1-2017

Minimization of Quantum Circuits using Quantum Operator Forms

Martin Lukac
Tohoku University

Michitaka Kameyama
Tohoku University

Marek Perkowski
Portland State University, marek.perkowski@pdx.edu

Pawel Kerntopf
Warsaw University of Technology

Citation Details

Lukac, M., Kameyama, M., Perkowski, M., & Kerntopf, P. (2017). Minimization of quantum circuits using quantum operator forms. arXiv preprint arXiv:1701.01999.

Minimization of Quantum Circuits using Quantum Operator Forms

Martin Lukac*, Michitaka Kameyama*,Marek Perkowski†,Pawel Kerntopf‡
*Graduate School of Information Sciences, Tohoku University, Sendai, Japan
Email: {lukacm,kameyama}@ecei.tohoku.ac.jp
†Department of Electrical and Computer Engineering, Portland State University, Portland, OR, USA
Email: mperkows@pdx.edu
‡Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland
Department of Theoretical Physics and Computer Science, University of Lodz, Lodz, Poland, Email: p.kerntopf@ii.pw.edu.pl

Abstract

In this paper we present a method for minimizing reversible quantum circuits using the Quantum Operator Form (QOF); a new representation of quantum circuit and of quantum-realized reversible circuits based on the CNOT, CV and CV† quantum gates. The proposed form is a quantum extension to the well known Reed-Muller but unlike the Reed-Muller form, the QOF allows the usage of different quantum gates. Therefore QOF permits minimization of quantum circuits by using properties of different gates than only the multi-control Toffoli gates. We introduce a set of minimization rules and a pseudo-algorithm that can be used to design circuits with the CNOT, CV and CV† quantum gates. We show how the QOF can be used to minimize reversible quantum circuits and how the rules allow to obtain exact realizations using the above mentioned quantum gates.

I. Introduction

In quantum and reversible circuits synthesis methods various representations are used for minimization, mapping or manipulation. The most famous of these forms is the Reed-Muller family (also known as Zhegalkin Polynomials) of expansions [17], [11], [13]. Reed-Muller form is particularly well suited for reversible logic because it is based on two-level AND/EXOR which can be directly mapped into reversible circuit using the Toffoli gates for instance.

However, it is now well known that mapping reversible circuits to reversible gates and then performing a technology mapping - such as mapping to the set of elementary gates CNOT, CV/CV† or to a LNN [1], [2], [3], [14], [12], [9] architecture restricted set does not always generate a minimal result. Also, it is not possible to obtain minimal realization on the level of quantum gates when minimizing solely on the level of Toffoli gates [15].

In this paper we propose the so called Quantum Operator Form (QOF) which is a quantum-expanded counter part of the classical Reed-Muller expansion. Starting from a set of simple rules extracted from the interaction of the CV and CV† operators we generalize them to various conditions and provide a mechanism to map arbitrary reversible circuit directly to quantum primitives. Using the rules of the interaction between quantum operators we then show how QOF permits for minimization and how to obtain forms of the QOF.

This paper is organized as follows. Section II introduces the Reed-Muller and more general the ESOP forms used for representation of reversible circuits. Section III describes the operators used and their interaction based. Section IV introduces a notation that allows to express operator required to allow a general quantum circuit expressions. Section V explains the principle of the QOF and Section VI provides an example demonstrating the capabilities of the QOF and simple rules of minimization. Finally Section VII introduces the notion of the weak canonicity of QOFs and a conclusion concludes the paper in Section VIII.

II. Reed-Muller Forms

In reversible logic the Reed-Muller (RM) form is very popular because a logic function expressed in the RM form can be directly mapped to a sequence of Toffoli gates. The RM form can be expressed by

$$F = \bigoplus_{i=1}^{2^n} \alpha_i c_i \tag{1}$$

with α_i being Boolean coefficients indicating if the c_i term is part of the RM expression or not [16]. The c_i term represents any product of positive or negative polarity literals. Consequently depending on the type of the term used there are three main classes of the RM forms: the PPRM (positive polarity Reed-Muller), FPRM (fixed polarity Reed-Muller) and GRM (generalized Reed-Muller). Examples of these forms are given in eq. 2

$$\begin{aligned} F_{PPRM} &= abc \oplus ad \oplus bce \oplus ade \\ F_{FPRM} &= abc \oplus a\bar{d}e \oplus bc\bar{d} \\ F_{GRM} &= abc \oplus \bar{a}bd \oplus e\bar{d} \end{aligned} \tag{2}$$

The difference between these forms and ESOP is that ESOP in general can be minimized by combining minterms to obtain one RM expression. For instance let $f_{esop} = abc \oplus a\bar{b}c \oplus abd$. Naturally, the two product terms abc and $a\bar{b}c$ in f result in ac.

The RM forms are very well suited for reversible circuit design because each term is directly mapped to a Toffoli gate. For instance the three forms described in eq. 2 are shown in corresponding circuits in Figure 1.

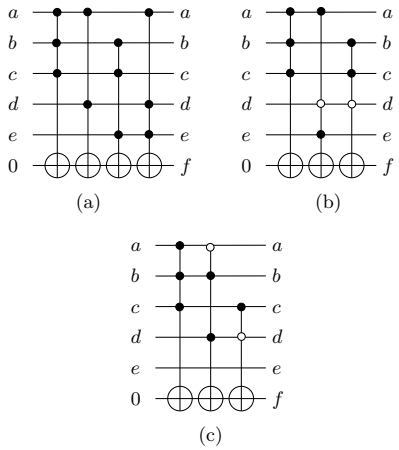


Fig. 1: Example circuits for (a) PPRM, (b) FPRM, (c) GRM.

