Reducing Quantum Cost in Reversible Toffoli Circuits

Marek Szyprkowski\(^1\)
1Institute of Computer Science
Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
m.szyprowski@ii.pw.edu.pl

Pawel Kerntopf\(^1,2\)
2Department of Theoretical Physics and Informatics
University of Łódź
Pomorska 149/153, 90-236 Łódź, Poland
p.kerntopf@ii.pw.edu.pl

Abstract—Recently, reversible circuit synthesis has been intensively studied. One of the problems that has not been solved for a long time was exact minimization of gate count (GC) in 4-bit circuits. Finally, last year a tool of practical usage for finding optimal gate count Toffoli networks for any 4-variable function was developed. However, not much work has been done yet on exact minimization of quantum cost (QC) in 4-bit circuits. This paper presents an application of the above mentioned tool to reducing QC of 4-bit reversible circuits. It is shown that for benchmarks and for designs taken from recent publications it is possible to obtain savings in QC of up to 74% comparing with previously known circuits.

Keywords—reversible circuits, synthesis, quantum cost

I. INTRODUCTION

A gate (circuit) is called reversible if there is a one-to-one correspondence between its inputs and outputs. Research on reversible logic circuits is motivated by advances in quantum computing, nanotechnology and low-power design. Therefore, reversible logic synthesis has been intensively studied. The attention has been focused on the synthesis of circuits built from the NCT library of gates.

Satisfactory reversible logic synthesis algorithms for arbitrary libraries of gates and cost functions have not been found yet. In addition, NCT library synthesis techniques developed for such circuits scale not well and optimal circuits not always can be found even for relatively small numbers of inputs and outputs. Many reversible circuit synthesis algorithms have been proposed. However, they sometimes generate very suboptimal circuits. Therefore, to improve circuit quality, different techniques, consisting of a series of local optimizations have been developed. However, the problem is that some circuits cannot be reduced only by local optimization. Also, for larger functions these algorithms require many iterations of local optimization that leads to a high runtime.

For many years only few exact optimal circuits have been found for \(n\)-variable functions with \(n > 3\). For example, it was even not known what is the maximal gate count in optimal circuits implementing 4-bit reversible functions. Last year a very fast tool capable of synthesizing optimal circuits for any 4-bit reversible specification was finally developed [2] what has become a real breakthrough. With this tool it was possible to establish that there are 144 4-bit functions requiring 15 gates in their optimal circuits and that there exists none requiring 16 gates [3, 12].

Quality of a reversible circuit is usually estimated by gate count (GC) or by a metric called quantum cost (QC). Much less effort has been devoted to minimization of QC in reversible circuits. Maslov et al. [12] used a mixture of different techniques (including MMD algorithm, Reed-Muller spectra based algorithm, template application and resynthesis) to improve either gate count or quantum cost what led to improving results for some benchmarks from [12]. However, exact minimization of QC was not the aim of this approach. Donald and Jha [1] added a new option for optimizing QC to their earlier algorithm. Due to this they were able to improve results for some benchmarks but have attempted to find exact minimal values for small circuits only. They performed similar experiments for an extended library of gates including also SWAP, Fredkin and Peres gates. Wille et al. [27] formulated a synthesis problem as a quantified Boolean formula and then solved it by applying Binary Decision Diagrams. This enabled to find the minimal as well as the maximal QCs for the specified number of gates up to seven gates. However, these calculations were performed only for minimal gate count circuits. Grosse et al. [4] considered synthesis for networks made of multiple-control Toffoli gates using SAT-like engines. Their approach to reducing quantum cost was the same, i.e. minimization of the number of gates as the first step and only then trying to reduce quantum cost with the fixed gate count. However, as we will show in the paper, finding exact minimal quantum cost circuits for larger functions requires considering circuits having greater number of gates than the minimal size ones.

Some efforts have been recently made to reduce QCs of designs. One of the recent approaches consists in looking for circuit realizations using quantum elementary gates like NOT, controlled NOT, two square roots of NOT [16, 17, 20, 21] or Hadamard gates [15]. However, we show that significant reduction of QC can also be obtained without considering elementary quantum gate library.

First, we have developed a tool similar to the one reported earlier in [2] capable of finding a circuit having a minimal number of gates for any 4-bit reversible function. Then we have programmed and run an approach capable of finding all circuits implementing a given reversible function with a specified value of QC. Our calculations have led to obtaining savings in QC over 50% for some known benchmark functions and for designs reported in recent publications on synthesis of reversible circuits. At the same time we have shown that the numbers of all quantum cost minimal circuits for larger designs achieve the order of 10\(^7\).
The paper is organized as follows. Section II recalls basic concepts of reversible logic. In Section III notions of cost functions and optimal reversible circuits are introduced. Sections IV presents a survey of recent results on synthesis of 3-bit and 4-bit reversible Boolean circuits. In Section V details of the fast algorithm for optimal gate count synthesis of 4-bit circuits are overviewed. Section VI contains description of our approach to searching for minimal quantum cost 4-bit reversible circuits. In Section VII our experimental results are collected and compared to known circuits from benchmark pages and from the literature. Section VIII summarizes the paper with conclusions and suggestion for further research.

II. PRELIMINARIES

Definition 1. A completely specified n-input n-output Boolean function (referred to as an n*n function) is called reversible if it maps each input assignment into a unique output assignment.

There are 2^n reversible n*n Boolean functions. For n = 3 this number is equal to 40,320 and for n = 4 is greater than 2*10^13.

Definition 2. An n-input n-output (n*n) gate (or circuit) is reversible if it realizes an n*n reversible function.

In a reversible circuit fanout of each gate output is always equal to 1. As a consequence n*n reversible circuits can be only build as a cascade of k*k reversible circuits (k ≤ n).

