2012 IEEE 42nd International Symposium on Multiple-Valued Logic

Exact Synthesis of Toffoli Gate Circuits with Negative Control Lines

Robert Wille* Mathias Soeken* Nils Przigoda* Rolf Drechsler*†

*Institute of Computer Science
University of Bremen, 28359 Bremen, Germany

†Cyber-Physical Systems, DFKI GmbH
28359 Bremen, Germany

{rwille,msoeken,przigoda,drechsle}@informatik.uni-bremen.de

***Abstract*—The development of synthesis approaches for reversible circuits is an active research area. Besides heuristic methods, also exact synthesis received significant attention. Here, circuits realizing the desired functions e.g. with a *minimal* number of gates are determined. However, so far exact synthesis considering Toffoli gate circuits with positive control lines only has been considered.**

In this paper, we are extending the scope of exact synthesis by additionally considering negative control lines in the circuits to be synthesized. For this purpose, we propose and evaluate a SAT-based synthesis method. Our experiments show that incorporating negative control lines leads to smaller circuits with respect to the number of gates. Furthermore, in some cases even the run-time of the synthesis can be improved.

I. Introduction

Reversible logic is an intensively studied research area, which has several applications in emerging technologies like quantum computation [1], low-power design [2], [3], and others. In order to obtain a circuit representation that realizes a given function, synthesis is performed. Therefore, several restrictions have to be considered. In particular, fan-out and feed-back are not directly allowed [1]. Consequently, a circuit for reversible logic usually is composed as a cascade of reversible gates. Most frequently applied is thereby the Toffoli gate originally proposed in [4].

Due to these restrictions and the different gate library, conventional synthesis methods often cannot be applied appropriately. Thus, researchers developed new synthesis approaches dedicated to reversible circuits (see e.g. [5], [6], [7], [8], [9], [10]). Most of the existing algorithms employ a heuristic nature, e.g. they realize circuits whose number of gates is not optimal.

In contrast, exact synthesis algorithms determine a *minimal* circuit realization for a given function, i.e. a circuit with a minimal number of gates or quantum costs, respectively. Ensuring minimality often causes a large computation time and, thus, exact approaches are only applicable to relatively small functions. Nevertheless, it is worth to consider exact methods, since

- they allow determining smaller circuits than the currently best known realizations,
- they allow the evaluation of the quality of heuristic approaches, and
- they allow the computation of minimal circuits as basic blocks for larger circuits.

For example, improving the heuristic results by 10 % is significant, if this leads to optimal solutions, but marginal if the generated circuits are still factors away from the optimum. To obtain such conclusions, the optimum must be available.

Motivated by this, researchers proposed different approaches aiming for the synthesis of circuits with optimal quantum costs [11], [12] as well as an optimal number of reversible gates [5], [13], [6]. However, in all these approaches synthesis of reversible circuits composed of Toffoli gates with positive control lines only have been considered.

In this work, we address exact synthesis of Toffoli gate circuits which additionally makes use of negative control lines in Toffoli gates. We propose a SAT-based synthesis method based on the concepts presented in [6]. More precisely, the synthesis problem is formulated as a sequence of decision problems. The decision problems are encoded as instances of *Boolean satisfiability* (SAT). As soon as one of these instances becomes satisfiable, a Toffoli circuit representation for the given function is found. In contrast to [6], we allow Toffoli gates with both, positive and negative control lines.

As shown by the experimental evaluation of the proposed approach, incorporating negative control lines leads to smaller circuits with respect to the number of gates. In fact, using the proposed approach enables the generation of Toffoli circuits which are smaller than the currently best know realizations — even smaller than the ones known as minimal today.

0195-623X/12 $26.00 © 2012 IEEE
DOI 10.1109/ISMVL.2012.71

Furthermore, additionally considering negative control lines sometimes even improves the run-time of the synthesis. This is due to the fact that, although recognizing negative control lines in fact increases the complexity of the search, the additional consideration of negative control lines often leads to smaller circuits resulting in an earlier termination of the algorithm.

The remainder of the paper is structured as follows. The next section provides an overview of the required background, i.e. reversible logic and reversible circuits as well as Boolean satisfiability are briefly reviewed. Section III described the proposed SAT-based synthesis method which generates circuits with the minimal number of gates. Finally, experimental results are provided and discussed in Section IV before the paper concludes with Section V.

II. Background

In order to keep the paper self-contained, the following section briefly reviews the basics on reversible logic and Boolean satisfiability.