Definition 1 (Toffoli gate) *A Toffoli gate is a single NOT operator controlled by a product of literals with positive and/or negative polarities.*

A literal is a variable or a negation of a variable and thus a Toffoli is a quantum gate that can have positive or negative controls. In pictorial representation of reversible gates positive controls are denoted by "black dots" and negative controls are denoted by "white dots" (see Figure 1).

Corollary 1 (Toffoli gate decomposition) *A Toffoli gate with $n > 2$ control bits can always be decomposed to $2^{n-2} + 1$ Toffoli gates with 2 control bits and with $n - 2$ ancilla bits.*

Notice that corollary 1 is quite easy to prove and thus in the case if the ancilla bits are not being reused the number of Toffoli gates in the decomposition is halved.

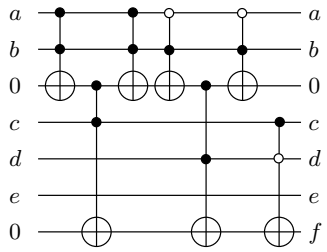


Fig. 2: Circuit from Figure 1(c) expanded to Toffoli gates with two control lines.

Corollary 2 (Upper bound on the Reed-Muller Form I) *An arbitrary Boolean function expressed in the RM form with $k > 1$ product terms can be built with a maximum of $2^{n_k-2} + k; n > 2$ two-bit controlled Toffoli gates; with n_k being the number of control bits in the k^{th} product term.*

The proof is easy to be verified by simply applying to arbitrary RM form the decomposition from corollary 1.

III. Permutative and Non-permutative Quantum Operators

The proposed Quantum Operator Form in this paper is using the CV/CV† and CNOT two-qubit quantum gates. Moreover the quantum gates are used with both positive and negative control variables. All the used gates are shown in Figure 3.

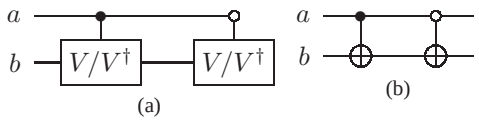


Fig. 3: The two-qubit primitive gates used in QOF (a) the CV/CV† gate with positive and negative control and (b) the CNOT gate with positive and negative control bit.

The V and V† gates are known for being the so called square-root of the NOT gate. Equation 3 represents this property in a formal way.

$$\begin{aligned} V * V = V^\dagger * V^\dagger = NOT \\ V * V^\dagger = V^\dagger * V = I \\ V * NOT = NOT * V = V^\dagger \\ V^\dagger * NOT = NOT * V^\dagger = V \end{aligned} \tag{3}$$

The V quantum operator has its matrix shown in eq. 4 and V† is its hermitian conjugate.

$$V = \begin{pmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{pmatrix} \quad V^\dagger = \begin{pmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{pmatrix} \tag{4}$$

Consequently, the CV and the CV† quantum gates are also called root squares of the CNOT permutative gate. The property from eq. 3 is known to have given rise to a family of Peres gates, all at the same cost [5]. Some of them are shown in Figure 4.

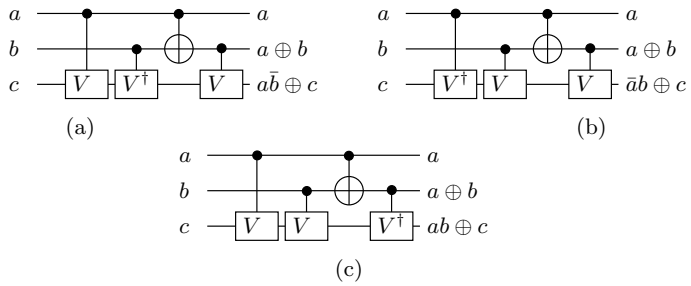
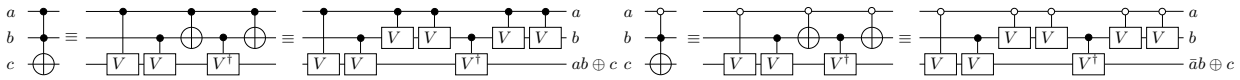


Fig. 4: Three types of the Peres gates generated by the permutation of the V and V† operators.

Corollary 3 (Quantum Operator Decomposition) *A two-qubit controlled NOT gate (Toffoli gate) can be decomposed into five two qubit quantum gates that can be directly mapped into hermitian operators such as electromagnetic or laser pulses. Examples of such decompositions are shown in Figure 5a. The decomposition can be further extended to seven quantum CV/CV† two-qubit quantum operators. Examples of such decompositions are shown in Figure 5b.*

Corollary 5b is based on the well known decomposition of Toffoli gates using CV/CV† primitives and thus it is easy to notice that it is true.



(a) $ab \oplus c$ decomposed using CNOT/CV/CV† and with CV/CV† (b) $\bar{a}b \oplus c$ decomposed using CNOT/CV/CV† and with CV/CV†

Fig. 5: Example of Quantum Operator Decomposition of the Toffoli gate.

