SN Computer Science (2022) 3:40
https://doi.org/10.1007/s42979-021-00900-5

ORIGINAL RESEARCH

Optimization of Reversible Circuits Using Gate Pair Classification

P. Sai Phaneendra[1] · Chetan Vudadha[2] · M. B. Srinivas[2]

Received: 29 July 2021 / Accepted: 21 September 2021 / Published online: 2 November 2021

Abstract
Research on reversible logic gained momentum in the past decade owing to its utility in emerging areas such as quantum computing, optical computing and low power circuit implementation, etc. Reversible circuits synthesized using existing techniques often tend to be sub-optimal; thus, post-synthesis optimization techniques are usually employed to reduce the 'circuit cost', a metric used to compare the reversible circuits. In this paper, a set of optimization techniques is proposed to minimize the circuit cost. These techniques rely on classifying a pair of reversible gates in a given circuit based on their structural similarity. An algorithm that maps the classifications with the optimization techniques to improve the cost of a circuit is also proposed. Results obtained for a set of benchmark reversible circuits confirm that the proposed methodology performs better in terms of circuit cost when compared to those available in the literature.

Keywords Reversible Logic · Optimization · Synthesis · Optimization Techniques · Reduction Rules

Introduction

Research on reversible logic has been motivated by its applications in emerging technologies like quantum computing because unitary transformation in quantum computing was shown to be reversible [1], optical computing [2–4] and the need for ultra low power design [5–7]. Since a reversible logic function has one-to-one and onto mapping between its inputs and outputs, it rules out fan-out and feedback which are used extensively in conventional logic circuits. This lead to research on synthesis techniques exclusively for reversible logic in the past decade [8, 9]. These techniques can be categorized broadly as (i) exact and (ii) heuristic. Exact techniques search for optimal solutions to generate minimum cost circuits but are applicable only for small functions (where number of variables typically does not exceed 6) [9]. For functions with a large number of variables, heuristic techniques have been proposed. These however generate circuits that are sub-optimal in terms of cost and thus have scope for improvement. Thus, post-synthesis optimization techniques [10–14] have been developed to reduce the cost of synthesized reversible circuits.

The technique presented in [14] reduces the cost of a reversible circuit with the help of additional lines. However, the increase in the number of lines directly affects the complexity of its implementation [15]. Technique described in [10, 16] uses template-based local optimization to reduce the cost of the circuit. In these approaches, different templates are applied on a set of gates. However, this method involves a search for templates, which becomes complex as the circuit size increases. Rule-based optimization algorithms have been presented in [11–13, 17], a simulated annealing-based approach using transformation rules has been developed in [18] and a shared cube-based approach is presented in [19]. In all of these approaches, a group of gates is replaced with a lesser cost gate netlist using a set of rules. However when a pair of gates is considered, these techniques are applicable only for a few combinations of gate pairs.

A lookup table (LUT)-based hierarchical synthesis approach is presented in [20] where the Boolean logic is synthesized using k input LUT (termed as k-LUT) network. These LUTs are converted to reversible circuits using techniques such as Exclusive-OR Sum-of-Product (ESOP) decomposition [21] and reduce the cost of the circuit by

This article is part of the topical collection "Hardware for AI, Machine Learning and Emerging Electronic Systems" guest edited by Himanshu Thapliyal, Saraju Mohanty and VS Kanchana Bhaaskaran.

✉ M. B. Srinivas
mbs@hyderabad.bits-pilani.ac.in

[1] Mediatek, Bangalore, India

[2] EEE Department, BITS-Pilani, Hyderabad Campus, Hyderabad, India

applying rule-based techniques. Recently some optimisation technique has been proposed in [22, 23] to map the *k*-LUTs to a equivalent set of gates targeting to reduce the Clifford+T gates, a quantum gate library which is a more popular metric currently because of its application in fault tolerant quantum computing.

In the existing optimization techniques, the rules or transformations are applicable only to a few combinations of gate pairs. In this paper, a framework that includes transformations that are applicable for several combinations of gate pairs is proposed. In this framework, initially a methodology to classify a pair of gates in a reversible circuit is proposed. This methodology is then used to identify the gate pairs that can be optimized using the existing techniques as opposed to those that cannot be. For these gate pairs that cannot be optimized further, techniques to transform them to equivalent lower cost gate netlists are proposed. Finally, an algorithm that uses these techniques to reduce the total cost of the reversible circuit is presented. A set of benchmark reversible circuits has been employed to evaluate the performance of this framework.

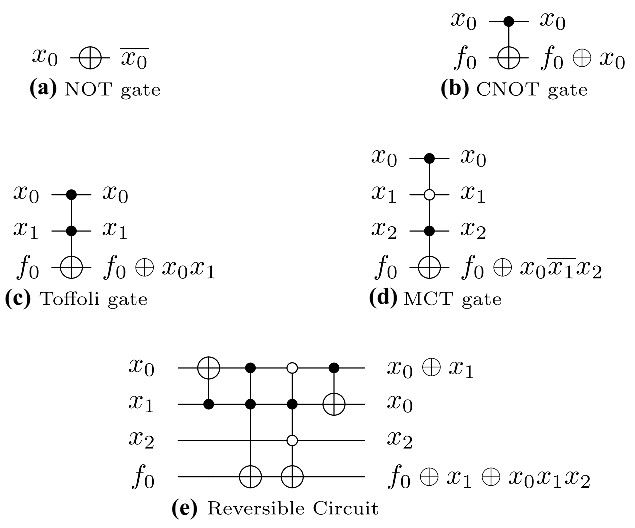
Background

Reversible Logic Circuits

A function *f* is called a *reversible logic function* if there is a one-to-one correspondence, i.e, bijective mapping between inputs and outputs. A reversible logic circuit for a function can be realized by cascading reversible gates like NOT, CNOT, Toffoli and multiple-control Toffoli (MCT) gates [24]. The standard representation of basic reversible gates and a cascade of these gates to form a reversible circuit is shown in Fig. 1. An MCT gate is represented as *MCT*(*C*; *t*) where *C* is a set of control lines and *t* is the target line. When all the positive (negative) control lines of the gate, represented as •(∘), are set to ‘1’ (‘0’), an MCT gate inverts the target line that is represented as ⊕. MCT gates with control lines as 0, 1 and 2 are termed as NOT, CNOT and Toffoli gates, respectively.

Definitions used in this paper for different lines of a reversible circuit are given below:

Control Connection	The *control connection* defines the state of a connection, i.e., positive or negative, of a given control line in a gate.
Equal Control Line	A line having same control connection and same value for a set of gates is termed as the *equal control line (ECL)*.
Complementary Control Line	A line having positive control connection on one gate and negative control connection on another gate is termed as *complementary control line (CCL)* for the gate pair.
Unused Line	A line which is neither a control line nor a target line of a gate is termed as *unused line* for that gate.



x_0 $\overline{x_0}$
(a) NOT gate
x_0 x_0 f_0 $f_0 \oplus x_0$
(b) CNOT gate
x_0 x_1 f_0 x_0 x_1 $f_0 \oplus x_0x_1$
(c) Toffoli gate
x_0 x_1 x_2 f_0 x_0 x_1 x_2 $f_0 \oplus x_0\overline{x_1}x_2$
(d) MCT gate
x_0 x_1 x_2 f_0 $x_0 \oplus x_1$ x_0 x_2 $f_0 \oplus x_1 \oplus x_0x_1x_2$
(e) Reversible Circuit

Fig. 1 Reversible gates and reversible circuit

Cost Metrics

Typically, two different cost models are used to evaluate and compare reversible circuits, (i) NCT library cost model and (ii) quantum cost model.

NCT Library Cost Model

The NCT library consists of NOT, CNOT and Toffoli gates. MCT gates in a reversible circuit with more than two control lines are decomposed into NCT gates using the approach presented in [25]. The cost of an MCT gate in terms of Toffoli gates is given in Eq. (1) where *c* and *n* denote the number of control lines of a gate and the total number of lines in a circuit, respectively. The NCT cost of an entire circuit is therefore the sum of all the NCT gates after decomposing each gate.

$$\text{Cost}_T(\text{MCT}) = \begin{cases} 8(c-3) \text{ if } c > \left\lceil \frac{n+1}{2} \right\rceil \\ 4(c-2) \text{ if } c \le \left\lceil \frac{n+1}{2} \right\rceil. \end{cases} \quad (1)$$

Quantum Cost Model

The Quantum Cost (QC) of a reversible gate is the number of primitive quantum gates required to implement the gate functionality. The primitive gates of quantum library used in this paper are NOT, CNOT, controlled V and controlled V+ gates [25]. The QC of a reversible gate is calculated using RevLib [26] and is given in Table 1 where c denotes the number of control line and n denotes the total number of lines in the circuit. When all control lines of an MCT gate are negative, QC is increased by a value of 2.

QC of each gate in a circuit is summed up to calculate the QC of that circuit. As an example, QC of the circuit shown in Fig. 1e is $1 + 5 + 13 + 1 = 20$.

Quadruple Representation for a Pair of Gates

Several techniques exist in the literature to reduce the quantum cost of a given gate pair. These techniques are chosen based on the control connections of gate pair control lines. However, there exists no methodology by which a pair of gates can be classified into different classes based on the control connections of each gate in that pair. Such a classification helps in identifying gate pairs to which existing optimization techniques can be applied as opposed to those that do not have optimization defined on them yet. In this paper, we propose a representation named *quadruple representation (QR)* to classify gate pairs, as given below:

The QR for a gate pair, g_i and g_j, is denoted as $\text{QR}(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ where,

- α: Number of equal control lines in the gate pair.
- β: Number of complementary control lines in the gate pair.
- γ: Number of control lines that have control connection only in the gate g_i.
- δ: Number of control lines that have control connection only in the gate g_j.

For a gate pair g_i and g_j, the sets which contain control connections corresponding to α, β, γ and δ are defined as K, C, P and Q, respectively. QR can be understood with the help of following example:

Example 1 Consider two gates g_1 and g_2 shown in Fig. 2. The gates have two equal control lines, $K = \{x_0, x_3\}$, one complementary control line, $C = \{x_1\}$, a line with control connection only on gate g_1, that is, $P = \{x_2\}$ and a line with control connection only on gate g_2, that is, $Q = \{x_4\}$. Thus, according to the QR given above,

$\text{QR}(g_1, g_2) = (2, 1, 1, 1)$.

