_2 = Group generated by $Fe_{12}, Fe_{31}, n_1, M_2$.
g_3 = Group generated by $Fe_{12}, Fe_{31}, n_1, M_3$.

Table 6
3-MBRLG: $M_2 = (2, 4, 6, 7, 8, 5)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	0	1
0	0	1	2	0	1	1	4
0	1	0	3	0	1	0	3
0	1	1	4	1	0	1	6
1	0	0	5	0	0	1	2
1	0	1	6	1	1	0	7
1	1	0	7	1	1	1	8
1	1	1	8	1	0	0	5

Table 7
3-MBRLG: $M_3 = (2, 4, 8, 7, 6, 5)$

Inputs				Outputs			
B_3	B_2	B_1	Encoding	P_3	P_2	P_1	Encoding
0	0	0	1	0	0	0	1
0	0	1	2	0	1	1	4
0	1	0	3	0	1	0	3
0	1	1	4	1	1	1	8
1	0	0	5	0	0	1	2
1	0	1	6	1	0	0	5
1	1	0	7	1	0	1	6
1	1	1	8	1	1	0	7

Using GAP software [31], we can compute the size (number of permutations) for each group:

$|S_8| = 40320,\ |g_1| = 40320,\ |g_2| = 40320,\ |g_3| = 40320.$

Since $|S_8| = |g_1| = |g_2| = |g_3|$, and $|S_8|$ contains all possible permutations for 3-input reversible gates, we know that our theorem holds. □

Notice that without M_1 or M_2 or M_3, the size of the subgroup generated by Fe_{12}, Fe_{31}, n_1, is only 32, which is smaller than $|S_8|$. Even if we factor in other variants of Feynman gates and inverters (by exchanging wire configurations), the generated group size is only 1344, which is still less than $|S_8|$. So using Feynman gate along with inverters (NOT gates) is not sufficient to synthesize all 3-input reversible logic functions. It is necessary to have some other additional gate, and we concentrate on majority gates such as M_1, M_2 or M_3. Furthermore, we have demonstrated in Section 2 that gates M_2 and M_3 have lower cost than the Miller gate (implemented through Toffoli or 2-qubit gates) in terms of quantum realizations of majority function.

We can also use 5-MBRLG's for logic synthesis. Consider the following gate where the last output is the MBF:

$$\begin{aligned} MG_5: \ & P_5 = f_{MB}(B_1, B_2, B_3, B_4, B_5), \\ & P_i = B_i \oplus B_5 \text{ for } i = 1, \ldots, 4. \end{aligned}$$

The element corresponding to the truth table of this gate is

$$\begin{aligned} MG_5 = {} & (8, 24, 25)(12, 28, 21)(14, 30, 19)(15, 31, 18) \\ & (16, 32, 17)(20, 29)(22, 27)(23, 26). \end{aligned} \tag{4}$$

Similarly, we can also find the subgroup for the exchangers (SWAP gates) and inverter (NOT gate):

$$\begin{aligned} e_{12} = {} & (2, 3)(6, 7)(10, 11)(14, 15) \\ & (18, 19)(22, 23)(26, 27)(30, 31), \\ e_{23} = {} & (3, 5)(4, 6)(11, 13)(12, 14) \\ & (19, 21)(20, 22)(27, 29)(28, 30), \\ e_{34} = {} & (5, 9)(6, 10)(7, 11)(8, 12) \\ & (21, 25)(22, 26)(23, 27)(24, 28), \\ e_{45} = {} & (9, 17)(10, 18)(11, 19)(12, 20) \\ & (13, 21)(14, 22)(15, 23)(16, 24), \\ n_1 = {} & (1, 2)(3, 4)(5, 6)(7, 8)(9, 10)(11, 12) \\ & (13, 14)(15, 16)(17, 18)(19, 20)(21, 22) \\ & (23, 24)(25, 26)(27, 28)(29, 30)(31, 32). \end{aligned}$$

Theorem 8. *Any* 5-*input reversible logic gate can be synthesized using SWAP gates* (2×2 *gate that exchanges the two wires*), *NOT gates and gates of the type* MG_5.

Proof. The size of the group generated by $e_{12}, e_{23}, e_{34}, e_{45}, n_1$, and MG_5 is equal to 32!, which is the same as the size of the symmetry group S_{32}. □

Theorem 9. *Any* 5-*input reversible logic gate can be synthesized using Feynman gates, NOT gates and gates of the type* MG_5.

Proof. SWAP gates can be synthesized by Feynman gates. Thus we can deduce this theorem from Theorem 8. □

5. Generalized majority-based reversible logic gates

We can invert any inputs of a majority Boolean function, resulting in generalized majority boolean function (GMBF). The GMBF is formed by composing a MBF with zero or more inverters at its inputs.