Corollary 4 (Upper bound on the Reed-Muller Form II) *An arbitrary Boolean function expressed in the RM form with $k > 1$ product terms can be built with a maximum of $5 * 2^{n_k - 2} + k; n > 2$ two-qubit quantum operators.*

Again corollary 4 is easily provable by applying the decomposition from corollary 3 to a RM form.

IV. Notation for Generalized Reed-Muller Forms

To represent the operator expansions as described in Corollary 1 and 3 an extended notation is used. This notation allows to express not only product terms controlling a NOT gate but also product terms controlling an arbitrary unitary operator as well as additional information allowing to precisely locate a particular quantum operator one some variables. To express two-qubit operator such as Controlled-V we use the following notation:

$$a_t^V \equiv \begin{array}{l} a \\ t \end{array} \quad \begin{array}{l} \bullet \\ \boxed{V} \end{array} \tag{5}$$

where the literal or product of literals are the control variables, the subscript shows the target variable of the applied unitary transform and the superscript shows what unitary transform is being applied. Eq. 5 shows an example of this generalized notation. A multi-qubit controlled terms use the following representation:

$$(abd)_t^V \equiv abd_t^V \equiv \begin{array}{l} a \\ b \\ c \\ d \\ t \end{array} \quad \begin{array}{l} \bullet \\ \bullet \\ \\ \bullet \\ \boxed{V} \end{array} \tag{6}$$

Eq. 5 and 6 show that anytime a variable is present in a product term, it always controls an operator that changes the value of a target bit(s). Finally note that a multi-qubit function can be such that generates output on some of the input qubits and does not use dedicated output qubits: e.g. CNOT or SWAP. In such case the above introduced notation using the dedicated output variable t is modified to the target variable b by changing the subscript:

$$\begin{aligned} a \oplus b =& (a \oplus b)_b^{NOT} \\ =& (a \oplus b)_b = \begin{array}{l} a \\ b \end{array} \quad \begin{array}{l} \bullet \\ \oplus \end{array} \quad \begin{array}{l} a \\ a \oplus b \end{array} \end{aligned} \tag{7}$$

For the simplicity of notation and better understanding, the NOT operation is dropped from the superscript: This is shown by the equivalence of expressions in eq. 7.

Notice that this notation creates cascades of either classical Toffoli gates or cascades of general quantum gates. For instance expanding the Toffoli gate a cascade of $CNOT, CV^\dagger, CNOT$ is created in such manner that the $cV^\dagger$ depends on the $CNOT$ gate. The dependency between gates in a cascade can be easily indicated by the correct subscripts and control variables, i.e. $(a \oplus b)_b b^\dagger (a \oplus b)$. However it is also required to point out that the order of the gates cannot be changed in the current form, i.e. $b^\dagger (a \oplus b)_b (a \oplus b) = b^\dagger$. Thus we introduce the $\circ$ operation as a notation for a sequence of gates that cannot be altered and that depend on each other with some intermediary variables. Using this notation a Toffoli gate written in the generalized notation and decomposed to its CV/CV† and CNOT primitives will take the following form:

$$\begin{aligned} F =& ab \oplus c \\ =& a_c^V b_c^V (a \oplus b)_b \circ b_c^{V\dagger} \circ (a \oplus b)_b \end{aligned} \tag{8}$$

Note that terms and operators connected by $\circ$ are considered as one and without manipulation cannot be separated.

Finally in the generalized RM notation for quantum circuit, the joining operation is not always the XOR. Thus individual terms are separated by the following rules:

- $\circ$ separates terms as described above,
- a superscript indicating what operation is being applied to the last variable in the control term (example in Figure 7)
- a superscript and a subscript indicating what unitary operator is being applied to which variable.

V. Quantum Operator Forms

The Quantum Operator Form allows to obtain an optimized representation of the quantum circuit. However, the transformations leading to the final form may seem counterintuitive because they create more complex circuits and require gates that might not be realizable. However, these transformations formalize the beginning of a step by step effective optimization process.

First, let us define the arbitrary controlled unitary operator:

Definition 2 *Arbitrary controlled Unitary operator (ACUO) is a single qubit quantum operator that is controlled by positive and/or negative literals.*

An example of ACUO using the NOT operator are shown in Figure 1.

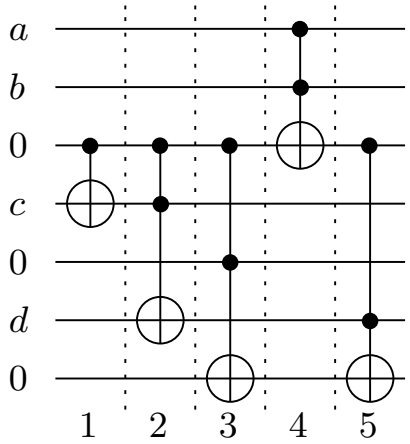


Fig. 6: Example of uninterrupted and interrupted lines: gates 1,2,3 and 5 are on a interrupted quantum line by gate 4. Gates 1 and 3 and 2 and 3 are on uninterrupted quantum lines but gates 1 and 2 are on interrupted quantum lines.

