

COMPLEXITY REDUCTION OF TOFFOLI NETWORKS BASED ON FDD

Suzana Stojković¹, Milena Stanković¹, Claudio Moraga^{2,3}

¹Faculty of Electronic Engineering, University of Niš, Serbia

²Centre for Soft Computing, Mieres, Asturias, Spain

³TU Dortmund University, Dortmund, Germany

Abstract. *Synthesis of switching functions by Toffoli gates has become a very important research topic in the last years, since Toffoli gates are used in the synthesis of reversible circuits. Early methods based on the truth-table representation of Boolean functions are applicable to functions with a relatively small number of variables. Later on, methods for synthesis by Toffoli gates based on decision diagrams (BDDs, FDDs or OKFDDs) were introduced and applied to the synthesis of both reversible and irreversible functions. This paper presents a method for the reduction of the number of lines and gates in the Toffoli gate realization of Boolean functions based on their Functional Decision Diagram (FDD) representation. Experiments show that, when the proposed reduction is used, the realization of the given function based on FDD will, on the average, be smaller in terms of the number of lines and the number of gates than the realizations based on an OKFDD, an optimal BDD or based on a FDD by using previously defined algorithms.*

Key words: *Switching functions synthesis, Toffoli gates, Binary Decision Diagrams, Functional Decision Diagrams*

1. INTRODUCTION

Toffoli gates have been used in the realizations of systems of reversible functions. A reversible function is a bijective function, i.e., it is a system of n functions with n input variables where there is a mapping of each input into a unique output. It follows that a reversible function can be realized by a cascade of Toffoli gates. Reversible circuits have lower power dissipation (heat) than classical (irreversible) circuits. Therefore, they are receiving increasing attention. However, most of the current methods for reversible synthesis (see, for example, [1], [2], [3] and the references therein) are limited by the complexity of computations and are applicable to functions of a relatively small number of variables. In all these approaches, the number of lines in the circuit is equal to the number of input and output variables. In [4-12], the synthesis approaches that can cope

Received August 19, 2014; received in revised form January 19, 2015

Corresponding author: Suzana Stojković

Faculty of Electronic Engineering, University of Niš, A. Medvedeva 14, 18000 Niš, Serbia

(e-mail: suzana.stojkovic@elfak.ni.ac.rs)

with Boolean functions of a large number of variables were proposed. All these methods are based on different types of *Decision Diagrams*.

The methods in [4] and [5] use *Binary Decision Diagrams* (BDDs) to represent functions and derive reversible circuits by mapping the non-terminal nodes into Toffoli cascades. Papers [6], [7] and [8] discuss the methods for reversible synthesis based on *Functional Decision Diagrams* (FDDs) with positive Davio nodes. In [9], a method for synthesis based on the *Kronecker Functional Decision Diagrams* (KFDDs) was proposed. The advantage is obtained from the reduced number of nodes in KFDDs compared to BDDs and FDDs, at the cost of more complex determination of these decision diagrams. In all the afore mentioned methods, circuits with Toffoli gates are obtained hierarchically, within a time and a memory linearly proportional to the decision diagram size. In [10] and [11], a reversible synthesis based on *Lattice Decision Diagrams* can be found. In [12], a method for the iterative synthesis of reversible cascades by analyzing properties of the function to be realized in the Walsh-Hadamard domain is proposed. The computation and the representation of the Walsh-Hadamard spectrum and the analysis are performed over BDDs.

The methods based on decision diagrams are applicable to the realizations of both reversible and irreversible systems of switching functions by Toffoli gates. However, the main disadvantage of this approach is the large number of lines in the finally produced circuit, because in almost all cases, for the realization of a node in the decision diagram a new additional line is introduced.

