Reversible Circuit Synthesis Using Binary Decision Diagrams

Krzysztof Podlaski
University of Lodz
Faculty of Physics and Applied Informatics
Email: podlaski@uni.lodz.pl

***Abstract*—Reversible circuit synthesis is an important branch of low power consumption circuit design. The idea of a logic circuit without loss of information during computation has an impact on power consumption and on the other hand makes the use of classical circuit synthesis algorithms not applicable. In the area of reversible circuit design there is still lack of satisfactory algorithms. During last 15 years many heuristic algorithms have been developed, however, they construct circuit implementations which are far from optimal. In the paper a new implementation of the known transformation based algorithm is presented. The existing transformation based algorithms use truth table during computation. This leads to important memory restrictions on the algorithm. On the other hand any Boolean function can be represented using Binary Decision Diagrams (BBD). This representation is more compact and uses less memory than truth table representation. Presented new implementation of transformation based algorithm can be used for synthesis of much larger reversible functions than for original version of the algorithm.**

***Index Terms*—reversible computing, reversible circuits design, binary decision diagrams, transformation based algorithm**

I. Introduction

Digital circuits with low energy consumption are one of the most important requirements for future electronic devices. Many researchers are working in this area. It is well known that reversible circuits lead to low energy consumption. It was proved by Landauer in [1], that any loss of information leads to energy dissipation. The reversible circuits do not loose any information, all information is transformed during operations but never lost. On the other hand relationship between reversible and quantum computation is another argument why this area of research is very important nowadays.

Many papers has been published on reversible circuit synthesis, unfortunately all of proposed algorithms are not satisfactory. Some of algorithms allow synthesising optimal circuits for reversible functions that operate on maximum 4 input bites, others can be used for larger functions but their results are very redundant [2]. One of the most recognized algorithm known as transformation based [3] has memory demands that rise rapidly with increase of circuit size. In literature we can find algorithms that use Binary Decision Diagrams to synthesize reversible functions. They allow to minimize memory requirements, but on the other hand resulting circuits are far from optimal ones [4], [5]. In this paper a new implementation of transformation based algorithm is presented. The original approach is based on truth table representation of Boolean function while the new one is based on Binary Decision Diagrams representation of reversible function. This new approach uses less memory and allows to solve larger function than original one. Moreover, proposed approach allows building circuit implementation of partially defined reversible functions.

The paper is organized as follows. Section II is an introduction to reversible circuit design. In the next section the transformation based algorithm is presented. In Section IV the new BBD version of transformation based algorithm is proposed. Section V contains experimental results and Section VI contains conclusions and final remarks.

II. Reversible Circuits Synthesis

The nature of reversible circuits makes impossible to simply translate synthesis methods from traditional circuit design to reversible one. In a reversible circuit fan-out of each gate output is always equal to 1, such a circuit is also built from a different set of gates. Many known methods of reversible circuits design have been developed [2]: cycle-based, transformation-based, exclusive or sum of products (ESOP) methods, binary decision diagrams (BDD) methods. The reversible synthesis task can be defined as the search for best circuit implementation of a given reversible function.

Definition 1 (Reversible function): A reversible $n*n$ function is a bijection mapping of n binary inputs onto n binary outputs.

Definition 2 (Sequence representation of a function): Any reversible $n*n$ function f can be represented as a sequence of output signals: $\{f(0), f(1), \ldots, f(2^n-2), f(2^n-1)\}$ and this representation is unique.

This implies that any reversible function can be described as a permutation operation. For example, the function $\{0,1,2,3,4,5,7,6\}$ has output values as follows: $f(0) = 0$, $f(1) = 1$, $\ldots$, $f(6) = 7$, $f(7) = 6$.

Definition 3 (Truth table representation of a function): Any reversible $n*n$ function f can be represented in truth table form where columns represent binary value of inputs or outputs and rows contain all possible configurations of input signals and resulting outputs. This representation is unique for a given order of input/output signals.

For example, the function $\{0,1,2,3,4,5,7,6\}$ can be represented by the truth table shown in Table I.

ISBN 978-83-63578-08-4

TABLE I
TRUTH TABLE REPRESENTATION OF FUNCTION $\{0,1,2,3,4,5,7,6\}$ WHERE i_1, i_2, i_3 DENOTE INPUT AND o_1, o_2, o_3 OUTPUT SIGNALS.

i_3	i_2	i_1	o_3	o_2	o_1
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Definition 4 (Distance measure for functions): Distance between two reversible $n*n$ functions f_1 and f_2, denoted as dist(f_1, f_2) is sum of Hamming distances between appropriate output signals in bit representation:

$$\text{dist}(f_1, f_2) = \sum_{i=0}^{2^n-1} \text{hamming}(f_1(i), f_2(i)). \quad (1)$$

Definition 5 (Reversible gate): An n-input n-output $(n*n)$ gate is reversible if it realizes a reversible $n*n$ function.