A. Reversible Logic and Reversible Circuits

A Boolean function $f : \mathbb{B}^n \rightarrow \mathbb{B}^n$ is *reversible* if it is bijective, i.e. if each input pattern is uniquely mapped to a corresponding output pattern. The *synthesis problem* is defined as the task of determining a reversible circuit for a given function f.

Reversible circuits differ from conventional circuits, since e.g. fanout and feedback are not directly allowed [1]. Usually, they are built as a cascade of reversible gates. Most frequently applied is thereby the Toffoli gate [4].

Definition 1: Let $X = \{x_1, \ldots, x_n\}$ be a set of variables or circuit lines. Then, a reversible circuit is described as a cascade $g_1 \ldots g_d$. A gate $g_i = (C_i, t_i)$, $i \in \{1, \ldots, d\}$, is a tuple of a set $C_i \subset \{x^\varrho \mid x \in X, \varrho \in \{-, +\}\}$ of (positive and negative) *control lines* and a *target line* $t_i \in X$ with $\{t_i\} \cap C_i = \emptyset$. The target line t_i of a Toffoli gate is inverted if and only if all positive (negative) control lines evaluate to one (zero). The values of all remaining lines are passed through the gate unaltered. That is, the Toffoli gate maps $(x_1, \ldots, x_{t_i}, \ldots, x_n)$ to $(x_1, \ldots, \bigwedge_{x \in C_i} \dot{x} \oplus x_{t_i}, \ldots, x_n)$ with $\dot{x} = x$ for any x^+ and $\dot{x} = \overline{x}$ for any x^-.

Example 1: Fig. 1 shows a reversible circuit with three lines and composed of four gates. The target lines are denoted by $\oplus$, while a $\bullet$ represents a positive control line and a $\circ$ represents a negative control line. For example, assigning the input pattern 001 to the circuit results in the output pattern 101. Due to the reversibility, this computation can be performed in both directions.

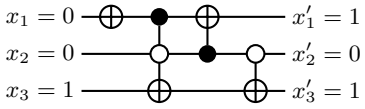


Fig. 1. Reversible circuit

B. Boolean Satisfiability

The *Boolean Satisfiability* (SAT) problem is defined as follows:

Definition 2: Let h be a Boolean function in *Conjunctive Normal Form* (CNF), i.e. a product-of-sum representation. Then, the SAT problem is to determine an assignment for the variables of h such that h evaluates to 1 or to prove that no such assignment exists.

The CNF consists of a conjunction of clauses. A clause is a disjunction of literals and each literal is a propositional variable or its negation.

Example 2: Let $h = (x_1 + x_2 + \overline{x}_3)(\overline{x}_1 + x_3)(\overline{x}_2 + x_3)$. Then, $x_1 = 1, x_2 = 1$ and $x_3 = 1$ is a satisfying assignment for h. The values of x_1 and x_2 ensure that the first clause becomes satisfied while x_3 ensures this for the remaining two clauses.

Once it is proven that no solution exist, an instance is called *unsatisfiable* (UNSAT), otherwise *satisfiable* (SAT).

SAT is one of the central $\mathcal{NP}$-complete problems. In fact, it was the first known $\mathcal{NP}$-complete problem that was proven by Cook in 1971 [14]. But in the past, efficient solving algorithms (so called *SAT solvers*) have been proposed (see e.g. [15]). Instead of simply traversing the complete space of assignments, intelligent decision heuristics, *conflict based learning*, and sophisticated engineering of the implication algorithm by *Boolean Constraint Propagation* (BCP) lead to an effective search procedure. Due to these efficient algorithms, problem instances consisting of hundreds of thousands of variables, millions of clauses, and tens of millions of literals can be handled.

III. The Synthesis Approach

In this work, exact synthesis of Toffoli gate circuits with negative control lines is considered and evaluated. Therefore, a synthesis approach is proposed which applies the general concepts introduced in [6]. This section describes the proposed synthesis approach in detail.