Definition 3 (Uninterrupted Quantum Line) *A line segment in a quantum circuit between two control inputs* a *and* b *laying on this line is called uninterrupted if the two points can be put as close to each other as possible.*

Corollary 5 *When the quantum gates having control inputs* a *and* b *are hermitian then the order of these gates can be reversed.*

Definition 4 (Interrupt Point) *An interrupt point is the output qubit of a quantum gate such as V, CNOT, Toffoli or so.*

Definition 5 (Terminal Gate) *A gate is called terminal if it has the target bit on one of the function output variables.*

Corollary 6 (Non-Terminal Gate CNOT Transformation) *Any terminal CV/CV† operator combined with a cascade of non-terminal quantum gates $CNOT$ can be expanded into a set of multi-qubit controlled terminal gates. Each of the resulting terminal gates corresponds to one of the terms of the expanded CNOT gate cascade. An example of such expansion is shown in Figure 7. Because expanded function is a sequence of EXORS (reversible function) of variables or products of variables, the number of obtained gates is given by $e = 2^{n-1}$; where n is the number of control bits.*

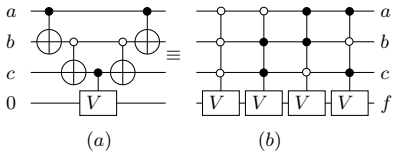


Fig. 7: Expansions - Non-Terminal Gate Transformation - Illustrated on the $(\bar{a} \oplus b \oplus c)^V$.

Proof: The proof is simple because the Non-Terminal Expansion concerns only product of single literal terms that are a sequence of XOR gates. As EXOR is a reversible logic gate, it always generates $2^n/2$ zeros and ones. Thus an EXOR of variables of the form $a \oplus b \oplus \ldots k \oplus l$ will generate an output function with exactly $2^l/2$ ones and $2^l/2 = 2^{l-1}$. ■

Definition 6 (Linearized Quantum Circuit) *A quantum circuit is called linearized if any gates defined on the exactly the same control qubits can be permuted without any change to the output function on the output variables.*

Thus Figure 7b is an example of a Linearized Quantum Circuit because all the gates can be permuted without modifying the output function. Notice that in the Linearized Quantum Circuit the only remaining bit operations are either the control or the unitary transform applied to the target qubit(s); this means that the Linearized Quantum Circuit is functionally equivalent to Reed-Muller with different gates. This allows us to define the Permutation Equivalent Gate.

Definition 7 (Permutation Equivalent Gate) *Two gates g_1 and g_2 in a linearized quantum circuit are called Permutation Equivalent (PE) if*

1) they are defined on completely different set of variables.

2) they use the same output bit as the target of the quantum operator.

For instance, a Toffoli gate shown in Figure 8a can be linearized to the Toffoli gate shown in Figure 8b by factoring the $(a \oplus b)_b$ into two adjacent $a\bar{b}V^\dagger$ and $\bar{a}bV^\dagger$ quantum gates. Consequently, Figure 8c shows that the CV gates (the right most gates in Figure 8a and 8b now can be moved arbitrarily in the resulting quantum circuit without changing the output function.

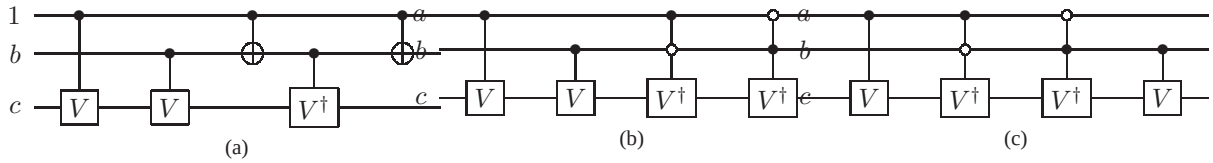


Fig. 8: (a) Toffoli gate, (b) its linearized equivalent and (c) a linearized Toffoli gate with a permuted CV gate.

Note that the linearization created a new type of Toffoli gates: the multi-controlled V/V† gates with mixed control (i.e. with positive/negative control inputs). Such gates (as $a\bar{b}V^\dagger$) might not always be realizable and thus we introduce the notion of *Virtual Gates*.

Definition 8 (Virtual Gate) *A Quantum Virtual Gate is any quantum gate including such quantum gates that cannot be used for the circuit design but only as intermediary stages during minimization. All Quantum Virtual Gates must be transformed into real quantum gates once the minimization process is finished.*

Corollary 7 (Creation of Virtual Gates) *The non-terminal gate transformation from Definition 6 creates Virtual Gates with as many control input points as the expanded control bit function.*

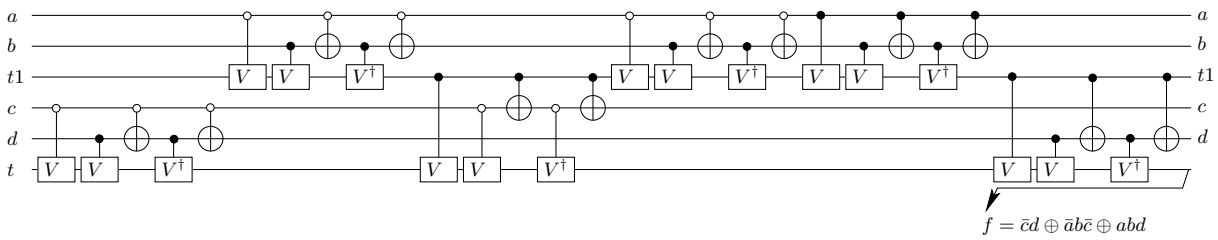


Fig. 9: Circuit corresponding to the expansion described by eq. 10

For example, in a Toffoli gate, the expanded function during the linearization process is the $(a \oplus b)^V$ function and thus it will create 2 two-qubit controlled V gates $\bar{a}b^V$ and $a\bar{b}^V$.