The problem of reducing the reversible realization of a switching function is discussed in many papers. Some methods reduce the number of additional lines by increasing the number of gates (see for example the methods presented in [13]). Other methods reduce the cost of the circuits by adding some additional lines (see for example [14]). Papers [15] and [16] present the methods that reduce both parameters: the number of lines and the number of gates. The method from paper [15] is based on the BDD and OKFDD with negated edges. The method proposed in [16] is based on the BDD representation of the switching function.

In this paper, we propose a method for reducing both parameters of the circuit complexity: the number of lines and the number of gates based on the *Functional Decision Diagram* (FDD) representation of the switching function instead of BDDs and KFDDs discussed in [13-16]. The reason for the change of the underlying data structure (FDDs instead of BDDs and KFDDs) is that the implementation of a positive Davio or negative Davio node by Toffoli gates is simpler than the implementation of the Shannon node.

The paper is organized as follows. In Section 2, the basic definitions necessary for understanding the concepts presented in the next sections are given. Section 3 describes the standard and optimal ways for the implementation of the FDD nodes by Toffoli gates. Experimental results that confirm the proposed method's advantage are shown in Section 4. Section 5 summarizes the results of the proposed method.

2. BASIC DEFINITIONS

The algorithm for reversible syntheses that will be presented in this paper is based on the FDD representation of the switching functions. The results will be compared with realizations that are done based on the BDD or KFDD. Because of that, this section introduces definitions of these types of the decision diagrams.

In the realizations of the functions, the Toffoli gates with one or more input lines are used, so the Toffoli gates are defined as well.

2.1. BDD, FDD and KFDD

For the definition of decision diagrams, the concept of a decision tree should be introduced.

A *Binary Decision Tree* (BDT) representing a Boolean function f is the binary tree created by the recursive application of the Shannon decomposition rule:

$$f = \bar{x}_i \cdot f(x_i = 0) \oplus x_i \cdot f(x_i = 1). \tag{1}$$

Additionally, instead of the Shannon decomposition, the function can be decomposed by using the positive (2) or the negative (3) Davio decomposition:

$$f = f(x_i = 0) \oplus x_i \cdot (f(x_i = 0) \oplus f(x_i = 1)), \tag{2}$$

$$f = \bar{x}_i \cdot (f(x_i = 0) \oplus f(x_i = 1)) \oplus f(x_i = 1). \tag{3}$$

The resulting decision tree using the Davio decompositions is named the *Functional Decision Tree* (FDT).

From the previous definition, it follows that in the *Functional Decision Tree* (FDT) for the given function f , in each non-terminal node either the positive or negative Davio decomposition can be used. In practice, the same decomposition is usually applied to all the nodes at the same level in the tree. In that case, a polarity vector defines the types of decompositions that are used in the levels. For example, in the FDT of a function with three variables, the polarity vector [101] means that at levels 1 and 3, the positive Davio decomposition is used, and at level 2, the negative one.

Example 1. Figure 1 shows the BDT (a) and the FDT (b) of the function $f(x_1, x_2, x_3, x_4) = \bar{x}_1 \bar{x}_2 + x_1 x_2 + x_3 + \bar{x}_4$. In this FDT, the positive Davio decomposition is used at all levels.

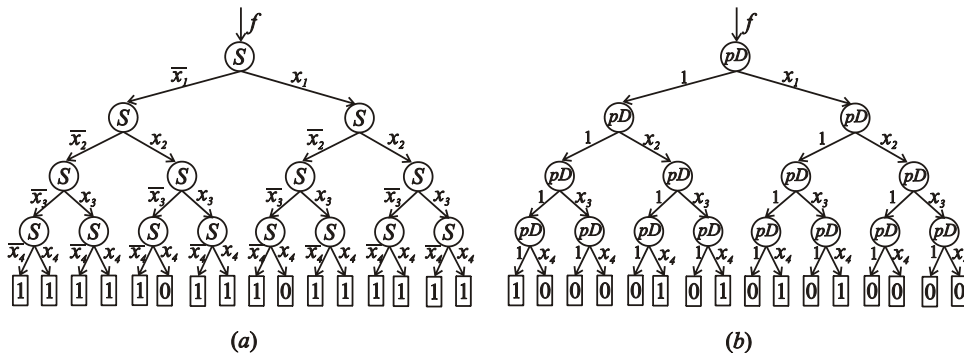


Fig. 1 BDT (a) and FDT (b) of the function f in Example 1.