For example, given a 3-input MBF,

$$f_{MB}(B_1, B_2, B_3) = B_1B_2 + B_1B_3 + B_2B_3.$$

We can invert its input B_2 to form a 3-input GMBF,

$$\begin{aligned} g_{MB}(B_1, B_2, B_3) &= f_{MB}\left(B_1, \overline{B_2}, B_3\right) \\ &= B_1\overline{B_2} + B_1B_3 + \overline{B_2}B_3. \end{aligned}$$

Since we can invert 0, 1, 2 or 3 inputs of the MBF, the total number of distinct 3-input GMBF's is: $C_0^3 + C_1^3 + C_2^3 + C_3^3 = 8$.

Definition 3 (*Generalized majority-based reversible logic gate*). A reversible logic gate is called a generalized majority-based reversible logic gate (GMBRLG) if it has an odd number of inputs and outputs, (i.e., $2k + 1$ inputs and $2k + 1$ outputs), such that at least one output is a GMBF of all its inputs.

We use n-GMBRLG to denote an n-input, n-output, (i.e., $n\times n$) GMBRLG, where n is an odd number. For example, a 3×3 GMBRLG is called a 3-GMBRLG.

Theorem 10. (i) *There are* 4608 3-*GMBRLG's where the last output* P_3 *is a GMBF*. (ii) *There are* 192 3-*GMBRLG's where every output is a GMBF*. (iii) *There are* 72 3-*MBRLG's where one output is a MBF and the other two outputs are GMBF's*.

Proof. (i) In Theorem 2, we showed there are 576 3-MBRLG's where the last output P_3 is the MBF of all inputs. In the example above, we also showed that the 3-input MBF is one of the 8 GMBF's. Thus the total number of 3-GMBRLG's is $8 \times 576 = 4608$.

(ii) There are 8 choices (3-input GMBF's) for the first output. When the first output function is determined, we can look at the truth table and realize that 6 out of the remaining 7 GMBF's can be used for the second output and the gate would still be reversible. For the last output, we use the truth table again and realize that 4 out of the remaining 6 choices (GMBF's) can be used and the gate is still reversible. Hence $8 \times 6 \times 4 = 192$.

(iii) We have 3 choices (3 outputs) to assign the MBF. After that, we have 6 GMBF's for the second output and 4 GMBF's for the third output. Thus $3 \times 6 \times 4 = 72$. □

6. Conclusion

We generalized the Miller gate concept to new families of gates and showed that any Boolean majority function with $2k+1$ inputs can be implemented in a reversible logic gate without additional garbage (constant) inputs. We introduced new gates with very low cost in quantum realization (better than the Miller gate) for computing the majority function. We proved that our low cost gates can be used to synthesize reversible circuit with inverters and Feynman gates. The small possible width of the scratchpad register continues and will continue to be one of the most difficult barriers to overcome. The families of gates introduced here enable us to realize quantum logic circuits with the smallest possible width.