A. General Concepts

In [6], the exact synthesis problem is formulated as a sequence of decision problems. Given the reversible function $f : \mathbb{B}^n \rightarrow \mathbb{B}^n$, it is checked whether f can be

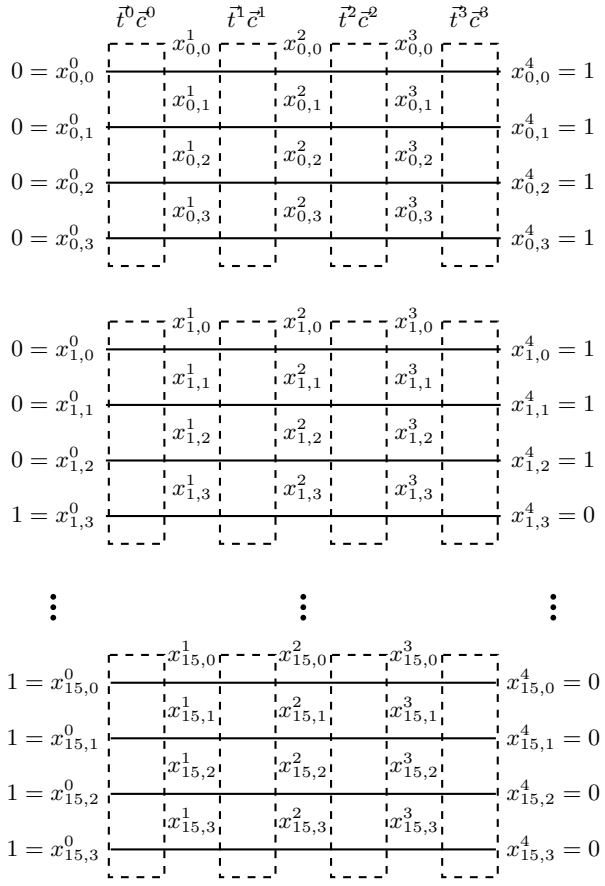


Fig. 2. Exact SAT formulation for $n = 4$ and $d = 4$.

synthesized using $d = 1$ Toffoli gates. If this fails, then d is increased until a realization has been determined. Since d is iteratively increased starting with $d = 1$, minimality is ensured. The respective checks are performed by

- encoding the synthesis problem as an instance of Boolean satisfiability and
- using a SAT solver to solve this instance.

For this purpose, SAT instances are created that become satisfiable if and only if a circuit with d gates representing the function exists. To this end, Boolean variables and constraints as introduced in the following are applied.

Definition 3: Let $f : \mathbb{B}^n \to \mathbb{B}^n$ be a reversible function to be synthesized. Then, variables $x^k_{i,0}, x^k_{i,1}, \ldots, x^k_{i,n-1}$ are applied to represent the input- (for $k = 0$), the output- (for $k = d$), and the auxiliary values (for $1 \leq k \leq d-1$) of the circuit to be synthesized for each truth table line i of f. Therefore, the left-hand side of the truth table corresponds to the variables $x^0_{i,0}, x^0_{i,1}, \ldots, x^0_{i,n-1}$, while the right-hand side corresponds to the variables $x^d_{i,0}, x^d_{i,1}, \ldots, x^d_{i,n-1}$.

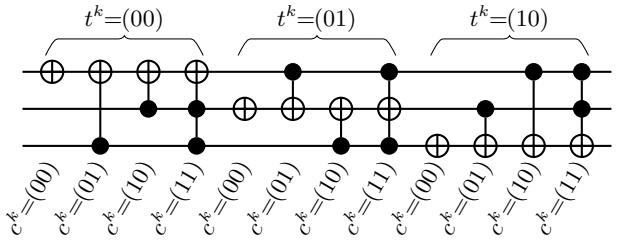


Fig. 3. Representation of Toffoli gates by assignments to $\vec{t}^k$ and $\vec{c}^k$

Fig. 2 shows the respective variables for a function f to be synthesized with $n = 4$ variables. The first row of Fig. 2 represents the variables for the first truth table line of f, the second row the ones for the second truth table line of f, and so on.

Furthermore, variables representing the type of a Toffoli gate are introduced:

Definition 4: Let $f : \mathbb{B}^n \to \mathbb{B}^n$ be a reversible function to be synthesized. Then, $t^k_{\lceil \log_2 n \rceil}, t^k_{\lceil \log_2 n \rceil - 1}, \ldots, t^k_1$ and $c^k_1, c^k_2, \ldots, c^k_{n-1}$ with $0 \leq k < d$ are introduced, whose assignments represent the type of the Toffoli gate at depth k (for brevity denoted by $\vec{t}^k$ and $\vec{c}^k$ in the following). The variable $\vec{t}^k$ is used as a binary encoding of a natural number $t^k \in \{0, \ldots, n-1\}$ which defines the chosen target line. In contrast, $\vec{c}^k$ denotes the control lines. More precisely, assigning $c^k_l = 1$ $(1 \leq l \leq n-1)$ means that line $(t^k + l) \mod n$ becomes a control line of the Toffoli gate at depth k.