Definition 9 (Quantum Operator Form (QOF)) *A QOF of a Boolean function is given by a set of quantum operator terms of a circuit in a linearized form. A QOF of a logic function contains only one type of a controlled function and is using Quantum Virtual Gates.*

Example 1 (QOF of ESOP) *Let $f = \bar{c}d \oplus \bar{a}b\bar{c} \oplus abd$. Using the approach described above we can transform f into the QOF by consecutive steps. First, expand function f into a two-control-bit Toffoli gates and explicitly adding all intermediary variables denoted tk and the output variable denoted by t:*

$$\begin{aligned} f =& \bar{c}d \oplus \bar{a}b\bar{c} \oplus abd \\ =& \bar{c}d_t \oplus [\bar{a}b_{t1} \circ t1c_t \circ \bar{a}b_{t1}] \oplus [ab_{t1} \circ t1d_t \circ ab_{t1}] \end{aligned} \tag{9}$$

The circuit corresponding to the expansion in eq. 9 is shown in Figure 10. Notice that we remove the Toffoli gate restoring the intermediary ancilla bits as they are not necessary for further computation. From eq. 9 we next convert all the Toffoli gates

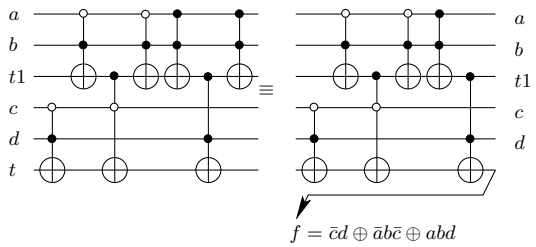


Fig. 10: Circuit corresponding to the expansion described by eq. 9

into the CV/CV† and CNOT two-bit operators.

$$\begin{aligned}
f =& \bar{c}d_t \oplus \bar{a}b_{t1} \circ t1c_t \circ \bar{a}b_{t1} \oplus ab_{t1} \circ t1d_t \circ ab_{t1} \\
=& \bar{c}_t^V d_t^V [\bar{c} \oplus d_d \circ d_t^{V\dagger} \circ \bar{c} \oplus d_d] \\
& \oplus \bar{a}_{t1}^V b_{t1}^V [\bar{a} \oplus b_b \circ b_{t1}^{V\dagger} \circ \bar{a} \oplus b_b] \\
& \circ t1\bar{c}_t^V \circ \bar{a}_{t1}^V b_{t1}^V [\bar{a} \oplus b_b \circ b_{t1}^{V\dagger} \circ \bar{a} \oplus b_b] \\
& \oplus a_{t1}^V b_{t1}^V [a \oplus b_b \circ b_{t1}^{V\dagger} \circ a \oplus b] \circ t1d_t^{V\dagger} \\
=& \bar{c}_t^V d_t^V [\bar{c} \oplus d_d \circ d_t^{V\dagger} \circ \bar{c} \oplus d_d] \\
& \oplus \bar{a}_{t1}^V b_{t1}^V [\bar{a} \oplus b_b \circ b_{t1}^{V\dagger} \circ \bar{a} \oplus b_b] \\
& \circ t1_t^V \bar{c}_t^V [t1 \oplus \bar{c}_c \circ c_t^{V\dagger} \circ t1 \oplus \bar{c}_c] \\
& \circ \bar{a}_{t1}^V b_{t1}^V [\bar{a} \oplus b_b \circ b_{t1}^{V\dagger} \circ \bar{a} \oplus b_b] \\
& \oplus a_{t1}^V b_{t1}^V [a \oplus b_b \circ b_{t1}^{V\dagger} \circ a \oplus b_b] \\
& \circ t1_t^V d_t^V \circ [t1 \oplus d_d \circ d_t^{V\dagger} \circ t1 \oplus d_d]
\end{aligned} \tag{10}$$

The circuit corresponding to eq. 10 is shown in Figure 9. Observe that in eq. 10 some terms are in square brackets. These terms are the ones that will be expanded using the non-terminal gate expansion and will result in virtual quantum gates. This leads in eq. 11 that represents a non-minimized QOF:

$$\begin{aligned}
f =& \bar{c}_t^\nabla d_t^\nabla \bar{c} \bar{d}_t^{\nabla\dagger} c d_t^{\nabla\dagger} \\
& \oplus \bar{a}_{t1}^\nabla b_{t1}^\nabla \bar{a} \bar{b}_{t1}^{\nabla\dagger} a b_{t1}^{\nabla\dagger} \\
& \circ t1_t^\nabla \bar{c}_t^\nabla t\bar{1} \bar{c}_t^{\nabla\dagger} t1 c_t^{\nabla\nabla} \\
& \circ \bar{a}_{t1}^\nabla b_{t1}^\nabla \bar{a} \bar{b}_{t1}^{\nabla\dagger} a b_{t1}^{\nabla\dagger} \\
& \oplus a_{t1}^\nabla b_{t1}^\nabla \bar{a} b_{t1}^{\nabla\dagger} a \bar{b}_{t1}^{\nabla\dagger} \\
& \circ t1_t^\nabla d_t^\nabla t\bar{1} d_t^{\nabla\dagger} t1 \bar{d}_t^{\nabla\dagger}
\end{aligned} \tag{11}$$

Notice that the V/V† operators have been replaced by $\nabla/\nabla\dagger$ to indicate that some of the gates can be virtual gates and this is only a transitory form for circuit minimization. The corresponding circuit is shown in Figure 11.