Definition 3. A set of reversible gates that can be used to build reversible circuits is called a gate library.

Many gate libraries have been examined in the literature. The so called NCT library consists of 1*1 NOT, 2*2 CNOT and 3*3 TOFFOLI gates. In this paper we will also discuss the usage of 4*4 TOFFOLI gates and 3*3 PERES gates. Below we define all these gates.

Definition 4. 1*1 NOT(x) gate performs the operation (x) → (x ⊕ 1), 2*2 CNOT(x,y) gate performs the operation (x, y) → (x, x ⊕ y), 3*3 TOFFOLI(x,y,z) gate performs the operation (x, y, z) → (x, y, z ⊕ xy), 4*4 TOFFOLI(x,y,z,u) gate performs the operation (x, y, z, u) → (x, y, z, u ⊕ xyz), 3*3 PERES(x,y,z) gate performs the operation (x, y, z) → (x, x ⊕ y, z ⊕ xy), where ⊕ denotes XOR operation.

The first four of the above defined gates (denoted N, C, T, T4, respectively) invert one input if and only if all others are 1, passing the other inputs unchanged to corresponding outputs. Each of the N, C, T, T4 gates is invertible, i.e. equal to its own inverse. Let us note that a PERES(x,y,z) gate is equivalent to a TOFFOLI(x,y,z) gate followed by a CNOT(x,y) gate. Inverted PERES gate is not equal to PERES gate. We will denote them by P⁻¹ and P, respectively [13].

III. COST FUNCTIONS AND OPTIMAL REVERSIBLE CIRCUITS

For any reversible function there exist many reversible circuits implementing it. Thus a cost function has to be defined to evaluate the quality of a circuit. Usually, additive cost functions are applied. Therefore, adding a gate to a circuit leads to increasing its cost. A cost of each gate type is often expressed as a non-zero value. The cost of the (hypothetically) simplest gate is assumed to be 1.

A number of gate cost functions have been proposed (see, e.g. [28]). The simplest cost function of a reversible circuit is equal to total number of gates. It is called gate count (GC in short). Although this cost function probably would not be applicable to future technologies, it is used in the literature for comparing the quality of synthesis algorithms. Nevertheless, it could be treated as an approximation of practical cost functions for small reversible circuits.

Other cost functions are also considered. The most widely used, called quantum cost (QC), is based on cost of elementary quantum gates as each reversible gate can be built from several elementary quantum gates. It is assumed that the cost of each elementary quantum gate equals 1, so the cost of a reversible gate equals to the total number of elementary quantum gates used. The quantum cost of N, C, P, T and T4 gates is assumed to be 1, 1, 4, 5 and 13, respectively (see for example [5, 12]).

By an optimal circuit we mean the best possible implementation, i.e. the one having the minimal cost. The set of optimal circuits implementing a reversible function depends on a gate library and a cost function. It should be noted that a reversible function may have many equivalent optimal circuits. A database of all optimal circuits can be generated by building recursively new circuits from all gates that belong to the gate library and selecting only the circuits with the minimal cost after incrementing the number of gates in the circuits by 1. Such databases have been described in the literature, but usually only one optimal circuit is stored for each function (see, e.g. [14, 18]).

Let us define an equivalence class in the set of reversible functions with respect to the cost of optimal circuits realizing these functions.

Definition 5. Two reversible Boolean functions are called cost-equivalent if they belong to the same equivalence class (called a cost-equivalence class, or a conjugacy class in [2]) under simultaneous input/output relabeling and inversion.

It can be easily shown that two cost-equivalent functions have the same or reverse optimal circuits (i.e. optimal circuits for all cost-equivalent functions have the same cost for all cost metrics [2]). A cost-equivalence class can contain maximally 2*n! functions (number of permutations of all variables doubled by the possibility of inversion).

We will be using the following popular vector notation for an n-variable reversible Boolean function f:

\[ f(0), f(1), \ldots, f(2^n-1) \]

where each binary vector f(i) will be expressed as a decimal.
Definition 6. The **canonical representative of a cost-equivalence class** is the function whose vector \([f(0), f(1), \ldots, f(2^n - 1)]\) is lexicographically smallest.

**Example 1.** Let us consider a reversible function \(f\) given in Table I. Its vector notation is \([1, 0, 3, 2, 5, 7, 4, 6]\). It can be easily checked that this is the canonical representative of the cost-equivalence class of \(f\) as all functions belonging to this class are as follows:

\[
\begin{align*}
[1, 0, 3, 2, 5, 7, 4, 6] & \quad [1, 0, 3, 2, 6, 4, 7, 5] & \quad [1, 0, 3, 5, 4, 2, 6] \\
[1, 0, 6, 2, 5, 4, 7, 3] & \quad [2, 3, 0, 1, 6, 4, 7, 5] & \quad [2, 3, 0, 1, 6, 4, 7, 5] \\
[2, 3, 0, 7, 6, 1, 4, 5] & \quad [2, 5, 0, 1, 6, 7, 4, 3] & \quad [4, 3, 6, 7, 0, 1, 2, 5] \\
[4, 5, 3, 7, 0, 1, 2, 6] & \quad [4, 5, 6, 1, 0, 7, 2, 3] & \quad [4, 5, 6, 2, 0, 1, 7, 3]
\end{align*}
\]

IV. A SURVEY OF RECENT RESULTS ON SYNTHESIS OF GATE COUNT OPTIMAL 3*3 AND 4*4 CIRCUITS

We have generated databases of all optimal circuits for 3*3 reversible Boolean functions for NCT library and two cost functions (GC and QC). After generating the databases we were using them in our research on synthesis of reversible circuits [6-9, 22-24]. During this research we have found all reversible functions requiring the maximal number of 8 gates in gate count optimal circuits. Next we discovered that for some of these functions longer circuits implementing them exist with much smaller QC. Fig. 1 shows such an example in which adding one gate leads to reducing QC from 24 to 13, i.e. by 45.8% (taking into account that two pairs of TOFFOLI and CNOT gates form two PERES gates). In general, this property is encountered in circuits of relatively large size. Thus, when looking for minimal quantum cost circuits one have to consider circuits of different sizes. This observation attracted our attention to the problem of minimizing QC in reversible circuits.