A BDT is transformed into a BDD by using the following reduction rules:

1. Share all the isomorphic sub-trees: if there are two terminal nodes with the same value, or two non-terminal nodes with isomorphic sub-trees, one of them is deleted. The edges pointing to the deleted node are directed to the remaining node.
2. Eliminate all the redundant nodes: if both outgoing edges from a non-terminal node point to the same sub-tree, this node is redundant and it is deleted. Its incoming edges are directed to the common sub-tree.

An FDT is transformed into an FDD by using reduction rule 1 above and the following reduction rules for suppression of the zeros ([17], [18]):

- 2.1. If the right outgoing edge from a positive Davio node points to the 0-node (terminal node labeled by the value 0), the node is deleted. The edges pointing to the deleted node are directed to its left sub-tree.
- 2.2. If the left outgoing edge from a negative Davio node points to the 0-node, the non-terminal node is deleted. The edges pointing to the deleted node are directed to its right sub-tree.

Example 2. Figure 2 shows the BDD (a) and the FDD (b) that are obtained by reduction of the BDT and FDT from Example 1.

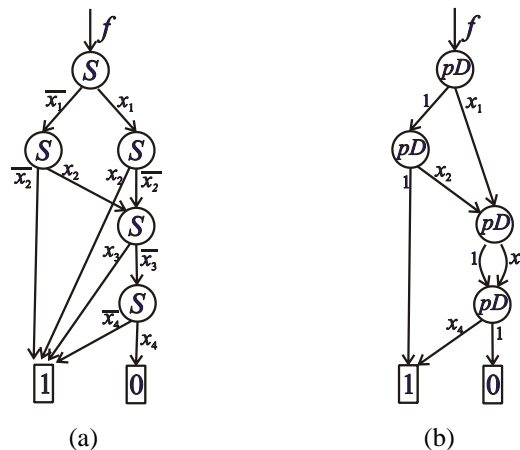


Fig. 2 BDD (a) and FDD (b) of the function from Example 1.

In the previous example, the FDD is more compact than the BDD for the considered function. But there are functions for which the BDD is more compact. There are no methods to predict whether the BDD or the FDD will be smaller for the given function (See the examples in Section 5).

The more general type of decision tree is the *Kronecker Functional Decision Tree* (KFDT) in which at each level either the Shannon, the positive Davio, or the negative Davio decomposition can be used. By the reduction of the KFDT, a *Kronecker Functional Decision Diagram* (KFDD) is obtained. Since the realization of the Shannon nodes by Toffoli gates is usually more complex than the corresponding realization of the positive or negative Davio nodes, the considerations in this paper are restricted to circuits derived from FDDs.

2.2. Toffoli Gates

Toffoli gates are used in the synthesis of reversible functions. Accordingly, Toffoli gates transform the set of input signals $(x, y_1, y_2, \dots, y_{n-1})$ into the set of output signals $(x \oplus y_1 y_2 \dots y_{n-1}, y_1, y_2, \dots, y_{n-1})$. A special case is the Toffoli gate with only one input and one output line that generates the complement of the input signal. Graphical symbols of the general Toffoli gate and (by extension) a Toffoli gate with one input and one output (that realizes the NOT operation) are shown in Figure 3.

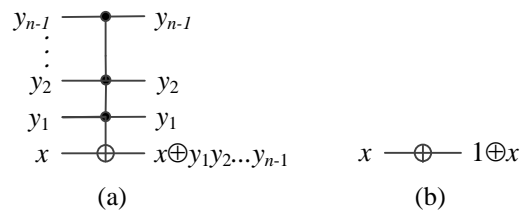


Fig. 3 Toffoli gate with n input and output lines (a) and with one input and output line (b).