References

[1] A. Al-Rabadi, Novel methods for reversible logic synthesis and their application to quantum computing, Ph.D. Thesis, Portland State University (October 2002).
[2] A. Al-Rabadi, L. Casperson, M. Perkowski, X. Song, Multiple-valued quantum logic, in: Proc. ULSI, 2002.
[3] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.A. Smolin, H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A 52 (5) (1995) 3457–3467.
[4] D. Coppersmith, E. Grossman, Generators for Certain Alternating Groups with Applications to Cryptography, SIAM J. Appl. Math. 29 (4) (1975) 624–627.
[5] A. De Vos, Reversible and endoreversible computing, Internat. J. Theoret. Phys. 34 (1995) 2251–2266.
[6] A. De Vos, Reversible computing, Progr. Quant. Electron 23 (1999) 1–49.
[7] A. De Vos, B. Raa, L. Storme, Generating the group of reversible logic gates, J. Phys. A: Math. Gen. 35 (2002) 7063–7078.
[8] D. Deutsch, Quantum computational networks, Roy. Soc. London Ser. A 425 (1989) 73–90.
[9] D.P. DiVincenzo, Two-bit gates are universal for quantum computation, Phys. Rev. A 51 (1995) 1015–1022.
[10] G. W. Dueck, D. Maslov, Reversible function synthesis with minimum garbage outputs, in: Proc. Sixth Internat. Symp. on Representations and Methodology of Future Computing Technologies (RM 2003), 2003.
[11] R.P. Feynman, Quantum mechanical computers, Opt. News 11 (1985) 11–20.
[12] E. Fredkin, T. Toffoli, Conservative logic, Internat. J. Theoret. Phys. 21 (1982) 219–253.
[13] W.N.N. Hung, X. Song, G. Yang, J. Yang, M.Perkowski, Quantum Logic Synthesis by Symbolic Reachability Analysis, in: Proc. 41st ACM/IEEE Design Automation Conference, 2004, pp. 838–841.
[14] K. Iwama, Y. Kambayashi, S. Yamashita, Transformation rules for designing cnot-based quantum circuits, in: Proc. Design Automation Conf., 2002.
[15] P. Kerntopf, Maximally efficient binary and multi-valued reversible gates, in: Proc. ULSI Workshop, 2001, pp. 55–58.
[16] P. Kerntopf, Synthesis of multipurpose reversible logic gates, in: Proc. EUROMICRO Symp. on Digital Systems Design, 2002, pp. 259–266.
[17] M.H.A. Khan, M. Perkowski, Multi-output esop synthesis with cascades of new reversible gate family, in: Proc. Sixth Internat. Symp. on Representations and Methodology of Future Computing Technologies (RM 2003), 2003.
[18] A. Khlopotine, M. Perkowski, P. Kerntopf, Reversible logic synthesis by gate composition, in: Proc. IEEE/ACM Internat. Workshop on Logic Synthesis, 2002, pp. 261–266.
[19] F. Luccio, L. Pagli, On a new boolean function with applications, IEEE Trans. Comput. 48 (3) (1999) 296–310.
[20] M. Lukac, M. Pivtoraiko, A. Mishchenko, M. Perkowski, Automated synthesis of generalized reversible cascades using genetic algorithms, in: Proc. Fifth Internat. Workshop on Boolean Problems, 2002, pp. 33–45.
[21] D. Maslov, G.W. Dueck, Garbage in reversible designs of multiple-output functions, in: Proc. Sixth Internat. Symp. on Representations and Methodology of Future Computing Technologies (RM 2003), 2003.
[22] D.M. Miller, Spectral and two-place decomposition techniques in reversible logic, in: Proc. IEEE Midwest Symp. on Circuits and Systems, 2002.
[23] D.M. Miller, G.W. Dueck, Spectral techniques for reversible logic synthesis, in: Proc. Sixth Internat. Symp. on Representations and Methodology of Future Computing Technologies (RM 2003), 2003.
[24] A. Mishchenko, M. Perkowski, Logic synthesis of reversible wave cascades, in: Proc. IEEE/ACM Internat. Workshop on Logic Synthesis, 2002, pp. 197–202.
[25] L. Nazhandali, K.A. Sakallah, Majority-based decomposition of carry logic in binary adders, in: Proc. IEEE/ACM Internat. Workshop on Logic Synthesis, 2002.
[26] A. Peres, Reversible logic and quantum computers, Phys. Rev. A 32 (1985) 3266–3276.
[27] M. Perkowski, L. Jozwiak, P. Kerntopf, A. Mishchenko, A. Al-Rabadi, A. Coppola, A. Buller, X. Song, M. M. H. A. Khan, S. Yanushkevich, V. Shmerko, M. Chrzanowska-Jeske, A general decomposition for reversible logic, in: Proc. Internat. Workshop on Applications of the Reed–Müller Expansion in Circuit Design, 2001, pp. 119–138.
[28] M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola, B. Massey, Regularity and symmetry as a base for efficient realization of reversible logic circuits, in: Proc. IEEE/ACM Internat. Workshop on Logic Synthesis, 2001, pp. 90–95.

[29] M. Perkowski, P. Kerntopf, A. Buller, M. Chrzanowska-Jeske, A. Mishchenko, X. Song, A. Al-Rabadi, L. Jozwiak, A. Coppola, B. Massey, Regular realization of symmetric functions using reversible logic, in: Proc. EUROMICRO Symp. on Digital Systems Design, 2001, pp. 245–252.
[30] P. Picton, A universal architecture for multiple-valued reversible logic, Multiple Valued Logic: Internat. J. 5 (2000) 27–37.
[31] M. Schoenert, Gap, Comput. Algebra Nederland Nieuwsbrief 9 (1992) 19–28.
[32] M. Schönert, et al., GAP—Groups, Algorithms, and Programming, Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany, fifth ed., 1995.
[33] V.V. Shende, A.K. Prasad, I.L. Markov, J.P. Hayes, Reversible logic circuit synthesis, in: Proc. IEEE/ACM Internat. Conf. on Computer Aided Design, 2002, pp. 353–360.
[34] T. Sleator, H. Weinfurter, Realizable universal quantum logic gates, Phys. Rev. Lett. 74 (20) (1995) 4087–4090.
[35] J.A. Smolin, D.P. DiVincenzo, Five two-bit quantum gates are sufficient to implement the quantum Fredkin gate, Phys. Rev. A 53 (1996) 2855–2856.
[36] L. Storme, A. De Vos, G. Jacobs, Group theoretical aspects of reversible logic gates, J. Universal Comput. Sci. 5 (1999) 307–321.
[37] C.E. Stroud, Reliability of majority voting based VLSI fault-tolerant circuits, IEEE Trans. VLSI Systems 2 (4).
[38] L.M.K. Vandersypen, M. Steffen, G. Breyta, C.S. Yannoni, M.H. Sherwood, I.L. Chuang, Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance, Nature 414 (2001) 883–887.