Definition 6 (Reversible circuit): A reversible $n*n$ circuit is a cascade of reversible gates with no more than n inputs that realizes a reversible $n*n$ function.

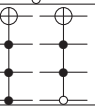
In order to find the best circuit representation of a given function first we have to define what the best means. In the area of reversible circuit synthesis cost measures are used, the lower the cost the better. There are two most acknowledged cost measures for reversible circuits: gate count and quantum cost. The first just counts how many gates were used in an implementation. The latter is based on implementations, of the function using basic quantum operations created for each of reversible gates [6]. Many researchers use NCT gate library that consists of the following gates:

- NOT(s) - an inverter, negates the signal on line s,
- CNOT($k; s$) - controlled NOT gate, negates the signal on line s if the signal on line k is equal to 1,
- TOFFOLI($k, l; s$) - Toffoli gate, negates the signal on line s if both signals on lines k and l are equal to 1,
- TOFFOLI($k_1, \cdots, k_m; s$) - generalized Toffoli gate, negates the signal on line s if all signals on lines $k_1, \cdots, k_m$ are equal to 1.

The presented library of gates uses a signal 1 as control signal. There is an extension of NCT library called mixed control Toffoli gates (MCT). This new library is built from the same types of gates as NCT but the control signals can be positive (1) or negative (0). In a circuit diagram the positive control is represented by solid dot (•) while the negative by a so called white dot (○). The quantum costs for the extended library of gates are presented in Table II.

There is an approach to reduce gate count or quantum cost introducing additional lines (ancilla lines). However, it is not obvious how to compare the quantum cost with hardware cost of additional line in a real reversible circuit implementation. In the presented paper no ancilla lines are used during synthesis.

TABLE II
REVERSIBLE GATES AND THEIR QUANTUM COST.

gate	symbol	no. control lines	Quantum cost
NOT		0	1
CNOT		1	1
CNOT		1	2
TOFFOLI		2	5
TOFFOLI		2	6
TOFFOLI		3	13

III. TRANSFORMATION BASED SYNTHESIS ALGORITHM

Transformation based algorithm uses truth table representation of reversible function. The simplest version of transformation algorithm was presented in [3] (see Algorithm 1).

Algorithm 1 Transformation based algorithm

Task: Find circuit representation of a given reversible function f.
Begin
if $f(0) \neq 0$ **then**
  Invert using NOT gate the outputs corresponding to 1-bits in $f(0)$.
end if
for all $i, 1 \leq i \leq 2^n - 1$ **do**
  Denote actual reversible specification as f^+.
  if $f^+(i) \neq i$ **then**
    Identify bits where $f^+(i)$ and i are different.
    Use $TOFFOLI$ gates that operate on identified bits.
    Insert gates to transform specification f^+ into f^{++} such that $f^{++}(i) = i$.
    Set f^{++} to be actual reversible specification.
  end if
end for
End
Algorithm result: Obtained actual reversible specification is a circuit representation of a given reversible function f.

Following [3] we can present an example of the Algorithm 1. Truth table for all steps of this algorithm is shown in Table III. We start with function $f = \{1, 0, 3, 2, 5, 7, 4, 6\}$. As in presented algorithm at first stage (i) we check value of $f(0)$. In this case it is $f(0) = 1$, thus we have to apply gate $NOT(o_3)$. After that our function have the form (ii). We can see that $f^+(i) = i$ for $0 \leq i \leq 4$, and $f^+(5) = 6$. In this case we have to apply $TOFFOLI(o_2, o_3; o_1)$. In next two steps we add gates $TOFFOLI(o_1, o_3; o_2)$ and $TOFFOLI(o_2, o_3; o_1)$ in order to obtain $f^+(6) = 6$ and $f^+(7) = 7$. We have to remember that each new gate is added on the left side of existing circuit. The resulting circuit is shown on Figure III.