3. OPTIMAL REALIZATION OF POSITIVE AND NEGATIVE DAVIO NODES BY TOFFOLI GATES

We use the same method for the reversible synthesis as in the approaches based on the BDD and KFDD, discussed in [4], [5], [9]. As proposed in [9], each node in the FDD maps into a Toffoli cascade as shown in Table 1. Realizations of Davio nodes that are proposed in the [9] we will note as “standard realizations” in the following text. It can be noticed that the nodes are always realized with additional lines. We observed that in some cases it is possible to have a simpler realization of the node, without additional lines, by the direct transformation of the inputs. However, if transformed, the initial input will be lost for future use. The additional lines are introduced to keep the input for future use. In the cases when the remainder function or an input variable is not used later, it is possible to select the implementation with a smaller number of lines and gates.

Table 2 shows the simplified realization of some Davio nodes which can be applied in the cases when it is not necessary to keep some inputs of the node for future use.

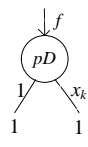
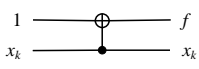
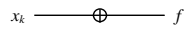
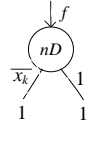
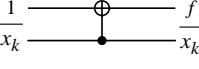
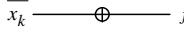
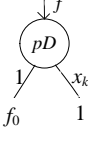
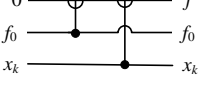
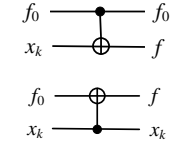
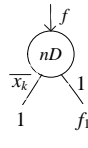
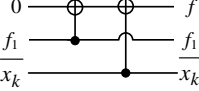
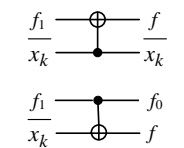

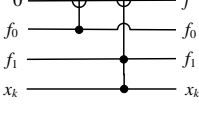
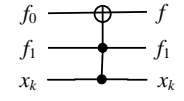
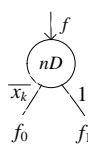
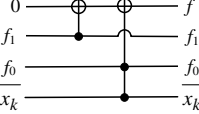
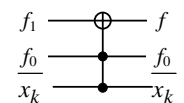
Table 2 shows possible reductions based on our approach:

1. For the node type marked as number 2, the output function f is equal to the \bar{x}_k which can be realized with one one-input Toffoli gate. However, in this case x_k will be lost for future use. If the x_k will not be used later, this node can be realized without additional lines.
2. The node type marked as number 3 realizes the function $f = f_0 + x_k$. There are two possible ways for the reduced realization of this node. In the first case the line x_k will be transformed into an output function, and in the second case, the line f_0 . If one of them will not be used later, the node can be realized without additional lines and with one gate (instead of 2 as it is proposed in the standard realization).

Table 1 Standard realizations of positive and negative Davio nodes by Toffoli gates

No.	Positive Davio (pD) nodes		Negative Davio (nD) nodes	
	Node type	Implementation by Toffoli gates	Node type	Implementation by Toffoli gates
1.				
2.				
3.				
4.				
5.				
6.				
7.				

Table 2 Reduced realizations of positive and negative Davio nodes by Toffoli gates

Node type	Standard implementation	Minimal implementation	Condition of minimization
<p><i>pD</i> No. 2:</p> 			if x_k is not used later
<p><i>nD</i> No. 2:</p> 			if $\overline{x_k}$ is not used later
<p><i>pD</i> No. 3:</p> 			if x_k is not used later
<p><i>nD</i> No. 3:</p> 			if f_1 is not used later
<p><i>pD</i> No. 6:</p> 			it is the last usage of the line f_0
<p><i>nD</i> No. 6:</p> 			it is the last usage of the f_1

- The node type marked as number 6 realizes the function $f = f_0 + x_k f_1$. The standard realization of this node can be reduced if the residual function f_0 is not used later. Reduced realization does not contain additional lines and contains one gate less than the standard realization.