Fig. 3 gives some examples for assignments to $\vec{t}^k$ and $\vec{c}^k$ with their respective Toffoli gate representation.

Using these variables, constraints are introduced assigning the input and output of the truth table to their respective $x^0_{i,j}$ and $x^d_{i,j}$ variables. Furthermore, for each gate to be synthesized at depth k, functional constraints are added so that (depending on the assignment to $\vec{t}^k$ and $\vec{c}^k$ as well as to the input $x^k_{i,j}$) the respective gate output $x^{k+1}_{i,j}$ is computed. As an example, consider $\vec{t}^k = (01)$ and $\vec{c}^k = (001)$, i.e. with $c^k_3 = 1$. This assignment states that the Toffoli gate at depth k has line $t^k = [01]_2 = 1$ as target line and line $(t^k + l) \mod n = (1 + 3) \mod 4 = 0$ as single control line. To cover this case, constraints

$$\vec{t}^k = (01) \wedge \vec{c}^k = (001) \Rightarrow \left[\begin{array}{lrcl} & x^{k+1}_{i,0} & = & x^k_{i,0} \\ \wedge & x^{k+1}_{i,1} & = & x^k_{i,1} \oplus x^k_{i,0} \\ \wedge & x^{k+1}_{i,2} & = & x^k_{i,2} \\ \wedge & x^{k+1}_{i,3} & = & x^k_{i,3} \end{array}\right. \tag{1}$$

are added for each truth table line i. In other words, the values of lines 0, 2, and 3 are passed through,

while the output value of line 1 becomes inverted, if line 0 is assigned 1. Similar constraints are added for all remaining cases.

As a result, a functional description has been constructed which is satisfiable, if there is a valid assignment to $\vec{t}^k$ and $\vec{c}^k$ such that for all truth table lines the desired input-output mapping is achieved. Then, the precise Toffoli gates are obtained by the assignments to $\vec{t}^k$ and $\vec{c}^k$ as depicted in Fig. 3. If there is no such assignment (i.e. the instance is unsatisfiable), then it has been proven, that no circuit representing the function with d gates exists.

B. Additional Consideration of Negative Control Lines

In order to additionally support negative control lines, the SAT formulation proposed in [6] and reviewed above has been extended. More precisely, the variables introduced in Definition 4 to represent the type of a Toffoli gate are extended as follows:

Definition 5: Let $f : \mathbb{B}^n \rightarrow \mathbb{B}^n$ be a reversible function to be synthesized. Then, the type of the a Toffoli gate still is represented by variables $\vec{t}^k$ and $\vec{c}^k$. However, in order to additionally enable the representation of negative control lines, the size of $\vec{c}^k$ is doubled, i.e. $\vec{c}^k = c_1^k, c_2^k, \ldots, c_{n-1}^k, c_n^k, \ldots, c_{2(n-1)}^k$. The first bits $c_1^k, c_2^k, \ldots, c_{n-1}^k$ still denote whether a line $(t^k + l) \mod n$ becomes a control line of the Toffoli gate at depth k, while the latter bits $c_n^k, c_{n+1}^k, \ldots, c_{2(n-1)}^k$ determine the polarity of the control line. This leads to the following encodings:

- $c_l^k = 0,\ c_{l+(n-1)}^k = 0$: no control line
- $c_l^k = 1,\ c_{l+(n-1)}^k = 0$: negative control line
- $c_l^k = 1,\ c_{l+(n-1)}^k = 1$: positive control line

The redundant encoding $c_l^k = 0,\ c_{l+(n-1)}^k = 1$ leads immediately to a conflict in the solver, which cuts a large portion of the search tree.

Example 3: Fig. 4 shows the adjustment of the c^k variables in comparison to Fig. 3. The upper part show the respective assignments for positive control lines, while the bottom part shows the respective assignments for negative control lines. Analogously, gates with mixed positive and negative control lines are encoded in a similar fashion.