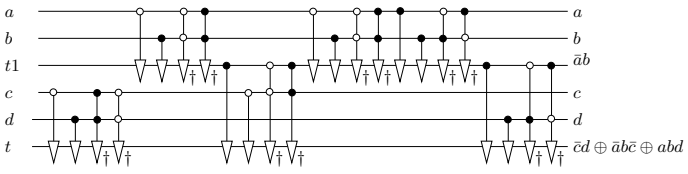


Fig. 11: Circuit using the virtual gates for the function *f*

VI. Minimizing QOF

From eq. 11 several minimizations can be performed using the notion of the Permutation Equivalent Gates (Definition 7)) that are located on Uninterrupted Lines (Definition 3). The first step is to find uninterrupted lines. This can be simply done by finding such control lines that do not appear in any term's subscript. From eq. 11 variable C and D are uninterrupted and gates defined solely on these variables can be grouped together and combined.

The minimizations steps are shown in Figure 12. Figure 12a shows that all gates that are defined on the same uninterrupted lines in Figure 11 are moved all to the right side of the circuit.

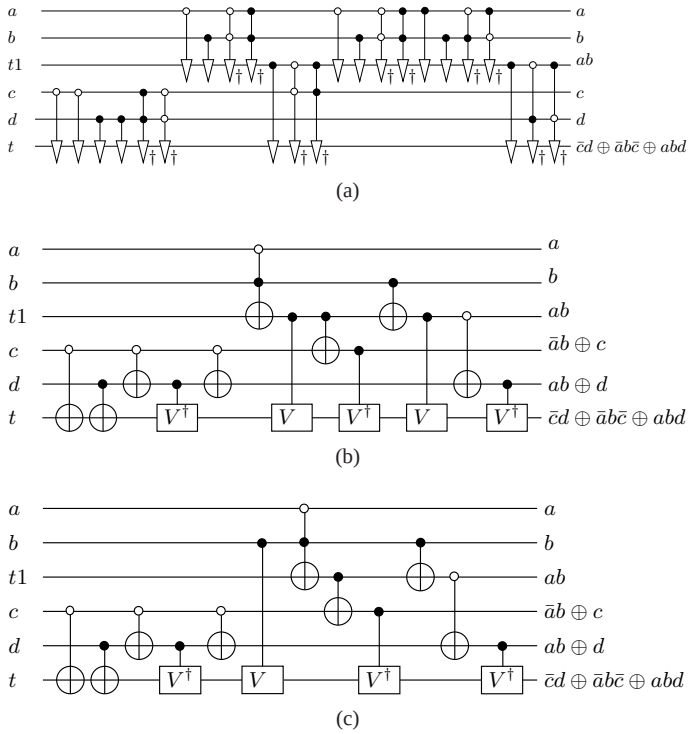


Fig. 12: Steps of the minimization of the function *f* shown in Figure 11

Next step is to look for gates that have the same output variable but are on interrupted line. The two topmost Toffoli gates are such gates and can be minimized by using the fact that it is possible to go from a two variable product term to another with at maximum two EXOR forms [4]. This is shown in Figure 12b. It is also possible to perform more complex pattern matching transformations. Figure 12c shows the merging of $\bar{A}B_{t1} \circ t1_t^V$ and $\bar{A}B_{t1} \circ B_{t1}^V \circ t1_t^V$ to B_t^V. In the QOF notation this amounts to search again for terms that are using the same control lines and then factoring out the interrupt points that allows to merge similar gates. Thus, from eq. 11 we obtain (we show only the concerned terms):

$$\begin{aligned} f = & \bar{a}_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} \bar{b}_{t1}^{\triangledown} a b_{t1}^{\triangledown} \\ & \circ t1_t^{\triangledown} \bar{c}_t^{\triangledown} t\bar{1} \bar{c}_t^{\dagger} t1 c_t^{\triangledown} \\ & \circ \bar{a}_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} \bar{b}_{t1}^{\triangledown} a b_{t1}^{\triangledown} \\ & \oplus a_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} b_{t1}^{\triangledown\dagger} a \bar{b}_{t1}^{\triangledown\dagger} \\ & \circ t1_t^{\triangledown} d_t^{\triangledown} t\bar{1} d_t^{\triangledown\dagger} t1 \bar{d}_t^{\triangledown\dagger} \end{aligned} \tag{12}$$

First, the two topmost Toffoli gates $\bar{a}_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} \bar{b}_{t1}^{\triangledown} a b_{t1}^{\triangledown}$ and $a_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} b_{t1}^{\triangledown\dagger} a \bar{b}_{t1}^{\triangledown\dagger}$ result in the second Toffoli gate being transformed to $b_{t1}^{\triangledown} b_{t1}^{\triangledown}$.

$$\begin{aligned} f = & \bar{a}_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} \bar{b}_{t1}^{\triangledown} a b_{t1}^{\triangledown} \\ & \circ t1_t^{\triangledown} \bar{c}_t^{\triangledown} t\bar{1} \bar{c}_t^{\dagger} t1 c_t^{\triangledown} \\ & \oplus b_{t1}^{\triangledown} b_{t1}^{\triangledown} \\ & \circ t1_t^{\triangledown} d_t^{\triangledown} t\bar{1} d_t^{\triangledown\dagger} t1 \bar{d}_t^{\triangledown\dagger} \end{aligned} \tag{13}$$