Due to the previous observations summarized in Table 2, in the realization of the nodes satisfying the conditions mentioned above, the number of lines and/or number of gates can be reduced.

Based on this, we propose the following algorithm for the synthesis of not necessarily reversible functions by Toffoli gates.

We assume that the function to be realized is specified by an FDD with positive and negative Davio nodes. In each node the number of input edges must be saved. For each line in realization, the number of future uses is also saved. In the special hash table (*Realization Table*), the realized nodes and the appropriate output lines of their realizations are stored.

The algorithm to realize a multi-output function (given by the FDD) by Toffoli gates consists of the following steps.

- Step 1. (Initialization of the input lines)* For each input variable x_i ($i \in [1, n]$) do:
- Step 1. 1.* Create the input line x_i .
 - Step 1. 2.* Set the number of the future use of the line created in the previous step on the number of the nodes at the appropriate level in the FDD.
 - Step 1. 3.* If i -th level contains the negative Davio nodes then add a NOT Toffoli gate on the created line.
- Step 2. (Implementation of nodes)* Traverse the FDD in the depth-first manner and for each visited node q do:
- Step 2. 1.* If q exists in the Realization Table then get it and return the output line, else go to the next step.
 - Step 2. 2.* If q is a terminal node then return the corresponding constant line, else go to the next step.
 - Step 2. 3.* Determine the type of node q .
 - Step 2. 4.* Get input lines for the realization of node q (the input lines are the output lines of the nonterminal child nodes and the input line corresponding to the variable in node q).
 - Step 2. 5.* If there is a reduced realization for the given node type and the number of the future usages of the corresponding input line is equal to 1, then create the reduced implementation of node q ; else create the standard implementation of node q .
 - Step 2. 7.* Decrement future usage of each input line.
 - Step 2. 8.* Increase future usage of the output line by the number of the input edges of node q .
 - Step 2. 9.* Return the output line.

In comparison with the algorithms discussed in [12], [13], the advantages of the proposed algorithm can be summarized as

- The algorithm reduces the number of lines and the number of nodes at the same time,
- The reduced implementation is generated directly, while in ([12], [13]), the unreduced implementation is first generated and the reduction is performed in the post-processing step.

Example 3. Figure 4 shows the realizations of the function f in Example 1 by the use of the proposed algorithm. Figure 3 (b) corresponds to the realization based on the FDD with positive Davio nodes. For comparison, the BDD-based realization is shown in Figure (a).

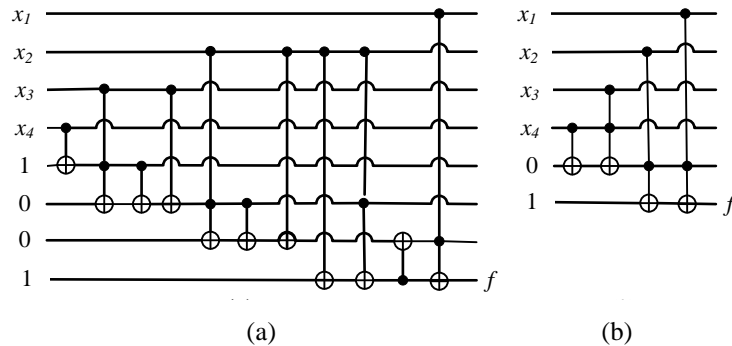


Fig. 4 Toffoli realization of the function f in Example 1 based on the BDD (a) and based on the FDD (b).

For this function, the size (number of non-terminal nodes) of the BDD is 5 and the size of the FDD is 4. The realization based on the BDD contains 11 gates and 4 additional lines, while the FDD-based realization contains 4 gates and 2 additional lines. Note that the achieved reduction of the network complexity is greater than the ratio of the sizes of these two DDs.