Lemma 1 stated below provides the possible number of classifications for a gate pair when at least one of the elements of the QR is not equal to zero. If all the elements in the QR of a gate pair are zero, then the gate pair is a cascade of two NOT gates that can be removed from the circuit.

Lemma 1 *In a reversible circuit, there exist* 15 ***classifications*** *for a gate pair g_i and g_j, where at least one of the elements* ***in*** $\text{QR}(g_i, g_j)$ *is a non-zero element.*

Proof The $\text{QR}(g_i, g_j)$ is a quadruple with four elements α, β, γ and δ. Depending on the elements that are either zero or non-zero, there exist $2^4 = 16$ unique classifications. Of these, one classification has all its elements as zero. Thus, the number of classifications in which at least one of the elements is non-zero is $16 - 1 = 15$. □

Table 1 Quantum Cost of an MCT gate

c	QC		
	$(n-(c+1)) \geq$		
	0	1	$c-2$
0	1		
1	1		
2	5		
3	13		
4	29		26
5	61	52	38
6	125	80	50
7	253	100	62
8	509	128	74
9	1021	152	86
> 9	$2^{c+1}-3$	$24(c+1)-88$	$12(c+1)-34$

where c is number of control lines in an MCT gate; n is total number of lines in the circuit

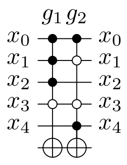


Fig. 2 Example for quadruple representation (QR)

An example for each of the classifications along with their QR is shown in Fig. 3.

There exist several rules in the literature to optimize some or a subset of few QR classifications as given below:

Rule R1 If $\mathrm{QR}(g_i, g_j) = (\alpha, 0, 0, 0)$, then the gates are identical and can be removed from the circuit using the *deletion rule* presented in [27].

Rule R2 If $\mathrm{QR}(g_i, g_j) = (\alpha, \beta, 0, 0)$ or $(0, \beta, 0, 0)$, then the gates can be decomposed into a network of smaller gates by applying the *rule based optimization* presented in [11].

Rule R3 If $\mathrm{QR}(g_i, g_j) = (\alpha, 1, 0, 0)$ or $(\alpha, 0, 1, 0)$ or $(\alpha, 0, 0, 1)$, then the gates can be merged into a single gate using the *merging rule* presented in [12].

Rule R4 If $\mathrm{QR}(g_i, g_j) = (\alpha, 1, 1, 0)$ or $(\alpha, 1, 0, 1)$, then the gates can be replaced with a lesser cost gate netlist using the *replacement rule* presented in [12].

Rule R5 If $\mathrm{QR}(g_i, g_j) = (\alpha, 0, \gamma, 1)$ or $(\alpha, 0, 1, \delta)$, then the cost of the gate pair can be reduced using the *cube pairing* technique presented in [17].

An example for each of the rules discussed above is given in Fig. 4. As mentioned earlier, these rules can be applied only on a few QR classifications. For example, classifications like $(\alpha, \beta, \gamma, \delta)$, $(\alpha, \beta, \gamma, 0)$, $(\alpha, \beta, 0, \delta)$, $(\alpha, 0, \gamma, \delta)$, $(\alpha, 0, \gamma, 0)$ and $(\alpha, 0, 0, \delta)$ do not have any optimization defined on them. In what follows new optimization techniques that are applicable to most of the classifications are described.

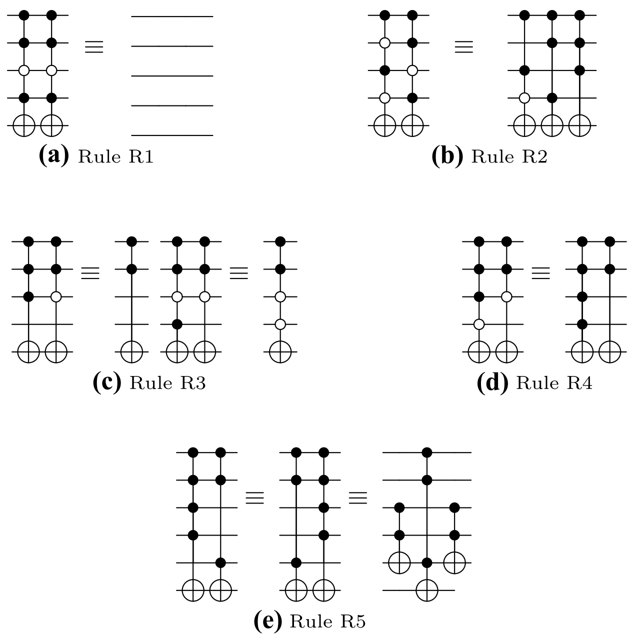
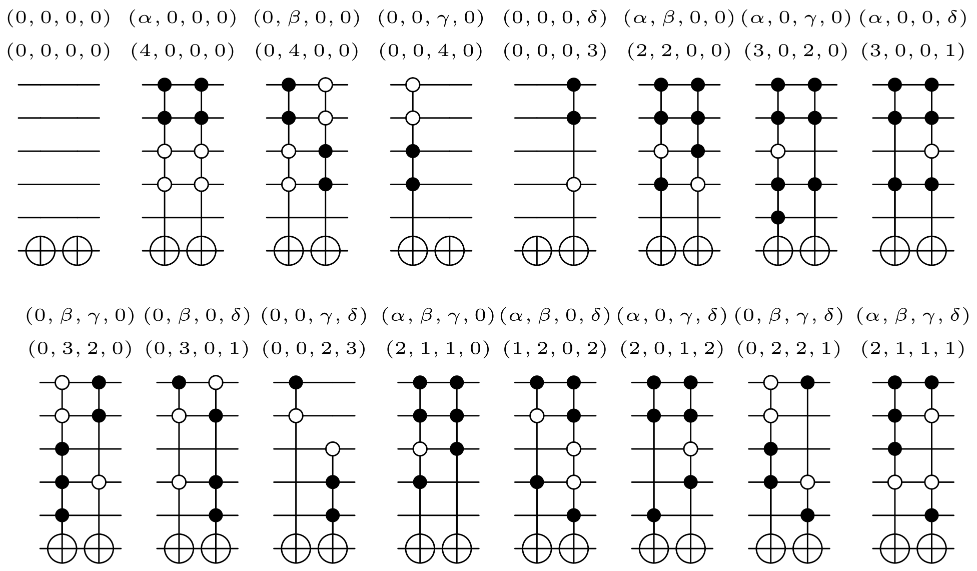


Fig. 4 Examples for Existing Optimization Rules

Optimization Techniques for Different QR Classifications

The first technique is based on a decomposition rule that decomposes a pair of gates into a network of smaller gates. This technique can be applied on the classifications where equal control lines are present between a pair of gates, i.e., $\alpha > 0$. Further, a complementary control line (CCL) transformation technique is presented to reduce the number of CCLs in a given gate pair. This technique is then used to

Fig. 3 Examples for different QR classifications


(0, 0, 0, 0) (0, 0, 0, 0)
(α, 0, 0, 0) (4, 0, 0, 0)
(0, β, 0, 0) (0, 4, 0, 0)
(0, 0, γ, 0) (0, 0, 4, 0)
(0, 0, 0, δ) (0, 0, 0, 3)
(α, β, 0, 0) (2, 2, 0, 0)
(α, 0, γ, 0) (3, 0, 2, 0)
(α, 0, 0, δ) (3, 0, 0, 1)
(0, β, γ, 0) (0, 3, 2, 0)
(0, β, 0, δ) (0, 3, 0, 1)
(0, 0, γ, δ) (0, 0, 2, 3)
(α, β, γ, 0) (2, 1, 1, 0)
(α, β, 0, δ) (1, 2, 0, 2)
(α, 0, γ, δ) (2, 0, 1, 2)
(0, β, γ, δ) (0, 2, 2, 1)
(α, β, γ, δ) (2, 1, 1, 1)

transform the pair of gates with $\alpha = 0$ such that the decomposition rule can be applied. Finally, a set of optimization techniques is proposed for specific QR classifications.

Optimization Technique for QR$(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ when $\alpha > 0$

This technique is based on a decomposition rule that decomposes a pair of gates using the number of equal lines (α) between the gate pair. This rule is described by Lemma 2 given below and is applicable for QR classifications in which $\alpha > 0$, i.e., $(\alpha, 0, \gamma, 0)$, $(\alpha, 0, 0, \delta)$, $(\alpha, 0, \gamma, \delta)$, $(\alpha, \beta, \gamma, 0)$, $(\alpha, \beta, 0, \delta)$ and $(\alpha, \beta, \gamma, \delta)$.

Lemma 2 (Rule R6—Decomposition Rule) : *Consider a pair of gates g_i and g_j with a same target line x_t. The QR$(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ and the sets of control lines are represented as K, C, P and Q. If an unused line 'x_u' is available, then the gate pair can be decomposed into a network of smaller gates as follows*:

$$g_i \circ g_j = \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup P;x_u) \circ \text{MCT}(C \cup Q;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u) \tag{2}$$

Proof The gates g_i and g_j are represented as $g_i = \text{MCT}(K \cup C \cup P;x_t)$ and $g_j = \text{MCT}(K \cup C \cup Q;x_t)$

According to [25], an MCT gate of size m (where $m \geqslant 5$) can be decomposed into a network of two gates of size p and two gates of size $m - p + 1$ placed alternatively, given at least one unused line in the circuit. Considering x_u as unused line, the gates g_i and g_j can be decomposed as follows:

$$\begin{aligned} g_i &= \text{MCT}(K \cup C \cup P;x_t) \\ &= \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \\ &\quad \circ \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \\ g_j &= \text{MCT}(K \cup C \cup Q;x_t) \\ &= \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u) \\ &\quad \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u) \end{aligned}$$

Therefore, cascading gates g_i and g_j are given as below:

$$\begin{aligned} g_i \circ g_j &= \text{MCT}(K \cup C \cup P;x_t) \circ \text{MCT}(K \cup C \cup Q;x_t) \\ &= \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \\ &\quad \circ \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \\ &\quad \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u) \\ &\quad \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u) \end{aligned} \tag{3}$$