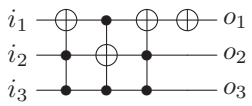


Fig. 1. Circuit representation for function in Table III

TABLE III
EXAMPLE OF TRANSFORMATION BASE ALGORITHM

	(i)	(ii)	(iii)	(iv)	(v)
000	001	00**0**	000	000	000
001	000	00**1**	001	001	001
010	011	01**0**	010	010	010
011	010	01**1**	011	011	011
100	101	10**0**	100	100	100
101	111	11**0**	11**1**	10**1**	101
110	100	10**1**	101	11**1**	11**0**
111	110	11**1**	10**0**	110	11**1**

The presented algorithm Algorithm 1 produces the circuit representation by introducing Toffoli gates manipulating only output signals. However, for each reversible function f there exists inverse function f^{-1}. We can always consider application of described transformation algorithm to obtain a circuit representation of inverse function f^{-1}. The best approach to do this is to apply presented method simultaneously in both directions, that means that at each step we choose to apply gates on input or output side.

The presented transformation based algorithm highly depends on order of input and output lines. If one wants to obtain best result we should check all possible ordering of lines. This will be impossible for high number of lines. The memory requirements grows rapidly with number of lines.

IV. BDD VERSION OF TRANSFORMATION BASED SYNTHESIS ALGORITHM

Every Boolean function can be represented in the form of a Binary Decision Diagram [7] that have been used for long time in synthesis of classical digital circuits. The BBD graph represents Shannon decomposition of a Boolean function:

$$f = \bar{x}_i \cdot f_{x_i=0} + x_i \cdot f_{x_i=1}. \tag{2}$$

Each node in BBD graph is labelled by a variable x_i used for decomposition. For example, if a node in a BDD representation of the function f is labelled with the variable x_i, its child nodes represent $f_{x_i=0}$ and $f_{x_i=1}$. The element $f_{x_i=0}(f_{x_i=1})$ is called the negative (positive) cofactor of f with respect to variable x_i (see Figure 2). Cofactors evaluating to the constant 0 or 1 are represented by terminal nodes.

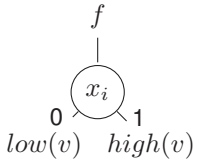


Fig. 2. Node in Binary Decision Diagram of function f connected to variable x_i

TABLE IV
TRUTH TABLE REPRESENTATION OF FUNCTION HAM3.

i_3	i_2	i_1	o_3	o_2	o_1
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	0	1	1

In the field of reversible computations BDD's have been introduced in 2001 in the paper [8]. Authors of [4] have used BBD in order to optimally represent Boolean function and then build resulting reversible circuit by mapping diagram nodes to a sequence of Toffoli gates. This approach can be used for very lerge reversible functions where number of input bits is much bigger than 10. On the other hand, circuits obtained via this approach are usually very redundant and use many so called garbage lines. Usually in most of papers authors focus on one of optimisation goals (gate count or quantum cost). Even though the relationship between presented cost functions and number of lines has not been decided [5], we should not allow the number of lines to increase too much. On that basis we can summarize that BDD synthesis using node mapping leads to unsatisfying results [9].

A. The transformation based algorithm

The BDD graph represents all information on described function and requires less memory than truth table (see Table IV and Figure 3). A path through BDD graph represents one row in the truth table. This allows realization of presented earlier transformation based algorithm in BDD representation. In traditional approach all gates were applied in order to transform the truth table of a given function into a truth table of identity function. In proposed BDD approach we will compare BDD of a function f and BDD of identity function, applying appropriate gates if these two differs. After introduction of a new gate we will obtain a new BDD diagram of a function f^+ inAlgorithm 2.

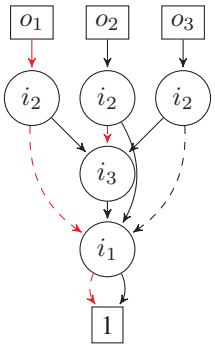


Fig. 3. BDD representation of function ham3 shown in Table IV. Dashed lines represent edge to negative cofactor. Red lines denote complementary edges, i.e. edges with additional negation of the signal.

Algorithm 2 BDD version of Transformation based algorithm

Task: Find circuit representation of a given reversible function f.
Begin
Create BDD diagram for a function f
if $f(0) \neq 0$ **then**
Invert using NOT gate the outputs corresponding to 1-bits in $f(0)$.
end if
for all paths in BDD **do**
Denote actual reversible specification as f^+ and create its BDD.
if $f^+(i) \neq i$ **then**
Identify edges/bits where BDD of $f^+(i)$ differs from BDD of identity function.
Use $TOFFOLI$ gates that operate on identified bits.
Insert gates to transform specification f^+ into f^{++} such that $f^{++}(i) = i$.
Set f^{++} to be actual reversible specification.
end if
end for
End
Algorithm result: Obtained actual reversible specification is a circuit representation of a given reversible function f.