The databases generated by us for NCT library revealed one more interesting property of the set of all optimal 3*3 circuits. Namely, the greater the size of an optimal circuit the greater the number of all optimal circuits implementing the given function, e.g. the maximal number of all gate count as well as quantum cost optimal 3*3 circuits is higher than 1000 [8]. One can expect that in case of greater numbers of inputs/outputs the same property will hold and the maximal number of all gate count optimal 3*3 circuits will grow at least exponentially. Our recent experiments for 4-bit functions presented at the end of this paper show that this hypothesis is justified.

| x0 | x1 | x2 | f(x1,x2) |
|----|----|----|---------|
| 0 0| 0 0| 0 1| 0 1      |
| 0 0| 0 1| 1 1| 0 1      |
| 0 1| 0 0| 1 0| 0 0      |
| 0 1| 0 1| 0 1| 1 0      |
| 1 0| 1 0| 0 0| 0 0      |
| 1 0| 1 1| 1 0| 0 0      |
| 1 1| 1 0| 0 1| 1 0      |
| 1 1| 1 1| 1 1| 1 1      |

Table I. Truth Table for Function F

| Gate count | # circuits |
|------------|------------|
| 11         | 12         |
| 12         | 2,288      |
| 13         | 187,945    |
| 14         | 11,056,332 |

Table II. Distribution of the Number of Circuits Implementing 4*4 Function NTH_PRIME4_INC [17] Over Their Size

Similar properties of reversible functions have been discovered by us with respect to the cardinality of the set of all (i.e. not only optimal) circuits implementing a given function. Table II shows how fast this number can grow for \(n = 4\) depending on the circuit size.

The above discussed properties demonstrate how difficult is the task of finding exact minimal quantum cost reversible circuits.

V. GATE COUNT OPTIMAL CIRCUITS SYNTHESIS

In the years 2007-2009 the problem of developing algorithms for synthesis of all gate count optimal 4*4 reversible circuits using NCT library have attracted attention of some researchers [10, 11, 25, 29-31]. Only in one of them it was claimed that a hash table has been constructed consisting all 8-gate optimal circuits which potentially enables constructing optimal circuits up to size of 16 gates [10]. However, besides one relatively simple example no experimental data were supplied. In addition, the example showed that the tool was quite slow.

Recently, Golubitsky, Falconer and Maslov [2] presented the algorithm for finding a gate count optimal reversible circuit for any 4*4 reversible function. The algorithm is based on the fact that the set of all functions that have an optimal circuit up to 9 gates can be effectively stored in memory of today computers. It also relies on the fact that for each cost-equivalent class of reversible functions it is sufficient to store only its canonical representative, so the amount of required memory can be reduced almost 48 times. The functions in such database are stored in hash tables.

The database can be used directly to find GC of an optimal circuit for any reversible function \(f\) up to 9 gates if the canonical representative of function \(f\) is used for the database lookup.

![Figure 1. Two circuits implementing the same reversible function](image-url)
Functions that require more than 9 gates in an optimal circuit need additional processing to determine GC of optimal circuits. Namely, the algorithm relies on the fact that any optimal circuit for the function \( f \) can be partitioned into two circuits which realize functions \( g \) and \( r \), such as:

\[
f = g \circ r,
\]

where symbol \( \circ \) denotes composition (cascading of the circuits). The above equation can be transformed into the following:

\[
f \circ r^{-1} = g.
\]

The algorithm iterates over all functions in the database that have optimal circuits of length \( i \) for \( i = \{1,...,9\} \), computes their inversions and then compose the function \( f \) with each of them. For the resulting function \( g \) it checks the length of an optimal circuit using the procedure from the beginning of the paragraph. If such function \( g \) with optimal circuit of length \( j \) has been found in the database, the length of the optimal circuit for the function \( f \) equals to \( i+j \) [2].

The above procedure provides an effective and very fast way of finding the length of an optimal circuit for any 4*4 reversible function. It can also be easily extended to a fast algorithm for finding an optimal circuit [2]. One just needs to store the last gate with each function in the database and the complete circuit can be easily constructed.

Using the computer system with 16 AMD Opteron 2300 MHz processors and 64GB RAM authors of [2] managed to create a database for 9-gate optimal circuits and synthesize an optimal circuit for a given 4*4 reversible function in about 0.01s on average.

A similar approach has been presented in [10]. However, the authors decided to store the complete circuits in the database. They only had managed to create a database for \( n=8 \). This decreased the synthesis speed so generating an optimal circuit for an example benchmark took approximately 35s.

VI. OUR SYNTHESIS ALGORITHM

We have decided to extend the method presented in [2] for developing a tool capable of finding all reversible circuits with a specified gate count implementing the given reversible function. The algorithm is very similar to many heuristic algorithms of reversible circuit synthesis described in the literature. In each step a reversible gate is selected and added at the end of the current gate cascade (see Fig. 2). It might be necessary to backtrack and try another gate if the circuit for the given reversible function has not been found (Fig. 3). For selecting a gate algorithms rely on the criterion called a complexity measure. It limits the search tree and allows to avoid selecting gates that do not lead to an optimal circuit [28], [29].

![Figure 2. A general scheme of one step of a search-based algorithm](image)

Definition 7. A partial realization of a given reversible Boolean function \( f \) is a gate cascade, such that after adding a gate (or gates) a circuit implementing \( f \) is created.