4. EXPERIMENTAL RESULTS

The proposed method for the FDD-based reversible synthesis was applied in the design of circuits realizing several MCNC benchmark irreversible functions and several reversible functions from the RevLib [19]. The results are shown in Tables 4 and 5, respectively.

In Table 4, the results of the synthesis of the MCNC function are compared with the results when synthesis is done based the optimal BDD and OKFDD presented in [7]. Recall that the optimal BDD assumes that optimization by variable reordering and complemented edges are performed. In Table 4, the number of lines is denoted as L , and the number of gates as G . The sizes of the corresponding DDs are denoted by S .

Notice that in Table 4, there are 5 cases where the realization based on FDDs requires more lines but less gates than realizations using other decision diagrams, but at the same time, are required. There are only two out of 16 benchmarks where the method fails to deliver an optimal solution.

Table 4 BDD, KFDD and FDD based realizations of MCNC functions

Function	In/Out	BDD			KFDD			FDD		
		S	L	G	S	L	G	S	L	G
9sym	9/1	25	27	62	26	29	60	27	12	26
con1	7/2	16	16	32	13	15	25	15	13	23
misex1	8/7	36	35	104	32	33	87	44	31	63
rd53	5/3	17	13	34	13	15	30	13	10	14
rd73	7/3	31	25	73	21	25	52	21	14	24
rd84	8/2	42	33	103	29	32	70	29	20	33
sqrt8	8/4	35	30	76	29	29	63	43	29	56
squar5	5/8	35	28	81	26	27	62	32	24	31
xor5	5/1	6	6	8	5	6	6	5	5	4
apex4	9/19	909	547	2551	915	552	2647	1106	574	1628
Bw	5/28	104	87	307	91	81	265	116	85	172
clip	9/5	87	66	228	85	66	185	152	82	217
ex1010	10/10	1080	670	2982	1062	658	2883	1494	761	2194
inc	7/9	73	53	187	70	56	196	99	66	142
5xp1	7/10	42	30	90	36	36	89	75	42	97
sao2	10/4	86	74	211	91	77	226	147	109	226

In Table 5, the results of the synthesis of the RevLib functions are compared with different previous algorithms for reversible synthesis based on DDs. The columns in the table contain the following results:

1. BDD - the results when the synthesis is done based on the optimal BDD presented in [5],
2. HR BDD and ER BDD - the results of the methods for reducing the number of lines in the realization based on the BDD that are presented in paper [13] (the HR BDD contains the results of a heuristic method for reducing the number of lines and the ER BDD contains the results of the exact method for line reduction),
3. PDD – the results when the synthesis is done based on the FDD with positive Davio nodes that is presented in papers [7] and [8].

In Table 5, it can be seen that out of 27 benchmark functions, 23 require fewer lines and fewer gates when using the proposed method than when using the BDD. In three cases some additional lines are required, but fewer gates are used. Only for the circuit alu_9 an additional line is required, keeping the same number of gates.

As it can be seen from the same table, both algorithms for postprocessing reduction presented in [12] create many additional gates. For 4 out of 10 functions, the realization by our algorithm requires fewer lines than realizations by a reduction in [12]. In 4 cases the same number of lines is generated and only in one case (for the circuit rd84) does our realization require more lines. But, the number of gates in our realization is on average 6.3 times smaller than the number of gates generated by a heuristic algorithm, and 1.84 times smaller than the number of gates that are generated by the exact algorithm.