Using these adjusted variables, the functional constraints are extended so that also negative control lines are considered. That is, Eq. (1) is adjusted as

$$\vec{t}^k = (01) \wedge \vec{c}^k = (001001) \Rightarrow \left[\begin{array}{lrcl} & x_{i,0}^{k+1} & = & x_{i,0}^k \\ \wedge & x_{i,1}^{k+1} & = & x_{i,1}^k \oplus x_{i,0}^k \\ \wedge & x_{i,2}^{k+1} & = & x_{i,2}^k \\ \wedge & x_{i,3}^{k+1} & = & x_{i,3}^k \end{array}\right.$$

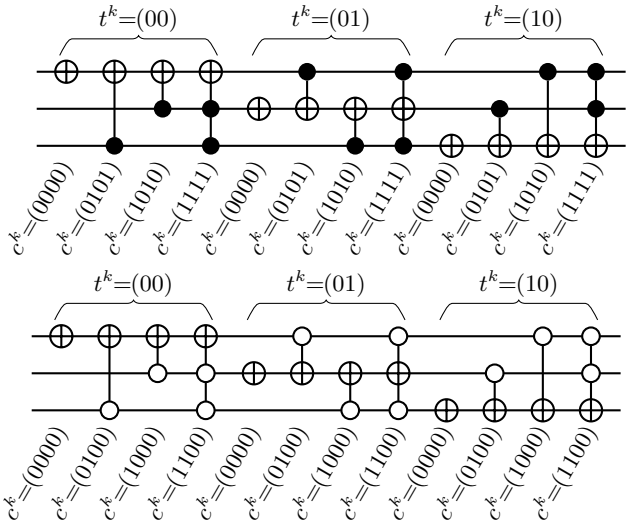
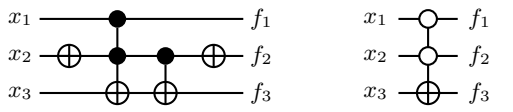


Fig. 4. Representation of Toffoli gates with negative controls by assignments to $\vec{t}^k$ and $\vec{c}^k$



(a) Only positive control lines

(b) Positive and negative control lines

Fig. 5. Different synthesis results when considering negative control lines

In case of a negative control line, the encoding is written as

$$\vec{t}^k = (01) \wedge \vec{c}^k = (001000) \Rightarrow \left[\begin{array}{lrcl} & x_{i,0}^{k+1} & = & x_{i,0}^k \\ \wedge & x_{i,1}^{k+1} & = & x_{i,1}^k \oplus \overline{x_{i,0}^k} \\ \wedge & x_{i,2}^{k+1} & = & x_{i,2}^k \\ \wedge & x_{i,3}^{k+1} & = & x_{i,3}^k \end{array}\right.$$

Example 4: Fig. 5 shows minimal circuits representing the functionality of a Toffoli gate with two negative control lines. If only positive control lines are considered during synthesis, no circuit with less than four gates results (see e.g. Fig. 5(a)). If in contrast negative control lines are support, obviously a single gate is sufficient to realize this functionality (see Fig. 5(b)).

IV. Experimental Results

In this section, experimental results obtained by exact synthesis with positive and negative control lines are presented and discussed. To this end, the approach described in Section III has been implemented in C++

TABLE I
EXPERIMENTAL RESULTS FOR ALL FUNCTIONS WITH $n = 3$

Gates	Circuits
0	1
1	12
2	102
3	625
4	2780
5	8921
6	17049
7	10253
8	577
Average	≈5.87

(a) Positive controls only [6]

Gates	Circuits
0	1
1	27
2	369
3	2925
4	13282
5	20480
6	3236
Average	≈4.58

(b) Proposed approach

on top of RevKit [16]. *SWORD* [17] has been utilized as underlying SAT solver. All experiments have been conducted using an Intel Pentium with 3.6GHz and 2GB of main memory. A timeout of 2000 CPU seconds has been applied.

In a first evaluation, all $2^3! = 40320$ possible reversible functions with $n = 3$ inputs/outputs have been considered. Table II(a) shows the results obtained by the original approach proposed in [6] (not considering negative control lines), while Table II(b) shows the results obtained by the proposed approach (additionally considering negative control lines). In both tables, the number of functions for each gate count and the average number of gates required are listed. Each circuit has been synthesized within less than one second. As can be seen, additionally considering negative control lines enables more compact circuit implementations with respect to the gate count.