Definition 8. A remainder function is a Boolean function to be implemented in a circuit that has to be added to the current partial realization to get a circuit implementing a specified reversible function.

In Fig. 2 \( f^T \) denotes the remainder function to be implemented after selecting \( G \) as the first gate of the circuit under construction.

Definition 9. A complexity measure is a function that assigns a number to each reversible function in such a manner that functions with complex optimal circuits have higher measures than functions with simple optimal circuits. The perfect complexity measure assigns a monotonically decreasing numbers to remainder functions after adding consecutive gates in an optimal circuit for a given reversible function.

Note that the simple search-based algorithm is only guided by a complexity measure. It selects only the gates that generate decreasing complexity measure for the remainder functions. Even such simple algorithm can find more than one reversible circuit if it continues to run after finding the first reversible gate cascade that implements the specified function and use backtracking to check all paths in search tree.

VII. EXPERIMENTAL RESULTS

It can be easily noticed that the gate count calculated with the algorithm from [2] is a perfect complexity measure for the heuristic algorithms. We have decided to perform an experiment for finding quantum cost optimal circuits with specified sizes for previously proposed 4*4 reversible benchmarks. As a result of our calculations of all optimal circuits for 3*3 reversible functions we expected in advance that in some cases the runtimes could be very high. The results of our calculations are presented in Tables III-VI. The reversible circuits that have been found with this approach improve substantially previously reported circuits with best known quantum cost. The minimal value of the quantum cost of circuits found by us is sometimes more than two times smaller than the maximal one. For some benchmarks improvement was up to 74%.
Our initial analysis of optimal circuits for 3*3 reversible functions showed that the quantum cost optimal circuits have higher average GC than the gate count optimal circuits. This is why we considered an extension to the algorithm for synthesis of all 4*4 reversible circuits. Namely, increasing by a the initial value considered as the complexity measure for function $f$, the algorithm will find all reversible circuits of length equal to $o$, $o+1$, ..., $o+a$, where $o$ is the length of the optimal circuit for the function $f$. The value of the parameter $a$ needs to be adjusted experimentally to limit the calculation time and the number of generated circuits. Both the calculation time and the number of the circuits grow exponentially with the increasing value of $a$ (see Tables III-VI).

The proposed approach allows to find the 4*4 reversible quantum cost optimal circuits for a specified length of the reversible gate cascade. This does not mean that these circuits are exact minimal in terms of quantum cost, because there might exist longer cascades with lower quantum cost value (see the results in Tables III-VI). Nevertheless, the circuits we have found using the above described method are usually better in terms of quantum cost than the best circuits known from the literature.

Our algorithm can be very easily parallelized. Calculations of the optimal circuit length for the remainder functions after applying all possible gates can be performed independently. In practical applications one should also consider adding a time limit, because the calculation take a lot of time for the parameter $a$ larger than 2 and circuits longer than 13 gates. The calculation time also depends on the size of the database and the computer system used for calculations. Our results have been obtained on IBM pSeries p5 550 with two 4-way 1.65GHz POWER5+ CPUs with total of 16GB of memory. We used a database for 8-gate circuits. All the times are given in cpu-seconds.

To present our results in a compact way we have shortened the names of gates:
- NOT(a) is denoted by Na,
- CNOT(a,b) is denoted by Ca-b,
- TOFFOLI(a,b,c) is denoted by Tab-c,
- TOFFOLI(a,b,c,d) is denoted by T4abc-d.

We have adopted a method of calculating quantum cost applied in [12]. Possibility of grouping T and C gates into Peres gates or inverted Peres gates are detected and marked in the following way: <Tab-c Ca-b> or <Ca-b Tab-c>.

Our results are partitioned into four tables for four sets of benchmarks to which they are compared. For example, Table III compares our synthesis results with the ones reported for the five cost-equivalent classes of maximal size (4g15b functions) posted in [12] and obtained with the fast optimization tool described in [2]. Quantum cost of our circuits (in bold in Table III) is on average 33.1% off from the earlier reported, with 49.2% for 4b15g_2 (see Table VII). Our circuits were found by generating all 15-gate gate count minimal circuits thus present exact minimal values for circuits with 15 gates. The interval width of quantum cost values varies from 36 (for 4b15g_1) up to 56 (for 4b15g_5). The number of generated circuits varies from 79277 (for 4b15g_5) up to 340066 (for 4b15g_1). The circuits collected in the Table III are also showed graphically in Fig. 4. Some of the other circuits found by us are shown in Fig. 5 and Fig. 6 together with the source circuits.

Figure 4. Our circuits with minimal quantum cost for 4b15g functions

Figure 5. Circuits for the function mini_alu: a) best known from [26], b) best result of our optimization (Table V)

Figure 6. Circuits for the function dmasks: a) example from [15], b) best result of our optimization (Table V)
### TABLE III. Quantum Cost Optimal Implementations with Specified Gate Count for 4x4 Benchmark Functions [12]

| GC  | QC | #circuits | time[s] | example circuit for QC |
|-----|----|-----------|---------|------------------------|
| 15  | 39-75 | 340066  | 366283 | Ca-d Ca-c Cd-b Ta-c-b Ca-a Cc-c <Tc-b-b Ce-d-b Cb-c > Tab-b Ca-b-b Cd-b Na Nb-a T4ab-c-d |
| 4b15g_2  | 1,19,0,9,10,2,11,3,15,5,12,7,14,16,0 | GC=15, QC=47 in [12] |
| 31-83 | 207520 | 395002 | Nb Cc-b Cd-b Ta-c-b Cd-b Ca-b-b Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> |
| 4b15g_3  | 3,1,7,13,11,8,15,2,5,10,6,9,14,12,4 | GC=15, QC=53 in [12] |
| 33-83 | 119857 | 369087 | Nb Cc-b Cd-b Ta-b-c Cb-a-a Ta-c-b Cd-b Ca-b-b Tb-c-b Cc-b Cc-d-b |