Table 5 BDD and FDD based realizations of RevLib functions

Function	In / Out	BDD		HR BDD		ER BDD		PDD		FDD	
		L	G	L	G	L	G	L	G	L	G
3_17_6	3/3	7	17	-	-	-	-	6	6	7	10
4_49_7	4/4	15	42	-	-	-	-	6	11	13	25
4mod5_8	4/1	7	8	6	10	6	10	6	5	5	7
aj-e11_81	4/4	15	38	-	-	-	-	-	-	12	22
alu_9	5/1	7	9	-	-	-	-	-	-	8	9
decod24_10	2/4	6	11	-	-	-	-	-	-	7	7
decod24-enable_32	3/4	9	14	-	-	-	-	-	-	9	9
ex-1_82	3/3	5	7	-	-	-	-	-	-	6	5
fredkin_3	3/3	5	6	-	-	-	-	-	-	4	4
graycode6_11	6/6	11	15	-	-	-	-	-	-	6	5
ham3_28	3/3	7	14	-	-	-	-	-	-	6	8
ham7_29	7/7	21	61	-	-	-	-	16	46	17	35
hwb5_13	5/5	28	88	25	146	26	89	22	71	26	57
hwb6_14	6/6	46	159	41	485	41	172	-	-	41	105
hwb7_15	7/7	76	278	65	586	66	288	-	-	66	186
hwb8_64	8/8	114	440	-	-	-	-	71	287	101	300
miller_5	3/3	6	16	-	-	-	-	-	-	8	12
mini-alu_84	4/2	10	20	9	95	9	19	-	-	9	14
mod5d2_17	5/5	11	20	-	-	-	-	-	-	8	13
one-two-three_27	3/3	9	16	-	-	-	-	-	-	9	10
peres_4	3/3	5	7	-	-	-	-	-	-	4	3
rd32_19	3/2	6	10	-	-	-	-	-	-	4	4
rd53_68	5/3	13	34	12	50	12	35	11	27	10	14
rd73_69	7/3	25	73	-	-	-	-	12	65	14	24
rd84_70	8/4	34	104	24	424	25	105	30	86	28	41
sym6_63	6/1	14	29	11	177	11	28	14	22	10	17
sym9_71	9/1	21	62	22	362	24	61	21	52	12	26

In comparison with the algorithm based on the FDD with positive Davio nodes, our algorithm has better results for both criteria for 5 of 11 functions. In 3 cases our algorithm produces a smaller number of gates and greater number of lines, and in 3 cases (for functions 3_17_6, 4_49_7 and hwb8_64) our algorithm produces a greater number of gates and lines. The advantage of our algorithm is that we use the FDD containing both types of nodes (a positive Davio and negative Davio). Because of that our results are better when the optimal polarity is not [111...1].

We optimized the FDD only by finding optimal polarity (i.e., by finding the polarity that produces the minimal number of lines in the Toffoli realization). For future work, the method should be improved by reordering the variables in the FDD.

5. CONCLUSION

In this paper, we propose a method for the reduction of the number of lines and gates in the FDD-based synthesis of switching functions by Toffoli gates. The motivation for this work is in the relationship between the Davio decomposition rules, used in FDDs, and the output of the Toffoli gate. In general case, an FDD node can be realized by one Toffoli gate.

The proposed method was experimentally confirmed. Our experiments show that in such cases of functions whose FDD is smaller than the BDD (and for functions whose BDD is slightly larger than its FDD), the FDD-based realization by Toffoli gates requires a smaller number of lines and gates than the BDD-based realization. It can be seen in Table 4 that for most benchmark functions the FDD-based realization is more compact than the KFDD-based realization, although the KFDD should be equal or smaller than the FDD. This is due to the fact that the realizations of nodes with Shannon decomposition are more complicated than the proposed realizations of the FDD nodes.