According to the deletion rule given in [27], a pair of gates which is equal and adjacent can be removed from the circuit. Thus the cascade of gates, $\text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(K \cup \{x_u\};x_t)$, above is removed and the reduced circuit is given as,

$$\begin{aligned} g_i \circ g_j &= \text{MCT}(C \cup P;x_u) \circ \text{MCT}(K \cup \{x_u\};x_t) \\ &\quad \circ \text{MCT}(C \cup P;x_u) \circ \text{MCT}(C \cup Q;x_u) \\ &\quad \circ \text{MCT}(K \cup \{x_u\};x_t) \circ \text{MCT}(C \cup Q;x_u). \end{aligned} \tag{4}$$

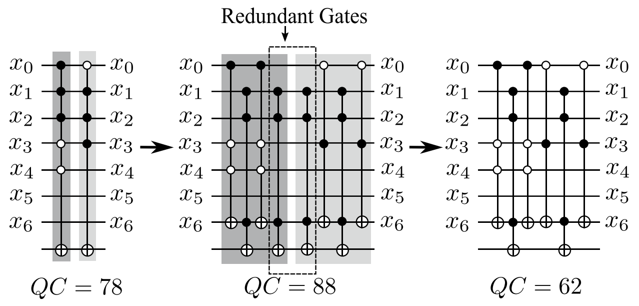


Fig. 6 Example for QR$(g_i, g_j) = (\alpha, \beta, \gamma, 0) = (2, 2, 1, 0)$

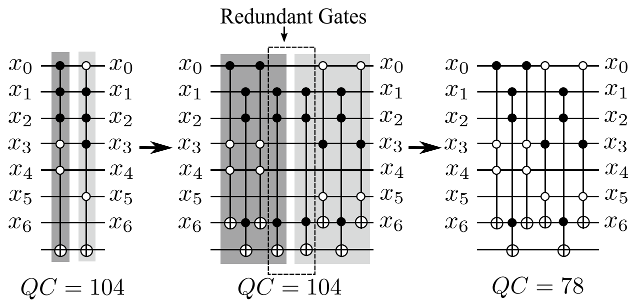


Fig. 5 Example for QR$(g_i, g_j) = (\alpha, \beta, \gamma, \delta) = (2, 2, 1, 1)$

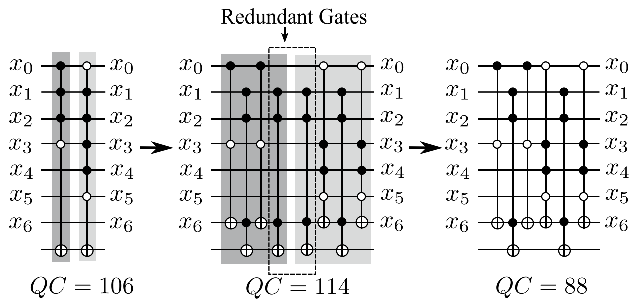


Fig. 7 Example for QR$(g_i, g_j) = (\alpha, \beta, 0, \delta) = (2, 2, 0, 2)$

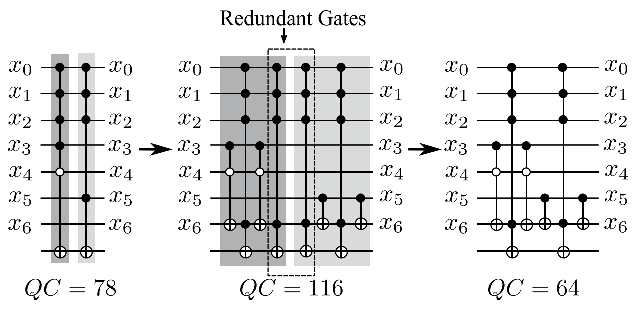


Fig. 8 Example for $\mathrm{QR}(g_i, g_j) = (\alpha, 0, \gamma, \delta) = (3, 0, 2, 1)$

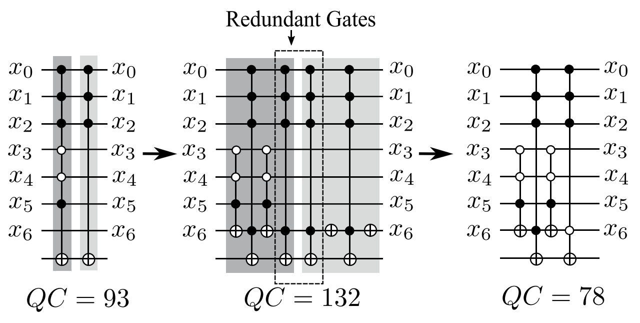


Fig. 9 Example for $\mathrm{QR}(g_i, g_j) = (\alpha, 0, \gamma, 0) = (3, 0, 3, 0)$

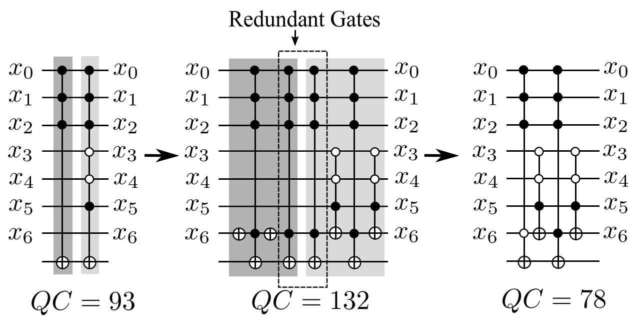


Fig. 10 Example for $\mathrm{QR}(g_i, g_j) = (\alpha, 0, 0, \delta) = (3, 0, 0, 3)$

□

Figures 5, 6, 7, 8, 9 and 10 illustrate the decomposition rule **R6** when applied to the gate pairs belonging to different classifications. The gate pairs are decomposed such that they generate redundant gates which are then removed from the circuit resulting in final gate netlist.

It can be seen from Table 2 that there is a reduction in quantum cost in each case after applying this decomposition rule.

It should be mentioned that the decomposition rule results in reduced quantum cost only if the number of unequal control lines in the gate pair, i.e., $\beta + \gamma$ or $\beta + \delta$, is less than half the total number of lines in the circuit. If this is not the case,

Table 2 Quantum Cost reduction for different classifications after applying decomposition Rule R6

$\mathrm{QR}(g_i, g_j)$	Illustration	Quantum Cost (QC)	
		Initial gate pair	Effect of applying rule R6
$(\alpha, \beta, \gamma, \delta) = (2, 2, 1, 1)$	Figure 5	104	78
$(\alpha, \beta, \gamma, 0) = (2, 2, 1, 0)$	Figure 6	78	62
$(\alpha, \beta, 0, \delta) = (2, 2, 0, 2)$	Figure 7	106	88
$(\alpha, 0, \gamma, \delta) = (3, 0, 2, 1)$	Figure 8	78	64
$(\alpha, 0, \gamma, 0) = (3, 0, 3, 0)$	Figure 9	93	78
$(\alpha, 0, 0, \delta) = (3, 0, 0, 3)$	Figure 10	93	78

then the rule results in a higher-cost gate netlist when compared with the quantum cost of individual gates. This result can be proved with the help of following lemma:

Lemma 3 *In a reversible circuit that has n lines, consider two gates g_i and g_j with same target line x_t and an unused line x_u. The $QR(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ and the sets of control lines corresponding to α, β, γ and δ are defined as K, C, P and Q respectively. The decomposition rule results in a higher-cost gate netlist when compared with the cost of individual gates under the following conditions*:

1. $\beta + \gamma \le \lceil \frac{n+1}{2} \rceil$, $\beta + \delta \le \lceil \frac{n+1}{2} \rceil$ *and* $\alpha \le 1$.
2. $\beta + \gamma > \lceil \frac{n+1}{2} \rceil$ *and/or* $\beta + \delta > \lceil \frac{n+1}{2} \rceil$.

Proof The cascade of gates g_i and g_j can be given as:

$$g_i \circ g_j = \mathrm{MCT}(K \cup C \cup P; x_t) \circ \mathrm{MCT}(K \cup C \cup Q; x_t). \quad (5)$$

Considering $\alpha + \beta + \gamma > \lceil \frac{n+1}{2} \rceil$ and $\alpha + \beta + \delta > \lceil \frac{n+1}{2} \rceil$, the NCT cost (Eq. (1)) to implement the gate pair g_i and g_j is

$$\begin{aligned} &8(\alpha + \beta + \gamma - 3) + 8(\alpha + \beta + \delta - 3) \\ &= 16\alpha + 16\beta + 8\gamma + 8\delta - 48. \end{aligned} \quad (6)$$

The decomposed gate netlist after applying the decomposition rule on this pair can be given as:

$$\begin{aligned} g_i \circ g_j = &\mathrm{MCT}(C \cup P; x_u) \circ \mathrm{MCT}(K \cup \{x_u\}; x_t) \circ \\ &\mathrm{MCT}(C \cup P; x_u) \circ \mathrm{MCT}(C \cup Q; x_u) \circ \\ &\mathrm{MCT}(K \cup \{x_u\}; x_t) \circ \mathrm{MCT}(C \cup Q; x_u) \end{aligned} \quad (7)$$

The pair of gates can be replaced only when the decomposed gate netlist will result in lesser cost when compared with the cost of the gate pair. The NCT cost (Eq. (1)) of the decomposed gate netlist for different cases is as given below:

Case 1: If $\beta + \gamma \le \lceil \frac{n+1}{2} \rceil$, $\beta + \delta \le \lceil \frac{n+1}{2} \rceil$ and $\alpha > \lceil \frac{n+1}{2} \rceil$ then the NCT cost of the decomposed gate netlist is

$$\begin{aligned}&2(4(\beta+\gamma-2)+4(\beta+\delta-2)+8(\alpha+1-3))\\ =&16\alpha+16\beta+8\gamma+8\delta-64\end{aligned} \tag{8}$$

The pair of gates can be replaced with the decomposed gate netlist only when the cost computed by Eq. (8) is less than that computed by Eq. (6)

$$\begin{aligned}&\implies 16\alpha+16\beta+8\gamma+8\delta-64<\\ &16\alpha+16\beta+8\gamma+8\delta-48\\ &-64<-48\end{aligned}$$

Thus, it follows from the above inequalities that the gate pair can be replaced by the decomposed gate netlist.