Second, as already introduced above and using the same method as in the previous case - searching for gates defined on the

same control variables: two CV gates defined $t1_t^{\triangledown}$ can be combined to:

$$\begin{aligned} f =& \bar{a}_{t1}^{\triangledown} b_{t1}^{\triangledown} \bar{a} \bar{b}_{t1}^{\triangledown} a b_{t1}^{\triangledown} \\ & \oplus b_t^{\triangledown} \circ \bar{c}_t^{\triangledown} t\bar{1} \bar{c}_t^{\dagger} t1 c_t^{\triangledown} \\ & \oplus b_{t1}^{\triangledown} b_{t1}^{\triangledown} \\ & \circ d_t^{\triangledown} t\bar{1} d_t^{\triangledown\dagger} t1 \bar{d}_t^{\triangledown\dagger} \end{aligned} \tag{14}$$

Finally, the form can be re-composed into Toffoli gates and is shown in Figure 12c.

VII. Canonicity of QOF

One of the important properties of PPRM is that it is a canonical form. Similarly, a desired property of the QOF is to be canonical. In the next section we provide a proof that under certain restrictions the QOF is a canonical representation and will can be called Quantum Operator Canonical Form (QOCF).

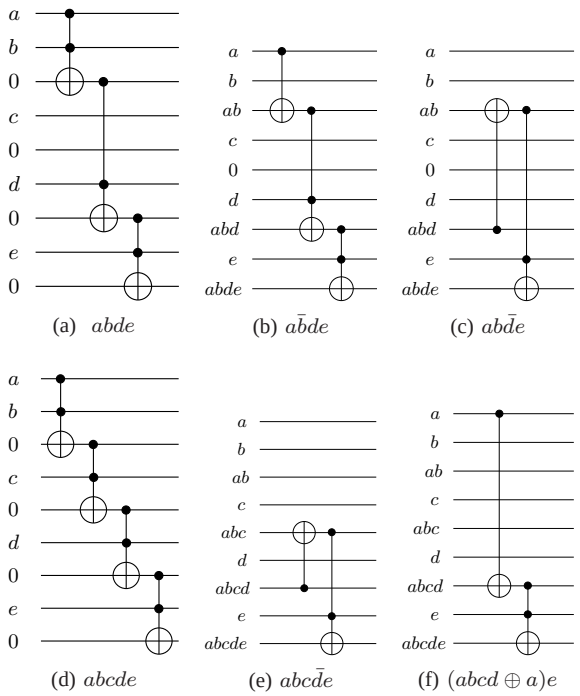
Definition 10 (Toffoli Gates Reduction) *Two Toffoli gates $T1$ and $T2$ defined on m and n control variables, respectively, can be reduced to one Toffoli gate with m control bits and to one XOR gate and one Toffoli gate with j control bits if:*

$$\{v_0, \ldots, v_j, \ldots, v_n\} = R(u_0, \ldots, u_j) P(v_{j+1}, \ldots, v_m\} \tag{15}$$

with $\{u_0, \ldots, u_m\}$ and $\{v_0, \ldots, v_n\}$ being the polarities of the control variables of the $T1$ and $T2$ gates, respectively. The P term represents the product of $m - j + 1$ variables and the R term represents the following irreducible and non-expanding operation:

$$R = u_e \ldots u_g \oplus u_e \ldots u_{g+1} \tag{16}$$

Examples of such reductions are shown in Figures 13a- 13e. Observe that this reduction means that any product of literals (a Toffoli gate) can be moved on the K-map to one of its adjacent minterms by the means of the transformation from Definition 10. Figure 13a is a term from which new terms are obtained by the transformations shown in Figures 13b and 13c.



(a) $abde$ (b) $a\bar{b}de$ (c) $abd\bar{e}$

(d) $abcde$ (e) $abc\bar{d}e$ (f) $(abcd \oplus a)e$

Fig. 13: Example of non reducible and reducible Toffoli gate reductions.

Similarly the term shown in Figure 13d is transformed into new terms using the circuits shown in Figures 13e and 13f. Observe that the transformation shown in Figure 13f is not a direct transformation as defined in Definition 10 and is shown as counter example.

Theorem 1 (Weak Canonicity of QOF) *A QOF of a Reed-Muller after the application linearization minimization (Section VI) and the transformation from Definition 10 is canonical with respect to the polarity of used literals if and only if variables wires are not altered and not permuted.*

Observe that based on Theorem 1, all intermediary and final products of literals can be created and manipulated only on the ancilla wires.

Proof: The remaining terms in the QOF are either defined on completely different variables (disjoint support set), on terms that are not adjacent on the K-map table or terms that are split by different Toffoli gates (Figure 14). Figure 14b shows

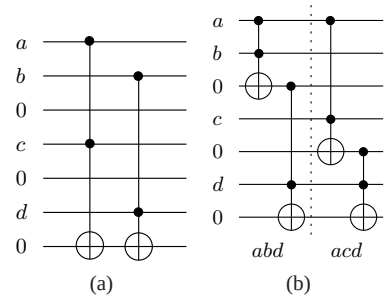


Fig. 14: Example of (a) disjoint support set product terms and (b) of terms that cannot be minimized without reordering the variables.

two Toffoli gates that cannot be minimized because they are defined on completely variables - also called disjoint support set. Figure 14b shows that terms abd and acd are both split by two Toffoli gates. Finally, Figure 14b show that if the variables are reordered from $abcd$ to $adbc$ one Toffoli gate can be removed thus minimizing the circuit.