REFERENCES

- [1] D. Maslov, D. W. Miller and G. V. Dueck, "Techniques for the synthesis of reversible Toffoli networks" *ACM Transaction on Design Automation of Electronic Systems*, Vol.12, No. 4, pp.42:1-42:20, 2007
- [2] J. Zhong and J. C. Muzio, "Improved Implementation of a Reed-Muller spectra based reversible synthesis algorithm" In: Proceedings of IEEE Pacific Rim Conference on Communication, Computers and Signal Processing, pp. 202-205, 2007
- [3] P. Gupta, A. Agrawal and N. K. Jha, "An algorithm for synthesis of reversible logic circuits", *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 25, No. 11, pp. 2317-2329, 2006
- [4] R. Wille and R. Drechsler R, "BDD-based synthesis of reversible logic for large functions", In: Proceedings of Design Automation Conference, San Francisco, 2009, pp. 270-275
- [5] R. Wille and R. Drechsler, "Effect of BDD optimization on synthesis of reversible and quantum logic", *Electronic Notes in Theoretical Computer Science*, vol. 253, no. 6, pp. 57-70, 2010
- [6] K. Takahashi, T. Hirayama, "Reversible Logic Synthesis from Positive Davio Trees of Logic Functions", In: Proceedings of IEEE TENCON Conference, Singapore, 2009, pp. 1-4
- [7] Y. Pang, J. Lin, S. Sultana, K. Radecka, "A Novel Method of Synthesizing Reversible Logic", In: Proceedings of IEEE Int. Symposium on Circuits and Systems, Rio de Janeiro, 2011, pp. 2857-2860
- [8] Y. Pang, S. Wang, Z. He, J. Lin, S. Sultana, K. Radecka, "Positive Davio Based Synthesis Algorithm for Reversible Logic", In: Proceeding of IEEE Int. Conference on Computer Design, Amherst, MA, 2011, pp. 212-218
- [9] M. Soeken, R. Wille and R. Drechsler, "Hierarchical synthesis of reversible circuits using positive and negative Davio decomposition", In: Proceedings of the Design and Test Workshop, Abu Dhabi, 2010, pp. 143-148
- [10] D. Shah, M. Perkowski, "Synthesis of Quantum Arrays with Low Quantum Cost from Kronecker Functional Lattice Diagrams", In: Proceedings of IEEE Congress on Evolutionary Computation, Barcelona, 2010, pp. 1-7
- [11] M. Perkowski, M. Lukac, D. Shah, M. Kameyama, "Synthesis of Quantum Circuits in Linear Nearest Neighbor Model Using Positive Davio Lattices", *Facta Universitatis (Niš)*, Series: Electronics and Energetics., vol. 24, no. 1, pp. 71-87, 2011
- [12] M. Stankovic and S. Stojkovic, "Reversible synthesis in the Walsh-Hadamard domain", In: Lecture Notes in Computer Science, vol. 6928, pp. 311-318, 2012
- [13] R. Wille, M. Soeken and R. Drechsler, "Reducing the Number of Lines in Reversible Circuits", In: Proceedings of the DAC 2010, Anaheim, California, 2010, pp. 647-652
- [14] D. M. Miller, R. Wille and R. Drechsler, "Reducing reversible circuits cost by adding lines", In: Proceedings of International Symposium on Multi-Valued Logic, Barcelona, Spain, 2010, pp. 217-222
- [15] E. Schoenborn, K. Datta, R. Wille, I. Sengupta, H. Rahaman, R. Drechsler, "Optimizing DD-based Synthesis of Reversible Circuits using Negative Control Lines", In: Proceedings of the Symposium on Design and Diagnostics of Electronic Circuits and Systems, Warsaw, Poland, 2014, pp. 129-134
- [16] M. Krishna, A. Chattopadhyay, "Efficient Reversible Logic Synthesis via Isomorphic Subgraph Matching", In: Proceedings of the IEEE International Symposium on Multiple-Valued Logic, Bremen, Germany, 2014, pp. 103-108
- [17] S. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems", In: Proceedings of the 30th International Conference on Design Automation, 1993, pp. 272-277
- [18] A. Mishchenko, "An Introduction to Zero-Suppressed Binary Decision Diagrams", Technical report 2001. (available at http://www.eecs.berkeley.edu/~alanmi/publications/2001/tech01_zdd.pdf)
- [19] An online Resource for Reversible Functions and Circuits: <http://revlib.org>