Case 2: If $\beta+\gamma \le \lceil\frac{n+1}{2}\rceil$, $\beta+\delta \le \lceil\frac{n+1}{2}\rceil$ and $\alpha \le \lceil\frac{n+1}{2}\rceil$ then the NCT cost of the decomposed gate netlist is

$$\begin{aligned}&2(4(\beta+\gamma-2)+4(\beta+\delta-2)+4(\alpha+1-2))\\ =&8\alpha+16\beta+8\gamma+8\delta-40\end{aligned} \tag{9}$$

To replace the pair of gates with the decomposed gate netlist, the cost computed by Eq. (9) should be less than the cost computed by Eq. (6)

$$\begin{aligned}&\implies 8\alpha+16\beta+8\gamma+8\delta-40<\\ &16\alpha+16\beta+8\gamma+8\delta-48\\ &\alpha>1\end{aligned}$$

From the above inequalities, it is clear that the gate pair can be replaced by the decomposed gate netlist only if $\alpha > 1$ proving the first condition of the lemma.

Case 3: If $\beta+\gamma > \lceil\frac{n+1}{2}\rceil$, $\beta+\delta > \lceil\frac{n+1}{2}\rceil$ and $\alpha \le \lceil\frac{n+1}{2}\rceil$ then the NCT cost of the decomposed gate netlist is

$$\begin{aligned}&2(8(\beta+\gamma-3)+8(\beta+\delta-3)+4(\alpha+1-2))\\ =&8\alpha+32\beta+16\gamma+16\delta-104\end{aligned} \tag{10}$$

To replace the pair of gates with the decomposed gate netlist, the cost computed by Eq. (10) should be less than the cost computed by Eq. (6)

$$\begin{aligned}&\implies 8\alpha+32\beta+16\gamma+16\delta-104<\\ &16\alpha+16\beta+8\gamma+8\delta-48\\ &\alpha+7<2\beta+\gamma+\delta\\ &\alpha+7<(\beta+\gamma)+(\beta+\delta)\end{aligned} \tag{11}$$

In this case, the minimum value of $\beta+\gamma$ (and $\beta+\delta$) and the maximum value of α, in terms of n are $\lceil\frac{n+1}{2}\rceil+1$ and $\lceil\frac{n+1}{2}\rceil$ respectively. Substituting these value in Eq. (11) results in $n < 9$. Thus, for the maximum value of n, i.e., 8, excluding target line and the unused line, a maximum of 6 control lines is possible for a given gate. However for $n = 8$, the minimum value of $\beta+\gamma$ is 6 which is the maximum number of control lines allowed in this case resulting in $\alpha = 0$. The condition for applying decomposition rule R6 is $\alpha > 0$ and thus proving the second condition of the lemma. □

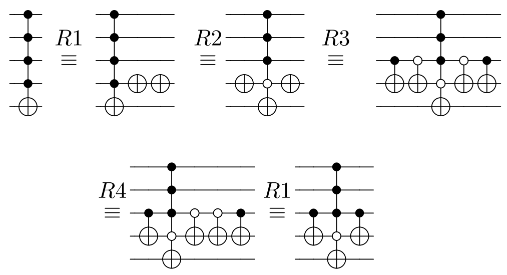


Fig. 11 Derivation of Transformation Rule T1

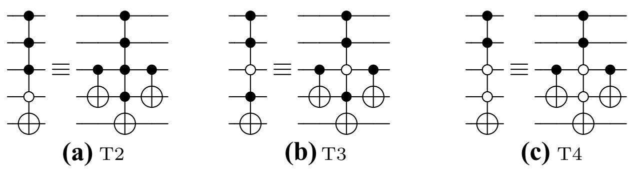


Fig. 12 Illustration of Transformation Rules

The above lemma shows that if $\beta+\gamma > \lceil\frac{n+1}{2}\rceil$ and $\beta+\delta > \lceil\frac{n+1}{2}\rceil$ then the decomposition rule can not obtain the cost reduction. However, if the values of $\beta+\gamma$ and $\beta+\delta$ are reduced such that $\beta+\gamma \le \lceil\frac{n+1}{2}\rceil$ and $\beta+\delta \le \lceil\frac{n+1}{2}\rceil$ then the proposed decomposition rule can be applied to achieve the cost reduction. In order to reduce the values of $\beta+\gamma$ and $\beta+\delta$ we propose a transformation technique explained in the following sub-section.

Complementary Control Line (CCL) Transformation Technique for Reducing β

This transformation technique converts complementary control lines (CCLs) (β) to equal control lines (ECLs) (α) by adding extra gates to the circuit. It is explained in Lemma 4 given below and the transformation rules used in this lemma can be described as follows:

T1 $\mathrm{MCT}(X\cup\{x_i,x_j\};x_t) = \mathrm{MCT}(\{x_i\};x_j)\circ\mathrm{MCT}(X\cup\{x_i,\overline{x_j}\};x_t)\circ\mathrm{MCT}(\{x_i\};x_j)$
T2 $\mathrm{MCT}(X\cup\{x_i,\overline{x_j}\};x_t) = \mathrm{MCT}(\{x_i\};x_j)\circ\mathrm{MCT}(X\cup\{x_i,x_j\};x_t)\circ\mathrm{MCT}(\{x_i\};x_j)$
T3 $\mathrm{MCT}(X\cup\{\overline{x_i},x_j\};x_t) = \mathrm{MCT}(\{x_i\};x_j)\circ\mathrm{MCT}(X\cup\{\overline{x_i},\overline{x_j}\};x_t)\circ\mathrm{MCT}(\{x_i\};x_j)$
T4 $\mathrm{MCT}(X\cup\{\overline{x_i},\overline{x_j}\};x_t) = \mathrm{MCT}(\{x_i\};x_j)\circ\mathrm{MCT}(X\cup\{\overline{x_i},x_j\};x_t)\circ\mathrm{MCT}(\{x_i\};x_j)$.

These transformation rules can be derived using the axioms presented in [28]. For example, the derivation of rule T1 is illustrated in Fig. 11. Similarly other three rules (T2, T3 and T4), illustrated in Fig. 12, can also be derived using these axioms.

Lemma 4 (CCL Transformation Rule R7) *Consider a pair of gates g_i and g_j with same target line x_t. $QR(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ while the sets of control lines are represented as K, C, P and Q. Consider two lines $x_i, x_j \in C$ such that the control connection of lines x_i and x_j in the gates g_i and g_j are positive and negative, respectively. The sets X and Y contain the control lines of gates g_i and g_j excluding lines x_i and x_j respectively. The complementary control line x_j can be transformed to equal control line using two extra CNOT gates and the resulting cascade of gates, g_i and g_j, is given as*:

$$g_i \circ g_j = \text{MCT}(\{x_i\};x_j) \circ \text{MCT}(X \cup \{x_i, \overline{x_j}\};x_t) \circ \text{MCT}(Y \cup \{\overline{x_i}, \overline{x_j}\};x_t) \circ \text{MCT}(\{x_i\};x_j)$$

Proof The gates g_i and g_j can be represented as:

$$g_i = \text{MCT}(X \cup \{x_i, x_j\};x_t)$$
$$g_j = \text{MCT}(Y \cup \{\overline{x_i}, \overline{x_j}\};x_t)$$

The transformation rules, T1, T2, T3 and T4 are applied to the gates depending on the control connections of lines x_i and x_j. Since gate g_i has $\{x_i, x_j\} \in C$ and gate g_j has $\{\overline{x_i}, \overline{x_j}\} \in C$, the transformation rules **T1** and **T4** are applied on the gates g_i and g_j, respectively, resulting in

$$g_i = \text{MCT}(\{x_i\};x_j) \circ \text{MCT}(X \cup \{x_i, \overline{x_j}\};x_t) \circ \text{MCT}(\{x_i\};x_j) \tag{12}$$

$$g_j = MCT(\{x_i\};x_j) \circ \text{MCT}(Y \cup \{\overline{x_i}, \overline{x_j}\};x_t) \circ \text{MCT}(\{x_i\};x_j). \tag{13}$$

The cascade of gates, g_i and g_j, using Eqs. (12) and (13) is shown below:

$$\begin{aligned} g_i \circ g_j = &\text{MCT}(\{x_i\};x_j) \circ \text{MCT}(X \cup \{x_i, \overline{x_j}\};x_t) \\ &\circ \text{MCT}(\{x_i\};x_j) \circ \text{MCT}(\{x_i\};x_j) \\ &\circ \text{MCT}(Y \cup \{\overline{x_i}, \overline{x_j}\};x_t) \circ \text{MCT}(\{x_i\};x_j). \end{aligned}$$

The third and fourth gate in the above gate netlist are the same and thus can be removed. The resulting gate netlist is given as:

$$g_i \circ g_j = \text{MCT}(\{x_i\};x_j) \circ \text{MCT}(X \cup \{x_i, \overline{x_j}\};x_t) \circ \text{MCT}(Y \cup \{\overline{x_i}, \overline{x_j}\};x_t) \circ \text{MCT}(\{x_i\};x_j) \tag{14}$$

From Eq. (14), the control line x_j is transformed to ECL using two CNOT gates. It may be noted however that the line x_i can not be transformed to ECL because it has to be used as a control line for the CNOT gates that are added. □

The rule R7 is applied recursively to transform β CCLs to $\beta - 1$ ECLs. The transformation when $\beta = 3$ is illustrated in the following example:

Example 2 Consider the gate pair g_1 and g_2 shown in Fig. 13a. $\text{QR}(g_1, g_2) = (0, 3, 0, 0)$ and the set which contains control lines corresponding to β is $C = \{x_0, x_1, x_2\}$. The gates g_1 and g_2 are transformed using Lemma 4 resulting in a gate netlist shown in Fig. 13b which consists of g_3, g_4 and two CNOT gates. The line x_1 which was CCL for g_1 and g_2 is transformed to ECL for gate pair, g_3 and g_4. Hence, the QR for the gates g_3 and g_4 is (1, 2, 0, 0) where $C = \{x_0, x_2\}$. Applying Lemma 4 on the gate pair g_3 and g_4 results in a gate netlist shown in Fig. 13c. It can be seen from Fig. 13 that the gate pair g_1 and g_2 with $\text{QR}(g_1, g_2) = (0, 3, 0, 0)$ is transformed to a netlist with four CNOT gates and a pair of gates, g_5 and g_6 where $\text{QR}(g_5, g_6) = (2, 1, 0, 0)$.