### TABLE IV. Quantum Cost Optimal Implementations with Specified Gate Count for 4x4 Benchmark Functions from [3, 2, 12]

| GC  | QC | #circuits | time[s] | example circuit for QC |
|-----|----|-----------|---------|------------------------|
| 4_49 | 12  | 30-72 | 374 | Cc-a Na Ca-d Tb-d-b <Cc-a Ta-b>b Tb-c-a Tb-d-b Cd-b Cd-c Na |
| 13  | 28-30 | 192072 | 389520 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Na Tb-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> |
| 14  | 28-67 | 1801004 | 3289578 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Tb-c-b <Cc-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> Cd-b Cb-c |
| decode42 | 12,4,8,0,3,5,6,7,9,10,11,12,13,14,15 | GC=10, QC=30 in [3] |
| 10  | 28-30 | 192072 | 389520 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Nb-a Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> |
| 11  | 29-51 | 264716 | 350715 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Na Tb-c-b <Cc-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> Cd-b Cb-c |
| mperk | 3,5,1,18,8,9,7,2,6,14,13,10,12,15 | GC=15, QC=43 in [12] |
| nth_prime4_inc | 0,2,12,8,5,9,11,1,6,10,13,3,14,7,15 | GC=11, QC=21 in [3] |
| 12  | 21-39 | 264716 | 350715 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Nb-a Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> |
| 11  | 20-60 | 380782 | 380782 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Nb-a Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> Cd-b Cb-c |
| 12  | 20-60 | 380782 | 380782 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Nb-a Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> Cd-b Cb-c |
| 13  | 19-67 | 1801004 | 380782 | Nb Cc-b Cd-b Ta-c-b Ca-b-a Nb-a Ta-c-b <Ca-a Ta-bb-b Cc-d-b Cc-c-d Ta-b-b> Cd-b Cb-c |
| imark | 4,5,12,3,5,6,8,7,0,10,13,9,2,4,14,11 | GC=12, QC=32 in [3] |
| 7  | 19-19 | 8 | 1 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a |
| 8  | 18-49 | 303 | 6 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a |
| 9  | 17-63 | 6437 | 108 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a |
| mperk | 3,11,2,10,7,6,5,4,3,2,1,0 | GC=9, QC=15 in [3] |
| 9  | 13-17 | 76 | 2 | Ne <Cc-b Tb-d-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a |
| 10  | 14-38 | 3411 | 49 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a |
| oc5 | 6,0,12,15,7,1,5,2,4,10,3,11,8,14,9 | GC=11, QC=39 in [3] |
| 11  | 39-39 | 48 | 11 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| 12  | 34-44 | 8844 | 235 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| 13  | 35-85 | 471283 | 9576 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| oc6 | 9,0,2,15,11,6,7,8,14,3,4,15,5,1,12,10 | GC=12, QC=42 in [3] |
| 12  | 38-68 | 4475 | 173 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| 13  | 37-89 | 328105 | 8341 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| oc7 | 6,15,9,5,13,2,7,12,0,1,10,11,14,8,4 | GC=11, QC=41 in [3] |
| 12  | 36-49 | 1423 | 56 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| 13  | 35-69 | 51771 | 2166 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| oc8 | 11,3,9,2,7,13,15,4,8,1,10,12,6,5 | GC=12, QC=48 in [3] |
| 10  | 42-48 | 473 | 64 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| 11  | 35-69 | 51771 | 2166 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Cd-a Cd-c Ta-b-c Nb-a Ta-c-b |
| nth_prime4_inc | 0,2,3,7,11,13,14,6,8,9,10,12,14,15 | GC=15, QC=51 in [12] |
| 12  | 53-55 | 12 | 10 | Ta-c-b Tb-d-b <Cc-a Tb-d-b> Tb-d-b Cd-a Cd-b Ta-b-a Tb-d-b |
| 13  | 32-46 | 2288 | 591 | Kb-d <Cc-b Tb-d-b Ta-d-b Ta-c-b Nb-a Ta-bb-b Ta-c-b |
| 14  | 31-93 | 187945 | 7282 | Kb-d <Cc-b Tb-d-b Ta-d-b Ta-c-b Nb-a Ta-bb-b Ta-c-b |
| 15  | 26-114 | 1105632 | 292578 | Kb-d <Cc-b Tb-d-b Ta-d-b Ta-c-b Nb-a Ta-bb-b Ta-c-b |
| primes4 | 2,3,5,7,11,13,0,1,4,6,8,9,10,12,14,15 | GC=10, QC=42 in [2] |
| 12  | 48-48 | 7 | 2 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Tb-d-b Cd-a Cd-c Nb-a Ta-c-b |
| 13  | 47-55 | 8552 | 147 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Tb-d-b Cd-a Cd-c Nb-a Ta-c-b |
| 12  | 22-92 | 430950 | 6789 | Tb-d-a Tb-c-b <Cc-a Tb-d-b> Tb-d-b Cd-a Cd-c Nb-a Ta-c-b |
| GC    | QC    | #circuits | time[s] | example circuit for QC |
|-------|-------|-----------|---------|------------------------|
| mini_alu | 0 | 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 | 6 | Gr 6, QC=62 in [26] |
| aj-e11 | 0 | 1,2,4,8,0,3,5,6,7,9,10,11,12,13,14,15 | 16 | Gr 10, QC=30 in [26] |
| mod10_171 | 0 | 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 | 17 | Gr 7QC=56 in [10] |
| msae | 0 | 11,3,9,7,2,13,15,14,1,4,0,10,12,6,5 | 24 | Gr=16, QC=72 in [19] |
| gyang | 0 | 2,5,3,15,4,6,7,8,9,10,11,12,14,10 | 25 | Gr=19, QC=103 in [31] |
| dmasl | 0 | 1,1,3,4,5,7,8,9,10,11,12,13,14,15,16,17 | 26 | Gr=16, QC=128 in [15] |
| App2.2 | 0 | 7,14,9,6,10,11,12,2,5,10,12,14,13,3,8 | 27 | Gr=18, QC=102 in [32] |
| App2.11 | 0 | 7,14,9,6,10,11,12,2,5,10,12,14,13,3,8 | 28 | Gr=14, QC=102 in [32] |
**TABLE VI. QUANTUM COST OPTIMAL IMPLEMENTATIONS WITH SPECIFIED GATE COUNT FOR 4*4 FUNCTIONS FROM [30]**