In a second evaluation, functions from RevLib [18] have been synthesized. Table II summarizes these results. Column *Benchmark* and Column n denote the name (including the RevLib identifier) and the number of circuit lines for each benchmark, respectively. Afterwards, the number d of gates, the quantum costs QC[1], as well as the required run-time t are reported for circuits obtained by the original approach as proposed in [6] (not considering negative control lines) and the the proposed approach (additionally considering negative control lines). Finally, the differences with respect to gates, quantum costs, and run-time are provided in the rightmost columns.

Additionally considering negative control lines in exact synthesis enables the determination of smaller reversible circuits. In fact, for the majority of the benchmarks from Table II, realizations with less gates can be obtained. Considering that the circuits obtained in [6] represent the best realizations obtained so far with respect to the number of gates, this is a significant result.

Furthermore, in many cases the proposed approach is able to realize the respective circuit in significantly less run-time. Improvements of several orders of magnitude are reported. In case of *aj-e11_complete_74*, the original approach even is not able to finish within the timeout of 2000 CPU seconds, while the proposed method generates the respective circuit in less than a second. At a first glance, this seems contradictory considering that recognizing negative control lines in fact increases the complexity of the search. However, this effect can be explained by the fact that the additional consideration of negative control lines often leads to smaller circuits which allows an earlier termination of the algorithm.

As a drawback, the improvement with respect to the number of gates does not reflect to the respective quantum costs. Only in some few cases (i.e. *4gt4-v0_incomplete_29*, *neg_toffoli*, and *one-two-three-v2_incomplete_45*) the costs can slightly be reduced. For the majority of benchmarks, the quantum costs remain the same or even increase. For circuits where no improvement of gate count is achieved, one can fallback to the circuit realization consisting of positive control lines only. Furthermore, determining better mappings of Toffoli circuits including negative control lines is subject to ongoing research (see e.g. [21], [22]). Thus, we expect better mappings and accordingly better quantum costs here.

V. CONCLUSIONS

In this work, we considered exact synthesis of reversible circuits composed of Toffoli gates with negative control lines. For this purpose, a SAT-based synthesis approach has been proposed and evaluated. Experiments provided interesting results. Additionally considering negative control lines leads to circuits smaller than best realizations obtained so far with respect to the number of gates. Furthermore, the run-time can significantly be decreased in many cases. This sometimes even enabled the determination of exact results, where previously introduced methods aborted due to time limitations.

ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG) (DR 287/20-1).

[1]In order to calculate the quantum costs, the metric provided in RevKit (using the calculations as introduced in [19] and optimized in [20]) are applied.

TABLE II
EXPERIMENTAL RESULTS FOR A SELECTION OF REVLIB FUNCTIONS

		Positive Controls Only [6]			Proposed Approach					
Benchmark	n	d	QC	t	d	QC	t	Δd	Δ QC	Δt
4gt11_incomplete_35	5	3	7	0.04	3	11	0.60	0	4	0.56
4gt12-v0_incomplete_37	5	5	32	7.39	4	40	76.28	-1	8	68.89
4gt13_incomplete_39	5	3	16	0.04	3	16	0.78	0	0	0.74
4gt4-v0_incomplete_29	5	6	46	47.64	4	44	1.30	-2	-2	-46.34
4gt4-v0_incomplete_33	5	5	28	8.84	4	41	1.60	-1	13	-7.24
4gt5_incomplete_32	5	4	30	0.15	4	44	3.58	0	14	3.43
4mod7-v0_incomplete_41	5	6	40	10.29	6	40	270.83	0	0	260.54
aj-e11_complete_74	4	–	–	*TO*	6	45	0.68	–	–	>2000.00
graycode6_complete_19	6	5	5	1082.90	5	5	0.16	0	0	-1082.74
ham3_complete_47	3	5	9	0.03	5	9	0.01	0	0	-0.02
miller_complete_5	3	5	17	892.79	5	23	0.01	0	6	-892.78
mod10_incomplete_78	4	6	31	0.41	4	36	<0.01	-2	5	-0.41
mod5mils_complete_26	4	5	13	0.23	3	13	0.01	-2	0	-0.22
neg_toffoli	5	4	8	<0.01	1	7	<0.01	-3	-1	0.00
one-two-three-v0_incomplete_43	3	8	41	127.45	7	56	341.29	-1	15	213.84
one-two-three-v1_incomplete_44	5	8	24	161.56	7	29	381.01	-1	5	219.45
one-two-three-v2_incomplete_45	5	8	37	174.43	7	31	305.57	-1	-6	131.14
one-two-three-v3_incomplete_46	5	8	28	293.68	7	62	310.03	-1	34	16.35
toffoli	5	1	5	<0.01	1	5	<0.01	0	0	0.00

REFERENCES

[1] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.