Since the proposed CCL transformation technique changes the number of equal and complementary control lines, it modifies QR of a gate pair. For example, consider a gate pair g_i and g_j with $\text{QR}(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$. This technique converts β CCLs to $\beta - 1$ equal control lines resulting in $\alpha + \beta - 1$ ECLs and one CCL, i.e., $\text{QR}(g_i, g_j) = (\alpha + \beta - 1, 1, \gamma, \delta)$. If the modified QR satisfies the condition in Lemma 3, then the decomposition rule can be applied to reduce the cost of the circuit. This process of applying the transformation rule to modify *QR* such that the proposed decomposition rule can be applied, is explained with the help of example given below:

Example 3 Consider two gates g_1 and g_2 as shown in Fig. 14a. The $\text{QR}(g_1, g_2) = (1, 3, 2, 1)$. According to Lemma 3, the decomposition rule does not result in a reduced cost when applied on this gate pair. The CCL transformation technique is applied on this gate pair resulting in a modified gate pair

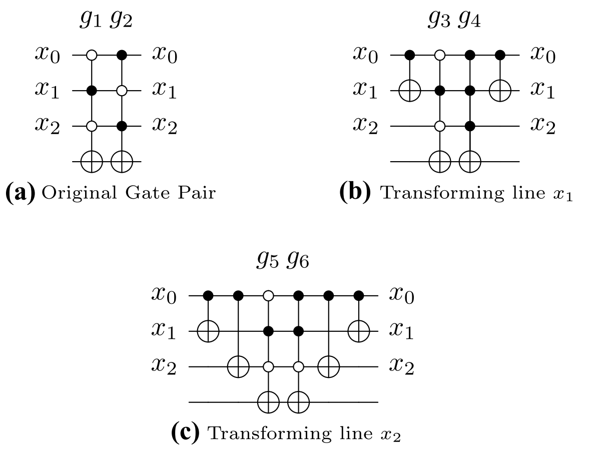

$g_1\ g_2$
x_0
x_1
x_2
(a) Original Gate Pair
$g_3\ g_4$
(b) Transforming line x_1
$g_5\ g_6$
(c) Transforming line x_2

Fig. 13 Illustration of Example 2

g_a and g_b with $\mathrm{QR}(g_a, g_b) = (3, 1, 2, 1)$ as shown in Fig. 14b. From this modified QR, it is evident that $\beta + \gamma \leq \lceil \frac{n+1}{2} \rceil$, $\beta + \delta \leq \lceil \frac{n+1}{2} \rceil$, $\alpha \leq \lceil \frac{n+1}{2} \rceil$ and according to Lemma 3, this results in a reduced gate netlist after applying the decomposition rule. The final decomposed gate netlist is shown in Fig. 14c where it can be seen that there is a reduction of the cost from 132 to 92.

So far, *QR* classifications with $\alpha > 0$ have been covered with the proposed decomposition and transformation techniques. In the next section, the optimization technique for *QR* classifications with $\alpha = 0$ is discussed.

Optimization Technique for $\mathrm{QR}(g_i, g_j) = (0, \beta, \gamma, \delta)$

The technique presented in the previous section cannot be applied directly on QR classifications of the type $(0, \beta, \gamma, \delta)$, $(0, \beta, 0, \delta)$ and $(0, \beta, \gamma, 0)$. However all these classifications have β CCLs which can be converted to equal control lines using the proposed CCL transformation technique. After this transformation, QR types are modified as given in Table 3 below:

It can be seen from the table that the *QR* classifications after CCL transformation have $\alpha > 0$ and hence are a subset of *QR* classifications discussed in Section "Optimization Technique for $\mathrm{QR}(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ when $\alpha > 0$". The following example illustrates the process of applying the proposed transformation rule to modify the *QR* such that the decomposition rule can be applied.

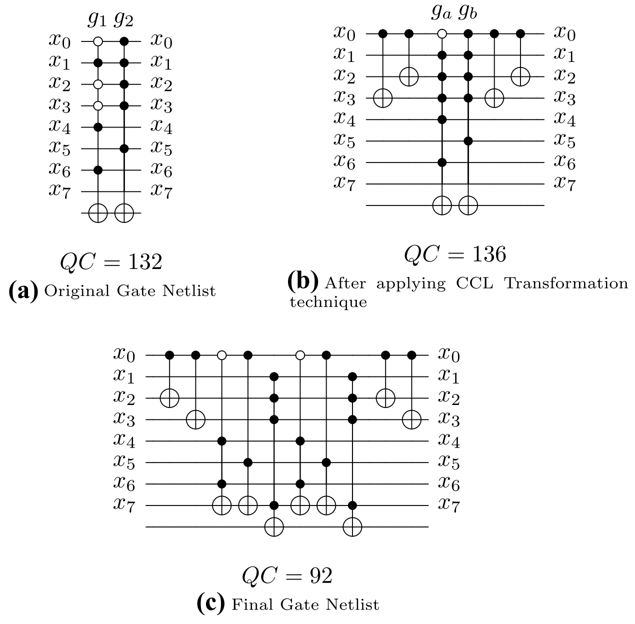

g_1 g_2
x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7
$QC = 132$
(a) Original Gate Netlist
g_a g_b
$QC = 136$
(b) After applying CCL Transformation technique
$QC = 92$
(c) Final Gate Netlist

Fig. 14 Illustration of Example 3

Example 4 Consider two gates g_1 and g_2 shown in Fig. 15a where $\mathrm{QR}(g_1, g_2) = (0, 4, 1, 1)$. The CCL transformation technique is applied on this gate pair resulting in a modified gate pair g_a and g_b with $\mathrm{QR}(g_a, g_b) = (3, 1, 1, 1)$ as shown in Fig. 15b. The decomposition rule can now be applied to obtain the final reduced gate netlist as shown in Fig. 15c. It is evident that there is a reduction in the cost from 104 to 78. Similar pattern can be observed with other classifications also.

Optimization Techniques for Specific QR Classifications

In this section, different optimization techniques are presented for specific QR classifications.

For $\mathrm{QR}(g_i, g_j) = (\alpha, \beta, 0, 0)$ and $\mathrm{QR}(g_i, g_j) = (0, \beta, 0, 0)$

The gate pairs that fall under QR classifications of type $(\alpha, \beta, 0, 0)$ and $(0, \beta, 0, 0)$ can be reduced by applying rule **R2**. However, the CCL transformation technique can further reduce the cost of the gate pair. This is achieved by converting $\beta - 1$ CCLs to ECLs which modifies the QR from $(\alpha, \beta, 0, 0)$ and $(0, \beta, 0, 0)$ to $(\alpha + \beta - 1, 1, 0, 0)$ and $(\beta - 1, 1, 0, 0)$, respectively. From the modified QRs, it is

Table 3 QR transformation

Before CCL transformation	After CCL transformation
$(0, \beta, \gamma, \delta)$	$(\beta - 1, 1, \gamma, \delta)$
$(0, \beta, \gamma, 0)$	$(\beta - 1, 1, \gamma, 0)$
$(0, \beta, 0, \delta)$	$(\beta - 1, 1, 0, \delta)$

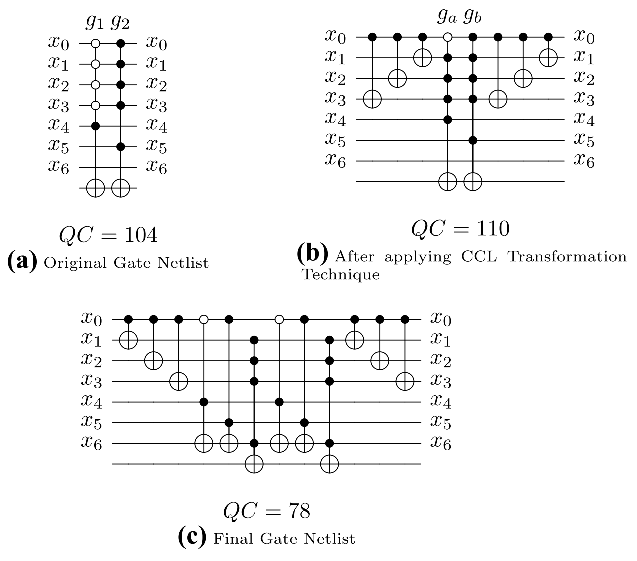

g_1 g_2
x_0 x_1 x_2 x_3 x_4 x_5 x_6
$QC = 104$
(a) Original Gate Netlist
g_a g_b
$QC = 110$
(b) After applying CCL Transformation Technique
$QC = 78$
(c) Final Gate Netlist

Fig. 15 Illustration of Example 4

evident that the rule R3 can be applied to reduce the resultant gate netlist. This process is explained with the following example:

Example 5 Consider a pair of gates with $\mathrm{QR}(g_1, g_2) = (0, 4, 0, 0)$ shown in Fig. 16a. The reduction of this pair using rule **R2** results in the gate netlist as shown in Fig. 16b. Applying CCL transformation on the gate pair modifies the QR to (3, 1, 0, 0) as shown in Fig. 16c. The modified gate pair can be merged into a single gate using the rule **R3** as given in Fig. 16d. From the figure it can be seen that while the cost of the circuit reduces to 52 from 58 as a result of applying rule **R2,** the same reduces to 19 after applying CCL transformation with rule **R3**.

For $QR(g_i, g_j) = (\alpha, \beta, 1, 0)$ and $QR(g_i, g_j) = (\alpha, \beta, 0, 1)$

Rule **R4** can be applied on the classifications of type $(\alpha, \beta, 1, 0)$ and $(\alpha, \beta, 0, 1)$ only if $\beta = 1$. However, if $\beta > 1$ then the CCL transformation can be used to modify the QR classifications from $(\alpha, \beta, 1, 0)$ and $(\alpha, \beta, 0, 1)$ to $(\alpha + \beta - 1, 1, 1, 0)$ and $(\alpha + \beta - 1, 1, 0, 1)$, respectively. This enables the use of rule **R4** on the modified QRs which is explained in the following example:

Example 6 Consider a pair of gates with $QR(g_1, g_2) = (1, 3, 1, 0)$ as shown in Fig. 17a. Applying CCL transformation on the gate pair modifies QR to (3, 1, 1, 0) as shown in Fig. 17b. Rule **R4** can now be applied on the modified gate pair and the resulting gate netlist is shown in Fig. 17c. It can be seen that there is a reduction in the cost from 78 to 69.