| GC | QC | #circuits | time[s] | example circuit for QC<sub>min</sub> |
|----|----|-----------|---------|--------------------------------------|
| f1 | [0,1,2,3,15,10,11,13,9,12,5,4,14,8,6,7] | 27-33 | 12 | GC=13, QC=24 in [30] |
| 9  | 10  | 680  | 16 | Cc-d <Td-b Cd-b> Tbd-c Cc-d Tac-d Tad-c | Cc-a Tac-d Cb-a Tad-b |
|   |     |      | 939| Cc-b <Cd-b Tac-b> Cc-d Tbc-a Cc-b Tbd-a |
| 11 | 22  | 131930| 24288|Cb-c <Tcd-b Cd-b> Cc-d Tac-d Cc-a Cc-b |
| g1 | [0,1,2,3,4,5,6,7,8,9,13,12,14,15,11,10] | 15-15| 2  | GC=7, QC=11 in [30] |
| 3  | 4  | 10 | 0 | Cc-d <Td-b Cd-b> Tbd-c Cc-d Tac-d Tad-c |
| 5  | 13  | 449 | 10 | Cc-a <Cd-b Tac-d> Cc-d Cc-a Cb-a |
| 7  | 62  | 5242| 79 |Cb-c <Cc-a Tac-d> Cc-d |
| 8  | 72  | 93966| 1396|Cc-a <Cd-b Tac-d> Cc-d |
| g2 | [0,1,2,3,4,5,6,7,8,10,12,14,15,11,9] | 20-20| 2  | GC=8, QC=12 in [30] |
| 4  | 5  | 18 | 0 | Tb-c <Cd-b Tbd-c> Tbd-a |
| 6  | 46  | 449 | 10 | Cc-a <Cd-b Tac-d> Cc-d Cc-a Cb-a |
| 7  | 67  | 5242| 79 |Cb-c <Cc-a Tac-d> Cc-d |
| 8  | 72  | 93966| 1396|Cc-a <Cd-b Tac-d> Cc-d |
| g3 | [0,1,2,3,4,5,6,7,11,9,15,13,10,8,14,12] | 15-31| 26 | GC=8, QC=12 in [30] |
| 3  | 4  | 358 | 10 | Cc-b <Cd-b Tac-d> Cc-b Cb-a |
| 5  | 52 | 13944| 183 |Cc-c <Cd-b Tac-d> Cc-d |
| 7  | 73  | 31370| 4680|Cc-c <Cd-b Tac-d> Cc-d Cb-a |
| g4 | [0,1,2,3,15,7,6,4,11,14,12,10,8] | 17-37| 149| GC=12, QC=18 in [30] |
| 9  | 17  | 19 | 5 | Cb-a <Cd-b Tac-d> Cc-b |
| 10 | 15-52| 13086| 180 |Cb-a <Cd-d Tac-b> Cc-d Cb-a |
| 11 | 17-79| 485674| 7034 |Cb-a <Cd-d Tac-b> Cc-d |
| g5 | [0,1,2,3,15,7,6,4,11,14,12,10,8] | 20-32| 54 | GC=11, QC=19 in [30] |
| 8  | 9  | 56 | 6 | Cb-a <Cd-b Tac-d> Cc-b |
| 10 | 19-53| 1863| 24 |Cc-b <Cd-b Tac-d> Cc-b |
| 11 | 19-79| 48579| 931 |Cb-a <Cd-b Tac-d> Cc-b |
| g6 | [0,1,2,3,14,5,13,7,6,4,9,10,8,11,12] | 32-32| 8 | GC=11, QC=19 in [30] |
| 7  | 9  | 72 | 1 | Cb-a <Cd-b Tac-d> Cc-b |
| 9  | 19-53| 1863| 24 |Cc-b <Cd-b Tac-d> Cc-b |
| 10 | 19-79| 48579| 931 |Cb-a <Cd-b Tac-d> Cc-b |
| g7 | [0,1,2,3,15,6,14,12,10,8,11,7,5] | 21-31| 26 | GC=11, QC=19 in [30] |
| 7  | 8  | 18 | 1 | Cb-a <Cd-b Tac-d> Cc-b |
| 9  | 19-53| 1863| 24 |Cc-b <Cd-b Tac-d> Cc-b |
| 10 | 19-79| 48579| 931 |Cb-a <Cd-b Tac-d> Cc-b |
| g8 | [0,1,2,3,14,5,13,7,6,4,9,10,8,11,12] | 20-32| 54 | GC=13, QC=23 in [30] |
| 9  | 31-51| 87 | 5 | Cb-a <Cd-b Tac-d> Cc-b |
| 10 | 24-70| 5634| 132 |Cb-a <Cd-b Tac-d> Cc-b |
| 11 | 21-91| 222473| 3460 |Cb-a <Cd-b Tac-d> Cc-b |
| g9 | [0,1,2,3,14,5,13,7,6,4,9,10,8,11,12] | 20-32| 54 | GC=13, QC=23 in [30] |
| 9  | 31-51| 87 | 5 | Cb-a <Cd-b Tac-d> Cc-b |
| 10 | 24-70| 5634| 132 |Cb-a <Cd-b Tac-d> Cc-b |
| 11 | 21-91| 222473| 3460 |Cb-a <Cd-b Tac-d> Cc-b |
Table IV compares our synthesis results to the ones collected in [2, 3] for a set of known benchmarks (we have omitted simple benchmarks 4-bit-7-8, rd32 and shift4 as exact minimal quantum cost circuits for them have already been found). The comparison is favorable for our circuits, (e. g. our circuits for the functions nth_prime4_inc and primes4 are approximately 50% off from those reported in [3]). Again, the lowest values of quantum cost for circuits found by us are shown in bold in the column QC. Note that usually initial increasing of circuit size leads to lowering quantum cost but further increasing of GC causes a rise of QC (see results for benchmarks imark and oc5).