As was shown, the linearization allows to remove all compatible terms defined on same variables (even within other more complex Toffoli gates) and adjacent Toffoli gates can be minimized by CNOT gate replacement. Consequently, a minimization that would further reduce the circuit would also require variable change, additional ancilla bit or the change of the value of a variable wire. Example of such transformations are for instance template reduction [8], [10]. ■

The algorithm to achieve the weak canonical form of the QOF can be obtained by combining an ordering algorithm such as the one described in [7] with a minimization procedures of the linearized quantum circuits. An algorithm for this approach is shown below in the pseudo-code 1.

Algorithm 1 Pseudo-code for generating Weakly Canonical QOF.

1: Order the XOR form of the circuit using Algorithm from [7]
2: Minimize the XOR form of the circuit using techniques from [7]
3: Minimize the resulting circuit using the linearization of the quantum circuit
4: Minimize the circuit using the Toffoli gate reduction method from def. 10

The importance of the weak canonicity is related to minimization and representation of quantum Boolean circuits. A canonical representation for quantum circuits is advantageous over the classical Reed-Muller because it not only it shows a non reducible representation using only truly quantum gates but it is also useful to be used as basis for other minimization methods for quantum circuits.

VIII. Conclusion

In this paper we presented an extension of the work on the symbolic operator approach introduced in [6]. The QOF can be used as a universal language to specifying a quantum function or a quantum circuit with a set of unitary operators and provides a set of tools to manipulate and minimize them.

Future work includes the introduction of efficient rules for the minimization of such expressions and an algorithm for verification of results.

Acknowledgments

P. Kerntopf was supported in part by the Polish Ministry of Science and Higher Education under Grant 4180/B/T02/2010/38.

REFERENCES

[1] A. Chakrabarti and S Sur-Kolay. Nearest neighbour based synthesis of quantum Boolean circuits. *Engineering Letters*, 15:2:356–361, 2007.

[2] A. Chakrabarti and S. Sur-Kolay. Rules for synthesizing quantum Boolean circuits using minimized nearest-neighbour templates. In *Proceedings of the 15th International Conference on Advanced Computing and Communications*, pages 183–189, 2007.

[3] Y. Hirata, M. Nakanishi, and S. Yamashita. An efficient method to convert arbitrary quantum circuits to ones on a linear nearest neighbor architecture. In *3^{rd} International conference on Quantum, Nano, and Micro Technologies*, pages 26–33, 2009.

[4] M. Lukac, M. Kameyama, M.D. Miller, and M. Perkowski. High speed genetic algorithms in quantum logic synthesis: Low level parallelization vs. representation. *Journal of Multiple-Valued Logic and Soft Computing*, 20(1-2):89–120, 2012.

[5] M. Lukac and M. Perkowski. Using exhaustive search for the discovery of a new family of optimum universal permutative binary quantum gates. In *Proceedings of International Workshop on Logic & Synthesis*, 2005.

[6] M. Lukac and M. Perkowski. Evolutionary approach to quantum symbolic logic synthesis. In *In Proceedings of the IEEE Congress on Computational Intelligence (WCCI)*, pages 3374–3380, 2008.

[7] M. Lukac, M. Perkowski, M. Kameyama, N. Alhagi, and P. Kerntopf. Synthesis of reversible cascades from relational specifications,. In *Reed-Muller Workshop*, 2011.

[8] D. Maslov, G.W. Dueck, D.M. Miller, and C. Negrevergne. Quantum circuit simplification and level compaction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(3):436–444, 2008.

[9] A. Matsuo and S. Yamashita. Changing the gate order for optimal lnn conversion. In *In Proceedings of the 3rd Workshop on Reversible Computation*, pages 175–186, 2011.

[10] D. M. Miller, R. Wille, and R. Drechsler. Reducing reversible circuit cost by adding lines. In *Proceedings of the 40th IEEE International Symposium on Multiple-Valued Logic*, pages 217–222, 2010.

[11] D. E. Muller. Application of boolean algebra to switching circuit design and to error detection. *IRE Transactions on Electronic Computers*, 3:6–12, 1954.

[12] M. Perkowski, M. Lukac, D. Shah, and M. Kameyama. Synthesis of quantum circuits in linear nearest neighbor model using positive davio lattices. *Facta Universitatis*, 24(1):71–87, 2011.

[13] I. S. Reed. A class of multiple-error-correcting codes and their decoding scheme. *IRE Transactions on Information Theory*, 4:38–49, 1954.

[14] M. Saeedi, R. Wille, and R. Drechsler. Synthesis of quantum circuits for linear nearest neighbor architectures. *Quantum Information Processing*, 10(3):355 – 377, 2011.

[15] Z. Sasanian and D.M. Miller. Mapping a multiple-control Toffoli gate cascade to an elementary quantum gate circuit. *Journal of Multiple-Valued Logic and Soft Computing*, 18(1):83–98, 2012.

[16] T. Sasao. *Switching Theory for Logic Synthesis*. Kluwer Academic Publishers, 1999.

[17] I. L. Zhegalkin. Arithmetization of symbolic logic, (in Russian). *Matematiceskij Sbornik*, 35:311–373, 1928.