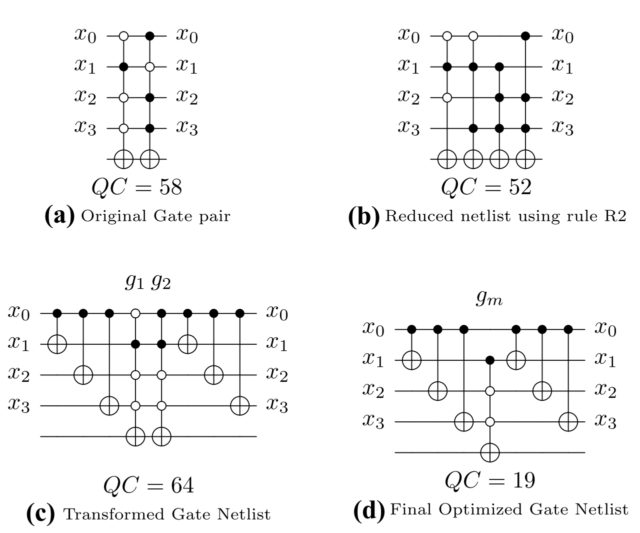


Fig. 16 Illustration of Example 5

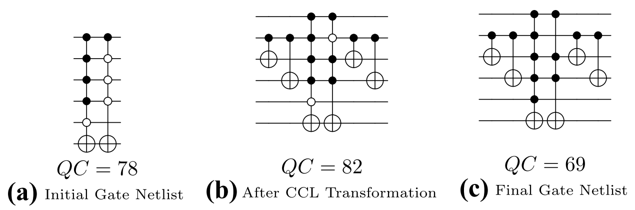


Fig. 17 Illustration of Example 6

For $\mathrm{QR}(g_i, g_j) = (\alpha, \beta, 1, 1)$

An optimization technique is presented in this subsection to reduce the cost of a gate pairs,g_i and g_j, with $\mathrm{QR}(g_i, g_j) = (\alpha, 1, 1, 1)$. This technique decomposes the gate pair into a network of smaller gates which is explained with the help of following lemma:

Lemma 5 (Swap Rule R8) *Consider two gates g_i and g_j with same target line x_t. $QR(g_i, g_j) = (\alpha, \beta, \gamma, \delta)$ and the sets of control lines corresponding to α, β, γ and δ are defined as K, C, P and Q respectively. If β, γ and δ are equal to* 1, *$C = \{x_C\}, P = \{x_P\}$ and $Q = x_Q$, then the gate pair g_i and g_j can be decomposed using the following rule*:

$$\begin{aligned} g_i \circ g_j = {} & \mathrm{MCT}\big(\{x_Q\}; x_P\}\big) \circ \mathrm{MCT}\big(\{x_C\} \cup \{x_P\}; x_Q\big) \\ & \circ \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\}; x_t\big) \\ & \circ \mathrm{MCT}\big(\{x_C\} \cup \{x_P\}; x_Q\big) \circ \mathrm{MCT}(\{x_Q\}; x_P\}). \end{aligned}$$

Proof The gates g_i and g_j are represented as $g_i = \mathrm{MCT}(K \cup C \cup P; x_t)$ and $g_j = \mathrm{MCT}(K \cup C \cup Q; x_t)$, respectively. If β, γ and δ are equal to 1, then there exists only one control line in the sets C, P and Q, respectively, i.e., $C = \{x_C\}, P = \{x_P\}$ and $Q = x_Q$. Assuming the control line x_C has a positive control connection in gate g_i and a negative control connection in gate g_j, they can be written as:

$$\begin{aligned} g_i = {} & \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\}; x_t\big) \\ g_j = {} & \mathrm{MCT}\big(K \cup \{\overline{x_C}\} \cup \{x_Q\}; x_t\big). \end{aligned}$$

Gate g_j can be decomposed into two gates using the rules presented in [11] which is given as follows:

$$g_j = \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_Q\}; x_t\big) \circ \mathrm{MCT}\big(K \cup \{x_Q\}; x_t\big).$$

The cascade of gate pair g_i and g_j is given as,

$$\begin{aligned} g_i \circ g_j = {} & \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\}; x_t\big) \\ & \circ \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_Q\}; x_t\big) \\ & \circ \mathrm{MCT}\big(K \cup \{x_Q\}; x_t\big). \end{aligned}$$

Using rule R5, the first two gates in the above netlist can be decomposed into a network of three gates as,

$$g_i \circ g_j = \mathrm{MCT}\big(\{x_Q\};x_P\}\big) \circ \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\};x_t\big) \\ \circ \mathrm{MCT}\big(\{x_Q\};x_P\}\big) \circ \mathrm{MCT}\big(K \cup \{x_Q\};x_t\big).$$

The position of last two gates can be interchanged using the moving rule presented in [27] and the resulting gate netlist is given as,

$$g_i \circ g_j = \mathrm{MCT}\big(\{x_Q\};x_P\}\big) \circ \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\};x_t\big) \\ \circ \mathrm{MCT}\big(K \cup \{x_Q\};x_t\big) \circ \mathrm{MCT}\big(\{x_Q\};x_P\}\big).$$

The second and third gates in the above netlist can be decomposed into a network of three gates using rule R5 and the resulting netlist is given as follows thus proving the lemma.

$$g_i \circ g_j = \mathrm{MCT}\big(\{x_Q\};x_P\}\big) \circ \mathrm{MCT}\big(\{x_C\} \cup \{x_P\};x_Q\big) \\ \circ \mathrm{MCT}\big(K \cup \{x_C\} \cup \{x_P\};x_t\big) \\ \circ \mathrm{MCT}\big(\{x_C\} \cup \{x_P\};x_Q\big) \circ \mathrm{MCT}\big(\{x_Q\};x_P\}\big).$$

□

The following example illustrates the above lemma:

Example 7 Consider a pair of gates with $\mathrm{QR}(g_1, g_2) = (1, 1, 1, 1)$ as shown in Fig. 18a. The set of control lines corresponding to this QR are $K = \{x_0\}$, $C = \{x_1\}$, $P = \{x_2\}$ and $Q = \{x_3\}$. According to Lemma 5, the gate pair is decomposed into a gate netlist as shown in Fig. 18b where it can be seen that there is a reduction in the QC from 26 to 17.

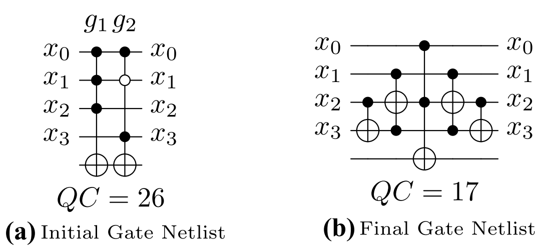
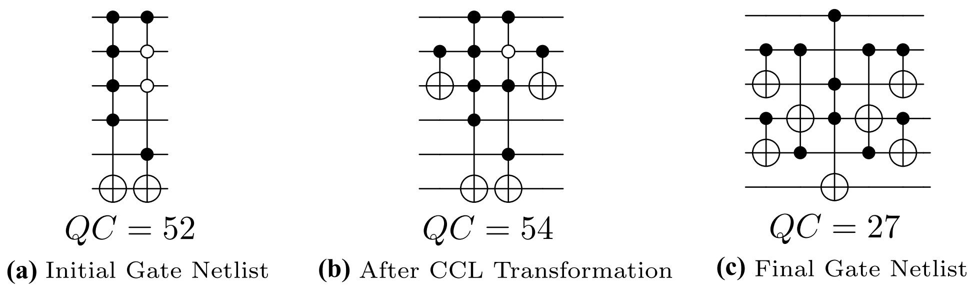


Fig. 18 Illustration of Example 7

Fig. 19 Illustration of Example 8



Although Rule **R8** adds two Toffoli and two CNOT gates, the gate pair is merged into a single gate with reduced control lines thereby reducing the cost of the gate pair. This technique can be applied only if $\beta = 1$ but not when $\beta > 1$. Thus, if the gate pair has $\mathrm{QR}(g_i, g_j) = (\alpha, \beta, 1, 1)$ then the CCL transformation technique can be applied to modify the QR classifications to $\mathrm{QR}(g_i, g_j) = (\alpha + \beta - 1, 1, 1, 1)$. This enables the use of Rule **R8** to reduce the cost of the gate pair. This process is illustrated in the following example:

Example 8 Consider a pair of gates with $\mathrm{QR}(g_1, g_2) = (1, 2, 1, 1)$ as shown in Fig. 19a. Since Lemma 5 can be applied only on gate pair with $\beta = 1$, CCL transformation is applied on this gate pair and the modified gate netlist is shown in Fig. 19b such that the QR is modified to (2, 1, 1, 1). From the modified QR, Rule **R8** can be applied on this gate netlist and the resulting circuit is shown in Fig. 19c. From the figure, it is evident that the cost of the gate pair is reduced from 52 to 27.

There can be no optimization technique that covers QR classifications of type $(0, 0, \gamma, 0)$, $(0, 0, 0, \delta)$ and $(0, 0, \gamma, \delta)$ because the gate pairs do not have any equal or complementary control lines. Table 4 summarizes the QR classifications and the appropriate combination of optimization techniques that can be applied on a given gate pair to improve the cost.

Post-Synthesis Optimization Algorithm

A ‘greedy’ algorithm for post-synthesis optimization is presented in Algorithm 1. This algorithm takes a reversible gate netlist G as an input and gets an optimized gate netlist G' as output. Initially, the gate netlist G is traversed and the gates with equal control lines but different target lines are merged using the function *target_merging* [29]. This avoids regeneration of the same gate for different target lines.