Table V compares our synthesis results to 4*4 designs we could find in RevLib [26] and in recent publications. For some of these functions (e. g. mini-alu and gyang) improvements (shown in bold) of over 60% are reported.

Table VI compares the results of our synthesis algorithm with the results in [30] where 4*4 quantum cost optimal circuits built from Peres gates were searched. For a fair comparison we calculated GC in circuits from [30] by considering a Peres gate as two gates like in our designs. For 12 out of 14 functions considered in [30] our designs have smaller QC and for the other two functions we have obtained the same minimal values of QC.

Table VII shows the collective data comparing our circuits to previously found ones realized by using gates exclusively from NCT library. These data were taken from Tables III-V. For some benchmarks and designs taken from the literature we have obtained savings in QC of over 74% comparing with previously known circuits. Average improvement is equal to 44.7 %. In Fig. 5 and Fig. 6 circuits for our best QC improvement (mini_alu [26] and dmasl [15]) are shown together with the source circuits found in the literature. For some of the benchmarks considered in the paper better results could probably be generated, however we had to stop calculations due to time limitations in our experiments.

Studying Tables III-VI one can see that increasing the gate count up to a certain value causes the reduction of quantum cost of best circuits. Usually this quantum cost reduction is achieved by replacing pairs of multiple-control Toffoli gates by cascades built from smaller gates. Further reduction of QC is achieved by grouping Toffoli and CNOT gates into Peres gates. The presented method does not guarantee finding quantum cost minimal circuits because the length (GC) of such circuits is not known. However, for an arbitrary selected circuit length it allows for generating 4*4 circuits with the least quantum cost if the computation time is not too large. Due to this property we were able to find circuits with QC smaller than results obtained by the other methods presented in the literature.
VIII. CONCLUSIONS AND FUTURE WORK

A common practice in synthesis of reduced quantum cost reversible circuits (e.g. see [5], page 704) is first find a gate count optimal circuit and then map the resulting circuit into quantum gates. However, as we showed in the paper, this approach does not lead to minimal quantum cost circuits.

It requires considering circuits having greater number of gates than the minimal one to be able to find exact minimal quantum cost circuits. Research presented in the paper can be extended in different directions. One possibility is to consider other extensions to quantum cost model, like for example inter-gate optimizations [16, 17, 20, 21]. The other possibility is to apply the proposed method for another criteria of optimality. Namely, the large number of the reversible circuits that realize the same reversible function suggest that the circuits might also differ significantly under other criteria, e.g. depth of a circuit, latency, testability properties or new cost functions (like LNN cost or transistor cost [28]).

Due to limitations in speed and memory space of current computer systems the proposed method cannot be used directly for circuits with more than 4 inputs/outputs. However, it might be applied to optimize some parts of \( n \times n \) circuits, \( n > 4 \), having relatively long 4*4 subcircuits.

ACKNOWLEDGMENT

This work was supported by the Polish Ministry of Science and Higher Education under Grant 4180/B/T02/2010/38.