[2] R. Wille, R. Drechsler, C. Oswald, and A. Garcia-Ortiz, "Automatic design of low-power encoders using reversible circuit synthesis," in *Design, Automation and Test in Europe*, 2012.

[3] B. Desoete and A. D. Vos, "A reversible carry-look-ahead adder using control gates," *INTEGRATION, the VLSI Jour.*, vol. 33, no. 1-2, pp. 89–104, 2002.

[4] T. Toffoli, "Reversible computing," in *Automata, Languages and Programming*, W. de Bakker and J. van Leeuwen, Eds. Springer, 1980, p. 632, technical Memo MIT/LCS/TM-151, MIT Lab. for Comput. Sci.

[5] V. V. Shende, A. K. Prasad, I. L. Markov, and J. P. Hayes, "Synthesis of reversible logic circuits," *IEEE Trans. on CAD*, vol. 22, no. 6, pp. 710–722, 2003.

[6] 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, pp. 703–715, 2009.

[7] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Design Automation Conf.*, 2003, pp. 318–323.

[8] P. Gupta, A. Agrawal, and N. K. Jha, "An algorithm for synthesis of reversible logic circuits," *IEEE Trans. on CAD*, vol. 25, no. 11, pp. 2317–2330, 2006.

[9] R. Wille and R. Drechsler, "BDD-based synthesis of reversible logic for large functions," in *Design Automation Conf.*, 2009, pp. 270–275.

[10] R. Wille, S. Offermann, and R. Drechsler, "SyReC: A programming language for synthesis of reversible circuits," in *Forum on Specification and Design Languages*, 2010, pp. 184–189.

[11] W. Hung, X. Song, G. Yang, J. Yang, and M. Perkowski, "Optimal synthesis of multiple output Boolean functions using a set of quantum gates by symbolic reachability analysis." *IEEE Trans. on CAD*, vol. 25, no. 9, pp. 1652–1663, 2006.

[12] D. Große, R. Wille, G. W. Dueck, and R. Drechsler, "Exact synthesis of elementary quantum gate circuits for reversible functions with don't cares," in *Int'l Symp. on Multi-Valued Logic*, 2008, pp. 214–219.

[13] R. Wille, H. M. Le, G. W. Dueck, and D. Große, "Quantified synthesis of reversible logic," in *Design, Automation and Test in Europe*, 2008, pp. 1015–1020.

[14] S. A. Cook, "The complexity of theorem proving procedures," in *Symposium on Theory of Computing*, 1971, pp. 151–158.

[15] N. Eén and N. Sörensson, "An extensible SAT solver," in *SAT 2003*, ser. Lecture Notes in Computer Science, vol. 2919, 2004, pp. 502–518.

[16] M. Soeken, S. Frehse, R. Wille, and R. Drechsler, "RevKit: An Open Source Toolkit for the Design of Reversible Circuits," in *Reversible Computation 2011*, ser. Lecture Notes in Computer Science, vol. 7165, 2012, pp. 64–76, RevKit is available at www.revkit.org.

[17] R. Wille, G. Fey, D. Große, S. Eggersglüß, and R. Drechsler, "SWORD: A SAT like prover using word level information," in *IFIP Int'l. Conf. on Very Large Scale Integration of System-on-Chip*, 2007, pp. 88–93.

[18] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler, "RevLib: an online resource for reversible functions and reversible circuits," in *Int'l Symp. on Multi-Valued Logic*, 2008, pp. 220–225, RevLib is available at http://www.revlib.org.

[19] A. Barenco, C. H. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *The American Physical Society*, vol. 52, pp. 3457–3467, 1995.

[20] Z. Sasanian and D. M. Miller, "Transforming MCT circuits to NCVW circuits," in *Reversible Computation 2011*, ser. Lecture Notes in Computer Science, vol. 7165, 2012, pp. 77–88.

[21] C. Moraga, "Hybrid GF(2) – Boolean expressions for quantum computing circuits," in *Reversible Computation 2011*, ser. Lecture Notes in Computer Science, vol. 7165, 2012, pp. 54–636.

[22] Z. Sasanian, R. Wille, and M. Miller, "Realizing reversible circuits using a new class of quantum gates," in *Design Automation Conference*, 2012.