Table 4 Summary of optimization techniques for different QR classifications

QR(g_i, g_j)	Conditions	Optimization techniques
$(0, 0, 0, 0)$	–	Deletion Rule R1
$(\alpha, 0, 0, 0)$	–	Deletion Rule R1
$(\alpha, \beta, 0, 0)$, $(0, \beta, 0, 0)$	$\beta = 1$	Rule R3
	$\beta > 1$	Proposed CCL transformation Rule R7+ Rule R3
$(0, 0, \gamma, 0)$, $(0, 0, 0, \delta)$, $(0, 0, \gamma, \delta)$	–	No reduction technique
$(\alpha, \beta, \gamma, \delta)$, $(\alpha, \beta, \gamma, 0)$, $(\alpha, \beta, 0, \delta)$, $(\alpha, 0, \gamma, \delta)$, $(\alpha, 0, \gamma, 0)$, $(\alpha, 0, 0, \delta)$	$(\beta = 0, \gamma = 0$ and $\delta = 1)$ or $(\beta = 0, \gamma = 1$ and $\delta = 0)$	Rule R3
	$(\beta = 1, \gamma = 1$ and $\delta = 0)$ or $(\beta = 1, \gamma = 0$ and $\delta = 1)$	Rule R4
	$(\beta > 1, \gamma = 0$ and $\delta = 1)$ or $(\beta > 1, \gamma = 1$ and $\delta = 0)$	Proposed CCL transformation Rule R7+ Rule R4
	$(\beta = 0, \gamma = 1$ and $\delta > 0)$ or $(\beta = 0, \gamma > 0$ and $\delta = 1)$	Cube Pairing Rule R5
	$(\gamma = 1$ and $\delta = 1)$	Proposed Swap Rule R8
	$\beta + \gamma \le \lceil \frac{n+1}{2} \rceil$, $\beta + \delta \le \lceil \frac{n+1}{2} \rceil$ and $\alpha > 1$	Proposed Decomposition Rule R6
	$\beta + \gamma > \lceil \frac{n+1}{2} \rceil$, $\beta + \delta > \lceil \frac{n+1}{2} \rceil$, $\alpha \le \lceil \frac{n+1}{2} \rceil$,$1 + \gamma \le \lceil \frac{n+1}{2} \rceil$ and $1 + \delta \le \lceil \frac{n+1}{2} \rceil$	Proposed CCL transformation Rule R7 + Proposed Decomposition Rule R6
	$\beta + \gamma > \lceil \frac{n+1}{2} \rceil$, $\beta + \delta > \lceil \frac{n+1}{2} \rceil$, $\alpha \le \lceil \frac{n+1}{2} \rceil$,$1 + \gamma > \lceil \frac{n+1}{2} \rceil$ and $1 + \delta > \lceil \frac{n+1}{2} \rceil$	No reduction technique
$(0, \beta, \gamma, \delta)$, $(0, \beta, \gamma, 0)$, $(0, \beta, 0, \delta)$	$\beta + \gamma \le \lceil \frac{n+1}{2} \rceil$, $\beta + \delta \le \lceil \frac{n+1}{2} \rceil$ and $\beta > 2$	Proposed CCL transformation Rule R7 + Proposed Decomposition Rule R6
	$\beta + \gamma > \lceil \frac{n+1}{2} \rceil$, $\beta + \delta > \lceil \frac{n+1}{2} \rceil$, $\beta > 2$,$1 + \gamma \le \lceil \frac{n+1}{2} \rceil$ and $1 + \delta \le \lceil \frac{n+1}{2} \rceil$	
	$\beta + \gamma > \lceil \frac{n+1}{2} \rceil$, $\beta + \delta > \lceil \frac{n+1}{2} \rceil$, $1 + \gamma > \lceil \frac{n+1}{2} \rceil$ and $1 + \delta > \lceil \frac{n+1}{2} \rceil$	No reduction technique

Algorithm 1 Post-Synthesis Greedy Optimization Method

```
1: Input: Reversible Gate Netlist $G$
2: Output: Modified Gate Netlist $G'$
3: begin
4: $GN = target\_merging(G)$
5: $G' = \phi$
6: while $GN$ is not empty do
7:     $flag = false$
8:     $G_i = select\_gate(GN)$
9:     $RemoveGate(GN, G_i)$
10:    $CostTable(G_i) = \phi$
11:    for each $G_j \in GN$ do
12:        $Status = False$
13:        $G_t = \phi$
14:        $NewCost = \infty$
15:        $Status$, $G_t$, $NewCost = translate(G_i, G_j)$
16:        if $Status$ is $True$ then
17:            $CostTable(G_i) = \{G_j, G_t, NewCost\}$
18:            $flag = True$
19:        end if
20:    end for
21:    if $flag$ is $True$ then
22:        $G_p, G_{new} = LeastCost(CostTable(G_i))$
23:        if $SizeOf(G_{new}) == 1$ then
24:            $Append(GN, G_{new})$
25:        else
26:            $Append(G', G_{new})$
27:        end if
28:        $RemoveGate(GN, G_p)$
29:    else
30:        $Append(G', G_i)$
31:    end if
32: end while
33: return $G'$
34: end
```

Each gate G_i is paired with every other gate G_j in *GN* and is given to the *translate* function. This function takes a gate pair, classifies it using the proposed representation and checks for the possibility of reduction using the optimization techniques presented in Table 4. If a possibility exists to optimize that pair, the function returns *Status* as *True*, the optimized gate netlist as G_t and its cost as *NewCost*. Also, the gate netlist G_t and its cost *NewCost* are added to the *CostTable*(G_i) and the *flag* is set to *True*.

After G_i is paired with every other gate G_j in *GN*, the status of *flag* is checked. If the *flag* is *False*, it indicates that there is no optimization possible for the gate G_i when paired with any other gate in the circuit and the gate G_i is added to the output gate netlist G'. If it is *True* then the function *LeastCost* scans the *CostTable*(G_i) and returns a gate G_p that results in maximum possible reduction when paired with G_i. This function also returns G_{new} which is the reduced gate netlist for the gate pair G_i and G_p. If the gate netlist G_{new} consists of a single gate then the gate is added to the initial netlist *GN* else it is added to the output gate netlist G'. Gate G_p is subsequently removed from the netlist *GN*. This process is repeated for each of the gates in the netlist. Thus, for 'n' number of gates in a netlist, the algorithm complexity is $O(n^2)$.

Table 5 Comparison with existing post-synthesis optimization algorithms

Benchmark	Lines (No. of Qubits)	Original Netlist [26]	[13]	[12]	Proposed Optimization	% Impr with		
						[26]	[13]	[12]
cordic_218	25	349522	348566	348532	74985	78.55	78.49	78.49
apex4_202	28	238146	237748	158095	38826	83.70	83.67	75.44
sym9_193	10	14193	12747	13090	3485	75.45	72.66	73.38
table3_264	28	80039	79326	61412	17963	77.56	77.36	70.75
in2_236	29	23814	23146	20600	7434	68.78	67.88	63.91
misex3_242	28	119177	115637	99119	38125	68.01	67.03	61.54
clip_206	14	6731	6535	6119	2354	65.03	63.98	61.53
misex3c_243	28	115190	111258	96064	39907	65.36	64.13	58.46
life_238	10	6767	5740	5744	2420	64.24	57.84	57.87
dist_223	13	7604	7288	6631	2875	62.19	60.55	56.64
sqn_258	10	2128	2041	1887	849	60.10	58.40	55.01
in0_235	26	20042	18999	16985	7761	61.28	59.15	54.31
inc_237	16	2145	2104	1745	929	56.69	55.85	46.76
apla_203	22	3444	3438	3029	1669	51.54	51.45	44.90
f51m_233	22	37417	33333	32882	18356	50.94	44.93	44.18
tial_265	22	56224	47145	47556	26644	52.61	43.48	43.97
max46_240	10	5444	4498	4538	2560	52.98	43.09	43.59
dc2_222	15	1898	1789	1688	1024	46.05	42.76	39.34
sqr6_259	18	1053	1034	876	567	46.15	45.16	35.27
decod_217	21	1746	1745	613	427	75.54	75.53	30.34
5xp1_194	17	1418	1327	1155	819	42.24	38.28	29.09
pm1_249	14	384	354	275	197	48.70	44.35	28.36
cu_219	25	1148	1054	954	702	38.85	33.40	26.42
mux_246	22	1078	804	804	598	44.53	25.62	25.62
dc1_220	11	425	419	249	187	56.00	55.37	24.90
cm150a_210	22	1096	822	822	618	43.61	24.82	24.82
frg1_234	31	15266	14737	14702	11560	24.28	21.56	21.37

Simulation Results

The post-synthesis optimization algorithm is implemented in Python on Intel Xeon workstation (with 8 GB of RAM and 1.7 GHz operating frequency) running Windows 10. This algorithm is applied on different benchmark reversible circuits obtained from RevLib library [26] to evaluate its efficiency in terms of cost. The quantum cost metric used in this work for the purpose of comparison is based on the NCV gate library but not Clifford+T gate library. This is because a major body of the literature contains papers based on NCV library.

Table 5 presents a comparison of the proposed optimization algorithm with existing post-synthesis optimization techniques presented in [12, 13]. The first column gives the name of the benchmark while second, third, fourth and fifth columns indicate the number of qubits, quantum cost of original gate netlist from RevLib and two existing techniques, respectively. Sixth column gives the quantum cost of final gate netlist obtained after applying the proposed optimization algorithm. Seventh, eighth and ninth columns give the percentage improvement over original gate netlist and the existing techniques. It can be seen from Table 5 that in the best case, there is a considerable reduction of quantum cost of up to 78.5%. An average cost improvement of about 47% is observed over all the benchmarks mentioned in the table. This is because, the proposed optimization techniques optimize more number of QR classifications of gate pairs (as seen from Table 4) when compared to the techniques presented in [12, 13].