REFERENCES

[1] J. Donald and N. K. Jha, “Reversible Logic Synthesis with Fredkin and Peres Gates,” ACM Journal on Emerging Technologies in Computing Systems, vol. 4, no. 1, 2008, pp. 1-19.
[2] O. Golubitsky, S. M. Falconer, and D. Maslov, “Synthesis of the Optimal 4-bit Reversible Circuits,” Proc. DAC, USA, 2010, pp. 653-656.
[3] O. Golubitsky and D. Maslov, “A Study of Optimal 4-bit Reversible Toffoli Circuits and Their Synthesis,” arXiv: 1103.2686v1, March 2011, 16 pages.
[4] D. Grosse, X. Chen, G. W. Dueck, and R. Drechsler, “Exact SAT-based Toffoli Network Synthesis,” Proc. 17th Great Lakes Symposium on VLSI, Italy, 2007, pp. 96-101.
[5] D. Grosse, R. Wille, G. W. Dueck, and R. Drechsler, “Exact Multiple Control Toffoli Network Synthesis with SAT Techniques,” IEEE Trans. on CAD, vol. 28, no. 5, 2009, pp. 703-715.
[6] P. Kerntopf, “Some Remarks on Reversible Logic Synthesis,” Proc. 8th Int. Reed-Muller Workshop, Oslo, Norway, May 2007, pp. 51-57.
[7] P. Kerntopf and M. Szyprowski, “On Some Properties of Reversible Boolean Functions,” Proc. 8th Int. Workshop on Boolean Problems, Freiberg, Germany, September 2008, pp. 25-32.
[8] P. Kerntopf and M. Szyprowski, “Properties of Hard Reversible Boolean Functions,” Proc. of the 2nd Workshop on Reversible Computation, Bremen, Germany, July 2010, pp. 41-48.
[9] P. Kerntopf and M. Szyprowski, “An Approach to Constructing Hard Reversible Functions,” Proc. 9th Int. Workshop on Boolean Problems, Freiberg, Germany, September 2010, pp. 57-64.
[10] Z. Li, H. Chen, B. Xu, X. Song, and X. Hua, “Fast Algorithm for 4-Qubit Reversible Logic Circuits Synthesis,” Proc. IEEE Congress on Evolutionary Computation, 2008, pp. 2202-2207.
[11] Z. Li, H. Chen, B. Xu, X. Song, and X. Hua, “Effective Hash-based Algorithm for Reversible Logic Circuits Synthesis with Minimum Cost,” 4th IEEE Int. Conf. on Natural Computation, 2008, pp. 623-627.
[12] D. Maslov, “Reversible Logic Synthesis Benchmarks Page,” http://www.cs.uvic.ca/~dmaslov.
[13] D. Maslov and G.W. Dueck, “Improved Quantum Cost for n-bit Toffoli Gates,” IEEE Electronic Letters, vol. 39, 2003, no 25, pp. 1790-1791.
[14] D. Maslov, G.W. Dueck, and D.M. Miller, “Techniques for the Synthesis of Reversible Toffoli Networks,” ACM Trans. on Design Automation of Electronic Systems, vol. 12, no. 4, 2007, pp. 1-28.
[15] D. Maslov and M. Saeedi, “Reversible circuit optimization via leaving the Boolean domain,” IEEE Trans. on CAD, in press, 2011, available at arXiv: 1103.0215v1, March 2011, 14 pages.
[16] D. M. Miller, “Lower Cost Quantum Gate Realizations of Multiple-Control Toffoli Gates,” Proc. IEEE Pacific Rim Conference, 2009, pp. 308-313.
[17] D. M. Miller and Z. Sasianan, “Lowering the Quantum Gate Cost of Reversible Circuits,” Proc. 53rd IEEE International Midwest Symposium on Circuits and Systems, 2010, pp. 260—263.
[18] A.K. Prasad, V.V. Shende, K.N. Patel, L.M. Markov, and J.P. Hayes, “Data Structures and Algorithms for Simplifying Reversible Circuits,” ACM Journal on Emerging Technologies in Computing Systems, vol. 2, no. 4, 2006, pp. 277-293.
[19] M. Saeedi, M. Saheb Zamani, and M. Sedighi, “Algebraic Characterization of CNOT-Based Quantum Circuits with Its Applications on Logic Synthesis,” Proc. 10th Euromicro Conference on Digital System Design, 2007, pp. 339-346.
[20] Z. Sasianan and D. M. Miller, “Mapping a Multiple-Control Toffoli Gate Cascade to an Elementary Quantum Gate Circuit,” Proc. 2nd Workshop on Reversible Computation, 2010, pp. 83—90.
[21] N. O. Scott and G. W. Dueck: “Pairwise Decomposition of Toffoli Gates in a Quantum Circuit,” Proc. Great Lakes Symposium on VLSI 2008, pp. 231—235.
[22] M. Szyprowski and P. Kerntopf, “Decompositions of Reversible Logic Circuits,” Pomerany-Automaityka-Kontrola, vol. 55, no. 8, 2009, pp. 609-611.
[23] M. Szyprowski and P. Kerntopf, “Ways to Improve Heuristic Algorithms for Reversible Circuit Synthesis,” Proc. 18th IEEE/ACM Int’l Workshop on Logic and Synthesis, 2009, pp. 202-209.
[24] M. Szyprowski and P. Kerntopf, “Estimating the Quality of Complexity Measures in Heuristics for Reversible Logic Synthesis,” IEEE Congress on Computational Intelligence, Congress on Evolutionary Computation, Barcelona, Spain, July 2010, 8 pages.
[25] D. Wang, H. Chen, and W. Zhu, “Bidirectional Matrix-based Algorithm for 4-qubit Reversible Logic Circuits Synthesis,” Proc. IEEE Congress on Evolutionary Computation, 2010, 5 pages.
[26] R. Wille, D. Grosse, L. Teuber, G. W. Dueck, and R. Drechsler, “RevLib: An Online Resource for Reversible Functions and Reversible Circuits,” http://www.revlib.org.
[27] R. Wille, H. M. Le, G. W. Dueck, and D. Grolle, “Quantified Synthesis of Reversible Logic,” Proc. DATE, 2008, pp. 1015-1020.
[28] R. Wille, M. Saeedi, and R. Drechsler, “Synthesis of Reversible Functions Beyond Gate Count and Quantum Cost,” Proc. 18th Int. Workshop on Logic and Synthesis, 2009, pp. 43-49.
[29] G. Yang, X. Song, W. N. N. Hung, F. Xie, and M. Perkowski, “Group Theory Based Synthesis of Binary Reversible Circuits,” Proc. 3rd Int. Conf. on Theory and Applications of Models of Computation, Lecture Notes in Computer Science, vol. 3959, 2006, pp. 365-374.
[30] G. Yang, X. Song, W. N. N. Hung, and M. Perkowski, “Bidirectional Synthesis of 4-Bit Reversible Circuits,” The Computer Journal, vol.51, no. 2, 2008, pp. 207-215.
[31] G. Yang., F. Xie, W. N. N. Hung, X. Song, and M. Perkowski, “Realization and Synthesis of Reversible Circuits,” Theoretical Computer Science, vol. 412, no. 17, April 2011, pp. 1606-1613.
[32] A. Younes, “Detection and Elimination of Non-trivial Reversible Identities,” arXiv: 1101.4384v1, January 2011, 12 pages.