V. Experimental Results

The proposed BBD transformation based algorithm was implemented in C++ using CUDD library for decision diagrams operations. The implementation was applied to selected benchmark functions from the well-known library of reversible benchmark functions [10]. Obtained results were compared to results obtained using transformation based algorithm implemented in RevKit [11].

We can observe that our results obtained are sometimes better than those obtained using original transformation based algorithm. The reason for such behaviour is the optimisation at the level of creation of BDD diagram. In our implementation the order of nodes in minimal BDD graph was used, while original implementations uses standard variable ordering.

VI. Conclusion

In the paper a BDD implementation of transformation based synthesis algorithm for reversible functions is presented. The created implementation gives similar result to basic implementation that uses truth table function representation. One of main advantages of BDD version is lower memory requirements, that is a result of more compact function representation. Moreover, using known optimisation methods designed for decision diagrams and implemented in CUDD we can easily create BDD diagram for a given function. This more compact diagram allows us to build circuit function representation that uses smaller number of gates. The presented algorithm will be used in future research as starting point to create new BDD based synthesis algorithms.

TABLE V
Experimental results of BDD Transformation based algorithm compared to original Transformation based implementation for selected benchmark functions

Benchmark	Lines	Transf. Based[11]	BDD Transf. based
4b_15g_1	4	27	24
4b_15g_2	4	24	24
4b_15g_3	4	32	26
4b_15g_4	4	28	28
4b_15g_5	4	28	28
oc5	4	16	16
oc6	4	18	15
oc7	4	25	20
oc8	4	21	21
nth_prime4	4	15	15
4_49	4	36	27
hwb4	4	19	19
4_49+hwb4	4	31	28
decode42	4	16	16
aj-e11	4	16	14
mod10_171	4	9	9
msaee	4	21	19
gyang	4	28	28
dmasl	4	15	15
App2.12	4	24	20
hwb4	4	19	19
hwb5	5	58	50
hwb6	6	164	148

Acknowledgment

Work leading to this paper was partially supported by the EU COST Action IC 1405 Reversible Computation Extending Horizons of Computing.

References

[1] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM Journal of Research and Development*, vol. 5, no. 3, pp. 183–191, 1961.

[2] M. Saeedi and I. L. Markov, "Synthesis and optimization of reversible circuits a survey," *ACM Computing Surveys*, vol. 45, no. 2, p. 21, 2013.

[3] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Proceedings of the 40th annual Design Automation Conference*. ACM, 2003, pp. 318–323.

[4] R. Wille and R. Drechsler, "Bdd-based synthesis of reversible logic," *Int. J. of App. Metaheuristic Comp.*, vol. 1, no. 4, pp. 25–41, 2010.

[5] R. Wille, M. Soeken, D. M. Miller, and R. Drechsler, "Trading off circuit lines and gate costs in the synthesis of reversible logic," *Integration*, vol. 47, no. 2, pp. 284–294, 2014.

[6] D. Maslov and G. W. Dueck, "Improved quantum cost for n-bit toffoli gates," *Electronics Letters*, vol. 39, no. 25, pp. 1790–1791, 2003.

[7] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Comput*, vol. C-35, no. 8, pp. 677–691, Aug 1986.

[8] M. Perkowski, L. Jozwiak, P. Kerntopf, A. Mishchenko, A. Al-Rabadi, A. Coppola, A. Buller, X. Song, S. Yanushkevich, V. P. Shmerko *et al.*, "A general decomposition for reversible logic," in *Proceedings of 5th International Workshop on Applications of Reed-Muller Expansion in Circuit Design, Starkville, MS, Aug. 10-11*, 2001, pp. 119–138.

[9] P. Kerntopf, M. Perkowski, and K. Podlaski, "Synthesis of reversible circuits: A view on the state-of-the-art," in *Proceedings of the 12th IEEE Conference on Nanotechnology (IEEE-NANO)*. IEEE, 2012, pp. 1–6.

[10] D. Maslov, "Reversible logic synthesis benchmarks page," *http://www.cs.uvic.ca/˜dmaslov*, 2005.

[11] M. Soeken, S. Frehse, R. Wille, and R. Drechsler, "Revkit: An open source toolkit for the design of reversible circuits," in *Reversible Computation - Third International Workshop, RC 2011, Gent, Belgium, July 4-5, 2011. Revised Papers*, 2011, pp. 64–76.