The reversible gate netlist obtained from the Exclusive-OR Sum of Product (ESOP)-based synthesis method presented in [21] is given as input for the optimization algorithm. A comparison of costs obtained from the optimization algorithm with different ESOP based methods [16–19, 30] is presented in Table 6. First column gives the name of the benchmark circuit while the columns 2-6 indicate the quantum cost of respective benchmark circuits realized with existing ESOP-based methods [16–19, 30]. Column 7 provides the quantum cost to realize that benchmark using the proposed optimization method. Column 8 shows the percentage improvement of quantum cost achieved compared

Table 6 Comparison with existing ESOP based methods

Bench-mark	[16]	[19]	[30]	[17]	[18]	Proposed optimization	% Impr
5xp1	1349	786	865	–	807	741	**5.73**
9symml	5781	10943	16487	1895	3406	2709	– 42.96
add6	6362	–	5084	2683	–	3132	– 16.73
alu2	5215	–	4476	–	3679	2685	**27.02**
alu3	2653	–	–	–	1919	1810	**5.68**
alu4	48778	41127	43850	–	38635	21104	**45.38**
apex4	256857	35840	50680	51284	–	39832	–11.14
apex5	–	33830	–	–	33803	26358	**22.02**
apla	4051	1683	–	–	1709	1607	**4.52**
bw	–	637	–	2233	790	807	-26.69
C17	97	–	–	–	–	74	**23.71**
clip	6616	3824	4484	–	3218	2184	**32.13**
cm150a	844	–	–	–	–	438	**48.1**
con1	207	162	–	–	–	136	**16.05**
cordic	349522	187620	–	–	111955	64624	**42.28**
cu	1332	781	–	–	780	603	**22.69**
dc2	1956	1084	–	–	1099	980	**9.59**
decod	1924	399	–	976	–	461	– 15.54
dist	7414	3700	–	–	–	2745	**25.81**
e64	–	–	–	–	24345	23751	**2.44**
ex1010	183726	52788	–	77293	–	49490	**6.25**
ex2	153	–	–	118	–	117	**0.85**
ex3	97	–	–	73	–	53	**27.4**
f2	274	112	–	116	–	87	**22.32**
f51m	34244	28382	–	–	25119	16850	**32.92**
frg2	–	112008	–	–	114239	88554	**20.94**
in0	22196	7949	–	–	–	7474	**5.98**
majority	147	–	–	106	–	134	– 26.42
max46	4432	–	–	3239	–	2254	**30.41**
misex1	1017	332	466	-	352	360	– 8.43
misex3	122557	49076	67206	–	54132	36624	**25.37**
misex3c	118578	49720	85330	52600	–	38037	**23.5**
mlp4	3827	2496	–	–	–	2079	**16.71**
mux	826	–	–	784	–	416	**46.94**
pm1	582	–	–	290	–	188	**35.17**
radd	798	–	–	349	–	470	– 34.67
rd84	2598	–	2062	–	1965	1371	**30.23**
root	3486	1811	–	–	1583	1439	**9.1**
sao2	7893	3767	5147	–	–	2561	**32.01**
spla	–	–	49419	–	45478	26251	**42.28**
sqn	2170	–	–	1183	–	831	**29.75**
sqr6	1090	583	–	–	–	557	**4.46**
sqrt8	584	–	461	–	–	298	**35.36**
squar5	476	–	251	–	–	227	**9.56**
t481	237	–	237	–	–	205	**13.5**
table3	86173	–	35807	–	32286	16997	**47.35**
urf3	–	53157	–	56766	–	48836	**8.13**
z4	674	489	–	260	–	310	–19.23

Bold values mentioned in Table 6 signify improvement in the cost of the respective benchmarks when the proposed approach isused to the existing methods that give the best reduction for that benchmark.

It can be seen from the table that there is an improvement of up to 48% in quantum cost. For arithmetic benchmark circuits like *frg2*, *in0*,*max46*, etc., and large benchmark circuits like *misex3*, *table3*, etc., there is a reduction in the quantum cost. However, for some benchmarks like *add6, bw, z4*, etc., the optimization method results in higher quantum cost when compared to the existing ones [17–19]. This is because only a small number of required gate pairs is available in these benchmarks that can be transformed using the algorithm and the techniques described earlier.

Briefly, the more recent LUT/ESOP-based techniques [20, 22, 23] focus on EPFL arithmetic benchmark circuits while our work focused on RevLib Library benchmark circuits. However, we would be keen to understand how our proposed algorithm would contribute to optimising the EPFL arithmetic benchmark circuits which is our on-going work currently.

Conclusions

In this paper, a representation has been proposed to classify a pair of gates in a reversible circuit based on its structural similarities. This classification aids in identifying the gate pairs that can be optimized as opposed to those that cannot be. In addition, a set of optimization techniques is proposed that can be applied on classifications that do not have any optimization defined on them. An optimization algorithm has also been presented that uses classifications and optimization techniques to improve the cost of the given gate netlist. Simulation results show that there is a significant reduction in the quantum cost of benchmark circuits with a maximum of 78.5% and an average of 47% when compared to the existing techniques. Further, a comparison of the proposed techniques with the existing ESOP-based ones shows an improvement in quantum cost of up to 48%, except in a few cases.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Nielsen MA, Chuang IL. Quantum computation and quantum information. Cambridge: Cambridge University Press; 2010.
2. Cuykendall R, Andersen DR. Reversible optical computing circuits. Opt Lett. 1987;12(7):542–4.
3. Ciapurin I, Glebov L, Smirnov V. A scheme for efficient quantum computation with linear optics. Phys Rev Lett. 2001;86(22):51885191.
4. Gao W-B, et al. Experimental realization of a controlled-NOT gate with four-photon six-qubit cluster states. Phys Rev Lett. 2010;104(2):020501.
5. Landauer R. Irreversibility and heat generation in the computing process. IBM J Res Dev. 1961;5(3):183–91. https://doi.org/10.1147/rd.53.0183.
6. Bennett CH. Logical reversibility of computation. IBM J Res Dev. 1973;17(6):525–32.
7. Wille R, Drechsler R, Osewold C, Garcia-Ortiz A. Automatic design of low-power encoders using reversible circuit synthesis. In: DATE ’12, 1036–1041 (EDA Consortium, San Jose, CA, USA, 2012). https://doi.org/10.5555/2492708.2492966.
8. Saeedi M, Markov IL. Synthesis and optimization of reversible circuits—a survey. ACM Comput Surveys (CSUR). 2013;45(2):21.
9. Wille R, Drechsler R. Towards a design flow for reversible logic. Berlin: Springer Science & Business Media; 2010.
10. Maslov D, Dueck GW, Miller DM. Techniques for the synthesis of reversible Toffoli networks. ACM Trans Des Autom Electron Syst. 2007. https://doi.org/10.1145/1278349.1278355.
11. Arabzadeh M, Saeedi M, Zamani MS. Rule-based optimization of reversible circuits. In: 15th Asia and South Pacific Design Automation Conference (ASP-DAC), IEEE Press;2010. p. 849–54. https://doi.org/10.1109/ASPDAC.2010.5419684.
12. Datta K, Sengupta I, Rahaman H. A post-synthesis optimization technique for reversible circuits exploiting negative control lines. IEEE Trans Comput. 2015;64(4):1208–14.
13. Datta, K. et al. Exploiting negative control lines in the optimization of reversible circuits. In: RC’13, 209–220 (Springer-Verlag, Berlin, Heidelberg, 2013). https://doi.org/10.1007/978-3-642-38986-3.
14. Miller DM, Wille R, Drechsler R. Reducing reversible circuit cost by adding lines. In: 40th IEEE International Symposium on Multiple-Valued Logic (ISMVL), IEEE Press; 2010. p. 217–22. https://doi.org/10.1109/ISMVL.2010.48.
15. Wille R, Soeken M, Drechsler R. Reducing the number of lines in reversible circuits. In: Design Automation Conference, IEEE Press; 2010. p. 647–52. https://doi.org/10.1145/1837274.1837439.
16. Rice J, Fazel K, Thornton M, Kent K. Toffoli gate cascade generation using ESOP minimization and QMDD-based swapping. In: Proceedings of the Reed-Muller Workshop (RM2009), 2009; p. 63–72.
17. Bandyopadhyay C, Rahaman H, Drechsler R. Improved cube list based cube pairing approach for synthesis of ESOP based reversible logic. In: Gavrilova, Marina L. et al., editors. Lecture Notes in Computer Science, Vol. 8911. Springer Berlin Heidelberg; 2014, p. 129–46.
18. Datta K, Gokhale A, Sengupta I, Rahaman H. An ESOP-based reversible circuit synthesis flow using simulated annealing. In: R. Chaki et al., editors. Advances in Intelligent Systems and Computing, Vol. 305. Springer India; 2015, p. 131–44.
19. Nayeem NM, Rice JE. A shared-cube approach to ESOP-based synthesis of reversible logic. Facta Univ-Ser Electron Energet. 2011;24(3):385–402.
20. Soeken M, Roetteler M, Wiebe N, De Micheli G. Hierarchical reversible logic synthesis using LUTs. In: 54th ACM/EDAC/IEEE Design Automation Conference (DAC),IEEE Press; 2017. p. 1–6. https://doi.org/10.1145/3061639.3062261.
21. Fazel K, Thornton M, Rice J. ESOP-based Toffoli gate cascade generation. In: IEEE Pacific Rim Conference on Communications, Computers and Signal Processing. IEEE Press; 2007. p. 206–9. https://doi.org/10.1109/PACRIM.2007.4313212

22. Meuli G, Soeken M, Roetteler M, Wiebe N, De Micheli G. A best-fit mapping algorithm to facilitate ESOP-decomposition in Clifford+T quantum network synthesis. In: 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), IEEE Press; 2018. p. 664–9. https://doi.org/10.1109/ASPDAC.2018.8297398.
23. Meuli G, Schmitt B, Ehlers R, Riener H, De Micheli G. Evaluating ESOP optimization methods in quantum compilation flows. Cham: Springer International Publishing; 2019. p. 191–206.
24. Toffoli T. Reversible computing. Berlin: Springer; 1980.
25. Barenco A, et al. Elementary gates for quantum computation. Phys Rev A. 1995;52(5):3457.
26. Wille R, Grosse D, Teuber L, Dueck G, Drechsler R. RevLib: an online resource for reversible functions and reversible circuits. In: 38th International Symposium on Multiple Valued Logic (ISMVL), IEEE Press; 2008. p. 220–5. https://doi.org/10.1109/ISMVL.2008.43.
27. Maslov D, Dueck GW, Miller DM. Toffoli network synthesis with templates. IEEE Trans Comput Aided Des Integr Circ Syst. 2005;24(6):807–17. https://doi.org/10.1109/TCAD.2005.847911.
28. Soeken M, Thomsen MK, Dueck GW, Miller DM. White dots do matter: rewriting reversible logic circuits. In: Dueck GW, Miller DM, editors. Reversible computation. Berlin: Springer Berlin Heidelberg; 2013. p. 196–208.
29. Wille R, Soeken M, Otterstedt C, Drechsler R. Improving the mapping of reversible circuits to quantum circuits using multiple target lines. In: 18th Asia and South Pacific Design Automation Conference (ASP-DAC), IEEE Press; 2013. p. 145–50. https://doi.org/10.1109/ASPDAC.2013.6509587.
30. Drechsler R, Finder A, Wille R. Improving ESOP-based synthesis of reversible logic using evolutionary algorithms. In: C. Di. Chio et al., editors. Lecture Notes in Computer Science,Vol. 6625. Springer Berlin Heidelberg; 2011, p. 151–61.